STATA POWER, PRECISION, AND SAMPLE-SIZE REFERENCE MANUAL
RELEASE 19

A Stata Press Publication
StataCorp LLC
College Station, Texas

Version 19

Published by Stata Press, 4905 Lakeway Drive, College Station, Texas 77845

ISBN-10: 1-59718-440-3
ISBN-13: 978-1-59718-440-3

The suggested citation for this software is

> StataCorp. 2025. *Stata 19*. Statistical software. StataCorp LLC.

The suggested citation for this manual is

> StataCorp. 2025. *Stata 19 Power, Precision, and Sample-Size Reference Manual*. College Station, TX: Stata Press.

Brief contents

Contents

Cross-referencing the documentation

When reading this manual, you will find references to other Stata manuals, for example, **[U] 27 Overview of Stata estimation commands**; [R] **regress**; and [D] **reshape**. The first example is a reference to chapter 27, *Overview of Stata estimation commands*, in the *User's Guide*; the second is a reference to the `regress` entry in the *Base Reference Manual*; and the third is a reference to the `reshape` entry in the *Data Management Reference Manual*.

All the manuals in the Stata Documentation have a shorthand notation:

[GSM]	*Getting Started with Stata for Mac*
[GSU]	*Getting Started with Stata for Unix*
[GSW]	*Getting Started with Stata for Windows*
[U]	*Stata User's Guide*
[R]	*Stata Base Reference Manual*
[ADAPT]	*Stata Adaptive Designs: Group Sequential Trials Reference Manual*
[BAYES]	*Stata Bayesian Analysis Reference Manual*
[BMA]	*Stata Bayesian Model Averaging Reference Manual*
[CAUSAL]	*Stata Causal Inference and Treatment-Effects Estimation Reference Manual*
[CM]	*Stata Choice Models Reference Manual*
[D]	*Stata Data Management Reference Manual*
[DSGE]	*Stata Dynamic Stochastic General Equilibrium Models Reference Manual*
[ERM]	*Stata Extended Regression Models Reference Manual*
[FMM]	*Stata Finite Mixture Models Reference Manual*
[FN]	*Stata Functions Reference Manual*
[G]	*Stata Graphics Reference Manual*
[H2OML]	*Machine Learning in Stata Using H2O: Ensemble Decision Trees Reference Manual*
[IRT]	*Stata Item Response Theory Reference Manual*
[LASSO]	*Stata Lasso Reference Manual*
[XT]	*Stata Longitudinal-Data/Panel-Data Reference Manual*
[META]	*Stata Meta-Analysis Reference Manual*
[ME]	*Stata Multilevel Mixed-Effects Reference Manual*
[MI]	*Stata Multiple-Imputation Reference Manual*
[MV]	*Stata Multivariate Statistics Reference Manual*
[PSS]	*Stata Power, Precision, and Sample-Size Reference Manual*
[P]	*Stata Programming Reference Manual*
[RPT]	*Stata Reporting Reference Manual*
[SP]	*Stata Spatial Autoregressive Models Reference Manual*
[SEM]	*Stata Structural Equation Modeling Reference Manual*
[SVY]	*Stata Survey Data Reference Manual*
[ST]	*Stata Survival Analysis Reference Manual*
[TABLES]	*Stata Customizable Tables and Collected Results Reference Manual*
[TS]	*Stata Time-Series Reference Manual*
[I]	*Stata Index*
[M]	*Mata Reference Manual*

[PSS-1] Introduction to power, precision, and sample-size analysis

Description

Sample-size determination is important for planning a study. It helps allocate necessary resources to the study. When a study uses hypothesis testing to make inference about parameters of interest, power and sample-size (PSS) analysis is used to investigate the optimal allocation of study resources to increase the likelihood of detecting the desired magnitude of the effect of interest. PSS analysis estimates the sample size required to achieve the desired power of a test in a future study. When a study uses confidence intervals (CIs) for inference, precision and sample-size (PrSS) analysis is used to estimate the required sample size to achieve the desired precision of a CI in a future study.

This manual describes the `power` command that provides PSS analysis for hypothesis testing (see [PSS-2] **power**) and the `ciwidth` command that provides PrSS analysis for CIs (see [PSS-3] **ciwidth**). Users can provide a list of parameters and perform sensitivity analysis. The results can be displayed in a table and in a graph; see [PSS-2] **power, table** and [PSS-2] **power, graph** for the `power` command and [PSS-3] **ciwidth, table** and [PSS-3] **ciwidth, graph** for the `ciwidth` command. You can also add your own methods to `power` ([PSS-2] ***power usermethod***) and `ciwidth` ([PSS-3] ***ciwidth usermethod***).

See [PSS-2] **Intro (power)** for a general introduction to PSS analysis and [PSS-3] **Intro (ciwidth)** for PrSS analysis.

Sample-size calculations for group sequential designs can be performed with the `gsdesign` command; see [ADAPT] **GSD intro** for a general introduction to group sequential designs, and see [ADAPT] **gs** for an introduction to the `gsdesign` command.

Also see

[PSS-2] **Intro (power)** — Introduction to power and sample-size analysis for hypothesis tests

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-3] **Intro (ciwidth)** — Introduction to precision and sample-size analysis for confidence intervals

[PSS-3] **ciwidth** — Precision and sample-size analysis for CIs

[PSS-5] **Glossary**

[PSS-2] Power and sample-size analysis

Intro (power) — Introduction to power and sample-size analysis for hypothesis tests

Description Remarks and examples References Also see

Description

Power and sample-size (PSS) analysis is essential for designing a statistical study that uses hypothesis testing for inference. It investigates the optimal allocation of study resources to increase the likelihood of the successful achievement of a study objective. PSS analysis provides an estimate of the sample size required to achieve the desired power of a test in a future study.

For precision and sample-size analysis for confidence intervals, see [PSS-3] **Intro (ciwidth)**.

For sample-size calculations for interim analyses in group sequential designs, see [ADAPT] **GSD intro**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes statistical methodology for PSS analysis and terminology that will be used throughout the manual. For a list of supported PSS methods and the description of the software, see [PSS-2] **power**. To see an example of PSS analysis in Stata, see *An example of PSS analysis in Stata*. For more information about PSS analysis, see Lachin (1981), Cohen (1988), Cohen (1992), Wickramaratne (1995), Lenth (2001), Chow et al. (2018), Julious (2010), and Ryan (2013), to name a few.

For precision and sample-size analysis for confidence intervals, see [PSS-3] **Intro (ciwidth)**.

For sample-size calculations for interim analyses in group sequential designs, see [ADAPT] **GSD intro**.

Power and sample-size analysis

Power and sample-size (PSS) analysis is a key component in designing a statistical study that uses hypothesis testing for inference. It investigates the optimal allocation of study resources to increase the likelihood of the successful achievement of a study objective.

How many subjects do we need in a study to achieve its research objectives? A study with too few subjects may have a low chance of detecting an important effect, and a study with too many subjects may offer very little gain and will thus waste time and resources. What are the chances of achieving the objectives of a study given available resources? Or what is the smallest effect that can be detected in a study given available resources? PSS analysis helps answer all of these questions. In what follows, when we refer to PSS analysis, we imply any of these goals.

We consider prospective PSS analysis (PSS analysis of a future study) as opposed to retrospective PSS analysis (analysis of a study that has already happened).

In the context of PSS analysis, hypothesis testing is the inferential method used to evaluate research objectives of a study. In this manual, we concentrate on the PSS analysis for hypothesis tests that include one-sample and two-sample tests of means, variances, proportions, correlations, and more. See [PSS-2] **power** for a full list of methods.

Before we discuss the components of PSS analysis, let us first revisit the basics of hypothesis testing.

Hypothesis testing

Recall that the goal of hypothesis testing is to evaluate the validity of a hypothesis, a statement about a population parameter of interest θ, a target parameter, based on a sample from the population. For simplicity, we consider a simple hypothesis test comparing a population parameter θ with 0. The two complementary hypotheses are considered: the null hypothesis H_0: $\theta = 0$, which typically corresponds to the case of "no effect", and the alternative hypothesis H_a: $\theta \neq 0$, which typically states that there is "an effect". An effect can be a decrease in blood pressure after taking a new drug, an increase in SAT scores after taking a class, an increase in crop yield after using a new fertilizer, a decrease in the proportion of defective items after the installation of new equipment, and so on.

The data are collected to obtain evidence against the postulated null hypothesis in favor of the alternative hypothesis, and hypothesis testing is used to evaluate the obtained data sample. The value of a test statistic (a function of the sample that does not depend on any unknown parameters) obtained from the collected sample is used to determine whether the null hypothesis can be rejected. If that value belongs to a rejection or critical region (a set of sample values for which the null hypothesis will be rejected) or, equivalently, falls above (or below) the critical values (the boundaries of the rejection region), then the null is rejected. If that value belongs to an acceptance region (the complement of the rejection region), then the null is not rejected. A critical region is determined by a hypothesis test.

A hypothesis test can make one of two types of errors: a type I error of incorrectly rejecting the null hypothesis and a type II error of incorrectly accepting the null hypothesis. The probability of a type I error is $\Pr(\text{reject } H_0 | H_0 \text{ is true})$, and the probability of a type II error is commonly denoted as $\beta = \Pr(\text{fail to reject } H_0 | H_0 \text{ is false})$.

A power function is a function of θ defined as the probability that the observed sample belongs to the rejection region of a test for a given parameter θ. A power function unifies the two error probabilities. A good test has a power function close to 0 when the population parameter belongs to the parameter's null space ($\theta = 0$ in our example) and close to 1 when the population parameter belongs to the alternative space ($\theta \neq 0$ in our example). In a search for a good test, it is impossible to minimize both error probabilities for a fixed sample size. Instead, the type-I-error probability is fixed at a small level, and the best test is chosen based on the smallest type-II-error probability.

An upper bound for a type-I-error probability is a significance level, commonly denoted as α, a value between 0 and 1 inclusively. Many tests achieve their significance level—that is, their type-I-error probability equals α, $\Pr(\text{reject } H_0|H_0 \text{ is true}) = \alpha$—for any parameter in the null space. For other tests, α is only an upper bound; see example 6 in [PSS-2] **power oneproportion** for an example of a test for which the nominal significance level is not achieved. In what follows, we will use the terms "significance level" and "type-I-error probability" interchangeably, making the distinction between them only when necessary.

Typically, researchers control the type I error by setting the significance level to a small value such as 0.01 or 0.05. This is done to ensure that the chances of making a more serious error are very small. With this in mind, the null hypothesis is usually formulated in a way to guard against what a researcher considers to be the most costly or undesirable outcome. For example, if we were to use hypothesis testing to determine whether a person is guilty of a crime, we would choose the null hypothesis to correspond to the person being not guilty to minimize the chances of sending an innocent person to prison.

The power of a test is the probability of correctly rejecting the null hypothesis when the null hypothesis is false. Power is inversely related to the probability of a type II error as $\pi = 1 - \beta = \Pr(\text{reject } H_0|H_0 \text{ is false})$. Minimizing the type-II-error probability is equivalent to maximizing power. The notion of power is more commonly used in PSS analysis than is the notion of a type-II-error probability. Typical values for power in PSS analysis are 0.8, 0.9, or higher depending on the study objective.

Hypothesis tests are subdivided into one sided and two sided. A one-sided or directional test asserts that the target parameter is large (an upper one-sided test $H\colon \theta > \theta_0$) or small ($H\colon \theta \leq \theta_0$), whereas a two-sided or nondirectional test asserts that the target parameter is either large or small ($H\colon \theta \neq \theta_0$). One-sided tests have higher power than two-sided tests. They should be used in place of a two-sided test only if the effect in the direction opposite to the tested direction is irrelevant; see *One-sided test versus two-sided test* below for details.

Another concept important for hypothesis testing is that of a p-value or observed level of significance. P-value is a probability of obtaining a test statistic as extreme or more extreme as the one observed in a sample assuming the null hypothesis is true. It can also be viewed as the smallest level of α that leads to the rejection of the null hypothesis. For example, if the p-value is less than 0.05, a test is considered to reject the null hypothesis at the 5% significance level.

For more information about hypothesis testing, see, for example, Casella and Berger (2002).

Next we review concepts specific to PSS analysis.

Components of PSS analysis

The general goal of PSS analysis is to help plan a study such that the chosen statistical method has high power to detect an effect of interest if the effect exists. For example, PSS analysis is commonly used to determine the size of the sample needed for the chosen statistical test to have adequate power to detect an effect of a specified magnitude at a prespecified significance level given fixed values of other study parameters. We will use the phrase "detect an effect" to generally mean that the collected data will support the alternative hypothesis. For example, detecting an effect may be detecting that the means of two groups differ, or that there is an association between the probability of a disease and an exposure factor, or that there is a nonzero correlation between two measurements.

The general goal of PSS analysis can be achieved in several ways. You can

- compute sample size directly given specified significance level, power, effect size, and other study parameters;
- evaluate the power of a study for a range of sample sizes or effect sizes for a given significance level and fixed values of other study parameters;
- evaluate the magnitudes of an effect that can be detected with reasonable power for specific sample sizes given a significance level and other study parameters;
- evaluate the sensitivity of the power or sample-size requirements to various study parameters.

The main components of PSS analysis are

- study design;
- statistical method;
- significance level, α;
- power, $1 - \beta$;
- a magnitude of an effect of interest or clinically meaningful difference, often expressed as an effect size, δ;
- sample size, N.

Below we describe each of the main components of PSS analysis in more detail.

Study design

A well-designed statistical study has a carefully chosen study design and a clearly specified research objective that can be formulated as a statistical hypothesis. A study can be observational, where subjects are followed in time, such as a cross-sectional study, or it can be experimental, where subjects are assigned a certain procedure or treatment, such as a randomized, controlled clinical trial. A study can involve one, two, or more samples. A study can be prospective, where the outcomes are observed given the exposures, such as a cohort study, or it can be retrospective, where the exposures are observed given the outcomes, such as a case–control study. A study can also use matching, where subjects are grouped based on selected characteristics such as age or race. A common example of matching is a paired study, consisting of pairs of observations that share selected characteristics.

Statistical method

A well-designed statistical study also has well-defined methods of analysis to be used to evaluate the objective of interest. For example, a comparison of two independent populations may involve an independent two-sample t test of means or a two-sample χ^2 test of variances, and so on. PSS computations are specific to the chosen statistical method and design. For example, the power of a balanced- or equal-allocation design is typically higher than the power of the corresponding unbalanced design.

Significance level

A significance level α is an upper bound for the probability of a type I error. With a slight abuse of terminology and notation, we will use the terms "significance level" and "type-I-error probability" interchangeably, and we will also use α to denote the probability of a type I error. When the two are different, such as for tests with discrete sampling distributions of test statistics, we will make a

distinction between them. In other words, unless stated otherwise, we will assume a size-α test, for which $\Pr(\text{reject}H_0|H_0 \text{ is true}) = \alpha$ for any θ in the null space, as opposed to a level-α test, for which $\Pr(\text{reject } H_0|H_0 \text{ is true}) \leq \alpha$ for any θ in the null space.

As we mentioned earlier, researchers typically set the significance level to a small value such as 0.01 or 0.05 to protect the null hypothesis, which usually represents a state for which an incorrect decision is more costly.

Power is an increasing function of the significance level.

Power

The power of a test is the probability of correctly rejecting the null hypothesis when the null hypothesis is false. That is, $\pi = 1 - \beta = \Pr(\text{reject } H_0|H_0 \text{ is false})$. Increasing the power of a test decreases the probability of a type II error, so a test with high power is preferred. Common choices for power are 90% and 80%, depending on the study objective.

We consider prospective power, which is the power of a future study.

Clinically meaningful difference and effect size

Clinically meaningful difference and effect size represent the magnitude of an effect of interest. In the context of PSS analysis, they represent the magnitude of the effect of interest to be detected by a test with a specified power. They can be viewed as a measure of how far the alternative hypothesis is from the null hypothesis. Their values typically represent the smallest effect that is of clinical significance or the hypothesized population effect size.

The interpretation of "clinically meaningful" is determined by the researcher and will usually vary from study to study. For example, in clinical trials, if no prior knowledge is available about the performance of the considered clinical procedure, then a standardized effect size (adjusted for standard deviation) between 0.25 and 0.5 may be considered clinically meaningful.

The definition of effect size is specific to the study design, analysis endpoint, and employed statistical model and test. For example, for a comparison of two independent proportions, an effect size may be defined as the difference between two proportions, the ratio of the two proportions, or the odds ratio. Effect sizes also vary in magnitude across studies: a treatment effect of 1% corresponding to an increase in mortality may be clinically meaningful, whereas a treatment effect of 10% corresponding to a decrease in a circumference of an ankle affected by edema may be of little importance. Effect size is usually defined in such a way that power is an increasing function of it (or its absolute value).

More generally, in PSS analysis, effect size summarizes the disparity between the alternative and null sampling distributions (sampling distributions under the alternative hypothesis and the null hypothesis, respectively) of a test statistic. The larger the overlap between the two distributions, the smaller the effect size and the more difficult it is to reject the null hypothesis, and thus there is less power to detect an effect.

For example, consider a z test for a comparison of a mean μ with 0 from a population with a known standard deviation σ. The null hypothesis is H_0: $\mu = 0$, and the alternative hypothesis is H_a: $\mu \neq 0$. The test statistic is a sample mean or sample average. It has a normal distribution with mean 0 and standard deviation σ as its null sampling distribution, and it has a normal distribution with mean μ different from 0 and standard deviation σ as its alternative sampling distribution. The overlap between these distributions is determined by the mean difference $\mu - 0 = \mu$ and standard deviation σ. The larger μ or, more precisely, the larger its absolute value, the larger the difference between the two populations, and thus the smaller

the overlap and the higher the power to detect the differences μ. The larger the standard deviation σ, the more overlap between the two distributions and the lower the power to detect the difference. Instead of being viewed as a function of μ and σ, power can be viewed as a function of their combination expressed as the standardized difference $\delta = (\mu - 0)/\sigma$. Then, the larger $|\delta|$, the larger the power; the smaller $|\delta|$, the smaller the power. The effect size is then the standardized difference δ.

To read more about effect sizes in Stata, see [R] **esize**, although PSS analysis may sometimes use different definitions of an effect size.

Sample size

Sample size is usually the main component of interest in PSS analysis. The sample size required to successfully achieve the objective of a study is determined given a specified significance level, power, effect size, and other study parameters. The larger the significance level, the smaller the sample size, with everything else being equal. The higher the power, the larger the sample size. The larger the effect size, the smaller the sample size.

When you compute sample size, the actual power (power corresponding to the obtained sample size) will most likely be different from the power you requested because sample size is an integer. In the computation, the resulting fractional sample size that corresponds to the requested power is usually rounded to the nearest integer. To be conservative, the sample size is rounded up to ensure that the actual power is at least as large as the requested power. For multiple-sample designs, fractional sample sizes may arise when you specify sample size to compute power or effect size. For example, to accommodate an odd total sample size of, say, 51 in a balanced two-sample design, each individual sample size must be 25.5. To be conservative, sample sizes are rounded down on input. The actual sample sizes in our example would be 25, 25, and 50. See *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for details about sample-size rounding.

For multiple samples, the allocation of subjects between groups also affects power. A balanced- or equal-allocation design—a design with equal numbers of subjects in each sample or group—generally has higher power than the corresponding unbalanced- or unequal-allocation design—a design with different numbers of subjects in each sample or group.

One-sided test versus two-sided test

Among other things that affect power is whether the employed test is directional (upper or lower one sided) or nondirectional (two sided). One-sided or one-tailed tests are more powerful than the corresponding two-sided or two-tailed tests. It may be tempting to choose a one-sided test over a two-sided test based on this fact. Despite having higher power, one-sided tests are generally not as common as two-sided tests. The direction of the effect, whether the effect is larger or smaller than a hypothesized value, is unknown in many applications, which requires the use of a two-sided test. The use of a one-sided test in applications in which the direction of the effect may be known is still controversial. The use of a one-sided test precludes the possibility of detecting an effect in the opposite direction, which may be undesirable in some studies. You should exercise caution when you decide to use a one-sided test because you will not be able to rule out the effect in the opposite direction if one were to happen. The results from a two-sided test have stronger justification.

Another consideration: Dropout

During the planning stage of a study, another important consideration is whether the data collection effort may result in missing observations. In clinical studies, the common term for this is dropout, when subjects fail to complete the study for reasons unrelated to study objectives.

If dropout is anticipated, its rate must be taken into consideration when determining the required sample size or computing other parameters. For example, if subjects are anticipated to drop out from a study with a rate of R_d, an ad hoc way to inflate the estimated sample size n is as follows: $n_d = n/(1 - R_d)$. Similarly, the input sample size must be adjusted as $n = n_d(1 - R_d)$, where n_d is the anticipated sample size.

Survival data

The prominent feature of survival data is that the outcome is the time from an origin to the occurrence of a given event (failure), often referred to as the analysis time. Analyses of such data use the information from all subjects in a study, both those who experience an event by the end of the study and those who do not. However, inference about the survival experience of subjects is based on the event times and therefore depends on the number of events observed in a study. Indeed, if none of the subjects fails in a study, then the survival rate cannot be estimated and survivor functions of subjects from different groups cannot be compared. Therefore, power depends on the number of events observed in a study and not directly on the number of subjects recruited to the study. As a result, to obtain the estimate of the required number of subjects, the probability that a subject experiences an event during the course of the study needs to be estimated in addition to the required number of events. This distinguishes sample-size determination for survival studies from that for other studies in which the endpoint is not measured as a time to failure.

All the above leads us to consider the following two types of survival studies. The first type (a type I study) is a study in which all subjects experience an event by the end of the study (no censoring), and the second type (a type II study) is a study that terminates after a fixed period regardless of whether all subjects experienced an event by that time. For a type II study, subjects who did not experience an event at the end of the study are known to be right-censored. For a type I study, when all subjects fail by the end of the study, the estimate of the probability of a failure in a study is one and the required number of subjects is equal to the required number of failures. For a type II study, the probability of a failure needs to be estimated and therefore various aspects that affect this probability (and usually do not come into play at the analysis stage) must be taken into account for the computation of the sample size.

Under the assumption of random censoring (Lachin 2011, 431; Lawless 2003, 52; Chow and Liu 2014, 391), the type of censoring pattern is irrelevant to the analysis of survival data in which the goal is to make inferences about the survival distribution of subjects. It becomes important, however, for sample-size determination because the probability that a subject experiences an event in a study depends on the censoring distribution. We consider the following two types of random censoring: administrative censoring and loss to follow-up.

Under administrative censoring, a subject is known to have experienced either of the two outcomes at the end of a study: survival or failure. The probability of a subject failing in a study depends on the duration of the study. Often in practice, subjects may withdraw from a study, say, because of severe side effects from a treatment or may be lost to follow-up because of moving to a different location. Here the information about the outcome that subject would have experienced at the end of the study had he completed the course of the study is unavailable, and the probability of experiencing an event by the end of the study is affected by the process governing withdrawal of subjects from the study. In the

literature, this type of censoring is often referred to as subject loss to follow-up, subject withdrawal, or sometimes subject dropout (Freedman 1982, Machin and Campbell 2005). Generally, great care must be taken when using this terminology because it may have slightly different meanings in different contexts. `power logrank` and `power cox` apply a conservative adjustment to the estimate of the sample size for withdrawal. `power exponential` assumes that losses to follow-up are exponentially distributed.

Another important component of sample-size and power determination that affects the estimate of the probability of a failure is the pattern of accrual of subjects into the study. The duration of a study is often divided into two phases: an accrual phase, during which subjects are recruited to the study, and a follow-up phase, during which subjects are followed up until the end of the study and no new subjects enter the study. For a fixed-duration study, fast accrual increases the average analysis time (average follow-up time) and increases the chance of a subject failing in a study, whereas slow accrual decreases the average analysis time and consequently decreases this probability. `power logrank` and `power exponential` provide facilities to account for uniform accrual, and for `power exponential` only, truncated exponential accrual.

All sample-size formulas used by `power`'s survival methods rely on the proportional-hazards assumption, that is, the assumption that the hazard ratio does not depend on time. See the documentation entry of each subcommand for the additional assumptions imposed by the methods it uses. In the case when the proportional-hazards assumption is suspect, or in the presence of other complexities associated with the nature of the trial (for example, lagged effect of a treatment, more than two treatment groups, clustered data) and with the behavior of participants (for example, noncompliance of subjects with the assigned treatment, competing risks), one may consider obtaining required sample size or power by simulation. Feiveson (2002) demonstrates an example of such simulation for clustered survival data. Also see Royston (2012) and Crowther and Lambert (2012) for ways of simulating complicated survival data. Barthel et al. (2006); Barthel, Royston, and Babiker (2005); Royston and Babiker (2002); Barthel, Royston, and Parmar (2009); and Royston and Barthel (2010) present sample-size and power computation for multiarm trials under more flexible design conditions.

Sensitivity analysis

Because of limited resources, it may not always be feasible to conduct a study under the original ideal specification. In this case, you may vary study parameters to find an appropriate balance between the desired detectable effect, sample size, available resources, and an objective of the study. For example, a researcher may decide to increase the detectable effect size to decrease the required sample size, or, rarely, to lower the desired power of the test. In some situations, it may not be possible to reduce the required sample size, in which case more resources must be acquired before the study can be conducted.

Power is a complicated function of all the components we described in the previous section—none of the components can be viewed in isolation. For this reason, it is important to perform sensitivity analysis, which investigates power for various specifications of study parameters, and refine the sample-size requirements based on the findings prior to conducting a study. Tables of power values (see [PSS-2] **power, table**) and graphs of power curves (see [PSS-2] **power, graph**) may be useful for this purpose.

An example of PSS analysis in Stata

Consider a study of math scores from the SAT exam. Investigators would like to test whether a new coaching program increases the average SAT math score by 20 points compared with the national average in a given year of 514. They do not anticipate the standard deviation of the scores to be larger than the

national value of 117. Investigators are planning to test the differences between scores by using a one-sample t test. Prior to conducting the study, investigators would like to estimate the sample size required to detect the anticipated difference by using a 5%-level two-sided test with 90% power. We can use the `power onemean` command to estimate the sample size for this study; see [PSS-2] **power onemean** for more examples.

Below we demonstrate PSS analysis of this example interactively, by typing the commands; see [PSS-2] **GUI (power)** for point-and-click analysis of this example.

We specify the reference or null mean value of 514 and the comparison or alternative value of 534 as command arguments following the command name. The values of standard deviation and power are specified in the corresponding `sd()` and `power()` options. `power onemean` assumes a 5%-level two-sided test, so we do not need to specify any additional options.

```
. power onemean 514 534, sd(117) power(0.9)

Performing iteration ...

Estimated sample size for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =    0.1709
           m0 =  514.0000
           ma =  534.0000
           sd =  117.0000

Estimated sample size:

            N =       362
```

The estimated required sample size is 362.

Investigators do not have enough resources to enroll that many subjects. They would like to estimate the power corresponding to a smaller sample of 300 subjects. To compute power, we replace the `power(0.9)` option with the `n(300)` option in the above command.

```
. power onemean 514 534, sd(117) n(300)

Estimated power for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0

Study parameters:

        alpha =    0.0500
            N =       300
        delta =    0.1709
           m0 =  514.0000
           ma =  534.0000
           sd =  117.0000

Estimated power:

        power =    0.8392
```

For a smaller sample of 300 subjects, the power decreases to 84%.

Investigators would also like to estimate the minimum detectable difference between the scores given a sample of 300 subjects and a power of 90%. To compute the standardized difference between the scores, or effect size, we specify both the power in the `power()` option and the sample size in the `n()` option.

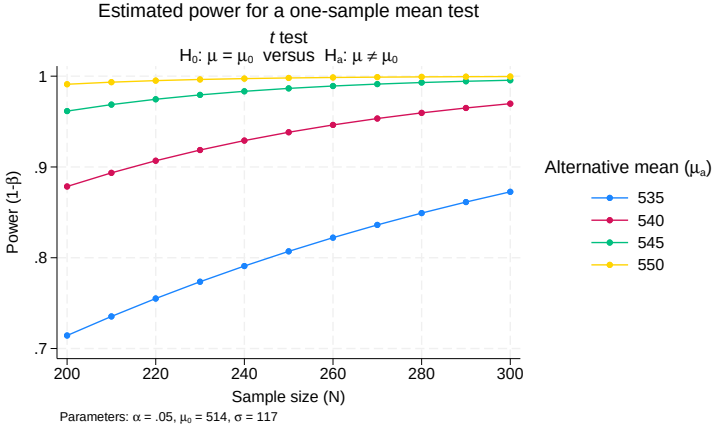
```
. power onemean 514, sd(117) power(0.9) n(300)
Performing iteration ...
Estimated target mean for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0; ma > m0
Study parameters:
        alpha =    0.0500
        power =    0.9000
            N =       300
           m0 =   514.0000
           sd =   117.0000
Estimated effect size and target mean:
        delta =    0.1878
           ma =   535.9671
```

The minimum detectable standardized difference given the requested power and sample size is 0.19, which corresponds to an average math score of roughly 536 and a difference between the scores of 22.

Continuing their analysis, investigators want to assess the impact of different sample sizes and score differences on power. They wish to estimate power for a range of alternative mean scores between 530 and 550 with an increment of 5 and a range of sample sizes between 200 and 300 with an increment of 10. They would like to see results on a graph.

We specify the range of alternative means as *numlist* (see **[U] 11.1.8 numlist**) in parentheses as the second command argument. We specify the range of sample sizes as a *numlist* in the `n()` option. We request a graph by specifying the `graph` option.

```
. power onemean 514 (535(5)550), sd(117) n(200(10)300) graph
```



The default graph plots the estimated power on the y axis and the requested sample size on the x axis. A separate curve is plotted for each of the specified alternative means. Power increases as the sample size increases or as the alternative mean increases. For example, for a sample of 220 subjects and an alternative mean of 535, the power is approximately 75%; and for an alternative mean of 550, the power is nearly 1. For a sample of 300 and an alternative mean of 535, the power increases to 87%. Investigators may now determine a combination of an alternative mean and a sample size that would satisfy their study objective and available resources.

If desired, we can also display the estimated power values in a table by additionally specifying the `table` option:

```
. power onemean 514 (530(5)550), sd(117) n(200(10)300) graph table
(output omitted)
```

The `power` command performs PSS analysis for a number of hypothesis tests for continuous, binary, and survival outcomes; see [PSS-2] **power** and method-specific entries for more examples. Also, in the absence of readily available PSS methods, consider performing PSS analysis by simulation; see, for example, Huber (2019a), Feiveson (2002), and Hooper (2013) for examples of how you can do this in Stata. You can also add your own methods to the `power` command as described in [PSS-2] ***power usermethod***; also see Huber (2019b).

Video example

A conceptual introduction to power and sample-size calculations

References

Barthel, F. M.-S., A. G. Babiker, P. Royston, and M. K. B. Parmar. 2006. Evaluation of sample size and power for multi-arm survival trials allowing for non-uniform accrual, non-proportional hazards, loss to follow-up and cross-over. *Statistics in Medicine* 25: 2521–2542. https://doi.org/10.1002/sim.2517.

Barthel, F. M.-S., P. Royston, and A. G. Babiker. 2005. A menu-driven facility for complex sample size calculation in randomized controlled trials with a survival or a binary outcome: Update. *Stata Journal* 5: 123–129.

Barthel, F. M.-S., P. Royston, and M. K. B. Parmar. 2009. A menu-driven facility for sample-size calculation in novel multiarm, multistage randomized controlled trials with a time-to-event outcome. *Stata Journal* 9: 505–523.

Blenkinsop, A., and B. Choodari-Oskooei. 2019. Multiarm, multistage randomized controlled trials with stopping boundaries for efficacy and lack of benefit: An update to nstage. *Stata Journal* 19: 782–802.

Casella, G., and R. L. Berger. 2002. *Statistical Inference*. 2nd ed. Pacific Grove, CA: Duxbury.

Chow, S.-C., and J.-P. Liu. 2014. *Design and Analysis of Clinical Trials: Concepts and Methodologies*. 3rd ed. Hoboken, NJ: Wiley.

Chow, S.-C., J. Shao, H. Wang, and Y. Lokhnygina. 2018. *Sample Size Calculations in Clinical Research*. 3rd ed. Boca Raton, FL: CRC Press.

Cohen, J. 1988. *Statistical Power Analysis for the Behavioral Sciences*. 2nd ed. Hillsdale, NJ: Erlbaum.

———. 1992. A power primer. *Psychological Bulletin* 112: 155–159. https://doi.org/10.1037//0033-2909.112.1.155.

Crowther, M. J., and P. C. Lambert. 2012. Simulating complex survival data. *Stata Journal* 12: 674–687.

Feiveson, A. H. 2002. Power by simulation. *Stata Journal* 2: 107–124.

Freedman, L. S. 1982. Tables of the number of patients required in clinical trials using the logrank test. *Statistics in Medicine* 1: 121–129. https://doi.org/10.1002/sim.4780010204.

Harrison, D. A., and A. R. Brady. 2004. Sample size and power calculations using the noncentral t-distribution. *Stata Journal* 4: 142–153.

Hemming, K., and J. Marsh. 2013. A menu-driven facility for sample-size calculations in cluster randomized controlled trials. *Stata Journal* 13: 114–135.

Hooper, R. 2013. Versatile sample-size calculation using simulation. *Stata Journal* 13: 21–38.

Huber, C. 2019a. Calculating power using Monte Carlo simulations, part 1: The basics. *The Stata Blog: Not Elsewhere Classified*. https://blog.stata.com/2019/01/10/calculating-power-using-monte-carlo-simulations-part-1-the-basics/.

———. 2019b. Calculating power using Monte Carlo simulations, part 2: Running your simulation using power. *The Stata Blog: Not Elsewhere Classified*. https://blog.stata.com/2019/01/29/calculating-power-using-monte-carlo-simulations-part-2-running-your-simulation-using-power/.

Julious, S. A. 2010. *Sample Sizes for Clinical Trials*. Boca Raton, FL: Chapman and Hall/CRC. https://doi.org/10.1201/9781584887409.

Kunz, C. U., and M. Kieser. 2011. Simon's minimax and optimal and Jung's admissible two-stage designs with or without curtailment. *Stata Journal* 11: 240–254.

Lachin, J. M. 1981. Introduction to sample size determination and power analysis for clinical trials. *Controlled Clinical Trials* 2: 93–113. https://doi.org/10.1016/0197-2456(81)90001-5.

———. 2011. *Biostatistical Methods: The Assessment of Relative Risks*. 2nd ed. Hoboken, NJ: Wiley.

Lawless, J. F. 2003. *Statistical Models and Methods for Lifetime Data*. 2nd ed. New York: Wiley.

Lenth, R. V. 2001. Some practical guidelines for effective sample size determination. *American Statistician* 55: 187–193. https://doi.org/10.1198/000313001317098149.

Machin, D., and M. J. Campbell. 2005. *Design of Studies for Medical Research*. Chichester, UK: Wiley.

Marley-Zagar, E., I. R. White, P. Royston, F. M.-S. Barthel, and A. G. Babiker. 2023. artbin: Extended sample size for randomized trials with binary outcomes. *Stata Journal* 23: 24–52.

Miller, D. J., J. T. Nguyen, and M. Bottai. 2020. emagnification: A tool for estimating effect-size magnification and performing design calculations in epidemiological studies. *Stata Journal* 20: 548–564.

Nash, S., K. E. Morgan, C. Frost, and A. Mulick. 2021. Power and sample-size calculations for trials that compare slopes over time: Introducing the slopepower command. *Stata Journal* 21: 575–601.

Newson, R. B. 2004. Generalized power calculations for generalized linear models and more. *Stata Journal* 4: 379–401.

Royston, P. 2012. Tools to simulate realistic censored survival-time distributions. *Stata Journal* 12: 639–654.

———. 2018. Power and sample-size analysis for the Royston–Parmar combined test in clinical trials with a time-to-event outcome. *Stata Journal* 18: 3–21.

Royston, P., and A. G. Babiker. 2002. A menu-driven facility for complex sample size calculation in randomized controlled trials with a survival or a binary outcome. *Stata Journal* 2: 151–163.

Royston, P., and F. M.-S. Barthel. 2010. Projection of power and events in clinical trials with a time-to-event outcome. *Stata Journal* 10: 386–394.

Ryan, T. P. 2013. *Sample Size Determination and Power*. Hoboken, NJ: Wiley. https://doi.org/10.1002/9781118439241.

Saunders, C. L., D. T. Bishop, and J. H. Barrett. 2003. Sample size calculations for main effects and interactions in case–control studies using Stata's nchi2 and npnchi2 functions. *Stata Journal* 3: 47–56.

Thompson, J. A., B. Leurent, S. Nash, L. H. Moulton, and R. J. Hayes. 2023. Cluster randomized controlled trial analysis at the cluster level: The clan command. *Stata Journal* 23: 754–773.

White, I. R., E. Marley-Zagar, T. P. Morris, M. K. B. Parmar, P. Royston, and A. G. Babiker. 2023. artcat: Sample-size calculation for an ordered categorical outcome. *Stata Journal* 23: 3–23.

Wickramaratne, P. J. 1995. Sample size determination in epidemiologic studies. *Statistical Methods in Medical Research* 4: 311–337. https://doi.org/10.1177/096228029500400404.

Also see

[PSS-2] **GUI (power)** — Graphical user interface for power and sample-size analysis

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-2] **power, graph** — Graph results from the power command

[PSS-3] **Intro (ciwidth)** — Introduction to precision and sample-size analysis for confidence intervals

[PSS-4] **Unbalanced designs** — Specifications for unbalanced designs

[PSS-5] **Glossary**

GUI (power) — Graphical user interface for power and sample-size analysis

Description Menu Remarks and examples Also see

Description

This entry describes the graphical user interface (GUI) for the `power` command. See [PSS-2] **power** for a general introduction to the `power` command.

Menu

Statistics > Power, precision, and sample size

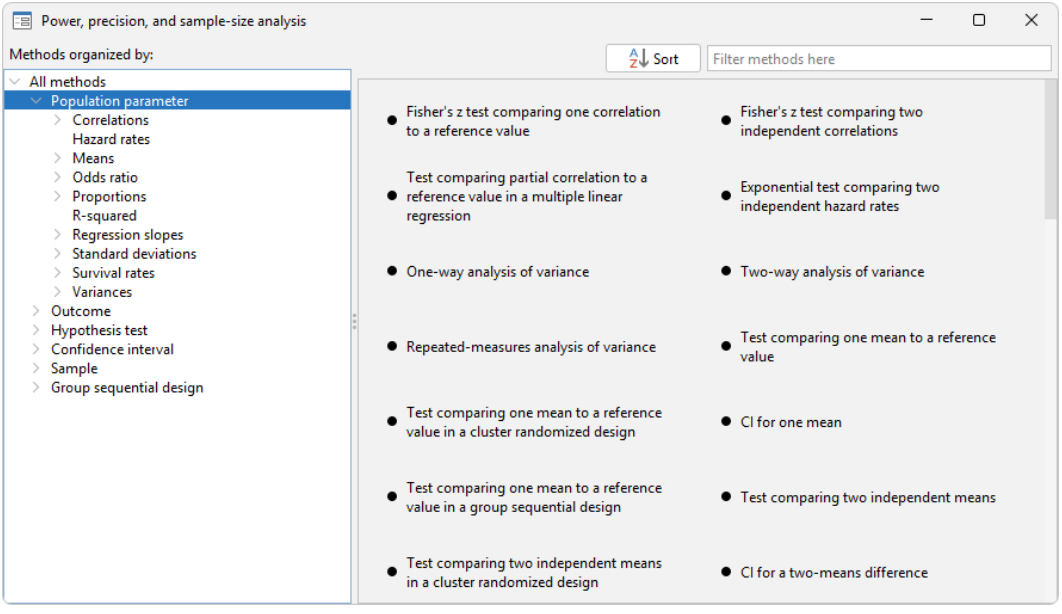
Remarks and examples

Remarks are presented under the following headings:

PSS Control Panel

You can perform PSS analysis interactively by typing the `power` command or by using a point-and-click GUI available via the PSS Control Panel.

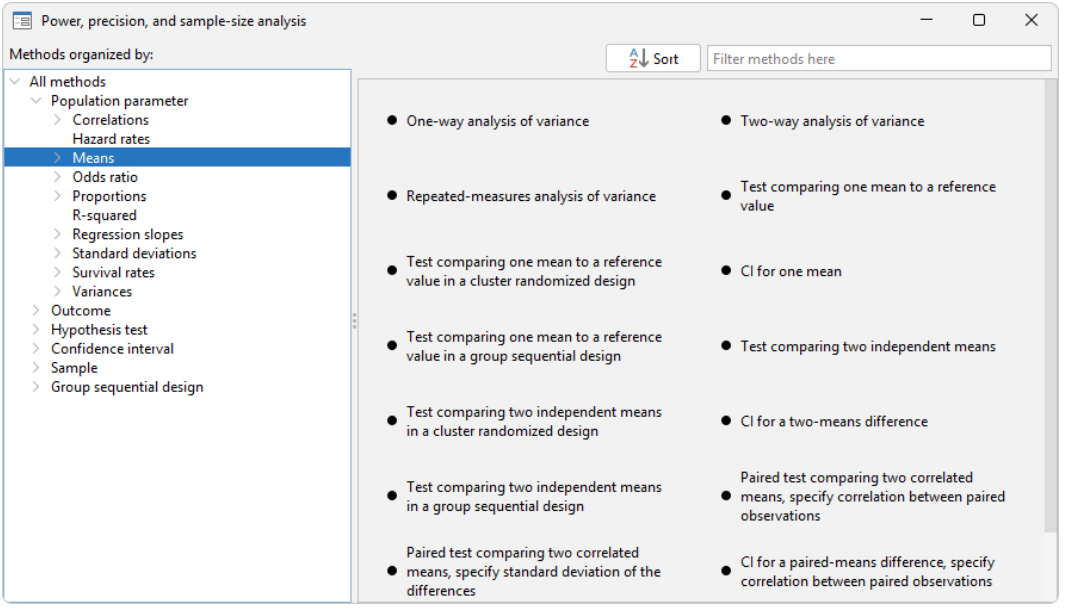
The PSS Control Panel can be accessed by selecting **Statistics** > **Power, precision, and sample size** from the Stata menu. It includes a tree-view organization of the PSS, PrSS, and group sequential design methods.



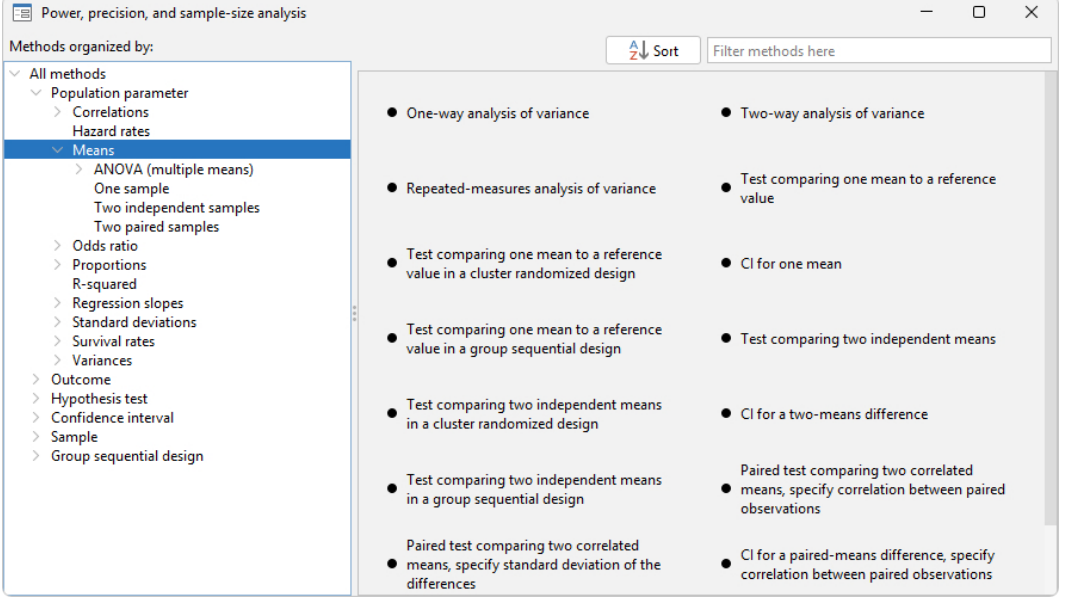
The left pane organizes the methods, and the right pane displays the methods corresponding to the selection in the left pane. On the left, the methods are organized by the type of population parameter, such as mean or proportion; the type of outcome, such as continuous or binary; the type of analysis, such as hypothesis test or confidence interval; and the type of sample, such as one sample or two samples. You click on one of the methods shown in the right pane to launch the dialog box for that method.

By default, methods are organized by **Population parameter**. We can find the method we want to use by looking for it in the right pane, or we can narrow down the type of method we are looking for by selecting one of the expanded categories in the left pane.

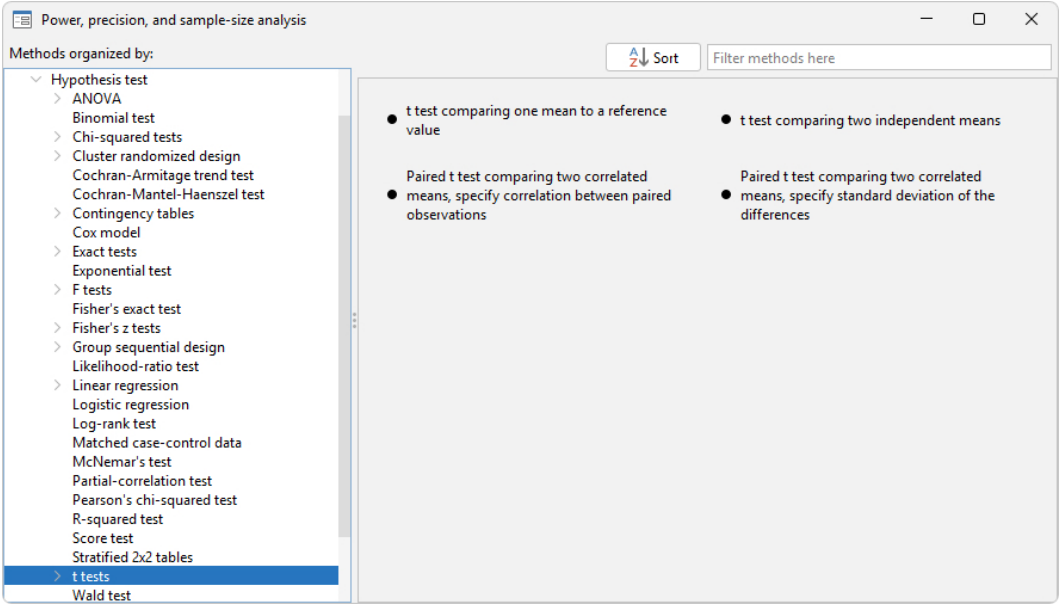
For example, if we are interested in means, we can click on **Means** within **Population parameter** to see all methods for means in the right pane.



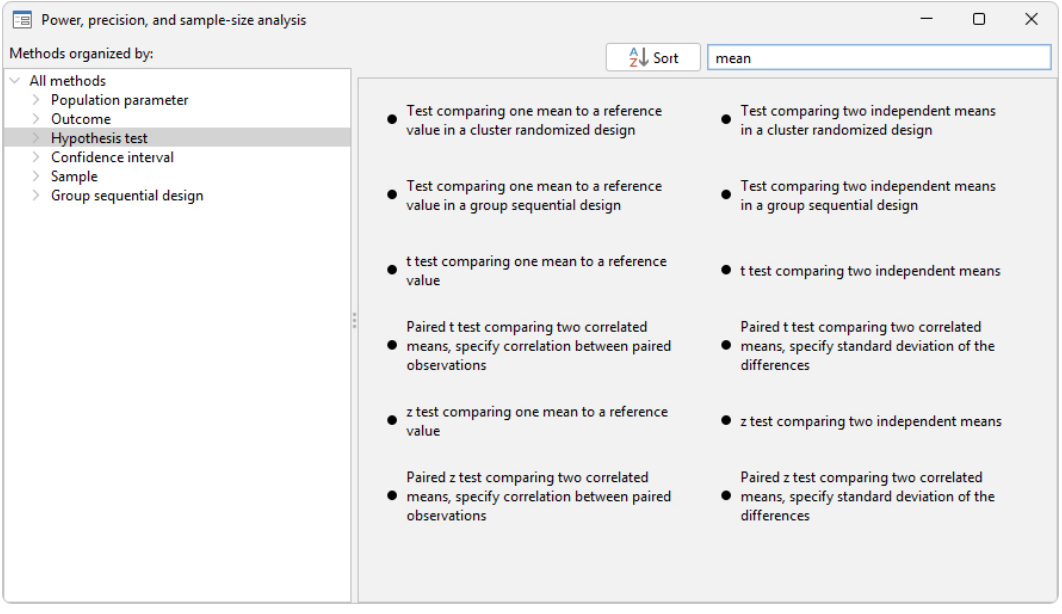
We can expand **Means** to further narrow down the choices by clicking on the symbol to the left of **Means**.



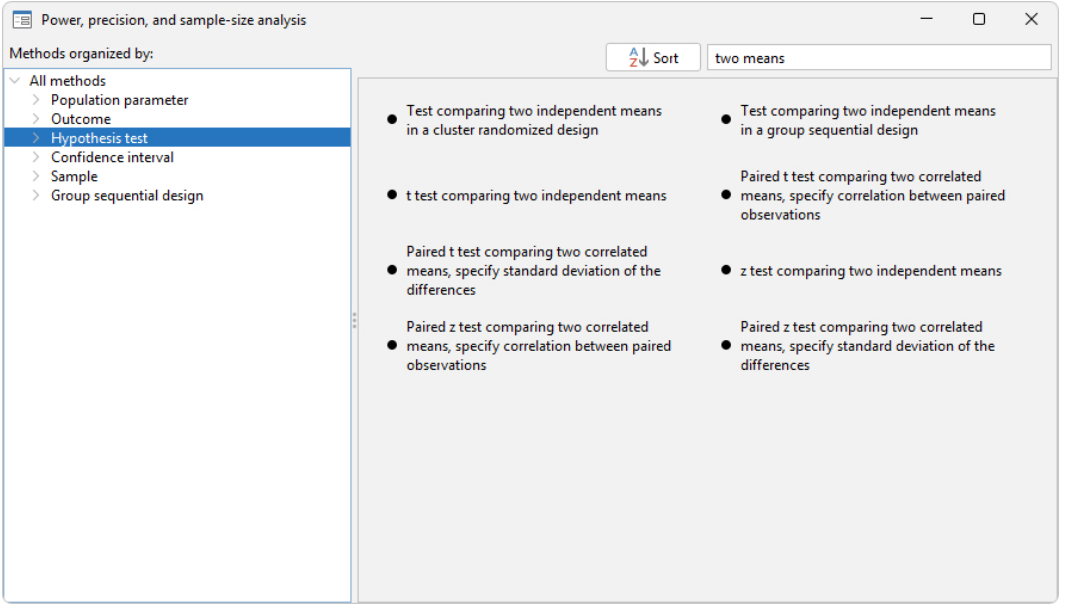
Or we can choose a method by the type of analysis by expanding **Hypothesis test** and selecting, for example, **t tests**:



We can also locate methods by searching the titles of methods. You specify the search string of interest in the *Filter* box at the top right of the PSS Control Panel. For example, if we type "mean" in the *Filter* box while keeping the focus on **Hypothesis test**, only test methods with a title containing "mean" will be listed in the right pane.



We can specify multiple words in the *Filter* box, and only methods with all the specified words in their titles will appear. For example, if we type "two means", only methods with the words "two" and "means" in their titles will be shown:

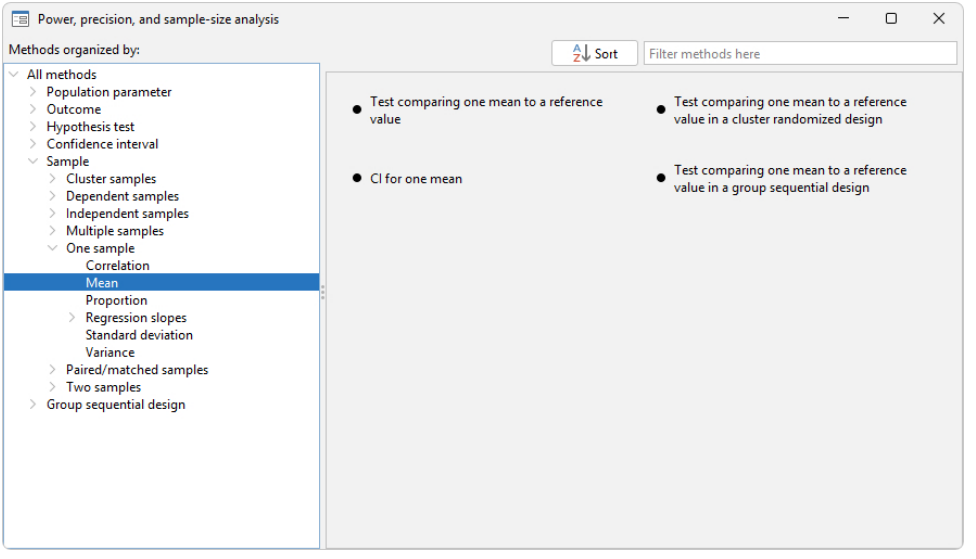


The search is performed within the group of methods selected by the choice in the left pane. In the above example, the search was done within **Hypothesis test**. When you search all methods, whether you select **Population parameter**, **Outcome**, or **Sample** in the left pane, the same set of methods appears in the right pane but in the order determined by the selected category.

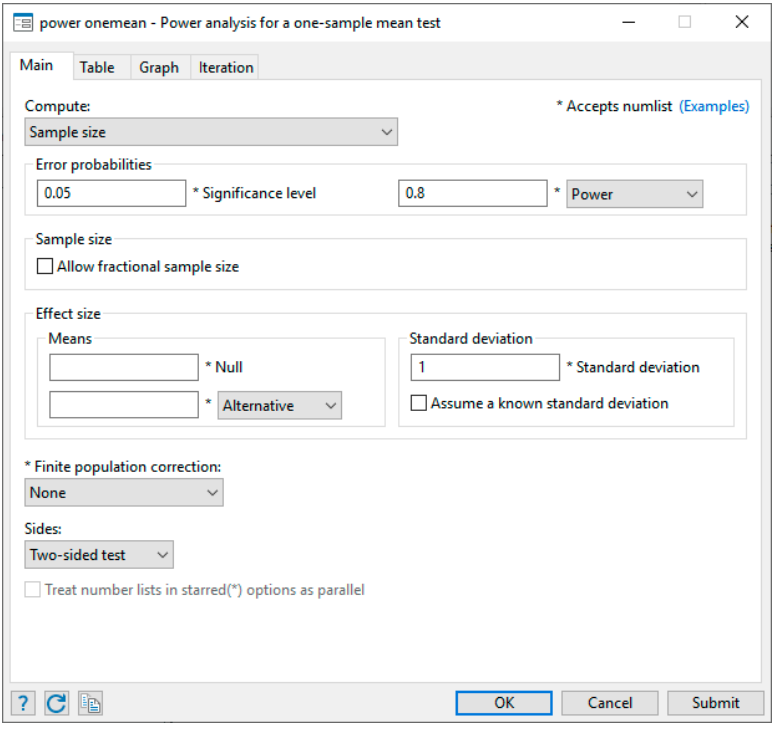
Example using PSS Control Panel

In *An example of PSS analysis in Stata* in [PSS-2] **Intro (power)**, we performed PSS analysis interactively by typing commands. We replicate the analysis by using the PSS Control Panel and dialog boxes.

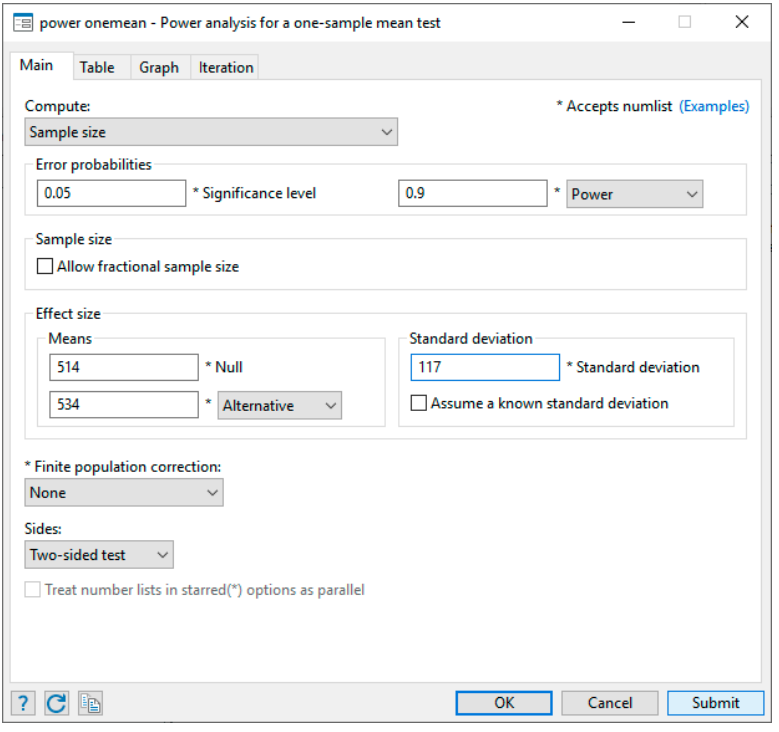
We first launch the PSS Control Panel from the **Statistics** > **Power, precision, and sample size** menu. We then narrow down to the desired dialog box by first choosing **Sample** in the left pane, then choosing **One sample** within that, and then choosing **Mean**. In the right pane, we see methods for testing the one-sample mean. We are interested in the **Test comparing one mean to a reference value**.



We invoke the dialog box by clicking on the corresponding method title in the right pane. The following appears:



Following the example from *An example of PSS analysis in Stata* in [PSS-2] **Intro (power)**, we now compute sample size. The first step is to choose which parameter to compute. The *Compute* drop-down box specifies `Sample size`, so we leave it unchanged. The next step is to specify error probabilities. The default significance level is already set to our desired value of 0.05, so we leave it unchanged. We change power from the default value of 0.8 to 0.9. We then specify a null mean of 514, an alternative mean of 534, and a standard deviation of 117 in the *Effect size* group of options. We leave everything else unchanged and click on the **Submit** button to obtain results.

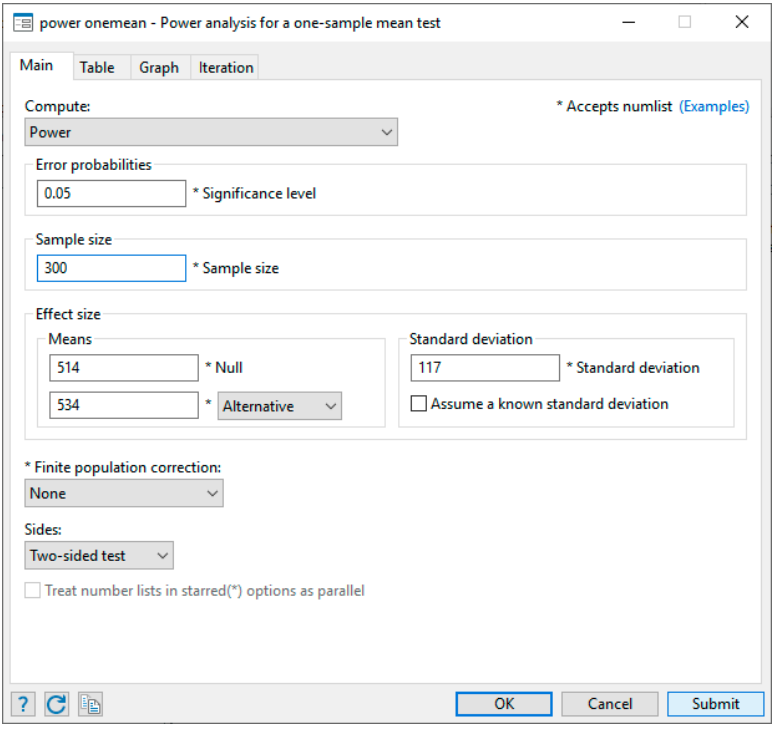


The following command is displayed in the Results window and executed:

```
. power onemean 514 534, power(0.9) sd(117)
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0
Study parameters:
        alpha =    0.0500
        power =    0.9000
        delta =    0.1709
           m0 =  514.0000
           ma =  534.0000
           sd =  117.0000
Estimated sample size:
            N =       362
```

We can verify that the command and results are exactly the same as what we specified in *An example of PSS analysis in Stata* of [PSS-2] **Intro (power)**.

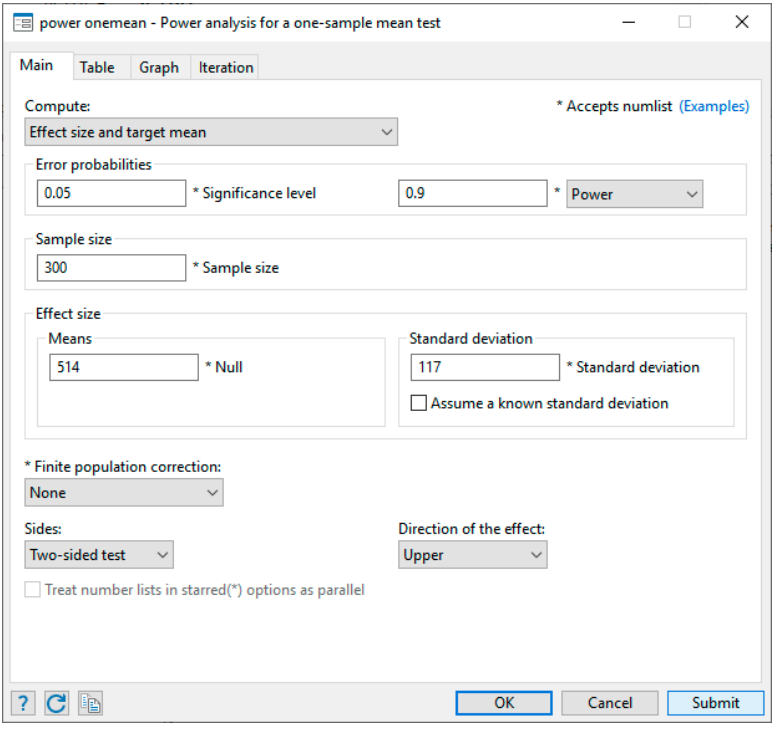
Continuing our PSS analysis, we now want to compute power for a sample of 300 subjects. We return to the dialog box and select `Power` under *Compute*. The only thing we need to specify is the sample size of 300:



The following command is issued after we click on the **Submit** button:

```
. power onemean 514 534, n(300) sd(117)

Estimated power for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0

Study parameters:

        alpha =    0.0500
            N =       300
        delta =    0.1709
           m0 =  514.0000
           ma =  534.0000
           sd =  117.0000

Estimated power:

        power =    0.8392
```

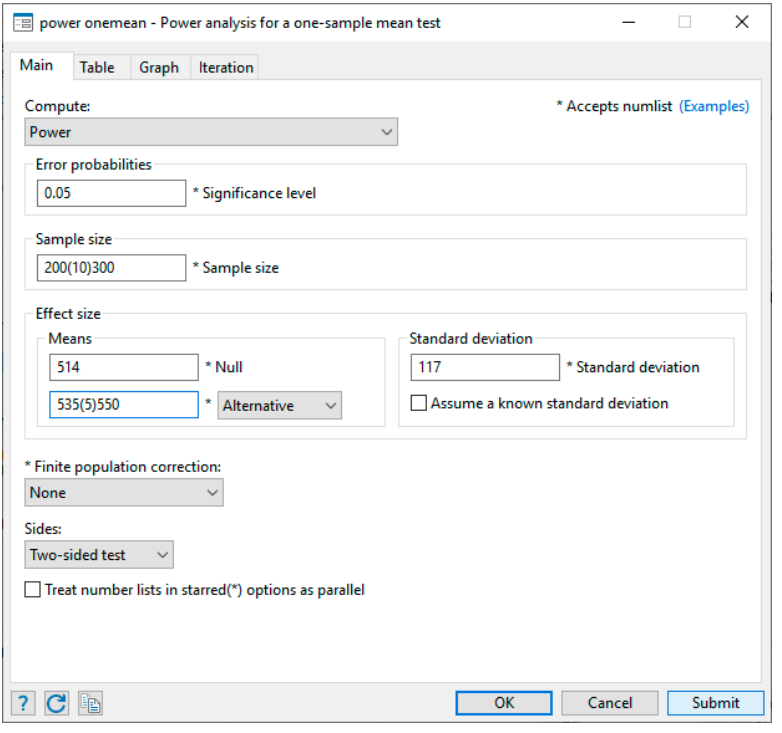
To compute effect size, we select `Effect size and target mean` under *Compute*. All the previously used values for power and sample size are preserved, so we do not need to specify anything additional.



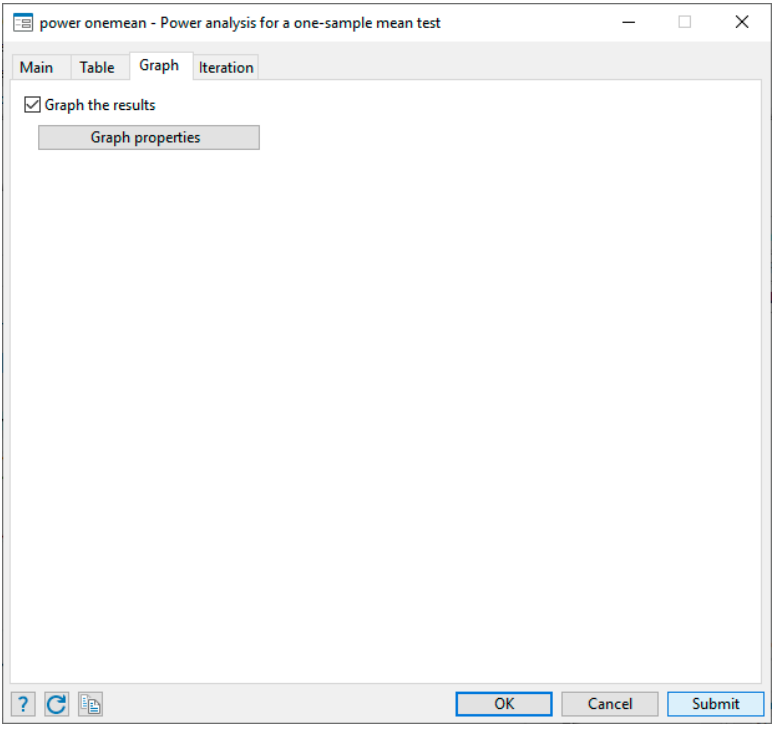
We click on the **Submit** button and get the following:

```
. power onemean 514, power(0.9) n(300) sd(117)

Performing iteration ...

Estimated target mean for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0; ma > m0

Study parameters:

        alpha =    0.0500
        power =    0.9000
            N =       300
           m0 =  514.0000
           sd =  117.0000

Estimated effect size and target mean:

        delta =    0.1878
           ma =  535.9671
```

To produce the graph from *An example of PSS analysis in Stata*, we first select `Power` under *Compute*. Then we specify the *numlist*s for sample size and alternative mean in the respective edit boxes:

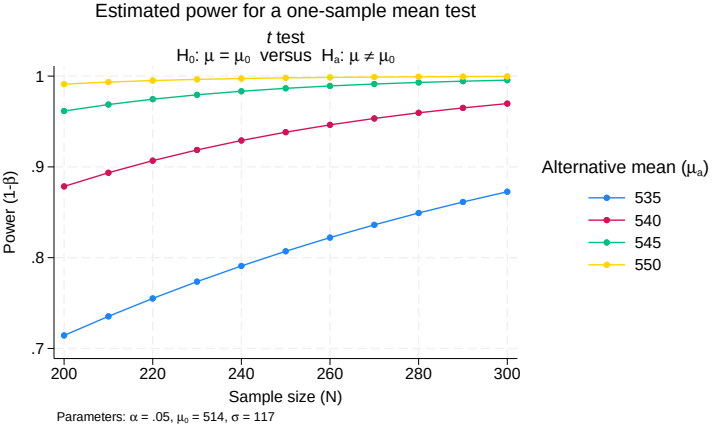


We also check the *Graph the results* box on the **Graph** tab:



We click on the **Submit** button and obtain the following command and graph:

```
. power onemean 514 (535(5)550), n(200(10)300) sd(117) graph
```



Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **Intro (power)** — Introduction to power and sample-size analysis for hypothesis tests

[PSS-5] **Glossary**

[ADAPT] **GSD intro** — Introduction to group sequential designs

power — Power and sample-size analysis for hypothesis tests

Description

The `power` command is useful for planning studies. It performs power and sample-size analysis for studies that use hypothesis testing to form inferences about population parameters. You can compute sample size given power and effect size, power given sample size and effect size, or the minimum detectable effect size and the corresponding target parameter given power and sample size. You can display results in a table ([PSS-2] **power, table**) and on a graph ([PSS-2] **power, graph**).

For precision and sample-size analysis for CIs, see [PSS-3] **ciwidth**.

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

`power` *method* `...[ , power(`*numlist*`)` *power_options* `... ]`

Compute power

`power` *method* `..., n(`*numlist*`)` [*power_options* `... ]`

Compute effect size and target parameter

`power` *method* `..., n(`*numlist*`) power(`*numlist*`)` [*power_options* `... ]`

method	Description
One sample	
onemean	One-sample mean test (one-sample t test)
oneproportion	One-sample proportion test
onecorrelation	One-sample correlation test
onevariance	One-sample variance test
Two independent samples	
twomeans	Two-sample means test (two-sample t test)
twoproportions	Two-sample proportions test
twocorrelations	Two-sample correlations test
twovariances	Two-sample variances test
Two paired samples	
pairedmeans	Paired-means test (paired t test)
pairedproportions	Paired-proportions test (McNemar's test)
Analysis of variance	
oneway	One-way ANOVA
twoway	Two-way ANOVA
repeated	Repeated-measures ANOVA
Linear regression	
oneslope	Slope test in a simple linear regression
rsquared	R^2 test in a multiple linear regression
pcorr	Partial-correlation test in a multiple linear regression
Logistic regression	
logistic *one binary*	Coefficient test in logistic regression with one binary predictor
logistic *two binary*	Coefficient test in logistic regression with two binary predictors
logistic *general*	Coefficient test in logistic regression with general predictors
Contingency tables	
cmh	Cochran–Mantel–Haenszel test (stratified 2×2 tables)
mcc	Matched case–control studies
trend	Cochran–Armitage trend test (linear trend in $J \times 2$ table)
Survival analysis	
cox	Cox proportional hazards model
exponential	Two-sample exponential test
logrank	Log-rank test
Cluster randomized design (CRD)	
onemean, cluster	One-sample mean test in a CRD
oneproportion, cluster	One-sample proportion test in a CRD
twomeans, cluster	Two-sample means test in a CRD
twoproportions, cluster	Two-sample proportions test in a CRD
logrank, cluster	Log-rank test in a CRD
User-defined methods	
usermethod	Add your own method to power

power_options	Description
Main	
* alpha(*numlist*)	significance level; default is alpha(0.05)
* power(*numlist*)	power; default is power(0.8)
* beta(*numlist*)	probability of type II error; default is beta(0.2)
* n(*numlist*)	total sample size; required to compute power or effect size
* n1(*numlist*)	sample size of the control group
* n2(*numlist*)	sample size of the experimental group
* nratio(*numlist*)	ratio of sample sizes, N2/N1; default is nratio(1), meaning equal group sizes
compute(N1 | N2)	solve for N1 given N2 or for N2 given N1
nfractional	allow fractional sample sizes
direction(upper|lower)	direction of the effect for effect-size determination; default is direction(upper), which means that the postulated value of the parameter is larger than the hypothesized value
onesided	one-sided test; default is two sided
parallel	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[no]table[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
saving(*filename* [, replace])	save the table data to *filename*; use replace to overwrite existing *filename*
Graph	
graph[(*graphopts*)]	graph results; see [PSS-2] **power, graph**
Iteration	
init(#)	initial value of the estimated parameter; default is method specific
iterate(#)	maximum number of iterations; default is iterate(500)
tolerance(#)	parameter tolerance; default is tolerance(1e-12)
ftolerance(#)	function tolerance; default is ftolerance(1e-12)
[no]log	suppress or display iteration log
[no]dots	suppress or display iterations as dots
notitle	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the parallel option.

Options n1(), n2(), nratio(), and compute() are available only for two-independent-samples methods.

Iteration options are available only with computations requiring iteration.

collect is allowed; see **[U] 11.1.10 Prefix commands**.

notitle does not appear in the dialog box.

Options

Main

alpha(*numlist*) sets the significance level of the test. The default is alpha(0.05).

power(*numlist*) sets the power of the test. The default is power(0.8). If beta() is specified, this value is set to be 1 − beta(). Only one of power() or beta() may be specified.

beta(*numlist*) sets the probability of a type II error of the test. The default is beta(0.2). If power() is specified, this value is set to be 1 − power(). Only one of beta() or power() may be specified.

n(*numlist*) specifies the total number of subjects in the study to be used for power or effect-size determination. If n() is specified, the power is computed. If n() and power() or beta() are specified, the minimum effect size that is likely to be detected in a study is computed.

n1(*numlist*) specifies the number of subjects in the control group to be used for power or effect-size determination.

n2(*numlist*) specifies the number of subjects in the experimental group to be used for power or effect-size determination.

nratio(*numlist*) specifies the sample-size ratio of the experimental group relative to the control group, N2/N1, for two-sample tests. The default is nratio(1), meaning equal allocation between the two groups.

compute(N1 | N2) requests that the power command compute one of the group sample sizes given the other one, instead of the total sample size, for two-sample tests. To compute the control-group sample size, you must specify compute(N1) and the experimental-group sample size in n2(). Alternatively, to compute the experimental-group sample size, you must specify compute(N2) and the control-group sample size in n1().

nfractional specifies that fractional sample sizes be allowed. When this option is specified, fractional sample sizes are used in the intermediate computations and are also displayed in the output.

Also see the description and the use of options n(), n1(), n2(), nratio(), and compute() for two-sample tests in [PSS-4] **Unbalanced designs**.

direction(upper | lower) specifies the direction of the effect for effect-size determination. For most methods, the default is direction(upper), which means that the postulated value of the parameter is larger than the hypothesized value. For survival methods, the default is direction(lower), which means that the postulated value is smaller than the hypothesized value.

onesided indicates a one-sided test. The default is two sided.

parallel requests that computations be performed in parallel over the lists of numbers specified for at least two study parameters as command arguments, starred options allowing *numlist*, or both. That is, when parallel is specified, the first computation uses the first value from each list of numbers, the second computation uses the second value, and so on. If the specified number lists are of different sizes, the last value in each of the shorter lists will be used in the remaining computations. By default, results are computed over all combinations of the number lists.

For example, let a_1 and a_2 be the list of values for one study parameter, and let b_1 and b_2 be the list of values for another study parameter. By default, power will compute results for all possible combinations of the two values in the two study parameters: (a_1, b_1), (a_1, b_2), (a_2, b_1), and (a_2, b_2). If parallel is specified, power will compute results for only two combinations: (a_1, b_1) and (a_2, b_2).

Table

notable, table, and table() control whether or not results are displayed in a tabular format. table is implied if any number list contains more than one element. notable is implied with graphical output—when either the graph or the graph() option is specified. table() is used to produce custom tables. See [PSS-2] **power, table** for details.

saving(*filename* [, replace]) creates a Stata data file (.dta file) containing the table values with variable names corresponding to the displayed *columns*. replace specifies that *filename* be overwritten if it exists. saving() is only appropriate with tabular output.

Graph

graph and graph() produce graphical output; see [PSS-2] **power, graph** for details.

The following options control an iteration procedure used by the power command for solving nonlinear equations.

Iteration

init(#) specifies an initial value for the estimated parameter. Each power method sets its own default value. See the documentation entry of the method for details.

iterate(#) specifies the maximum number of iterations for the Newton method. The default is iterate(500).

tolerance(#) specifies the tolerance used to determine whether successive parameter estimates have converged. The default is tolerance(1e-12). See *Convergence criteria* in [M-5] **solvenl( )** for details.

ftolerance(#) specifies the tolerance used to determine whether the proposed solution of a nonlinear equation is sufficiently close to 0 based on the squared Euclidean distance. The default is ftolerance(1e-12). See *Convergence criteria* in [M-5] **solvenl( )** for details.

log and nolog specify whether an iteration log is to be displayed. The iteration log is suppressed by default. Only one of log, nolog, dots, or nodots may be specified.

dots and nodots specify whether a dot is to be displayed for each iteration. The iteration dots are suppressed by default. Only one of dots, nodots, log, or nolog may be specified.

The following option is available with power but is not shown in the dialog box:

notitle prevents the command title from displaying.

Remarks and examples

Remarks are presented under the following headings:

This section describes how to perform power and sample-size analysis using the `power` command. For a software-free introduction to power and sample-size analysis, see [PSS-2] **Intro (power)**.

Using the power command

The `power` command computes sample size, power, or minimum detectable effect size and the corresponding target parameter for various hypothesis tests. You can also add your own methods to the `power` command as described in [PSS-2] ***power usermethod***.

All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option. You can specify the `onesided` option to request a one-sided test.

By default, the `power` command computes sample size for the default power of 0.8. You may change the value of power by specifying the `power()` option. Instead of power, you can specify the probability of a type II error in the `beta()` option.

To compute power, you must specify the sample size in the `n()` option.

To compute power or sample size, you must also specify a magnitude of the effect desired to be detected by a hypothesis test. `power`'s methods provide several ways in which an effect can be specified. For example, for a one-sample mean test, you can specify either the target mean or the difference between the target mean and a reference mean; see [PSS-2] **power onemean**.

You can also compute the smallest magnitude of the effect or the minimum detectable effect size (MDES) and the corresponding target parameter that can be detected by a hypothesis test given power and sample size. To compute MDES, you must specify both the desired power in the `power()` option or the probability of a type II error in the `beta()` option and the sample size in the `n()` option. In addition to the effect size, `power` also reports the estimated value of the parameter of interest, such as the mean under the alternative hypothesis for a one-sample test or the experimental-group proportion for a two-sample test of independent proportions. By default, when the postulated value is larger than the hypothesized value, the `power` command assumes an effect in the upper direction, the `direction(upper)` option. You may request an estimate of the effect in the opposite, lower, direction by specifying the `direction(lower)` option.

For hypothesis tests comparing two independent samples, you can compute one of the group sizes given the other one instead of the total sample size. In this case, you must specify the label of the group size you want to compute in the `compute()` option and the value of the other group size in the respective `n#()` option. For example, if we wanted to find the size of the second group given the size of the first group, we would specify the combination of options `compute(N2)` and `n1(#)`.

A balanced design is assumed by default for two-independent-samples hypothesis tests, but you can request an unbalanced design. For example, you can specify the allocation ratio n_2/n_1 between the two groups in the `nratio()` option or the individual group sizes in the `n1()` and `n2()` options. See [PSS-4] **Unbalanced designs** for more details about various ways of specifying an unbalanced design.

For sample-size determination, the reported integer sample sizes may not correspond exactly to the specified power because of rounding. To obtain conservative results, the `power` command rounds up the sample size to the nearest integer so that the corresponding power is at least as large as the requested one. You can specify the `nfractional` option to obtain the corresponding fractional sample size.

Some of `power`'s computations require iteration. The defaults chosen for the iteration procedure should be sufficient for most situations. In a rare situation when you may want to modify the defaults, the `power` command provides options to control the iteration procedure. The most commonly used is the `init()` option for supplying an initial value of the estimated parameter. This option can be useful in situations where the computations are sensitive to the initial values. If you are performing computations for many combinations of various study parameters, you may consider reducing the default maximum number of iterations of 500 in the `iterate()` option so that the command is not spending time on calculations in difficult-to-compute regions of the parameter space. By default, `power` suppresses the iteration log. If desired, you can specify the `log` option to display the iteration log or the `dots` option to display iterations as dots to monitor the progress of the iteration procedure.

The `power` command can produce results for one study scenario or for multiple study scenarios when multiple values of the parameters are specified; see *Specifying multiple values of study parameters* below for details.

For a single result, `power` displays results as text. For multiple results or if the `table` option is specified, `power` displays results in a table. You can also display multiple results on a graph by specifying the `graph` option. Graphical output suppresses the table of the results; use the `table` option to also see the tabular output. You can customize the default tables and graphs by specifying suboptions within the respective options `table()` and `graph()`; see [PSS-2] **power, table** and [PSS-2] **power, graph** for details.

You can also save the tabular output to a Stata dataset by using the `saving()` option.

Specifying multiple values of study parameters

The `power` command can produce results for one study scenario or for multiple study scenarios when multiple values of the parameters are supplied to the supported options. The options that support multiple values specified as a *numlist* are marked with a star in the syntax diagram.

For example, the `n()` option supports multiple values. You can specify multiple sample sizes as individual values, `n(100 150 200)`, or as a range of values, `n(100(50)200)`; see **[U] 11.1.8 numlist** for other specifications.

In addition to options, you may specify multiple values of command arguments, values specified after the command name. For example, let $\#_1$ and $\#_2$ be the first and the second command arguments in

```
. power twoproportions #1 #2, ...
```

If we want to specify multiple values for the command arguments, we must enclose these values in parentheses. For example,

```
. power twoproportions (0.1 0.2) (0.1 0.2 0.3 0.4), ...
```

or, more generally,

```
. power twoproportions (numlist) (numlist), ...
```

When multiple values are specified in multiple options or for multiple command arguments, the `power` command computes results for all possible combinations formed by the values from every option and command argument. In some cases, you may want to compute results in parallel for specific sets of values of the specified parameters. To request this, you can specify the `parallel` option. If the specified number lists are of varying sizes, *numlist* with the maximum size determines the number of final results produced by `power`. The last value from *numlist* of smaller sizes will be used in the subsequent computations.

For example,

```
. power twoproportions (0.1 0.2) 0.4, n(100 200)
```

is equivalent to

```
. power twoproportions 0.1 0.4, n(100)
. power twoproportions 0.2 0.4, n(100)
. power twoproportions 0.1 0.4, n(200)
. power twoproportions 0.2 0.4, n(200)
```

When the `parallel` option is specified,

```
. power twoproportions (0.1 0.2) 0.4, n(100 200) parallel
```

is equivalent to

```
. power twoproportions 0.1 0.4, n(100)
. power twoproportions 0.2 0.4, n(200)
```

When the `parallel` option is specified and *numlist* is of different sizes, the last value of the shorter *numlist* is used in the subsequent computations. For example,

```
. power twoproportions (0.1 0.2 0.3) 0.4, n(100 200) parallel
```

is equivalent to

```
. power twoproportions 0.1 0.4, n(100)
. power twoproportions 0.2 0.4, n(200)
. power twoproportions 0.3 0.4, n(200)
```

One-sample tests

The `power` command provides PSS computations for four one-sample tests. `power onemean` performs PSS analysis for a one-sample mean test; `power oneproportion` performs PSS analysis for a one-sample proportion test; `power onecorrelation` performs PSS analysis for a one-sample correlation test; and `power onevariance` performs PSS analysis for a one-sample variance test.

power onemean provides PSS computations for a one-sample t test assuming known or unknown population standard deviation. It also provides a way to adjust computations for a finite population sample. See [PSS-2] **power onemean**.

power oneproportion provides PSS computations for a test that compares one proportion with a reference value. By default, the computations are based on a large-sample z test that uses the normal approximation of the distribution of the test statistic. You may choose between two large-sample tests: the score test or Wald test. You may also compute power for the small-sample binomial test by specifying the test(binomial) option. See [PSS-2] **power oneproportion**.

power onecorrelation provides PSS computations for a test that compares one correlation with a reference value. The computations are based on a Fisher's z transformation of a correlation coefficient. See [PSS-2] **power onecorrelation**.

power onevariance provides PSS computations for a test that compares one variance with a reference value. The computations are based on a χ^2 test of the ratio of the variance to its reference value. You can perform computations in the variance or standard deviation metric. See [PSS-2] **power onevariance**.

All one-sample methods compute sample size given power and target parameter, power given sample size and target parameter, or MDES and the corresponding target parameter given power and sample size.

For PSS determination, an effect may be supplied by specifying the null and alternative values of the target parameter as command arguments $\#_0$ and $\#_a$:

. power *onesample* $\#_0$ $\#_a$, ...

Instead of the alternative value $\#_a$, you can specify the ratio of the alternative value to the null value in the ratio() option and the null value as $\#_0$ for power onevariance,

. power onevariance $\#_0$, ratio(#) ...

or you can specify the difference between the alternative value and the null value in the diff() option and the null value as $\#_0$ for other methods,

. power *onesample* $\#_0$, diff(#) ...

For sample-size determination, the reported sample size is rounded up to the nearest integer. This ensures that the corresponding actual power is at least as large as the specified power. You can specify the nfractional option to obtain the corresponding fractional sample size, or you can recompute the actual power using the reported rounded value; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for details.

Below we show a quick example of PSS analysis for a one-sample mean test. See entries of the one-sample methods for more examples.

▷ Example 1: PSS analysis for a one-sample mean test

A group of pediatricians would like to study the exposure of infants to television. The group wants to investigate whether the average number of hours watched per day by infants between 3 and 12 months of age is greater than 2 hours. Before conducting a study, pediatricians would like to determine how many infants they need to enroll in the study. The analysis will use the one-sample t test to compare the mean of the obtained sample with the reference value. An earlier pilot study reported an average of 2.5 hours watched per day with a standard deviation of 0.8. Pediatricians would like to compute the sample

size required to detect a mean of 2.5 using a two-sided test with 5% significance level and 80% power. Although pediatricians suspect that the effect is in the upper direction—more than two hours watched on average—they prefer to obtain the required sample size for a two-sided test instead of a one-sided test.

We use `power onemean` to compute the required sample size. We specify the reference or null value of 2 and the comparison or alternative value of 2.5 as command arguments. We also specify the standard deviation of 0.8 in the `sd()` option. We omit the `alpha(0.05)` and `power(0.8)` options because the desired values are the defaults for these options. The default test is two sided, so we do not need to supply any additional information to the command.

```
. power onemean 2 2.5, sd(0.8)
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.6250
           m0 =    2.0000
           ma =    2.5000
           sd =    0.8000
Estimated sample size:
            N =        23
```

All `power` commands have a similar output format. Information about the test and tested hypothesis is displayed first. The input and implied values of the study parameters are displayed next under `Study parameters`. The estimated parameters, such as the sample size in this example, are displayed last.

Pediatricians need to enroll 23 infants in the study to detect a standardized difference of 0.625 between the alternative mean of 2.5 and the null mean of 2 given a standard deviation of 0.8 using a 5%-level two-sided one-sample t test with 80% power.

The pediatricians believe that they have resources to enroll more infants. They wish to compute the power that corresponds to the sample size of 50. To compute the corresponding power, we specify a sample size of 50 in the `n()` option:

```
. power onemean 2 2.5, sd(0.8) n(50)
Estimated power for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0
Study parameters:
        alpha =    0.0500
            N =        50
        delta =    0.6250
           m0 =    2.0000
           ma =    2.5000
           sd =    0.8000
Estimated power:
        power =    0.9911
```

The power increases to 99% for a larger sample of 50 infants.

The pediatricians also want to find out what is the smallest mean difference they can detect with the larger sample of 50 infants while keeping the power at 80%. They assume the effect to be in the upper direction for this computation. To compute the minimum detectable difference, we specify both the sample size in the `n()` option and the power in the `power()` option.

```
. power onemean 2, sd(0.8) n(50) power(0.8)

Performing iteration ...

Estimated target mean for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0; ma > m0

Study parameters:

        alpha =    0.0500
        power =    0.8000
            N =        50
           m0 =    2.0000
           sd =    0.8000

Estimated effect size and target mean:

        delta =    0.4042
           ma =    2.3233
```

The smallest standardized difference that can be detected given the study parameters is about 0.4, with a corresponding mean of 2.32.

◁

Two-sample tests

The `power` command provides PSS computations for four two-sample tests. `power twomeans` performs PSS analysis for a two-sample means test; `power twoproportions` performs PSS analysis for a two-sample proportions test; `power twocorrelations` performs PSS analysis for a two-sample correlations test; and `power twovariances` performs PSS analysis for a two-sample variances test.

`power twomeans` provides PSS computations for a two-sample means test that compares the means of two independent populations. The computations provided assume known or unknown and equal or unequal population standard deviations of the two groups. See [PSS-2] **power twomeans**.

`power twoproportions` provides PSS computations for a two-sample proportions test that compares the proportions in two independent populations with binary outcomes. Three tests are supported: the large-sample Pearson's χ^2 test, the large-sample likelihood-ratio test, and the small-sample Fisher's exact test. Several effect specifications are available. For example, you can specify the effect of interest as a risk difference, or a relative risk, or an odds ratio. See [PSS-2] **power twoproportions**.

`power twocorrelations` provides PSS computations for a two-sample correlations test that compares the correlation coefficients of two independent populations. The computations are based on a Fisher's z transformation of a correlation coefficient. See [PSS-2] **power twocorrelations**.

`power twovariances` provides PSS computations for a two-sample variances test that compares the variances of two independent populations. The computations are based on an F test of the ratio of variances. You can perform computations in the variance or standard deviation metric. See [PSS-2] **power twovariances**.

Also see *Survival analysis* for power and sample-size analysis for a two-sample comparison of survivor functions using the `power logrank` and `power exponential` commands.

All two-sample methods compute sample size given power and the control-group and experimental-group values of the target parameter, power given sample size and the control-group and experimental-group values of the target parameter, or MDES and the corresponding target value of the parameter in the experimental group given power, sample size, and the control-group parameter value.

To compute sample size or power, you can specify the magnitude of the effect of interest in two ways: by directly specifying the alternative values of the target parameter in two groups or by specifying the control-group alternative value and the corresponding relation of the experimental-group value to the control-group alternative value.

The two alternative values are specified as command arguments: the alternative value of the target parameter in the control or reference group, $\#_{a1}$, and the alternative value of the target parameter in the experimental or comparison group, $\#_{a2}$:

. `power` *twosample* $\#_{a1}$ $\#_{a2}$`, ...`

The experimental-group alternative value, $\#_{a2}$, may be omitted if an option containing the relationship between the two alternative values is specified. For example, for `power twomeans` and `power twocorrelations`, such an option is `diff()`, and it specifies the difference between the experimental-group and control-group alternative values:

. `power twomeans` $\#_{a1}$`, diff(#) ...`

For `power twovariances`, such an option is `ratio()`, and it contains the ratio of the experimental-group alternative value to the control-group value:

. `power twovariances` $\#_{a1}$`, ratio(#) ...`

`power twoproportions` provides several alternative specifications in which a difference between the two populations may be expressed. For example, you can express the "difference" as an odds ratio of the experimental group to the control group,

. `power twoproportions` $\#_{a1}$`, oratio(#) ...`

or as a relative risk,

. `power twoproportions` $\#_{a1}$`, rrisk() ...`

In addition to the total sample size, two-sample methods provide a way to solve for one of the group sizes when the other group size is fixed. This can be achieved by specifying the `compute()` option. To compute the size of the first group, you must specify the `compute(N1)` option and the size of the second group in the `n2()` option. To compute the size of the second group, you must specify the `compute(N2)` option and the size of the first group in the `n1()` option.

To compute power, you can specify a total sample size in the `n()` option, group sample sizes in the `n1()` and `n2()` options, or one of the group sample sizes and its ratio, n_2/n_1, in the `nratio()` option; see [PSS-4] **Unbalanced designs** for more specifications.

Below we show a quick example of PSS analysis for a two-sample means test. See entries of the two-sample methods for more examples.

▷ Example 2: PSS analysis for a two-sample mean test

A pharmaceutical company would like to conduct a study to compare a new weight-loss drug with an older drug. Investigators are planning to use a two-sample t test to compare the average weight loss for the two drugs. The average weight loss of people taking the old drug for 3 months is 12 pounds, with a standard deviation of 5.5 pounds. The new drug is expected to produce an average weight loss of 16 pounds, with a standard deviation of 5 pounds for the same period of time. Investigators want to find out how many subjects they need to recruit into the study to detect the specified difference using a 5% level two-sided test with 90% power.

We use `power twomeans` to perform PSS analyses. We specify the control-group mean 12 and the experimental-group mean 16 as command arguments after the command name. We specify the respective standard deviations in the `sd1()` and `sd2()` options. The default power is set to 0.8, so we specify `power(0.9)` to request 90% power.

```
. power twomeans 12 16, sd1(5.5) sd2(5) power(0.9)

Performing iteration ...

Estimated sample sizes for a two-sample means test
Satterthwaite's t test assuming unequal variances
H0: m2 = m1  versus  Ha: m2 != m1

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =    4.0000
           m1 =   12.0000
           m2 =   16.0000
          sd1 =    5.5000
          sd2 =    5.0000

Estimated sample sizes:

            N =        76
  N per group =        38
```

We need a sample of 76 subjects, 38 per group, to detect a difference of 4 between the control-group mean of 12 and the experimental-group mean of 16 given the respective standard deviations of 5.5 and 5 with 90% power using a 5%-level two-sided two-sample means t test.

The default test is two sided. You may specify the `onesided` option to request a one-sided test. The default design is also balanced; see [PSS-4] **Unbalanced designs** for examples of unbalanced designs.

The investigators hope to keep the sample size under 60 and would like to compute the power corresponding to this sample size. To compute the corresponding power, we specify the `n(60)` option instead of the `power()` option:

```
. power twomeans 12 16, sd1(5.5) sd2(5) n(60)

Estimated power for a two-sample means test
Satterthwaite's t test assuming unequal variances
H0: m2 = m1  versus  Ha: m2 != m1

Study parameters:

        alpha =    0.0500
            N =        60
  N per group =        30
        delta =    4.0000
           m1 =   12.0000
           m2 =   16.0000
          sd1 =    5.5000
          sd2 =    5.0000

Estimated power:

        power =    0.8259
```

The power decreases to 83% for the smaller sample of 60 subjects.

To keep the power at 90%, the investigators want to compute the smallest difference between the experimental-group mean and the control-group mean (in the upper direction) given the sample of 60 subjects. For this computation, we specify both options `n(60)` and `power(0.9)`:

```
. power twomeans 12, sd1(5.5) sd2(5) n(60) power(0.9)

Performing iteration ...

Estimated experimental-group mean for a two-sample means test
Satterthwaite's t test assuming unequal variances
H0: m2 = m1  versus  Ha: m2 != m1; m2 > m1

Study parameters:

        alpha =    0.0500
        power =    0.9000
            N =        60
  N per group =        30
           m1 =   12.0000
          sd1 =    5.5000
          sd2 =    5.0000

Estimated effect size and experimental-group mean:

        delta =    4.4744
           m2 =   16.4744
```

The smallest detectable mean difference is 4.47, with a corresponding value of the experimental-group mean of 16.47.

◁

Paired-sample tests

The `power` command provides PSS computations for two tests of paired samples. `power pairedmeans` performs PSS analysis for a two-sample paired-means test, and `power pairedproportions` performs PSS analysis for a two-sample paired-proportions test.

`power pairedmeans` provides PSS computations for a two-sample paired t test assuming known or unknown population standard deviation of the differences between paired observations. You can specify standard deviations of each group and a correlation between paired observations, or you can directly specify the standard deviation of the differences between observations. You can obtain results for a nonzero null hypothesis of a difference between the two paired means. The command also provides a way to adjust computations for a finite population sample. See [PSS-2] **power pairedmeans**.

power pairedproportions provides PSS computations for a two-sample paired-proportions test that compares proportions in two paired (correlated) samples. The computations are based on McNemar's test of marginal homogeneity. You can specify either the discordant proportions or the marginal proportions. A number of effect specifications are available. For example, you can specify the effect of interest as a relative risk or an odds ratio. See [PSS-2] **power pairedproportions**.

Both paired methods compute sample size given power and target parameter, power given sample size and target parameter, or MDES and the corresponding target parameter given power and sample size.

For power and sample-size determination of power pairedmeans, an effect may be supplied by specifying the alternative values of the two means, pretreatment and posttreatment, as command arguments m_{a1} and m_{a2}:

power pairedmeans m_{a1} m_{a2}, ...

Instead of the alternative value m_{a2}, you can specify the difference between the two alternative values in the altdiff() option and the alternative pretreatment mean value m_{a1}:

power pairedmeans m_{a1}, altdiff() ...

You may omit both alternative values and specify only the difference between them in the altdiff() option:

power pairedmeans, altdiff() ...

By default, the null value of the difference between the pretreatment and posttreatment means is zero, but you may change it by specifying the nulldiff() option.

For PSS determination of power pairedproportions, there are a number of ways of specifying an effect of interest; see *Alternative ways of specifying effect* in [PSS-2] **power pairedproportions**. Two main specifications include the specification of discordant proportions and the specification of marginal probabilities. Specifically, you can supply the information about the effect of interest as discordant proportions p_{12} and p_{21},

power pairedproportions p_{12} p_{21}, ...

or as marginal proportions p_{1+} and p_{+1}:

power pairedproportions p_{1+} p_{+1}, corr(*numlist*) ...

When you specify marginal proportions, you must also specify the correlation between paired observations in the corr() option.

For sample-size determination, the reported sample size is rounded up to the nearest integer. This ensures that the corresponding actual power is at least as large as the specified power. You can specify the nfractional option to obtain the corresponding fractional sample size or you can recompute the actual power using the reported rounded value; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for details.

Below we show a quick example of PSS analyses for a two-sample paired-means test. See [PSS-2] **power pairedmeans** and [PSS-2] **power pairedproportions** for more examples.

▷ Example 3: PSS analysis for a two-sample paired-means test

A forester would like to study the effects of a fertilizer treatment on heights of Virginia pine trees. The trees are planted in pairs with only one of them receiving the fertilizer treatment. The average height of untreated trees is 27.5 feet, with a standard deviation of 4.5 feet. The fertilizer treatment is expected

to increase the average height to 30 feet, with the same standard deviation of 4.5 feet. The correlation between the paired tree heights is expected to be 0.4. The forester would like to know how many pairs of pine trees need to be planted so that a 5%-level two-sided paired-means t test detects the anticipated difference with 80% power.

We use `power pairedmeans` for power and sample-size analysis. We supply the alternative pretreatment and posttreatment means of 27.5 and 30, respectively, as command arguments after the command name. The standard deviations of the two groups are the same, so we specify their common value in the `sd()` option. We specify the correlation of 0.4 in the `corr()` option. The default value for power is 0.8 and for significance level is 0.05, so we omit the corresponding options `power(0.8)` and `alpha(0.05)`.

```
. power pairedmeans 27.5 30, sd(4.5) corr(0.4)

Performing iteration ...

Estimated sample size for a two-sample paired-means test
Paired t test assuming sd1 = sd2 = sd
H0: d = d0  versus  Ha: d != d0

Study parameters:

        alpha =    0.0500          ma1 =   27.5000
        power =    0.8000          ma2 =   30.0000
        delta =    0.5072           sd =    4.5000
           d0 =    0.0000         corr =    0.4000
           da =    2.5000
         sd_d =    4.9295

Estimated sample size:

            N =        33
```

The forester needs 33 pairs of pine trees to run the experiment.

The forester has resources to plant more trees and would like to compute the power corresponding to the larger sample. To compute power given sample size, we specify sample size in the `n()` option:

```
. power pairedmeans 27.5 30, sd(4.5) corr(0.4) n(50)

Estimated power for a two-sample paired-means test
Paired t test assuming sd1 = sd2 = sd
H0: d = d0  versus  Ha: d != d0

Study parameters:

        alpha =    0.0500          ma1 =   27.5000
            N =        50          ma2 =   30.0000
        delta =    0.5072           sd =    4.5000
           d0 =    0.0000         corr =    0.4000
           da =    2.5000
         sd_d =    4.9295

Estimated power:

        power =    0.9400
```

The power increases to 0.94.

The forester may also wish to know the smallest detectable difference between average tree heights of the fertilized group and of the control group that can be detected with 80% power and sample size of 50. To compute this value, we specify both options `n(50)` and `power(0.8)`:

```
. power pairedmeans 27.5, sd(4.5) corr(0.4) n(50) power(0.8)
Performing iteration ...
Estimated target parameters for a two-sample paired-means test
Paired t test assuming sd1 = sd2 = sd
H0: d = d0  versus  Ha: d != d0; da > d0
Study parameters:
        alpha =    0.0500          ma1 =   27.5000
        power =    0.8000           sd =    4.5000
            N =        50         corr =    0.4000
           d0 =    0.0000
         sd_d =    4.9295
Estimated effect size and target parameters:
        delta =    0.4042
           da =    1.9924
          ma2 =   29.4924
```

The smallest detectable difference is 1.99, with a corresponding value of the average tree height for the fertilized trees of 29.5.

◁

Analysis of variance models

The `power` command provides PSS computations for three types of analyses of variance (ANOVA) designs: one way, two way, and repeated measures. `power oneway` performs PSS analysis for a one-way ANOVA. `power twoway` performs PSS analysis for a two-way ANOVA. `power repeated` performs PSS analysis for a repeated-measures ANOVA.

`power oneway` provides PSS computations for a one-way ANOVA model. You can choose between the overall F test of the equality of group means and a test of a mean contrast. You can either specify group means or specify their variability in the computations. See [PSS-2] **power oneway**.

`power twoway` provides PSS computations for a two-way fixed-effects ANOVA model. You can choose the overall F test of the main effect of a row factor, a column factor, or a row-by-column interaction. You can either specify cell means or specify the variance explained by the tested effect. See [PSS-2] **power twoway**.

`power repeated` provides PSS computations for one-way and two-way fixed-effects repeated-measures ANOVA models. You can choose the overall F test of the main effect of a between-subjects factor, a within-subject factor, or a between–within factor interaction. You can either specify cell means or specify the variance explained by the tested effect. See [PSS-2] **power repeated**.

All methods compute sample size given power and effect size, power given sample size and effect size, or effect size given power and sample size.

For power and sample-size determination of `power oneway`, an effect may be supplied by specifying the alternative values of group means as command arguments m_{a1}, m_{a2}, m_{a3}, and so on:

`power oneway` m_{a1} m_{a2} [m_{a3} ...], ...

Instead of the alternative group means, you can specify the variance of the group means in the `varmeans()` option and the number of groups in the `ngroups()` option:

`power oneway, ngroups() varmeans()` ...

For power and sample-size determination of `power twoway` and `power repeated`, an effect may be supplied by specifying the alternative values of cell means as command arguments $m_{a1,1}$, $m_{a1,2}$, and so on, in a matrix form:

`power twoway` $m_{a1,1}$ $m_{a1,2}$ [...] \ $m_{a2,1}$ $m_{a2,2}$ [...], ...

`power repeated` $m_{a1,1}$ $m_{a1,2}$ [...] [\ $m_{a2,1}$ $m_{a2,2}$ [...]], ...

Instead of the alternative cell means, you can specify the variance of the tested effect in the `vareffect()` option and the dimensions of the cell-means matrix: number of rows and columns for `power twoway` and number of groups and repeated measures for `power repeated`:

```
power twoway, nrows() ncols() factor() vareffect() ...
power repeated, ngroups() nrepeated() factor() vareffect() ...
```

The means can also be supplied as a matrix at the command line. For example, suppose that we have three groups.

`power oneway` m_{a1} m_{a2} m_{a3}, ...

The above command would be equivalent to

`matrix means =` (m_{a1},m_{a2},m_{a3})

```
power oneway means, ...
```

There are also other alternative specifications of an effect with these commands. See the specific entry of each command.

For sample-size determination, the reported sample size is rounded up to the nearest integer. This ensures that the corresponding actual power is at least as large as the specified power. You can specify the `nfractional` option to obtain the corresponding fractional sample size, or you can recompute the actual power using the reported rounded value; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for details.

Below we show a quick example of PSS analysis for a one-way ANOVA model. See [PSS-2] **power oneway**, [PSS-2] **power twoway**, and [PSS-2] **power repeated** for more examples.

▷ Example 4: PSS analysis for a one-way ANOVA model

Researchers would like to compare the effects of four drugs on systolic blood pressure. They would like to use a one-way ANOVA model to test the equality of mean blood-pressure measurements across four drugs. To conduct a study, the researchers need an estimate for the number of subjects to be enrolled in a study. From a previous pilot study, the variance between group means was estimated to be 57, and the error variance was estimated to be 115. The researchers would like to compute the required sample size to detect the effect size of $0.7040 = \sqrt{57/115}$ with 80% power using a 5%-level F test of the equality of means assuming a balanced design.

We use `power oneway` to compute the sample size. We specify the number of groups and the estimates of variances in the corresponding options. The default value for power is 0.8 and for significance level is 0.05, so we omit the corresponding options `power(0.8)` and `alpha(0.05)`.

```
. power oneway, ngroups(4) varmeans(57) varerror(115)

Estimated sample size for one-way ANOVA
F test for group effect
H0: delta = 0  versus  Ha: delta != 0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.7040
          N_g =         4
        Var_m =   57.0000
        Var_e =  115.0000

Estimated sample sizes:

            N =        28
  N per group =         7
```

The researchers need to recruit 28 subjects, 7 subjects per group, for this study.

Unfortunately, the researchers can afford to recruit only 20 subjects. They wish to compute the power corresponding to this smaller sample size. To compute power, we additionally specify sample size in the `n()` option:

```
. power oneway, n(20) ngroups(4) varmeans(57) varerror(115)

Estimated power for one-way ANOVA
F test for group effect
H0: delta = 0  versus  Ha: delta != 0

Study parameters:

        alpha =    0.0500
            N =        20
  N per group =         5
        delta =    0.7040
          N_g =         4
        Var_m =   57.0000
        Var_e =  115.0000

Estimated power:

        power =    0.6400
```

The power decreases to 0.64.

The researchers are not satisfied with such a low power. They now would like to compute the smallest effect size and the corresponding variance of means that can be detected with the power of 80% and the sample size of 20. To compute effect size, we specify both power and sample size in respective options:

```
. power oneway, n(20) power(0.8) ngroups(4) varerror(115)
Performing iteration ...
Estimated between-group variance for one-way ANOVA
F test for group effect
H0: delta = 0  versus  Ha: delta != 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
            N =        20
  N per group =         5
          N_g =         4
        Var_e =  115.0000
Estimated effect size and between-group variance:
        delta =    0.8353
        Var_m =   80.2329
```

The smallest detectable effect size is 0.8353, with a corresponding value of the between-group variance of 80.2329.

◁

Linear regression

The `power` command provides PSS computations for a linear regression model. `power oneslope` provides PSS computations for a slope test in a simple linear regression. `power rsquared` provides PSS computations for an R^2 test in a multiple linear regression. `power pcorr` provides PSS computations for a partial-correlation test in a multiple linear regression.

`power oneslope` provides estimates of sample size, power, or target slope in a simple linear regression. It supports multiple ways of specifying the effect size, which is defined as the difference between the alternative and null values of the slope multiplied by the ratio of standard deviations of the covariate to the error term. Instead of specifying the standard deviation of the error term using the `sderror()` option, users can specify the standard deviation of the dependent variable in `sdy()` or the correlation between the dependent variable and the covariate of interest in `corr()`. See [PSS-2] **power oneslope**.

`power rsquared` reports estimates of sample size, power, or target R^2 in a multiple linear regression using an R^2 test. An R^2 test is an F test of the coefficient of determination, R^2, which is used to test the significance of coefficients in a multiple linear regression. When the `ncontrol()` option is not specified, the computation is based on a test of all coefficients in the model. When the `ncontrol()` option is specified, the computation is based on a test of a subset of coefficients in the full model against the reduced model. See [PSS-2] **power rsquared**.

`power pcorr` provides estimates of sample size, power, or target squared partial correlation for a partial-correlation test in a multiple linear regression. `power pcorr` is an alternative to `power rsquared, ncontrol()` for testing the significance of a subset of coefficients using a partial-correlation test. See [PSS-2] **power pcorr**.

Below we show two examples of PSS analysis for a linear regression model.

▷ Example 5: Sample size for the test of the slope in a simple linear regression model

Consider a hypothetical study for which the goal is to investigate the effect of average time spent per day exercising on BMI, measured in kg/m^2. The parameter of interest is the slope coefficient b, which measures the effect of exercising on BMI. Our null hypothesis is H_0: $b = 0$ versus a two-sided alternative H_a: $b \neq 0$.

We wish to compute the sample size required to detect a drop in BMI of 0.1 $\mathrm{kg/m^2}$ per minute of exercise, with 80% power using a 5%-level two-sided test. We assume a standard deviation of 10 minutes for time spent exercising in `sdx()` and 4.0 $\mathrm{kg/m^2}$ for BMI in `sdy()`.

```
. power oneslope 0 -0.1, sdx(10) sdy(4)

Performing iteration ...

Estimated sample size for a linear regression slope test
t test
H0: b = b0  versus  Ha: b != b0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =   -0.2582
           b0 =    0.0000
           ba =   -0.1000
          sdx =   10.0000
      sderror =    3.8730
          sdy =    4.0000

Estimated sample size:

            N =       120
```

The required sample size is 120. See [PSS-2] **power oneslope** for details.

◁

▷ Example 6: Power of an R^2 test in a multiple linear regression model

Consider a hypothetical study for which the goal is to investigate the effect of verbal aptitude and extraversion on sales, controlling for age, education, and prior experience.

Suppose that all five variables—verbal aptitude, extraversion, age, education, and prior experience—explain about 10% of the variance of the sales and that the three control variables—age, education, and prior experience—explain about 6% of the variance of the sales. We want to compute the power of detecting a 4% change in the R^2 after adding the two tested variables, verbal aptitude and extraversion, to the model, with 100 subjects at a 5% significance level:

```
. power rsquared .06 .1, ntested(2) ncontrol(3) n(100)

Estimated power for multiple linear regression
F test for R2 testing subset of coefficients
H0: R2_F = R2_R  versus  Ha: R2_F != R2_R

Study parameters:

        alpha =    0.0500
            N =       100
        delta =    0.0444
         R2_R =    0.0600
         R2_F =    0.1000
      R2_diff =    0.0400
     ncontrol =         3
      ntested =         2

Estimated power:

        power =    0.4431
```

The achieved power is about 44%. See [PSS-2] **power rsquared** for details.

◁

Logistic regression

The power command provides PSS computations for tests of one predictor in logistic regression models. power logistic has three syntaxes that provide PSS computations for logistic regression models with one binary predictor, two binary predictors, and arbitrary predictors. See [PSS-2] **power logistic** for an overview of the three syntaxes.

When you want to estimate sample size, power, or effect size for a coefficient test in a logistic regression model with a single binary predictor, the one binary syntax of power logistic makes it easy. You can specify the population prevalence of binary predictor X as either a proportion or odds. Model parameters can be specified directly, or computed from known quantities such as the population probability of success, or the probability of success when $X = 1$ or $X = 0$. See [PSS-2] **power logistic onebin**.

The two binary syntax of power logistic makes it easy to estimate sample size, power, or effect size for logistic regression models with two binary predictors: predictor of interest X and nuisance covariate Z. The prevalences of X and Z can be input as odds or proportions. Logistic regression parameters can be specified directly, or by using intuitive quantities such as the population probability of success and the probability of success given the values of X and Z. See [PSS-2] **power logistic twobin**.

The general syntax of power logistic is versatile, allowing you to specify predictor of interest X and up to 20 nuisance covariates from any of 11 supported distributions. Parameters from the logistic regression model may be specified directly, or provided as the probability of success when all predictors are at their means or the probability of success when $X = 0$ and all nuisance covariates are at their means. See [PSS-2] **power logistic general**.

We continue with two examples of PSS analysis with a logistic regression model.

▷ Example 7: Sample size for a coefficient test in a logistic regression model with two binary predictors

Suppose that you are working in market research, investigating the impact of a new color scheme in promotional emails to existing customers. You are designing a study that will use logistic regression to analyze the effect of the color scheme on the click-through rate.

Binary outcome Y is an indicator of whether a customer clicks on the link in the email. Half of the emails will use the new color scheme, and predictor of interest X is an indicator of whether the new scheme was used in the email. Approximately 65% of customers check their email using a mobile device, and we will include mobile device usage as binary nuisance covariate Z.

Previous email campaigns using the old color scheme have averaged a 5% click-through rate on mobile devices and a 7% click-through rate otherwise. We use the two binary syntax of power logistic to calculate the sample size needed to detect an odds ratio of 1.2 for the effect of the new color scheme, using a 5%-level test with 80% power. We omit the alpha(0.05) and power(0.8) options because the specified values are their defaults, but we specify the target odds ratio as the argument and the proportions of X and Z in the px() and pz() options. We also specify the click-through rate with the old color scheme for mobile and desktop devices using the pycond01() and pycond00() options, which correspond to the conditional probability of Y when X equals zero and Z equals one ($\Pr(Y = 1|X = 0, Z = 1)$) and the conditional probability of Y when X and Z are both equal to zero ($\Pr(Y = 1|X = 0, Z = 0)$).

```
. power logistic 1.2, px(0.5) pz(0.65) pycond01(0.05) pycond00(0.07)
Estimated sample size for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR = 1  versus  Ha: OR != 1
Study parameters:
        alpha =    0.0500
        power =    0.8000
           px =    0.5000
           pz =    0.6500
      oratiox =    1.2000
     pycond01 =    0.0500
     pycond00 =    0.0700
Estimated sample size:
            N =    16,226
```

The required sample size is 16,226. See [PSS-2] **power logistic twobin** for details.

◁

▷ Example 8: Power for a coefficient test in a logistic regression model with multiple nuisance covariates

Continuing with example 7, we plan to send 16,226 marketing emails, but we have one more piece of information about our customers: the number of purchases they made in the past year. The purchase count follows a Poisson distribution with a mean of 3 and an odds ratio of 1.3; we will include it in our logistic regression model as nuisance covariate Z_2. We want to compute the power of detecting the odds ratio of 1.2 for the effect of the new color scheme, but for our new logistic regression model with both covariates.

To include Z_2 in our calculations, we must use the general syntax of `power logistic`, which has different options than the two binary syntax. We now specify predictor of interest X using the `x()` option, and the nuisance covariates using the `z1()` and `z2()` options. Within these options, the `distribution()` suboption specifies the distribution of the predictor, and the `oratio()` suboption specifies the odds ratio. Instead of `pycond01()` and `pycond00()`, we now specify `pycondx0()`, the click-through rate when $X = 0$ and both Z variables are at their means, which is 5.63%. We also specify the number of emails we plan to send in the `n()` option.

```
. power logistic, n(16226) x(distribution(bernoulli 0.5) oratio(1.2))
> z1(distribution(bernoulli 0.65) oratio(0.7))
> z2(distribution(poisson 3) oratio(1.3)) pycondx0(0.0563)

Estimated power for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR = 1  versus  Ha: OR != 1

Study parameters:

        alpha =    0.0500
            N =    16,226
     pycondx0 =    0.0563

Covariate of interest X: Bernoulli discretized into 2 bins

      oratiox =    1.2000
           px =    0.5000

Nuisance covariate Z1: Bernoulli discretized into 2 bins

     oratioz1 =    0.7000
          pz1 =    0.6500

Nuisance covariate Z2: Poisson discretized into 2500 bins

     oratioz2 =    1.3000
          mz2 =    3.0000

Estimated power:

        power =    0.8990
```

Including additional information about our customers in the logistic regression model will increase the power to nearly 90%. See [PSS-2] **power logistic general** for details.

◁

Contingency tables

The `power` command provides PSS computations for three types of analyses of contingency tables.

`power cmh` performs PSS analysis for a Cochran–Mantel–Haenszel (CMH) test of association in stratified 2×2 tables. The command accommodates unbalanced stratum sizes and unbalanced group sizes within each stratum. See [PSS-2] **power cmh**.

`power mcc` performs PSS analysis for a test of association between a risk factor and a disease in $1{:}M$ matched case–control studies. See [PSS-2] **power mcc**.

`power trend` performs PSS analysis for a test of a linear trend in a probability of response in $J \times 2$ tables, also known as a Cochran–Armitage test. It accommodates unbalanced designs and unequally spaced exposure levels (doses). With equally spaced exposure levels, a continuity correction is available. See [PSS-2] **power trend**.

All methods compute sample size given power and effect size and power given sample size and effect size. `power cmh` and `power mcc` also compute effect size given power and sample size.

For sample-size determination, the reported sample sizes are rounded up to the nearest integer. This ensures that the corresponding actual power is at least as large as the specified power. You can specify the `nfractional` option to obtain the corresponding fractional sample sizes, or you can recompute the actual power using the reported rounded values; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for details.

Below we show a quick example of PSS analysis for a Cochran–Armitage test by using `power trend`; see [PSS-2] **power trend** for more examples.

▷ Example 9: Sample size for a Cochran–Armitage trend test

Consider a study investigating the effectiveness of a new topical antibiotic for the treatment of skin infections.

Suppose that in previous studies of the treatment, we observed the following proportions of successfully treated cases at different doses. We may hypothesize that these represent the probability of a successful treatment for each dose.

Doses/day	Proportion successes
1	0.80
2	0.85
3	0.90

We wish to determine the minimum sample size required for a clinical trial designed to detect a dose–response trend with 80% power using a two-sided 5%-level Cochran–Armitage test.

To compute the required sample size, we specify the values 0.80, 0.85, and 0.90 as the alternative success probabilities for each of the three doses after the command name. We omit the `alpha(0.05)` and `power(0.8)` options because the specified values are their defaults.

```
. power trend .80 .85 .90
note: exposure levels are assumed to be equally spaced.

Performing iteration ...

Estimated sample size for a trend test
Cochran-Armitage trend test
H0: b = 0  versus  Ha: b != 0;  logit(p) = a + b*x

Study parameters:

        alpha =    0.0500
        power =    0.8000
          N_g =         3
           p1 =    0.8000
           p2 =    0.8500
           p3 =    0.9000

Estimated sample sizes:

            N =       597
  N per group =       199
```

A total sample of 597 individuals, 199 individuals per group, must be obtained to detect a linear trend in probability of a successful treatment with 80% power using a two-sided 5%-level Cochran–Armitage test.

Suppose that we can recruit only 300 subjects. We can check how such a reduction in sample size affects the power. To compute power, we specify the alternative group probabilities, as before, and the total sample size in the `n()` option.

```
. power trend .80 .85 .90, n(300)
note: exposure levels are assumed to be equally spaced.

Estimated power for a trend test
Cochran-Armitage trend test
H0: b = 0  versus  Ha: b != 0;  logit(p) = a + b*x

Study parameters:

        alpha =    0.0500
            N =       300
  N per group =       100
          N_g =         3
           p1 =    0.8000
           p2 =    0.8500
           p3 =    0.9000

Estimated power:

        power =    0.5082
```

With a sample of 300 subjects in this study, the power to detect a linear trend in probabilities decreases dramatically from 0.8 to 0.5, which is unacceptably low for practical purposes.

◁

Survival analysis

The `power` command provides PSS computations for survival analysis comparing two survivor functions using the log-rank test or the exponential test, as well as for more general survival analysis investigating the effect of a single covariate in a Cox proportional hazards regression model, possibly in the presence of other covariates. It provides the estimate of the number of events required to be observed (or the expected number of events) in a study. The minimal effect size (minimal detectable difference, expressed as the hazard ratio or the log hazard-ratio) may also be obtained for the log-rank test and for the Wald test on a single coefficient from the Cox model.

`power cox` provides estimates of sample size, power, or the minimal detectable value of the coefficient when an effect of a single covariate on subject survival is to be explored using Cox proportional hazards regression. It is assumed that the effect is to be tested using the partial likelihood from the Cox model (for example, score or Wald test) on the coefficient of the covariate of interest. See [PSS-2] **power cox**.

`power exponential` reports estimates of sample size or power when the disparity in the two exponential survivor functions is to be tested using the exponential test, the parametric test comparing the two exponential hazard rates. In particular, we refer to the (exponential) hazard-difference test as the exponential test for the difference between hazards and the (exponential) log hazard-ratio test as the exponential test for the log of the hazard ratio or, equivalently, for the difference between log hazards. See [PSS-2] **power exponential**.

`power logrank` reports estimates of sample size, power, or minimal detectable value of the hazard ratio (or log hazard-ratio) in the case when the two survivor functions are to be compared using the log-rank test. The only requirement about the distribution of the survivor functions is that the two survivor functions must satisfy the proportional-hazards assumption. See [PSS-2] **power logrank**.

For sample-size and power computations, the default effect size corresponds to a value of the hazard ratio of 0.5 and may be changed by specifying the `hratio()` option. The hazard ratio is defined as a ratio of hazards of the experimental group to the control group (or the less favorable of the two groups). Other ways of specifying the effect size are available, and these are particular to each subcommand.

By default, all subcommands assume a type I study, that is, perform computations for uncensored survival data. The censoring information may be taken into account by specifying the appropriate arguments or options. See [PSS-2] **power cox**, [PSS-2] **power logrank**, and [PSS-2] **power exponential** for details.

▷ Example 10: Sample size for the test of the effect of a covariate in the Cox model

Consider a hypothetical study for which the goal is to investigate the effect of the expression of one gene on subject survival with the Cox proportional hazards regression model. Suppose that the Wald test is to be used to test the coefficient on the gene after fitting the Cox model. Gene expression values measure the level of activity of the gene. Consider the scenario described in Simon, Radmacher, and Dobbin (2002) in which the hazard ratio of 3 associated with a one-unit change in the $\log_2$ intensity of a gene (or, respectively, with a twofold change in gene expression level) is desired to be detected with 95% power using a two-sided, 0.001-level test. The estimate of the standard deviation of the $\log_2$-intensity level of the gene over the entire set of samples is assumed to be 0.75.

```
. power cox, hratio(3) sd(0.75) power(0.95) alpha(0.001)

Estimated sample size for Cox PH regression
Wald test
H0: beta1 = 0  versus  Ha: beta1 != 0

Study parameters:

        alpha =    0.0010
        power =    0.9500
        delta =    1.0986  (coefficient)
       hratio =    3.0000
           sd =    0.7500

Censoring:

         Pr_E =    1.0000

Estimated number of events and sample size:

            E =        36
            N =        36
```

Provided that all subjects experience an event in this study, a total of 36 events is required to be observed in the study to ensure the specified power.

See [PSS-2] **power cox** for more details.

◁

▷ Example 11: Sample size for two-sample test of exponential survivor functions

Consider an example from Lachin (2011, 490) of a study comparing two therapies, the combination of a new therapy with the standard one versus the standard alone, in the treatment of lupus nephritis patients. From previous studies, the survivor function of the control group treated with the standard therapy was log linear with a constant yearly hazard rate of 0.3. The number of events (failures) required to ensure 90% power to detect a 50% risk reduction, $\Delta = 0.5$, (or, respectively, the log hazard-ratio of $\ln(0.5) = -0.6931$) with a one-sided test at a 0.05 significance level was obtained to be 72 under equal-group allocation. In the absence of censoring, Lachin (2011) determined that a total of 72 subjects (36 per group) would have to be recruited to the study. To obtain this same estimate with

power exponential, we supply the control hazard rate 0.3 as an argument and specify the power(0.9), onesided, and loghazard options to request 90% power, a one-sided test, and sample-size determination for the exponential log hazard-ratio test (or test for the log-hazard difference), respectively.

```
. power exponential 0.3, power(0.9) onesided loghazard
note: input parameters are hazard rates.

Estimated sample sizes for two-sample comparison of survivor functions
Exponential test, log hazard-ratio, conditional
H0: ln(HR) = 0  versus  Ha: ln(HR) < 0

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =   -0.6931  (log hazard-ratio)

Survival information:

           h1 =    0.3000
           h2 =    0.1500
       hratio =    0.5000

Estimated sample sizes:

            N =        72
  N per group =        36
```

Further, the study was planned to continue for 6 years with a recruitment period of 4 years. Subjects who did not experience an event by the end of 6 years were censored. For this fixed-duration study with uniform entry (recruitment), the estimate of the sample size increases from 72 to 128. We specify the length of the accrual and the follow-up periods in the aperiod() and fperiod() options, respectively. We also request to display the expected number of events by using the show option.

```
. power exponential 0.3, power(0.9) onesided loghazard aperiod(4) fperiod(2) show
note: input parameters are hazard rates.

Estimated sample sizes for two-sample comparison of survivor functions
Exponential test, log hazard-ratio, conditional
H0: ln(HR) = 0  versus  Ha: ln(HR) < 0

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =   -0.6931  (log hazard-ratio)

Accrual and follow-up information:

     duration =    6.0000
    follow-up =    2.0000
      accrual =    4.0000  (uniform)

Survival information:

           h1 =    0.3000
           h2 =    0.1500
       hratio =    0.5000

Estimated expected number of events:

         E|Ha =        72      E|H0 =        74
        E1|Ha =        44     E1|H0 =        37
        E2|Ha =        28     E2|H0 =        37

Estimated sample sizes:

            N =       128
  N per group =        64
```

Under the alternative hypothesis of H_a: $\ln(\Delta) = -0.6931$, where $\ln(\Delta)$ denotes the log hazard-ratio of the experimental group to the control group, we expect to observe 44 events in the control group and 28 events in the experimental group. A total of 128 subjects (64 per group) is required to be enrolled into the study to observe an expected total of 72 events under the alternative.

See [PSS-2] **power exponential** for more examples.

◁

▷ Example 12: Sample size for the log-rank test

Consider an example from Machin and Campbell (2005) of a study comparing two forms of surgical resection for patients with gastric cancer. From a prestudy survey, the baseline 5-year survival rate was expected to be 20% and an anticipated increase in survival in the experimental group expressed as a hazard ratio of 0.6667 (corresponding to a 5-year survival rate of approximately 34%) was desired to be detected with 90% power using a two-sided, 0.05 level, log-rank test under 1:1 randomization. To obtain the estimate of the sample size for this study, we use `power logrank` with survival proportion in the control group 0.2 supplied as an argument, the `hratio(0.6667)` option to request a hazard ratio of 0.6667, and the `power(0.9)` option to request 90% power.

```
. power logrank 0.2, hratio(0.6667) power(0.9)

Estimated sample sizes for two-sample comparison of survivor functions
Log-rank test, Freedman method
H0: HR = 1  versus  Ha: HR != 1

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =    0.6667  (hazard ratio)
       hratio =    0.6667

Censoring:

           s1 =    0.2000
           s2 =    0.3420
         Pr_E =    0.7290

Estimated number of events and sample sizes:

            E =       263
            N =       362
  N per group =       181
```

From the output, 263 events (failures) are required to be observed in this study to ensure 90% power to detect a hazard ratio of 0.6667 by using the log-rank test. The respective estimate of the total number of subjects required to observe 263 events in a 5-year study is 362 with 181 subjects per surgical group. Our estimate, 181, of each group's sample size is close to the manually computed estimate of 180 from Machin and Campbell (2005). This is a fixed-duration study in which 20% of subjects were expected to survive (be censored) by the end of the study.

See [PSS-2] **power logrank** for more detailed examples and other available methods of sample-size computation for this type of analysis.

◁

Cluster randomized designs

So far, all power analyses have assumed simple randomization of the subjects in the study. We could instead have a cluster randomized design (CRD). In a CRD, groups of subjects or clusters are randomized instead of individual subjects, so the sample size is determined by the number of clusters and the cluster size. The sample-size determination thus consists of the determination of the number of clusters given cluster size or the determination of cluster size given the number of clusters.

`power` supports CRDs with methods `onemean`, `oneproportion`, `twomeans`, `twoproportions`, and `logrank`. To request computations for a CRD, you specify the `cluster` option, include the number of clusters `k()` with one-sample methods and `k1()` or `k2()` with two-sample methods, or include the cluster size `m()`, `m1()`, or `m2()`. In addition to power and effect size, all methods compute the numbers of clusters given the cluster sizes or the cluster sizes given the numbers of clusters. Two-sample methods can also compute the number of clusters or cluster size of one group given that of the other group.

A CRD requires more subjects to obtain the same statistical power as the corresponding individual-level design because the subjects within a cluster are correlated. Power and sample-size computations in a CRD account for this intraclass correlation. All `power, cluster` methods use the default intraclass correlation of 0.5, but you may change this by using the `rho()` option.

By default, all methods assume equal cluster sizes or equal numbers of subjects in each cluster. In practice, cluster sizes often vary, in which case you may provide the coefficient of variation of the cluster sizes in the `cvcluster()` option to account for varying cluster sizes.

Below we show a short example of PSS analysis for `power onemean` for the one-sample case and `power twoproportions` for the two-sample case. See [PSS-2] **power onemean, cluster**, [PSS-2] **power oneproportion, cluster**, [PSS-2] **power twomeans, cluster**, [PSS-2] **power twoproportions, cluster**, and [PSS-2] **power logrank, cluster** for more examples.

▷ Example 13: Number of clusters for a one-sample mean test in a CRD, specifying cluster size

Consider an example that studies the effectiveness of coaching programs in improving the verbal part of SAT scores. Previous studies found that students retaking the SAT exams without any coaching program improve their scores by 15 points on average with a standard deviation of about 40 points. The population standard deviation is assumed to be 40. We assume that students are sampled from a set of classes and that the scores of students from the same class are correlated. We plan on sampling 10 students from each class and assume that the intraclass correlation is 0.3.

A new coaching program claims to improve average SAT scores by 40 points. The changes in scores are assumed to be approximately normally distributed. The parameter of interest in this example is the mean change in the test scores. To test the claim, investigators wish to conduct another study and compute the number of classes that is required to detect a mean change in scores of 40 points with 80% power using a 5%-level two-sided test:

```
. power onemean 15 40, m(10) sd(40) rho(0.3)

Performing iteration ...

Estimated number of clusters for a one-sample mean test
Cluster randomized design, z test
H0: m = m0  versus  Ha: m != m0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.3249
           m0 =   15.0000
           ma =   40.0000
           sd =   40.0000

Cluster design:

            M =        10
          rho =    0.3000

Estimated number of clusters and sample size:

            K =         8
            N =        80
```

We find that 8 classes with 10 students per class, a total of 80 students, are required to detect a shift of 40 points in average SAT scores given the standard deviation of 40 points with 80% power using a 5%-level two-sided test. See [PSS-2] **power onemean, cluster** for more information.

◁

▷ Example 14: Numbers of clusters for a two-sample proportions test in a CRD, specifying cluster sizes

Consider a study investigating the effectiveness of a program to promote after-school activities in increasing the rate of students participating in the after-school club. Schools that are involved in the study will be randomly assigned either to the experimental group that participates in the program or to the control group that does not. A researcher plans to recruit 50 students from each school and assumes an intraclass correlation of 0.2. The researcher wants to be able to detect an increase of 0.2 in the anticipated control-group rate of 0.4, which corresponds to the experimental-group rate of 0.6.

To compute the number of schools in each group required to detect the desired rate with 80% power using a 5%-level two-sided test, we type

```
. power twoproportions 0.4 0.6, m1(50) m2(50) rho(0.2)
Performing iteration ...
Estimated numbers of clusters for a two-sample proportions test
Cluster randomized design, Pearson's chi-squared test
H0: p2 = p1  versus  Ha: p2 != p1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.2000  (difference)
           p1 =    0.4000
           p2 =    0.6000
Cluster design:
           M1 =        50
           M2 =        50
          rho =    0.2000
Estimated numbers of clusters and sample sizes:
           K1 =        21
           K2 =        21
           N1 =     1,050
           N2 =     1,050
```

We find that for 50 students, 21 schools per group, with a total of 1,050 students per group, are required to detect a 0.2 difference in participation rates in the after-school club with 80% power using a 5%-level two-sided test. See [PSS-2] **power twoproportions, cluster** for more information.

◁

Tables of results

When `power` is used to perform computations for a single set of study parameters, the results can be displayed either as text or in a table. The default is to display results as text:

```
. power onemean 0 0.2
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.2000
           m0 =    0.0000
           ma =    0.2000
           sd =    1.0000
Estimated sample size:
            N =       199
```

You can specify the `table` option to display results in a table:

```
. power onemean 0 0.2, table
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0
```

alpha	power	N	delta	m0	ma	sd
.05	.8	199	.2	0	.2	1

For multiple sets of study parameters, when command arguments or options contain number lists, the results are automatically displayed in a table:

```
. power onemean 0 (0.2 0.5)
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0
```

alpha	power	N	delta	m0	ma	sd
.05	.8	199	.2	0	.2	1
.05	.8	34	.5	0	.5	1

In this example, we specified two values for the second argument.

Default tables can be modified by specifying the `table()` option. For example, we can change the order in which the columns are displayed:

```
. power onemean 0 (0.2 0.5), table(alpha power N m0 ma sd delta)
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0
```

alpha	power	N	m0	ma	sd	delta
.05	.8	199	0	.2	1	.2
.05	.8	34	0	.5	1	.5

Or we can change column labels:

```
. power onemean 0 (0.2 0.5), table(, labels(N "Sample size"))
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0
```

alpha	power	Sample size	delta	m0	ma	sd
.05	.8	199	.2	0	.2	1
.05	.8	34	.5	0	.5	1

Or we can select which columns we want to display:

```
. power onemean 0 (0.2 0.5), table(alpha beta N m0 ma sd)
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0
```

alpha	beta	N	m0	ma	sd
.05	.2	199	0	.2	1
.05	.2	34	0	.5	1

For more examples, see [PSS-2] **power, table**.

Power curves

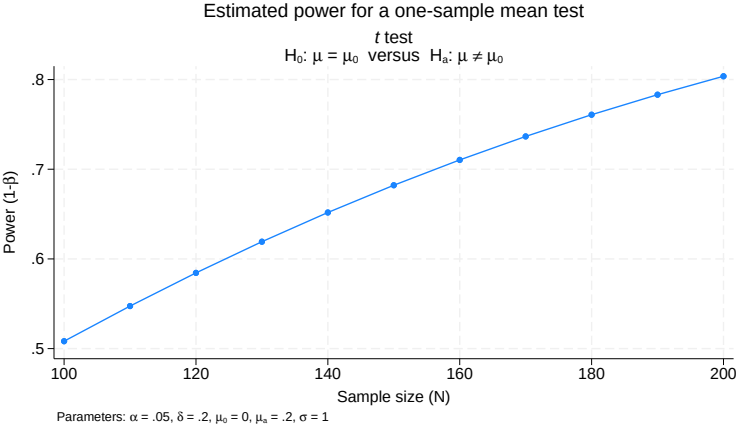
During the planning stage of a study, it is difficult to decide on a number of subjects to be enrolled in a study on the basis of only one set of study parameters. It is common to investigate the effect of various study scenarios on power. Power curves, or plots of estimated power versus one of the study parameters, are commonly used for this purpose.

The `power` command provides the `graph` and `graph()` options to produce power curves.

More precisely, when `graph` is specified, the estimated parameter such as power or sample size is plotted on the y axis, and the varying parameter is plotted on the x axis.

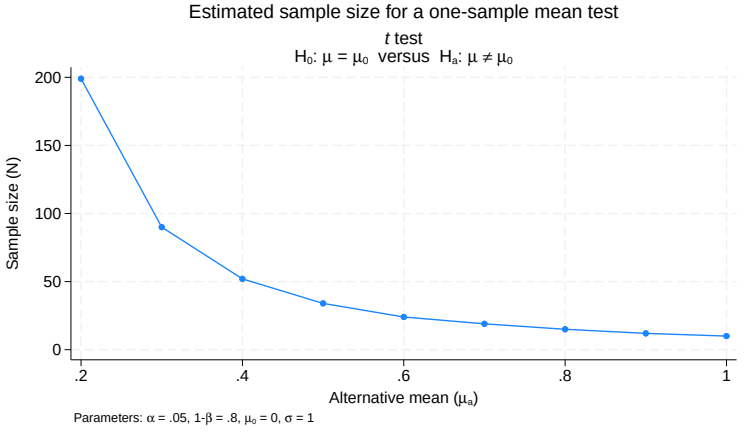
For example, we compute power and plot it as a function of sample size for a range of sample-size values between 100 and 200 with a step size of 10:

```
. power onemean 0 0.2, n(100(10)200) graph
```



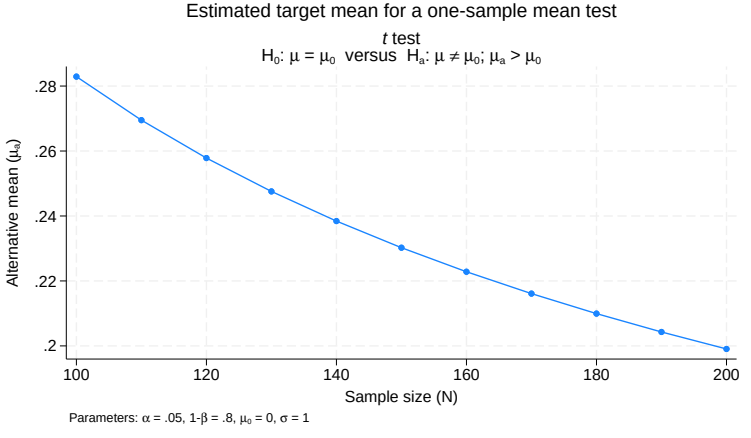
Or we can compute sample size and plot it as a function of the alternative mean when the mean ranges between 0.2 and 1 with a step size of 0.1:

```
. power onemean 0 (0.2(0.1)1), graph
```



Or we can compute the alternative mean for a given power of 80% and a range of sample-size values between 100 and 200 with a step size of 10, and plot it against the sample size:

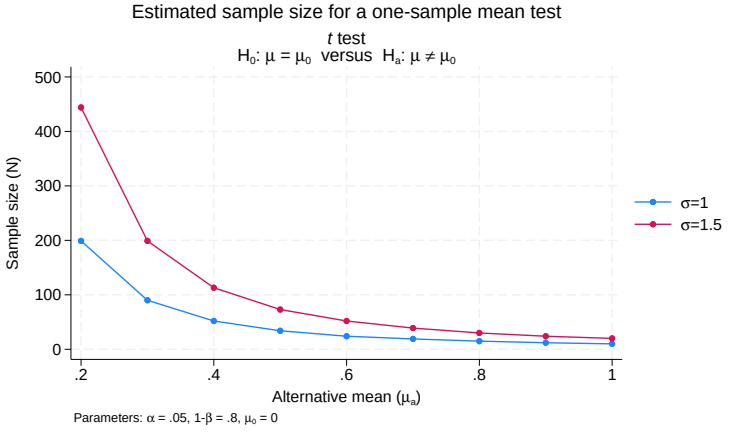
```
. power onemean 0, n(100(10)200) power(0.8) graph
```



The above graphs are the default graphs produced by `power, graph`. Similarly to tabular output, you can customize graphical output by specifying the `graph()` option.

For example, we modify the look of the default graph by using the `graph(nosimplelabels legend(title(""))` option. `nosimplelabels` requests that the graph legend include the column symbol and an equal sign; `legend(title(""))` requests that the legend not have a title.

```
. power onemean 0 (0.2(0.1)1), sd(1 1.5) graph(nosimplelabels legend(title("")))
```



By default, when a graph is produced, the tabular output is suppressed. You can specify the `table` option if you also want to see results in a table.

For more examples, see [PSS-2] **power, graph**.

Add your own methods to power

The `power` command provides many built-in methods, but sometimes, you may want to compute sample size or power yourself. For example, you may need to do this by simulation, or you may want to use a method that is not available in any software package. `power` makes it easy for you to add your own method. All you need to do is to write a program that computes sample size, power, or effect size, and the `power` command will do the rest for you. It will deal with the support of multiple values in options and with automatic generation of graphs and tables of results.

Suppose you want to add the method called `mymethod` to the `power` command. Just follow these three steps:

1. Create a program that computes sample size, power, or effect size and follows `power`'s naming convention: `power_cmd_mymethod`.
2. Store results following `power`'s simple naming conventions for results. For example, store the value of power in `r(power)`, the value of sample size in `r(N)`, and so on.
3. Place your program `power_cmd_mymethod` where Stata can find it.

To show how easy this all is, let's write a program to compute power for a one-sample z test given sample size, standardized difference, and significance level. For simplicity, we assume a two-sided test.

We will call our new method `myztest`.

```
program power_cmd_myztest, rclass
        version 19.5      // (or version 19 if you do not have StataNow)
                                  //  parse options
        syntax , n(integer)       /// sample size
                 STDDiff(real)    /// standardized diff.
                 Alpha(string)    /// significance level

                                  //  compute power
        tempname power
        scalar `power' = normal(`stddiff'*sqrt(`n') - ///
                         invnormal(1-`alpha'/2))

                                  // return results
        return scalar power   = `power'
        return scalar N       = `n'
        return scalar alpha   = `alpha'
        return scalar stddiff = `stddiff'
end
```

The computation in this program takes only one line, but it could be as complicated as we like. It could even involve simulation to compute the power.

With our program in hand, we can type

```
. power myztest, n(20) stddiff(1) alpha(.05)
```

`power` will find our program, supply it with the `n(20)`, `stddiff(1)`, and `alpha(.05)` options, and use its returned results to produce

```
. power myztest, n(20) stddiff(1) alpha(.05)
Estimated power
Two-sided test
```

alpha	power	N
.05	.994	20

That was not too impressive. Our program did all the work.

What if we supplied `power` with a list of sample sizes?

```
. power myztest, n(10 15 20 25) stddiff(1)
Estimated power
Two-sided test
```

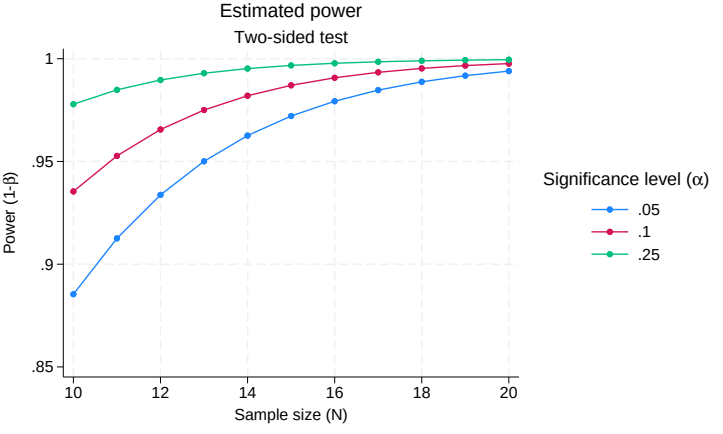
alpha	power	N
.05	.8854	10
.05	.9721	15
.05	.994	20
.05	.9988	25

`power` has taken our list of sample sizes and computed powers for all of them—even though our program could only compute a single power!

Moreover, we can use `power`'s standard `table()` option to control exactly how that table looks; see *Table of results* for more examples of tables. `power` also has hooks that let our program determine how the columns are labeled and how the table appears.

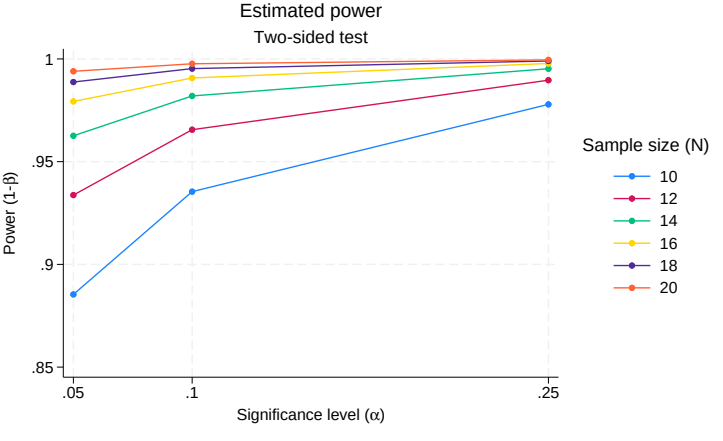
We can supply both sample sizes and significance levels and request a graph instead of a table:

```
. power myztest, n(10(1)20) alpha(.05 .10 .25) stddiff(1) graph
```



We can even request that the graph show α on the x axis with separate plots for each sample size.

```
. power myztest, n(10(2)20) alpha(.05 .10 .25) stddiff(1) graph(xdim(alpha))
```



All this may make it worth writing more complicated programs to compute power for more complicated tests and comparisons.

See [PSS-2] ***power usermethod*** for more examples.

Stored results

`power` stores the following in `r()`:

Scalars

`r(alpha)`	significance level
`r(power)`	power
`r(beta)`	probability of a type II error
`r(delta)`	effect size
`r(N)`	total sample size

r(N_a)	actual sample size
r(N1)	sample size of the control group
r(N2)	sample size of the experimental group
r(nratio)	ratio of sample sizes, N2/N1
r(nratio_a)	actual ratio of sample sizes
r(nfractional)	1 if nfractional is specified, 0 otherwise
r(onesided)	1 for a one-sided test, 0 otherwise
r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise
r(init)	initial value of the estimated parameter
r(maxiter)	maximum number of iterations
r(iter)	number of iterations performed
r(tolerance)	requested parameter tolerance
r(deltax)	final parameter tolerance achieved
r(ftolerance)	requested distance of the objective function from zero
r(function)	final distance of the objective function from zero
r(converged)	1 if iteration algorithm converged, 0 otherwise

Macros

r(type)	test
r(method)	the name of the specified method
r(direction)	upper or lower
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results

Also see *Stored results* in the method-specific manual entries for the full list of stored results.

Methods and formulas

By default, the power command rounds sample sizes to integers and uses integer values in the computations. To ensure conservative results, the command rounds down the input sample sizes and rounds up the output sample sizes. See *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for details.

Some of power's methods require iteration. For example, the sample size for a two-sided test is usually solved iteratively from the two-sided power equation. Most methods use Mata function solvenl() and its Newton's method described in *Newton-type methods* in [M-5] **solvenl( )** to solve a nonlinear power equation. Other methods use a bisection method to find a root of a nonlinear power equation.

See *Methods and formulas* in the method-specific manual entries for details.

References

Batistatou, E., C. Roberts, and S. Roberts. 2014. Sample size and power calculations for trials and quasi-experimental studies with clustering. *Stata Journal* 14: 159–175.

Cattaneo, M. D., R. Titiunik, and G. Vazquez-Bare. 2019. Power calculations for regression-discontinuity designs. *Stata Journal* 19: 210–245.

Earnest, A. 2017. *Essentials of a Successful Biostatistical Collaboration*. Boca Raton, FL: CRC Press.

Gallis, J. A., X. Wang, P. J. Rathouz, J. S. Preisser, F. Li, and E. L. Turner. 2022. power swgee: GEE-based power calculations in stepped wedge cluster randomized trials. *Stata Journal* 22: 811–841.

Huber, C. 2019a. Calculating power using Monte Carlo simulations, part 1: The basics. *The Stata Blog: Not Elsewhere Classified*. https://blog.stata.com/2019/01/10/calculating-power-using-monte-carlo-simulations-part-1-the-basics/.

———. 2019b. Calculating power using Monte Carlo simulations, part 2: Running your simulation using power. *The Stata Blog: Not Elsewhere Classified*. https://blog.stata.com/2019/01/29/calculating-power-using-monte-carlo-simulations-part-2-running-your-simulation-using-power/.

Lachin, J. M. 2011. *Biostatistical Methods: The Assessment of Relative Risks*. 2nd ed. Hoboken, NJ: Wiley.

Linden, A. 2024. Stata tip 154: Computing power and sample size for prospective diagnostic accuracy studies using Stata's official power commands. *Stata Journal* 24: 174–181.

Ma, X., and Y. B. Cheung. 2022. crtrest: A command for ratio estimators of intervention effects on event rates in cluster randomized trials. *Stata Journal* 22: 908–923.

Machin, D. 2004. On the evolution of statistical methods as applied to clinical trials. *Journal of Internal Medicine* 255: 521–528. https://doi.org/10.1111/j.1365-2796.2004.01319.x.

Machin, D., and M. J. Campbell. 2005. *Design of Studies for Medical Research*. Chichester, UK: Wiley.

Simon, R., R. D. Radmacher, and K. Dobbin. 2002. Design of studies using DNA microarrays. *Genetic Epidemiology* 23: 21–36. https://doi.org/10.1002/gepi.202.

Thompson, J., C. Davey, R. J. Hayes, J. Hargreaves, and K. Fielding. 2019. Permutation tests for stepped-wedge cluster-randomized trials. *Stata Journal* 19: 803–819.

Wittes, J. 2002. Sample size calculations for randomized control trials. *Epidemiologic Reviews* 24: 39–53. https://doi.org/10.1093/epirev/24.1.39.

Also see

[PSS-2] **Intro (power)** — Introduction to power and sample-size analysis for hypothesis tests

[PSS-5] **Glossary**

[ADAPT] **GSD intro** — Introduction to group sequential designs

power usermethod — Add your own methods to the power command

Description Syntax Remarks and examples References Also see

Description

The power command allows you to add your own methods to power and produce tables and graphs of results automatically.

Syntax

Compute sample size

power *usermethod* ... [, power(*numlist*) *poweropts useropts*]

Compute power

power *usermethod* ..., *nspec* [*poweropts useropts*]

Compute effect size

power *usermethod* ..., *nspec* power(*numlist*) [*poweropts useropts*]

usermethod is the name of the method you would like to add to the power command. When naming your power methods, you should follow the same convention as for naming the programs you add to Stata—do not pick "nice" names that may later be used by Stata's official methods. The length of *usermethod* may not exceed 16 characters.

useropts are the options supported by your method *usermethod*.

nspec contains n(*numlist*) for a one-sample test or any of the sample-size options of *poweropts* such as n1(*numlist*) and n2(*numlist*) for a two-sample test.

collect is allowed; see **[U] 11.1.10 Prefix commands**.

Remarks and examples

Adding your own methods to power is easy. Suppose you want to add a method called mymethod to power. Simply

1. write an r-class program called power_cmd_mymethod that computes power, sample size, or effect size and follows power's convention for naming common options and storing results; and
2. place the program where Stata can find it.

You are done. You can now use mymethod within power like any other official power method.

Remarks are presented under the following headings:

A quick example

Before we discuss the technical details in the following sections, let's try an example. Let's write a program to compute power for a one-sample z test given sample size, standardized difference, and significance level. For simplicity, we assume a two-sided test. We will call our new method `myztest`.

We create an ado-file called `power_cmd_myztest.ado` that contains the following Stata program:

```
// evaluator
program power_cmd_myztest, rclass
        version 19.5      // (or version 19 if you do not have StataNow)
                        /* parse options */
        syntax, n(integer)       /// sample size
                 STDDiff(real)   /// standardized difference
                 Alpha(string)   /// significance level

/* compute power */
tempname power
scalar `power' = normal(`stddiff'*sqrt(`n') - invnormal(1-`alpha'/2))

/* return results */
return scalar power   = `power'
return scalar N       = `n'
return scalar alpha   = `alpha'
return scalar stddiff = `stddiff'
end
```

Our ado-program consists of three sections: the `syntax` command for parsing options, the power computation, and stored or returned results. The three sections work as follows:

The `power_cmd_myztest` program has two of `power`'s common options, `n()` for sample size and `alpha()` for significance level, and it has its own option, `stddiff()`, to specify a standardized difference.

After the options are parsed, the power is computed and stored in a temporary scalar `` `power' ``.

Finally, the resulting power and other results are stored in return scalars. Following `power`'s convention for naming commonly returned results, the computed power is stored in `r(power)`, the sample size in `r(N)`, and the significance level in `r(alpha)`. The program additionally stores the standardized difference in `r(stddiff)`.

We can now use `myztest` within `power` as we would any other existing method of `power`:

```
. power myztest, alpha(0.05) n(10) stddiff(0.25)
Estimated power
Two-sided test
```

alpha	power	N
.05	.1211	10

We can compute results for multiple sample sizes by specifying multiple values in the `n()` option. Note that our `power_cmd_myztest` program accepts only one value at a time in `n()`. When a numlist is specified in the `power` command's `n()` option, `power` automatically handles that *numlist* for us.

```
. power myztest, alpha(0.05) n(10 20) stddiff(0.25)
Estimated power
Two-sided test
```

alpha	power	N
.05	.1211	10
.05	.1999	20

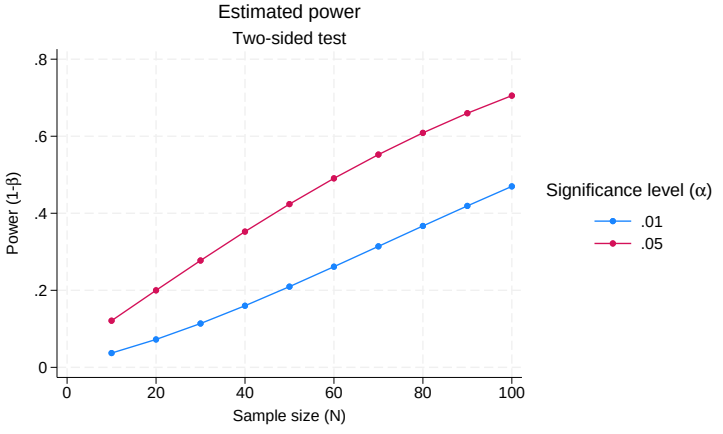
We can also compute results for multiple sample sizes and significance levels without any additional effort on our part:

```
. power myztest, alpha(0.01 0.05) n(10 20) stddiff(0.25)
Estimated power
Two-sided test
```

alpha	power	N
.01	.03711	10
.01	.07245	20
.05	.1211	10
.05	.1999	20

We can even produce a graph by merely specifying the `graph` option:

```
. power myztest, alpha(0.01 0.05) n(10(10)100) stddiff(0.25) graph
```



The above is just a simple example. Your program can be as complicated as you would like: you can even use simulations to compute your results. You can also customize your tables and graphs with a little extra effort.

Steps for adding a new method to the power command

Suppose you want to add your own method, *usermethod*, to the `power` command. Here is an outline of the steps to follow:

1. Create the evaluator, an r-class program called `power_cmd_`*usermethod* and defined by the ado-file `power_cmd_`*usermethod*`.ado`, that performs power and sample-size computations and follows `power`'s convention for naming options and storing results.
2. Optionally, create an initializer, an s-class program called `power_cmd_`*usermethod*`_init` and defined by the ado-file `power_cmd_`*usermethod*`_init.ado`, that specifies information about table columns, options that may allow a numlist, and so on.
3. Optionally, create a parser, a program called `power_cmd_`*usermethod*`_parse` and defined by the ado-file `power_cmd_`*usermethod*`_parse.ado`, that checks the syntax of user-specific options, *useropts*.
4. Place all of your programs where Stata can find them.

You can now use your *usermethod* with `power`:

```
. power usermethod ...
```

You may also use programs within `power` that are not defined by an ado-file (that is, they were defined in a do-file or by hand).

Convention for naming options and storing results

For the `power` command to automatically recognize its common options, you must ensure that you follow `power`'s naming convention for these options in your program. For example, `power` specifies the significance level in the `alpha()` option with minimum abbreviation of `a()`. You need to ensure that you use the same option with the same abbreviation in your evaluator to specify the significance level. The same applies to all of `power`'s common options described in [PSS-2] **power**.

You can specify additional method-specific options, but `power` will not know about them unless you make it aware of them; see *Allowing multiple values in method-specific options* for details.

To produce tables and graphs of results, you must ensure that your evaluator follows `power`'s convention for storing results. `power`'s commonly stored results are described in *Stored results* of [PSS-2] **power**. For example, the value for power should be stored in the scalar `r(power)`, the value for a total sample size in the scalar `r(N)`, the value for a significance level in `r(alpha)`, and so on.

You can also store additional method-specific results, but `power` will not know about them unless you make it aware of them; see *Customizing default tables* for details.

Allowing multiple values in method-specific options

By default, the `power` command accepts multiple values only within its common options. If you want to allow multiple values in the method-specific options *useropts*, you need to let `power` know about them. This is done via the initializer.

To allow the specification of multiple values, or a numlist, in method-specific options, you need to list the names of the options with proper abbreviations in an s-class macro `s(pss_numopts)` within the `power_cmd_`*usermethod*`_init` program.

Recall our earlier example in which we added the `myztest` method, calculating the power of a two-sided one-sample z test, to `power`. We computed powers for multiple values of significance level and sample size. What if we would also like to specify multiple values of standardized differences in the `stddiff()` option of `myztest`? If we do this now, we will receive an error message,

```
. power myztest, alpha(0.05) n(10) stddiff(0.25 0.5)
option stddiff() invalid
r(198);
```

because the `stddiff()` option is not allowed to include *numlist* by the evaluator and is not one of `power`'s common options. To make `power` recognize this option as one allowing *numlist*, we need to specify this in the initializer. Following the guidelines, we create an ado-file called `power_cmd_myztest_init.ado` and specify the name of the `stddiff()` option (with the corresponding abbreviation) in the s-class macro `s(pss_numopts)` within the `power_cmd_myztest_init` program.

```
// initializer
program power_cmd_myztest_init, sclass
        version 19.5      // (or version 19 if you do not have StataNow)
        sreturn clear
        sreturn local pss_numopts "STDDiff"
end
```

We now can specify multiple standardized differences:

```
. power myztest, alpha(0.05) n(10) stddiff(0.25 0.5)
Estimated power
Two-sided test
```

alpha	power	N
.05	.1211	10
.05	.3524	10

Customizing default tables

The `power` command with user-defined methods always displays results in a table. By default, it displays columns `alpha`, `power` or `beta` (whichever is specified), and `N`, which contain the significance level, the power, and the sample size, respectively. See *Setting supported columns* and *Modifying the default table columns* for details on how to customize the default table columns.

The default column labels are the column names, and the default formats are `%7.4g` for `alpha` and `power` and `%7.0gc` for `N`. These and other settings controlling the look of the default table can be changed as described in *Modifying the look of the default table*.

You can always use the `table()` option to customize your table. However, if you want to modify how the table looks by default, you need to follow the steps described in the following sections:

Setting supported columns

The `power` command automatically supports a number of columns, such as `alpha`, `beta`, `power`, `N`, etc. The supported columns are the columns that can be accessed within `power`'s options `table()` and `graph()`.

Most of the time, you will have additional columns, *usercolnames*, which you will want `power` to support. To make `power` recognize the columns as supported columns, you must list the names of the additional columns, *usercolnames*, in an s-class macro `s(pss_colnames)` in the initializer. Columns *usercolnames* will then be added to `power`'s list of supported columns. Columns *usercolnames* will also be displayed in the default table unless `s(pss_tabcolnames)` or `s(pss_alltabcolnames)` is set.

If you want to reset `power`'s list of supported columns, that is, to specify all the supported columns manually, you should use the `s(pss_allcolnames)` macro. The supported columns will then include only the ones you listed in the macro. If you specify `s(pss_allcolnames)`, you must remember to include `power`'s main columns `N`, `power`, and `alpha` in your list. Otherwise, you may not be able to use some of `power`'s functionality, such as default graphs. If `s(pss_colnames)` is specified together with `s(pss_allcolnames)`, the former will be ignored. The specified supported columns will be automatically displayed in the default table unless `s(pss_alltabcolnames)` is set.

The values corresponding to the specified columns must be stored by the evaluator in `r()` scalars with the same names as the column names. For example, the value for column `alpha` is stored in `r(alpha)`, the value for column `power` is stored in `r(power)`, and the value for column `N` is stored in `r(N)`.

Any column not listed in `s(pss_colnames)` or `s(pss_allcolnames)` will not be available within the `power` command. For example, any reference to such a column within `power`'s options `table()` and `graph()` will result in an error.

Modifying the default table columns

By default, `power` displays the specified supported columns. If you want to display only a subset of those columns, you can use either `s(pss_tabcolnames)` or `s(pss_alltabcolnames)` to specify the columns to be displayed. You specify additional columns to be displayed in `s(pss_tabcolnames)` and a complete list of the displayed columns in `s(pss_alltabcolnames)`. If you specify `s(pss_tabcolnames)`, the displayed columns will include `alpha`, `power`, or `beta` (whichever is specified with the command), `N`, and the additional columns you specified. If you specify `s(pss_alltabcolnames)`, only the columns listed in this macro will be displayed. One situation when you may want to do this is if you want to change the order in which the columns are displayed by default. If you specify both macros, `s(pss_tabcolnames)` will be ignored. You can specify only the names of supported columns in these macros.

Modifying the look of the default table

The default table column labels are the column names. You can change this by specifying your own column labels in the `s(pss_collabels)` macro. The labels must be properly quoted if they contain spaces or quotes. The labels must be specified for all columns listed in `s(pss_colnames)` or `s(pss_allcolnames)`.

The default column formats are `%7.0gc` for sample sizes and `%7.4g` for all other columns. You can change this by specifying your own column formats in the `s(pss_colformats)` macro. The formats must be quoted and specified for all columns listed in `s(pss_colnames)` or `s(pss_allcolnames)`.

The default column widths are the widths of the default formats plus one. You can specify your own column widths in the `s(pss_colwidths)` macro. The widths must be specified for all columns listed in `s(pss_colnames)` or `s(pss_allcolnames)`.

Example continued

Continuing our `myztest` example, we want to add a column containing the specified standardized differences to the list of supported columns. The specified standardized difference is stored in `r(stddiff)` in the `myztest` evaluator, so the name of our column is `stddiff`. We specify it in `s(pss_colnames)` in our initializer as follows:

```
// initializer
program drop power_cmd_myztest_init
program power_cmd_myztest_init, sclass
        version 19.5      // (or version 19 if you do not have StataNow)
        sreturn clear
        sreturn local pss_numopts "STDDiff"
        sreturn local pss_colnames "stddiff"  // <-- new line
end
```

We rerun our command now and see that the `stddiff` column is added to the default table:

```
. power myztest, alpha(0.05) n(10) stddiff(0.25)
Estimated power
Two-sided test
```

alpha	power	N	stddiff
.05	.1211	10	.25

We can also change the default column label of the `stddiff` column to "Std. difference". Note that we can do this within `power`'s option `table()`, but if we want this label to show up automatically in the default table, we should specify it in the initializer. We specify the column label in the `s(pss_collabels)` macro.

```
// initializer
program drop power_cmd_myztest_init
program power_cmd_myztest_init, sclass
        version 19.5      // (or version 19 if you do not have StataNow)
        sreturn clear
        sreturn local pss_numopts "STDDiff"
        sreturn local pss_colnames "stddiff"
        sreturn local pss_collabels `""Std. difference""'  // <-- new line
end
```

The column containing standardized differences now has the new label

```
. power myztest, alpha(0.05) n(10) stddiff(0.25)
Estimated power
Two-sided test
```

alpha	power	N	Std. difference
.05	.1211	10	.25

Customizing default graphs

By default, `power` plots the estimated power on the y axis and the specified sample size on the x axis or the estimated sample size on the y axis and the specified power on the x axis. If `s(pss_target)` is specified, the estimated sample size is plotted against the specified target parameter. For effect-size computation, the target parameter must be specified in `s(pss_target)`, and it is plotted on the x axis against the specified sample size. See [PSS-2] **power, graph** for details about other default settings.

You can overwrite the default column labels displayed on the graph by specifying the `s(pss_colgrlabels)` macro. The specification of the graph labels is the same as the specification of table column labels.

You can also overwrite the default symbols that are used to label the results on the graph by specifying the new symbols in the macro `s(pss_colgrsymbols)`. The symbols must be specified for all columns listed in `s(pss_colnames)` or `s(pss_allcolnames)`.

Other settings

When you add your own method to `power`, the effect-size parameter is not defined. You can define it yourself by specifying the name of the column containing the values of the effect-size parameter in the `s(pss_delta)` macro. The effect-size parameter can then be accessed using the column name `delta` and will be displayed in the default table as `delta` unless the `s(pss_notabdelta)` macro is set to `notabdelta`.

The target parameter is not set by `power` for newly added methods. You can set it yourself by specifying the name of the column containing the values of the target parameter in the `s(pss_target)` macro. You must set this macro if you want to obtain default graphs for effect-size determination. The target parameter can then be accessed using the column name `target`.

If the target parameter is set in the `s(pss_target)` macro, you can also specify its label in `s(pss_targetlabel)`. This label will be used in the title for the effect-size determination and as the axis label for the graph column `target`.

If your method supports command arguments, the arguments specified directly following the method name, you can specify their corresponding column names in the `s(pss_argnames)` macro. You can then refer to these arguments as `arg1`, `arg2`, and so on, when producing tables or graphs.

`power` *usermethod* uses the following generic titles: "Estimated sample size" for sample-size determination, "Estimated power" for power determination, and "Estimated target parameter" for effect-size determination. You can extend these titles to be more specific to your method by adding text in the `s(pss_title)` macro. For example, if `s(pss_title)` contains "for my test", the resulting titles will be "Estimated sample size for my test", "Estimated power for my test", and "Estimated target parameter for my test". Also see `s(pss_targetlabel)` for how to include a label for the target parameter in the title.

`power` *usermethod* uses the following generic subtitles: "Two-sided test" for a two-sided test or "One-sided test" for a one-sided test when the `onesided` option is specified. You can change the default subtitle by specifying the `s(pss_subtitle)` macro.

Optionally, `power` *usermethod* can display a hypothesis statement if macros `s(pss_hyp_lhs)` and `s(pss_hyp_rhs)` are specified. `s(pss_hyp_lhs)` must contain the parameter of interest, and `s(pss_hyp_rhs)` will typically contain the null or comparison value. For example, if `s(pss_hyp_lhs)` contains `beta1` and `s(pss_hyp_rhs)` contains `0`, `power` *usermethod* will display

```
Ho: beta1 = 0  versus  Ha: beta1 != 0
```

for a two-sided test and

```
Ho: beta1 = 0, one-sided alternative
```

for a one-sided test. The same hypotheses will appear on the graph, unless `s(pss_grhyp_lhs)` and `s(pss_grhyp_rhs)` are specified. These macros are useful if you want to include parameters as symbols on the graph. In our example, we could have defined `s(pss_grhyp_lhs)` as `{&beta}{sub:1}` and `s(pss_grhyp_rhs)` as `0` to include "beta1" as the corresponding symbol on the graph; see [G-4] ***text***.

Handling parsing more efficiently

The power command checks its common options, but you are responsible for checking your method-specific options, *useropts*, or their interaction with power's common options. You can certainly do this in your evaluator. However, the checks will then be performed each time your evaluator is called. You can instead perform all of your checks once within the parser.

Your parser may be an s-class command and may set any of the s() results typically specified in the initializer. This may be useful, for example, for building the columns displayed in the default table dynamically, depending on which options were specified. If all desired s() results are set in the parser, you do not need an initializer.

More examples: Adding two-sample methods

All the examples so far showed how to add a one-sample method to power. We now demonstrate how to add a two-sample method. (Support for multiple-sample methods is not yet available.)

The steps for adding your own two-sample methods are the same as those for adding one-sample methods, except you may need to set the s(pss_samples) macro to contain twosample in the initializer. If any of the two-sample options n1(), n2(), and nratio() are specified, power automatically recognizes the method as a two-sample method. If these options are not used and you need the method to be recognized as a two-sample method, you must set s(pss_samples) to twosample. If you do not want power to respect the two-sample options, you should set s(pss_samples) to onesample.

For illustration, let's add a method comparing two independent proportions using a large-sample χ^2 test. (Note that this method is available in the official power twoproportions command.) For simplicity, we will compute the power of a two-sided test. We will call our new method powertwoprop.

We write our evaluator and save it as power_cmd_powertwoprop.ado.

```
// evaluator
program power_cmd_powertwoprop, rclass
        version 19.5      // (or version 19 if you do not have StataNow)
        //parse command arguments and options
        syntax anything(id="proportions"),       ///
               [ Alpha(real 0.05)       /// significance level
                 n(string)              /// total sample size
                 n1(string) n2(string)  /// group sample sizes
                 NRATio(real 1)         /// N2/N1
                 Power(string)          ///
               ]
        //parse specification of proportions
        gettoken p1 rest : anything
        gettoken p2 rest : rest
        if (`"`p2'"'==="") {
                di as err "Experimental-group proportion must be specified"
                exit 198
        }
        if (`"`rest'"'!="") {
                di as err "Only two proportions may be specified"
                exit 198
        }
        //sample size must be specified to compute power
        if (`"`n'`n1'`n2'"'==="") {
                di as err "One of {bf:n()}, {bf:n1()}, or {bf:n2()} " ///
                          "is required to compute power"
                exit 198
        }
```

```
        //handle some sample-size specifications
        if (`"`n'"'=="") {
                tempname n
                if (`"`n2'"'=="") {
                        tempname n2
                        scalar `n2' = ceil(`nratio'*`n1')
                }
                else if (`"`n1'"'=="") {
                        tempname n1
                        scalar `n1' = ceil(`n2'/`nratio')
                }
                scalar `n'   = `n1'+`n2'
                local nratio = `n2'/`n1'
        }
        else {
                if (`"`n1'"'!="") {
                        tempname n2
                        scalar `n2' = `n' - `n1'
                }
                else if (`"`n2'"'!="") {
                        tempname n1
                        scalar `n1' = `n' - `n2'
                }
                else {
                        tempname n1 n2
                        scalar `n1' = ceil(`n'/(1+`nratio'))
                        scalar `n2' = `n'-`n1'
                }
        }

        //compute power
        tempname diff pbar sigma_D sigma_p crv power
        scalar    `diff' = `p2' - `p1'
        scalar    `pbar' = (`n1'*`p1'+`n2'*`p2')/`n'
        scalar `sigma_D' = sqrt(`p1'*(1-`p1')/`n1'+`p2'*(1-`p2')/`n2')
        scalar `sigma_p' = sqrt(`pbar'*(1-`pbar')*(1/`n1'+1/`n2'))
        scalar     `crv' = invnormal(1-`alpha'/2)*`sigma_p'
        scalar   `power' = normal((`diff'-`crv')/`sigma_D') ///
                         + normal((-`diff'-`crv')/`sigma_D')

        //return results
        return scalar alpha   = `alpha'
        return scalar power   = `power'
        return scalar N       = `n'
        return scalar N1      = `n1'
        return scalar N2      = `n2'
        return scalar nratio  = `nratio'
        return scalar p1      = `p1'
        return scalar p2      = `p2'
end
```

We can now use `powertwoprop` with the `power` command. We specify the two proportions following the command name and group sample sizes in the `n1()` and `n2()` options.

```
. power powertwoprop 0.1 0.3, n1(40) n2(60)
Estimated power
Two-sided test
```

alpha	power	N
.05	.6743	100

As with one-sample methods, we can use an initializer (saved in `power_cmd_powertwoprop_init.ado`) to include additional columns in our default table.

```
// initializer
program power_cmd_powertwoprop_init, sclass
        version 19.5      // (or version 19 if you do not have StataNow)
        sreturn clear
        sreturn local pss_colnames "N1 N2 nratio p1 p2"
        sreturn local pss_samples "twosample"
end

. power powertwoprop 0.1 0.3, n1(40) n2(60)
Estimated power
Two-sided test
```

alpha	power	N	N1	N2	nratio	p1	p2
.05	.6743	100	40	60	1.5	.1	.3

Initializer's s() return settings

The following `s()` results may be set by the initializer or parser:

Macros

s(pss_samples)	`onesample` for a one-sample test or `twosample` for a two-sample test
s(pss_colnames)	columns to be added to the default supported columns
s(pss_allcolnames)	all supported columns
s(pss_tabcolnames)	columns to be added to the default table
s(pss_alltabcolnames)	all columns to be displayed in the default table
s(pss_collabels)	labels for the specified columns
s(pss_colformats)	formats for the specified columns
s(pss_colwidths)	widths for the specified columns
s(pss_colgrlabels)	labels to be used to label columns on the graph
s(pss_colgrsymbols)	symbols to be used to label columns on the graph
s(pss_delta)	column name containing the effect-size parameter
s(pss_target)	column name containing the target parameter
s(pss_targetlabel)	label for the target parameter
s(pss_argnames)	column names containing command arguments
s(pss_title)	method-specific title
s(pss_subtitle)	subtitle
s(pss_hyp_lhs)	left-hand-side parameter or value for the hypothesis
s(pss_hyp_rhs)	right-hand-side parameter or value for the hypothesis
s(pss_grhyp_lhs)	left-hand-side parameter or value for the hypothesis on the graph
s(pss_grhyp_rhs)	right-hand-side parameter or value for the hypothesis on the graph

References

Cain, M. 2021. Calculating power using Monte Carlo simulations, part 5: Structural equation models. *The Stata Blog: Not Elsewhere Classified.* https://blog.stata.com/2021/08/19/calculating-power-using-monte-carlo-simulations-part-5-structural-equation-models/.

Huber, C. 2019. Calculating power using Monte Carlo simulations, part 2: Running your simulation using power. *The Stata Blog: Not Elsewhere Classified.* https://blog.stata.com/2019/01/29/calculating-power-using-monte-carlo-simulations-part-2-running-your-simulation-using-power/.

Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **Intro (power)** — Introduction to power and sample-size analysis for hypothesis tests

[PSS-5] **Glossary**

[ADAPT] ***gsdesign usermethod*** — Add your own methods to the gsdesign command

power, graph — Graph results from the power command

Description

The graph() option of power specifies that results of the power command be graphed.

While there are many options for controlling the look of the graph, you will often merely need to specify the graph option with your power command.

Quick start

Graph of sample-size estimates versus the specified list of power values

```
power onemean 0 0.5, power(0.6(0.1)0.9) graph
```

Graph of sample-size estimates versus probability of type II error

```
power onemean 0 0.5, power(0.6(0.1)0.9) graph(xdimension(beta))
```

Graph of power estimates versus the specified list of sample sizes

```
power onemean 0 0.5, n(10(10)50) graph
```

Add labels for distinct values on the y axis

```
power onemean 0 0.5, n(10(10)50) graph(yvalues)
```

Same as above, but display the power estimates with only three decimal points

```
power onemean 0 0.5, n(10(10)50) graph(yvalues ylabel(,format(%4.3f)))
```

Plots of power for each significance level

```
power twomeans 2.5 3, n(50(10)100) alpha(0.05 0.1) graph
```

Same as above, but produce a separate subgraph for each significance level

```
power twomeans 2.5 3, n(50(10)100) alpha(0.05 0.1) ///
  graph(bydimension(alpha))
```

Plots of power for combinations of alternative means and significance levels

```
power twomeans 2.5 (3 3.5 4), n(50(10)100) alpha(0.05 0.1) graph
```

Same as above

```
power twomeans 2.5 (3 3.5 4), n(50(10)100) alpha(0.05 0.1) ///
  graph(plotdimension(alpha m2))
```

Same as above, but use only alternative means as a plot dimension

```
power twomeans 2.5 (3 3.5 4), n(50(10)100) alpha(0.05 0.1) ///
  graph(plotdimension(m2))
```

Same as above, but using significance levels as by() subgraphs

```
power twomeans 2.5 (3 3.5 4), n(50(10)100) alpha(0.05 0.1) ///
  graph(bydimension(alpha))
```

Same as above, but produce separate graphs for each significance level

```
power twomeans 2.5 (3 3.5 4), n(50(10)100) alpha(0.05 0.1) ///
  graph(graphdimension(alpha))
```

Menu

Statistics > Power, precision, and sample size

Syntax

Produce default graph

```
power ..., graph ...
```

Graph power against sample size

```
power ..., graph(y(power) x(N)) ...
```

Graph sample size against target parameter

```
power ..., graph(y(N) x(target)) ...
```

Graph effect size against sample size

```
power ..., graph(y(delta) x(N)) ...
```

Produce other custom graphs

```
[power] ..., graph(graphopts) ...
```

graphopts	Description
Main	
`ydimension(`*dimlist* [`,` *dimopts*]`)`	use *dimlist* to define y axis
`xdimension(`*dimlist* [`,` *dimopts*]`)`	use *dimlist* to define x axis
`plotdimension(`*dimlist* [`,` *dimopts*]`)`	create plots for groups in *dimlist*
`bydimension(`*dimlist* [`,` *dimopts*]`)`	create subgraphs for groups in *dimlist*
`graphdimension(`*dimlist* [`,` *dimopts*]`)`	create graphs for groups in *dimlist*
`horizontal`	swap x and y axes
`schemegrid`	do not apply default x and y grid lines
`name(`*name* | *stub* [`,` `replace`]`)`	name of graph, or *stub* if multiple graphs
Labels	
`yregular`	place regularly spaced ticks and labels on the y axis
`xregular`	place regularly spaced ticks and labels on the x axis
`yvalues`	place ticks and labels on the y axis for each distinct value
`xvalues`	place ticks and labels on the x axis for each distinct value
`collabels(`*colspec*`)`	change default labels for columns
`nolabels`	label groups with their values, not their labels
`allsimplelabels`	forgo column label and equal signs in all labels
`nosimplelabels`	include column label and equal signs in all labels
`eqseparator(`*string*`)`	replace equal sign separator with *string*
`separator(`*string*`)`	separator for labels when multiple columns are specified in a dimension
`noseparator`	do not use a separator
`format(`*%fmt*`)`	format for converting numeric values to labels
Plot	
`plotopts(`*plot_options*`)`	affect rendition of all plots
`plot`#`opts(`*plot_options*`)`	affect rendition of #th plot
`recast(`*plottype*`)`	plot all plots using *plottype*
Add plots	
`addplot(`*plot*`)`	add other plots to the generated graph
Y axis, X axis, Titles, Legend, Overall, By	
twoway_options	any options documented in [G-3] ***twoway_options***
`byopts(`*byopts*`)`	how subgraphs are combined, labeled, etc.

dimlist may contain any of the following columns:

column	Description
`alpha`	significance level
`power`	power
`beta`	type-II-error probability
`N`	total number of subjects
`N1`	number of subjects in the control group
`N2`	number of subjects in the experimental group
`nratio`	ratio of sample sizes, experimental to control
`K`	number of clusters
`K1`	number of clusters in the control group
`K2`	number of clusters in the experimental group
`kratio`	ratio of numbers of clusters, experimental to control
`M`	cluster size
`M1`	cluster size in the control group
`M2`	cluster size in the experimental group
`mratio`	ratio of cluster sizes, experimental to control
`delta`	effect size
`target`	target parameter
method_columns	columns specific to the method specified with `power`

colspec is

column "*label*" [*column* "*label*" [...]]

dimopts	Description
`labels(`*lablist*`)`	list of quoted strings to label each level of the dimension
`elabels(`*elablist*`)`	list of enumerated labels
`nolabels`	label groups with their values, not their labels
`allsimplelabels`	forgo column name and equal signs in all labels
`nosimplelabels`	include column name and equal signs in all labels
`eqseparator(`*string*`)`	replace equal sign separator with *string* in the dimension
`separator(`*string*`)`	separator for labels when multiple columns are specified in the dimension
`noseparator`	do not use a separator
`format(`*%fmt*`)`	format for converting numeric values to labels

where *lablist* is defined as

"*label*" ["*label*" [...]]

elablist is defined as

"*label*" [# "*label*" [...]]

and the #s are the levels of the dimension.

plot_options	Description
marker_options	change look of markers (color, size, etc.)
marker_label_options	add marker labels; change look or position
cline_options	change look of the line

Suboptions

The following are suboptions within the `graph()` option of the `power` command.

Main

`ydimension()`, `xdimension()`, `plotdimension()`, `bydimension()`, and `graphdimension()` specify the dimension to be used for the graph's y axis, x axis, plots, `by()` subgraphs, and graphs.

The default dimensions are based on your analysis. The y dimension is power for power determination, sample size for sample-size determination, and target parameter for effect-size determination. If there is only one column containing multiple values, this column is plotted on the x dimension. Otherwise, the x dimension is sample size for power determination, target parameter for sample-size determination, and sample size for effect-size determination. Other columns that contain multiple values are used as plot dimensions. See *Default graphs* below for details. You may override the defaults and explicitly control which columns are used on each dimension of the graph using these dimension suboptions.

Each of these suboptions supports suboptions that control the labeling of the dimension—axis labels for `ydimension()` and `xdimension()`, plot labels for `plotdimension()`, subgraph titles for `bydimension()`, and graph titles for `graphdimension()`.

For examples using the dimension suboptions, see *Changing default graph dimensions* below.

`ydimension(`*dimlist* [, *dimopts*]`)` specifies the columns for the y axis in *dimlist* and controls the content of those labels with *dimopts*.

`xdimension(`*dimlist* [, *dimopts*]`)` specifies the columns for the x axis in *dimlist* and controls the content of those labels with *dimopts*.

`plotdimension(`*dimlist* [, *dimopts*]`)` specifies in *dimlist* the columns whose levels determine the plots and optionally specifies in *dimopts* the content of the plots' labels.

`bydimension(`*dimlist* [, *dimopts*]`)` specifies in *dimlist* the columns whose levels determine the `by()` subgraphs and optionally specifies in *dimopts* the content of the subgraphs' titles.

`graphdimension(`*dimlist* [, *dimopts*]`)` specifies in *dimlist* the columns whose levels determine the graphs and optionally specifies in *dimopts* the content of the graphs' titles.

See the definition of *columns in graph* in [PSS-5] **Glossary**.

`horizontal` reverses the default x and y axes. By default, the values computed by `power` are plotted on the y axis, and the x axis represents one of the other columns. Specifying `horizontal` swaps the axes.

One common use is to put sample size on the x axis even when it is the value computed by `power`. This suboption can also be useful with the long labels produced when the `parallel` option is specified with `power`.

See *Parallel plots* below for an example of the `horizontal` suboption.

`schemegrid` specifies that x and y grid lines not always be drawn on the `power` graph. Instead, whether grid lines are drawn will be determined by the current scheme.

`name(`*name* | *stub* [`, replace`]`)` specifies the name of the graph or graphs. If the `graphdimension()` suboption is specified, then the argument of `name()` is taken to be *stub*, and graphs named *stub*1, *stub*2, … are created.

`replace` specifies that existing graphs of the same name may be replaced.

If `name()` is not specified, default names are used, and the graphs may be replaced by subsequent `power` graphs or other graphing commands.

Labels

All the suboptions listed under the **Labels** tab may be specified directly within the `graph()` option. All of them except `yregular`, `xregular`, `yvalues`, and `xvalues` may be specified as *dimopts* within `ydimension()`, `xdimension()`, `plotdimension()`, `bydimension()`, and `graphdimension()`. When suboptions are specified in one of the dimension options, only the labels for that dimension are affected. When suboptions are specified outside the dimension options, all labels on all dimensions are affected. Specifications within the dimension options take precedence.

`yregular` and `yvalues` specify how tick marks and labels are to be placed on the y axis.

`yregular` specifies that regularly spaced ticks and labels be placed on the y axis.

`yvalues` specifies that a tick and label be placed for each distinct value.

If neither is specified, an attempt is made to choose the most reasonable option based on your results. Labeling may also be specified using the standard `graph twoway` axis labeling rules and options.

`xregular` and `xvalues` do the same for tick marks and labels to be placed on the x axis.

`collabels(`*colspec*`)` specifies labels to be used on the graph for the specified columns. For example, `collabels(N "N")` specifies that wherever the column `N` is used on a graph—axis label, plot label, graph title, legend title, etc.—"N" be shown rather than the default label "Sample size".

Multiple columns may be relabeled by typing, for example,

```
collabels(N "N" ma "Alternative mean")
```

and SMCL tags for Greek characters and other typesetting can be used by typing, for example,

```
collabels(alpha "{&alpha}" ma "{&mu}{sub:a}")
```

See the definition of *columns in graph* in [PSS-5] **Glossary**.

`nolabels` specifies that value labels not be used to construct graph labels and titles for the levels in the dimension. By default, if a column in a dimension has value labels, those labels are used to construct labels and titles for axis ticks, plots, subgraphs, and graphs.

`allsimplelabels` and `nosimplelabels` control whether a graph's labels and titles include just the values of the columns or also include column labels and equal signs. The default depends on whether the dimension is an axis dimension or one of the plot, by, and graph dimensions. It also depends on whether the values for the level of the dimension are labeled. An example of a simple label is "alpha" or ".05" and of a nonsimple label is "alpha=.05".

In power, graph simple labels are almost universally best for x and y axes and also best for most plot labels. Labels with an equal sign are typically preferred for subgraph and graph titles. These are the defaults used by power, graph. The allsimplelabels and nosimplelabels suboptions let you override the default labeling.

allsimplelabels specifies that all titles and labels use just the value or value label of the column.

nosimplelabels specifies that all titles and labels include *dimname*= before the value or value label.

eqseparator(*string*) specifies a custom separator between column labels and values in labels. Use *string* in place of the default equal sign. This option is for use with nosimplelabels.

separator(*string*) and noseparator control the separator between label sections when more than one column is used to specify a dimension. The default separator is a comma followed by a space, but no separator may be requested with noseparator, or the default may be changed to any string with separator().

For example, if bydimension(a b) is specified, the subgraph labels in our graph legend might be "a=1, b=1", "a=1, b=2", ... Specifying separator(:) would create labels "a=1:b=1", "a=1:b=2", ...

format(*%fmt*) specifies how numeric values are to be formatted for display as axis labels, labels on plots, and titles on subgraphs and graphs.

Plot

plotopts(*plot_options*) affects the rendition of all plots. The *plot_options* can affect the size and color of markers, whether and how the markers are labeled, and whether and how the points are connected; see [G-3] ***marker_options***, [G-3] ***marker_label_options***, and [G-3] ***cline_options***.

These settings may be overridden for specific plots by using the plot#opts() suboption.

plot#opts(*plot_options*) affects the rendition of the #th plot. The *plot_options* can affect the size and color of markers, whether and how the markers are labeled, and whether and how the points are connected; see [G-3] ***marker_options***, [G-3] ***marker_label_options***, and [G-3] ***cline_options***.

recast(*plottype*) specifies that results be plotted using *plottype*. *plottype* may be scatter, line, connected, area, bar, spike, dropline, or dot; see [G-2] **graph twoway**. When recast() is specified, the plot-rendition options appropriate to the specified *plottype* may be used in lieu of *plot_options*. For details on those suboptions, follow the appropriate link from [G-2] **graph twoway**.

You may specify recast() within a plotopts() or plot#opts() suboption. It is better, however, to specify it as documented here, outside those suboptions. When it is specified outside those suboptions, you have greater access to the plot-specific rendition suboptions of your specified *plottype*.

Add plots

addplot(*plot*) provides a way to add other plots to the generated graph; see [G-3] ***addplot_option***.

If multiple graphs are drawn by a single power command or if *plot* specifies plots with multiple y variables, for example, scatter y1 y2 x, then the graph's legend will not clearly identify all the plots and will require customization using the legend() suboption; see [G-3] ***legend_options***.

Y axis, X axis, Titles, Legend, Overall, By

twoway_options are any of the options documented in [G-3] ***twoway_options***. These include options for titling the graph (see [G-3] ***title_options***); for saving the graph to disk (see [G-3] ***saving_option***); for controlling the labeling and look of the axes (see [G-3] ***axis_options***); for controlling the look,

contents, position, and organization of the legend (see [G-3] ***legend_options***); for adding lines (see [G-3] ***added_line_options***) and text (see [G-3] ***added_text_options***); and for controlling other aspects of the graph's appearance (see [G-3] ***twoway_options***).

The `label()` suboption of the `legend()` option has no effect on `power, graph`. Use the `order()` suboption instead.

`byopts(`*byopts*`)` affects the appearance of the combined graph when `bydimension()` is specified or when the default graph has subgraphs, including the overall graph title, the position of the legend, and the organization of subgraphs. See [G-3] ***by_option***.

Remarks and examples

Remarks are presented under the following headings:

`power, graph` produces power curves and other graphical output from the `power` command. Power graphs are useful for visualizing the results of sensitivity analysis, which investigates the effect of varying study parameters on power, sample size, or other components of the study. The true values of study parameters are usually unknown. Power and sample-size analysis uses best guesses for these values. It is important to evaluate the sensitivity of the computed power or sample size to the chosen values of study parameters. For example, to evaluate variability of power values, you can compute powers for various ranges of values for the parameters of interest and display the resulting powers in a table (see [PSS-2] **power, table**) or plot them on a graph.

Using power, graph

In most cases, you will probably be satisfied with the graphs that `power` produces by default when you specify the `graph` option. For other cases, `power, graph()` offers many options for you to produce the graph you desire.

Think of `power, graph()` as graphing the columns of `power, table`. One of the columns will be placed on the x axis, another will be placed on the y axis, and, if you have more columns with varying values, separate plots will be created for each. Similarly, we use the terms "column symbol", "column name", and "column label" to refer to symbols, names, and labels that appear in tables when tabular output is requested.

By default, `power, graph` plots the column corresponding to the estimated parameter on the y axis: `power`, when power is computed; `N` (in most cases), when sample size is computed; and `target`, when the target parameter is computed. When there is only one varying column, the x axis uses this column by default. When there are multiple varying columns, the default x axis depends on what is being computed.

In a cluster randomized design (CRD), the sample-size determination consists of the determination of the number of clusters given cluster size or the determination of cluster size given the number of clusters. Thus, when the number of clusters is computed, `power, graph` by default plots the number of clusters, `K`, on the y axis for one-sample methods and the number of clusters in the experimental group, `K2`, for two-sample methods. When the cluster size is computed, `power, graph` plots the cluster size, `M`, for one-sample methods and the cluster size in the experimental group, `M2`, for two-sample methods.

If power is computed (power determination), the default x axis is either the sample size, if sample size varies, or the target parameter, if the target parameter varies and sample size does not vary. If neither the sample size nor the target parameter varies, the power is plotted against one of the other varying parameters. In a CRD, the sample size varies whenever the number of clusters varies, the cluster size varies, or both vary. The default x axis is then the number of clusters if the number of clusters varies or it is the cluster size if the cluster size varies and the number of clusters does not.

If sample size is computed (sample-size determination), the default x axis is either the target parameter, if the target parameter varies, or the power, if power varies and the target parameter does not vary. If neither the target parameter nor the power varies, the sample size is plotted against one of the other varying parameters. For a CRD, by "sample size" we mean either the number of clusters or the cluster size, whichever one is computed.

If target parameter is computed (effect-size determination), the default x axis is either sample size, if sample size varies, or power, if power varies and sample size does not vary. If neither sample size nor power varies, the target parameter is plotted against one of the other varying parameters. For a CRD, when sample size varies, the default x axis is the number of clusters if the number of clusters varies or it is the cluster size if the cluster size varies and the number of clusters does not.

You can also plot the effect size `delta` instead of the target parameter `target` by specifying the `ydimension(delta)` suboption within `graph()`; see example 4. The graphs of `delta` may not reflect all the unique combinations of other study parameters, because effect size is not necessarily a one-to-one function of the constituent study parameters.

`power, graph()` provides great flexibility for customizing graphical output. You can make minor changes such as modifying the graph or axes titles or modifying the line color and style, or you can completely redesign the graph by changing the axes and style of the graph. The Graph Editor can also be used for additional customization; see [G-1] **Graph Editor**.

When you produce a graph, the table of results is suppressed. You can request that the table be displayed in addition to the graph by specifying the `table` option with `graph()`.

Graph symbols

Whenever space allows, such as on y and x axes, graphical output displays *extended column labels*, which include column labels and column symbols in parentheses. In other cases, such as in legend labels or by graph titles, graphical output includes only column (parameter) symbols for readability.

The following common symbols are used. See the documentation entry of the specified power method for additional symbols specific to that method.

Symbol	Description
α	significance level
β	probability of a type II error
$1-\beta$	power
N	total sample size
N_1	sample size of the control group
N_2	sample size of the experimental group
N_2/N_1	ratio of sample sizes, experimental to control
K	number of clusters
K_1	number of clusters in the control group
K_2	number of clusters in the experimental group
K_2/K_1	ratio of numbers of clusters, experimental to control
M	cluster size
M_1	cluster size of the control group
M_2	cluster size of the experimental group
M_2/M_1	ratio of cluster sizes, experimental to control
δ	effect size
method_symbols	symbols specific to the *method* specified with `power`

Default graphs

We start with a demonstration of several default power graphs and then show how you can produce custom power graphs in the subsequent sections.

In what follows, we graph the results of power and sample-size analysis for a two-sided 5%-level one-sample t test comparing the population mean with a hypothesized value; see [PSS-2] **power onemean**.

▷ Example 1: Power curves

When we compute power given a range of sample sizes, `power, graph` plots power on the y axis and sample size on the x axis.

```
. power onemean 0 1, n(10(2)40) graph
```

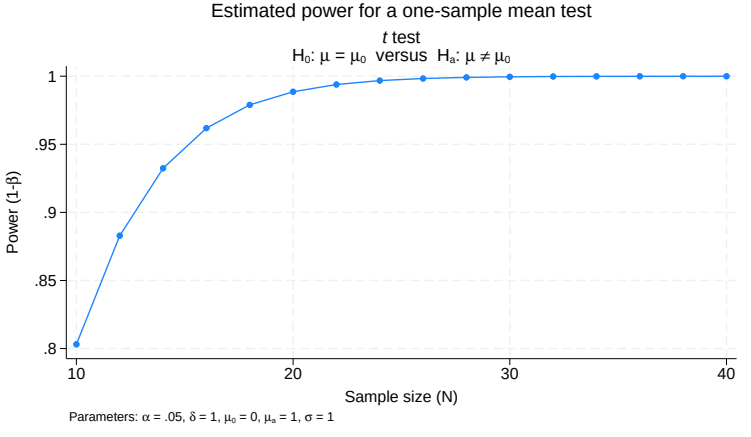


Figure 1.

As expected, power increases as sample size increases.

The default axis labels include column labels and column symbols in parentheses. The labels can be changed as we show in example 6. The values of constant parameters are displayed in the note titled "Parameters": significance level α is 0.05, effect size δ (the standardized mean difference $(\mu_a - \mu_0)/\sigma$ for a one-sample t test) is 1, null mean μ_0 is 0, alternative mean μ_a is 1, and standard deviation σ is 1.

In addition to varying sample size, we may compute powers for different alternative values of the mean.

```
. power onemean 0 (0.8 1), n(10(2)40) graph
```

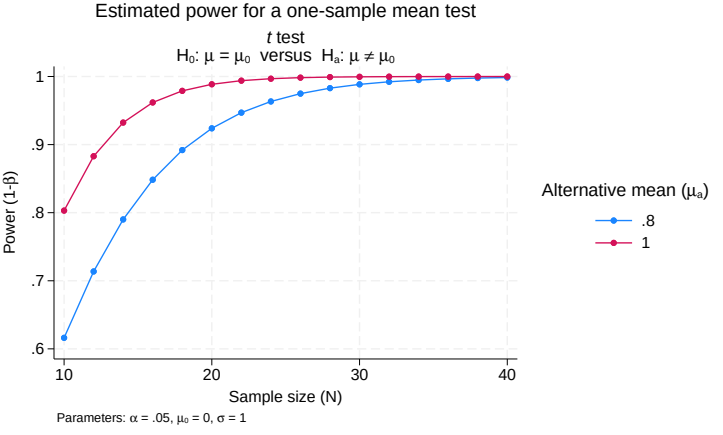


Figure 2.

For a given sample size, the larger the alternative value, the higher the power.

`power, graph` displays two power curves corresponding to the specified alternative values on one plot. The first curve is displayed in navy, and the second curve is displayed in maroon. The default colors of the lines and, in general, the overall look of the graph are determined by the current graph scheme. The scheme used here is `stgcolor`; see [G-2] **set scheme** for details. We also show how to change the default look of the curves in example 7.

We can obtain power curves for varying values of several parameters. For example, below we compute powers for varying values of alternative mean, sample size, and standard deviation.

```
. power onemean 0 (0.8 1), n(10(2)40) sd(1 1.5) graph
```

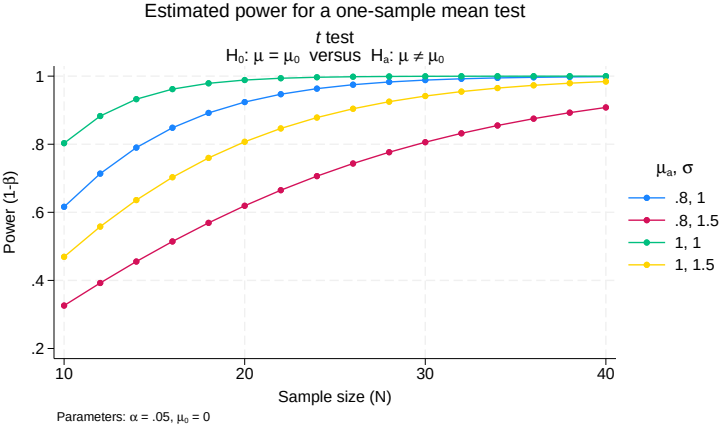


Figure 3.

The larger the standard deviation, the lower the power. This can be seen more easily in figure 11 and figure 21.

`power, graph` plots a separate curve for each unique combination of the values of the alternative mean and standard deviation on one plot. Alternatively, you can display curves on separate plots (by graphs) or even on separate graphs; see example 5. Instead of the extended legend label, as displayed in figure 2, the title of the legend now displays only column symbols because the legend contains more than one column. For an alternative look of the legend, see figure 17.

If we specify only one sample size in the previous figure, the values of the alternative mean will be plotted on the x axis. You can try this yourself if you would like.

◁

▷ Example 2: Sample-size curves

Instead of power curves, we can plot estimated sample sizes for a range of power values to get an idea of how the requirement on sample size changes for powers of interest.

```
. power onemean 0 1, power(0.8(0.05)0.95) graph
```

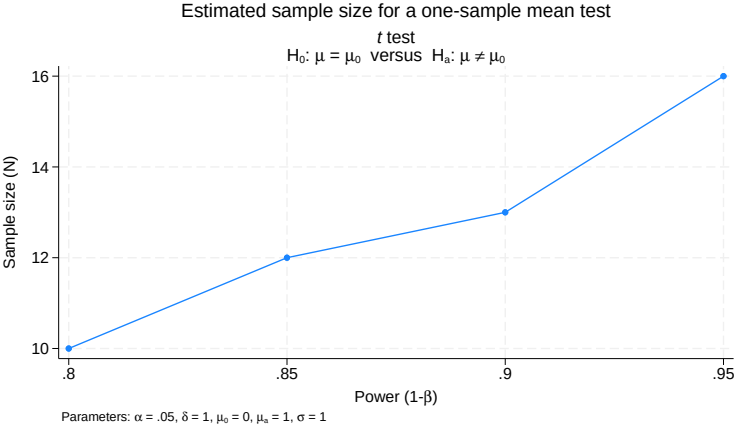


Figure 4.

The sample size increases as the power increases.

The look of this graph is the same as that of the graph in figure 1, except sample size is plotted on the y axis and power is plotted on the x axis.

We may want to investigate how other study parameters such as alternative mean and standard deviation affect the sample-size requirement for given powers.

```
. power onemean 0 (0.3(0.1)1), power(0.8 0.9) sd(1 1.5) graph
```

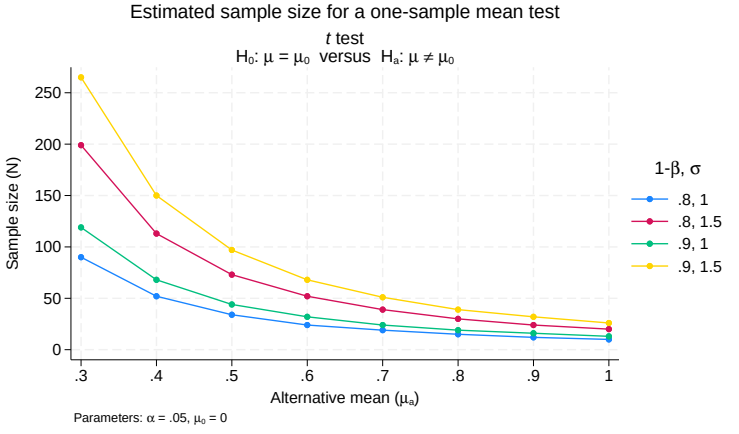


Figure 5.

The larger the alternative mean, or more precisely, the larger the standardized difference between the alternative and null means, the larger the required sample size.

When multiple study parameters each contain multiple values, as in the above figure, the default x axis for sample-size curves is the target parameter (the alternative mean in our example), provided that the target parameter varies. You can plot a different parameter on the x axis such as standard deviation; see example 4 about how to change the default axes.

Let's now see an example of computing the number of clusters in a CRD. Recall that in a CRD, the sample-size determination consists of the determination of the number of clusters given cluster size or the determination of cluster size given the number of clusters.

We want to plot the estimated number of clusters for a range of power values to get an idea of how many clusters are required for the powers of interest. Suppose the cluster size is 5 (`m(5)`).

```
. power onemean 0 1, power(0.8(0.05)0.95) m(5) graph
```

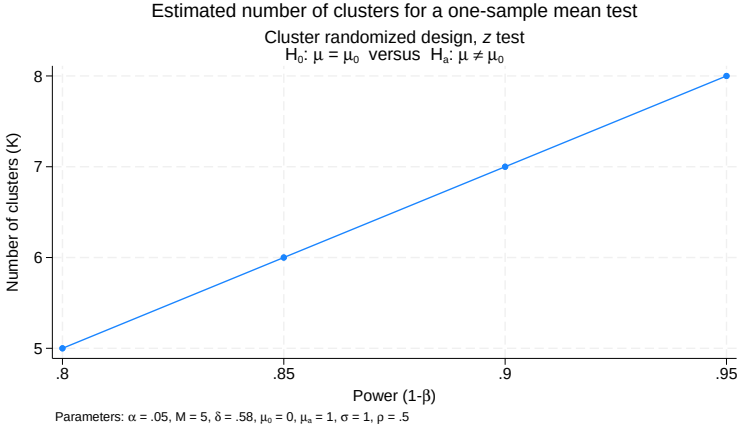


Figure 6.

The number of clusters increases as the power increases.

◁

▷ Example 3: “Effect-size” curves

What we mean by “effect-size” curves are curves with an effect of interest, which may or may not be the actual effect size, plotted on the y axis. In fact, `power, graph` by default plots the target parameter on the y axis.

We can plot the alternative mean (the target parameter), or more precisely, the smallest value of the alternative mean that can be detected using the considered t test, against the sample size for specified values of power and default values of other study parameters.

```
. power onemean 0, power(0.8 0.9) n(10(2)40) graph
```

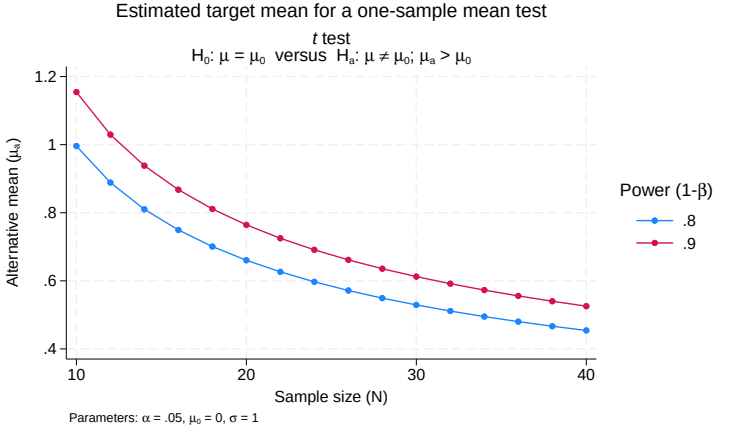


Figure 7.

◁

The larger the sample size and the smaller the power, the smaller the values of the alternative mean that can be detected by the test. The default x axis is the sample size whenever the sample size varies.

If desired, you can plot the actual effect size instead of the target parameter on the y axis; see example 4 for details.

Changing default graph dimensions

So far, we have demonstrated the graphs that `power, graph` produces by default. In this section, we demonstrate how you can modify the default graphs.

We can use `power, graph()` to modify graphs by specifying suboptions to control the look of the graph.

▷ Example 4: Changing default graph axes

The default y axis corresponds to the computed study parameter—power for power determination, sample size for sample-size determination, and target parameter for effect-size determination. You would rarely need to change this dimension. One example when this may be useful is when you want to plot the estimated probability of a type II error, β, instead of the estimated power.

Following figure 1, let's plot the estimated probability of a type II error instead of power on the y axis. We specify the name of the column to be displayed on the y axis, `beta`, in the `ydimension()` suboption of `power`'s `graph()` option. We use the minimum abbreviation `y()` of the suboption.

```
. power onemean 0 1, n(10(2)40) graph(y(beta))
```

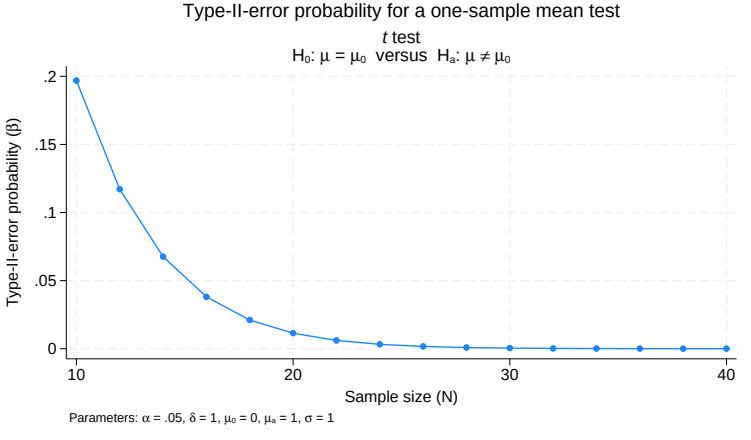


Figure 8.

The probability of a type II error decreases as the sample size increases. This is expected considering the relationship between power, $1 - \beta$, and β.

In example 3, we plotted the minimum detectable values of the target parameter, alternative mean, against sample size. Instead of the alternative mean, we can plot the corresponding effect size, `delta`.

```
. power onemean 0, power(0.8 0.9) n(10(2)40) graph(y(delta))
```

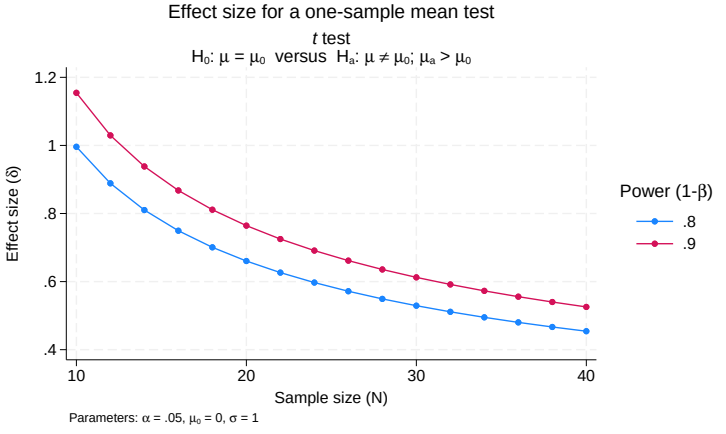


Figure 9.

The effect size is now plotted on the y axis. The y values are the same as in figure 7 because the effect size corresponds to the alternative mean when the null mean is 0 and the standard deviation is 1. Note also that for a one-sample t test, the computed effect size is not affected by the specified values of the null mean or standard deviation, so the effect-size curves will stay the same if you vary these parameters.

When the `ydimension()` suboption is specified, the y axis is replaced with the specified column, and the column corresponding to the default y axis is used as a plot dimension. When the `ydimension()` suboption contains `delta`, the `target` column is omitted from the graph. When the `ydimension()` suboption contains `beta`, the `power` column is omitted from the graph.

In figure 2, by default, the power is plotted against varying values of the sample size, and a separate curve is plotted for each of the varying values of the alternative mean. We can change the default x axis by specifying the `xdimension()` suboption (abbreviated to `x()`) within `power`'s `graph()` option.

```
. power onemean 0 (0.3(0.1)1), n(10 20 40) graph(x(ma))
```

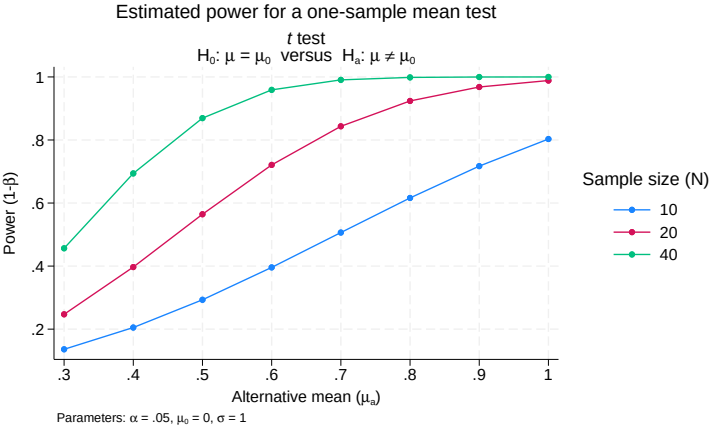


Figure 10.

The x axis now contains the values of the alternative mean, and a separate curve is now plotted for each sample size.

When the `xdimension()` suboption is specified, the x axis is replaced with the specified column, and the column corresponding to the default x axis is used as a plot dimension. When the `xdimension()` suboption contains `delta`, the `target` column is omitted from the graph. When the `xdimension()` suboption contains `beta`, the `power` column is omitted from the graph.

◁

Example 5: By graphs and multiple graphs

Let's return to figure 3 demonstrating multiple power curves on one graph. Suppose we want to more easily see the impact of standard deviation on power given an alternative mean value. We can produce a separate plot for each of the mean values by specifying the column `ma` in the `bydimension()` suboption (abbreviated to `by()`) within `power`'s `graph()` option.

```
. power onemean 0 (0.8 1), n(10(2)40) sd(1 1.5) graph(by(ma))
```

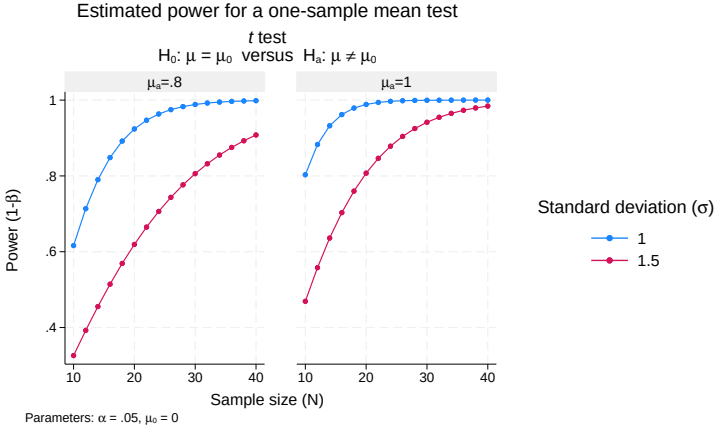


Figure 11.

For examples of how to modify the look of a by graph, see example 8.

In the presence of many varying parameters, even by graphs may look crowded. In this case, you may consider producing multiple by graphs. In the example above, suppose that we also want to vary the significance level α. We add the `alpha(0.05 0.1)` option to the previous command.

```
. power onemean 0 (0.8 1), n(10(2)40) sd(1 1.5) alpha(0.05 0.1) graph(by(ma))
(output omitted)
```

The above command produces a graph containing two by graphs. Each by graph contains four curves in which each corresponds to a unique combination of values of the standard deviation and significance level. We leave this for you to verify.

This is a lot of information for a single graph to convey, so instead, we request that a separate graph be produced for each of the significance levels by specifying the `graphdimension(alpha)` suboption (abbreviated to `graph()`) within `power`'s `graph()` option.

```
. power onemean 0 (0.8 1), n(10(2)40) sd(1 1.5) alpha(0.05 0.1)
> graph(by(ma) graph(alpha))
```

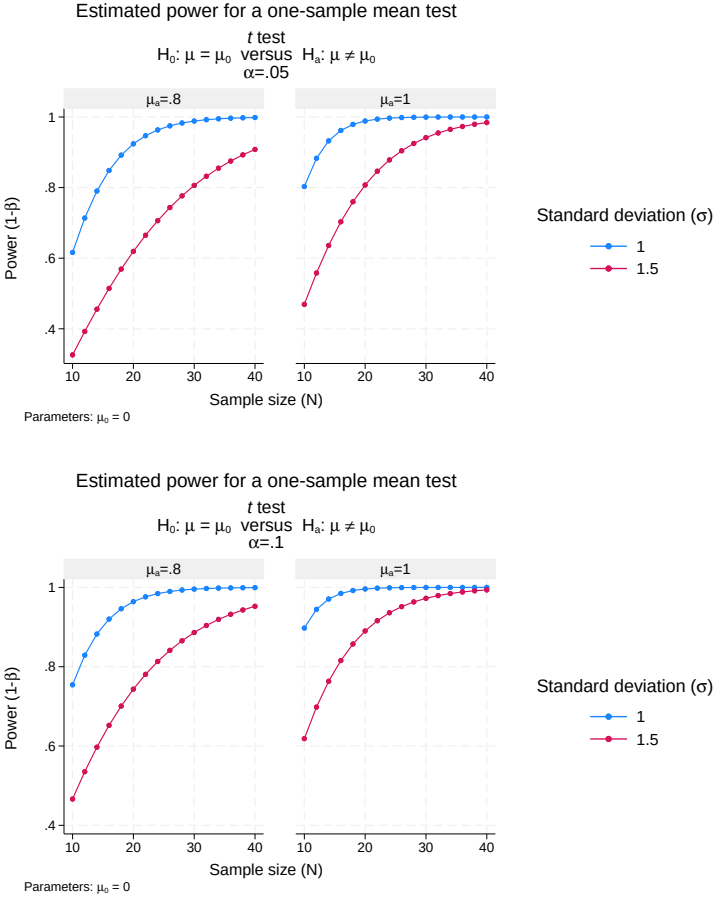


Figure 12.

Changing the look of graphs

▷ Example 6: Modifying axis labels

Reasonable defaults for axis labels are chosen based on your results. You can modify the defaults by using any of `power, graph()`'s labeling suboptions or `graph twoway`'s *axis_label_options*; see [G-3] ***axis_label_options***.

For example, we can request that ticks and labels of the y and x axes be placed for each distinct value instead of using equally spaced values as in figure 1.

```
. power onemean 0 1, n(10(2)20) graph(yvalues xvalues ylabel(, format(%4.3f)))
```

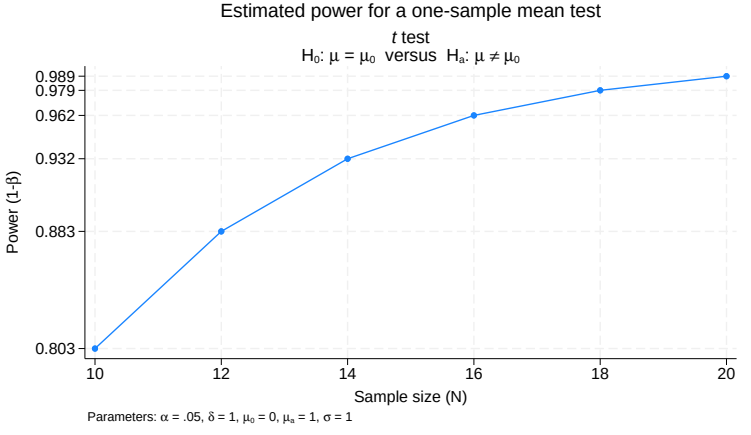


Figure 13.

In this example, we specified fewer sample sizes to obtain a more readable graph. To further improve readability, we also changed the default format of the values on the y axis to show only three decimal points by using `ylabel(, format(%4.3f))`.

We can use `ylabel()` and `xlabel()` to add text labels for some of the axis values. For example, suppose that our budget is 30 subjects. We can use `xlabel()` to label the sample-size value of 30 as "Budgeted".

```
. power onemean 0 1, n(10(2)40) graph(xlabel(30 "Budgeted", add))
```

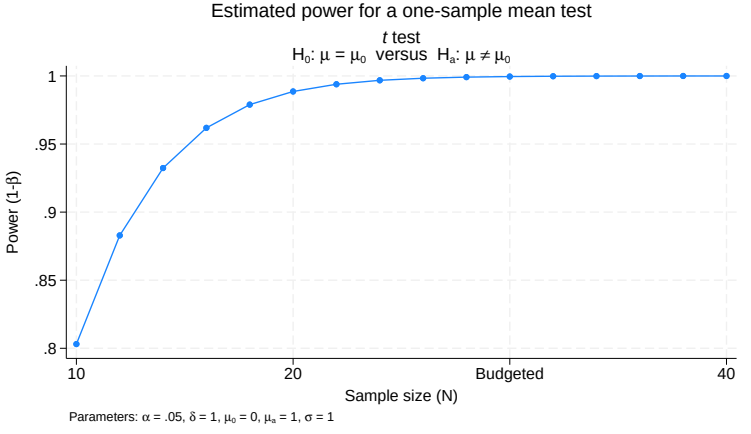


Figure 14.

We can use `ytitle()` and `xtitle()` to change the axis titles.

```
. power onemean 0 1, n(10(2)20) graph(ytitle("Power") xtitle("Sample size")
> title("Estimated power") subtitle("") note(""))
```

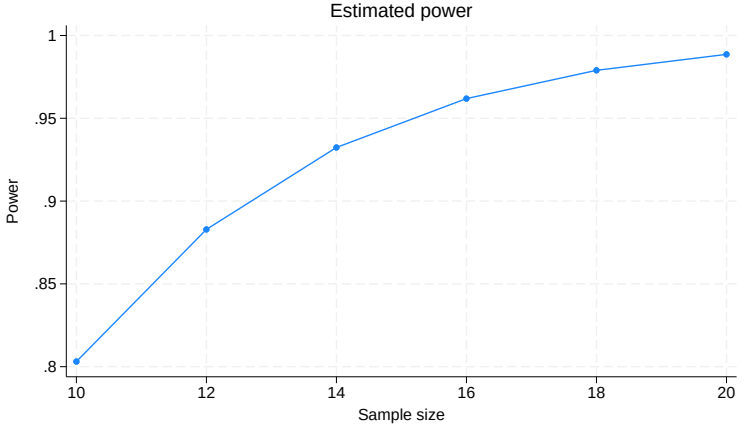


Figure 15.

In addition to modifying the axis titles, we also shortened the default title and suppressed the default subtitle and note.

You may find the `collabels()` suboption useful to override the default column labels. The specified column labels will be used wherever the corresponding column is used on the graph.

For example, change the default labels of the power, sample-size, and alternative-mean columns to, respectively, “Power”, “N”, and “Alternative mean” in figure 2 as follows:

```
. power onemean 0 (0.8 1), n(10(2)40)
> graph(collabels(N "N" power "Power" ma "Alternative mean"))
```

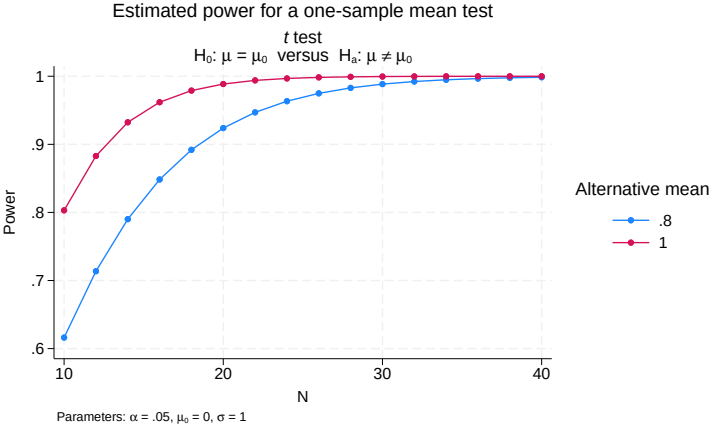


Figure 16.

For overlaid plots, we may consider alternative labeling of the plotted curves in the legend by using the `nosimplelabels` suboption (abbreviated to `nosimple`). We also suppress the legend title and request that an equality sign with a space on each side be used as a separator.

```
. power onemean 0 (0.8 1), n(10(2)40)
> graph(nosimple legend(title("")) eqsep(" = "))
```

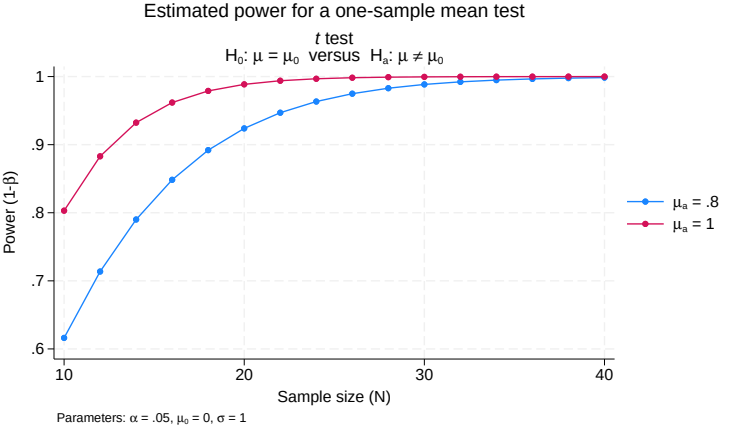


Figure 17.

Example 7: Plot options

We can use the `plotopts()` and `plot#opts()` suboptions within `graph()` to modify the default look of the plotted lines. If there are multiple curves, the `plotopts()` suboption will apply changes to all curves. Use the corresponding `plot#opts()` suboption to change the look of only the specific #th curve.

Here are a few examples of using these suboptions.

```
. power onemean 0 (0.1(0.1)1), graph(plotopts(mlabel(N) mlabpos(1)))
```

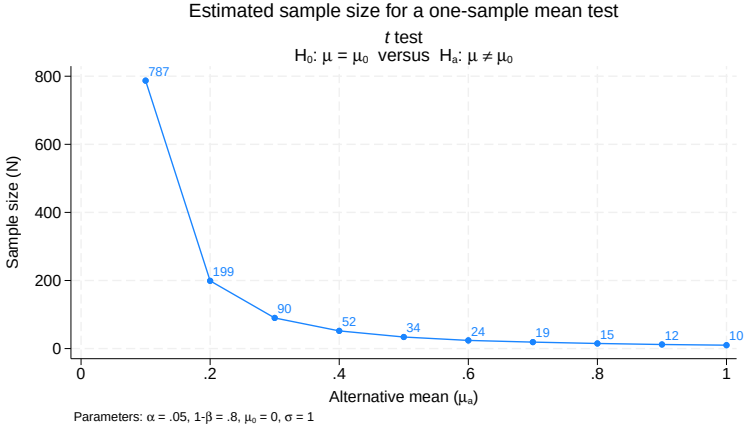


Figure 18.

We specified `mlabel()` within the `plotopts()` suboption to label each data point on the graph with its corresponding sample-size value. `mlabpos()` was used to place the marker labels at the one o'clock position.

For plots containing multiple curves such as in figure 3, the `plotopts()` suboption controls the look of all curves. For example, we can change the marker symbol from the default solid circle to a solid triangle.

```
. power onemean 0 (0.8 1), n(10(2)40) sd(1 1.5) graph(plotopts(msymbol(T)))
```

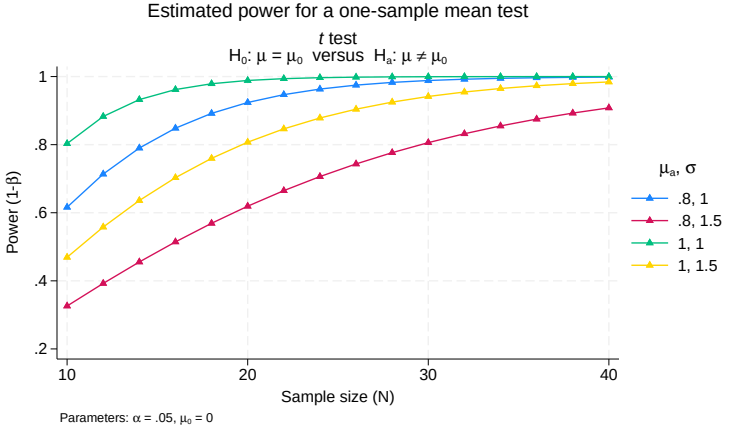


Figure 19.

To control the look of each curve, we can use multiple `plot#opts()` suboptions. For example, we can request that the curves corresponding to the same standard deviation be plotted using the same color:

```
. power onemean 0 (0.8 1), n(10(2)40) sd(1 1.5)
> graph(plot3opts(color(stblue)) plot4opts(color(stred)))
```

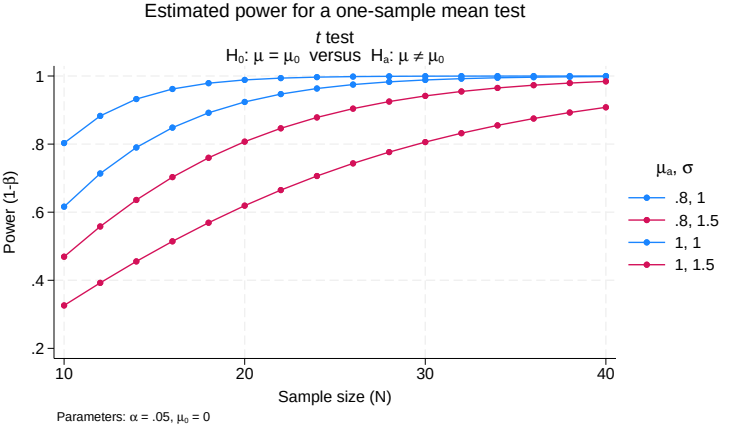


Figure 20.

◁

▷ Example 8: Modifying the look of by-graphs

In figure 11, we created a by-graph, where each graph appeared tall and narrow because of the legend on the right side. To allow extra width for the graphs, we could move the legend to the bottom and place the labels in two columns by using the `legend()` option. Instead, we will specify `scheme(stcolor_alt)` within `power`'s `graph()` option to use a scheme that automates the placement of the legend at the bottom.

```
. power onemean 0 (0.8 1), n(10(2)40) sd(1 1.5) graph(by(ma) scheme(stcolor_alt))
```

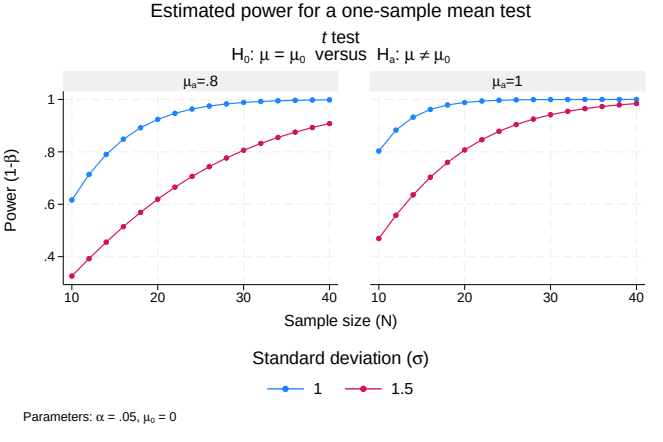


Figure 21.

The look of by graphs is further controlled by the `byopts()` suboption specified within `power`'s `graph()` option.

For example, in figure 11, we can specify `yrescale` within the `byopts()` suboption to allow the scales of the two by graphs to differ. We use the alternative means of 0.5 and 1 instead of 0.8 and 1 to demonstrate differences between scales.

```
. power onemean 0 (0.5 1), n(10(2)40) sd(1 1.5) graph(by(ma)
> scheme(stcolor_alt) byopts(yrescale))
```

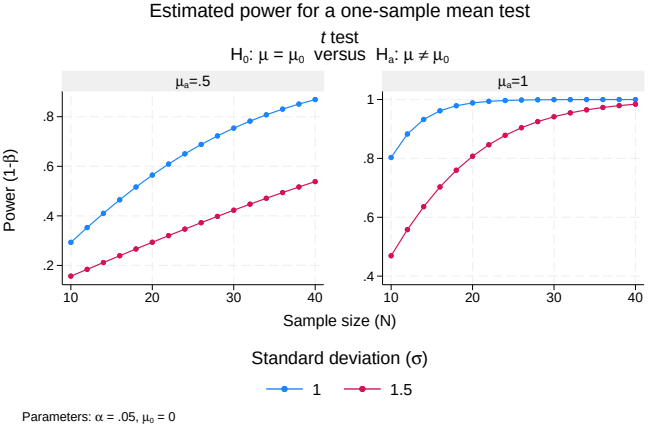


Figure 22.

We can use `byopts()` to change the overall graph title and subtitle.

```
. power onemean 0 (0.5 1), n(10(2)40) sd(1 1.5) graph(by(ma)
> scheme(stcolor_alt) byopts(yrescale title("Power vs sample size")
> subtitle("")))
```

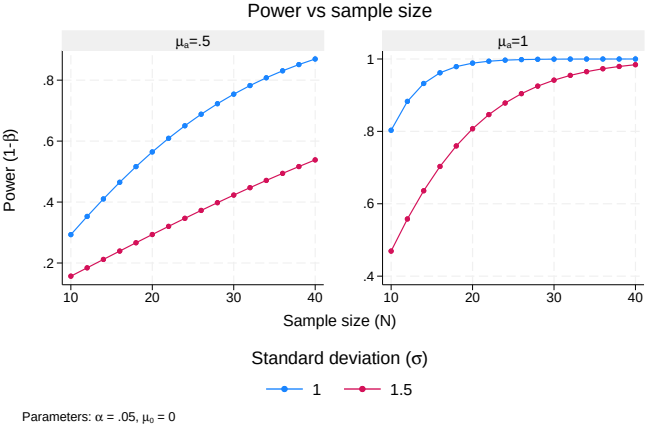


Figure 23.

Note that if you use `title()` and `subtitle()` outside `byopts()`, you will change the title and subtitle of the individual by graphs and not the overall graph.

Parallel plots

Sometimes, you may be interested in comparing powers of parallel sets of parameters, that is, parameters that vary in parallel instead of being nested. In this situation, the results represent a collection of data points rather than a curve, and they are displayed on the graph as a scatterplot without connecting points.

For such parallel plots, the default display of the results on the y axis may be cumbersome. A more appealing look may be a graph that swaps the y and x axes, the horizontal graph. Such a look may be achieved by specifying the `horizontal` suboption within `graph()`.

```
. power onemean 0 (0.1(0.1)0.9), sd(1(0.1)1.9) parallel
> graph(x(ma sd) horizontal nosimple ytitle(""))
```

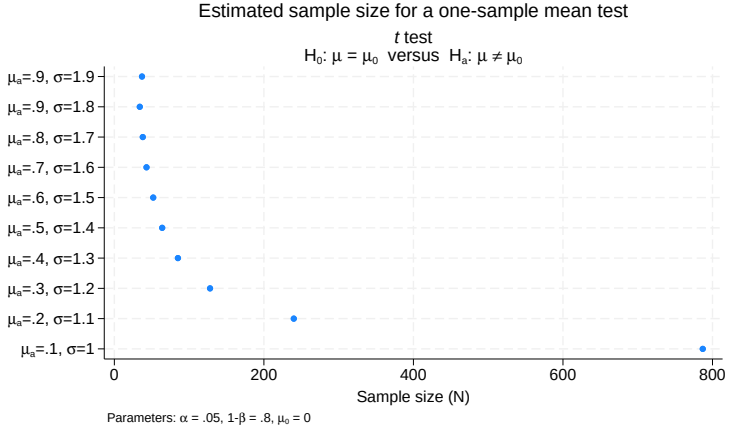


Figure 24.

To improve the look of the horizontal graph, we specified the `nosimplelabels` suboption to request that the labels on the y axis include the parameter symbol; we also suppressed the y-axis title.

Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, table** — Produce table of results from the power command

power, table — Produce table of results from the power command

Description

`power, table` displays results in a tabular format. `table` is implied if any of the `power` command's arguments or options contain more than one element. The `table` option is useful if you are producing graphs and would like to see the table as well or if you are producing results one case at a time using a loop and wish to display results in a table. The `notable` option suppresses table results; it is implied with the graphical output of `power, graph`; see [PSS-2] **power, graph**.

Quick start

Sample size for a one-sample mean test in tabular format

```
power onemean 0.4 0.65, table
```

Same as above, but label the alpha column "Sig. level"

```
power onemean 0.4 0.65, table(, labels(alpha "Sig. level"))
```

Sample sizes for a two-sample means test in a table with only significance level, power, group sample sizes, and null and alternative means

```
power twomeans 2 3, n(30(10)60) table(alpha power N1 N2 m1 m2)
```

Same as above, but display power estimates with only two decimal points

```
power twomeans 2 3, n(30(10)60) ///
  table(alpha power N1 N2 m1 m2, formats(power "%3.2f"))
```

Menu

Statistics > Power, precision, and sample size

Syntax

Produce default table

```
power ..., table ...
```

Suppress table

```
power ..., notable ...
```

Produce custom table

```
power ..., table([colspec] [, tableopts]) ...
```

where *colspec* is

```
column[:label] [column[:label] [...]]
```

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

tableopts	Description
Table	
`add`	add *columns* to the default table
`labels(`*labspec*`)`	change default labels for specified columns; default labels are column names
`widths(`*widthspec*`)`	change default column widths; default is specific to each column
`formats(`*fmtspec*`)`	change default column formats; default is specific to each column
`noformat`	do not use default column formats
`separator(#)`	draw a horizontal separator line every # lines; default is `separator(0)`, meaning no separator lines
`divider`	draw divider lines between columns
`byrow`	display rows as computations are performed; seldom used
`noheader`	suppress table header; seldom used
`continue`	draw a continuation border in the table output; seldom used

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`noheader` and `continue` are not shown in the dialog box.

column	Description
`alpha`	significance level
`power`	power
`beta`	type-II-error probability
`N`	total number of subjects
`N1`	number of subjects in the control group
`N2`	number of subjects in the experimental group
`nratio`	ratio of sample sizes, experimental to control
`K`	number of clusters
`K1`	number of clusters in the control group
`K2`	number of clusters in the experimental group
`kratio`	ratio of numbers of clusters, experimental to control
`M`	cluster size
`M1`	cluster size in the control group
`M2`	cluster size in the experimental group
`mratio`	ratio of cluster sizes, experimental to control
`delta`	effect size
`target`	target parameter
`_all`	display all supported columns
method_columns	columns specific to the method specified with `power`

By default, the following columns are displayed:

`alpha` and `power` are always displayed;
`N` is always displayed except for two-sample methods in a CRD;
`N1` and `N2` are displayed for two-sample methods except for a CRD;
`kratio` and `mratio` are available for two-sample methods in a CRD;
`delta` is displayed when defined by the method;
additional columns specific to each `power` method may be displayed.

Suboptions

The following are suboptions within the `table()` option of the `power` command.

Table

`add` requests that the columns specified in *colspec* be added to the default table. The columns are added to the end of the table.

`labels(`*labspec*`)` specifies the labels to be used in the table for the specified columns. *labspec* is

column `"`*label*`"` [*column* `"`*label*`"` [. . .]]

`labels()` takes precedence over the specification of column labels in *colspec*.

`widths(`*widthspec*`)` specifies column widths. The default values are the widths of the default column formats plus one. If the `noformat` option is used, the default for each column is nine. The column widths are adjusted to accommodate longer column labels and larger format widths. *widthspec* is either a list of values including missing values (*numlist*) or

column # [*column* # [. . .]]

For the value-list specification, the number of specified values may not exceed the number of columns in the table. A missing value (`.`) may be specified for any column to indicate the default width. If fewer widths are specified than the number of columns in the table, the last width specified is used for the remaining columns.

The alternative column-list specification provides a way to change widths of specific columns.

`formats(`*fmtspec*`)` specifies column formats. The default is `%7.0gc` for integer-valued columns and `%7.4g` for real-valued columns. *fmtspec* is either a string value-list of formats that may include empty strings or a column list:

column "*fmt*" [*column* "*fmt*" [...]]

For the value-list specification, the number of specified values may not exceed the number of columns in the table. An empty string (`""`) may be specified for any column to indicate the default format. If fewer formats are specified than the number of columns in the table, the last format specified is used for the remaining columns.

The alternative column-list specification provides a way to change formats of specific columns.

`noformat` requests that the default formats not be applied to the column values. If this suboption is specified, the column values are based on the column width.

`separator(#)` specifies how often separator lines should be drawn between rows of the table. The default is `separator(0)`, meaning that no separator lines should be displayed.

`divider` specifies that divider lines be drawn between columns. The default is no dividers.

`byrow` specifies that table rows be displayed as computations are performed. By default, the table is displayed after all computations are performed. This suboption may be useful when the computation of each row of the table takes a long time.

The following suboptions are available but are not shown in the dialog box:

`noheader` prevents the table header from displaying. This suboption is useful when the command is issued repeatedly, such as within a loop.

`continue` draws a continuation border at the bottom of the table. This suboption is useful when the command is issued repeatedly, such as within a loop.

Remarks and examples

Remarks are presented under the following headings:

`power, table` displays results from the `power` command in a table. This is useful for sensitivity analysis, which investigates the effect of varying study parameters on power, sample size, or other components of the study. The true values of study parameters are usually unknown. PSS analysis uses best guesses for these values. It is important to evaluate the sensitivity of the computed power or sample size to the chosen values of study parameters. For example, to evaluate variability of power values, you can compute powers for various ranges of values for the parameters of interest and display the resulting powers in a table or plot them on a graph (see [PSS-2] **power, graph**).

Using power, table

If you specify the `table` option or include more than one element in command arguments or in options allowing multiple values, the `power` command displays results in a tabular form. If desired, you can suppress the table by specifying the `notable` option. The `table` option is useful if you are producing graphical output or if you are producing results one case at a time, such as within a loop, and wish to display results in a table; see example 4 below.

Each method specified with the `power` command has its own default table. Among the columns that are always included in the default table are significance level (`alpha`), power (`power`), total sample size (`N`)—except for two-sample methods in a cluster randomized design (CRD)—and effect size (`delta`)—if effect size is defined by the method.

Depending on the method and study design, additional columns are also included by default. One-sample methods in a CRD include the number of clusters (`K`) and cluster size (`M`). Two-sample methods in an individual-level design include the sample sizes of the control and experimental groups (`N1` and `N2`). Two-sample methods in a CRD include the numbers of clusters of the control and experimental groups (`K1` and `K2`) and the cluster sizes of the two groups (`M1` and `M2`).

You can build your own table by specifying the columns and, optionally, their labels in the `table()` option. You can also add columns to the default table by specifying `add` within `power`'s `table()` option. The columns are displayed in the order they are specified. Each method provides its own list of supported columns; see the description of the `table()` option for each method. You can further customize the table by specifying various suboptions within `power`'s `table()` option.

The default column labels are the column names. You can provide your own column labels in *colspec* or by specifying `table()`'s suboption `labels()`. Labels containing spaces should be enclosed in quotes, and labels containing quotes should be enclosed in compound quotes. The `labels()` suboption is useful for changing the labels of existing columns; see example 2 below for details.

The default formats are `%7.4g` for real-valued columns and `%7.0gc` for integer-valued columns. If the `noformat` suboption is specified, the default column widths are nine characters. You can use `formats()` to change the default column formats and `widths()` to change the default column widths. The `formats()` and `widths()` suboptions provide two alternative specifications, a value-list specification or a column-list specification. The value-list specification accepts a list of values—strings for formats and numbers for widths—corresponding to each column of the displayed table. Empty strings (`""`) for formats and missing values (`.`) for widths are allowed and denote the default values. It is an error to specify more values than the number of displayed columns. If fewer values are specified, then the last value specified is used for the remaining columns. The column-list specification includes a list of pairs containing a column name followed by the corresponding value of the format or width. This specification is useful if you want to modify the formats or the widths of only selected columns. For column labels or formats exceeding the default column width, the widths of the respective columns are adjusted to accommodate the column labels and the specified formats.

If you specify the `noformat` suboption, the default formats are ignored, and the format of a column is determined by the column width: if the column width is #, the displayed format is `%(#-2).#g`. For example, if the column width is 9, the displayed format is `%7.0g`.

You may further customize the look of the table by using `separator(#)` to include separator lines after every # lines and by using the `divider` suboption to include divider lines between columns.

The `noheader` and `continue` suboptions are useful when you are building your own table within a loop; see example 4 in *Custom tables*.

In what follows, we demonstrate the default and custom tables of the results from power and sample-size analysis for a two-sided 5%-level one-sample t test comparing the population mean with a hypothesized value; see [PSS-2] **power onemean**.

Default tables

If there is only one set of results, the `power` command displays those results as text. When the `power` command has multiple sets of results, they are automatically displayed in a table. You can also specify the `table` option at any time to request that results be displayed in a table.

The displayed columns are specific to the chosen method of analysis and to the options specified with the command. The columns that always appear in the table include the significance level (`alpha`), power (`power`), and total sample size (`N`). If the concept of effect size is defined for the chosen method, the effect size (`delta`) is also displayed in the default table.

▷ Example 1: Default tables from power onemean

Suppose we want to explore the requirements on the sample size for a one-sample mean comparison test to detect means of different magnitudes. For simplicity, we consider only two target mean values, 1 and 2, and keep all other study parameters of `power onemean` at their default values: power at 80%, two-sided significance level at 0.05, and the estimate of the population standard deviation at 1. We use a zero null value of the mean. See [PSS-2] **power onemean** for details.

```
. power onemean 0 (1 2)
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0
```

alpha	power	N	delta	m0	ma	sd
.05	.8	10	1	0	1	1
.05	.8	5	2	0	2	1

As we mentioned earlier, the `alpha`, `power`, `N`, and `delta` columns are usually displayed in the default table. The `power onemean` command additionally displays columns containing the null mean, the alternative mean, and the standard deviation.

If we need to account for finite population, we can specify the `fpc()` option with `power onemean`. The default table will contain an additional column, `fpc`.

```
. power onemean 0 (1 2), fpc(500)
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0
```

alpha	power	N	delta	m0	ma	sd	fpc
.05	.8	10	1	0	1	1	500
.05	.8	5	2	0	2	1	500

If there is only one set of results, the power command displays those results as text. If desired, you can request a table by specifying the table option.

```
. power onemean 0 1, table
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0
```

alpha	power	N	delta	m0	ma	sd
.05	.8	10	1	0	1	1

Modifying default tables

We can modify labels, widths, and formats of the default columns by specifying the corresponding suboptions within the table() option. We can also add columns to the default table by using table()'s suboption add.

Example 2: Modifying default tables from power onemean

We can change the default labels of all or selected columns by using the labels() suboption within power's table() option. For example, we can change the labels of the sample-size columns and standard deviation columns of the first table in example 1 to "Sample size" and "Std. dev.", respectively.

```
. power onemean 0 (1 2), table(, labels(N "Sample size" sd "Std. dev."))
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0
```

alpha	power	Sample size	delta	m0	ma	Std. dev.
.05	.8	10	1	0	1	1
.05	.8	5	2	0	2	1

We can also change default column formats and widths by using the formats() and widths() suboptions.

```
. power onemean 0 (1 2), n(5) table(, labels(N "Sample size" sd "Std. dev.")
> widths(N 14 sd 14) formats(power "%7.5f"))
Estimated power for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0
```

alpha	power	Sample size	delta	m0	ma	Std. dev.
.05	0.40139	5	1	0	1	1
.05	0.90888	5	2	0	2	1

For this table, we switched from a sample-size determination to power determination by specifying the n() option. We changed the default column widths of the sample-size columns and standard deviation columns to 14. We also changed the default %7.4g format of the power column to %7.5f.

We can add columns to the default table by listing the names of the columns in the `table()` option and specifying its suboption `add`.

```
. power onemean 0 (1 2), table(diff, add)
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0
```

alpha	power	N	delta	m0	ma	sd	diff
.05	.8	10	1	0	1	1	1
.05	.8	5	2	0	2	1	2

We added the column `diff`, which contains the difference between the alternative and hypothesized values of the mean, to the default table produced by `power onemean`.

◁

Custom tables

We can use the `table()` option to build custom tables, which contain the columns you want in the order you want. You can also build a table within a `foreach` or `forvalues` loop as we demonstrate in example 4 below. This is useful in the case when you want to obtain multiple sets of results over parameters of the `power` command that do not allow the *numlist* specification.

▷ Example 3: Producing custom tables

As an example of a custom table, we produce a table containing only four columns: significance level, power, sample size, and effect size.

```
. power onemean 0 (1 2), table(alpha:"Significance level"
> power:Power N:"Sample size" delta:"Effect size", widths(. 15))
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0
```

Significance level	Power	Sample size	Effect size
.05	.8	10	1
.05	.8	5	2

To improve the look of the table, we also specified the `widths(. 15)` suboption to increase the column widths of the last 3 columns to 15, leaving the width of the first column, the significance level, at its default value.

We can use the _all qualifier to request that all table columns supported by the method be displayed.

```
. power onemean 0 (1 2), table(_all)
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0
```

alpha	power	beta	N	delta	m0	ma	diff	sd
.05	.8	.2	10	1	0	1	1	1
.05	.8	.2	5	2	0	2	2	1

◁

▷ Example 4: Building table using a loop

Some options of power commands may not allow the *numlist* specification. In this case, you can build a table manually by using a loop such as foreach (see [P] **foreach**) or forvalues (see [P] **forvalues**) loop. One way to do this is to write a program that loops over parameters of interest. We demonstrate a program that loops over varying values of the alternative mean of power onemean. You can easily adapt this program to meet your needs.

```
program dotable
        args ma
        numlist "`ma'"                  // expand the numeric list in macro ma
        local ma "`r(numlist)'"
        local nvals : list sizeof ma
        local i 1
        foreach val of local ma {       // loop over numeric values in ma
                if (`i'==1) {
                        power onemean 0 `val', table(, continue)
                }
                else if (`i'<`nvals') {
                        power onemean 0 `val', table(, noheader continue) notitle
                }
                else {
                        power onemean 0 `val', table(, noheader) notitle
                }
                local ++i
        }
end
```

The dotable program accepts one argument, ma, which may contain one or more numeric values of the alternative mean specified as *numlist*. The program uses combinations of continue, noheader, and notitle to display a table. The first call to power onemean requests that the table be displayed without the bottom line by specifying the continue suboption within table(). The subsequent calls (except the last) specify the continue suboption, the notitle option with power onemean, and noheader within the table() option to request that neither the output before the table nor the table header be displayed. The last call omits the continue suboption so that the bottom line is displayed.

As a result, we obtain the following table:

```
. dotable "1(1)4"
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0
```

alpha	power	N	delta	m0	ma	sd
.05	.8	10	1	0	1	1
.05	.8	5	2	0	2	1
.05	.8	4	3	0	3	1
.05	.8	3	4	0	4	1

Stored results

`power, table` stores the following in `r()` in addition to other results stored by `power`:

Scalars

`r(separator)`	number of lines between separator lines in the table
`r(divider)`	`1` if `divider` is requested in the table, `0` otherwise

Macros

`r(columns)`	displayed table columns
`r(labels)`	table column labels
`r(widths)`	table column widths
`r(formats)`	table column formats

Matrices

`r(pss_table)`	table of results

Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, graph** — Graph results from the power command

power onemean — Power analysis for a one-sample mean test

Description

`power onemean` computes sample size, power, or target mean for a one-sample mean test. By default, it computes sample size for given power and the values of the mean parameters under the null and alternative hypotheses. Alternatively, it can compute power for given sample size and values of the null and alternative means or the target mean for given sample size, power, and the null mean. For power and sample-size analysis in a cluster randomized design, see [PSS-2] **power onemean, cluster**. Also see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

For precision and sample-size analysis for a CI for a population mean, see [PSS-3] **ciwidth onemean**.

Quick start

Sample size for two-sided test of H_0: $\mu = 10$ versus H_a: $\mu \neq 10$ with null mean $m_0 = 10$, alternative mean $m_a = 15$, and standard deviation of 12 using default power of 0.8 and significance level $\alpha = 0.05$

```
power onemean 10 15, sd(12)
```

Same as above, but for a one-sided test with power of 0.9

```
power onemean 10 15, sd(12) power(.9) onesided
```

Same as above, but specified as m_0 and difference $m_a - m_0 = 5$

```
power onemean 10, sd(12) power(.9) onesided diff(5)
```

Power for a sample size of 75

```
power onemean 10 15, sd(12) n(75)
```

Power for sample sizes of 50, 60, 70, and 80

```
power onemean 10 15, sd(12) n(50(10)80)
```

Same as above, but display results in a graph of power versus sample size

```
power onemean 10 15, sd(12) n(50(10)80) graph
```

Effect size and target mean for $m_0 = 10$ with standard deviation of 4, for a sample size of 40, power of 0.9, and $\alpha = 0.01$

```
power onemean 10, sd(4) n(40) power(.9) alpha(.01)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
power onemean m0 ma [ , power(numlist) options ]
```

Compute power

```
power onemean m0 ma , n(numlist) [ options ]
```

Compute effect size and target mean

```
power onemean m0 , n(numlist) power(numlist) [ options ]
```

where m_0 is the null (hypothesized) mean or the value of the mean under the null hypothesis and m_a is the alternative (target) mean or the value of the mean under the alternative hypothesis. m_0 and m_a may each be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

options	Description
Main	
* `alpha(`*numlist*`)`	significance level; default is `alpha(0.05)`
* `power(`*numlist*`)`	power; default is `power(0.8)`
* `beta(`*numlist*`)`	probability of type II error; default is `beta(0.2)`
* `n(`*numlist*`)`	sample size; required to compute power or effect size
`nfractional`	allow fractional sample size
* `diff(`*numlist*`)`	difference between the alternative mean and the null mean, $m_a - m_0$; specify instead of the alternative mean m_a
* `sd(`*numlist*`)`	standard deviation; default is `sd(1)`
`knownsd`	request computation assuming a known standard deviation; default is to assume an unknown standard deviation
* `fpc(`*numlist*`)`	finite population correction (FPC) as a sampling rate or as a population size
`direction(upper|lower)`	direction of the effect for effect-size determination; default is `direction(upper)`, which means that the postulated value of the parameter is larger than the hypothesized value
`onesided`	one-sided test; default is two sided
`parallel`	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
`[no]table[(`*tablespec*`)]`	suppress table or display results as a table; see [PSS-2] **power, table**
`saving(`*filename* `[, replace])`	save the table data to *filename*; use `replace` to overwrite existing *filename*
Graph	
`graph[(`*graphopts*`)]`	graph results; see [PSS-2] **power, graph**
Iteration	
`init(#)`	initial value for sample size or mean; default is to use normal approximation
`iterate(#)`	maximum number of iterations; default is `iterate(500)`
`tolerance(#)`	parameter tolerance; default is `tolerance(1e-12)`
`ftolerance(#)`	function tolerance; default is `ftolerance(1e-12)`
`[no]log`	suppress or display iteration log
`[no]dots`	suppress or display iterations as dots
`cluster`	perform computations for a CRD; see [PSS-2] **power onemean, cluster**
`notitle`	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the `parallel` option.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`cluster` and `notitle` do not appear in the dialog box.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`N`	number of subjects	N
`delta`	effect size	δ
`m0`	null mean	μ_0
`ma`	alternative mean	μ_a
`diff`	difference between the alternative and null means	$\mu_a-\mu_0$
`sd`	standard deviation	σ
`fpc`	FPC as population size	N_{pop}
	FPC as sampling rate	γ
`target`	target parameter; synonym for `ma`	
`_all`	display all supported columns	

Column `beta` is shown in the default table in place of column `power` if specified.

Columns `diff` and `fpc` are shown in the default table if specified.

Options

Main

`alpha()`, `power()`, `beta()`, `n()`, `nfractional`; see [PSS-2] **power**. The `nfractional` option is allowed only for sample-size determination.

`diff(`*numlist*`)` specifies the difference between the alternative mean and the null mean, $m_a - m_0$. You can specify either the alternative mean m_a as a command argument or the difference between the two means in `diff()`. If you specify `diff(#)`, the alternative mean is computed as $m_a = m_0 + \#$. This option is not allowed with the effect-size determination.

`sd(`*numlist*`)` specifies the sample standard deviation or the population standard deviation. The default is `sd(1)`. By default, `sd()` specifies the sample standard deviation. If `knownsd` is specified, `sd()` specifies the population standard deviation.

`knownsd` requests that the standard deviation be treated as known in the computation. By default, the standard deviation is treated as unknown, and the computation is based on a t test, which uses a Student's t distribution as a sampling distribution of the test statistic. If `knownsd` is specified, the computation is based on a z test, which uses a normal distribution as the sampling distribution of the test statistic.

`fpc(`*numlist*`)` requests that a finite population correction be used in the computation. If `fpc()` has values between 0 and 1, it is interpreted as a sampling rate, n/N, where N is the total number of units in the population. When sample size n is specified, if `fpc()` has values greater than n, it is interpreted as a population size, but it is an error to have values between 1 and n. For sample-size determination, `fpc()` with a value greater than 1 is interpreted as a population size. It is an error for `fpc()` to have a mixture of sampling rates and population sizes.

direction(), onesided, parallel; see [PSS-2] **power**.

Table

table, table(), notable; see [PSS-2] **power, table**.

saving(); see [PSS-2] **power**.

Graph

graph, graph(); see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

init(#) specifies the initial value of the sample size for the sample-size determination or the initial value of the mean for the effect-size determination. The default is to use a closed-form normal approximation to compute an initial value of the sample size or mean.

iterate(), tolerance(), ftolerance(), log, nolog, dots, nodots; see [PSS-2] **power**.

The following options are available with power onemean but are not shown in the dialog box:

cluster; see [PSS-2] **power onemean, cluster**.

notitle; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the power onemean command and the methodology for power and sample-size analysis for a one-sample mean test. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis, and see [PSS-2] **power** for a general introduction to the power command using hypothesis tests. Also see [PSS-2] **power onemean, cluster** for power and sample-size analysis in a cluster randomized design.

Introduction

There are many examples of studies where a researcher would like to compare an observed mean with a hypothesized mean. A company that provides preparatory classes for a standardized exam would like to see if the mean score of students who take its classes is higher than the national average. A fitness center would like to know if its average clients' weight loss is greater than zero after six months. Or a government agency would like to know if a job training program results in higher wages than the national average.

This entry describes power and sample-size analysis for the inference about the population mean performed using hypothesis testing. Specifically, we consider the null hypothesis H_0: $\mu = \mu_0$ versus the two-sided alternative hypothesis H_a: $\mu \neq \mu_0$, the upper one-sided alternative H_a: $\mu > \mu_0$, or the lower one-sided alternative H_a: $\mu < \mu_0$.

The considered one-sample mean tests rely on the assumption that a random sample is normally distributed or that the sample size is large. Different test statistics can be based on whether the variance of the sampling process is known a priori. In the case of a known variance, the test statistic follows a standard normal distribution under the null hypothesis, and the corresponding test is known as a z test. In the case of an unknown variance, an estimate of the population variance is used to form a test statistic, which follows a Student's t distribution under the null hypothesis, and the corresponding test is known as a t test.

The random sample is typically drawn from an infinite population. When the sample is drawn from a population of a fixed size, sampling variability must be adjusted for a finite population size.

The `power onemean` command provides power and sample-size analysis for the comparison of a mean with a reference value using a t test or a z test.

Using power onemean

`power onemean` computes sample size, power, or target mean for a one-sample mean test. All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option. You can specify the `onesided` option to request a one-sided test.

By default, all computations are based on a t test, which assumes an unknown standard deviation, and use the default value of 1 as the estimate of the standard deviation. You may specify other values for the standard deviation in the `sd()` option. For a known standard deviation, you can specify the `knownsd` option to request a z test.

To compute sample size, you must specify the means under the null and alternative hypotheses, m_0 and m_a, respectively, and, optionally, the power of the test in the `power()` option. The default power is set to 0.8.

To compute power, you must specify the sample size in the `n()` option and the means under the null and alternative hypotheses, m_0 and m_a, respectively.

Instead of the alternative mean, m_a, you may specify the difference $m_a - m_0$ between the alternative mean and the null mean in the `diff()` option when computing sample size or power.

To compute effect size, the standardized difference between the alternative and null means, and the corresponding target mean, you must specify the sample size in the `n()` option, the power in the `power()` option, the null mean m_0, and, optionally, the direction of the effect. The direction is upper by default, `direction(upper)`, which means that the target mean is assumed to be larger than the specified null mean value. This is also equivalent to the assumption of a positive effect size. You can change the direction to lower, which means that the target mean is assumed to be smaller than the specified null value, by specifying the `direction(lower)` option. This is equivalent to assuming a negative effect size.

By default, the computed sample size is rounded up. You can specify the `nfractional` option to see the corresponding fractional sample size; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. The `nfractional` option is allowed only for sample-size determination.

Some of power onemean's computations require iteration. For example, when the standard deviation is unknown, computations use a noncentral Student's t distribution. Its degrees of freedom depends on the sample size, and the noncentrality parameter depends on the sample size and effect size. Therefore, the sample-size and effect-size determinations require iteration. The default initial values of the estimated parameters are obtained by using a closed-form normal approximation. They may be changed by specifying the init() option. See [PSS-2] **power** for the descriptions of other options that control the iteration procedure.

All computations assume an infinite population. For a finite population, use the fpc() option to specify a sampling rate or a population size. When this option is specified, a finite population correction is applied to the population standard deviation. The correction factor depends on the sample size; therefore, computing sample size for a finite population requires iteration even for a known standard deviation. The initial value for the sample size is based on the corresponding sample size assuming an infinite population.

In the following sections, we describe the use of power onemean accompanied by examples for computing sample size, power, and target mean.

Computing sample size

To compute sample size, you must specify the means under the null and alternative hypotheses, m_0 and m_a, respectively, and, optionally, the power of the test in the power() option. A default power of 0.8 is assumed if power() is not specified.

▷ Example 1: Sample size for a one-sample mean test

Consider an example from Tamhane and Dunlop (2000, 209) that discusses the effectiveness of coaching programs in improving the verbal part of SAT scores. Previous studies found that students retaking the SAT exams without any coaching program improve their scores by 15 points on average with a standard deviation of about 40 points. A new coaching program claims to improve the SAT scores by 40 points above the average. The changes in scores are assumed to be approximately normally distributed. The parameter of interest in this example is the mean change in the test scores. To test the claim, investigators wish to conduct another study and compute the sample size that is required to detect a mean change in scores of 40 points with 80% power using a 5%-level two-sided test. We assume that the true population standard deviation is unknown and use its estimate from previous studies to compute the sample size:

```
. power onemean 15 40, sd(40)

Performing iteration ...

Estimated sample size for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.6250
           m0 =   15.0000
           ma =   40.0000
           sd =   40.0000

Estimated sample size:

            N =        23
```

We find that a sample of 23 subjects is required to detect a shift of 40 points in average SAT scores given the standard deviation of 40 points with 80% power using a 5%-level two-sided test.

As we mentioned in *Using power onemean* and as is also indicated in the output, sample-size computation requires iteration when the standard deviation is unknown. The iteration log is suppressed by default, but you can display it by specifying the `log` option.

◁

▷ Example 2: Specifying difference between means

Instead of the alternative mean change of 40 as in example 1, we can specify the difference of 25 between the mean changes in scores under the alternative and null hypotheses in the `diff()` option and obtain the same results.

```
. power onemean 15, diff(25) sd(40)
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.6250
           m0 =   15.0000
           ma =   40.0000
         diff =   25.0000
           sd =   40.0000
Estimated sample size:
            N =        23
```

When we specify the `diff()` option, the difference between the alternative and null values is also reported in the output.

◁

▷ Example 3: Known variance

If we know the population standard deviation, we can use the `knownsd` option to request a z test.

```
. power onemean 15 40, sd(40) knownsd
Performing iteration ...
Estimated sample size for a one-sample mean test
z test
H0: m = m0  versus  Ha: m != m0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.6250
           m0 =   15.0000
           ma =   40.0000
           sd =   40.0000
Estimated sample size:
            N =        21
```

The output now indicates that the computation is based on a z test instead of a t test. We find that a smaller sample of 21 subjects is required to detect the same effect size as in example 1 when the standard deviation is known.

◁

Computing power

To compute power, you must specify the sample size in the `n()` option and the means under the null and alternative hypotheses, m_0 and m_a, respectively.

▷ Example 4: Power of a one-sample mean test

Continuing with example 1, we will suppose that we are designing a new study and anticipate to obtain a sample of 30 subjects. To compute the power corresponding to this sample size given the study parameters from example 1, we specify the sample size of 30 in the `n()` option:

```
. power onemean 15 40, n(30) sd(40)

Estimated power for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0

Study parameters:

        alpha =    0.0500
            N =        30
        delta =    0.6250
           m0 =   15.0000
           ma =   40.0000
           sd =   40.0000

Estimated power:

        power =    0.9112
```

With a larger sample size, the power of the test increases to about 91.12%.

◁

▷ Example 5: Multiple values of study parameters

To investigate the effect of a finite population size on power, we can specify a list of population sizes in the `fpc()` option:

```
. power onemean 15 40, n(30) sd(40) fpc(100 500 1000)

Estimated power for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0
```

alpha	power	N	delta	m0	ma	sd	fpc
.05	.9769	30	.625	15	40	40	100
.05	.9267	30	.625	15	40	40	500
.05	.919	30	.625	15	40	40	1000

As expected, when the population size increases, the power tends to get closer to that obtained by assuming an infinite population size.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

▷ Example 6: Reproducing published results from a text book

We can also reproduce the example from Tamhane and Dunlop (2000, 213–214). The authors consider a one-sided test with a 0.132 significance level and a known standard deviation and compute the power to be 95.3%. We can replicate their example by typing

```
. power onemean 15 40, n(20) sd(40) alpha(0.132) onesided knownsd

Estimated power for a one-sample mean test
z test
H0: m = m0  versus  Ha: m > m0

Study parameters:

        alpha =    0.1320
            N =        20
        delta =    0.6250
           m0 =   15.0000
           ma =   40.0000
           sd =   40.0000

Estimated power:

        power =    0.9533
```

◁

Computing effect size and target mean

Effect size δ for a one-sample mean test is defined as the ratio of the difference between the alternative and null values of the mean to the standard deviation, $\delta = (\mu_a - \mu_0)/\sigma$.

Sometimes, we may be interested in determining the smallest effect and the corresponding alternative or target mean that yield a statistically significant result for prespecified sample size and power. In this case, power, sample size, and null mean must be specified. In addition, you must also decide on the direction of the effect: upper, which means $\mu_a > \mu_0$, or lower, which means $\mu_a < \mu_0$. The direction may be specified in the `direction()` option; `direction(upper)` is the default.

▷ Example 7: Minimum detectable value of the mean change in SAT scores

Continuing with example 4, we may also be interested to find the smallest mean change in SAT scores that can be detected with a power of 80% given a sample of 30 subjects. To compute this, we specify only the null value of 15 as the command argument and also specify the sample size and power in the `n(30)` and `power(0.8)` options, respectively. We use the same value of 40 for the standard deviation as in example 4.

```
. power onemean 15, n(30) power(0.8) sd(40)

Performing iteration ...

Estimated target mean for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0; ma > m0

Study parameters:

        alpha =    0.0500
        power =    0.8000
            N =        30
           m0 =   15.0000
           sd =   40.0000

Estimated effect size and target mean:

        delta =    0.5292
           ma =   36.1694
```

The estimated smallest mean change in SAT scores is 36.17, which corresponds to the effect size of 0.53. Compared with example 1, for the same power of 80%, this example shows a smaller difference between the mean SAT scores of the two programs for a larger sample of 30 subjects.

In the above, we assumed the effect to be in the upper direction. By symmetry, the effect size in the lower direction will be −0.53, which can also be obtained by specifying `direction(lower)` in the above example.

◁

Performing hypothesis tests on mean

In this section, we briefly demonstrate the use of the `ttest` command for testing hypotheses about means; see [R] **ttest** for details. Suppose we wish to test the hypothesis that the mean is different from a reference value on the collected data. We can use the `ttest` command to do this. Below we demonstrate the use of this command for the analysis of `sat.dta`.

▷ Example 8: Testing for mean

Suppose that we wish to test whether the mean verbal SAT score is equal to 600. We use the `ttest` command to do this as follows:

```
. use https://www.stata-press.com/data/r19/sat
(Fictional SAT data)
. ttest score == 600
One-sample t test
```

Variable	Obs	Mean	Std. err.	Std. dev.	[95% conf.	interval]
score	75	504.8	15.24616	132.0356	474.4214	535.1786

```
    mean = mean(score)                                        t =  -6.2442
H0: mean = 600                                 Degrees of freedom =       74

   Ha: mean < 600               Ha: mean != 600                 Ha: mean > 600
 Pr(T < t) = 0.0000         Pr(|T| > |t|) = 0.0000          Pr(T > t) = 1.0000
```

We find statistical evidence to reject the null hypothesis of H_0: $\mu_{\text{SAT}} = 600$ versus a two-sided alternative H_a: $\mu_{\text{SAT}} \neq 600$ at the 5% significance level; the p-value < 0.0000.

We use the estimates based on this study to perform a sample-size analysis we would have conducted before the study.

```
. power onemean 600 505, sd(132)
Performing iteration ...
Estimated sample size for a one-sample mean test
t test
H0: m = m0  versus  Ha: m != m0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =   -0.7197
           m0 =  600.0000
           ma =  505.0000
           sd =  132.0000
Estimated sample size:
            N =        18
```

We find that the sample size required to detect a mean score of 505 with 80% power using a 5%-level two-sided test is only 18. The current sample contains 75 subjects, which would allow us to detect a potentially smaller (in absolute value) difference between the alternative and null means.

◁

Video examples

Sample-size calculation for comparing a sample mean to a reference value

Power calculation for comparing a sample mean to a reference value

Minimum detectable effect size for comparing a sample mean to a reference value

Stored results

`power onemean` stores the following in `r()`:

Scalars

`r(alpha)`	significance level
`r(power)`	power
`r(beta)`	probability of a type II error
`r(delta)`	effect size
`r(N)`	sample size
`r(nfractional)`	1 if `nfractional` is specified, 0 otherwise
`r(onesided)`	1 for a one-sided test, 0 otherwise
`r(m0)`	mean under the null hypothesis
`r(ma)`	mean under the alternative hypothesis
`r(diff)`	difference between the alternative and null means
`r(sd)`	standard deviation
`r(knownsd)`	1 if option `knownsd` is specified, 0 otherwise
`r(fpc)`	finite population correction (if specified)
`r(separator)`	number of lines between separator lines in the table
`r(divider)`	1 if `divider` is requested in the table, 0 otherwise
`r(init)`	initial value for sample size or mean
`r(maxiter)`	maximum number of iterations
`r(iter)`	number of iterations performed
`r(tolerance)`	requested parameter tolerance
`r(deltax)`	final parameter tolerance achieved
`r(ftolerance)`	requested distance of the objective function from zero
`r(function)`	final distance of the objective function from zero
`r(converged)`	1 if iteration algorithm converged, 0 otherwise

Macros

`r(type)`	`test`
`r(method)`	`onemean`
`r(direction)`	`upper` or `lower`
`r(columns)`	displayed table columns
`r(labels)`	table column labels
`r(widths)`	table column widths
`r(formats)`	table column formats

Matrices

`r(pss_table)`	table of results

Methods and formulas

Let $x_1, \ldots, x_n$ be a sequence of n independent and identically distributed random variables drawn from a normal population with mean μ and variance σ^2. Let

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \text{and} \qquad s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2$$

denote the sample mean and the sample variance, respectively. Let μ_0 and μ_a denote the null and alternative values of the mean parameter, respectively.

A one-sample mean test involves testing the null hypothesis $H_0\colon \mu = \mu_0$ versus the two-sided alternative hypothesis $H_a\colon \mu \neq \mu_0$, the upper one-sided alternative $H_a\colon \mu > \mu_0$, or the lower one-sided alternative $H_a\colon \mu < \mu_0$.

If the `nfractional` option is not specified, the computed sample size is rounded up.

The following formulas are based on Chow et al. (2018).

Methods and formulas are presented under the following headings:

Known standard deviation

In the case of a known standard deviation, the sampling distribution of the test statistic $z = \sqrt{n}(\overline{x} - \mu_0)/\sigma$ under the null hypothesis follows the standard normal distribution, and the corresponding test is known as a z test.

Let α be the significance level, β be the probability of a type II error, and $z_{1-\alpha}$ and z_β be the $(1-\alpha)$th and the βth quantiles of the standard normal distribution.

The power $\pi = 1 - \beta$ is computed using

$$\pi = \begin{cases} \Phi\left(\sqrt{n}\delta - z_{1-\alpha}\right) & \text{for an upper one-sided test} \\ \Phi\left(-\sqrt{n}\delta - z_{1-\alpha}\right) & \text{for a lower one-sided test} \\ \Phi\left(\sqrt{n}\delta - z_{1-\alpha/2}\right) + \Phi\left(-\sqrt{n}\delta - z_{1-\alpha/2}\right) & \text{for a two-sided test} \end{cases} \tag{1}$$

where $\Phi(\cdot)$ is the cdf of the standard normal distribution and $\delta = (\mu_a - \mu_0)/\sigma$ is the effect size.

The sample size n for a one-sided test is computed by inverting a one-sided power equation from (1):

$$n = \left(\frac{z_{1-\alpha} - z_\beta}{\delta}\right)^2 \tag{2}$$

Similarly, the absolute value of the effect size for a one-sided test is computed as follows:

$$|\delta| = \frac{(z_{1-\alpha} - z_\beta)}{\sqrt{n}} \tag{3}$$

Note that the magnitude of the effect size is the same regardless of the direction of the test.

The minimum detectable value of the mean for a one-sided test is computed as $\mu_a = \mu_0 + (z_{1-\alpha} - z_\beta)\sigma/\sqrt{n}$ when $\mu_a > \mu_0$ and as $\mu_a = \mu_0 - (z_{1-\alpha} - z_\beta)\sigma/\sqrt{n}$ when $\mu_a < \mu_0$.

Sample size and minimum detectable value of the mean for a two-sided test are computed iteratively using the two-sided power equation from (1). The initial values are obtained from (2) and (3), correspondingly, with $\alpha/2$ in place of α.

Unknown standard deviation

In the case of an unknown standard deviation, an unbiased estimator s is used in place of σ in the definition of a z test statistic. The sampling distribution of the test statistic $t = \sqrt{n}(\overline{x} - \mu_0)/s$ under the null hypothesis follows a Student's t distribution with $n-1$ degrees of freedom, and the corresponding test is known as a t test.

Let $t_{n-1,\alpha}$ denote the αth quantile of a Student's t distribution with $n-1$ degrees of freedom. Under the alternative hypothesis, the test statistic follows a noncentral Student's t distribution, and the power is computed using

$$\pi = \begin{cases} 1 - T_{n-1,\lambda}\left(t_{n-1,1-\alpha}\right) & \text{for an upper one-sided test} \\ T_{n-1,\lambda}\left(-t_{n-1,1-\alpha}\right) & \text{for a lower one-sided test} \\ 1 - T_{n-1,\lambda}\left(t_{n-1,1-\alpha/2}\right) + T_{n-1,\lambda}\left(-t_{n-1,1-\alpha/2}\right) & \text{for a two-sided test} \end{cases} \tag{4}$$

where $T_{n-1,\lambda}(\cdot)$ is the cumulative noncentral Student's t distribution with a noncentrality parameter $\lambda = \sqrt{n}\delta$.

Sample size and minimum detectable value of the mean are obtained by iteratively solving nonlinear equations in (4), for n and δ, respectively. The default initial values for the iterative procedure are calculated from (2) and (3), respectively, assuming a normal distribution.

Finite population size

The above formulas assume that the random sample is drawn from an infinite population. In cases when the size of the population is known, we need to make the following adjustment to the standard deviation,

$$\sigma_{\text{fpc}} = \sigma\sqrt{\left(1 - \frac{n}{N}\right)}$$

where σ_{fpc} is the population standard deviation adjusted for finite population size. The correction factor depends on the sample size; therefore, computing sample size in this case requires iteration. The initial value for the sample size is based on the corresponding normal approximation with infinite population size.

References

Chow, S.-C., J. Shao, H. Wang, and Y. Lokhnygina. 2018. *Sample Size Calculations in Clinical Research*. 3rd ed. Boca Raton, FL: CRC Press.

Tamhane, A. C., and D. D. Dunlop. 2000. *Statistics and Data Analysis: From Elementary to Intermediate*. Upper Saddle River, NJ: Prentice Hall.

Also see

[PSS-2] **power onemean, cluster** — Power analysis for a one-sample mean test, CRD

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-3] **ciwidth onemean** — Precision analysis for a one-mean CI

[PSS-5] **Glossary**

[ADAPT] **gsdesign onemean** — Group sequential design for a one-sample mean test

[R] **ttest** — t tests (mean-comparison tests)

power onemean, cluster — Power analysis for a one-sample mean test, CRD

Description

`power onemean, cluster` computes the number of clusters, cluster size, power, or target mean for a one-sample mean test in a cluster randomized design (CRD). It computes the number of clusters given cluster size, power, and the values of the null and alternative means. It also computes cluster size given the number of clusters, power, and the values of the null and alternative means. Alternatively, it computes power given the number of clusters, cluster size, and the values of the null and alternative means, or it computes the target mean given the number of clusters, cluster size, power, and the null mean. See [PSS-2] **power onemean** for a general discussion of power and sample-size analysis for a one-sample mean test. Also see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Quick start

Compute number of clusters for two-sided test of H_0: $\mu = 10$ versus H_a: $\mu \neq 10$ with null mean $m_0 = 10$, alternative mean $m_a = 15$, standard deviation of 12, and cluster size of 5, using default intraclass correlation of 0.5, power of 0.8, and significance level $\alpha = 0.05$

```
power onemean 10 15, m(5) sd(12)
```

Same as above, but with an intraclass correlation of 0.2

```
power onemean 10 15, m(5) sd(12) rho(0.2)
```

Same as above, but the cluster size varies with a coefficient of variation of 0.6

```
power onemean 10 15, m(5) sd(12) rho(0.2) cvcluster(0.6)
```

Compute cluster size when 30 clusters are sampled

```
power onemean 10 15, k(30) sd(12)
```

Power for 30 clusters with cluster size of 5

```
power onemean 10 15, k(30) m(5) sd(12)
```

Same as above, but for 30, 35, 40, 45, and 50 clusters and display results in a graph of power versus number of clusters

```
power onemean 10 15, k(30(5)50) m(5) sd(12) graph
```

Effect size and target mean for $m_0 = 10$ with standard deviation of 4, for 8 clusters with cluster size of 5, power of 0.9, and $\alpha = 0.01$

```
power onemean 10, k(8) m(5) power(0.9) sd(4) alpha(0.01)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute number of clusters

```
power onemean m0 ma , { m(numlist) | n(numlist) cluster } [ options ]
```

Compute cluster size

```
power onemean m0 ma , k(numlist) [ options ]
```

Compute power

```
power onemean m0 ma , k(numlist) { m(numlist) | n(numlist) } [ options ]
```

Compute effect size and target mean

```
power onemean m0 , k(numlist) { m(numlist) | n(numlist) } power(numlist) [ options ]
```

where m_0 is the null (hypothesized) mean or the value of the mean under the null hypothesis and m_a is the alternative (target) mean or the value of the mean under the alternative hypothesis. m_0 and m_a may each be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

options	Description
Main	
`cluster`	perform computations for a CRD; implied by `k()` or `m()`
* `alpha(`*numlist*`)`	significance level; default is `alpha(0.05)`
* `power(`*numlist*`)`	power; default is `power(0.8)`
* `beta(`*numlist*`)`	probability of type II error; default is `beta(0.2)`
* `k(`*numlist*`)`	number of clusters
* `m(`*numlist*`)`	cluster size
* `n(`*numlist*`)`	number of observations
`nfractional`	allow fractional number of clusters, cluster size, and sample size
* `diff(`*numlist*`)`	difference between the alternative mean and the null mean, $m_a - m_0$; specify instead of the alternative mean m_a
* `sd(`*numlist*`)`	standard deviation; default is `sd(1)`
* `rho(`*numlist*`)`	intraclass correlation; default is `rho(0.5)`
* `cvcluster(`*numlist*`)`	coefficient of variation for cluster sizes
`direction(upper|lower)`	direction of the effect for effect-size determination; default is `direction(upper)`, which means that the postulated value of the parameter is larger than the hypothesized value
`onesided`	one-sided test; default is two sided
`parallel`	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
`[no]table[(`*tablespec*`)]`	suppress table or display results as a table; see [PSS-2] **power, table**
`saving(`*filename*` [, replace])`	save the table data to *filename*; use `replace` to overwrite existing *filename*
Graph	
`graph[(`*graphopts*`)]`	graph results; see [PSS-2] **power, graph**
Iteration	
`init(#)`	initial value for number of clusters, cluster size, or mean
`iterate(#)`	maximum number of iterations; default is `iterate(500)`
`tolerance(#)`	parameter tolerance; default is `tolerance(1e-12)`
`ftolerance(#)`	function tolerance; default is `ftolerance(1e-12)`
`[no]log`	suppress or display iteration log
`[no]dots`	suppress or display iterations as dots
`notitle`	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see [U] **11.1.8 numlist**. Also see the `parallel` option.

`collect` is allowed; see [U] **11.1.10 Prefix commands**.

`notitle` does not appear in the dialog box.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`K`	number of clusters	K
`M`	cluster size	M
`N`	number of observations	N
`delta`	effect size	δ
`m0`	null mean	μ_0
`ma`	alternative mean	μ_a
`diff`	difference between the alternative and null means	$\mu_a-\mu_0$
`sd`	standard deviation	σ
`rho`	intraclass correlation	ρ
`CV_cluster`	coefficient of variation for cluster sizes	$\mathrm{CV}_{\mathrm{cl}}$
`target`	target parameter; synonym for `ma`	
`_all`	display all supported columns	

Column `beta` is shown in the default table in place of column `power` if specified.
Columns `diff` and `CV_cluster` are shown in the default table if specified.

Options

Main

`cluster` specifies that computations should be performed for a CRD. This option is implied when either the `k()` or `m()` option is specified. It is required if the `n()` option is used to compute the number of clusters.

`alpha()`, `power()`, `beta()`; see [PSS-2] **power**.

`k(`*numlist*`)` specifies the number of clusters. This option is required to compute the cluster size, power, or effect size.

`m(`*numlist*`)` specifies the cluster size. This option or the `n()` option is required to compute the number of clusters, power, or effect size. `m()` may contain noninteger values. In this case or if the `cvcluster()` option is specified, `m()` represents the average cluster size.

`n(`*numlist*`)` specifies the number of observations. This option or the `m()` option is required to compute the number of clusters, power, or effect size.

`nfractional`; see [PSS-2] **power**. The `nfractional` option is allowed when computing the number of clusters and cluster size to display fractional (without rounding) values of the number of clusters, cluster size, and sample size.

`diff()`, `sd()`; see [PSS-2] **power onemean**.

`rho(`*numlist*`)` specifies the intraclass correlation. The default is `rho(0.5)`.

`cvcluster(`*numlist*`)` specifies the coefficient of variation for cluster sizes. This option is used with varying cluster sizes.

`direction()`, `onesided`, `parallel`; see [PSS-2] **power**.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(`#`)` specifies the initial value for the number of clusters or cluster size for sample-size determination or the initial value for the mean for the effect-size determination. The default is to use a closed-form normal approximation to compute an initial value for the estimated parameter.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following option is available with `power onemean, cluster` but is not shown in the dialog box:

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

`power onemean, cluster` requests that computations for the `power onemean` command be done for a CRD. In a CRD, groups of subjects or clusters are randomized instead of individual subjects, so the sample size is determined by the number of clusters and the cluster size. The sample-size determination thus consists of the determination of the number of clusters given cluster size or the determination of cluster size given the number of clusters. For a general discussion of using `power onemean`, see [PSS-2] **power onemean**. The discussion below is specific to the CRD.

Using power onemean, cluster

If you specify the `cluster` option, include `k()` to specify the number of clusters or include `m()` to specify the cluster size, the `power onemean` command will perform computations for a one-sample mean test in a CRD.

All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option. You can specify the `onesided` option to request a one-sided test.

To compute the number of clusters, you must specify the means under the null and alternative hypotheses as command arguments m_0 and m_a, respectively, and specify the cluster size in the `m()` option. Instead of specifying the `m()` option, you may specify the sample size in the `n()` option and specify the `cluster` option, so that `power onemean` will perform its computation for a cluster randomized design instead of the default individual-level design. You may also specify the power of the test in the `power()` option.

To compute cluster size, you must specify the null mean m_0, the alternative mean m_a, and the number of clusters in the `k()` option. You may also specify the power of the test in the `power()` option.

To compute power, you must specify the number of clusters in the `k()` option, the cluster size in the `m()` option or the sample size in the `n()` option, the null mean m_0, and the alternative mean m_a.

Instead of the alternative mean m_a, you may specify the difference $m_a - m_0$ between the alternative mean and the null mean in the `diff()` option when computing the number of clusters, cluster size, or power.

The effect size δ is defined as the standardized difference between the alternative and null means. In a CRD, the effect size δ is also adjusted for the cluster design; see *Methods and formulas*.

To compute effect size and the corresponding target mean, you must specify the number of clusters in the `k()` option, the cluster size in the `m()` option or the sample size in the `n()` option, the power in the `power()` option, and the null mean m_0. You may also specify the direction of the effect in the `direction()` option. The direction is upper by default, `direction(upper)`; see *Using power onemean* in [PSS-2] **power onemean** for other details.

The computations for a CRD are based on a z test that relies on (asymptotic) normality of the data and assumes known standard deviation, which you may specify in the `sd()` option. Otherwise, the default value of one is used.

All computations assume an intraclass correlation of 0.5. You can change this by specifying the `rho()` option. Also, all clusters are assumed to be of the same size unless the coefficient of variation for cluster sizes is specified in the `cvcluster()` option.

By default, the computed number of clusters, cluster size, and sample size is rounded up. However, you can specify the `nfractional` option to see the corresponding fractional values; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. If the `cvcluster()` option is specified when computing cluster size, then cluster size represents the average cluster size and is thus not rounded. When sample size is specified in the `n()` option, fractional cluster size may be reported to accommodate the specified number of clusters and sample size.

Some of `power onemean, cluster`'s computations require iteration, such as to compute the number of clusters for a two-sided test; see *Methods and formulas* for details and [PSS-2] **power** for the descriptions of options that control the iteration procedure.

Computing number of clusters

To compute the number of clusters, you must specify the means under the null and alternative hypotheses as command arguments m_0 and m_a, respectively, and specify the cluster size in the `m()` option. Instead of specifying the `m()` option, you may specify the sample size in the `n()` option and specify the `cluster` option, so that `power onemean` will perform its computation for a cluster randomized design instead of the default individual-level design. You may also specify the power of the test in the `power()` option.

Example 1: Number of clusters for a one-sample mean test in a CRD, specifying cluster size

Consider an example from Tamhane and Dunlop (2000, 209) that discusses the effectiveness of coaching programs in improving the verbal part of SAT scores. Previous studies found that students retaking the SAT exams without any coaching program improve their scores by 15 points on average with a standard deviation of about 40 points. The population standard deviation is assumed to be 40.

Unlike Tamhane and Dunlop (2000, 209), we assume that students are sampled from a set of classes and that the scores of students from the same class are correlated. We plan on sampling 10 students from each class and assume that the intraclass correlation is 0.3.

A new coaching program claims to improve average SAT scores by 40 points. The changes in scores are assumed to be approximately normally distributed. The parameter of interest in this example is the mean change in the test scores. To test the claim, investigators wish to conduct another study and compute the number of classes that is required to detect a difference of 25 points when the mean change in scores increases from 15 points to 40 points with 80% power using a 5%-level two-sided test:

```
. power onemean 15 40, m(10) sd(40) rho(0.3)

Performing iteration ...

Estimated number of clusters for a one-sample mean test
Cluster randomized design, z test
H0: m = m0  versus  Ha: m != m0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.3249
           m0 =   15.0000
           ma =   40.0000
           sd =   40.0000

Cluster design:

            M =        10
          rho =    0.3000

Estimated number of clusters and sample size:

            K =         8
            N =        80
```

We find that given 10 students per class, 8 classes and thus a total of 80 students are required to detect a 25-point increase in the mean change in SAT scores given the standard deviation of 40 points with 80% power using a 5%-level two-sided test. The effect size (`delta`) is calculated using the given information about the null and alternative means, standard deviation, and cluster design; see *Methods and formulas*.

Example 2: Number of clusters for a one-sample mean test in a CRD, with varying cluster sizes

Instead of a constant number of students in each class as in example 1, we assume that the number of students selected from each class will vary. Suppose that the average of the class sizes is 10 and that the coefficient of variation for class sizes is 1.2. To account for varying cluster sizes, we specify the coefficient of variation in the `cvcluster()` option.

```
. power onemean 15 40, m(10) sd(40) rho(0.3) cvcluster(1.2)
Performing iteration ...
Estimated number of clusters for a one-sample mean test
Cluster randomized design, z test
H0: m = m0  versus  Ha: m != m0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.2868
           m0 =   15.0000
           ma =   40.0000
           sd =   40.0000
Cluster design:
    Average M =   10.0000
          rho =    0.3000
        CV_cl =    1.2000
Estimated number of clusters and sample size:
            K =        10
            N =       100
```

The required number of classes is 10 and total number of subjects is 100. When we compare this with the 8 classes required in example 1, we see that when the number of students per class varies, we need more classes and thus more students to achieve the same power.

◁

▷ Example 3: Number of clusters for a one-sample mean test in a CRD, specifying sample size

Suppose that for our study, we plan to recruit a total of 100 students, but we need to know the required number of classes and how many students to recruit in each class. In this case, we specify the `n(100)` option. Because neither the `k()` nor the `m()` option is specified, we also need to specify the `cluster` option for power computations to be performed for a CRD.

```
. power onemean 15 40, cluster n(100) sd(40) rho(0.3)
Performing iteration ...
Estimated number of clusters for a one-sample mean test
Cluster randomized design, z test
H0: m = m0  versus  Ha: m != m0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.2963
           m0 =   15.0000
           ma =   40.0000
           sd =   40.0000
Cluster design:
            N =       100
          rho =    0.3000
Estimated number of clusters and cluster size:
            K =         8
    Average M =   12.5000
```

To achieve the desired power, we need to recruit on average 12.5 students per class from 8 classes. `power onemean` did not round the cluster size of 12.5 to meet our requirement that there is a total of 100 students. In practice, you can decide to recruit 12 students from some classes and 13 from other classes to have roughly constant class sizes.

◁

Computing cluster size

To compute cluster size, you must specify the null mean m_0, the alternative mean m_a, and the number of clusters in the `k()` option. You may also specify the power of the test in the `power()` option.

▷ Example 4: Cluster size for a one-sample mean test in a CRD

Continuing with example 1, suppose that we are designing a new study but we would like to select 12 classes, and we need to know how many students to recruit from each class. Given other study parameters from example 1, we compute the required number of students by specifying 12 clusters in the `k()` option.

```
. power onemean 15 40, k(12) sd(40) rho(0.3)

Performing iteration ...

Estimated cluster size for a one-sample mean test
Cluster randomized design, z test
H0: m = m0  versus  Ha: m != m0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.4941
           m0 =   15.0000
           ma =   40.0000
           sd =   40.0000

Cluster design:

            K =        12
          rho =    0.3000

Estimated cluster size and sample size:

            M =         3
            N =        36
```

With 12 classes, we need to recruit 3 students per class for a total of 36 students.

◁

Computing power

To compute power, you must specify the number of clusters in the `k()` option, the cluster size in the `m()` option or the sample size in the `n()` option, the null mean m_0, and the alternative mean m_a.

▷ Example 5: Power for a one-sample mean test in a CRD

Continuing with example 4, suppose that we can recruit 10 students from each of the 12 classes and we want to compute power for this design. Given other study parameters from example 4, we compute the power by specifying 12 clusters in the `k()` option and the cluster size of 10 in the `m()` option:

```
. power onemean 15 40, k(12) m(10) sd(40) rho(0.3)
Estimated power for a one-sample mean test
Cluster randomized design, z test
H0: m = m0  versus  Ha: m != m0
Study parameters:
        alpha =    0.0500
        delta =    0.3249
           m0 =   15.0000
           ma =   40.0000
           sd =   40.0000
Cluster design:
            K =        12
            M =        10
            N =       120
          rho =    0.3000
Estimated power:
        power =    0.9451
```

The computed power is about 95%.

◁

▷ Example 6: Multiple values of study parameters

To investigate the effect of the number of clusters on power, we can specify a list of numbers in the `k()` option:

```
. power onemean 15 40, k(4(2)12) m(10) sd(40) rho(0.3) table(power K)
Estimated power for a one-sample mean test
Cluster randomized design, z test
H0: m = m0  versus  Ha: m != m0
```

power	K
.5379	4
.7112	6
.828	8
.9013	10
.9451	12

In this example, we also specified the `table(power K)` option to list the only two columns that vary. As expected, as the number of clusters increases, the power tends to get closer to 1.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

Computing effect size and target mean

The effect size δ is defined as the standardized difference between the alternative and null means. In a CRD, the effect size δ is also adjusted for the cluster design; see *Methods and formulas*.

To compute effect size and the corresponding target mean, you must specify the number of clusters in the `k()` option, the cluster size in the `m()` option or the sample size in the `n()` option, the power in the `power()` option, and the null mean m_0. You may also specify the direction of the effect in the `direction()` option. The direction is upper by default, `direction(upper)`; see *Using power onemean* in [PSS-2] **power onemean** for other details.

Example 7: Effect size for a one-sample mean test in a CRD

Continuing with example 5, we may be interested in finding the minimum value of the mean that can be detected with a sample of 12 classes and 10 students per class and 80% power. To compute this, we specify the null value of 15 as the command argument and the required options `k(12)`, `m(10)`, and `power(0.8)` and continue to use `sd(40)` and `rho(0.3)`.

```
. power onemean 15, k(12) m(10) power(0.8) sd(40) rho(0.3)

Performing iteration ...

Estimated target mean for a one-sample mean test
Cluster randomized design, z test
H0: m = m0  versus  Ha: m != m0; ma > m0

Study parameters:

        alpha =    0.0500
        power =    0.8000
           m0 =   15.0000
           sd =   40.0000

Cluster design:

            K =        12
            M =        10
            N =       120
          rho =    0.3000

Estimated effect size and target mean:

        delta =    0.2557
           ma =   34.6777
```

Given the null value of 15, the minimum detectable value of the mean is about 34.68, which corresponds to an effect size of about 0.26. The computed target mean is smaller than the alternative mean of 40 that was specified in previous examples. Because we use more classes here than the number of classes computed in example 1, use larger cluster size than that computed in example 4, and use lower power than that computed in example 5, we can detect a smaller effect size and thus smaller target mean.

Performing hypothesis tests on mean in a CRD

There are different ways to account for a CRD or for clustered data when performing hypothesis tests that compare the mean with a reference value. With large samples or when you know population standard deviation and intraclass correlation, the simplest way is to use a z test that accounts for clustered data.

In this section, we briefly demonstrate the `ztest` command for testing hypotheses about means with clustered data; see [R] **ztest** for details. We will use the same `sat.dta` we used in *Performing hypothesis tests on mean* in [PSS-2] **power onemean**. Here we consider that SAT scores of students from the same class may be correlated.

▷ Example 8: Testing for mean with clustered data

As in example 8 of [PSS-2] **power onemean**, suppose that we want to test whether the mean verbal SAT score is equal to 600 while accounting for the correlated scores within classes.

The `ztest` command provides the `cluster()` option to account for clustered data. In addition to the cluster identifier in `cluster()`, we must specify the population standard deviation in `sd()` and the population intraclass correlation in `rho()`. Suppose that the population standard deviation is 132 and the population intraclass correlation is 0.7.

```
. use https://www.stata-press.com/data/r19/sat
(Fictional SAT data)
. ztest score == 600, cluster(class) sd(132) rho(0.7)
One-sample z test                               Number of clusters =        15
Cluster variable: class                         Cluster size       =         5
                                                Intraclass corr.   =    0.7000
------------------------------------------------------------------------------
Variable |      Obs        Mean    Std. err.   Std. dev.   [95% conf. interval]
---------+--------------------------------------------------------------------
   score |       75       504.8    29.71222         132    446.5651    563.0349
------------------------------------------------------------------------------
    mean = mean(score)                                            z =  -3.2041
H0: mean = 600

   Ha: mean < 600               Ha: mean != 600                 Ha: mean > 600
 Pr(Z < z) = 0.0007         Pr(|Z| > |z|) = 0.0014          Pr(Z > z) = 0.9993
```

There is statistical evidence that the mean verbal SAT score is different from 600 (two-sided p-value $=$ 0.0014) and that it is actually smaller than 600 (lower one-sided p-value $=$ 0.0007).

If we were to design another similar study based on the estimates from this study, we could compute the required number of classes as follows:

```
. power onemean 600 505, sd(132) rho(0.7) m(5)
Performing iteration ...
Estimated number of clusters for a one-sample mean test
Cluster randomized design, z test
H0: m = m0  versus  Ha: m != m0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =   -0.3692
           m0 =  600.0000
           ma =  505.0000
           sd =  132.0000
Cluster design:
            M =         5
          rho =    0.7000
Estimated number of clusters and sample size:
            K =        12
            N =        60
```

Given 5 students per class, we need 12 classes and a total of 60 students to detect the difference between the null value of 600 and the alternative value of 505, assuming a standard deviation of 132 and an intraclass correlation of 0.7, with 80% power using a 5%-level two-sided test.

Compared with the required sample size of 18 students in example 8 of [PSS-2] **power onemean**, a sample size of 60 students is needed here to detect the same mean difference when observations are correlated.

◁

Stored results

`power onemean, cluster` stores the following in `r()`:

Scalars

`r(alpha)`	significance level
`r(power)`	power
`r(beta)`	probability of a type II error
`r(delta)`	effect size
`r(K)`	number of clusters
`r(M)`	cluster size
`r(N)`	number of subjects
`r(nfractional)`	1 if `nfractional` is specified, 0 otherwise
`r(onesided)`	1 for a one-sided test, 0 otherwise
`r(m0)`	mean under the null hypothesis
`r(ma)`	mean under the alternative hypothesis
`r(diff)`	difference between the alternative and null means
`r(sd)`	standard deviation
`r(rho)`	intraclass correlation
`r(CV_cluster)`	coefficient of variation for cluster sizes
`r(separator)`	number of lines between separator lines in the table
`r(divider)`	1 if `divider` is requested in the table, 0 otherwise
`r(init)`	initial value for estimated parameter
`r(maxiter)`	maximum number of iterations
`r(iter)`	number of iterations performed
`r(tolerance)`	requested parameter tolerance
`r(deltax)`	final parameter tolerance achieved
`r(ftolerance)`	requested distance of the objective function from zero
`r(function)`	final distance of the objective function from zero
`r(converged)`	1 if iteration algorithm converged, 0 otherwise

Macros

`r(type)`	`test`
`r(method)`	`onemean`
`r(design)`	`CRD`
`r(direction)`	`upper` or `lower`
`r(columns)`	displayed table columns
`r(labels)`	table column labels
`r(widths)`	table column widths
`r(formats)`	table column formats

Matrices

`r(pss_table)`	table of results

Methods and formulas

For the computation in a CRD, we assume that the standard deviation is known; see *Known standard deviation* under *Methods and formulas* in [PSS-2] **power onemean** for the common notation for a one-sample mean test.

Methods and formulas are presented under the following headings:

Equal cluster sizes

In a CRD, let K be the number of clusters, M be the number of observations in each cluster, and n be the total number of subjects, where $n = MK$. Let y_{ij} be the jth ($j = 1, 2, \ldots, M$) observation from the ith cluster ($i = 1, 2, \ldots, K$), which is assumed to follow a normal distribution with mean μ and variance σ^2. Let ρ be the intraclass correlation.

Let

$$\overline{y} = \frac{1}{n}\sum_{i=1}^{K}\sum_{j=1}^{M} y_{ij}$$

be the observed mean. Let μ_0 be the mean under the null hypothesis, and let DE be the design effect,

$$\text{DE} = 1 + \rho(M - 1)$$

The sampling distribution of the test statistic $z = \sqrt{n}(\overline{y} - \mu_0)/(\sigma\sqrt{\text{DE}})$ under the null hypothesis follows a standard normal distribution; see, for example, Ahn, Heo, and Zhang (2015). In a CRD, the observed mean has a variance of $\sigma^2\text{DE}$.

Let α be the significance level, β be the probability of a type II error, and $z_{1-\alpha}$ and z_β be the $(1-\alpha)$th and the βth quantiles of the standard normal distribution.

The power $\pi = 1 - \beta$ is computed using

$$\pi = \begin{cases} \Phi\left(\sqrt{n}\delta - z_{1-\alpha}\right) & \text{for an upper one-sided test} \\ \Phi\left(-\sqrt{n}\delta - z_{1-\alpha}\right) & \text{for a lower one-sided test} \\ \Phi\left(\sqrt{n}\delta - z_{1-\alpha/2}\right) + \Phi\left(-\sqrt{n}\delta - z_{1-\alpha/2}\right) & \text{for a two-sided test} \end{cases} \tag{1}$$

where $\Phi(\cdot)$ is the c.d.f. of the standard normal distribution and $\delta = (\mu_a - \mu_0)/(\sigma\sqrt{\text{DE}})$ is the effect size.

Given the cluster size, the number of clusters K for a one-sided test is computed by inverting a one-sided power equation from (1):

$$K = \left(\frac{z_{1-\alpha} - z_\beta}{\delta\sqrt{M}}\right)^2 \tag{2}$$

Given the number of clusters K, the cluster size M for a one-sided test is computed from (2), after substituting for δ,

$$M = \frac{1 - \rho}{\frac{K(\mu_a - \mu_0)^2}{\sigma^2\left(z_{1-\alpha} - z_\beta\right)^2} - \rho} \tag{3}$$

The absolute value of the effect size for a one-sided test is computed as follows:

$$|\delta| = \frac{(z_{1-\alpha} - z_{\beta})}{\sqrt{n}} \tag{4}$$

The minimum detectable value of the mean for a one-sided test is computed as
$\mu_a = \mu_0 + (z_{1-\alpha} - z_{\beta})\sigma\sqrt{\text{DE}}/\sqrt{n}$ when $\mu_a > \mu_0$ and as
$\mu_a = \mu_0 - (z_{1-\alpha} - z_{\beta})\sigma\sqrt{\text{DE}}/\sqrt{n}$ when $\mu_a < \mu_0$.

The number of clusters, cluster size, and minimum detectable value of the mean for a two-sided test are computed iteratively using the two-sided power equation from (1). The initial values are obtained from (2), (3), and (4), with $\alpha/2$.

Unequal cluster sizes

For unequal cluster sizes, we assume that the cluster sizes are independent and identically distributed and are small relative to the number of clusters; see Ahn, Heo, and Zhang (2015) for details. Let the coefficient of variation of the cluster sizes be CV_{cl}. According to van Breukelen, Candel, and Berger (2007) and Campbell and Walters (2014), to adjust for varying cluster sizes, define the relative efficiency (RE) of unequal versus equal cluster sizes as

$$\text{RE} = 1 - \lambda(1 - \lambda)\text{CV}_{\text{cl}}^2$$

where $\lambda = \rho M/(\rho M + 1 - \rho)$. With unequal cluster sizes, the effect size δ becomes

$$\delta = \frac{\mu_a - \mu_0}{\sigma\sqrt{\text{DE}/\text{RE}}} \tag{5}$$

The number of clusters for a one-sided test is computed using (2) after substituting δ from (5).

The effect size for a one-sided test is computed using (4) and the minimum detectable value of the mean is computed as
$\mu_a = \mu_0 + (z_{1-\alpha} - z_{\beta})\sigma\sqrt{\text{DE}/\text{RE}}/\sqrt{n}$ when $\mu_a > \mu_0$ and as
$\mu_a = \mu_0 - (z_{1-\alpha} - z_{\beta})\sigma\sqrt{\text{DE}/\text{RE}}/\sqrt{n}$ when $\mu_a < \mu_0$.

In all other cases, parameters are computed iteratively using the power equations in (1) with δ as defined in (5).

References

Ahn, C., M. Heo, and S. Zhang. 2015. *Sample Size Calculations for Clustered and Longitudinal Outcomes in Clinical Research*. Boca Raton, FL: CRC Press. https://doi.org/10.1201/b17822.

Campbell, M. J., and S. J. Walters. 2014. *How to Design, Analyse and Report Cluster Randomised Trials in Medicine and Health Related Research*. Chichester, UK: Wiley. https://doi.org/10.1002/9781118763452.

Gallis, J. A., F. Li, H. Yu, and E. L. Turner. 2018. cvcrand and cptest: Commands for efficient design and analysis of cluster randomized trials using constrained randomization and permutation tests. *Stata Journal* 18: 357–378.

Tamhane, A. C., and D. D. Dunlop. 2000. *Statistics and Data Analysis: From Elementary to Intermediate*. Upper Saddle River, NJ: Prentice Hall.

van Breukelen, G. J. P., M. J. J. M. Candel, and M. P. F. Berger. 2007. Relative efficiency of unequal versus equal cluster sizes in cluster randomized and multicentre trials. *Statistics in Medicine* 26: 2589–2603. https://doi.org/10.1002/sim.2740.

Also see

[PSS-2] **power onemean** — Power analysis for a one-sample mean test

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[R] **ztest** — *z* tests (mean-comparison tests, known variance)

power twomeans — Power analysis for a two-sample means test

Description

`power twomeans` computes sample size, power, or the experimental-group mean for a two-sample means test. By default, it computes sample size for the given power and the values of the control-group and experimental-group means. Alternatively, it can compute power for given sample size and values of the control-group and experimental-group means or the experimental-group mean for given sample size, power, and the control-group mean. For power and sample-size analysis in a cluster randomized design, see [PSS-2] **power twomeans, cluster**. Also see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

For precision and sample-size analysis for a CI for the difference between two means from independent samples, see [PSS-3] **ciwidth twomeans**.

Quick start

Sample size for a test of H_0: $\mu_1 = \mu_2$ versus H_a: $\mu_1 \neq \mu_2$ given alternative control-group mean $m_1 = 8$ and alternative experimental-group mean $m_2 = 12$ with shared standard deviation of 9 using default power of 0.8 and significance level $\alpha = 0.05$

```
power twomeans 8 12, sd(9)
```

Same as above, but for m_2 equal to 10, 11, 12, 13, and 14

```
power twomeans 8 (10(1)14), sd(9)
```

Same as above, but display results in a graph of sample size versus m_2

```
power twomeans 8 (10(1)14), sd(9) graph
```

Same as above, but specify different standard deviations $s_1 = 7$ and $s_2 = 10$

```
power twomeans 8 (10(1)14), sd1(7) sd2(10) graph
```

Sample size for one-sided test with power of 0.9

```
power twomeans 8 12, sd(9) power(.9) onesided
```

Same as above, specified as μ_1 and difference between means $m_2 - m_1 = 4$

```
power twomeans 8, sd(9) power(.9) onesided diff(4)
```

Power for a total sample size of 74 with balanced group sizes

```
power twomeans 8 12, sd(9) n(74)
```

Same as above, but for sample sizes of 45 and 30 in groups 1 and 2, respectively

```
power twomeans 8 12, sd(9) n1(45) n2(30)
```

Effect size and target mean difference for a sample size of 200 with power of 0.8

```
power twomeans 8, sd(9) power(.8) n(200)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
power twomeans m1 m2 [ , power(numlist) options ]
```

Compute power

```
power twomeans m1 m2 , n(numlist) [ options ]
```

Compute effect size and experimental-group mean

```
power twomeans m1 , n(numlist) power(numlist) [ options ]
```

where m_1 is the mean in the control (reference) group and m_2 is the mean in the experimental (comparison) group. m_1 and m_2 may each be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

options	Description
Main	
* `alpha(`*numlist*`)`	significance level; default is `alpha(0.05)`
* `power(`*numlist*`)`	power; default is `power(0.8)`
* `beta(`*numlist*`)`	probability of type II error; default is `beta(0.2)`
* `n(`*numlist*`)`	total sample size; required to compute power or effect size
* `n1(`*numlist*`)`	sample size of the control group
* `n2(`*numlist*`)`	sample size of the experimental group
* `nratio(`*numlist*`)`	ratio of sample sizes, `N2/N1`; default is `nratio(1)`, meaning equal group sizes
`compute(N1 | N2)`	solve for `N1` given `N2` or for `N2` given `N1`
`nfractional`	allow fractional sample sizes
* `diff(`*numlist*`)`	difference between the experimental-group mean and the control-group mean, $m_2 - m_1$; specify instead of the experimental-group mean m_2
* `sd(`*numlist*`)`	common standard deviation of the control and the experimental groups assuming equal standard deviations in both groups; default is `sd(1)`
* `sd1(`*numlist*`)`	standard deviation of the control group; requires `sd2()`
* `sd2(`*numlist*`)`	standard deviation of the experimental group; requires `sd1()`
`knownsds`	request computation assuming known standard deviations for both groups; default is to assume unknown standard deviations
`direction(upper|lower)`	direction of the effect for effect-size determination; default is `direction(upper)`, which means that the postulated value of the parameter is larger than the hypothesized value
`onesided`	one-sided test; default is two sided
`parallel`	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
`[no]table[(`*tablespec*`)]`	suppress table or display results as a table; see [PSS-2] **power, table**
`saving(`*filename* `[, replace])`	save the table data to *filename*; use `replace` to overwrite existing *filename*
Graph	
`graph[(`*graphopts*`)]`	graph results; see [PSS-2] **power, graph**

Iteration

`init(#)`	initial value for sample sizes or experimental-group mean
`iterate(#)`	maximum number of iterations; default is `iterate(500)`
`tolerance(#)`	parameter tolerance; default is `tolerance(1e-12)`
`ftolerance(#)`	function tolerance; default is `ftolerance(1e-12)`
`[no]log`	suppress or display iteration log
`[no]dots`	suppress or display iterations as dots
`cluster`	perform computations for a CRD; see [PSS-2] **power twomeans, cluster**
`notitle`	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the `parallel` option.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`cluster` and `notitle` do not appear in the dialog box.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`N`	total number of subjects	N
`N1`	number of subjects in the control group	N_1
`N2`	number of subjects in the experimental group	N_2
`nratio`	ratio of sample sizes, experimental to control	N_2/N_1
`delta`	effect size	δ
`m1`	control-group mean	μ_1
`m2`	experimental-group mean	μ_2
`diff`	difference between the experimental-group mean and the control-group mean	$\mu_2-\mu_1$
`sd`	common standard deviation	σ
`sd1`	control-group standard deviation	σ_1
`sd2`	experimental-group standard deviation	σ_2
`target`	target parameter; synonym for `m2`	
`_all`	display all supported columns	

Column `beta` is shown in the default table in place of column `power` if specified.

Columns `nratio`, `diff`, `sd`, `sd1`, and `sd2` are shown in the default table if the corresponding options are specified.

Options

Main

`alpha()`, `power()`, `beta()`, `n()`, `n1()`, `n2()`, `nratio()`, `compute()`, `nfractional`; see [PSS-2] **power**.

`diff(`*numlist*`)` specifies the difference between the experimental-group mean and the control-group mean, $m_2 - m_1$. You can specify either the experimental-group mean m_2 as a command argument or the difference between the two means in `diff()`. If you specify `diff(#)`, the experimental-group mean is computed as $m_2 = m_1 + \#$. This option is not allowed with the effect-size determination.

`sd(`*numlist*`)` specifies the common standard deviation of the control and the experimental groups assuming equal standard deviations in both groups. The default is `sd(1)`.

`sd1(`*numlist*`)` specifies the standard deviation of the control group. If you specify `sd1()`, you must also specify `sd2()`.

`sd2(`*numlist*`)` specifies the standard deviation of the experimental group. If you specify `sd2()`, you must also specify `sd1()`.

`knownsds` requests that standard deviations of each group be treated as known in the computations. By default, standard deviations are treated as unknown, and the computations are based on a two-sample t test, which uses a Student's t distribution as a sampling distribution of the test statistic. If `knownsds` is specified, the computation is based on a two-sample z test, which uses a normal distribution as the sampling distribution of the test statistic.

`direction()`, `onesided`, `parallel`; see [PSS-2] **power**.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies the initial value for the estimated parameter. For sample-size determination, the estimated parameter is either the control-group size n_1 or, if `compute(N2)` is specified, the experimental-group size n_2. For the effect-size determination, the estimated parameter is the experimental-group mean m_2. The default initial values for a two-sided test are obtained as a closed-form solution for the corresponding one-sided test with the significance level $\alpha/2$. The default initial values for the t test computations are based on the corresponding large-sample normal approximation.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following options are available with `power twomeans` but are not shown in the dialog box:

`cluster`; see [PSS-2] **power twomeans, cluster**.

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `power twomeans` command and the methodology for power and sample-size analysis for a two-sample means test. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis and [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests. Also see [PSS-2] **power twomeans, cluster** for power and sample-size analysis in a cluster randomized design.

Introduction

The analysis of means is one of the most commonly used approaches in a wide variety of statistical studies. Many applications lead to the study of two independent means, such as studies comparing the average mileage of foreign and domestic cars, the average SAT scores obtained from two different coaching classes, the average yields of a crop due to a certain fertilizer, and so on. The two populations of interest are assumed to be independent.

This entry describes power and sample-size analysis for the inference about two population means performed using hypothesis testing. Specifically, we consider the null hypothesis H_0: $\mu_2 = \mu_1$ versus the two-sided alternative hypothesis H_a: $\mu_2 \neq \mu_1$, the upper one-sided alternative H_a: $\mu_2 > \mu_1$, or the lower one-sided alternative H_a: $\mu_2 < \mu_1$.

The considered two-sample tests rely on the assumption that the two random samples are normally distributed or that the sample size is large. Suppose that the two samples are normally distributed. If variances of the considered populations are known a priori, the test statistic has a standard normal distribution under the null hypothesis, and the corresponding test is referred to as a two-sample z test. If variances of the two populations are not known, then the null sampling distribution of the test statistic depends on whether the two variances are assumed to be equal. If the two variances are assumed to be equal, the test statistic has an exact Student's t distribution under the null hypothesis. The corresponding test is referred to as a two-sample t test. If the two variances are not equal, then the distribution can only be approximated by a Student's t distribution; the degrees of freedom is approximated using Satterthwaite's method. We refer to this test as Satterthwaite's t test. For a large sample, the distribution of the test statistic is approximately normal, and the corresponding test is a large-sample z test.

The `power twomeans` command provides power and sample-size analysis for the above tests.

Using power twomeans

`power twomeans` computes sample size, power, or experimental-group mean for a two-sample means test. All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option. You can specify the `onesided` option to request a one-sided test. By default, all computations assume a balanced- or equal-allocation design; see [PSS-4] **Unbalanced designs** for a description of how to specify an unbalanced design.

By default, all computations are for a two-sample t test, which assumes equal and unknown standard deviations. By default, the common standard deviation is set to one but may be changed by specifying the `sd()` option. To specify different standard deviations, use the respective `sd1()` and `sd2()` options. These options must be specified together and may not be used in combination with `sd()`. When `sd1()` and `sd2()` are specified, the computations are based on Satterthwaite's t test, which assumes unequal and unknown standard deviations. If standard deviations are known, use the `knownsds` option to request that computations be based on a two-sample z test.

To compute the total sample size, you must specify the control-group mean m_1, the experimental-group mean m_2, and, optionally, the power of the test in the `power()` option. The default power is set to 0.8.

Instead of the total sample size, you can compute one of the group sizes given the other one. To compute the control-group sample size, you must specify the `compute(N1)` option and the sample size of the experimental group in the `n2()` option. Likewise, to compute the experimental-group sample size, you must specify the `compute(N2)` option and the sample size of the control group in the `n1()` option.

To compute power, you must specify the total sample size in the `n()` option, the control-group mean m_1, and the experimental-group mean m_2.

Instead of the experimental-group mean m_2, you may specify the difference $m_2 - m_1$ between the experimental-group mean and the control-group mean in the `diff()` option when computing sample size or power.

To compute effect size, the difference between the experimental-group mean and the null mean, and the experimental-group mean, you must specify the total sample size in the `n()` option, the power in the `power()` option, the control-group mean m_1, and, optionally, the direction of the effect. The direction is upper by default, `direction(upper)`, which means that the experimental-group mean is assumed to be larger than the specified control-group value. You can change the direction to be lower, which means that the experimental-group mean is assumed to be smaller than the specified control-group value, by specifying the `direction(lower)` option.

Instead of the total sample size `n()`, you can specify individual group sizes in `n1()` and `n2()`, or specify one of the group sizes and `nratio()` when computing power or effect size. Also see *Two samples* in [PSS-4] **Unbalanced designs** for more details.

In the following sections, we describe the use of `power twomeans` accompanied by examples for computing sample size, power, and experimental-group mean.

Computing sample size

To compute sample size, you must specify the control-group mean m_1, the experimental-group mean m_2, and, optionally, the power of the test in the `power()` option. A default power of 0.8 is assumed if `power()` is not specified.

▷ Example 1: Sample size for a two-sample means test

Consider a study investigating the effects of smoking on lung function of males. The response variable is forced expiratory volume (FEV), measured in liters (L), where better lung function implies higher values of FEV. We wish to test the null hypothesis H_0: $\mu_1 = \mu_2$ versus a two-sided alternative hypothesis H_a: $\mu_1 \neq \mu_2$, where μ_1 and μ_2 are the mean FEV for nonsmokers and smokers, respectively.

Suppose that the mean FEV from previous studies was reported to be 3 L for nonsmokers and 2.7 L for smokers. We are designing a new study and wish to find out how many subjects we need to enroll so that the power of a 5%-level two-sided test to detect the specified difference between means is at least 80%. We assume equal numbers of subjects in each group and a common standard deviation of 1.

```
. power twomeans 3 2.7
Performing iteration ...
Estimated sample sizes for a two-sample means test
t test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =   -0.3000
           m1 =    3.0000
           m2 =    2.7000
           sd =    1.0000
Estimated sample sizes:
            N =       352
  N per group =       176
```

We need a total sample of 352 subjects, 176 per group, to detect the specified mean difference between the smoking and nonsmoking groups with 80% power using a two-sided 5%-level test.

The default computation is for the case of equal and unknown standard deviations, as indicated by the output. You can specify the `knownsds` option to request the computation assuming known standard deviations.

◁

▷ Example 2: Sample size assuming unequal standard deviations

Instead of assuming equal standard deviations as in example 1, we use the estimates of the standard deviations from previous studies as our hypothetical values. The standard deviation of FEV for the nonsmoking group was reported to be 0.8 L and that for the smoking group was reported to be 0.7 L. We specify standard deviations in the `sd1()` and `sd2()` options.

```
. power twomeans 3 2.7, sd1(0.8) sd2(0.7)
Performing iteration ...
Estimated sample sizes for a two-sample means test
Satterthwaite's t test assuming unequal variances
H0: m2 = m1  versus  Ha: m2 != m1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =   -0.3000
           m1 =    3.0000
           m2 =    2.7000
          sd1 =    0.8000
          sd2 =    0.7000
Estimated sample sizes:
            N =       200
  N per group =       100
```

The specified standard deviations are smaller than one, so we obtain a smaller required total sample size of 200 compared with example 1.

◁

Example 3: Specifying difference between means

Instead of the mean FEV of 2.7 for the smoking group as in example 2, we can specify the difference between the two means of $2.7 - 3 = -0.3$ in the `diff()` option.

```
. power twomeans 3, sd1(0.8) sd2(0.7) diff(-0.3)

Performing iteration ...

Estimated sample sizes for a two-sample means test
Satterthwaite's t test assuming unequal variances
H0: m2 = m1  versus  Ha: m2 != m1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =   -0.3000
           m1 =    3.0000
           m2 =    2.7000
         diff =   -0.3000
          sd1 =    0.8000
          sd2 =    0.7000

Estimated sample sizes:

            N =       200
  N per group =       100
```

We obtain the same results as in example 2. The difference between means is now also reported in the output following the individual means.

Example 4: Computing one of the group sizes

Suppose we anticipate a sample of 120 nonsmoking subjects. We wish to compute the required number of subjects in the smoking group, keeping all other study parameters as in example 2. We specify the number of subjects in the nonsmoking group in the `n1()` option and specify the `compute(N2)` option.

```
. power twomeans 3 2.7, sd1(0.8) sd2(0.7) n1(120) compute(N2)

Performing iteration ...

Estimated sample sizes for a two-sample means test
Satterthwaite's t test assuming unequal variances
H0: m2 = m1  versus  Ha: m2 != m1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =   -0.3000
           m1 =    3.0000
           m2 =    2.7000
          sd1 =    0.8000
          sd2 =    0.7000
           N1 =       120

Estimated sample sizes:

            N =       202
           N2 =        82
```

We need a sample of 82 smoking subjects given a sample of 120 nonsmoking subjects.

▷ Example 5: Unbalanced design

By default, `power twomeans` computes sample size for a balanced- or equal-allocation design. If we know the allocation ratio of subjects between the groups, we can compute the required sample size for an unbalanced design by specifying the `nratio()` option.

Continuing with example 2, we will suppose that we anticipate to recruit twice as many smokers than nonsmokers; that is, $n_2/n_1 = 2$. We specify the `nratio(2)` option to compute the required sample size for the specified unbalanced design.

```
. power twomeans 3 2.7, sd1(0.8) sd2(0.7) nratio(2)

Performing iteration ...

Estimated sample sizes for a two-sample means test
Satterthwaite's t test assuming unequal variances
H0: m2 = m1  versus  Ha: m2 != m1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =   -0.3000
           m1 =    3.0000
           m2 =    2.7000
          sd1 =    0.8000
          sd2 =    0.7000
        N2/N1 =    2.0000

Estimated sample sizes:

            N =       237
           N1 =        79
           N2 =       158
```

We need a total sample size of 237 subjects, which is larger than the required total sample size for the corresponding balanced design from example 2.

Also see *Two samples* in [PSS-4] **Unbalanced designs** for more examples of unbalanced designs for two-sample tests.

◁

Computing power

To compute power, you must specify the total sample size in the `n()` option, the control-group mean m_1, and the experimental-group mean m_2.

▷ Example 6: Power of a two-sample means test

Continuing with example 1, we will suppose that we have resources to enroll a total of only 250 subjects, assuming equal-sized groups. To compute the power corresponding to this sample size given the study parameters from example 1, we specify the total sample size in `n()`:

```
. power twomeans 3 2.7, n(250)
Estimated power for a two-sample means test
t test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1
Study parameters:
        alpha =    0.0500
            N =       250
  N per group =       125
        delta =   -0.3000
           m1 =    3.0000
           m2 =    2.7000
           sd =    1.0000
Estimated power:
        power =    0.6564
```

With a total sample of 250 subjects, we obtain a power of only 65.64%.

◁

▷ Example 7: Multiple values of study parameters

In this example, we assess the effect of varying the common standard deviation (assuming equal standard deviations in both groups) of FEV on the power of our study.

Continuing with example 6, we compute powers for a range of common standard deviations between 0.5 and 1.5 with the step size of 0.1. We specify the corresponding numlist in the `sd()` option.

```
. power twomeans 3 2.7, sd(0.5(0.1)1.5) n(250)
Estimated power for a two-sample means test
t test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1
```

alpha	power	N	N1	N2	delta	m1	m2	sd
.05	.9972	250	125	125	-.3	3	2.7	.5
.05	.976	250	125	125	-.3	3	2.7	.6
.05	.9215	250	125	125	-.3	3	2.7	.7
.05	.8397	250	125	125	-.3	3	2.7	.8
.05	.747	250	125	125	-.3	3	2.7	.9
.05	.6564	250	125	125	-.3	3	2.7	1
.05	.5745	250	125	125	-.3	3	2.7	1.1
.05	.5036	250	125	125	-.3	3	2.7	1.2
.05	.4434	250	125	125	-.3	3	2.7	1.3
.05	.3928	250	125	125	-.3	3	2.7	1.4
.05	.3503	250	125	125	-.3	3	2.7	1.5

The power decreases from 99.7% to 35.0% as the common standard deviation increases from 0.5 to 1.5 L.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

Computing effect size and experimental-group mean

Effect size δ for a two-sample means test is defined as the difference between the experimental-group mean and the control-group mean $\delta = \mu_2 - \mu_1$.

Sometimes, we may be interested in determining the smallest effect and the corresponding experimental-group mean that yield a statistically significant result for prespecified sample size and power. In this case, power, sample size, and control-group mean must be specified. In addition, you must also decide on the direction of the effect: upper, meaning $m_2 > m_1$, or lower, meaning $m_2 < m_1$. The direction may be specified in the `direction()` option; `direction(upper)` is the default.

▷ Example 8: Minimum detectable change in the experimental-group mean

Continuing with example 6, we compute the smallest change in the mean of the smoking group that can be detected given a total sample of 250 subjects and 80% power, assuming equal-group allocation. To solve for the mean FEV of the smoking group, after the command name, we specify the nonsmoking-group mean of 3, total sample size `n(250)`, and power `power(0.8)`.

Because our initial study was based on the hypothesis that FEV for the smoking group is lower than that of the nonsmoking group, we specify the `direction(lower)` option to compute the smoking-group mean that is lower than the specified nonsmoking-group mean.

```
. power twomeans 3, n(250) power(0.8) direction(lower)

Performing iteration ...

Estimated experimental-group mean for a two-sample means test
t test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1; m2 < m1

Study parameters:

        alpha =    0.0500
        power =    0.8000
            N =       250
  N per group =       125
           m1 =    3.0000
           sd =    1.0000

Estimated effect size and experimental-group mean:

        delta =   -0.3558
           m2 =    2.6442
```

We find that the minimum detectable value of the effect size is -0.36, which corresponds to the mean FEV of 2.64 for the smoking group.

◁

Testing a hypothesis about two independent means

After data are collected, we can use the `ttest` command to test the equality of two independent means using a t test; see [R] **ttest** for details. In this section, we demonstrate the use of `ttesti`, the immediate form of the `test` command, which can be used to test a hypothesis using summary statistics instead of the actual data values.

▷ Example 9: Two-sample t test

Consider an example from van Belle et al. (2004, 129), where newborn infants were divided into two groups: a treatment group, where infants received daily "walking stimulus" for eight weeks, and a control group, where no stimulus was provided. The goal of this study was to test whether receiving the walking stimulus during stages of infancy induces the walking ability to develop sooner.

The average number of months before the infants started walking was recorded for both groups. The authors provide estimates of the average of 10.125 months for the treatment group with estimated standard deviation of 1.447 months and 12.35 months for the control group with estimated standard deviation of 0.9618 months. The sample sizes for treatment and control groups were 6 and 5, respectively. We supply these estimates to the `ttesti` command and use the `unequal` option to perform a t test assuming unequal variances.

```
. ttesti 6 10.125 1.447 5 12.35 0.9618, unequal
Two-sample t test with unequal variances
```

	Obs	Mean	Std. err.	Std. dev.	[95% conf.	interval]
x	6	10.125	.5907353	1.447	8.606467	11.64353
y	5	12.35	.43013	.9618	11.15577	13.54423
Combined	11	11.13636	.501552	1.66346	10.01884	12.25389
diff		-2.225	.7307394		-3.887894	-.562106

```
    diff = mean(x) - mean(y)                                      t =  -3.0449
H0: diff = 0                     Satterthwaite's degrees of freedom =  8.66326
    Ha: diff < 0                 Ha: diff != 0                 Ha: diff > 0
 Pr(T < t) = 0.0073         Pr(|T| > |t|) = 0.0145          Pr(T > t) = 0.9927
```

We reject the null hypothesis of H_0: $\mu_C = \mu_T$ against the two-sided alternative H_a: $\mu_C \neq \mu_T$ at the 5% significance level; the p-value $= 0.0145$.

We use the estimates of this study to perform a sample-size analysis we would have conducted before a new study. In our analysis, we assume equal-group allocation.

```
. power twomeans 10.125 12.35, power(0.8) sd1(1.447) sd2(0.9618)
Performing iteration ...
Estimated sample sizes for a two-sample means test
Satterthwaite's t test assuming unequal variances
H0: m2 = m1  versus  Ha: m2 != m1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    2.2250
           m1 =   10.1250
           m2 =   12.3500
          sd1 =    1.4470
          sd2 =    0.9618
Estimated sample sizes:
            N =        14
  N per group =         7
```

We find that the sample size required to detect a difference of 2.225 ($12.35 - 10.125 = 2.225$) given the control-group standard deviation of 1.447 and the experimental-group standard deviation of 0.9618 using a 5%-level two-sided test is 7 in each group.

◁

Stored results

power twomeans stores the following in r():

Scalars

r(alpha)	significance level
r(power)	power
r(beta)	probability of a type II error
r(delta)	effect size
r(N)	total sample size
r(N_a)	actual sample size
r(N1)	sample size of the control group
r(N2)	sample size of the experimental group
r(nratio)	ratio of sample sizes, N2/N1
r(nratio_a)	actual ratio of sample sizes
r(nfractional)	1 if nfractional is specified, 0 otherwise
r(onesided)	1 for a one-sided test, 0 otherwise
r(m1)	control-group mean
r(m2)	experimental-group mean
r(diff)	difference between the experimental- and control-group means
r(sd)	common standard deviation of the control and experimental groups
r(sd1)	standard deviation of the control group
r(sd2)	standard deviation of the experimental group
r(knownsds)	1 if option knownsds is specified, 0 otherwise
r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise
r(init)	initial value for sample sizes or experimental-group mean
r(maxiter)	maximum number of iterations
r(iter)	number of iterations performed
r(tolerance)	requested parameter tolerance
r(deltax)	final parameter tolerance achieved
r(ftolerance)	requested distance of the objective function from zero
r(function)	final distance of the objective function from zero
r(converged)	1 if iteration algorithm converged, 0 otherwise

Macros

r(type)	test
r(method)	twomeans
r(direction)	upper or lower
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results

Methods and formulas

Consider two independent samples with n_1 subjects in the control group and n_2 subjects in the experimental group. Let $x_{11},\ldots,x_{1n_1}$ be a random sample of size n_1 from a normal population with mean μ_1 and variance σ_1^2. Let $x_{21},\ldots,x_{2n_2}$ be a random sample of size n_2 from a normal population with mean μ_2 and variance σ_2^2. Let effect size δ be the difference between the experimental-group mean and the control-group mean, $\delta=\mu_2-\mu_1$. The sample means and variances for the two independent samples are

$$\overline{x}_1 = \frac{1}{n_1}\sum_{i=1}^{n_1} x_{1i} \qquad \text{and} \qquad s_1^2 = \frac{1}{n_1-1}\sum_{i=1}^{n_1}(x_{1i}-\overline{x}_1)^2$$

$$\overline{x}_2 = \frac{1}{n_2}\sum_{i=1}^{n_2} x_{2i} \qquad \text{and} \qquad s_2^2 = \frac{1}{n_2-1}\sum_{i=1}^{n_2}(x_{2i}-\overline{x}_2)^2$$

where $\overline{x}_j$ and s_j^2 are the respective sample means and sample variances of the two samples.

A two-sample means test involves testing the null hypothesis $H_0\colon \mu_2 = \mu_1$ versus the two-sided alternative hypothesis $H_a\colon \mu_2 \neq \mu_1$, the upper one-sided alternative $H_a\colon \mu_2 > \mu_1$, or the lower one-sided alternative $H_a\colon \mu_2 < \mu_1$.

The two-sample means test can be performed under four different assumptions: 1) population variances are known and not equal; 2) population variances are known and equal; 3) population variances are unknown and not equal; and 4) population variances are unknown and equal.

Let σ_D denote the standard deviation of the difference between the two sample means. The test statistic of the form

$$\text{TS} = \frac{(\overline{x}_2 - \overline{x}_1) - (\mu_2 - \mu_1)}{\sigma_D} \tag{1}$$

is used in each of the four cases described above. Each case, however, determines the functional form of σ_D and the sampling distribution of the test statistic (1) under the null hypothesis.

Let $R = n_2/n_1$ denote the allocation ratio. Then $n_2 = R \times n_1$ and power can be viewed as a function of n_1. Therefore, for sample-size determination, the control-group sample size n_1 is computed first. The experimental-group size n_2 is then computed as $R \times n_1$, and the total sample size is computed as $n = n_1 + n_2$. By default, sample sizes are rounded to integer values; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for details.

The following formulas are based on Armitage, Berry, and Matthews (2002); Chow et al. (2018); and Dixon and Massey (1983).

Methods and formulas are presented under the following headings:

Known standard deviations

Below we present formulas for the computations that assume unequal standard deviations. When standard deviations are equal, the corresponding formulas are special cases of the formulas below with $\sigma_1 = \sigma_2 = \sigma$.

When the standard deviations of the control and the experimental groups are known, the test statistic in (1) is a z test statistic

$$z = \frac{(\overline{x}_2 - \overline{x}_1) - (\mu_2 - \mu_1)}{\sqrt{\sigma_1^2/n_1 + \sigma_2^2/n_2}}$$

with $\sigma_D = \sqrt{\sigma_1^2/n_1 + \sigma_2^2/n_2}$. The sampling distribution of this test statistic under the null hypothesis is standard normal. The corresponding test is referred to as a z test.

Let α be the significance level, β be the probability of a type II error, and $z_{1-\alpha}$ and z_β be the $(1-\alpha)$th and the βth quantiles of a standard normal distribution.

The power $\pi = 1 - \beta$ is computed using

$$\pi = \begin{cases} \Phi\left(\frac{\delta}{\sigma_D} - z_{1-\alpha}\right) & \text{for an upper one-sided test} \\ \Phi\left(-\frac{\delta}{\sigma_D} - z_{1-\alpha}\right) & \text{for a lower one-sided test} \\ \Phi\left(\frac{\delta}{\sigma_D} - z_{1-\alpha/2}\right) + \Phi\left(-\frac{\delta}{\sigma_D} - z_{1-\alpha/2}\right) & \text{for a two-sided test} \end{cases} \tag{2}$$

where $\Phi(\cdot)$ is the cdf of a standard normal distribution.

For a one-sided test, the control-group sample size n_1 is computed as follows:

$$n_1 = \left(\frac{z_{1-\alpha} - z_\beta}{\mu_2 - \mu_1}\right)^2 \left(\sigma_1^2 + \frac{\sigma_2^2}{R}\right) \tag{3}$$

For a one-sided test, if one of the group sizes is known, the other one is computed using the following formula. For example, to compute n_1 given n_2, we use the following formula:

$$n_1 = \frac{\sigma_1^2}{\left(\frac{\mu_2 - \mu_1}{z_{1-\alpha} - z_\beta}\right)^2 - \frac{\sigma_2^2}{n_2}} \tag{4}$$

For a two-sided test, sample sizes are computed by iteratively solving the two-sided power equation in (2). The default initial values for the iterative procedure are calculated from the respective equations (3) and (4), with α replaced with $\alpha/2$.

The absolute value of the effect size for a one-sided test is obtained by inverting the corresponding one-sided power equation in (2):

$$|\delta| = \sigma_D(z_{1-\alpha} - z_\beta)$$

Note that the magnitude of the effect size is the same regardless of the direction of the test.

The experimental-group mean for a one-sided test is then computed as

$$\mu_2 = \begin{cases} \mu_1 + (z_{1-\alpha} - z_\beta)\sqrt{\sigma_1^2/n_1 + \sigma_2^2/n_2} & \text{when } \mu_2 > \mu_1 \\ \mu_1 - (z_{1-\alpha} - z_\beta)\sqrt{\sigma_1^2/n_1 + \sigma_2^2/n_2} & \text{when } \mu_2 < \mu_1 \end{cases}$$

For a two-sided test, the experimental-group mean is computed by iteratively solving the two-sided power equation in (2) for μ_2. The default initial value is obtained from the corresponding one-sided computation with $\alpha/2$.

Unknown standard deviations

When the standard deviations of the control group and the experimental group are unknown, the test statistic in (1) is a t test statistic

$$t = \frac{(\overline{x}_2 - \overline{x}_1) - (\mu_2 - \mu_1)}{s_D}$$

where s_D is the estimated standard deviation of the sample mean difference. The sampling distribution of this test statistic under the null hypothesis is (approximately) a Student's t distribution with ν degrees of freedom. Parameters ν and s_D are defined below, separately for the case of equal and unequal standard deviations.

Let $t_{\nu,\alpha}$ denote the αth quantile of a Student's t distribution with ν degrees of freedom. Under the alternative hypothesis, the test statistic follows a noncentral Student's t distribution with ν degrees of freedom and noncentrality parameter λ.

The power is computed from the equations,

$$\pi = \begin{cases} 1 - T_{\nu,\lambda}\left(t_{\nu,1-\alpha}\right) & \text{for an upper one-sided test} \\ T_{\nu,\lambda}\left(-t_{\nu,1-\alpha}\right) & \text{for a lower one-sided test} \\ 1 - T_{\nu,\lambda}\left(t_{\nu,1-\alpha/2}\right) + T_{\nu,\lambda}\left(-t_{\nu,1-\alpha/2}\right) & \text{for a two-sided test} \end{cases} \tag{5}$$

where $\lambda = |\mu_2 - \mu_1|/s_D$.

Sample sizes and the experimental-group mean are obtained by iteratively solving the nonlinear equation (5) for n_1, n_2, and μ_2, respectively. For sample-size and effect-size computations, the default initial values for the iterative procedure are calculated using the corresponding formulas assuming known standard deviations from the previous subsection.

Unequal standard deviations

In the case of unequal standard deviations,

$$s_D = \sqrt{s_1^2/n_1 + s_2^2/n_2}$$

and the degrees of freedom ν of the test statistic is obtained by Satterthwaite's formula:

$$\nu = \frac{\left(\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}\right)^2}{\frac{(s_1^2/n_1)^2}{n_1 - 1} + \frac{(s_2^2/n_2)^2}{n_2 - 1}}$$

The sampling distribution of the test statistic under the null hypothesis is an approximate Student's t distribution. We refer to the corresponding test as Satterthwaite's t test.

Equal standard deviations

In the case of equal standard deviations,

$$s_D = s_p\sqrt{1/n_1 + 1/n_2}$$

where $s_p = \left\{\sum_{i=1}^{n_1}(x_{1i} - \overline{x}_1)^2 + \sum_{i=1}^{n_2}(x_{2i} - \overline{x}_2)^2\right\}/(n_1 + n_2 - 2)$ is the pooled-sample standard deviation.

The degrees of freedom ν is

$$\nu = n_1 + n_2 - 2$$

The sampling distribution of the test statistic under the null hypothesis is exactly a Student's t distribution. We refer to the corresponding test as a two-sample t test.

References

Armitage, P., G. Berry, and J. N. S. Matthews. 2002. *Statistical Methods in Medical Research*. 4th ed. Oxford: Blackwell.

Chow, S.-C., J. Shao, H. Wang, and Y. Lokhnygina. 2018. *Sample Size Calculations in Clinical Research*. 3rd ed. Boca Raton, FL: CRC Press.

Dixon, W. J., and F. J. Massey, Jr. 1983. *Introduction to Statistical Analysis*. 4th ed. New York: McGraw–Hill.

Juul, S., and M. Frydenberg. 2021. *An Introduction to Stata for Health Researchers*. 5th ed. College Station, TX: Stata Press.

van Belle, G., L. D. Fisher, P. J. Heagerty, and T. S. Lumley. 2004. *Biostatistics: A Methodology for the Health Sciences*. 2nd ed. New York: Wiley.

Also see

[PSS-2] **power twomeans, cluster** — Power analysis for a two-sample means test, CRD

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power oneway** — Power analysis for one-way analysis of variance

[PSS-2] **power twoway** — Power analysis for two-way analysis of variance

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-3] **ciwidth twomeans** — Precision analysis for a two-means-difference CI

[PSS-5] **Glossary**

[ADAPT] **gsdesign twomeans** — Group sequential design for a two-sample means test

[R] **ttest** — t tests (mean-comparison tests)

power twomeans, cluster — Power analysis for a two-sample means test, CRD

Description

`power twomeans, cluster` computes group-specific numbers of clusters, group-specific cluster sizes, power, or the experimental-group mean for a two-sample means test in a cluster randomized design (CRD). It computes group-specific numbers of clusters given cluster sizes, power, and the values of the control-group and experimental-group means. It also computes group-specific cluster sizes given numbers of clusters, power, and the values of the control-group and experimental-group means. Alternatively, it computes power given numbers of clusters, cluster sizes, and the values of the control-group and experimental-group means, or it computes the experimental-group mean given numbers of clusters, cluster sizes, power, and the control-group mean. See [PSS-2] **power twomeans** for a general discussion of power and sample-size analysis for a two-sample means test. Also see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Quick start

Numbers of clusters for a test of $H_0\colon \mu_1 = \mu_2$ versus $H_a\colon \mu_1 \neq \mu_2$ given alternative control- and experimental-group means $m_1 = 8$ and $m_2 = 12$ with common standard deviation of 9 and cluster sizes of 5 using default intraclass correlation of 0.5, power of 0.8, and significance level $\alpha = 0.05$

```
power twomeans 8 12, m1(5) m2(5) sd(9)
```

Same as above, but assume the intraclass correlation is 0.4

```
power twomeans 8 12, m1(5) m2(5) sd(9) rho(0.4)
```

Same as above, and assume that cluster sizes vary with an average of 5 and coefficient of variation of 0.8

```
power twomeans 8 12, m1(5) m2(5) sd(9) rho(0.4) cvcluster(0.8)
```

Group-specific numbers of clusters using a ratio of experimental to control clusters of 0.5

```
power twomeans 8 12, m1(5) m2(5) kratio(0.5) sd(9)
```

Cluster sizes for a test of $H_0\colon \mu_1 = \mu_2$ versus $H_a\colon \mu_1 \neq \mu_2$ given alternative control-group and experimental-group means $m_1 = 1$ and $m_2 = 1.5$ for 60 equal-sized clusters in the control group and 30 clusters in the experimental group using default intraclass correlation of 0.5, standard deviation of 1, power of 0.8, and significance level $\alpha = 0.05$

```
power twomeans 1 1.5, k1(60) k2(30)
```

Same as above, but compute experimental-group cluster size given the control-group cluster size of 5

```
power twomeans 1 1.5, k1(60) k2(30) m1(5) compute(M2)
```

Power for 20 clusters with cluster sizes of 5 in the control and experimental groups

```
power twomeans 1 1.5, k1(20) k2(20) m1(5) m2(5)
```

Power for multiple numbers of clusters in the experimental group

```
power twomeans 1 1.5, k1(20) k2(20(5)50) m1(5) m2(5)
```

Same as above, and display results in a graph of power versus the number of clusters in the experimental group

```
power twomeans 1 1.5, k1(20) k2(20(5)50) m1(5) m2(5) graph
```

Effect size and target experimental-group mean with power of 0.8

```
power twomeans 1, k1(20) k2(20) m1(5) m2(5) power(0.8)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute numbers of clusters

```
power twomeans m1 m2 , { mspec | nspec cluster } [ options ]
```

Compute cluster sizes

```
power twomeans m1 m2 , kspec [ options ]
```

Compute power

```
power twomeans m1 m2 , kspec { mspec | nspec } [ options ]
```

Compute effect size and experimental-group mean

```
power twomeans m1 , kspec { mspec | nspec } power(numlist) [ options ]
```

where m_1 is the mean in the control (reference) group and m_2 is the mean in the experimental (comparison) group. m_1 and m_2 may each be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

kspec is one of

```
k1()  k2()
k1()  [kratio()]
k2()  [kratio()]
```

mspec is one of

```
m1()  m2()
m1()  [mratio()]
m2()  [mratio()]
```

nspec is one of

```
n1()  n2()
n1()  [nratio()]
n2()  [nratio()]
```

options	Description
Main	
`cluster`	perform computations for a CRD; implied by `k1()`, `k2()`, `m1()`, or `m2()`
* `alpha(`*numlist*`)`	significance level; default is `alpha(0.05)`
* `power(`*numlist*`)`	power; default is `power(0.8)`
* `beta(`*numlist*`)`	probability of type II error; default is `beta(0.2)`
* `k1(`*numlist*`)`	number of clusters in the control group
* `k2(`*numlist*`)`	number of clusters in the experimental group
* `kratio(`*numlist*`)`	cluster ratio, K2/K1; default is `kratio(1)`
* `m1(`*numlist*`)`	cluster size of the control group
* `m2(`*numlist*`)`	cluster size of the experimental group
* `mratio(`*numlist*`)`	cluster-size ratio, M2/M1; default is `mratio(1)`
* `n1(`*numlist*`)`	sample size of the control group
* `n2(`*numlist*`)`	sample size of the experimental group
* `nratio(`*numlist*`)`	sample-size ratio, N2/N1; default is `nratio(1)`
`compute(K1 | K2 | M1 | M2)`	solve for the number of clusters or cluster size in one group given the other group
`nfractional`	allow fractional numbers of clusters, cluster sizes, and sample sizes
* `diff(`*numlist*`)`	difference between the experimental-group mean and the control-group mean, $m_2 - m_1$; specify instead of the experimental-group mean m_2
* `sd(`*numlist*`)`	common standard deviation of the control and the experimental groups assuming equal standard deviations in both groups; default is `sd(1)`
* `sd1(`*numlist*`)`	standard deviation of the control group; requires `sd2()`
* `sd2(`*numlist*`)`	standard deviation of the experimental group; requires `sd1()`
* `rho(`*numlist*`)`	intraclass correlation; default is `rho(0.5)`
* `cvcluster(`*numlist*`)`	coefficient of variation for cluster sizes
`direction(upper|lower)`	direction of the effect for effect-size determination; default is `direction(upper)`, which means that the postulated value of the parameter is larger than the hypothesized value
`onesided`	one-sided test; default is two sided
`parallel`	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[`no`]`table`[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
`saving(`*filename* [`, replace`]`)`	save the table data to *filename*; use `replace` to overwrite existing *filename*
Graph	
`graph`[(*graphopts*)]	graph results; see [PSS-2] **power, graph**

Iteration

`init(#)`	initial value for numbers of clusters, cluster sizes, or experimental-group mean
`iterate(#)`	maximum number of iterations; default is `iterate(500)`
`tolerance(#)`	parameter tolerance; default is `tolerance(1e-12)`
`ftolerance(#)`	function tolerance; default is `ftolerance(1e-12)`
[`no`]`log`	suppress or display iteration log
[`no`]`dots`	suppress or display iterations as dots
`notitle`	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the `parallel` option.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`notitle` does not appear in the dialog box.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
alpha	significance level	α
power	power	$1-\beta$
beta	type-II-error probability	β
K1	number of clusters in the control group	K_1
K2	number of clusters in the experimental group	K_2
kratio	ratio of numbers of clusters, experimental to control	K_2/K_1
M1	cluster size of the control group	M_1
M2	cluster size of the experimental group	M_2
mratio	ratio of cluster sizes, experimental to control	M_2/M_1
N	total number of observations	N
N1	number of observations in the control group	N_1
N2	number of observations in the experimental group	N_2
nratio	ratio of sample sizes, experimental to control	N_2/N_1
delta	effect size	δ
m1	control-group mean	μ_1
m2	experimental-group mean	μ_2
diff	difference between the experimental-group mean and the control-group mean	$\mu_2-\mu_1$
sd	common standard deviation	σ
sd1	control-group standard deviation	σ_1
sd2	experimental-group standard deviation	σ_2
rho	intraclass correlation	ρ
CV_cluster	coefficient of variation for cluster sizes	$\mathrm{CV}_{\mathrm{cl}}$
target	target parameter; synonym for m2	
_all	display all supported columns	

Column beta is shown in the default table in place of column power if specified.

Column N is shown in the table if specified.

Columns N1 and N2 are shown in the default table if n1() or n2() is specified.

Columns nratio, diff, and CV_cluster are shown in the default table if specified.

Options

Main

cluster specifies that computations should be performed for a CRD. This option is implied when the k1(), k2(), m1(), or m2() option is specified. cluster is required to compute the numbers of clusters when *nspec* is used to specify sample sizes instead of *mspec* for cluster sizes.

alpha(), power(), beta(); see [PSS-2] **power**.

k1(*numlist*) specifies the number of clusters in the control group.

k2(*numlist*) specifies the number of clusters in the experimental group.

kratio(*numlist*) specifies the ratio of the numbers of clusters of the experimental group relative to the control group, K2/K1. The default is kratio(1), meaning equal numbers of clusters in the two groups.

m1(*numlist*) specifies the cluster size of the control group. m1() may contain noninteger values.

m2(*numlist*) specifies the cluster size of the experimental group. m2() may contain noninteger values.

`mratio(`*numlist*`)` specifies the ratio of cluster sizes of the experimental group relative to the control group, `M2/M1`. The default is `mratio(1)`, meaning equal cluster sizes in the two groups.

`n1()`, `n2()`, `nratio()`; see [PSS-2] **power**.

`compute(K1 | K2 | M1 | M2)` solve for the number of clusters or cluster size of one group given the other group.

`nfractional`; see [PSS-2] **power**. The `nfractional` option displays fractional (without rounding) values of the numbers of clusters, cluster sizes, and sample sizes.

`diff()`, `sd()`, `sd1()`, `sd2()`; see [PSS-2] **power twomeans**.

`rho(`*numlist*`)` specifies the intraclass correlation. The default is `rho(0.5)`.

`cvcluster(`*numlist*`)` specifies the coefficient of variation for cluster sizes. This option is used with varying cluster sizes.

`direction()`, `onesided`, `parallel`; see [PSS-2] **power**.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies the initial value for the numbers of clusters or cluster sizes for sample-size determination or the initial value for the experimental-group mean for the effect-size determination. The default is to use a closed-form normal approximation to compute an initial value for the estimated parameter.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following option is available with `power twomeans, cluster` but is not shown in the dialog box:

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

`power twomeans, cluster` requests that computations for the `power twomeans` command be done for a CRD. In a CRD, groups of subjects or clusters are randomized instead of individual subjects, so the sample size is determined by the numbers of clusters and the cluster sizes. The sample-size determination thus consists of the determination of the numbers of clusters given cluster sizes or the determination of cluster sizes given the numbers of clusters. For a general discussion of using `power twomeans`, see [PSS-2] **power twomeans**. The discussion below is specific to the CRD.

Using power twomeans, cluster

If you specify the `cluster` option, include `k1()` or `k2()` to specify the number of clusters or include `m1()` or `m2()` to specify the cluster size, the `power twomeans` command will perform computations for a two-sample means test in a CRD.

All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option. You can specify the `onesided` option to request a one-sided test. By default, all computations assume a balanced or equal-allocation design, meaning equal numbers of clusters and cluster sizes in both groups; see [PSS-4] **Unbalanced designs** for a description of how to specify an unbalanced design.

To compute the number of clusters in both groups, you must provide cluster sizes for both groups. There are multiple ways to supply cluster sizes, but the most common is to specify the cluster size of the control group in the `m1()` option and the cluster size of the experimental group in the `m2()` option. See *mspec* and *nspec* under *Syntax* for other specifications. When *nspec* is specified, the `cluster` option is also required to request that `power twomeans` perform computations for a CRD. The number of clusters is assumed to be equal in the two groups, but you can change this by specifying the ratio of the numbers of clusters in the experimental to the control group in the `kratio()` option. Other parameters are specified as described in *Using power twomeans* in [PSS-2] **power twomeans**.

To compute the cluster sizes in both groups, you must provide the numbers of clusters in both groups. There are several ways to supply the numbers of clusters; see *kspec* under *Syntax*. The most common is to specify the numbers of clusters in the control group and the experimental group in the `k1()` and `k2()` options, respectively. Equal cluster sizes are assumed in the two groups, but you can change this by specifying the ratio of the cluster sizes in the experimental to that of the control group in the `mratio()` option. Other parameters are specified as described in *Using power twomeans* in [PSS-2] **power twomeans**.

You can also compute the number of clusters or the cluster size in one of the groups given the number of clusters or the cluster size in the other group by specifying the `compute()` option. For example, to compute the number of clusters in the control group, you specify `compute(K1)` and provide the number of clusters in the experimental group in `k2()`. Likewise, to compute the cluster size in the control group, you specify `compute(M1)` and provide the cluster size of the experimental group in `m2()`. You can compute the number of clusters or cluster size for the experimental group in a similar manner.

The power and effect-size determination is the same as described in *Using power twomeans* in [PSS-2] **power twomeans**, but the sample-size information is supplied as the numbers of clusters *kspec* and either cluster sizes using *mspec* or, less commonly, sample sizes using *nspec*.

All computations assume an intraclass correlation of 0.5. You can change this by specifying the `rho()` option. Also, all clusters are assumed to be of the same size unless the coefficient of variation for cluster sizes is specified in the `cvcluster()` option.

By default, the computed numbers of clusters, cluster sizes, and sample sizes are rounded up. However, you can specify the `nfractional` option to see the corresponding fractional values; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. If the `cvcluster()` option is specified when computing cluster sizes, then cluster sizes represent average cluster sizes and are thus not rounded. When sample sizes are specified using *nspec*, fractional cluster sizes may be reported to accommodate the specified numbers of clusters and sample sizes.

Some of `power twomeans, cluster`'s computations require iteration, such as to compute the numbers of clusters for a two-sided test; see *Methods and formulas* for details and [PSS-2] **power** for the descriptions of options that control the iteration procedure.

Computing numbers of clusters

To compute the numbers of clusters in each group, you must either provide the cluster size for each group using *mspec* or specify the `cluster` option and provide the sample sizes of both groups using *nspec*. The most common method is to use *mspec* of `m1()` and `m2()`. In addition, the control- and experimental-group means must be specified.

▷ Example 1: Numbers of clusters for a two-sample means test in a CRD, specify cluster sizes

Consider an example from Ahn, Heo, and Zhang (2015, 37) of a hypothetical cluster randomized trial in which the goal is to assess the effect of a health promotion program on increasing the level of physical activity measured in kcal/kg/day of individuals in church congregations. In this study, the church congregation is the unit of randomization and the individual participant is the unit of analysis. Churches will be randomly assigned either to the experimental group that participates in the promotion program or to the control group. Investigators plan to recruit 20 church members from each participating church and would like to detect a mean difference of 1.1 kcal/kg/day between the experimental and control groups. From previous studies, the common standard deviation is 3.67 kcal/kg/day. The investigator assumes an intraclass correlation of 0.025.

To compute the numbers of churches required to detect a mean change in physical activity of 1.1 kcal/kg/day with 80% power using a 5%-level two-sided test, we type

```
. power twomeans 0 1.1, m1(20) m2(20) sd(3.67) rho(0.025)

Performing iteration ...

Estimated numbers of clusters for a two-sample means test
Cluster randomized design, z test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    1.1000
           m1 =    0.0000
           m2 =    1.1000
           sd =    3.6700

Cluster design:

           M1 =        20
           M2 =        20
          rho =    0.0250

Estimated numbers of clusters and sample sizes:

           K1 =        13
           K2 =        13
           N1 =       260
           N2 =       260
```

We find that with 20 members per church, 13 churches and thus a total of 260 members per group are required to detect a change of 1.1 kcal/kg/day in physical activity given the standard deviation of 3.67 kcal/kg/day with 80% power using a 5%-level two-sided test.

For power twomeans, the actual value of the control-group mean does not change the results as long as the difference between the means is fixed. In this example, we used the control-group mean of 0.

◁

▷ Example 2: Numbers of clusters for a two-sample means test in a CRD, varying cluster sizes

Instead of a constant number of members in each church as in example 1, we assume that the numbers of members selected from each church vary. Suppose that the numbers of members selected from each church have a mean of 20 and a standard deviation of 4 in both groups and thus have a coefficient of variation of 0.2. To compute the numbers of clusters when cluster sizes vary, we specify the coefficient of variation in the cvcluster() option.

```
. power twomeans 0 1.1, m1(20) m2(20) sd(3.67) rho(0.025) cvcluster(0.2)

Performing iteration ...

Estimated numbers of clusters for a two-sample means test
Cluster randomized design, z test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    1.1000
           m1 =    0.0000
           m2 =    1.1000
           sd =    3.6700

Cluster design:

   Average M1 =   20.0000
   Average M2 =   20.0000
          rho =    0.0250
        CV_cl =    0.2000

Estimated numbers of clusters and sample sizes:

           K1 =        14
           K2 =        14
           N1 =       280
           N2 =       280
```

The required number of churches in each group is 14, which is slightly larger than the required number of churches of 13 in example 1. When the number of members selected from each church varies, we need more churches to achieve the same power.

◁

▷ Example 3: Numbers of clusters for a two-sample means test in a CRD, specify sample sizes

Suppose that for our study, we can recruit only 200 members per group because of limited funding. We need to know the number of churches in each group and how many members to recruit in each church. In this case, we specify the n1(200) and n2(200) options. Because none of the k1(), k2(), m1(), or m2() options are specified, we also need to specify the cluster option so that computations are performed for a CRD instead of the conventional individual-level design.

```
. power twomeans 0 1.1, cluster n1(200) n2(200) sd(3.67) rho(0.025)
Performing iteration ...
Estimated numbers of clusters for a two-sample means test
Cluster randomized design, z test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    1.1000
           m1 =    0.0000
           m2 =    1.1000
           sd =    3.6700
Cluster design:
           N1 =       200
           N2 =       200
          rho =    0.0250
Estimated numbers of clusters and cluster sizes:
           K1 =        30
           K2 =        30
   Average M1 =    6.6667
   Average M2 =    6.6667
```

To achieve the desired power, we need to recruit about 6.67 members on average per church from 30 churches to each of the control and experimental groups. `power twomeans, cluster` did not round the cluster sizes of 6.67 to meet our required total of 200 members per group. In practice, you can either decide to recruit 6 members from some churches and 7 from other churches to have roughly constant cluster sizes or decide to change the total number of members you want to recruit.

◁

Computing number of clusters in one group

To compute the number of clusters in one of the groups, you must specify the `compute()` option and the number of clusters in the other group. For example, to compute the number of clusters in the experimental group, you must specify the `compute(K2)` option and provide the number of clusters in the control group in the `k1()` option. Similarly, you can compute the number of clusters for the control group. In addition, you must provide cluster sizes of both groups using *mspec* or sample sizes of both groups using *nspec* and the control- and experimental-group means.

▷ Example 4: Number of clusters in the experimental group for a two-sample means test in a CRD

Continuing with example 1, suppose that we are designing a new study and we are planning to recruit 25 churches for the control group. We want to know the minimum number of churches we need to recruit to the experimental group. Given other study parameters from example 1, we compute the number of churches in the experimental group by specifying the `compute(K2)` option and the number of clusters in the control group of 25 in the `k1()` option.

```
. power twomeans 0 1.1, compute(K2) k1(25) m1(20) m2(20) sd(3.67) rho(0.025)
Performing iteration ...
Estimated experimental-group number of clusters for a two-sample means test
Cluster randomized design, z test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    1.1000
           m1 =    0.0000
           m2 =    1.1000
           sd =    3.6700
Cluster design:
           K1 =        25
           M1 =        20
           M2 =        20
           N1 =       500
          rho =    0.0250
Estimated number of clusters and sample size:
           K2 =         9
           N2 =       180
```

With 25 churches in the control group, we need to recruit 9 churches for the experimental group.

◁

Computing cluster sizes

To compute cluster sizes in both groups, you must provide the numbers of clusters in both groups by using *kspec*. The most common method is to specify the numbers of clusters in the control and experimental groups in the `k1()` and `k2()` options, respectively. In addition, the control- and experimental-group means must be specified.

▷ Example 5: Cluster sizes for a two-sample means test in a CRD

Continuing with example 1, suppose that we are designing a new study and we are planning to recruit 30 churches with 15 churches in each group. Given other study parameters from example 1, we compute the numbers of members to recruit from each church by specifying 15 clusters in the `k1()` and `k2()` options.

```
. power twomeans 0 1.1, k1(15) k2(15) sd(3.67) rho(0.025)
Performing iteration ...
Estimated cluster sizes for a two-sample means test
Cluster randomized design, z test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    1.1000
           m1 =    0.0000
           m2 =    1.1000
           sd =    3.6700
Cluster design:
           K1 =        15
           K2 =        15
          rho =    0.0250
Estimated cluster sizes and sample sizes:
           M1 =        17
           M2 =        17
           N1 =       255
           N2 =       255
```

With 15 churches per group, we need to recruit 17 members per church for a total of 255 members per group.

Computing power

To compute power in a CRD, you supply the sample-size information as the numbers of clusters by using *kspec* along with either the cluster sizes by using *mspec* or, less commonly, the sample sizes by using *nspec*. The most common method is to specify the `k1()`, `k2()`, `m1()`, and `m2()` options. In addition, the control- and experimental-group means must be specified.

Example 6: Power for a two-sample means test in a CRD

Continuing with example 1, suppose that we can recruit 20 members from each of 30 churches (15 churches per group) and we want to compute power for this design. Given other study parameters from example 1, we compute the power by specifying 15 in the `k1()` and `k2()` options and the cluster size of 20 in the `m1()` and `m2()` options:

```
. power twomeans 0 1.1, k1(15) k2(15) m1(20) m2(20) sd(3.67) rho(0.025)
Estimated power for a two-sample means test
Cluster randomized design, z test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1
Study parameters:
        alpha =    0.0500
        delta =    1.1000
           m1 =    0.0000
           m2 =    1.1000
           sd =    3.6700
Cluster design:
           K1 =        15
           K2 =        15
           M1 =        20
           M2 =        20
           N1 =       300
           N2 =       300
          rho =    0.0250
Estimated power:
        power =    0.8560
```

The computed power is about 86%.

Example 7: Multiple values of study parameters

To investigate the effect of the number of clusters in the experimental group on power, we can specify a list of numbers of clusters in the `k2()` option:

```
. power twomeans 0 1.1, k1(15) k2(5(10)45) m1(20) m2(20) sd(3.67) rho(0.025)
> table(power K2)
Estimated power for a two-sample means test
Cluster randomized design, z test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1
```

power	K2
.5704	5
.856	15
.9221	25
.947	35
.9592	45

In this example, we also specified the `table(power K2)` option to list the only two columns that vary. As expected, as the number of clusters in the experimental group increases, the power tends to get closer to 1.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

Computing effect size and experimental-group mean

Effect size δ for a two-sample means test is defined as the difference between the experimental-group mean and the control-group mean, $\delta = \mu_2 - \mu_1$. To compute effect size in a CRD, you supply the sample-size information as the numbers of clusters by using *kspec* along with either the cluster sizes by using *mspec* or, less commonly, the sample sizes by using *nspec*. The most common method is to specify the `k1()`, `k2()`, `m1()`, and `m2()` options. In addition, power and control-group mean must be specified. You must also decide on the direction of the effect, which is specified in the `direction()` option. For the default, upper, meaning $m_2 > m_1$, `power twomeans, cluster` uses `direction(upper)`. For lower, meaning $m_2 < m_1$, specify `direction(lower)`.

▷ Example 8: Effect size for a two-sample means test in a CRD

Continuing with example 6, we may also be interested in finding the minimum value of the difference in physical activity level between the two groups that can be detected with a sample of 15 churches per group, 20 members per church, and 80% power. To compute this, we specify the control-group mean of 0 as the command argument and the required options `k1(15)`, `k2(15)`, `m1(20)`, `m2(20)`, and `power(0.8)` and continue to use `sd(3.67)` and `rho(0.025)`.

```
. power twomeans 0, k1(15) k2(15) m1(20) m2(20) power(0.8) sd(3.67) rho(0.025)

Performing iteration ...

Estimated experimental-group mean for a two-sample means test
Cluster randomized design, z test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1; m2 > m1

Study parameters:

        alpha =    0.0500
        power =    0.8000
           m1 =    0.0000
           sd =    3.6700

Cluster design:

           K1 =        15
           K2 =        15
           M1 =        20
           M2 =        20
           N1 =       300
           N2 =       300
          rho =    0.0250

Estimated effect size and experimental-group mean:

        delta =    1.0196
           m2 =    1.0196
```

Given 15 churches per group with 20 members per church and 80% power, the minimum detectable value of the difference in the physical activity level is about 1.02.

◁

Testing hypotheses about two means in a CRD

There are different ways to account for a CRD or for clustered data when performing hypothesis tests that compare means in two groups. With large samples or when you know the intraclass correlation and group-specific population standard deviations, the simplest way is to use a z test that accounts for clustered data; see [R] **ztest** for details. More commonly, two-level models such as those fit by `mixed` (see [ME] **mixed**) are used because they also allow adjusting for covariates.

In this section, we briefly demonstrate the `ztest` command for comparing means of two groups with clustered data.

▷ Example 9: Two-sample means test with clustered data

Consider example 6 in [R] **ztest** that compared the means of (log) BMI in two groups of patients with type-2 diabetes from a randomized controlled trial of patient-centered care of diabetes in general practice. The two groups included the comparison group that provided patients with routine care and an intervention group that provided patients with patient-centered care; see example 6 in [R] **ztest** for details. We replicate the analysis from that example below.

For clustered data, `ztest` requires that we specify the cluster identifier in the `cluster()` option and population intraclass correlation in the `rho()` option. We must also specify a common population standard deviation or group-specific population standard deviations in the respective options. We specify a common population standard deviation in the `sd()` option.

```
. use https://www.stata-press.com/data/r19/dcfd_trial
(BMI data from Diabetes Care from Diagnosis trial (Kinmonth et al., 1998))
. ztest lbmi, by(group) cluster(practice) rho(0.028) sd(0.35)

Two-sample z test
Cluster variable: practice

Group: Control                          Group: Interv.
  Number of clusters =        20          Number of clusters =        18
  Avg. cluster size  =      5.10          Avg. cluster size  =      7.67
  CV cluster size    =    0.5330          CV cluster size    =    0.5126
  Intraclass corr.   =    0.0280          Intraclass corr.   =    0.0280

------------------------------------------------------------------------------
       Group |      Obs        Mean    Std. err.   Std. dev.   [95% conf. interval]
-------------+----------------------------------------------------------------
     Control |      102     2.62954    .0372502         .35    2.556531    2.702549
      Interv |      138    2.749023    .0332182         .35    2.683916     2.81413
-------------+----------------------------------------------------------------
        diff |           -.1194831    .0499102                -.2173054   -.0216608
------------------------------------------------------------------------------
    diff = mean(Control) - mean(Interv.)                          z =  -2.3940
H0: diff = 0

    Ha: diff < 0                 Ha: diff != 0                 Ha: diff > 0
 Pr(Z < z) = 0.0083         Pr(|Z| > |z|) = 0.0167          Pr(Z > z) = 0.9917
```

There is statistical evidence to reject the null hypothesis that the two group means are the same at the 5% significance level.

Suppose that we want to use the results of this study to design another study that compares the two types of care of diabetes in the same population. Specifically, we want to compute the required number of clusters given the average cluster sizes of 5.1 and 7.67 in two groups, the intraclass correlation of 0.028, and the coefficient of variation of cluster sizes of 0.53, as shown in the output above. The coefficients of variation of cluster sizes are slightly different between the two groups—we use the larger value with `power twomeans`. We also use the observed mean estimates of 2.6 and 2.75 in the computation.

```
. power twomeans 2.6 2.75, m1(5.1) m2(7.67) cvcluster(0.53) rho(0.028) sd(0.35)
Performing iteration ...
Estimated numbers of clusters for a two-sample means test
Cluster randomized design, z test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.1500
           m1 =    2.6000
           m2 =    2.7500
           sd =    0.3500
Cluster design:
   Average M1 =    5.1000
   Average M2 =    7.6700
          rho =    0.0280
        CV_cl =    0.5300
Estimated numbers of clusters and sample sizes:
           K1 =        17
           K2 =        17
           N1 =        87
           N2 =       131
```

The required number of clusters for each group is 17. Given varying cluster sizes, we need to have a total of 87 patients in the control group and a total of 131 patients in the intervention group.

◁

Stored results

`power twomeans, cluster` stores the following in `r()`:

Scalars

`r(alpha)`	significance level
`r(power)`	power
`r(beta)`	probability of a type II error
`r(delta)`	effect size
`r(K1)`	number of clusters in the control group
`r(K2)`	number of clusters in the experimental group
`r(kratio)`	ratio of numbers of clusters, `K2/K1`
`r(M1)`	cluster size of the control group
`r(M2)`	cluster size of the experimental group
`r(mratio)`	ratio of cluster sizes, `M2/M1`
`r(N)`	total sample size
`r(N1)`	sample size of the control group
`r(N2)`	sample size of the experimental group
`r(nratio)`	ratio of sample sizes, `N2/N1`
`r(nfractional)`	1 if `nfractional` is specified, 0 otherwise
`r(onesided)`	1 for a one-sided test, 0 otherwise
`r(m1)`	control-group mean
`r(m2)`	experimental-group mean
`r(diff)`	difference between the experimental- and control-group means
`r(sd)`	common standard deviation of the control and experimental groups
`r(sd1)`	standard deviation of the control group
`r(sd2)`	standard deviation of the experimental group
`r(rho)`	intraclass correlation
`r(CV_cluster)`	coefficient of variation for cluster sizes

r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise
r(init)	initial value for estimated parameter
r(maxiter)	maximum number of iterations
r(iter)	number of iterations performed
r(tolerance)	requested parameter tolerance
r(deltax)	final parameter tolerance achieved
r(ftolerance)	requested distance of the objective function from zero
r(function)	final distance of the objective function from zero
r(converged)	1 if iteration algorithm converged, 0 otherwise

Macros

r(type)	test
r(method)	twomeans
r(design)	CRD
r(direction)	upper or lower
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results

Methods and formulas

For the computation in a CRD, we assume the standard deviations of the two groups are known. See *Known standard deviations* under *Methods and formulas* in [PSS-2] **power twomeans** for the common notation for a two-sample means test.

Methods and formulas are presented under the following headings:

Introduction
Equal cluster sizes
Unequal cluster sizes

Introduction

In a CRD, let K_1 and K_2 be the numbers of clusters in the control and experimental groups, respectively, and M_1 and M_2 be the cluster sizes of the control and experimental groups, respectively. We have $n_1 = K_1M_1$ and $n_2 = K_2M_2$. Let R_k be the ratio of the numbers of clusters, K_2/K_1, and R_m be the ratio of the cluster sizes, M_2/M_1. Let ρ be the intraclass correlation coefficient and DE_1 and DE_2 be the design effect in the control and experimental groups, with

$$\text{DE}_1 = 1 + \rho(M_1 - 1) \qquad \text{and} \qquad \text{DE}_2 = 1 + \rho(M_2 - 1)$$

Similarly to the discussion for the two-sample means test in the individual-level design, the test statistic is

$$\text{TS} = \frac{(\overline{x}_2 - \overline{x}_1) - (\mu_2 - \mu_1)}{\sigma_D}$$

where $\overline{x}_1$ and $\overline{x}_2$ are the sample means of the two groups and σ_D is the standard deviation of the mean difference. The sampling distribution of the test statistic TS under the null hypothesis follows a standard normal distribution; see, for example, Ahn, Heo, and Zhang (2015).

The power $\pi = 1 - \beta$ is computed using

$$\pi = \begin{cases} \Phi\left(\frac{\delta}{\sigma_D} - z_{1-\alpha}\right) & \text{for an upper one-sided test} \\ \Phi\left(-\frac{\delta}{\sigma_D} - z_{1-\alpha}\right) & \text{for a lower one-sided test} \\ \Phi\left(\frac{\delta}{\sigma_D} - z_{1-\alpha/2}\right) + \Phi\left(-\frac{\delta}{\sigma_D} - z_{1-\alpha/2}\right) & \text{for a two-sided test} \end{cases} \quad (1)$$

where $\Phi(\cdot)$ is the c.d.f. of a standard normal distribution, $\delta = \mu_2 - \mu_1$, and σ_D is defined in the subsequent sections.

Equal cluster sizes

When the cluster sizes are equal, the standard deviation of the mean difference σ_D in (1) is computed as

$$\sigma_D = \sqrt{\sigma_1^2 \mathrm{DE}_1/n_1 + \sigma_2^2 \mathrm{DE}_2/n_2}$$

where σ_1^2 and σ_2^2 are group-specific variances.

Given the cluster sizes M_1 and M_2 and the ratio of the numbers of clusters R_k, the numbers of clusters K_1 and K_2 for a one-sided test are computed as follows. K_1 is computed by inverting a one-sided power equation from (1)

$$K_1 = \left(\frac{z_{1-\alpha} - z_\beta}{\mu_2 - \mu_1}\right)^2 \left(\frac{\sigma_1^2 \mathrm{DE}_1}{M_1} + \frac{\sigma_2^2 \mathrm{DE}_2}{M_2 R_k}\right) \quad (2)$$

Then, K_2 is computed using $K_2 = R_k K_1$.

For a one-sided test, to compute the number of clusters in one group given that of the other one—for example, to compute K_1 given K_2—we use the following formula:

$$K_1 = \frac{\sigma_1^2 \mathrm{DE}_1/M_1}{(\mu_2 - \mu_1)^2/(z_{1-\alpha} - z_\beta)^2 - \sigma_2^2 \mathrm{DE}_2/K_2 M_2} \quad (3)$$

Similarly, we can compute K_2 given K_1.

Given the numbers of clusters K_1 and K_2 and the cluster-size ratio R_m, the cluster sizes M_1 and M_2 for a one-sided test are computed as follows. M_1 is computed as

$$M_1 = \frac{(1-\rho)(\sigma_1^2/K_1 + \sigma_2^2/K_2 R_m)}{(\mu_2 - \mu_1)^2/(z_{1-\alpha} - z_\beta)^2 - \rho(\sigma_1^2/K_1 + \sigma_2^2/K_2)} \quad (4)$$

and $M_2 = R_m M_1$.

For a one-sided test, to compute the cluster size in one group given that of the other one—for example, to compute M_1 given M_2—we use the following formula:

$$M_1 = \frac{(1-\rho)(\sigma_1^2/K_1)}{(\mu_2 - \mu_1)^2/(z_{1-\alpha} - z_\beta)^2 - \sigma_2^2 \mathrm{DE}_2/K_2 M_2 - \rho\sigma_1^2/K_1} \quad (5)$$

Similarly, we can compute M_2 given M_1.

The absolute value of the effect size for a one-sided test is computed as follows:

$$|\delta| = \sigma_D(z_{1-\alpha} - z_\beta) \tag{6}$$

Note that the magnitude of the effect size is the same regardless of the direction of the test.

The experimental-group mean for a one-sided test is then computed as

$$\mu_2 = \begin{cases} \mu_1 + (z_{1-\alpha} - z_\beta)\sqrt{\sigma_1^2 \text{DE}_1/n_1 + \sigma_2^2 \text{DE}_2/n_2} & \text{when } \mu_2 > \mu_1 \\ \mu_1 - (z_{1-\alpha} - z_\beta)\sqrt{\sigma_1^2 \text{DE}_1/n_1 + \sigma_2^2 \text{DE}_2/n_2} & \text{when } \mu_2 < \mu_1 \end{cases} \tag{7}$$

The numbers of clusters, cluster sizes, and minimum detectable value of the experimental-group mean for a two-sided test are computed iteratively using the two-sided power equation from (1). The initial values are obtained from the corresponding one-sided equations [(2) through (6)] with $\alpha/2$.

Unequal cluster sizes

For unequal cluster sizes, we assume that the cluster sizes are independent and identically distributed and are small relative to the number of clusters; see Ahn, Heo, and Zhang (2015) for details. Let the coefficient of variation of the cluster sizes be CV_{cl}. According to van Breukelen, Candel, and Berger (2007) and Campbell and Walters (2014), to adjust for varying cluster sizes, define the relative efficiency (RE) of unequal versus equal cluster sizes as

$$\text{RE}_i = 1 - \lambda_i(1 - \lambda_i)\text{CV}_{\text{cl}}^2$$

where $\lambda_i = \rho M_i/(\rho M_i + 1 - \rho)$, where $i = 1$ corresponds to the control group and $i = 2$ corresponds to the experimental group. Under unequal cluster sizes, the standard deviation of the mean difference σ_D becomes

$$\sigma_D = \sqrt{\frac{\sigma_1^2 \text{DE}_1}{n_1 \text{RE}_1} + \frac{\sigma_2^2 \text{DE}_1}{n_2 \text{RE}_2}} \tag{8}$$

By substituting σ_D in (1) and replacing $\sigma_1^2\text{DE}_1$ and $\sigma_2^2\text{DE}_2$ with $\sigma_1^2\text{DE}_1/\text{RE}_1$ and $\sigma_2^2\text{DE}_2/\text{RE}_2$ in (2), (3), and (7), we can obtain the formulas for computing the numbers of clusters and experimental-group mean for a one-sided test. In all other cases, parameters are computed iteratively using the power equations (1) with σ_D as defined in (8).

References

Ahn, C., M. Heo, and S. Zhang. 2015. *Sample Size Calculations for Clustered and Longitudinal Outcomes in Clinical Research.* Boca Raton, FL: CRC Press. https://doi.org/10.1201/b17822.

Campbell, M. J., and S. J. Walters. 2014. *How to Design, Analyse and Report Cluster Randomised Trials in Medicine and Health Related Research.* Chichester, UK: Wiley. https://doi.org/10.1002/9781118763452.

Gallis, J. A., F. Li, H. Yu, and E. L. Turner. 2018. cvcrand and cptest: Commands for efficient design and analysis of cluster randomized trials using constrained randomization and permutation tests. *Stata Journal* 18: 357–378.

van Breukelen, G. J. P., M. J. J. M. Candel, and M. P. F. Berger. 2007. Relative efficiency of unequal versus equal cluster sizes in cluster randomized and multicentre trials. *Statistics in Medicine* 26: 2589–2603. https://doi.org/10.1002/sim.2740.

Also see

[PSS-2] **power twomeans** — Power analysis for a two-sample means test

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[ME] **mixed** — Multilevel mixed-effects linear regression

[R] **ztest** — *z* tests (mean-comparison tests, known variance)

power pairedmeans — Power analysis for a two-sample paired-means test

Description

`power pairedmeans` computes sample size, power, or target mean difference for a two-sample paired-means test. By default, it computes sample size for given power and the values of the null and alternative mean differences. Alternatively, it can compute power for given sample size and the values of the null and alternative mean differences or the target mean difference for given sample size, power, and the null mean difference. Also see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

For precision and sample-size analysis for a CI for the difference between two means from paired samples, see [PSS-3] **ciwidth pairedmeans**.

Quick start

Sample size for a test of H_0: $\mu_2 - \mu_1 = d = 0$ versus H_a: $d \neq 0$ given alternative pretreatment mean $m_{a1} = 73$ and alternative posttreatment mean $m_{a2} = 57$ with standard deviation of the differences $\sigma_d = 36$ using default power of 0.8 and significance level $\alpha = 0.05$

```
power pairedmeans 73 57, sddiff(36)
```

Same as above, specified using the difference between means of -16

```
power pairedmeans, altdiff(-16) sddiff(36)
```

Same as above, but instead of standard deviation of the differences, specify correlation between paired observations of 0.5 with pretreatment standard deviation of 29 and posttreatment standard deviation of 40

```
power pairedmeans 73 57, corr(.5) sd1(29) sd2(40)
```

For differences in means of -20, -18, -16, -14, -12, and -10

```
power pairedmeans, altdiff(-20(2)-10) sddiff(36)
```

Power for a sample size of 23

```
power pairedmeans 73 57, sddiff(36) n(23)
```

Effect size and target mean difference for sample sizes 20, 30, and 40 with power of 0.85

```
power pairedmeans 73, sddiff(36) power(.85) n(20(10)40)
```

Same as above, but display results as a graph of target mean difference versus sample size

```
power pairedmeans 73, sddiff(36) power(.85) n(20(10)40) graph
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
power pairedmeans m_a1 m_a2 , corrspec [ power(numlist) options ]
```

Compute power

```
power pairedmeans m_a1 m_a2 , corrspec n(numlist) [ options ]
```

Compute effect size and target mean difference

```
power pairedmeans [ m_a1 ] , corrspec n(numlist) power(numlist) [ options ]
```

where *corrspec* is one of

```
sddiff()
corr() [ sd() ]
corr() [ sd1() sd2() ]
```

m_{a1} is the alternative pretreatment mean or the pretreatment mean under the alternative hypothesis, and m_{a2} is the alternative posttreatment mean or the value of the posttreatment mean under the alternative hypothesis. m_{a1} and m_{a2} may each be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

options	Description
Main	
* alpha(*numlist*)	significance level; default is alpha(0.05)
* power(*numlist*)	power; default is power(0.8)
* beta(*numlist*)	probability of type II error; default is beta(0.2)
* n(*numlist*)	sample size; required to compute power or effect size
nfractional	allow fractional sample size
* nulldiff(*numlist*)	null difference, the difference between the posttreatment mean and the pretreatment mean under the null hypothesis; default is nulldiff(0)
* altdiff(*numlist*)	alternative difference $d_a = m_{a2} - m_{a1}$, the difference between the posttreatment mean and the pretreatment mean under the alternative hypothesis
* sddiff(*numlist*)	standard deviation σ_d of the differences; may not be combined with corr()
* corr(*numlist*)	correlation between paired observations; required unless sddiff() is specified
* sd(*numlist*)	common standard deviation; default is sd(1) and requires corr()
* sd1(*numlist*)	standard deviation of the pretreatment group; requires corr()
* sd2(*numlist*)	standard deviation of the posttreatment group; requires corr()
knownsd	request computation assuming a known standard deviation σ_d; default is to assume an unknown standard deviation
* fpc(*numlist*)	finite population correction (FPC) as a sampling rate or population size
direction(upper|lower)	direction of the effect for effect-size determination; default is direction(upper), which means that the postulated value of the parameter is larger than the hypothesized value
onesided	one-sided test; default is two sided
parallel	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[no]table[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
saving(*filename* [, replace])	save the table data to *filename*; use replace to overwrite existing *filename*
Graph	
graph[(*graphopts*)]	graph results; see [PSS-2] **power, graph**

Iteration

`init(#)`	initial value for sample size or mean difference; default is to use normal approximation
`iterate(#)`	maximum number of iterations; default is `iterate(500)`
`tolerance(#)`	parameter tolerance; default is `tolerance(1e-12)`
`ftolerance(#)`	function tolerance; default is `ftolerance(1e-12)`
[`no`]`log`	suppress or display iteration log
[`no`]`dots`	suppress or display iterations as dots
`notitle`	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the `parallel` option.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`notitle` does not appear in the dialog box.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`N`	number of subjects	N
`delta`	effect size	δ
`d0`	null mean difference	d_0
`da`	alternative mean difference	d_a
`ma1`	alternative pretreatment mean	μ_{a1}
`ma2`	alternative posttreatment mean	μ_{a2}
`sd_d`	standard deviation of the differences	σ_d
`sd`	common standard deviation	σ
`sd1`	standard deviation of the pretreatment group	σ_1
`sd2`	standard deviation of the posttreatment group	σ_2
`corr`	correlation between paired observations	ρ
`fpc`	FPC as a population size	N_{pop}
	FPC as a sampling rate	γ
`target`	target parameter; synonym for `da`	
`_all`	display all supported columns	

Column `beta` is shown in the default table in place of column `power` if specified.

Columns `ma1`, `ma2`, `sd`, `sd1`, `sd2`, `corr`, and `fpc` are shown in the default table if specified.

Options

Main

`alpha()`, `power()`, `beta()`, `n()`, `nfractional`; see [PSS-2] **power**. The `nfractional` option is allowed only for sample-size determination.

`nulldiff(`*numlist*`)` specifies the difference between the posttreatment mean and the pretreatment mean under the null hypothesis. The default is `nulldiff(0)`, which means that the pretreatment mean equals the posttreatment mean under the null hypothesis.

`altdiff(`*numlist*`)` specifies the alternative difference $d_a = m_{a2} - m_{a1}$, the difference between the posttreatment mean and the pretreatment mean under the alternative hypothesis. This option is the alternative to specifying the alternative means m_{a1} and m_{a2}. If m_{a1} is specified in combination with `altdiff(#)`, then $m_{a2} = \# + m_{a1}$.

`sddiff(`*numlist*`)` specifies the standard deviation σ_d of the differences. Either `sddiff()` or `corr()` must be specified.

`corr(`*numlist*`)` specifies the correlation between paired, pretreatment and posttreatment, observations. This option along with `sd1()` and `sd2()` or `sd()` is used to compute the standard deviation of the differences unless that standard deviation is supplied directly in the `sddiff()` option. Either `corr()` or `sddiff()` must be specified.

`sd(`*numlist*`)` specifies the common standard deviation of the pretreatment and posttreatment groups. Specifying `sd(#)` implies that both `sd1()` and `sd2()` are equal to #. Options `corr()` and `sd()` are used to compute the standard deviation of the differences unless that standard deviation is supplied directly with the `sddiff()` option. The default is `sd(1)`.

`sd1(`*numlist*`)` specifies the standard deviation of the pretreatment group. Options `corr()`, `sd1()`, and `sd2()` are used to compute the standard deviation of the differences unless that standard deviation is supplied directly with the `sddiff()` option.

`sd2(`*numlist*`)` specifies the standard deviation of the posttreatment group. Options `corr()`, `sd1()`, and `sd2()` are used to compute the standard deviation of the differences unless that standard deviation is supplied directly with the `sddiff()` option.

`knownsd` requests that the standard deviation of the differences σ_d be treated as known in the computations. By default, the standard deviation is treated as unknown, and the computations are based on a paired t test, which uses a Student's t distribution as a sampling distribution of the test statistic. If `knownsd` is specified, the computation is based on a paired z test, which uses a normal distribution as the sampling distribution of the test statistic.

`fpc(`*numlist*`)` requests that a finite population correction be used in the computation. If `fpc()` has values between 0 and 1, it is interpreted as a sampling rate, n/N, where N is the total number of units in the population. When sample size n is specified, if `fpc()` has values greater than n, it is interpreted as a population size, but it is an error to have values between 1 and n. For sample-size determination, `fpc()` with a value greater than 1 is interpreted as a population size. It is an error for `fpc()` to have a mixture of sampling rates and population sizes.

`direction()`, `onesided`, `parallel`; see [PSS-2] **power**.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies the initial value of the sample size for the sample-size determination or the initial value of the mean difference for the effect-size determination. The default is to use a closed-form normal approximation to compute an initial value of the sample size or mean difference.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following option is available with `power pairedmeans` but is not shown in the dialog box:

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `power pairedmeans` command and the methodology for power and sample-size analysis for a two-sample paired-means test. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis and [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Introduction

The analysis of paired means is commonly used in settings such as repeated-measures designs with before and after measurements on the same individual or cross-sectional studies of paired measurements from twins. For example, a company might initiate a voluntary exercise program and would like to test that the average weight loss of participants from beginning to six months is greater than zero. Or a school district might design an intensive remedial program for students with low math scores and would like to know if the students' math scores improve from the pretest to the posttest. For paired data, the inference is made on the mean difference accounting for the dependence between the two groups.

This entry describes power and sample-size analysis for the inference about the population mean difference performed using hypothesis testing. Specifically, we consider the null hypothesis H_0: $d = d_0$ versus the two-sided alternative hypothesis H_a: $d \neq d_0$, the upper one-sided alternative H_a: $d > d_0$, or the lower one-sided alternative H_a: $d < d_0$. The parameter d is the mean difference between the posttreatment mean μ_2 and pretreatment mean μ_1.

A two-sample paired-means test assumes that the two correlated samples are drawn from two normal populations or that the sample size is large. When the population variances are known, the sampling distribution of the test statistic under the null hypothesis is standard normal, and the corresponding test is known as a paired z test. If the population variances are unknown, the sampling distribution of the test statistic under the null hypothesis is Student's t, and the corresponding test is known as a paired t test.

The random sample is typically drawn from an infinite population. When the sample is drawn from a population of a fixed size, sampling variability must be adjusted for a finite population size.

The `power pairedmeans` command provides power and sample-size analysis for the comparison of two correlated means using a paired t test or a paired z test.

Using power pairedmeans

`power pairedmeans` computes sample size, power, or target mean difference for a two-sample paired-means test. All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option. You can specify the `onesided` option to request a one-sided test.

By default, all computations are based on a paired t test, which assumes an unknown standard deviation of the differences. For a known standard deviation, you can specify the `knownsd` option to request a paired z test.

For all computations, you must specify either the standard deviation of the differences in the `sddiff()` option or the correlation between the paired observations in the `corr()` option. If you specify the `corr()` option, then individual standard deviations of the pretreatment and posttreatment groups may also be specified in the respective `sd1()` and `sd2()` options. By default, their values are set to 1. When the two standard deviations are equal, you may specify the common standard deviation in the `sd()` option instead of specifying them individually.

To compute sample size, you must specify the pretreatment and posttreatment means under the alternative hypothesis, m_{a1} and m_{a2}, respectively, and, optionally, the power of the test in the `power()` option. The default power is set to 0.8.

To compute power, you must specify the sample size in the `n()` option and the pretreatment and posttreatment means under the alternative hypothesis, m_{a1} and m_{a2}, respectively.

Instead of the alternative means m_{a1} and m_{a2}, you can specify the difference $m_{a2} - m_{a1}$ between the alternative posttreatment mean and the alternative pretreatment mean in the `altdiff()` option when computing sample size or power.

By default, the difference between the posttreatment mean and the pretreatment mean under the null hypothesis is set to zero. You may specify other values in the `nulldiff()` option.

To compute effect size, the standardized difference between the alternative and null mean differences, and target mean difference, you must specify the sample size in the `n()` option, the power in the `power()` option, and, optionally, the direction of the effect. The direction is upper by default, `direction(upper)`, which means that the target mean difference is assumed to be larger than the specified null value. This is also equivalent to the assumption of a positive effect size. You can change the direction to be lower, which means that the target mean difference is assumed to be smaller than the specified null value, by specifying the `direction(lower)` option. This is equivalent to assuming a negative effect size.

By default, the computed sample size is rounded up. You can specify the `nfractional` option to see the corresponding fractional sample size; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. The `nfractional` option is allowed only for sample-size determination.

Some of `power pairedmeans`'s computations require iteration. For example, when the standard deviation of the differences is unknown, computations use a noncentral Student's t distribution. Its degrees of freedom depends on the sample size, and the noncentrality parameter depends on the sample size and effect size. Therefore, the sample-size and effect-size determinations require iteration. The default initial

values of the estimated parameters are obtained by using a closed-form normal approximation. They may be changed by specifying the `init()` option. See [PSS-2] **power** for the descriptions of other options that control the iteration procedure.

All computations assume an infinite population. For a finite population, use the `fpc()` option to specify a sampling rate or a population size. When this option is specified, a finite population correction is applied to the standard deviation of the differences. The correction factor depends on the sample size; therefore, computing sample size in this case requires iteration. The initial value for sample-size determination in this case is based on the corresponding normal approximation with a finite population size.

In the following sections, we describe the use of `power pairedmeans` accompanied by examples for computing sample size, power, and target mean difference.

Computing sample size

To compute sample size, you must specify the pretreatment and posttreatment means under the alternative hypothesis, m_{a1} and m_{a2}, respectively, or the difference between them in `altdiff()` and, optionally, the power of the test in the `power()` option. A default power of 0.8 is assumed if `power()` is not specified.

▷ Example 1: Sample size for a two-sample paired-means test

Consider a study of low birthweight (LBW) infants as in Howell (2002, 186). The variable of interest is the Bayley mental development index (MDI) of infants when they are 6-, 12-, and 24-months old. Previous research suggested that the MDI scores for LBW children might decline significantly between 6 and 24 months of age. Suppose we would like to conduct a similar study where the null hypothesis of interest is no difference between 6-month and 24-month MDI scores, H_0: $d = 0$, and the two-sided alternative is H_a: $d \neq 0$, implying the existence of a difference.

In this example, we use the estimates from Howell (2002, 193) as our study parameters. The mean MDI score of a 6-month group was estimated to be 111. We want to obtain the minimum sample size that is required to detect the mean MDI score of 106.71 in a 24-month group with a power of 80% using a 5%-level two-sided test. The standard deviation of the differences was previously estimated to be 16.04. To compute the sample size, we specify the alternative means after the command name and standard deviation of the differences in `sddiff()`.

```
. power pairedmeans 111 106.71, sddiff(16.04)
Performing iteration ...
Estimated sample size for a two-sample paired-means test
Paired t test
H0: d = d0  versus  Ha: d != d0
Study parameters:
        alpha =    0.0500          ma1 =  111.0000
        power =    0.8000          ma2 =  106.7100
        delta =   -0.2675
           d0 =    0.0000
           da =   -4.2900
         sd_d =   16.0400
Estimated sample size:
            N =       112
```

As we mentioned in the previous section, sample-size determination requires iteration in the case of an unknown standard deviation. By default, `power pairedmeans` suppresses the iteration log, which may be displayed by specifying the `log` option.

A sample of 112 subjects is required for the test to detect the resulting difference of −4.29 with a power of 80%.

Study parameters are divided into two columns. The parameters that are always displayed are listed in the first column, and the parameters that are displayed only if they are specified are listed in the second column.

In this example, we specified optional command arguments containing the alternative pretreatment mean `ma1` and the alternative posttreatment mean `ma2`. Because these arguments are optional, they are listed in the second column.

◁

▷ Example 2: Specifying mean differences

Instead of the individual alternative means, we can specify their difference, $106.71 - 111 = -4.29$, in the `altdiff()` option.

```
. power pairedmeans, altdiff(-4.29) sddiff(16.04)
Performing iteration ...
Estimated sample size for a two-sample paired-means test
Paired t test
H0: d = d0  versus  Ha: d != d0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =   -0.2675
           d0 =    0.0000
           da =   -4.2900
         sd_d =   16.0400
Estimated sample size:
            N =       112
```

We obtain the same results as in example 1.

◁

Example 3: Specifying individual standard deviations

Howell (2002) also reported the group-specific standard deviations: 13.85 in the 6-month group and 12.95 in the 24-month group. Using the values of individual standard deviations and the standard deviation of the differences from the previous example, we obtain the correlation between the 6-month group and the 24-month group to be $(13.85^2 + 12.95^2 - 16.04^2)/(2 \times 13.85 \times 12.95) = 0.285$. To compute the sample size, we specify the group-specific standard deviations in `sd1()` and `sd2()` and the correlation in `corr()`.

```
. power pairedmeans 111 106.71, corr(0.285) sd1(13.85) sd2(12.95)
Performing iteration ...
Estimated sample size for a two-sample paired-means test
Paired t test
H0: d = d0  versus  Ha: d != d0
Study parameters:
        alpha =    0.0500          ma1 =  111.0000
        power =    0.8000          ma2 =  106.7100
        delta =   -0.2675          sd1 =   13.8500
           d0 =    0.0000          sd2 =   12.9500
           da =   -4.2900         corr =    0.2850
         sd_d =   16.0403
Estimated sample size:
            N =       112
```

We obtain the same sample size as in example 1.

The correlation and standard deviations are reported in the second column.

Example 4: Specifying common standard deviation

If standard deviations in both groups are equal, we may specify the common standard deviation in option `sd()`. As a demonstration, we use the average of the individual standard deviations $(13.85 + 12.95)/2 = 13.4$ as our common standard deviation.

```
. power pairedmeans 111 106.71, corr(0.285) sd(13.4)
Performing iteration ...
Estimated sample size for a two-sample paired-means test
Paired t test assuming sd1 = sd2 = sd
H0: d = d0  versus  Ha: d != d0
Study parameters:
        alpha =    0.0500          ma1 =  111.0000
        power =    0.8000          ma2 =  106.7100
        delta =   -0.2677           sd =   13.4000
           d0 =    0.0000         corr =    0.2850
           da =   -4.2900
         sd_d =   16.0241
Estimated sample size:
            N =       112
```

The resulting standard deviation of the differences of 16.0241 is close to our earlier estimate of 16.04, so the computed sample size is the same as the sample size in example 1.

▷ Example 5: Nonzero null

In all the previous examples, we assumed that the difference between the 6-month and 24-month means is zero under the null hypothesis. For a nonzero null hypothesis, you can specify the corresponding null value in the `nulldiff()` option.

Continuing with example 2, we will suppose that we are testing the nonzero null hypothesis of H_0: $d = d_0 = -1$. We compute the sample size as follows:

```
. power pairedmeans, nulldiff(-1) altdiff(-4.29) sddiff(16.04)
Performing iteration ...
Estimated sample size for a two-sample paired-means test
Paired t test
H0: d = d0  versus  Ha: d != d0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =   -0.2051
           d0 =   -1.0000
           da =   -4.2900
         sd_d =   16.0400
Estimated sample size:
            N =       189
```

Compared with example 2, the absolute value of the effect size `delta` decreases to 0.2051, and thus a larger sample of 189 subjects is required to detect this smaller effect.

◁

Computing power

To compute power, you must specify the sample size in the `n()` option and the pretreatment and post-treatment means under the alternative hypothesis, m_{a1} and m_{a2}, respectively, or the difference between them in the `altdiff()` option.

▷ Example 6: Power of a two-sample paired-means test

Continuing with example 1, we will suppose that because of limited resources, we anticipate to obtain a sample of only 100 subjects. To compute power, we specify the sample size in the `n()` option:

```
. power pairedmeans 111 106.71, n(100) sddiff(16.04)
Estimated power for a two-sample paired-means test
Paired t test
H0: d = d0  versus  Ha: d != d0
Study parameters:
        alpha =    0.0500          ma1 =  111.0000
            N =       100          ma2 =  106.7100
        delta =   -0.2675
           d0 =    0.0000
           da =   -4.2900
         sd_d =   16.0400
Estimated power:
        power =    0.7545
```

Compared with example 1, the power decreases to 75.45%.

◁

Example 7: Known standard deviation

In the case of a known standard deviation σ_d, you can specify the `knownsd` option to request a paired z test. Using the same study parameters as in example 6, we can compute the power as follows:

```
. power pairedmeans 111 106.71, n(100) sddiff(16.04) knownsd

Estimated power for a two-sample paired-means test
Paired z test
H0: d = d0  versus  Ha: d != d0

Study parameters:

        alpha =    0.0500         ma1 =  111.0000
            N =       100         ma2 =  106.7100
        delta =   -0.2675
           d0 =    0.0000
           da =   -4.2900
         sd_d =   16.0400

Estimated power:

        power =    0.7626
```

The power of 76.26% of a paired z test is close to the power of 75.45% of a paired t test obtained in example 6.

◁

Example 8: Multiple values of study parameters

Continuing with example 3, we will suppose that we would like to assess the effect of varying correlation on the power of our study. The standard deviation of the MDI scores for infants aged 6 months is 13.85 and that for infants aged 24 months is 12.95, which are obtained from Howell (2002, 193). We believe the data on pairs to be positively correlated because we expect a 6-month-old infant with a high score to have a high score at 24 months of age as well. We specify a range of correlations between 0.1 and 0.9 with the step size of 0.1 in the `corr()` option:

```
. power pairedmeans 111 106.71, n(100) sd1(13.85) sd2(12.95) corr(0.1(0.1)0.9)
> table(alpha N power corr sd_d delta)

Estimated power for a two-sample paired-means test
Paired t test
H0: d = d0  versus  Ha: d != d0
```

alpha	N	power	corr	sd_d	delta
.05	100	.656	.1	17.99	-.2385
.05	100	.7069	.2	16.96	-.2529
.05	100	.7632	.3	15.87	-.2703
.05	100	.8239	.4	14.7	-.2919
.05	100	.8859	.5	13.42	-.3196
.05	100	.9425	.6	12.01	-.3571
.05	100	.983	.7	10.41	-.412
.05	100	.9988	.8	8.518	-.5037
.05	100	1	.9	6.057	-.7083

As the correlation increases, the power also increases. This is because the standard deviation of the differences is negatively related to correlation when the correlation is positive. As the correlation increases, the standard deviation of the differences decreases, thus resulting in higher power. Likewise, the opposite is true when the correlation is negative.

For multiple values of parameters, the results are automatically displayed in a table. In the above, we use the `table()` option to build a custom table. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

Computing effect size and target mean difference

Effect size δ for a two-sample paired-means test is defined as a standardized difference between the alternative mean difference d_a and the null mean difference d_0, $\delta = (d_a - d_0)/\sigma_d$.

Sometimes, we may be interested in determining the smallest effect and the corresponding mean difference that yield a statistically significant result for prespecified sample size and power. In this case, power, sample size, and the alternative pretreatment mean must be specified. By default, the null mean difference is set to 0. In addition, you must also decide on the direction of the effect: upper, meaning $d_a > d_0$, or lower, meaning $d_a < d_0$. The direction may be specified in the `direction()` option; `direction(upper)` is the default.

▷ Example 9: Minimum detectable value of the effect size

Continuing with example 6, we may be interested to find the minimum effect size with a power of 80% given a sample of 100 subjects. To compute the smallest effect size and the corresponding target mean difference, we specify the sample size `n(100)`, power `power(0.8)`, and the standard deviation of the differences `sddiff(16.04)`:

```
. power pairedmeans 111, n(100) power(0.8) sddiff(16.04)

Performing iteration ...

Estimated target parameters for a two-sample paired-means test
Paired t test
H0: d = d0  versus  Ha: d != d0; da > d0

Study parameters:

        alpha =    0.0500          ma1 =  111.0000
        power =    0.8000
            N =       100
           d0 =    0.0000
         sd_d =   16.0400

Estimated effect size and target parameters:

        delta =    0.2829
           da =    4.5379
          ma2 =  115.5379
```

The smallest detectable value of the effect size is 0.28, which corresponds to the alternative mean difference of 4.54. Compared with example 1, for the same power of 80%, the target mean difference increased to 4.54 when the sample size was reduced to 100 subjects.

◁

Testing a hypothesis about two correlated means

In this section, we demonstrate the use of the `ttest` command for testing hypotheses about paired means. Suppose we wish to test the hypothesis that the means of the paired samples are the same. We can use the `ttest` command to do this. We demonstrate the use of this command using the fictional `bpwide` dataset; see [R] **ttest** for details.

▷ Example 10: Testing means from paired data

Suppose that we have a sample of 120 patients. We are interested in investigating whether a certain drug induces a change in the systolic blood pressure. We record blood pressures for each patient before and after the drug is administered. In this case, each patient serves as his or her own control. We wish to test whether the mean difference between the posttreatment and pretreatment systolic blood pressures are significantly different from zero.

```
. use https://www.stata-press.com/data/r19/bpwide
(Fictional blood-pressure data)
. ttest bp_before == bp_after
Paired t test
```

Variable	Obs	Mean	Std. err.	Std. dev.	[95% conf.	interval]
bp_bef~e	120	156.45	1.039746	11.38985	154.3912	158.5088
bp_after	120	151.3583	1.294234	14.17762	148.7956	153.921
diff	120	5.091667	1.525736	16.7136	2.070557	8.112776

```
     mean(diff) = mean(bp_before - bp_after)                      t =   3.3372
 H0: mean(diff) = 0                               Degrees of freedom =      119

 Ha: mean(diff) < 0           Ha: mean(diff) != 0           Ha: mean(diff) > 0
 Pr(T < t) = 0.9994         Pr(|T| > |t|) = 0.0011          Pr(T > t) = 0.0006
```

We find statistical evidence to reject the null hypothesis of H_0: $d = 0$ versus the two-sided alternative H_a: $d \neq 0$ at the 5% significance level; the p-value $= 0.0011$.

We use the estimates of this study to perform a sample-size analysis we would have conducted before the study.

```
. power pairedmeans, altdiff(5.09) sddiff(16.71)
Performing iteration ...
Estimated sample size for a two-sample paired-means test
Paired t test
H0: d = d0  versus  Ha: d != d0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.3046
           d0 =    0.0000
           da =    5.0900
         sd_d =   16.7100
Estimated sample size:
            N =        87
```

We find that the sample size required to detect a mean difference of 5.09 for given standard deviation of the differences of 16.71 with 80% power using a 5%-level two-sided test is 87.

◁

Video examples

Sample-size calculation for comparing sample means from two paired samples

Power calculation for comparing sample means from two paired samples

Minimum detectable effect size for comparing sample means from two paired samples

Stored results

power pairedmeans stores the following in r():

Scalars

r(alpha)	significance level
r(power)	power
r(beta)	probability of a type II error
r(delta)	effect size
r(N)	sample size
r(nfractional)	1 if nfractional is specified, 0 otherwise
r(onesided)	1 for a one-sided test, 0 otherwise
r(d0)	difference between the posttreatment and pretreatment means under the null hypothesis
r(da)	difference between the posttreatment and pretreatment means under the alternative hypothesis
r(ma1)	pretreatment mean under the alternative hypothesis
r(ma2)	posttreatment mean under the alternative hypothesis
r(corr)	correlation between paired observations
r(sd_d)	standard deviation of the differences
r(sd1)	standard deviation of the pretreatment group
r(sd2)	standard deviation of the posttreatment group
r(sd)	common standard deviation
r(knownsd)	1 if option knownsd is specified, 0 otherwise
r(fpc)	finite population correction
r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise
r(init)	initial value for sample size or target mean difference
r(maxiter)	maximum number of iterations
r(iter)	number of iterations performed
r(tolerance)	requested parameter tolerance
r(deltax)	final parameter tolerance achieved
r(ftolerance)	requested distance of the objective function from zero
r(function)	final distance of the objective function from zero
r(converged)	1 if iteration algorithm converged, 0 otherwise

Macros

r(type)	test
r(method)	pairedmeans
r(direction)	upper or lower
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results

Methods and formulas

Consider a sequence of n paired observations denoted by X_{ij} for $i = 1, \ldots, n$ and groups $j = 1, 2$. Individual observation corresponds to the pair (X_{i1}, X_{i2}), and inference is made on the differences within the pairs. Let $d = \mu_2 - \mu_1$ denote the mean difference, where μ_j is the population mean of group j, and $D_i = X_{i2} - X_{i1}$ denote the difference between individual observations. Let d_0 and d_a denote the null and alternative values of the mean difference d. Let $\overline{d} = \sum_{i=1}^{n} D_i/n$ denote the sample mean difference.

Unlike a two-sample means test where we consider two independent samples, a paired-means test allows the two groups to be dependent. As a result, the standard deviation of the differences is given by $\sigma_d = \sqrt{\sigma_1^2 + \sigma_2^2 - 2\rho\sigma_1\sigma_2}$, where σ_1 and σ_2 are the pretreatment and posttreatment group standard deviations, respectively, and ρ is the correlation between the paired measurements.

Power, sample-size, and effect-size determination for a paired-means test is analogous to a one-sample mean test where the sample of differences D_i's is treated as a single sample. See *Methods and formulas* in [PSS-2] **power onemean**.

Also see Armitage, Berry, and Matthews (2002); Dixon and Massey (1983); and Chow et al. (2018) for more details.

References

Armitage, P., G. Berry, and J. N. S. Matthews. 2002. *Statistical Methods in Medical Research*. 4th ed. Oxford: Blackwell.

Chow, S.-C., J. Shao, H. Wang, and Y. Lokhnygina. 2018. *Sample Size Calculations in Clinical Research*. 3rd ed. Boca Raton, FL: CRC Press.

Dixon, W. J., and F. J. Massey, Jr. 1983. *Introduction to Statistical Analysis*. 4th ed. New York: McGraw–Hill.

Howell, D. C. 2002. *Statistical Methods for Psychology*. 5th ed. Belmont, CA: Wadsworth.

Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power repeated** — Power analysis for repeated-measures analysis of variance

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-3] **ciwidth pairedmeans** — Precision analysis for a paired-means-difference CI

[PSS-5] **Glossary**

[R] **ttest** — t tests (mean-comparison tests)

power oneproportion — Power analysis for a one-sample proportion test

Description

`power oneproportion` computes sample size, power, or target proportion for a one-sample proportion test. By default, it computes sample size for given power and the values of the proportion parameters under the null and alternative hypotheses. Alternatively, it can compute power for given sample size and values of the null and alternative proportions or the target proportion for given sample size, power, and the null proportion. For power and sample-size analysis in a cluster randomized design, see [PSS-2] **power oneproportion, cluster**. Also see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Quick start

Sample size for a test of H_0: $\pi = 0.2$ versus H_a: $\pi \neq 0.2$ with null proportion $p_0 = 0.2$, alternative proportion $p_a = 0.1$, default power of 0.8, and significance level $\alpha = 0.05$

```
power oneproportion .2 .1
```

Same as above, but for p_a equal to 0.05, 0.075, 0.1, 0.125, and 0.15

```
power oneproportion .2 (.05(.025).15)
```

Same as above, but display results as a graph of sample size versus alternative proportion

```
power oneproportion .2 (.05(.025).15), graph
```

Sample size for one-sided test with power of 0.9

```
power oneproportion .2 .1, power(.9) onesided
```

Sample size for a Wald test instead of the default score test

```
power oneproportion .2 .1, test(wald)
```

Power for a sample size of 120

```
power oneproportion .2 .1, n(120)
```

Same as above, but for sample sizes of 110, 120, 130, and 140

```
power oneproportion .2 .1, n(110(10)140)
```

Effect size and target proportion for $p_0 = 0.3$, sample size of 75, and power of 0.8

```
power oneproportion .3, n(75) power(.8)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
power oneproportion p0 pa [ , power(numlist) options ]
```

Compute power

```
power oneproportion p0 pa , n(numlist) [ options ]
```

Compute effect size and target proportion

```
power oneproportion p0 , n(numlist) power(numlist) [ options ]
```

where p_0 is the null (hypothesized) proportion or the value of the proportion under the null hypothesis and p_a is the alternative (target) proportion or the value of the proportion under the alternative hypothesis. p_0 and p_a may each be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

options	Description
`test(`*test*`)`	specify the type of test; default is `test(score)`
Main	
* `alpha(`*numlist*`)`	significance level; default is `alpha(0.05)`
* `power(`*numlist*`)`	power; default is `power(0.8)`
* `beta(`*numlist*`)`	probability of type II error; default is `beta(0.2)`
* `n(`*numlist*`)`	sample size; required to compute power or effect size
`nfractional`	allow fractional sample size
* `diff(`*numlist*`)`	difference between the alternative proportion and the null proportion, $p_a - p_0$; specify instead of the alternative proportion p_a
`critvalues`	show critical values for the binomial test
`continuity`	apply continuity correction to the normal approximation of the discrete distribution
`direction(upper|lower)`	direction of the effect for effect-size determination; default is `direction(upper)`, which means that the postulated value of the parameter is larger than the hypothesized value
`onesided`	one-sided test; default is two sided
`parallel`	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
`[no]table[(`*tablespec*`)]`	suppress table or display results as a table; see [PSS-2] **power, table**
`saving(`*filename* `[, replace])`	save the table data to *filename*; use `replace` to overwrite existing *filename*
Graph	
`graph[(`*graphopts*`)]`	graph results; see [PSS-2] **power, graph**
Iteration	
`init(#)`	initial value for sample size or proportion
`iterate(#)`	maximum number of iterations; default is `iterate(500)`
`tolerance(#)`	parameter tolerance; default is `tolerance(1e-12)`
`ftolerance(#)`	function tolerance; default is `ftolerance(1e-12)`
`[no]log`	suppress or display iteration log
`[no]dots`	suppress or display iterations as dots
`cluster`	perform computations for a CRD; see [PSS-2] **power oneproportion, cluster**
`notitle`	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the `parallel` option.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`cluster` and `notitle` do not appear in the dialog box.

test	Description
`score`	score test; the default
`wald`	Wald test
`binomial`	binomial test

`test()` does not appear in the dialog box. The dialog box selected is determined by the `test()` specification.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`alpha_a`	observed significance level	α_a
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`N`	number of subjects	N
`delta`	effect size	δ
`p0`	null proportion	p_0
`pa`	alternative proportion	p_a
`diff`	difference between the alternative and null proportions	$p_a - p_0$
`C_l`	lower critical value	C_l
`C_u`	upper critical value	C_u
`target`	target parameter; synonym for `pa`	
`_all`	display all supported columns	

Column `beta` is shown in the default table in place of column `power` if specified.

Column `diff` is shown in the default table if specified.

Columns `alpha_a`, `C_l`, and `C_u` are available when the `test(binomial)` option is specified.

Columns `C_l` and `C_u` are shown in the default table, if the `critvalues` option is specified.

Options

`test(`*test*`)` specifies the type of the test for power and sample-size computations. *test* is one of `score`, `wald`, or `binomial`.

`score` requests computations for the score test. This is the default test.

`wald` requests computations for the Wald test. This corresponds to computations using the value of the alternative proportion instead of the default null proportion in the formula for the standard error of the estimator of the proportion.

`binomial` requests computations for the binomial test. The computation using the binomial distribution is not available for sample-size and effect-size determinations; see example 7 for details. Iteration options are not allowed with this test.

Main

`alpha()`, `power()`, `beta()`, `n()`, `nfractional`; see [PSS-2] **power**. The `nfractional` option is allowed only for sample-size determination.

`diff(`*numlist*`)` specifies the difference between the alternative proportion and the null proportion, $p_a - p_0$. You can specify either the alternative proportion p_a as a command argument or the difference between the two proportions in `diff()`. If you specify `diff(#)`, the alternative proportion is computed as $p_a = p_0 + \#$. This option is not allowed with the effect-size determination.

`critvalues` requests that the critical values be reported when the computation is based on the binomial distribution.

`continuity` requests that continuity correction be applied to the normal approximation of the discrete distribution. `continuity` cannot be specified with `test(binomial)`.

`direction()`, `onesided`, `parallel`; see [PSS-2] **power**.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies the initial value of the sample size for the sample-size determination or the initial value of the proportion for the effect-size determination.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following options are available with `power oneproportion` but are not shown in the dialog box:

`cluster`; see [PSS-2] **power oneproportion, cluster**.

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `power oneproportion` command and the methodology for power and sample-size analysis for a one-sample proportion test. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis and [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests. Also see [PSS-2] **power oneproportion, cluster** for power and sample-size analysis in a cluster randomized design.

Introduction

There are many examples of studies where a researcher would like to compare an observed proportion with a hypothesized proportion. A political campaign might like to know if the proportion of a country's population that supports a new legislative initiative is greater than 50%. A veterinary drug manufacturer might test a new topical treatment to kill fleas on dogs. It would like to know the sample size necessary to demonstrate that the treatment is effective in ridding at least 80% of the test dogs of fleas. The Nevada Gaming Control Board might test a Las Vegas casino's slot machines to verify that it meets the statutory minimum payout percentage of 75%. The board would like to know the number of "pulls" necessary to reject the one-sided null hypothesis that the payout percentage is less than 75%.

The analysis of proportions is carried out in experiments or observational studies where the response variable is binary. Each observation is an outcome from a Bernoulli trial with a fixed probability p of observing an event of interest in a population. Hypothesis testing of binomial outcomes relies on a set of assumptions: 1) Bernoulli outcome is observed a fixed number of times; 2) the probability p is fixed across all trials; and 3) individual trials are independent.

This entry describes power and sample-size analysis for the inference about the population proportion performed using hypothesis testing. Specifically, we consider the null hypothesis H_0: $p = p_0$ versus the two-sided alternative hypothesis H_a: $p \neq p_0$, the upper one-sided alternative H_a: $p > p_0$, or the lower one-sided alternative H_a: $p < p_0$.

Two common hypothesis tests for a one-sample proportion are the small-sample binomial test and the asymptotic (large-sample) normal test. The binomial test is based on the binomial distribution, the exact sampling distribution, of the test statistic and is commonly known as an "exact binomial" test. The asymptotic normal test is based on the large-sample normal approximation of the sampling distribution of the test statistic and is often referred to as a z test.

`power oneproportion` provides power and sample-size analysis for both the binomial and a large-sample z test of a one-sample proportion.

Using power oneproportion

`power oneproportion` computes sample size, power, or target proportion for a one-sample proportion test. All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option. You can specify the `onesided` option to request a one-sided test.

`power oneproportion` performs power analysis for three different tests, which can be specified within the `test()` option. The default is a large-sample score test (`test(score)`), which approximates the sampling distribution of the test statistic by the standard normal distribution. You may instead request computations based on a large-sample Wald test by specifying the `test(wald)` option. For power determination, you can also request the small-sample binomial test by specifying the `test(binomial)` option. The binomial test is not available for the sample-size and effect-size determinations; see example 7 for details.

To compute sample size, you must specify the proportions under the null and alternative hypotheses, p_0 and p_a, respectively, and, optionally, the power of the test in the `power()` option. The default power is set to 0.8.

To compute power, you must specify the sample size in the `n()` option and the proportions under the null and alternative hypotheses, p_0 and p_a, respectively.

Instead of the alternative proportion p_a, you may specify the difference $p_a - p_0$ between the alternative proportion and the null proportion in the `diff()` option when computing sample size or power.

To compute effect size, the difference between the alternative and null proportions, and target proportion, you must specify the sample size in the `n()` option, the power in the `power()` option, the null proportion p_0, and, optionally, the direction of the effect. The direction is upper by default, `direction(upper)`, which means that the target proportion is assumed to be larger than the specified null value. You can change the direction to lower, which means that the target proportion is assumed to be smaller than the specified null value, by specifying the `direction(lower)` option.

By default, the computed sample size is rounded up. You can specify the `nfractional` option to see the corresponding fractional sample size; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. The `nfractional` option is allowed only for sample-size determination.

Some of `power oneproportion`'s computations require iteration. For example, for a large-sample z test, sample size for a two-sided test is obtained by iteratively solving a nonlinear power equation. The default initial value for the sample size for the iteration procedure is obtained using a closed-form one-sided formula. If desired, it may be changed by specifying the `init()` option. See [PSS-2] **power** for the descriptions of other options that control the iteration procedure.

In the following sections, we describe the use of `power oneproportion` accompanied with examples for computing sample size, power, and target proportion.

Computing sample size

To compute sample size, you must specify the proportions under the null and alternative hypotheses, p_0 and p_a, respectively, and, optionally, the power of the test in the `power()` option. A default power of 0.8 is assumed if `power()` is not specified.

▷ Example 1: Sample size for a one-sample proportion test

Consider a study of osteoporosis in postmenopausal women from Chow et al. (2018, 45). The term "osteoporosis" refers to the decrease in bone mass that is most prevalent in postmenopausal women. Females diagnosed with osteoporosis have vertebral bone density more than 10% below the average bone density of women with similar demographic characteristics such as age, height, weight, and race.

The World Health Organization (WHO) defines osteoporosis as having the bone density value that is smaller than 2.5 standard deviations below the peak bone mass levels in young women. Suppose investigators wish to assess the effect of a new treatment on increasing the bone density for women diagnosed with osteoporosis. The treatment is deemed successful if a subject's bone density improves by more than one standard deviation of her measured bone density.

Suppose that previous studies have reported a response rate of 30% for women with increased bone density after treatment. Investigators expect the new treatment to generate a higher response rate of roughly 50%. The goal is to obtain the minimum required sample size to detect an alternative proportion of 0.5 using the test of H_0: $p = 0.3$ versus H_a: $p \neq 0.3$ with 80% power and 5% significance level. To compute sample size, we specify the null and alternative proportions after the command name:

```
. power oneproportion 0.3 0.5
Performing iteration ...
Estimated sample size for a one-sample proportion test
Score z test
H0: p = p0  versus  Ha: p != p0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.2000
           p0 =    0.3000
           pa =    0.5000
Estimated sample size:
            N =        44
```

We find that at least 44 subjects are needed to detect a change in proportion from 0.3 to 0.5 with 80% power using a 5%-level two-sided test.

◁

▷ Example 2: Specifying the difference between proportions

Instead of the alternative proportion, we can specify the difference of $0.05 - 0.03 = 0.2$ between the alternative proportion and the null proportion in the `diff()` option and obtain the same results:

```
. power oneproportion 0.3, diff(0.2)
Performing iteration ...
Estimated sample size for a one-sample proportion test
Score z test
H0: p = p0  versus  Ha: p != p0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.2000
           p0 =    0.3000
           pa =    0.5000
         diff =    0.2000
Estimated sample size:
            N =        44
```

The difference between proportions is now also displayed in the output.

◁

▷ Example 3: Wald test

The default computation is based on a score test and thus uses the null proportion as the estimate of the true proportion in the formula for the standard error. We can request the computation based on a Wald test by specifying the `test(wald)` option. In this case, the alternative proportion will be used as an estimate of the true proportion in the formula for the standard error.

```
. power oneproportion 0.3 0.5, test(wald)
Performing iteration ...
Estimated sample size for a one-sample proportion test
Wald z test
H0: p = p0  versus  Ha: p != p0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.2000
           p0 =    0.3000
           pa =    0.5000
Estimated sample size:
            N =        50
```

We find that the required sample size increases to 50 subjects.

◁

Computing power

To compute power, you must specify the sample size in the `n()` option and the proportions under the null and alternative hypotheses, p_0 and p_a, respectively.

▷ Example 4: Power of a one-sample proportion test

Continuing with example 1, we will suppose that we are designing a new study and anticipate to obtain a sample of 30 subjects. To compute the power corresponding to this sample size given the study parameters from example 1, we specify the sample size of 30 in the `n()` option:

```
. power oneproportion 0.3 0.5, n(30)
Estimated power for a one-sample proportion test
Score z test
H0: p = p0  versus  Ha: p != p0
Study parameters:
        alpha =    0.0500
            N =        30
        delta =    0.2000
           p0 =    0.3000
           pa =    0.5000
Estimated power:
        power =    0.6534
```

As expected, with a smaller sample size, we achieve a lower power (only 65.34%).

◁

▷ Example 5: Multiple values of study parameters

To see the effect of sample size on power, we can specify a range of sample sizes in the `n()` option.

```
. power oneproportion 0.3 0.5, n(40(1)50)
Estimated power for a one-sample proportion test
Score z test
H0: p = p0  versus  Ha: p != p0
```

alpha	power	N	delta	p0	pa
.05	.7684	40	.2	.3	.5
.05	.7778	41	.2	.3	.5
.05	.787	42	.2	.3	.5
.05	.7958	43	.2	.3	.5
.05	.8043	44	.2	.3	.5
.05	.8124	45	.2	.3	.5
.05	.8203	46	.2	.3	.5
.05	.8279	47	.2	.3	.5
.05	.8352	48	.2	.3	.5
.05	.8422	49	.2	.3	.5
.05	.849	50	.2	.3	.5

As expected, power is an increasing function of the sample size.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

▷ Example 6: Sign test

We can use `power oneproportion` to perform power and sample-size analysis for a nonparametric sign test comparing the median of a sample with a reference value. The sign test for comparing a median is simply a test of a binomial proportion with the reference (null) value of 0.5, H_0: $p = 0.5$.

For example, consider a study similar to the one described in example 1. Suppose we want to test whether the median bone density exceeds a threshold value in a population of females who received a certain treatment. This is equivalent to testing whether the proportion p of bone-density values exceeding the threshold is greater than 0.5, that is, H_0: $p = 0.5$ versus H_a: $p > 0.5$. Suppose that from previous studies such proportion was estimated to be 0.7. We anticipate to enroll 30 subjects and would like to compute the corresponding power of an upper one-sided small-sample binomial test to detect the change in proportion from 0.5 to 0.7.

```
. power oneproportion 0.5 0.7, n(30) test(binomial) onesided

Estimated power for a one-sample proportion test
Binomial test
H0: p = p0  versus  Ha: p > p0

Study parameters:

        alpha =    0.0500
            N =        30
        delta =    0.2000
           p0 =    0.5000
           pa =    0.7000

Estimated power and alpha:

        power =    0.7304
 actual alpha =    0.0494
```

For a sample size of 30 subjects, we obtain a power of 73% to detect the difference of 0.2 between the alternative and null values. In addition to power, `power oneproportion` also displays the actual (observed) significance level, which is 0.0494 in our example and is very close to the specified significance level of 0.05.

When the sampling distribution of the test statistic is discrete such as for the binomial test, the specified nominal significance level may not be possible to precisely achieve, because the space of the observed significance levels is discrete. As such, `power oneproportion` also displays the observed significance level given the specified sample size, power, and other study parameters. Also see example 7.

◁

▷ Example 7: Saw-toothed power function

In example 6, we briefly described one issue arising with power and sample-size analysis for the binomial test. The observed significance levels are discrete because the binomial sampling distribution of the test statistic is discrete. Another related issue arising because of the discrete nature of the sampling distribution is the nonmonotonic relationship between power and sample size—as the sample size increases, the corresponding power may not necessarily increase. The power function may have a so-called *saw-toothed* shape (Chernick and Liu 2002), where it increases initially, then drops, then increases again, and so on. See figure 1 below for an example.

To demonstrate the issue, we return to example 5 and plot powers for a range of sample size values between 45 and 60. We specify the `graph()` option to produce a graph and the `table()` option to produce a table; see [PSS-2] **power, graph** and [PSS-2] **power, table** for more details about the graphical and tabular outputs from `power`. Within `graph()`, we request that the reference line be plotted on the y axis at a power of 0.8 and that the data points bee labeled with the corresponding sample sizes. Within `table()`, we specify the `formats()` suboption to display only three digits after the decimal point for the `power` and `alpha_a` columns. We also specify the `critvalues` option to display columns containing critical values in the table.

```
. power oneprop 0.3 0.5, n(45(1)60) test(binomial) critvalues
> table(, formats(alpha_a "%7.3f" power "%7.3f"))
> graph(yline(0.8) plotopts(mlabel(N)))
Estimated power for a one-sample proportion test
Binomial test
H0: p = p0  versus  Ha: p != p0
```

alpha	alpha_a	power	N	delta	p0	pa	C_l	C_u
.05	0.034	0.724	45	.2	.3	.5	7	21
.05	0.035	0.769	46	.2	.3	.5	7	21
.05	0.037	0.809	47	.2	.3	.5	7	21
.05	0.026	0.765	48	.2	.3	.5	7	22
.05	0.042	0.804	49	.2	.3	.5	8	22
.05	0.031	0.760	50	.2	.3	.5	8	23
.05	0.031	0.799	51	.2	.3	.5	8	23
.05	0.033	0.834	52	.2	.3	.5	8	23
.05	0.037	0.795	53	.2	.3	.5	9	24
.05	0.037	0.830	54	.2	.3	.5	9	24
.05	0.038	0.860	55	.2	.3	.5	9	24
.05	0.028	0.825	56	.2	.3	.5	9	25
.05	0.043	0.855	57	.2	.3	.5	10	25
.05	0.044	0.881	58	.2	.3	.5	10	25
.05	0.032	0.851	59	.2	.3	.5	10	26
.05	0.033	0.877	60	.2	.3	.5	10	26

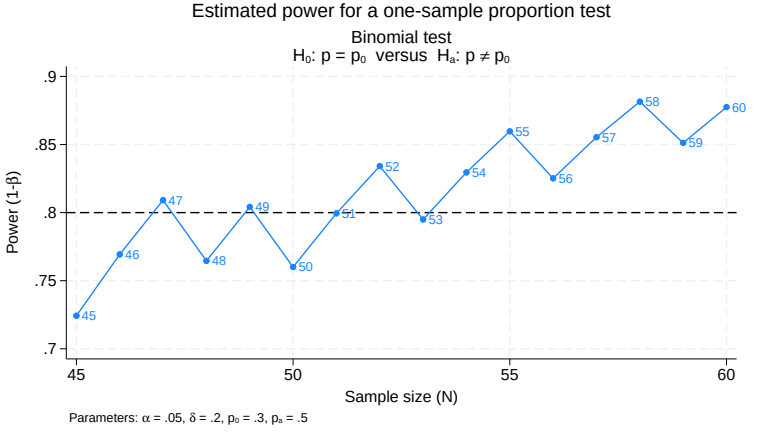


Figure 1. Saw-toothed power function

The power is not a monotonic function of the sample size. Also from the table, we can see that all the observed significance levels are smaller than the specified level of 0.05.

To better understand what is going on, we will walk through the steps of power determination. First, the critical values are determined as the minimum value C_l and the maximum value C_u between 0 and n that satisfy the following inequalities,

$$\Pr(X \leq C_l | p = p_0) \leq \alpha/2 \text{ and } \Pr(X \geq C_u | p = p_0) \leq \alpha/2$$

where the number of successes X has a binomial distribution with the total number of trials n and a probability of a success in a single trial p, $X \sim \text{Bin}(n, p)$. The power is then computed as the sum of the above two probabilities with $p = p_a$.

For example, let's compute the power for the first row of the table. The sample size is 45, the lower critical value is 7, and the upper critical value is 21. We use the probability functions `binomial()` and `binomialtail()` to compute the respective lower- and upper-tailed probabilities of the binomial distribution.

```
. di "Lower tail: " binomial(45,7,0.3)
Lower tail: .0208653
. di "Upper tail: " binomialtail(45,21,0.3)
Upper tail: .01352273
. di "Obs. level: " binomial(45,7,0.3) + binomialtail(45,21,0.3)
Obs. level: .03438804
. di "Power: " binomial(45,7,0.5) + binomialtail(45,21,0.5)
Power: .7242594
```

Each of the tails is less than 0.025 ($\alpha/2 = 0.05/2 = 0.025$). The observed significance level and power match the results from the first row of the table.

Now let's increase the lower critical value by one, $C_l = 8$, and decrease the upper critical value by one, $C_u = 20$:

```
. di "Lower tail: " binomial(45,8,0.3)
Lower tail: .04711667
. di "Upper tail: " binomialtail(45,20,0.3)
Upper tail: .02834511
```

Each of the tail probabilities now exceeds 0.025. If we could use values between 7 and 8 and between 20 and 21, we could match the tails exactly to 0.025, and then the monotonicity of the power function would be preserved. This is impossible for the binomial distribution (or any discrete distribution) because the number of successes must be integers.

Because of the saw-toothed nature of the power curve, obtaining an optimal sample size becomes tricky. If we wish to have power of 80%, then from the above table and graph, we see that potential sample sizes are 47, 49, 52, 54, and so on. One may be tempted to choose the smallest sample size for which the power is at least 80%. This, however, would not guarantee that the power is at least 80% for any larger sample size. Instead, Chernick and Liu (2002) suggest selecting the smallest sample size after which the troughs of the power curve do not go below the desired power. Following this recommendation in our example, we would pick a sample size of 54, which corresponds to the observed significance level of 0.037 and power of 0.83.

In the above, we showed the power curve for the sample sizes between 45 and 60. It may be a good idea to also look at the power plot for larger sample sizes to verify that the power continues to increase and does not drop below the desired power level.

◁

Computing effect size and target proportion

In an analysis of a one-sample proportion, the effect size δ is often defined as the difference between the alternative proportion and the null proportion, $\delta = p_a - p_0$.

Sometimes, we may be interested in determining the smallest effect and the corresponding alternative or target proportion that yield a statistically significant result for prespecified sample size and power. In this case, power, sample size, and null proportion must be specified. In addition, you must also decide on the direction of the effect: upper, meaning $p_a > p_0$, or lower, meaning $p_a < p_0$. The direction may be specified in the `direction()` option; `direction(upper)` is the default.

▷ Example 8: Minimum detectable value of the proportion

Continuing with example 4, we may also be interested to find the minimum value of the proportion that can be detected with a power of 80% given a sample of 30 subjects. To compute this, after the command name, we specify the null proportion of 0.3, sample size `n(30)`, and power `power(0.8)`:

```
. power oneproportion 0.3, n(30) power(0.8)

Performing iteration ...

Estimated target proportion for a one-sample proportion test
Score z test
H0: p = p0  versus  Ha: p != p0; pa > p0

Study parameters:

        alpha =    0.0500
        power =    0.8000
            N =        30
           p0 =    0.3000

Estimated effect size and target proportion:

        delta =    0.2406
           pa =    0.5406
```

The smallest detectable value of the proportion is 0.54.

In the above, we assumed the effect to be in the upper direction, $p_a > p_0$. We can obtain the results in the lower direction by specifying the `direction(lower)` option.

◁

Performing hypothesis tests on proportion

In this section, we briefly demonstrate how you can test hypotheses about proportions; see [R] **prtest** and [R] **bitest** for details. Suppose we wish to test the hypothesis that the proportion is different from a reference value on the collected data. We can use the `prtest` command or the `bitest` command to do this.

▷ Example 9: Testing for proportion

We use `lbw.dta`, which contains data on birthweights of infants from a sample of 189 females. One of the variables in the dataset is variable `ui`, which records the presence or absence of uterine irritability. Although the real objective of this study is different, suppose we wish to test the null hypothesis that the proportion of women in a sample who experience uterine irritability is equal to 0.20. We can use the `prtest` command to perform a large-sample test of a single proportion.

```
. use https://www.stata-press.com/data/r19/lbw
(Hosmer & Lemeshow data)
. prtest ui==0.2
One-sample test of proportion                   Number of obs      =       189
```

Variable	Mean	Std. err.	[95% conf. interval]	
ui	.1481481	.0258404	.0975019	.1987944

```
    p = proportion(ui)                                          z =  -1.7821
H0: p = 0.2
     Ha: p < 0.2                 Ha: p != 0.2                 Ha: p > 0.2
 Pr(Z < z) = 0.0374         Pr(|Z| > |z|) = 0.0747          Pr(Z > z) = 0.9626
```

We do not have statistical evidence to reject the null hypothesis of H_0: $p = 0.2$ versus a two-sided alternative H_a: $p \neq 0.2$ at least at the 5% significance level; the p-value $= 0.0747 > 0.05$.

If our true objective were to study uterine irritability in the population of females, we would have performed the corresponding power and sample-size analysis before collecting the data. For example, using the estimates of `lbw.dta`, we can use `power oneproportion` to compute the required sample size for a 5%-level two-sided large-sample z test to detect the change in proportion from the reference value of 0.2 to approximately 0.148 with a power of, say, 80%:

```
. power oneproportion 0.2 0.148
Performing iteration ...
Estimated sample size for a one-sample proportion test
Score z test
H0: p = p0  versus  Ha: p != p0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =   -0.0520
           p0 =    0.2000
           pa =    0.1480
Estimated sample size:
            N =       434
```

We find that we need 434 subjects, many more than the current sample of 189, to detect the specified change in proportions.

◁

Video examples

Sample-size calculation for comparing a sample proportion to a reference value

Power calculation for comparing a sample proportion to a reference value

Minimum detectable effect size for comparing a sample proportion to a reference value using Stata

Stored results

power oneproportion stores the following in r():

Scalars

r(alpha)	significance level
r(alpha_a)	actual significance level of the binomial method
r(power)	power
r(beta)	probability of a type II error
r(delta)	effect size
r(N)	sample size
r(nfractional)	1 if nfractional is specified, 0 otherwise
r(onesided)	1 for a one-sided test, 0 otherwise
r(p0)	proportion under the null hypothesis
r(pa)	proportion under the alternative hypothesis
r(diff)	difference between the alternative and null proportions
r(C_l)	lower critical value of the binomial distribution
r(C_u)	upper critical value of the binomial distribution
r(continuity)	1 if continuity correction is used, 0 otherwise
r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise
r(init)	initial value for sample size or proportion
r(maxiter)	maximum number of iterations
r(iter)	number of iterations performed
r(tolerance)	requested parameter tolerance
r(deltax)	final parameter tolerance achieved
r(ftolerance)	requested distance of the objective function from zero
r(function)	final distance of the objective function from zero
r(converged)	1 if iteration algorithm converged, 0 otherwise

Macros

r(type)	test
r(method)	oneproportion
r(test)	score, wald, or binomial
r(direction)	upper or lower
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results

Methods and formulas

Let $x_1, \ldots, x_n$ be a sequence of n independent and identically distributed Bernoulli random variates. Let $x_i = 1$ denote a success and $x_i = 0$ denote a failure. Let $P(x_i = 1) = p$ denote the probability of a success in the population. Each individual observation is a Bernoulli trial with a success probability p, which implies that the sum $X = \sum_{i=1}^{n} x_i$ has a binomial distribution with mean np and standard deviation $\sqrt{np(1-p)}$. Let

$$\hat{p} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \text{and} \qquad \text{se}(\hat{p}) = \sqrt{\frac{\hat{p}(1-\hat{p})}{n}}$$

denote the sample proportion and its standard error, respectively. Let p_0 and p_a denote the null and alternative values of the proportion parameter, respectively.

A one-sample proportion test involves testing the null hypothesis H_0: $p = p_0$ versus the two-sided alternative hypothesis H_a: $p \neq p_0$, the upper one-sided alternative H_a: $p > p_0$, or the lower one-sided alternative H_a: $p < p_0$.

If the `nfractional` option is not specified, the computed sample size is rounded up.

The following formulas are based on Chow et al. (2018).

Methods and formulas are presented under the following headings:

Large-sample normal approximation

For a large sample, the distribution of the sample proportion $\hat{p}$ may be approximated by the normal distribution with mean p and variance $p(1-p)/n$. Two test statistics are considered: the score test statistic $z = (\hat{p} - p_0)/\sqrt{p_0(1-p_0)/n}$ and the Wald test statistic $z = (\hat{p} - p_0)/\sqrt{\hat{p}(1-\hat{p})/n}$. The score test statistic uses the null value of the proportion to construct the standard error, which leads to its sampling distribution being closer to the standard normal distribution than if the Wald statistic were used (Agresti [2013, 13]).

Let α be the significance level, β be the probability of a type II error, and $z_{1-\alpha}$ and z_β be the $(1-\alpha)$th and the βth quantiles of the standard normal distribution. Also let $\eta = \sqrt{\{p_0(1-p_0)\}/\{p_a(1-p_a)\}}$.

The power $\pi = 1 - \beta$ of the score z test is computed using

$$\pi = \begin{cases} \Phi\left(\frac{\sqrt{n}(p_a-p_0)-c}{\sqrt{p_a(1-p_a)}} - z_{1-\alpha}\eta\right) & \text{upper one sided} \\ \Phi\left(\frac{-\sqrt{n}(p_a-p_0)-c}{\sqrt{p_a(1-p_a)}} - z_{1-\alpha}\eta\right) & \text{lower one sided} \\ \Phi\left(\frac{\sqrt{n}(p_a-p_0)-c}{\sqrt{p_a(1-p_a)}} - z_{1-\alpha/2}\eta\right) + \Phi\left(\frac{-\sqrt{n}(p_a-p_0)-c}{\sqrt{p_a(1-p_a)}} - z_{1-\alpha/2}\eta\right) & \text{two sided} \end{cases} \quad (1)$$

where $\Phi(\cdot)$ is the cdf of the standard normal distribution, and c is the normal-approximation continuity correction: $c = 1/(2\sqrt{n})$ if the `continuity` option is specified, and $c = 0$ otherwise.

The power of the Wald z test can be obtained from (1) by replacing the term $p_0(1-p_0)$ in η with $p_a(1-p_a)$ so that $\eta = 1$.

The sample size n for a one-sided test is computed using

$$n = \left\{ \frac{z_{1-\alpha}\sqrt{p_0(1-p_0)} + z_{1-\beta}\sqrt{p_a(1-p_a)}}{\delta} \right\}^2$$

If the `continuity` option is specified, the sample size n_c for a one-sided test is computed as

$$n_c = \frac{n}{4}\left(1 + \sqrt{1 + \frac{2}{n|p_a - p_0|}}\right)^2$$

where n is the sample size computed without the correction (Fleiss, Levin, and Paik 2003; Levin and Chen 1999).

The sample size for a two-sided test and minimum detectable value of the proportion are computed iteratively using the corresponding power equation from (1).

Binomial test

Power of the binomial test is computed using the binomial (exact) sampling distribution of the test statistic. Consider a one-sided test given by

$$H_0\colon p = p_0 \qquad \text{versus} \qquad H_a\colon p > p_0$$

Let X denote the number of successes in the sample. The null hypothesis is rejected if X is greater than a critical value k such that the resulting p-value is less than or equal to the significance level α.

The p-value for testing the above one-sided hypothesis can be obtained from the following equation:

$$P(X \geq k; n, p_0) = \sum_{i=k}^{n} \binom{n}{i} p_0^i (1-p_0)^{n-i}$$

The p-value for testing the two-sided hypothesis $H_a\colon p \neq p_0$ is given by

$$2 \times \min\left\{P(X \geq k; n, p_0), P(X \leq k; n, p_0)\right\}$$

For a one-sided test, the power of the test is computed from the following nonlinear equation:

$$\pi = 1 - \beta = \sum_{k=0}^{n} \binom{n}{k} p_a^k (1-p_a)^{n-k} I\left\{P(X \geq k; n, p_0) \leq \alpha\right\}$$

Power for a two-sided test can be obtained by replacing the indicator function above with $I\left[2 \times \min\left\{P(X \geq k; n, p_0), P(X \leq k; n, p_0)\right\} \leq \alpha\right]$.

The computational details may be found in Krishnamoorthy and Peng (2007).

References

Agresti, A. 2013. *Categorical Data Analysis*. 3rd ed. Hoboken, NJ: Wiley.

Chernick, M. R., and C. Y. Liu. 2002. The saw-toothed behavior of power versus sample size and software solutions: Single binomial proportion using exact methods. *American Statistician* 56: 149–155. https://doi.org/10.1198/000313002317572835.

Chow, S.-C., J. Shao, H. Wang, and Y. Lokhnygina. 2018. *Sample Size Calculations in Clinical Research*. 3rd ed. Boca Raton, FL: CRC Press.

Fleiss, J. L., B. Levin, and M. C. Paik. 2003. *Statistical Methods for Rates and Proportions*. 3rd ed. New York: Wiley. https://doi.org/10.1002/0471445428.

Krishnamoorthy, K., and J. Peng. 2007. Some properties of the exact and score methods for binomial proportion and sample size calculation. *Communications in Statistics—Simulation and Computation* 36: 1171–1186. https://doi.org/10.1080/03610910701569218.

Levin, B., and X. Chen. 1999. Is the one-half continuity correction used once or twice to derive a well-known approximate sample size formula to compare two independent binomial distributions? *American Statistician* 53: 62–66. https://doi.org/10.1080/00031305.1999.10474431.

Also see

[PSS-2] **power oneproportion, cluster** — Power analysis for a one-sample proportion test, CRD

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[ADAPT] **gsdesign oneproportion** — Group sequential design for a one-sample proportion test

[R] **bitest** — Binomial probability test

[R] **prtest** — Tests of proportions

power oneproportion, cluster — Power analysis for a one-sample proportion test, CRD

Description

`power oneproportion, cluster` computes the number of clusters, cluster size, power, or target proportion for a one-sample proportion test in a cluster randomized design (CRD). It computes the number of clusters given cluster size, power, and the values of the null and alternative proportions. It also computes cluster size given the number of clusters, power, and the values of the null and alternative proportions. Alternatively, it computes power given the number of clusters, cluster size, and the values of the null and alternative proportions, or it computes the target proportion given the number of clusters, cluster size, power, and the null proportion. See [PSS-2] **power oneproportion** for a general discussion of power and sample-size analysis for a one-sample proportion test. Also see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Quick start

Compute number of clusters for two-sided test of H_0: $\pi = 0.2$ versus H_a: $\pi \neq 0.2$ with null proportion $p_0 = 0.2$, alternative proportion $p_a = 0.1$, and cluster size of 5, using default intraclass correlation of 0.5, power of 0.8, and significance level $\alpha = 0.05$

```
power oneproportion 0.2 0.1, m(5)
```

Same as above, but with an intraclass correlation of 0.7

```
power oneproportion 0.2 0.1, m(5) rho(0.7)
```

Same as above, but the cluster size varies with a coefficient of variation of 0.6

```
power oneproportion 0.2 0.1, m(5) rho(0.7) cvcluster(0.6)
```

Compute cluster size when 52 clusters are sampled:

```
power oneproportion 0.2 0.1, k(52)
```

Power for 52 clusters with cluster size of 5

```
power oneproportion 0.2 0.1, k(52) m(5)
```

Power for 20, 30, 40, and 50 clusters

```
power oneproportion 0.2 0.1, k(20(10)50) m(5)
```

Same as above, but display results in a graph of power versus number of clusters

```
power oneproportion 0.2 0.1, k(20(10)50) m(5) graph
```

Effect size and target proportion for $p_0 = 0.2$ with 40 clusters of size 5, power of 0.9, and $\alpha = 0.01$, and default direction `upper`

```
power oneproportion 0.2, k(40) m(5) power(0.9) alpha(0.01)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute number of clusters

```
power oneproportion p0 pa , { m(numlist) | n(numlist) cluster } [ options ]
```

Compute cluster size

```
power oneproportion p0 pa , k(numlist) [ options ]
```

Compute power

```
power oneproportion p0 pa , k(numlist) { m(numlist) | n(numlist) } [ options ]
```

Compute effect size and target proportion

```
power oneproportion p0 , k(numlist) { m(numlist) | n(numlist) } power(numlist)
    [ options ]
```

where p_0 is the null (hypothesized) proportion or the value of the proportion under the null hypothesis and p_a is the alternative (target) proportion or the value of the proportion under the alternative hypothesis. p_0 and p_a may each be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

options	Description
Main	
`cluster`	perform computations for a CRD; implied by `k()` or `m()`
* `alpha(`*numlist*`)`	significance level; default is `alpha(0.05)`
* `power(`*numlist*`)`	power; default is `power(0.8)`
* `beta(`*numlist*`)`	probability of type II error; default is `beta(0.2)`
* `k(`*numlist*`)`	number of clusters
* `m(`*numlist*`)`	cluster size
* `n(`*numlist*`)`	number of observations
`nfractional`	allow fractional number of clusters, cluster size, and sample size
* `diff(`*numlist*`)`	difference between the alternative proportion and the null proportion, $p_a - p_0$; specify instead of the alternative proportion p_a
* `rho(`*numlist*`)`	intraclass correlation; default is `rho(0.5)`
* `cvcluster(`*numlist*`)`	coefficient of variation for cluster sizes
`direction(upper|lower)`	direction of the effect for effect-size determination; default is `direction(upper)`, which means that the postulated value of the parameter is larger than the hypothesized value
`onesided`	one-sided test; default is two sided
`parallel`	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[`no`]`table`[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
`saving(`*filename* [`, replace`]`)`	save the table data to *filename*; use `replace` to overwrite existing *filename*
Graph	
`graph`[(*graphopts*)]	graph results; see [PSS-2] **power, graph**
Iteration	
`init(#)`	initial value for number of clusters, cluster size, or proportion
`iterate(#)`	maximum number of iterations; default is `iterate(500)`
`tolerance(#)`	parameter tolerance; default is `tolerance(1e-12)`
`ftolerance(#)`	function tolerance; default is `ftolerance(1e-12)`
[`no`]`log`	suppress or display iteration log
[`no`]`dots`	suppress or display iterations as dots
`notitle`	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the `parallel` option.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`notitle` does not appear in the dialog box.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`K`	number of clusters	K
`M`	cluster size	M
`N`	number of observations	N
`delta`	effect size	δ
`p0`	null proportion	p_0
`pa`	alternative proportion	p_a
`diff`	difference between the alternative and null proportions	$p_a - p_0$
`rho`	intraclass correlation	ρ
`CV_cluster`	coefficient of variation for cluster sizes	CV_{cl}
`target`	target parameter; synonym for `pa`	
`_all`	display all supported columns	

Column `beta` is shown in the default table in place of column `power` if specified.

Columns `diff` and `CV_cluster` are shown in the default table if specified.

Options

Main

`cluster` specifies that computations should be performed for a CRD. This option is implied when either the `k()` or `m()` option is specified. It is required if the `n()` option is used to compute the number of clusters.

`alpha()`, `power()`, `beta()`; see [PSS-2] **power**.

`k(`*numlist*`)` specifies the number of clusters. This option is required to compute the cluster size, power, or effect size.

`m(`*numlist*`)` specifies the cluster size. This option or the `n()` option is required to compute the number of clusters, power, or effect size. `m()` may contain noninteger values. In this case or if the `cvcluster()` option is specified, `m()` represents the average cluster size.

`n(`*numlist*`)` specifies the number of observations. This option or the `m()` option is required to compute the number of clusters, power, or effect size.

`nfractional`; see [PSS-2] **power**. The `nfractional` option is allowed when computing the number of clusters and cluster size to display fractional (without rounding) values of the number of clusters, cluster size, and sample size.

`diff(`*numlist*`)` specifies the difference between the alternative proportion and the null proportion, $p_a - p_0$. You can specify either the alternative proportion p_a as a command argument or the difference between the two proportions in `diff()`. If you specify `diff(#)`, the alternative proportion is computed as $p_a = p_0 + \#$. This option is not allowed with the effect-size determination.

`rho(`*numlist*`)` specifies the intraclass correlation. The default is `rho(0.5)`.

`cvcluster(`*numlist*`)` specifies the coefficient of variation for cluster sizes. This option is used with varying cluster sizes.

`direction()`, `onesided`, `parallel`; see [PSS-2] **power**.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies the initial value for the number of clusters or cluster size for sample-size determination or the initial value for the proportion for the effect-size determination. The default is to use a closed-form normal approximation to compute an initial value for the estimated parameter.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following option is available with `power oneproportion, cluster` but is not shown in the dialog box:

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

`power oneproportion, cluster` requests that computations for the `power oneproportion` command be done for a CRD. In a CRD, groups of subjects or clusters are randomized instead of individual subjects, so the sample size is determined by the number of clusters and the cluster size. The sample-size determination thus consists of the determination of the number of clusters given cluster size or the determination of cluster size given the number of clusters. For a general discussion of using `power oneproportion`, see [PSS-2] **power oneproportion**. The discussion below is specific to the CRD.

Using power oneproportion, cluster

If you specify the `cluster` option, include `k()` to specify the number of clusters or include `m()` to specify the cluster size, the `power oneproportion` command will perform computations for a one-sample proportion test in a CRD. The computations for a CRD are based on the large-sample Wald z test.

All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option. You can specify the `onesided` option to request a one-sided test.

To compute the number of clusters, you must specify the proportions under the null and alternative hypotheses as command arguments p_0 and p_a, respectively, and specify the cluster size in the `m()` option. Instead of specifying the `m()` option, you may specify the sample size in the `n()` option and specify the `cluster` option, so that `power onemean` will perform its computation for a cluster randomized design instead of the default individual-level design. You may also specify the power of the test in the `power()` option.

To compute cluster size, you must specify the null proportion p_0, the alternative proportion p_a, and the number of clusters in the `k()` option. You may also specify the power of the test in the `power()` option.

To compute power, you must specify the number of clusters in the `k()` option, the cluster size in the `m()` option or the sample size in the `n()` option, the null proportion p_0, and the alternative proportion p_a.

Instead of the alternative proportion p_a, you may specify the difference $p_a - p_0$ between the alternative proportion and the null proportion in the `diff()` option when computing sample size or power.

The effect size δ is defined as the difference between the alternative and null proportions. In a CRD, the effect size δ is also adjusted for the cluster design; see *Methods and formulas*.

To compute effect size and the corresponding target proportion, you must specify the number of clusters in the `k()` option, the cluster size in the `m()` option or the sample size in the `n()` option, the power in the `power()` option, and the null proportion p_0. You may also specify the direction of the effect in the `direction()` option. The direction is upper by default, `direction(upper)`; see *Using power oneproportion* in [PSS-2] **power oneproportion** for other details.

All computations assume an intraclass correlation of 0.5. You can change this by specifying the `rho()` option. Also, all clusters are assumed to be of the same size unless the coefficient of variation for cluster sizes is specified in the `cvcluster()` option.

By default, the computed number of clusters, cluster size, and sample size is rounded up. However, you can specify the `nfractional` option to see the corresponding fractional values; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. If the `cvcluster()` option is specified when computing cluster size, then cluster size represents the average cluster size and is thus not rounded. When sample size is specified in the `n()` option, fractional cluster size may be reported to accommodate the specified number of clusters and sample size.

Some of `power oneproportion, cluster`'s computations require iteration, such as to compute the number of clusters for a two-sided test; see *Methods and formulas* for details and [PSS-2] **power** for the descriptions of options that control the iteration procedure.

Computing number of clusters

To compute the number of clusters, you must specify the proportions under the null and alternative hypotheses as command arguments p_0 and p_a, respectively, and specify the cluster size in the `m()` option. Instead of specifying the `m()` option, you may specify the sample size in the `n()` option and specify the `cluster` option, so that `power onemean` will perform its computation for a cluster randomized design instead of the default individual-level design. You may also specify the power of the test in the `power()` option.

Example 1: Number of clusters for a one-sample proportion test in a CRD, specifying cluster size

Ahn, Heo, and Zhang (2015, 33) demonstrate sample-size computations for a clustered binary outcome by using the data from Hujoel, Moulton, and Loesche (1990) as pilot data. The data recorded positive test results from an enzymatic diagnostic test (EDT) of a specific (target) infection. There were 29 subjects in the study, and each subject had multiple infected sites, as determined by a gold standard test, which were then retested for the presence of the target infection using the EDT. The number of infected sites varied among subjects with an average of 4.897 sites, and observations within a subject were correlated with an intraclass correlation of 0.2. Ahn, Heo, and Zhang (2015) used these estimates to compute the required number of clusters for a new study to test whether the proportion of infected sites detected by the EDT is 0.6, H_0: $p = 0.6$, against the alternative H_a: $p = 0.7$. We demonstrate how to use `power oneproportion, cluster` to compute the required number of clusters.

For simplicity, we assume a constant cluster size across subjects and use an integer cluster size of 5. To detect a proportion of 0.7 against the reference value of 0.6 with 80% power using a 5%-level two-sided test, we type

```
. power oneproportion 0.6 0.7, m(5) rho(0.2)
Performing iteration ...
Estimated number of clusters for a one-sample proportion test
Cluster randomized design, Wald z test
H0: p = p0  versus  Ha: p != p0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.1000
           p0 =    0.6000
           pa =    0.7000
Cluster design:
            M =         5
          rho =    0.2000
Estimated number of clusters and sample size:
            K =        60
            N =       300
```

We find that given 5 sites per subject, 60 subjects and thus a total of 300 infected sites are required to detect a proportion of 0.7 for the infection of interest against a reference proportion of 0.6 with 80% power using a 5%-level two-sided test. The effect size (`delta`) is calculated as the difference between the alternative and null proportions.

▷ Example 2: Number of clusters for a one-sample proportion test in a CRD, with varying cluster sizes

Unlike the simplified case in example 1, in a practical study, the number of infected sites per subject may vary. We use the average number of infected sites of 4.897 and a coefficient of variation of 0.25. To account for varying cluster sizes, we specify `m(4.897)` and `cvcluster(0.25)`.

```
. power oneproportion 0.6 0.7, m(4.897) rho(0.2) cvcluster(0.25)
Performing iteration ...
Estimated number of clusters for a one-sample proportion test
Cluster randomized design, Wald z test
H0: p = p0  versus  Ha: p != p0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.1000
           p0 =    0.6000
           pa =    0.7000
Cluster design:
    Average M =    4.8970
          rho =    0.2000
       CV_cl =    0.2500
Estimated number of clusters and sample size:
            K =        61
            N =       299
```

We now need 61 subjects for a total of 299 sites to achieve the same power.

◁

Computing cluster size

To compute cluster size, you must specify the null proportion p_0, the alternative proportion p_a, and the number of clusters in the `k()` option. You may also specify the power of the test in the `power()` option.

▷ Example 3: Cluster size for a one-sample proportion test in a CRD

Continuing with example 1, suppose that we are designing a new study and would like to recruit 80 subjects in the study. We would like to get an idea of how many infected sites we need to achieve 80% power. Given the study parameters from example 1, we compute the number of infected sites by specifying 80 clusters in the `k()` option.

```
. power oneproportion 0.6 0.7, k(80) rho(0.2)

Performing iteration ...

Estimated cluster size for a one-sample proportion test
Cluster randomized design, Wald z test
H0: p = p0  versus  Ha: p != p0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.1000
           p0 =    0.6000
           pa =    0.7000

Cluster design:

            K =        80
          rho =    0.2000

Estimated cluster size and sample size:

            M =         3
            N =       240
```

To achieve the desired power with 80 subjects, we will need to observe 3 sites per subject.

◁

Computing power

To compute power, you must specify the number of clusters in the `k()` option, the cluster size in the `m()` option or the sample size in the `n()` option, the null proportion p_0, and the alternative proportion p_a.

▷ Example 4: Power for a one-sample proportion test in a CRD

Continuing with example 1, suppose that we have 80 subjects and each subject has 5 infected sites. Given the study parameters from example 1, we compute the power by specifying 80 clusters in the `k()` option and cluster size of 5 in the `m()` option:

```
. power oneproportion 0.6 0.7, k(80) m(5) rho(0.2)

Estimated power for a one-sample proportion test
Cluster randomized design, Wald z test
H0: p = p0  versus  Ha: p != p0

Study parameters:

        alpha =    0.0500
        delta =    0.1000
           p0 =    0.6000
           pa =    0.7000

Cluster design:

            K =        80
            M =         5
            N =       400
          rho =    0.2000

Estimated power:

        power =    0.9020
```

The computed power is about 90%.

◁

▷ Example 5: Multiple values of study parameters

To investigate the effect of the number of clusters on power, we can specify a list of numbers in the `k()` option:

```
. power oneproportion 0.6 0.7, k(20(20)100) m(5) rho(0.2)
Estimated power for a one-sample proportion test
Cluster randomized design, Wald z test
H0: p = p0  versus  Ha: p != p0
```

alpha	power	K	M	N	delta	p0	pa	rho
.05	.3696	20	5	100	.1	.6	.7	.2
.05	.6332	40	5	200	.1	.6	.7	.2
.05	.8043	60	5	300	.1	.6	.7	.2
.05	.902	80	5	400	.1	.6	.7	.2
.05	.9532	100	5	500	.1	.6	.7	.2

As expected, as the number of clusters increases, the power tends to get closer to 1.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

Computing effect size and target proportion

The effect size δ is defined as the difference between the alternative and null proportions. In a CRD, the effect size δ is also adjusted for the cluster design; see *Methods and formulas*.

To compute effect size and the corresponding target proportion, you must specify the number of clusters in the `k()` option, the cluster size in the `m()` option or the sample size in the `n()` option, the power in the `power()` option, and the null proportion p_0. You may also specify the direction of the effect in the `direction()` option. The direction is upper by default, `direction(upper)`; see *Using power oneproportion* in [PSS-2] **power oneproportion** for other details.

▷ Example 6: Effect size for a one-sample proportion test in a CRD

Continuing with example 4, we may also be interested in finding the minimum value of the proportion that can be detected with a sample of 80 subjects, 5 infected sites per subject, and 80% power. To compute this, we specify the null value of 0.6 as the command argument and the required options `k(80)`, `m(5)`, and `power(0.8)` and continue to use `rho(0.2)`.

```
. power oneproportion 0.6, k(80) m(5) power(0.8) rho(0.2)
Performing iteration ...
Estimated target proportion for a one-sample proportion test
Cluster randomized design, Wald z test
H0: p = p0  versus  Ha: p != p0; pa > p0
Study parameters:
        alpha =    0.0500
        power =    0.8000
           p0 =    0.6000
Cluster design:
            K =        80
            M =         5
            N =       400
          rho =    0.2000
Estimated effect size and target proportion:
        delta =    0.0871
           pa =    0.6871
```

Given the null value of 0.6, the minimum detectable value of the proportion is about 0.69, which is slightly smaller than the alternative proportion of 0.7 used in previous examples, because here we use more subjects than, for instance, in example 1, more sites per subject than in example 3, and lower power than in example 4.

◁

Performing hypothesis tests on proportion in a CRD

`power oneproportion, cluster` performs PSS computations based on a large-sample test of proportion that accounts for a CRD or for clustered data. We can perform this test by using `prtest, cluster()`; see [R] **prtest**. In this section, we briefly demonstrate how to test the hypothesis that the proportion is different from a reference value on the collected clustered data by using `prtest`.

▷ Example 7: Testing for proportion with clustered data

Ahn, Heo, and Zhang (2015, 33) report the data from Hujoel, Moulton, and Loesche (1990) on positive test results from the EDT; see example 1 for details about the study. Let's use `prtest` to test the null hypothesis H_0: $p = 0.6$.

For clustered data, `prtest` requires that we specify the cluster identifier in the `cluster()` option and population intraclass correlation in the `rho()` option. We use the intraclass correlation of 0.2 as in Ahn, Heo, and Zhang (2015, 33).

```
. use https://www.stata-press.com/data/r19/infection
(Target infections detected by EDT (Hujoel, Moulton, and Loesche 1990))
. prtest infection == 0.6, cluster(subject) rho(0.2)
One-sample test of proportion                  Number of obs      =       142
Cluster variable: subject                      Number of clusters =        29
                                               Avg. cluster size  =      4.90
                                               CV cluster size    =    0.2419
                                               Intraclass corr.   =    0.2000

------------------------------------------------------------------------------
    Variable |       Mean   Std. err.                      [95% conf. interval]
-------------+----------------------------------------------------------------
   infection |   .6619718   .0537974                      .5565308    .7674129
------------------------------------------------------------------------------

    p = proportion(infection)                                     z =   1.1123
H0: p = 0.6
     Ha: p < 0.6                 Ha: p != 0.6                 Ha: p > 0.6
 Pr(Z < z) = 0.8670         Pr(|Z| > |z|) = 0.2660          Pr(Z > z) = 0.1330
```

We do not find any statistical evidence to reject the null hypothesis of H_0: $p = 0.6$.

Suppose that we want to design a new similar study and use the estimates from this study to compute the required number of clusters. We are interested in detecting the alternative value of, say, 0.66 with 80% power for a 5%-level two-sided test. To compute the required number of clusters, we use the average cluster size of 4.9 as observed in this study.

```
. power oneproportion 0.6 0.66, m(4.9) rho(0.2)
Performing iteration ...
Estimated number of clusters for a one-sample proportion test
Cluster randomized design, Wald z test
H0: p = p0  versus  Ha: p != p0
Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.0600
           p0 =    0.6000
           pa =    0.6600

Cluster design:

    Average M =    4.9000
          rho =    0.2000

Estimated number of clusters and sample size:

            K =       178
            N =       873
```

We need 178 subjects to detect the 0.06 difference between the alternative and null proportions, given the null proportion of 0.6, with 80% power using a 5%-level two-sided test.

◁

Stored results

power oneproportion, cluster stores the following in r():

Scalars

r(alpha)	significance level
r(power)	power
r(beta)	probability of a type II error
r(delta)	effect size
r(K)	number of clusters
r(M)	cluster size
r(N)	number of subjects
r(nfractional)	1 if nfractional is specified, 0 otherwise
r(onesided)	1 for a one-sided test, 0 otherwise
r(p0)	proportion under the null hypothesis
r(pa)	proportion under the alternative hypothesis
r(diff)	difference between the alternative and null proportions
r(rho)	intraclass correlation
r(CV_cluster)	coefficient of variation for cluster sizes
r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise
r(init)	initial value for estimated parameter
r(maxiter)	maximum number of iterations
r(iter)	number of iterations performed
r(tolerance)	requested parameter tolerance
r(deltax)	final parameter tolerance achieved
r(ftolerance)	requested distance of the objective function from zero
r(function)	final distance of the objective function from zero
r(converged)	1 if iteration algorithm converged, 0 otherwise

Macros

r(type)	test
r(method)	oneproportion
r(design)	CRD
r(test)	wald
r(direction)	upper or lower
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results

Methods and formulas

The computation for a CRD is based on the Wald test under the large-sample normal approximation, adjusted for the cluster design; see *Large-sample normal approximation* under *Methods and formulas* in [PSS-2] **power oneproportion** for the common notation for a one-sample proportion test.

Methods and formulas are presented under the following headings:

Equal cluster sizes
Unequal cluster sizes

Equal cluster sizes

In a CRD, let K be the number of clusters, M be the number of observations in each cluster, and n be the total number of subjects, where $n = MK$. Let x_{ij} be the outcome of a Bernoulli trial of the jth ($j = 1, 2, \ldots, M$) observation from the ith cluster ($i = 1, 2, \ldots, K$). Let ρ be the intraclass correlation and DE be the design effect defined as

$$\text{DE} = 1 + \rho(M - 1)$$

Let $P(x_{ij} = 1) = p$ denote the probability of a success in the population. Each individual observation is a Bernoulli trial with a success probability p. Let

$$\hat{p} = \frac{1}{n}\sum_{i=1}^{K}\sum_{j=1}^{M} x_{ij} \qquad \text{and} \qquad \text{se}(\hat{p}) = \sqrt{\frac{\hat{p}(1-\hat{p})\text{DE}}{n}}$$

denote the sample proportion and its standard error, respectively. Let p_0 and p_a denote the respective null and alternative values of the proportion parameters.

For a large sample, the distribution of the sample proportion $\hat{p}$ may be approximated by the normal distribution with proportion p and variance $p(1-p)\text{DE}/n$. The Wald test statistic $z = (\hat{p} - p_0)/\sqrt{\hat{p}(1-\hat{p})\text{DE}/n}$ under the null hypothesis follows a standard normal distribution; see, for example, Ahn, Heo, and Zhang (2015).

Let α be the significance level, β be the probability of a type II error, and $z_{1-\alpha}$ and z_β be the $(1-\alpha)$th and the βth quantiles of the standard normal distribution. Let

$$p_{\text{std}} = \frac{(p_a - p_0)}{\sqrt{p_a(1-p_a)\text{DE}}} \tag{1}$$

The power $\pi = 1 - \beta$ is computed using

$$\pi = \begin{cases} \Phi\left(\sqrt{n}p_{\text{std}} - z_{1-\alpha}\right) & \text{for an upper one-sided test} \\ \Phi\left(-\sqrt{n}p_{\text{std}} - z_{1-\alpha}\right) & \text{for a lower one-sided test} \\ \Phi\left(\sqrt{n}p_{\text{std}} - z_{1-\alpha/2}\right) + \Phi\left(-\sqrt{n}p_{\text{std}} - z_{1-\alpha/2}\right) & \text{for a two-sided test} \end{cases} \tag{2}$$

where $\Phi(\cdot)$ is the c.d.f. of the standard normal distribution.

Given the cluster size M, the number of clusters K for a one-sided test is computed by inverting a one-sided power equation from (2),

$$K = \left(\frac{z_{1-\alpha} - z_\beta}{p_{\text{std}}\sqrt{M}}\right)^2 \tag{3}$$

Given the sample size n, the number of clusters K for a one-sided test is computed as

$$K = \frac{n\left(p_a - p_0\right)^2}{\rho p_a(1-p_a)\left(z_{1-\alpha} - z_\beta\right)^2} - \frac{1}{\rho} + 1 \tag{4}$$

Given the number of clusters K, the cluster size M for a one-sided test is computed by solving (2), after substituting p_{std} from (1),

$$M = \frac{1-\rho}{\frac{K(p_a-p_0)^2}{p_a(1-p_a)(z_{1-\alpha}-z_\beta)^2} - \rho} \tag{5}$$

The number of clusters and cluster size for a two-sided test are computed iteratively using the two-sided power equation from (2). The initial values are obtained from (3), (4), and (5), with $\alpha/2$.

The minimum detectable value of the proportion is computed iteratively using the corresponding power equation from (2).

Unequal cluster sizes

For unequal cluster sizes, we assume that the cluster sizes are independent and identically distributed and are small relative to the number of clusters; see Ahn, Heo, and Zhang (2015) for details. Let the coefficient of variation of the cluster sizes be CV_{cl}. According to van Breukelen, Candel, and Berger (2007) and Campbell and Walters (2014), to adjust for varying cluster sizes, define the relative efficiency (RE) of unequal versus equal cluster sizes as

$$\text{RE} = 1 - \lambda(1-\lambda)\text{CV}_{\text{cl}}^2$$

where $\lambda = \rho M/(\rho M + 1 - \rho)$. With unequal cluster sizes, p_{std} becomes

$$p_{\text{std}} = \frac{(p_a - p_0)}{\sqrt{p_a(1-p_a)\text{DE}/\text{RE}}} \tag{6}$$

With p_{std} as defined in (6), we can obtain the formula for computing the number of clusters given cluster size for a one-sided test using (3). In all other cases, parameters are computed iteratively using the power equations in (2) with p_{std} from (6).

References

Ahn, C., M. Heo, and S. Zhang. 2015. *Sample Size Calculations for Clustered and Longitudinal Outcomes in Clinical Research.* Boca Raton, FL: CRC Press. https://doi.org/10.1201/b17822.

Campbell, M. J., and S. J. Walters. 2014. *How to Design, Analyse and Report Cluster Randomised Trials in Medicine and Health Related Research.* Chichester, UK: Wiley. https://doi.org/10.1002/9781118763452.

Gallis, J. A., F. Li, H. Yu, and E. L. Turner. 2018. cvcrand and cptest: Commands for efficient design and analysis of cluster randomized trials using constrained randomization and permutation tests. *Stata Journal* 18: 357–378.

Hujoel, P. P., L. H. Moulton, and W. J. Loesche. 1990. Estimation of sensitivity and specificity of site-specific diagnostic tests. *Journal of Periodontal Research* 25: 193–196. https://doi.org/10.1111/j.1600-0765.1990.tb00903.x.

van Breukelen, G. J. P., M. J. J. M. Candel, and M. P. F. Berger. 2007. Relative efficiency of unequal versus equal cluster sizes in cluster randomized and multicentre trials. *Statistics in Medicine* 26: 2589–2603. https://doi.org/10.1002/sim.2740.

Also see

[PSS-2] **power oneproportion** — Power analysis for a one-sample proportion test

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[R] **prtest** — Tests of proportions

power twoproportions — Power analysis for a two-sample proportions test

Description

`power twoproportions` computes sample size, power, or the experimental-group proportion for a two-sample proportions test. By default, it computes sample size for given power and the values of the control-group and experimental-group proportions. Alternatively, it can compute power for given sample size and values of the control-group and experimental-group proportions or the experimental-group proportion for given sample size, power, and the control-group proportion. For power and sample-size analysis in a cluster randomized design, see [PSS-2] **power twoproportions, cluster**. Also see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Quick start

Sample size for a test of $H_0\colon \pi_1 = \pi_2$ versus $H_a\colon \pi_1 \neq \pi_2$ given alternative control-group proportion $p_1 = 0.8$, alternative experimental-group proportion $p_2 = 0.65$ with default power of 0.8 and significance level $\alpha = 0.05$

```
power twoproportions .8 .65
```

Same as above, but specified as p_1 and difference between proportions $p_2 - p_1 = -0.15$

```
power twoproportions .8, diff(-.15)
```

Same as above, but specified as $p_1 = 0.8$ and ratio $p_2/p_1 = 0.8125$

```
power twoproportions .8, ratio(.8125)
```

Same as above, but specified as $p_1 = 0.8$ and odds ratio $\{p_2/(1-p_2)\}/\{p_1/(1-p_1)\} = 0.464$

```
power twoproportions .8, oratio(.464)
```

Power for sample sizes of 50 and 80 in groups 1 and 2, respectively

```
power twoproportions 0.8 0.65, n1(50) n2(80)
```

Power for total sample sizes of 150, 170, 190, 210, and 230

```
power twoproportions .8 .65, n(150(20)230)
```

Same as above, but display results in a graph of power versus sample size

```
power twoproportions .8 .65, n(150(20)230) graph
```

Same as above, but with difference equal to 0.1, 0.15, and 0.2

```
power twoproportions .65, n(150(20)200) graph diff(.1(.05).2)
```

Display results in a table showing total sample size, difference, and power

```
power twoproportions .65, n(150(20)200) table(N diff power)  ///
  diff(.1(.05).2)
```

Effect size and target p_2 for $p_1 = 0.6$ with sample size of 200, power of 0.8, and $\alpha = 0.01$

```
power twoproportions .6, n(200) power(.8) alpha(.01)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
power twoproportions p1 p2 [ , power(numlist) options ]
```

Compute power

```
power twoproportions p1 p2 , n(numlist) [ options ]
```

Compute effect size and experimental-group proportion

```
power twoproportions p1 , n(numlist) power(numlist) [ options ]
```

where p_1 is the proportion in the control (reference) group and p_2 is the proportion in the experimental (comparison) group. p_1 and p_2 may each be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

options	Description
`test(`*test*`)`	specify the type of test; default is `test(chi2)`
Main	
* `alpha(`*numlist*`)`	significance level; default is `alpha(0.05)`
* `power(`*numlist*`)`	power; default is `power(0.8)`
* `beta(`*numlist*`)`	probability of type II error; default is `beta(0.2)`
* `n(`*numlist*`)`	total sample size; required to compute power or effect size
* `n1(`*numlist*`)`	sample size of the control group
* `n2(`*numlist*`)`	sample size of the experimental group
* `nratio(`*numlist*`)`	ratio of sample sizes, `N2/N1`; default is `nratio(1)`, meaning equal group sizes
`compute(N1 | N2)`	solve for `N1` given `N2` or for `N2` given `N1`
`nfractional`	allow fractional sample sizes
* `diff(`*numlist*`)`	difference between the experimental-group and control-group proportions, $p_2 - p_1$; specify instead of the experimental-group proportion p_2
* `ratio(`*numlist*`)`	ratio of the experimental-group proportion to the control-group proportion, p_2/p_1; specify instead of the experimental-group proportion p_2
* `rdiff(`*numlist*`)`	risk difference, $p_2 - p_1$; synonym for `diff()`
* `rrisk(`*numlist*`)`	relative risk, p_2/p_1; synonym for `ratio()`
* `oratio(`*numlist*`)`	odds ratio, $\{p_2(1-p_1)\}/\{p_1(1-p_2)\}$
`effect(`*effect*`)`	specify the type of effect to display; default is `effect(diff)`
`continuity`	apply continuity correction to the normal approximation of the discrete distribution
`direction(upper|lower)`	direction of the effect for effect-size determination; default is `direction(upper)`, which means that the postulated value of the parameter is larger than the hypothesized value
`onesided`	one-sided test; default is two sided
`parallel`	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[`no`]`table`[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
`saving(`*filename* [`, replace`]`)`	save the table data to *filename*; use `replace` to overwrite existing *filename*
Graph	
`graph`[(*graphopts*)]	graph results; see [PSS-2] **power, graph**

Iteration

`init(#)`	initial value for sample sizes or experimental-group proportion
`iterate(#)`	maximum number of iterations; default is `iterate(500)`
`tolerance(#)`	parameter tolerance; default is `tolerance(1e-12)`
`ftolerance(#)`	function tolerance; default is `ftolerance(1e-12)`
`[no]log`	suppress or display iteration log
`[no]dots`	suppress or display iterations as dots
`cluster`	perform computations for a CRD; see [PSS-2] **power twoproportions, cluster**
`notitle`	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the `parallel` option.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`cluster` and `notitle` do not appear in the dialog box.

test	Description
`chi2`	Pearson's χ^2 test; the default
`lrchi2`	likelihood-ratio test
`fisher`	Fisher–Irwin's exact conditional test

`test()` does not appear in the dialog box. The dialog box selected is determined by the `test()` specification.

effect	Description
`diff`	difference between proportions, $p_2 - p_1$; the default
`ratio`	ratio of proportions, p_2/p_1
`rdiff`	risk difference, $p_2 - p_1$
`rrisk`	relative risk, p_2/p_1
`oratio`	odds ratio, $\{p_2(1-p_1)\}/\{p_1(1-p_2)\}$

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
alpha	significance level	α
alpha_a	observed significance level	α_a
power	power	$1-\beta$
beta	type-II-error probability	β
N	total number of subjects	N
N1	number of subjects in the control group	N_1
N2	number of subjects in the experimental group	N_2
nratio	ratio of sample sizes, experimental to control	N_2/N_1
delta	effect size	δ
p1	control-group proportion	p_1
p2	experimental-group proportion	p_2
diff	difference between the experimental-group proportion and the control-group proportion	p_2-p_1
ratio	ratio of the experimental-group proportion to the control-group proportion	p_2/p_1
rdiff	risk difference	p_2-p_1
rrisk	relative risk	p_2/p_1
oratio	odds ratio	θ
target	target parameter; synonym for p2	
_all	display all supported columns	

Column beta is shown in the default table in place of column power if specified.

Column alpha_a is available when the test(fisher) option is specified.

Columns nratio, diff, ratio, rdiff, rrisk, and oratio are shown in the default table if specified.

Options

test(*test*) specifies the type of the test for power and sample-size computations. *test* is one of chi2, lrchi2, or fisher.

chi2 requests computations for the Pearson's χ^2 test. This is the default test.

lrchi2 requests computations for the likelihood-ratio test.

fisher requests computations for Fisher–Irwin's exact conditional test. Iteration options are not allowed with this test.

Main

alpha(), power(), beta(), n(), n1(), n2(), nratio(), compute(), nfractional; see [PSS-2] **power**.

diff(*numlist*) specifies the difference between the experimental-group proportion and the control-group proportion, $p_2 - p_1$. You can specify either the experimental-group proportion p_2 as a command argument or the difference between the two proportions in diff(). If you specify diff(#), the experimental-group proportion is computed as $p_2 = p_1 + \#$. This option is not allowed with the effect-size determination and may not be combined with ratio(), rdiff(), rrisk(), or oratio().

`ratio(`*numlist*`)` specifies the ratio of the experimental-group proportion to the control-group proportion, p_2/p_1. You can specify either the experimental-group proportion p_2 as a command argument or the ratio of the two proportions in `ratio()`. If you specify `ratio(#)`, the experimental-group proportion is computed as $p_2 = p_1 \times \#$. This option is not allowed with the effect-size determination and may not be combined with `diff()`, `rdiff()`, `rrisk()`, or `oratio()`.

`rdiff(`*numlist*`)` specifies the risk difference $p_2 - p_1$. This is a synonym for the `diff()` option, except the results are labeled as risk differences. This option is not allowed with the effect-size determination and may not be combined with `diff()`, `ratio()`, `rrisk()`, or `oratio()`.

`rrisk(`*numlist*`)` specifies the relative risk or risk ratio $p_2 - p_1$. This is a synonym for the `ratio()` option, except the results are labeled as relative risks. This option is not allowed with the effect-size determination and may not be combined with `diff()`, `ratio()`, `rdiff()`, or `oratio()`.

`oratio(`*numlist*`)` specifies the odds ratio $\{p_2(1-p_1)\}/\{p_1(1-p_2)\}$. You can specify either the experimental-group proportion p_2 as a command argument or the odds ratio in `oratio()`. If you specify `oratio(#)`, the experimental-group proportion is computed as $p_2 = 1/\{1 + (1-p_1)/(p_1 \times \#)\}$. This option is not allowed with the effect-size determination and may not be combined with `diff()`, `ratio()`, `rdiff()`, or `rrisk()`.

`effect(`*effect*`)` specifies the type of the effect size to be reported in the output as `delta`. *effect* is one of `diff`, `ratio`, `rdiff`, `rrisk`, or `oratio`. By default, the effect size `delta` is the difference between proportions. If `diff()`, `ratio()`, `rdiff()`, `rrisk()`, or `oratio()` is specified, the effect size `delta` will contain the effect corresponding to the specified option. For example, if `oratio()` is specified, `delta` will contain the odds ratio.

`continuity` requests that continuity correction be applied to the normal approximation of the discrete distribution. `continuity` cannot be specified with `test(fisher)` or `test(lrchi2)`.

`direction()`, `onesided`, `parallel`; see [PSS-2] **power**.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies the initial value for the estimated parameter. For sample-size determination, the estimated parameter is either the control-group size n_1 or, if `compute(N2)` is specified, the experimental-group size n_2. For the effect-size determination, the estimated parameter is the experimental-group proportion p_2. The default initial values for sample sizes for a two-sided test are based on the corresponding one-sided large-sample z test with the significance level $\alpha/2$. The default initial value for the experimental-group proportion is computed using the bisection method.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following options are available with `power twoproportions` but are not shown in the dialog box:

`cluster`; see [PSS-2] **power twoproportions, cluster**.

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `power twoproportions` command and the methodology for power and sample-size analysis for a two-sample proportions test. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis and [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests. Also see [PSS-2] **power twoproportions, cluster** for power and sample-size analysis in a cluster randomized design.

Introduction

The comparison of two independent proportions arises in studies involving two independent binomial populations. There are many examples of studies where a researcher would like to compare two independent proportions. A pediatrician might be interested in the relationship between low birthweight and the mothers' use of a particular drug during pregnancy. He or she would like to test the null hypothesis that there is no difference in the proportion of low-birthweight babies for mothers who took the drug and mothers who did not. A drug manufacturer may want to test the developed new topical treatment for a foot fungus by testing the null hypothesis that the proportion of successfully treated patients is the same in the treatment and placebo groups.

Hypothesis testing of binomial outcomes relies on a set of assumptions: 1) a Bernoulli outcome is observed a fixed number of times; 2) the probability p of observing an event of interest in one trial is fixed across all trials; and 3) individual trials are independent. Each of the two populations must conform to the assumptions of a binomial distribution.

This entry describes power and sample-size analysis for the inference about two population proportions performed using hypothesis testing. Specifically, we consider the null hypothesis H_0: $p_2 = p_1$ versus the two-sided alternative hypothesis H_a: $p_2 \neq p_1$, the upper one-sided alternative H_a: $p_2 > p_1$, or the lower one-sided alternative H_a: $p_2 < p_1$.

The large-sample Pearson's χ^2 and likelihood-ratio tests are commonly used to test hypotheses about two independent proportions. The test of Fisher (1935) and Irwin (1935) is commonly used to compare the two proportions in small samples.

The `power twoproportions` command provides power and sample-size analysis for these three tests. For Fisher's exact test, the direct computation is available only for the power of the test. Estimates of the sample size and effect size for Fisher's exact test are difficult to compute directly because of the discrete nature of the sampling distribution of the test statistic. They can, however, be obtained indirectly on the basis of the power computation; see example 8 for details.

Using power twoproportions

power twoproportions computes sample size, power, or experimental-group proportion for a two-sample proportions test. All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the alpha() option. You can specify the onesided option to request a one-sided test. By default, all computations assume a balanced or equal-allocation design; see [PSS-4] **Unbalanced designs** for a description of how to specify an unbalanced design.

power twoproportions performs power analysis for three different tests, which can be specified within the test() option. The default is Pearson's χ^2 test (test(chi2)), which approximates the sampling distribution of the test statistic by the standard normal distribution. You may instead request computations based on the likelihood-ratio test by specifying the test(lrchi2) option. To request Fisher's exact conditional test based on the hypergeometric distribution, you can specify test(fisher). The fisher method is not available for computing sample size or effect size; see example 8 for details.

To compute the total sample size, you must specify the control-group proportion p_1, the experimental-group proportion p_2, and, optionally, the power of the test in the power() option. The default power is set to 0.8.

Instead of the total sample size, you can compute one of the group sizes given the other one. To compute the control-group sample size, you must specify the compute(N1) option and the sample size of the experimental group in the n2() option. Likewise, to compute the experimental-group sample size, you must specify the compute(N2) option and the sample size of the control group in the n1() option.

To compute power, you must specify the total sample size in the n() option, the control-group proportion p_1, and the experimental-group proportion p_2.

Instead of the experimental-group proportion p_2, you can specify other alternative measures of effect when computing sample size or power; see *Alternative ways of specifying effect* below.

To compute effect size and the experimental-group proportion, you must specify the total sample size in the n() option, the power in the power() option, the control-group proportion p_1, and optionally, the direction of the effect. The direction is upper by default, direction(upper), which means that the experimental-group proportion is assumed to be larger than the specified control-group value. You can change the direction to be lower, which means that the experimental-group proportion is assumed to be smaller than the specified control-group value, by specifying the direction(lower) option.

There are multiple definitions of the effect size for a two-sample proportions test. The effect() option specifies what definition power twoproportions should use when reporting the effect size, which is labeled as delta in the output of the power command. The available definitions are the difference between the experimental-group proportion and the control-group proportion (diff), the ratio of the experimental-group proportion to the control-group proportion (ratio), the risk difference $p_2 - p_1$ (rdiff), the relative risk p_2/p_1 (rrisk), and the odds ratio $\{p_2(1-p_1)\}/\{p_1(1-p_2)\}$ (oratio). When effect() is specified, the effect size delta contains the estimate of the corresponding effect and is labeled accordingly. By default, delta corresponds to the difference between proportions. If any of the options diff(), ratio(), rdiff(), rrisk(), or oratio() are specified and effect() is not specified, delta will contain the effect size corresponding to the specified option.

Instead of the total sample size n(), you can specify individual group sizes in n1() and n2(), or specify one of the group sizes and nratio() when computing power or effect size. Also see *Two samples* in [PSS-4] **Unbalanced designs** for more details.

Alternative ways of specifying effect

As we mentioned above, `power twoproportions` provides a number of ways to specify the disparity between the control-group and experimental-group proportions for sample-size and power determinations.

You can specify the control-group proportion p_1 and the experimental-group proportion p_2 directly, after the command name:

`power twoproportions` p_1 p_2 `, ...`

For this specification, the default effect size `delta` displayed by the `power` command is the difference $p_2 - p_1$ between the proportions. You can use the `effect()` option to request another type of effect. For example, if you specify `effect(oratio)`,

`power twoproportions` p_1 p_2 `, effect(oratio) ...`

the effect size `delta` will correspond to the odds ratio.

Alternatively, you can specify the control-group proportion p_1 and one of the options `diff()`, `ratio()`, `rdiff()`, `rrisk()`, or `oratio()`. For these specifications, the effect size `delta` will contain the effect corresponding to the option. If desired, you can change this by specifying the `effect()` option.

Specify difference $p_2 - p_1$ between the two proportions:

`power twoproportions` p_1 `, diff(`*numlist*`) ...`

Specify risk difference $p_2 - p_1$:

`power twoproportions` p_1 `, rdiff(`*numlist*`) ...`

Specify ratio p_2/p_1 of the two proportions:

`power twoproportions` p_1 `, ratio(`*numlist*`) ...`

Specify relative risk or risk ratio p_2/p_1:

`power twoproportions` p_1 `, rrisk(`*numlist*`) ...`

Specify odds ratio $\{p_2(1-p_1)\}/\{p_1(1-p_2)\}$:

`power twoproportions` p_1 `, oratio(`*numlist*`) ...`

In the following sections, we describe the use of `power twoproportions` accompanied by examples for computing sample size, power, and experimental-group proportions.

Computing sample size

To compute sample size, you must specify the control-group proportion p_1, the experimental-group proportion p_2, and, optionally, the power of the test in the `power()` option. A default power of 0.8 is assumed if `power()` is not specified.

▷ Example 1: Sample size for a two-sample proportions test

Consider a study investigating the effectiveness of aspirin in reducing the mortality rate due to myocardial infarction (heart attacks). Let p_A denote the proportion of deaths for aspirin users in the population and p_N denote the corresponding proportion for nonusers. We are interested in testing the null hypothesis H_0: $p_A - p_N = 0$ against the two-sided alternative hypothesis H_a: $p_A - p_N \neq 0$.

Previous studies indicate that the proportion of deaths due to heart attacks is 0.015 for nonusers and 0.001 for users. Investigators wish to determine the minimum sample size required to detect an absolute difference of $|0.001 - 0.015| = 0.014$ with 80% power using a two-sided 5%-level test.

To compute the required sample size, we specify the values 0.015 and 0.001 as the control- and experimental-group proportions after the command name. We omit options `alpha(0.05)` and `power(0.8)` because the specified values are their defaults.

```
. power twoproportions 0.015 0.001

Performing iteration ...

Estimated sample sizes for a two-sample proportions test
Pearson's chi-squared test
H0: p2 = p1  versus  Ha: p2 != p1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =   -0.0140  (difference)
           p1 =    0.0150
           p2 =    0.0010

Estimated sample sizes:

            N =     1,270
  N per group =       635
```

A total sample of 1,270 individuals, 635 individuals per group, must be obtained to detect an absolute difference of 0.014 between proportions of aspirin users and nonusers with 80% power using a two-sided 5%-level Pearson's χ^2 test.

◁

▷ Example 2: Alternative ways of specifying effect

The displayed effect size `delta` in example 1 is the difference between the experimental-group proportion and the control-group proportion. We can redefine the effect size to be, for example, the odds ratio by specifying the `effect()` option.

```
. power twoproportions 0.015 0.001, effect(oratio)

Performing iteration ...

Estimated sample sizes for a two-sample proportions test
Pearson's chi-squared test
H0: p2 = p1  versus  Ha: p2 != p1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.0657  (odds ratio)
           p1 =    0.0150
           p2 =    0.0010

Estimated sample sizes:

            N =     1,270
  N per group =       635
```

The effect size `delta` now contains the estimated odds ratio and is labeled correspondingly.

Instead of the estimate of the proportion in the experimental group, we may have an estimate of the odds ratio $\{p_2(1-p_1)\}/\{p_1(1-p_2)\}$. For example, the estimate of the odds ratio in this example is 0.0657. We can specify the value of the odds ratio in the `oratio()` option instead of specifying the experimental-group proportion 0.001:

```
. power twoproportions 0.015, oratio(0.0657)

Performing iteration ...

Estimated sample sizes for a two-sample proportions test
Pearson's chi-squared test
H0: p2 = p1  versus  Ha: p2 != p1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.0657  (odds ratio)
           p1 =    0.0150
           p2 =    0.0010
   odds ratio =    0.0657

Estimated sample sizes:

            N =     1,270
  N per group =       635
```

The results are identical to the prior results. The estimate of the odds ratio is now displayed in the output, and the effect size `delta` now corresponds to the odds ratio.

We can also specify the following measures as input parameters: difference between proportions in the `diff()` option, risk difference in the `rdiff()` option, ratio of the proportions in the `ratio()` option, or relative risk in the `rrisk()` option.

◁

▷ Example 3: Likelihood-ratio test

Instead of the Pearson's χ^2 test as in example 1, we can compute sample size for the likelihood-ratio test by specifying the `test(lrchi2)` option.

```
. power twoproportions 0.015 0.001, test(lrchi2)

Performing iteration ...

Estimated sample sizes for a two-sample proportions test
Likelihood-ratio test
H0: p2 = p1  versus  Ha: p2 != p1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =   -0.0140  (difference)
           p1 =    0.0150
           p2 =    0.0010

Estimated sample sizes:

            N =     1,062
  N per group =       531
```

The required total sample size of 1,062 is smaller than that for the Pearson's χ^2 test.

◁

▷ Example 4: Computing one of the group sizes

Suppose we anticipate a sample of 600 aspirin users and wish to compute the required number of nonusers given the study parameters from example 1. We specify the number of aspirin users in `n2()`, and we also include `compute(N1)`:

```
. power twoproportions 0.015 0.001, n2(600) compute(N1)
Performing iteration ...
Estimated sample sizes for a two-sample proportions test
Pearson's chi-squared test
H0: p2 = p1  versus  Ha: p2 != p1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =   -0.0140  (difference)
           p1 =    0.0150
           p2 =    0.0010
           N2 =       600
Estimated sample sizes:
            N =     1,317
           N1 =       717
```

We require a sample of 717 nonusers given 600 aspirin users for a total of 1,317 subjects. The total number of subjects is larger for this unbalanced design compared with the corresponding balanced design in example 1.

◁

▷ Example 5: Unbalanced design

By default, `power twoproportions` computes sample size for a balanced or equal-allocation design. If we know the allocation ratio of subjects between the groups, we can compute the required sample size for an unbalanced design by specifying the `nratio()` option.

Continuing with example 1, we will suppose that we anticipate to recruit twice as many aspirin users as nonusers; that is, $n_2/n_1 = 2$. We specify the `nratio(2)` option to compute the required sample size for the specified unbalanced design.

```
. power twoproportions 0.015 0.001, nratio(2)
Performing iteration ...
Estimated sample sizes for a two-sample proportions test
Pearson's chi-squared test
H0: p2 = p1  versus  Ha: p2 != p1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =   -0.0140  (difference)
           p1 =    0.0150
           p2 =    0.0010
        N2/N1 =    2.0000
Estimated sample sizes:
            N =     1,236
           N1 =       412
           N2 =       824
```

We need a total sample size of 1,236 subjects.

Also see *Two samples* in [PSS-4] **Unbalanced designs** for more examples of unbalanced designs for two-sample tests.

◁

Computing power

To compute power, you must specify the total sample size in the `n()` option, the control-group proportion p_1, and the experimental-group proportion p_2.

▷ Example 6: Power of a two-sample proportions test

Continuing with example 1, we will suppose that we anticipate a sample of only 1,100 subjects. To compute the power corresponding to this sample size given the study parameters from example 1, we specify the sample size in `n()`:

```
. power twoproportions 0.015 0.001, n(1100)

Estimated power for a two-sample proportions test
Pearson's chi-squared test
H0: p2 = p1  versus  Ha: p2 != p1

Study parameters:

        alpha =    0.0500
            N =     1,100
  N per group =       550
        delta =   -0.0140  (difference)
           p1 =    0.0150
           p2 =    0.0010

Estimated power:

        power =    0.7416
```

With a smaller sample of 1,100 subjects, we obtain a lower power of 74% compared with example 1.

◁

▷ Example 7: Multiple values of study parameters

In this example, we would like to assess the effect of varying the proportion of aspirin users on the power of our study. Suppose that the total sample size is 1,100 with equal allocation between groups, and the value of the proportion in the nonusing group is 0.015. We specify a range of proportions for aspirin users from 0.001 to 0.009 with a step size of 0.001 as *numlist* in parentheses as the second argument of the command:

```
. power twoproportions 0.015 (0.001(0.001)0.009), n(1100)

Estimated power for a two-sample proportions test
Pearson's chi-squared test
H0: p2 = p1  versus  Ha: p2 != p1
```

alpha	power	N	N1	N2	delta	p1	p2
.05	.7416	1,100	550	550	-.014	.015	.001
.05	.6515	1,100	550	550	-.013	.015	.002
.05	.5586	1,100	550	550	-.012	.015	.003
.05	.4683	1,100	550	550	-.011	.015	.004
.05	.3846	1,100	550	550	-.01	.015	.005
.05	.3102	1,100	550	550	-.009	.015	.006
.05	.2462	1,100	550	550	-.008	.015	.007
.05	.1928	1,100	550	550	-.007	.015	.008
.05	.1497	1,100	550	550	-.006	.015	.009

From the table, the power decreases from 74% to 15% as the proportion of deaths for aspirin users increases from 0.001 to 0.009 or the absolute value of the effect size (measured as the difference between the proportion of deaths for aspirin users and that for nonusers) decreases from 0.014 to 0.006.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

▷ Example 8: Saw-toothed power function

We can also compute power for the small-sample Fisher's exact conditional test. The sampling distribution of the test statistic for this test is discrete. As such, Fisher's exact test shares the same issues arising with power and sample-size analysis as described in detail for the binomial one-sample proportion test in example 7 of [PSS-2] **power oneproportion**. In particular, the power function of Fisher's exact test has a saw-toothed shape as a function of the sample size. Here, we demonstrate the saw-toothed shape of the power function and refer you to example 7 of [PSS-2] **power oneproportion** for details.

Let's plot powers of the Fisher's exact test for a range of experimental-group sizes between 50 and 65 given the control-group proportion of 0.6, the experimental-group proportion of 0.25, and the control-group size of 25. We specify the `graph()` option to produce a graph and the `table()` option to produce a table; see [PSS-2] **power, graph** and [PSS-2] **power, table** for more details about the graphical and tabular outputs from power. Within `graph()`, we specify options to request that the reference line be plotted on the y axis at a power of 0.8 and that the data points be labeled with the corresponding sample sizes. Within `table()`, we specify the `formats()` option to display only three digits after the decimal point for the `power` and `alpha_a` columns.

```
. power twoproportions 0.6 0.25, test(fisher) n1(25) n2(50(1)65)
> graph(yline(0.8) plotopts(mlabel(N)))
> table(, formats(alpha_a "%7.3f" power "%7.3f"))
```

```
Estimated power for a two-sample proportions test
Fisher's exact test
H0: p2 = p1  versus  Ha: p2 != p1
```

alpha	alpha_a	power	N	N1	N2	delta	p1	p2
.05	0.027	0.771	75	25	50	-.35	.6	.25
.05	0.030	0.793	76	25	51	-.35	.6	.25
.05	0.028	0.786	77	25	52	-.35	.6	.25
.05	0.027	0.782	78	25	53	-.35	.6	.25
.05	0.030	0.804	79	25	54	-.35	.6	.25
.05	0.029	0.793	80	25	55	-.35	.6	.25
.05	0.028	0.786	81	25	56	-.35	.6	.25
.05	0.030	0.814	82	25	57	-.35	.6	.25
.05	0.028	0.802	83	25	58	-.35	.6	.25
.05	0.029	0.797	84	25	59	-.35	.6	.25
.05	0.029	0.823	85	25	60	-.35	.6	.25
.05	0.028	0.813	86	25	61	-.35	.6	.25
.05	0.029	0.807	87	25	62	-.35	.6	.25
.05	0.029	0.819	88	25	63	-.35	.6	.25
.05	0.029	0.821	89	25	64	-.35	.6	.25
.05	0.027	0.816	90	25	65	-.35	.6	.25

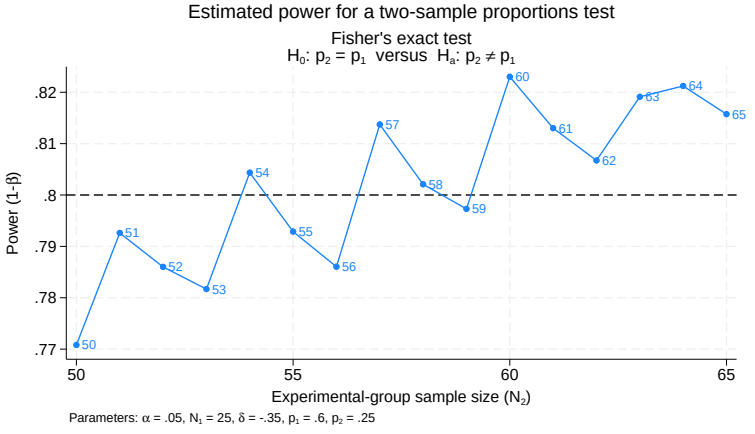


Figure 1. Saw-toothed power function

The power is not a monotonic function of the sample size. Also from the table, we see that all the observed significance levels are smaller than the specified level of 0.05.

Because of the saw-toothed nature of the power curve, obtaining an optimal sample size becomes tricky. For example, if we wish to have power of 80%, then from the above table and graph, we see that potential experimental-group sample sizes are 54, 57, 58, 60, and so on. One may be tempted to choose the smallest sample size for which the power is at least 80%. This, however, would not guarantee that the power is at least 80% for any larger sample size. Instead, Chernick and Liu (2002) suggest selecting the smallest sample size after which the troughs of the power curve do not go below the desired power. Following this recommendation in our example, we would pick a sample size of 60, which corresponds to the observed significance level of 0.028 and power of 0.823.

◁

Computing effect size and experimental-group proportion

There are multiple definitions of the effect size for a two-sample proportions test. By default, effect size δ is defined as the difference between the experimental-group proportion and the control-group proportion, $\delta = p_2 - p_1$, also known as a risk difference. Other available measures of the effect size are the ratio of the experimental-group proportion to the control-group proportion $\delta = p_2/p_1$, also known as a relative risk or risk ratio, and the odds ratio $\delta = \{p_2(1-p_1)\}/\{p_1(1-p_2)\}$.

Sometimes, we may be interested in determining the smallest effect and the corresponding experimental-group proportion that yield a statistically significant result for prespecified sample size and power. In this case, power, sample size, and control-group proportion must be specified. In addition, you must also decide on the direction of the effect: upper, meaning $p_2 > p_1$, or lower, meaning $p_2 < p_1$. The direction may be specified in the `direction()` option; `direction(upper)` is the default.

The underlying computations solve the corresponding power equation for the value of the experimental-group proportion given power, sample size, and other study parameters. The effect size is then computed from the specified control-group proportion and the computed experimental-group proportion using the corresponding functional relationship. The difference between proportions is reported by default, but you can request other measures by specifying the `effect()` option.

▷ Example 9: Minimum detectable change in the experimental-group proportion

Continuing with example 6, we will compute the smallest change in the proportion of deaths for aspirin users less than that for nonusers that can be detected given a total sample of 1,100 individuals and 80% power. To solve for the proportion of aspirin users in the experimental group, after the command name, we specify the control group (nonaspirin users), proportion of `0.015`, total sample size `n(1100)`, and power `power(0.8)`:

```
. power twoproportions 0.015, n(1100) power(0.8) direction(lower)

Performing iteration ...

Estimated experimental-group proportion for a two-sample proportions test
Pearson's chi-squared test
H0: p2 = p1  versus  Ha: p2 != p1; p2 < p1

Study parameters:

        alpha =    0.0500
        power =    0.8000
            N =     1,100
  N per group =       550
           p1 =    0.0150

Estimated effect size and experimental-group proportion:

        delta =   -0.0147  (difference)
           p2 =    0.0003
```

We find that given the proportion of nonusers of 0.015, the smallest (in absolute value) difference between proportions that can be detected in this study is -0.0147, which corresponds to the proportion of aspirin users of 0.0003.

Although the difference between proportions is reported by default, we can request that another risk measure be reported by specifying the `effect()` option. For example, we can request that the odds ratio be reported instead:

```
. power twoproportions 0.015, n(1100) power(0.8) direction(lower) effect(oratio)

Performing iteration ...

Estimated experimental-group proportion for a two-sample proportions test
Pearson's chi-squared test
H0: p2 = p1  versus  Ha: p2 != p1; p2 < p1

Study parameters:

        alpha =    0.0500
        power =    0.8000
            N =     1,100
  N per group =       550
           p1 =    0.0150

Estimated effect size and experimental-group proportion:

        delta =    0.0195  (odds ratio)
           p2 =    0.0003
```

The corresponding value of the odds ratio in this example is 0.0195.

In these examples, we computed the experimental-group proportion assuming a lower direction, $p_2 < p_1$, which required you to specify the `direction(lower)` option. By default, experimental-group proportion is computed for an upper direction, meaning that the proportion is greater than the specified value of the control-group proportion.

◁

Testing a hypothesis about two independent proportions

After the initial planning, we collect data and wish to test the hypothesis that the proportions from two independent populations are the same. We can use the `prtest` command to perform such hypothesis tests; see [R] **prtest** for details.

▷ Example 10: Two-sample proportions test

Consider a 2×3 contingency table provided in table 2.1 of Agresti (2013, 38). The table is obtained from a report by the Physicians' Health Study Research Group at Harvard Medical School that investigated the relationship between aspirin use and heart attacks.

The report presents summary data on fatal and nonfatal heart attacks. In the current example, we combine these two groups into a single group representing the total cases with heart attacks for aspirin users and nonusers. The estimated proportion of heart attacks in the control group, nonaspirin users, is $189/11034 = 0.0171$ and in the experimental group, aspirin users, is $104/11037 = 0.0094$.

```
. prtesti 11034 0.0171 11037 0.0094
Two-sample test of proportions                    x: Number of obs =    11034
                                                  y: Number of obs =    11037
------------------------------------------------------------------------------
             |       Mean   Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
           x |      .0171   .0012342                      .014681     .019519
           y |      .0094   .0009185                     .0075997    .0112003
-------------+----------------------------------------------------------------
        diff |      .0077   .0015385                     .0046846    .0107154
             |   under H0:   .0015393     5.00   0.000
------------------------------------------------------------------------------
        diff = prop(x) - prop(y)                                  z =   5.0023
    H0: diff = 0

    Ha: diff < 0                 Ha: diff != 0                 Ha: diff > 0
 Pr(Z < z) = 1.0000         Pr(|Z| > |z|) = 0.0000          Pr(Z > z) = 0.0000
```

Let p_A and p_N denote the proportions of heart attacks in the population for aspirin users and nonusers, respectively. From the above results, we find a statistically significant evidence to reject the null hypothesis H_0: $p_A = p_N$ versus a two-sided alternative H_a: $p_A \neq p_N$ at the 5% significance level; the p-value is very small.

We use the parameters of this study to perform a sample-size analysis we would have conducted before the study.

```
. power twoproportions 0.0171 0.0094
Performing iteration ...
Estimated sample sizes for a two-sample proportions test
Pearson's chi-squared test
H0: p2 = p1  versus  Ha: p2 != p1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =   -0.0077  (difference)
           p1 =    0.0171
           p2 =    0.0094
Estimated sample sizes:
            N =     6,922
  N per group =     3,461
```

We find that for Pearson's χ^2 test, a total sample size of 6,922, assuming a balanced design, is required to detect the difference between the control-group proportion of 0.0171 and the experimental-group proportion of 0.0094 with 80% power using a 5%-level two-sided test.

◁

Video examples

How to calculate sample size for two independent proportions

How to calculate power for two independent proportions

How to calculate minimum detectable effect size for two independent proportions

Stored results

`power twoproportions` stores the following in `r()`:

Scalars

`r(alpha)`	significance level
`r(alpha_a)`	actual significance level of the Fisher's exact test
`r(power)`	power
`r(beta)`	probability of a type II error
`r(delta)`	effect size
`r(N)`	total sample size
`r(N_a)`	actual sample size
`r(N1)`	sample size of the control group
`r(N2)`	sample size of the experimental group
`r(nratio)`	ratio of sample sizes, `N2/N1`
`r(nratio_a)`	actual ratio of sample sizes
`r(nfractional)`	1 if `nfractional` is specified, 0 otherwise
`r(onesided)`	1 for a one-sided test, 0 otherwise
`r(p1)`	control-group proportion
`r(p2)`	experimental-group proportion
`r(diff)`	difference between the experimental- and control-group proportions
`r(ratio)`	ratio of the experimental-group proportion to the control-group proportion
`r(rdiff)`	risk difference
`r(rrisk)`	relative risk
`r(oratio)`	odds ratio
`r(continuity)`	1 if continuity correction is used, 0 otherwise
`r(separator)`	number of lines between separator lines in the table
`r(divider)`	1 if `divider` is requested in the table, 0 otherwise
`r(init)`	initial value for sample sizes or experimental-group proportion
`r(maxiter)`	maximum number of iterations
`r(iter)`	number of iterations performed
`r(tolerance)`	requested parameter tolerance
`r(deltax)`	final parameter tolerance achieved
`r(ftolerance)`	requested distance of the objective function from zero
`r(function)`	final distance of the objective function from zero
`r(converged)`	1 if iteration algorithm converged, 0 otherwise

Macros

`r(type)`	`test`
`r(method)`	`twoproportions`
`r(test)`	`chi2`, `lrchi2`, or `fisher`
`r(effect)`	specified effect: `diff`, `ratio`, etc.
`r(direction)`	`upper` or `lower`
`r(columns)`	displayed table columns
`r(labels)`	table column labels

`r(widths)`	table column widths
`r(formats)`	table column formats

Matrices

`r(pss_table)`	table of results

Methods and formulas

Consider two independent samples of sizes n_1 and n_2. Let $x_{11}, \ldots, x_{1n_1}$ be a random sample from a binomial distribution with the success probability p_1. We refer to this sample as a control group. Let $x_{21}, \ldots, x_{2n_2}$ be a random sample from a binomial distribution with the success probability p_2, the experimental group. Let $x_{ij} = 1$ denote a success and $x_{ij} = 0$ denote a failure. The two samples are assumed to be independent.

The sample proportions and their respective standard errors are

$$\hat{p}_1 = \frac{1}{n_1}\sum_{i=1}^{n_1} x_{1i} \qquad \text{and} \qquad \text{se}(\hat{p}_1) = \sqrt{\frac{\hat{p}_1(1-\hat{p}_1)}{n_1}}$$

$$\hat{p}_2 = \frac{1}{n_2}\sum_{i=1}^{n_2} x_{2i} \qquad \text{and} \qquad \text{se}(\hat{p}_2) = \sqrt{\frac{\hat{p}_2(1-\hat{p}_2)}{n_2}}$$

A two-sample proportions test involves testing the null hypothesis H_0: $p_2 = p_1$ versus the two-sided alternative hypothesis H_a: $p_2 \neq p_1$, the upper one-sided alternative H_a: $p_2 > p_1$, or the lower one-sided alternative H_a: $p_2 < p_1$.

Let $R = n_2/n_1$ denote the allocation ratio. Then $n_2 = R \times n_1$ and power can be viewed as a function of n_1. Therefore, for sample-size determination, the control-group sample size n_1 is computed first. The experimental-group size n_2 is then computed as $R \times n_1$, and the total sample size is computed as $n = n_1 + n_2$. By default, sample sizes are rounded to integer values; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for details.

The formulas below are based on Fleiss, Levin, and Paik (2003) and Agresti (2013).

Methods and formulas are presented under the following headings:

Effect size

The measures of risk or effect size can be defined in a number of ways for the two-sample proportions test. By default, the effect size is the difference between the experimental-group proportion and the control-group proportion. Other available risk measures include risk difference, risk ratio or relative risk, and odds ratio.

By default or when `effect(diff)` or `effect(rdiff)` is specified, the effect size is computed as

$$\delta = p_2 - p_1$$

When `effect(ratio)` or `effect(rrisk)` is specified, the effect size is computed as

$$\delta = p_2/p_1$$

When `effect(oratio)` is specified, the effect size is computed as

$$\delta = \theta = \{p_2(1-p_1)\}/\{p_1(1-p_2)\}$$

If `diff()`, `rdiff()`, `ratio()`, `rrisk()`, or `oratio()` is specified, the value of the experimental-group proportion p_2 is computed using the corresponding formula from above.

Pearson's χ^2 test

For a large sample size, a binomial process can be approximated by a normal distribution. The asymptotic sampling distribution of the test statistic

$$z = \frac{(\hat{p}_2 - \hat{p}_1) - (p_2 - p_1)}{\sqrt{\hat{\overline{p}}(1-\hat{\overline{p}})\left(\frac{1}{n_1} + \frac{1}{n_2}\right)}}$$

is standard normal, where $\overline{p} = (n_1 p_1 + n_2 p_2)/(n_1 + n_2)$ is the pooled proportion and $\hat{\overline{p}}$ is its estimator. The square of this statistic, z^2, has an approximate χ^2 distribution with one degree of freedom, and the corresponding test is known as Pearson's χ^2 test.

Let α be the significance level, β be the probability of a type II error, and $z_{1-\alpha}$ and z_β be the $(1-\alpha)$th and the βth quantiles of the standard normal distribution.

Let $\sigma_D = \sqrt{p_1(1-p_1)/n_1 + p_2(1-p_2)/n_2}$ be the standard deviation of the difference between proportions and $\sigma_p = \sqrt{\overline{p}(1-\overline{p})\,(1/n_1 + 1/n_2)}$ be the pooled standard deviation.

The power $\pi = 1 - \beta$ is computed using

$$\pi = \begin{cases} \Phi\left\{\frac{(p_2-p_1)-c-z_{1-\alpha}\sigma_p}{\sigma_D}\right\} & \text{for an upper one-sided test} \\ \Phi\left\{\frac{-(p_2-p_1)-c-z_{1-\alpha}\sigma_p}{\sigma_D}\right\} & \text{for a lower one-sided test} \\ \Phi\left\{\frac{(p_2-p_1)-c-z_{1-\alpha/2}\sigma_p}{\sigma_D}\right\} + \Phi\left\{\frac{-(p_2-p_1)-c-z_{1-\alpha/2}\sigma_p}{\sigma_D}\right\} & \text{for a two-sided test} \end{cases} \quad (1)$$

where $\Phi(\cdot)$ is the cdf of the standard normal distribution, and c is the normal-approximation continuity correction. For equal sample sizes, $n_1 = n_2 = n/2$, the continuity correction is expressed as $c = 2/n$ (Levin and Chen 1999).

For a one-sided test, given the allocation ratio $R = n_2/n_1$, the total sample size n is computed by inverting the corresponding power equation in (1),

$$n = \frac{\left\{z_{1-\alpha}\sqrt{\overline{p}(1-\overline{p})} - z_\beta\sqrt{w_2 p_1(1-p_1) + w_1 p_2(1-p_2)}\right\}^2}{w_1 w_2\,(p_2 - p_1)^2} \quad (2)$$

where $w_1 = 1/(1+R)$ and $w_2 = R/(1+R)$. Then n_1 and n_2 are computed as $n_1 = n/(1+R)$ and $n_2 = R \times n_1$, respectively. If the `continuity` option is specified, the sample size n_c for a one-sided test is computed as

$$n_c = \frac{n}{4}\left(1 + \sqrt{1 + \frac{2}{n w_1 w_2 |p_2 - p_1|}}\right)^2$$

where n is the sample size computed without the correction. For unequal sample sizes, the continuity correction generalizes to $c = 1/(2nw_1w_2)$ (Fleiss, Levin, and Paik 2003).

For a two-sided test, the sample size is computed by iteratively solving the two-sided power equation in (1). The initial values for the two-sided computations are obtained from (2) with the significance level $\alpha/2$.

If one of the group sizes is known, the other one is computed by iteratively solving the corresponding power equation in (1). The initial values are obtained from (2) by assuming that $R = 1$.

The experimental-group proportion p_2 is computed by iteratively solving the corresponding power equations in (1). The default initial values are obtained by using a bisection method.

Likelihood-ratio test

Let $q_1 = 1 - p_1$, $q_2 = 1 - p_2$, and $\bar{q} = 1 - \bar{p} = 1 - (n_1p_1 + n_2p_2)/(n_1 + n_2)$. The likelihood-ratio test statistic is given by

$$G = \sqrt{2n\left\{\frac{n_1p_1}{n}\ln\left(\frac{p_1}{\bar{p}}\right) + \frac{n_1q_1}{n}\ln\left(\frac{q_1}{\bar{q}}\right) + \frac{n_2p_2}{n}\ln\left(\frac{p_2}{\bar{p}}\right) + \frac{n_2q_2}{n}\ln\left(\frac{q_2}{\bar{q}}\right)\right\}}$$

The power $\pi = 1 - \beta$ is computed using

$$\pi = \begin{cases} \Phi\left(G - z_{1-\alpha}\right) & \text{for an upper one-sided test} \\ \Phi\left(-G - z_{1-\alpha}\right) & \text{for a lower one-sided test} \\ \Phi\left(G - z_{1-\alpha/2}\right) + \Phi\left(-G - z_{1-\alpha/2}\right) & \text{for a two-sided test} \end{cases} \tag{3}$$

For a one-sided test, given the allocation ratio R, the total sample size n is computed by inverting the corresponding power equation in (3),

$$n = \frac{\left(z_{1-\alpha} - z_{\beta}\right)^2}{2\left\{w_1p_1\ln\left(\frac{p_1}{\bar{p}}\right) + w_1q_1\ln\left(\frac{q_1}{\bar{q}}\right) + w_2p_2\ln\left(\frac{p_2}{\bar{p}}\right) + w_2q_2\ln\left(\frac{q_2}{\bar{q}}\right)\right\}} \tag{4}$$

where $w_1 = 1/(1 + R)$ and $w_2 = R/(1 + R)$. Then n_1 and n_2 are computed as $n_1 = n/(1 + R)$ and $n_2 = R \times n_1$, respectively.

For a two-sided test, the sample size is computed by iteratively solving the two-sided power equation in (3). The initial values for the two-sided computations are obtained from equation (4) with the significance level $\alpha/2$.

If one of the group sizes is known, the other one is computed by iteratively solving the corresponding power equation in (3). The initial values are obtained from (4) by assuming that $R = 1$.

The experimental-group proportion p_2 is computed by iteratively solving the corresponding power equations in (3). The default initial values are obtained by using a bisection method.

Fisher's exact conditional test

Power computation for Fisher's exact test is based on Casagrande, Pike, and Smith (1978a). We present formulas from the original paper with a slight change in notation: we use $\widetilde{p_1}$ and $\widetilde{p_2}$ in place of p_1 and p_2 and η in place of θ. The change in notation is to avoid confusion between our use of group

proportions p_1 and p_2 and their use in the paper—compared with our definitions, the roles of p_1 and p_2 in the paper are reversed in the definitions of the hypotheses and other measures such as the odds ratio. In our definitions, $\widetilde{p_1} = p_2$ is the proportion in the experimental group, $\widetilde{p_2} = p_1$ is the proportion in the control group, and $\eta = 1/\theta$. Also we denote $\widetilde{n_1} = n_2$ to be the sample size of the experimental group and $\widetilde{n_2} = n_1$ to be the sample size of the control group.

Let k be the number of successes in the experimental group, and let m be the total number of successes in both groups. The conditional distribution of k is given by

$$p(k|m,\eta) = \frac{\binom{\widetilde{n_1}}{k}\binom{\widetilde{n_2}}{m-k}\eta^k}{\sum_i \binom{\widetilde{n_1}}{i}\binom{\widetilde{n_2}}{m-i}\eta^i}$$

where $\eta = \{\widetilde{p_1}(1-\widetilde{p_2})\}/\{\widetilde{p_2}(1-\widetilde{p_1})\}$, and the range of i is given by $L = \max(0, m-\widetilde{n_2})$ to $U = \min(\widetilde{n_1}, m)$.

Assume an upper one-sided test given by

$$H_0\colon \widetilde{p_1} = \widetilde{p_2} \qquad \text{versus} \qquad H_a\colon \widetilde{p_1} > \widetilde{p_2} \tag{5}$$

The hypothesis (5) in terms of η can be expressed as follows:

$$H_0\colon \eta = 1 \qquad \text{versus} \qquad H_a\colon \eta > 1$$

Let k_u be the critical value of k such that the following inequalities are satisfied:

$$\sum_{i=k_u}^{U} p(i|m,\eta=1) \le \alpha \qquad \text{and} \sum_{i=k_u-1}^{U} p(i|m,\eta=1) > \alpha \tag{6}$$

The conditional power is

$$\beta(\eta|m) = \sum_{i=k_u}^{U} p(i|m,\eta)$$

For a lower one-sided hypothesis $H_a\colon \widetilde{p_1} < \widetilde{p_2}$, the corresponding hypothesis in terms of η is given by

$$H_0\colon \eta = 1 \qquad \text{versus} \qquad H_a\colon \eta < 1$$

The conditional power in this case is

$$\beta(\eta|m) = \sum_{i=L}^{k_l} p(i|m,\eta)$$

where k_l is the critical value of k such that the following inequalities are satisfied:

$$\sum_{i=L}^{k_l} p(i|m,\eta=1) \le \alpha \qquad \text{and} \sum_{i=L}^{k_l+1} p(i|m,\eta=1) > \alpha \tag{7}$$

For a two-sided test, the critical values k_l and k_u are calculated using the inequalities (6) and (7) with $\alpha/2$, respectively.

Finally, the unconditional power is calculated as

$$\beta(\eta) = \sum_j \beta(\eta|j)P(j)$$

where j takes the value from 0 to n, and

$$P(j) = \sum_{i=L}^{U} \binom{\widetilde{n_1}}{i} \widetilde{p_1}^i (1-\widetilde{p_1})^{\widetilde{n_1}-i} \binom{\widetilde{n_2}}{j-i} \widetilde{p_2}^{j-i} (1-\widetilde{p_2})^{\widetilde{n_2}-j+i}$$

where $L = \max(0, j-\widetilde{n_2})$ and $U = \min(\widetilde{n_1}, j)$.

References

Agresti, A. 2013. *Categorical Data Analysis*. 3rd ed. Hoboken, NJ: Wiley.

Casagrande, J. T., M. C. Pike, and P. G. Smith. 1978a. The power function of the "exact" test for comparing two binomial distributions. *Journal of the Royal Statistical Society*, C ser., 27: 176–180. https://doi.org/10.2307/2346945.

———. 1978b. An improved approximate formula for calculating sample sizes for comparing two binomial distributions. *Biometrics* 34: 483–486. https://doi.org/10.2307/2530613.

Chernick, M. R., and C. Y. Liu. 2002. The saw-toothed behavior of power versus sample size and software solutions: Single binomial proportion using exact methods. *American Statistician* 56: 149–155. https://doi.org/10.1198/000313002317572835.

Fisher, R. A. 1935. *The Design of Experiments*. Edinburgh: Oliver and Boyd.

Fleiss, J. L., B. Levin, and M. C. Paik. 2003. *Statistical Methods for Rates and Proportions*. 3rd ed. New York: Wiley. https://doi.org/10.1002/0471445428.

Irwin, J. O. 1935. Tests of significance for differences between percentages based on small numbers. *Metron* 12: 83–94.

Levin, B., and X. Chen. 1999. Is the one-half continuity correction used once or twice to derive a well-known approximate sample size formula to compare two independent binomial distributions? *American Statistician* 53: 62–66. https://doi.org/10.1080/00031305.1999.10474431.

Also see

[PSS-2] **power twoproportions, cluster** — Power analysis for a two-sample proportions test, CRD

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[ADAPT] **gsdesign twoproportions** — Group sequential design for a two-sample proportions test

[R] **bitest** — Binomial probability test

[R] **prtest** — Tests of proportions

Description

`power twoproportions, cluster` computes group-specific numbers of clusters, group-specific cluster sizes, power, or the experimental-group proportion for a two-sample proportions test in a cluster randomized design (CRD). It computes group-specific numbers of clusters given cluster sizes, power, and the values of the control-group and experimental-group proportions. It also computes group-specific cluster sizes given numbers of clusters, power, and the values of the control-group and experimental-group proportions. Alternatively, it computes power given numbers of clusters, cluster sizes, and the values of the control-group and experimental-group proportions, or it computes the experimental-group proportion given numbers of clusters, cluster sizes, power, and the control-group proportion. See [PSS-2] **power twoproportions** for a general discussion of power and sample-size analysis for a two-sample proportions test. Also see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Quick start

Number of clusters for a test of H_0: $\pi_1 = \pi_2$ versus H_a: $\pi_1 \neq \pi_2$ given alternative control- and experimental-group proportions $p_1 = 0.8$ and $p_2 = 0.6$ with common cluster size of 5 using default intraclass correlation of 0.5, power of 0.8, and significance level $\alpha = 0.05$

```
power twoproportions 0.8 0.6, m1(5) m2(5)
```

Same as above, but assume the intraclass correlation is 0.4

```
power twoproportions 0.8 0.6, m1(5) m2(5) rho(0.4)
```

Same as above, but with an average cluster size of 5 and a coefficient of variation of 0.1

```
power twoproportions 0.8 0.6, m1(5) m2(5) rho(0.4) cvcluster(0.1)
```

Group-specific numbers of clusters where the ratio of experimental to control group clusters is 0.5

```
power twoproportions 0.8 0.6, m1(5) m2(5) rho(0.4) kratio(0.5)
```

Cluster sizes for a test of H_0: $\pi_1 = \pi_2$ versus H_a: $\pi_1 \neq \pi_2$ given alternative control- and experimental-group proportions $p_1 = 0.4$ and $p_2 = 0.2$ for 60 clusters in the control group and 40 clusters in the experimental group using default intraclass correlation of 0.5, power of 0.8, and significance level $\alpha = 0.05$

```
power twoproportions 0.4 0.2, k1(60) k2(40)
```

Same as above, but compute experimental-group cluster size given the control-group cluster size of 10

```
power twoproportions 0.4 0.2, k1(60) k2(40) m1(10) compute(M2)
```

Power given control-group proportion $p_1 = 0.1$, experimental-group proportion $p_2 = 0.2$, and 20 clusters with cluster sizes of 5 in the control and experimental groups

```
power twoproportions 0.1 0.2, k1(20) k2(20) m1(5) m2(5)
```

Power for multiple numbers of clusters in the experimental group

```
power twoproportions 0.1 0.2, k1(20) k2(20(5)50) m1(5) m2(5)
```

Same as above, but display results in a graph of power versus the number of clusters in the experimental group

```
power twoproportions 0.1 0.2, k1(20) k2(20(5)50) m1(5) m2(5) graph
```

Effect size and experimental-group proportion with power of 0.8

```
power twoproportions 0.1, k1(20) k2(20) m1(5) m2(5) power(0.8)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute numbers of clusters

```
power twoproportions p1 p2 , {mspec | nspec cluster} [ options ]
```

Compute cluster sizes

```
power twoproportions p1 p2 , kspec [ options ]
```

Compute power

```
power twoproportions p1 p2 , kspec { mspec | nspec } [ options ]
```

Compute effect size and experimental-group proportion

```
power twoproportions p1 , kspec { mspec | nspec } power(numlist) [ options ]
```

where p_1 is the proportion in the control (reference) group and p_2 is the proportion in the experimental (comparison) group. p_1 and p_2 may each be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

kspec is one of

```
k1()  k2()
k1()  [kratio()]
k2()  [kratio()]
```

mspec is one of

```
m1()  m2()
m1()  [mratio()]
m2()  [mratio()]
```

nspec is one of

```
n1()  n2()
n1()  [nratio()]
n2()  [nratio()]
```

options	Description
Main	
`cluster`	perform computations for a CRD; implied by `k1()`, `k2()`, `m1()`, or `m2()`
* `alpha(`*numlist*`)`	significance level; default is `alpha(0.05)`
* `power(`*numlist*`)`	power; default is `power(0.8)`
* `beta(`*numlist*`)`	probability of type II error; default is `beta(0.2)`
* `k1(`*numlist*`)`	number of clusters in the control group
* `k2(`*numlist*`)`	number of clusters in the experimental group
* `kratio(`*numlist*`)`	cluster ratio, `K2`/`K1`; default is `kratio(1)`
* `m1(`*numlist*`)`	cluster size of the control group
* `m2(`*numlist*`)`	cluster size of the experimental group
* `mratio(`*numlist*`)`	cluster-size ratio, `M2`/`M1`; default is `mratio(1)`
* `n1(`*numlist*`)`	sample size of the control group
* `n2(`*numlist*`)`	sample size of the experimental group
* `nratio(`*numlist*`)`	sample-size ratio, `N2`/`N1`; default is `nratio(1)`
`compute(K1 | K2 | M1 | M2)`	solve for the number of clusters or cluster size in one group given the other group
`nfractional`	allow fractional numbers of clusters, cluster sizes, and sample sizes
* `diff(`*numlist*`)`	difference between the experimental-group and control-group proportions, $p_2 - p_1$; specify instead of the experimental-group proportion p_2
* `ratio(`*numlist*`)`	ratio of the experimental-group proportion to the control-group proportion, p_2/p_1; specify instead of the experimental-group proportion p_2
* `rdiff(`*numlist*`)`	risk difference, $p_2 - p_1$; synonym for `diff()`
* `rrisk(`*numlist*`)`	relative risk, p_2/p_1; synonym for `ratio()`
* `oratio(`*numlist*`)`	odds ratio, $\{p_2(1-p_1)\}/\{p_1(1-p_2)\}$
`effect(`*effect*`)`	specify the type of effect to display; default is `effect(diff)`
* `rho(`*numlist*`)`	intraclass correlation; default is `rho(0.5)`
* `cvcluster(`*numlist*`)`	coefficient of variation for cluster sizes
`direction(upper|lower)`	direction of the effect for effect-size determination; default is `direction(upper)`, which means that the postulated value of the parameter is larger than the hypothesized value
`onesided`	one-sided test; default is two sided
`parallel`	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)

Table

[no]table[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
saving(*filename* [, replace])	save the table data to *filename*; use replace to overwrite existing *filename*

Graph

graph[(*graphopts*)]	graph results; see [PSS-2] **power, graph**

Iteration

init(#)	initial value for numbers of clusters, cluster sizes, or experimental-group proportion
iterate(#)	maximum number of iterations; default is iterate(500)
tolerance(#)	parameter tolerance; default is tolerance(1e-12)
ftolerance(#)	function tolerance; default is ftolerance(1e-12)
[no]log	suppress or display iteration log
[no]dots	suppress or display iterations as dots
notitle	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the parallel option.

collect is allowed; see **[U] 11.1.10 Prefix commands**.

notitle does not appear in the dialog box.

effect	Description
diff	difference between proportions, $p_2 - p_1$; the default
ratio	ratio of proportions, p_2/p_1
rdiff	risk difference, $p_2 - p_1$
rrisk	relative risk, p_2/p_1
oratio	odds ratio, $\{p_2(1-p_1)\}/\{p_1(1-p_2)\}$

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
alpha	significance level	α
power	power	$1-\beta$
beta	type-II-error probability	β
K1	number of clusters in the control group	K_1
K2	number of clusters in the experimental group	K_2
kratio	ratio of numbers of clusters, experimental to control	K_2/K_1
M1	cluster size of the control group	M_1
M2	cluster size of the experimental group	M_2
mratio	ratio of cluster sizes, experimental to control	M_2/M_1
N	total number of observations	N
N1	number of observations in the control group	N_1
N2	number of observations in the experimental group	N_2
nratio	ratio of sample sizes, experimental to control	N_2/N_1
delta	effect size	δ
p1	control-group proportion	p_1
p2	experimental-group proportion	p_2
diff	difference between the experimental-group proportion and the control-group proportion	p_2-p_1
ratio	ratio of the experimental-group proportion to the control-group proportion	p_2/p_1
rdiff	risk difference	p_2-p_1
rrisk	relative risk	p_2/p_1
oratio	odds ratio	θ
rho	intraclass correlation	ρ
CV_cluster	coefficient of variation for cluster sizes	$\mathrm{CV}_{\mathrm{cl}}$
target	target parameter; synonym for p2	
_all	display all supported columns	

Column beta is shown in the default table in place of column power if specified.

Column N is shown in the table if specified.

Columns N1 and N2 are shown in the default table if n1() or n2() is specified.

Columns nratio, diff, ratio, rdiff, rrisk, oratio, and CV_cluster are shown in the default table if specified.

Options

Main

cluster specifies that computations should be performed for a CRD. This option is implied when the k1(), k2(), m1(), or m2() option is specified. cluster is required to compute the numbers of clusters when *nspec* is used to specify sample sizes instead of *mspec* for cluster sizes.

alpha(), power(), beta(); see [PSS-2] **power**.

k1(*numlist*) specifies the number of clusters in the control group.

k2(*numlist*) specifies the number of clusters in the experimental group.

kratio(*numlist*) specifies the ratio of the numbers of clusters of the experimental group relative to the control group, K2/K1. The default is kratio(1), meaning equal numbers of clusters in the two groups.

`m1(`*numlist*`)` specifies the cluster size of the control group. `m1()` may contain noninteger values.

`m2(`*numlist*`)` specifies the cluster size of the experimental group. `m2()` may contain noninteger values.

`mratio(`*numlist*`)` specifies the ratio of cluster sizes of the experimental group relative to the control group, `M2`/`M1`. The default is `mratio(1)`, meaning equal cluster sizes in the two groups.

`n1()`, `n2()`, `nratio()`; see [PSS-2] **power**.

`compute(K1 | K2 | M1 | M2)` solve for the number of clusters or cluster size of one group given the other group.

`nfractional`; see [PSS-2] **power**. The `nfractional` option displays fractional (without rounding) values of the numbers of clusters, cluster sizes, and sample sizes.

`diff()`, `ratio()`, `rdiff()`, `rrisk()`, `oratio()`, `effect()`; see [PSS-2] **power twoproportions**.

`rho(`*numlist*`)` specifies the intraclass correlation. The default is `rho(0.5)`.

`cvcluster(`*numlist*`)` specifies the coefficient of variation for cluster sizes. This option is used with varying cluster sizes.

`direction()`, `onesided`, `parallel`; see [PSS-2] **power**.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies the initial value for the numbers of clusters or cluster sizes for sample-size determination or the initial value for the experimental-group proportion for the effect-size determination. The default is to use a closed-form normal approximation to compute an initial value for the estimated parameter.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following option is available with `power twoproportions, cluster` but is not shown in the dialog box:

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

`power twoproportions, cluster` requests that computations for the `power twoproportions` command be done for a CRD. In a CRD, groups of subjects or clusters are randomized instead of individual subjects, so the sample size is determined by the numbers of clusters and the cluster sizes. The sample-size determination thus consists of the determination of the numbers of clusters given cluster sizes or the determination of cluster sizes given the numbers of clusters. For a general discussion of using `power twoproportions`, see [PSS-2] **power twoproportions**. The discussion below is specific to the CRD.

Using power twoproportions, cluster

If you specify the `cluster` option, include `k1()` or `k2()` to specify the number of clusters or include `m1()` or `m2()` to specify the cluster size, the `power twoproportions` command will perform computations for a two-sample proportions test in a CRD. The computations for a CRD are based on the large-sample Pearson's χ^2 test.

All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option. You can specify the `onesided` option to request a one-sided test. By default, all computations assume a balanced or equal-allocation design, meaning equal numbers of clusters and cluster sizes in both groups; see [PSS-4] **Unbalanced designs** for a description of how to specify an unbalanced design.

To compute the number of clusters in both groups, you must provide cluster sizes for both groups. There are multiple ways to supply cluster sizes, but the most common is to specify the cluster size of the control group in the `m1()` option and the cluster size of the experimental group in the `m2()` option. See *mspec* and *nspec* under *Syntax* for other specifications. When *nspec* is specified, the `cluster` option is also required to request that `power twoproportions` perform computations for a CRD. The number of clusters is assumed to be equal in the two groups, but you can change this by specifying the ratio of the numbers of clusters in the experimental to the control group in the `kratio()` option. Other parameters are specified as described in *Using power twoproportions* in [PSS-2] **power twoproportions**.

To compute the cluster sizes in both groups, you must provide the numbers of clusters in both groups. There are several ways to supply the numbers of clusters; see *kspec* under *Syntax*. The most common is to specify the numbers of clusters in the control group and the experimental group in the `k1()` and `k2()` options, respectively. Equal cluster sizes are assumed in the two groups, but you can change this by specifying the ratio of the cluster sizes in the experimental to that of the control group in the `mratio()` option. Other parameters are specified as described in *Using power twoproportions* in [PSS-2] **power twoproportions**.

You can also compute the number of clusters or the cluster size in one of the groups given the number of clusters or the cluster size in the other group by specifying the `compute()` option. For example, to compute the number of clusters in the control group, you specify `compute(K1)` and provide the number

of clusters in the experimental group in `k2()`. Likewise, to compute the cluster size in the control group, you specify `compute(M1)` and provide the cluster size of the experimental group in `m2()`. You can compute the number of clusters or cluster size for the experimental group in a similar manner.

Instead of the experimental-group proportion p_2, you can specify other alternative measures of effect when computing numbers of clusters, cluster sizes, or power; see *Alternative ways of specifying effect* in [PSS-2] **power twoproportions**.

The power and effect-size determination is the same as described in *Using power twoproportions* in [PSS-2] **power twoproportions**, but the sample-size information is supplied as the numbers of clusters *kspec* and either cluster sizes using *mspec* or, less commonly, sample sizes using *nspec*.

All computations assume an intraclass correlation of 0.5. You can change this by specifying the `rho()` option. Also, all clusters are assumed to be of the same size unless the coefficient of variation for cluster sizes is specified in the `cvcluster()` option.

By default, the computed numbers of clusters, cluster sizes, and sample sizes are rounded up. However, you can specify the `nfractional` option to see the corresponding fractional values; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. If the `cvcluster()` option is specified when computing cluster sizes, then cluster sizes represent average cluster sizes and are thus not rounded. When sample sizes are specified using *nspec*, fractional cluster sizes may be reported to accommodate the specified numbers of clusters and sample sizes.

Some of `power twoproportions, cluster`'s computations require iteration, such as to compute the numbers of clusters for a two-sided test; see *Methods and formulas* for details and [PSS-2] **power** for the descriptions of options that control the iteration procedure.

Computing numbers of clusters

To compute the numbers of clusters in each group, you must either provide the cluster size for each group using *mspec* or specify the `cluster` option and provide the sample sizes of both groups using *nspec*. The most common method is to use *mspec* of `m1()` and `m2()`. In addition, the control- and experimental-group proportions must be specified.

▷ Example 1: Numbers of clusters for a two-sample proportions test in a CRD, specify cluster sizes

Consider a study investigating the effectiveness of a program to promote after-school activities in increasing the rate of students participating in the after-school club. Schools that are involved in the study will be randomly assigned either to the experimental group that participates in the program or to the control group that does not. A researcher plans to recruit 50 students from each school and assumes an intraclass correlation of 0.2. The researcher wants to be able to detect an increase of 0.2 in the anticipated control-group rate of 0.4, which corresponds to the experimental-group rate of 0.6.

To compute the number of schools in each group required to detect the desired rate with 80% power using a 5%-level two-sided test, we type

```
. power twoproportions 0.4 0.6, m1(50) m2(50) rho(0.2)
Performing iteration ...
Estimated numbers of clusters for a two-sample proportions test
Cluster randomized design, Pearson's chi-squared test
H0: p2 = p1  versus  Ha: p2 != p1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.2000  (difference)
           p1 =    0.4000
           p2 =    0.6000
Cluster design:
           M1 =        50
           M2 =        50
          rho =    0.2000
Estimated numbers of clusters and sample sizes:
           K1 =        21
           K2 =        21
           N1 =     1,050
           N2 =     1,050
```

We find that for 50 students per school, 21 schools per group and thus a total of 1,050 students per group are required to detect a 0.2 difference, from 0.4 to 0.6, in participation rates in the after-school club with 80% power using a 5%-level two-sided test.

Example 2: Numbers of clusters for a two-sample proportions test in a CRD, specify sample sizes

Suppose that for our study, we can recruit only 1,000 students per group because of limited funding. We need to know the number of schools in each group and how many students to recruit from each school. In this case, we specify the `n1(1000)` and `n2(1000)` options. Because none of the `k1()`, `k2()`, `m1()`, or `m2()` options are specified, we also need to specify the `cluster` option to request the test for a CRD instead of the conventional individual-level design.

```
. power twoproportions 0.4 0.6, cluster n1(1000) n2(1000) rho(0.2)
Performing iteration ...
Estimated numbers of clusters for a two-sample proportions test
Cluster randomized design, Pearson's chi-squared test
H0: p2 = p1  versus  Ha: p2 != p1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.2000  (difference)
           p1 =    0.4000
           p2 =    0.6000
Cluster design:
           N1 =     1,000
           N2 =     1,000
          rho =    0.2000
Estimated numbers of clusters and cluster sizes:
           K1 =        22
           K2 =        22
   Average M1 =   45.4545
   Average M2 =   45.4545
```

To achieve the desired power, we need to recruit an average of about 45 students per school from 22 schools for each of the control and experimental groups. `power twoproportions, cluster` did not round the cluster sizes of 45.45 to meet our required total of 1,000 students per group. In practice, you can decide to recruit 45 members from some schools and 46 from other schools to have roughly constant cluster sizes or decide to change the total number of students you want to recruit.

◁

Computing number of clusters in one group

To compute the number of clusters in one of the groups, you must specify the `compute()` option and the number of clusters in the other group. For example, to compute the number of clusters in the experimental group, you must specify the `compute(K2)` option and provide the number of clusters in the control group in the `k1()` option. Similarly, you can compute the number of clusters for the control group. In addition, you must provide cluster sizes *mspec* or sample sizes *nspec* of both groups and the control- and experimental-group proportions.

▷ Example 3: Number of clusters in the experimental group for a two-sample proportions test in a CRD

Continuing with example 1, suppose that we are designing a new study and we are planning to recruit 30 schools for the control group. We want to know the minimum number of schools we need to recruit to the experimental group. Given other study parameters from example 1, we compute the number of schools in the experimental group by specifying the `compute(K2)` option and the number of clusters in the control group of 30 in the `k1()` option:

```
. power twoproportions 0.4 0.6, compute(K2) k1(30) m1(50) m2(50) rho(0.2)
Performing iteration ...
Estimated treatment-group number of clusters for a two-sample proportions test
Cluster randomized design, Pearson's chi-squared test
H0: p2 = p1  versus  Ha: p2 != p1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.2000  (difference)
           p1 =    0.4000
           p2 =    0.6000
Cluster design:
           K1 =        30
           M1 =        50
           M2 =        50
           N1 =     1,500
          rho =    0.2000
Estimated number of clusters and sample size:
           K2 =        17
           N2 =       850
```

With 30 schools in the control group, we need to recruit 17 schools for the experimental group.

◁

Computing cluster sizes

To compute cluster sizes in both groups, you must provide the numbers of clusters in both groups by using *kspec*. The most common method is to specify the numbers of clusters in the control and experimental groups in the `k1()` and `k2()` options, respectively. In addition, the control- and experimental-group proportions must be specified.

▷ Example 4: Cluster sizes for a two-sample proportions test in a CRD

Continuing with example 1, suppose that we are designing a new study and we are planning to recruit 40 schools, with 20 schools in each group. Given other study parameters from example 1, we compute the number of students to recruit from each school by specifying 20 clusters in the `k1()` and `k2()` options:

```
. power twoproportions 0.4 0.6, k1(20) k2(20) rho(0.2)
Performing iteration ...
Estimated cluster sizes for a two-sample proportions test
Cluster randomized design, Pearson's chi-squared test
H0: p2 = p1  versus  Ha: p2 != p1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.2000  (difference)
           p1 =    0.4000
           p2 =    0.6000
Cluster design:
           K1 =        20
           K2 =        20
          rho =    0.2000
Estimated cluster sizes and sample sizes:
           M1 =       127
           M2 =       127
           N1 =     2,540
           N2 =     2,540
```

With 20 schools per group, we need to recruit 127 students per school for a total of 2,540 students per group.

◁

Computing power

To compute power in a CRD, you supply the sample-size information as the numbers of clusters by using *kspec* along with either the cluster sizes by using *mspec* or, less commonly, the sample sizes by using *nspec*. The most common method is to specify the `k1()`, `k2()`, `m1()`, and `m2()` options. In addition, the control- and experimental-group proportions must be specified.

▷ Example 5: Power for a two-sample proportions test in a CRD

Continuing with example 1, suppose that we can recruit 50 students from each of 40 schools (20 schools per group) and we want to compute power for this design. Given other study parameters from example 1, we compute the power by specifying 20 clusters in the `k1()` and `k2()` options and the cluster size of 50 in the `m1()` and `m2()` options:

```
. power twoproportions 0.4 0.6, k1(20) k2(20) m1(50) m2(50) rho(0.2)
Estimated power for a two-sample proportions test
Cluster randomized design, Pearson's chi-squared test
H0: p2 = p1  versus  Ha: p2 != p1
Study parameters:
        alpha =    0.0500
        delta =    0.2000  (difference)
           p1 =    0.4000
           p2 =    0.6000
Cluster design:
           K1 =        20
           K2 =        20
           M1 =        50
           M2 =        50
           N1 =     1,000
           N2 =     1,000
          rho =    0.2000
Estimated power:
        power =    0.7815
```

The computed power is about 78%.

◁

▷ Example 6: Multiple values of study parameters

To investigate the effect of the number of clusters in the experimental group on power, we can specify a list of numbers in the `k2()` option:

```
. power twoproportions 0.4 0.6, k1(20) k2(5(10)45) m1(50) m2(50) rho(0.2)
> table(power K2)
Estimated power for a two-sample proportions test
Cluster randomized design, Pearson's chi-squared test
H0: p2 = p1  versus  Ha: p2 != p1
```

power	K2
.4095	5
.7164	15
.8233	25
.8721	35
.8987	45

In this example, we also specified the `table(power K2)` option to list the only two columns that vary. As expected, as the number of clusters increases, the power tends to get closer to 1.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

Computing effect size and experimental-group proportion

There are multiple definitions of the effect size for a two-sample proportions test; see *Computing effect size and experimental-group proportion* in [PSS-2] **power twoproportions** for details. To compute effect size in a CRD, you supply the sample-size information as the numbers of clusters by using

kspec along with either the cluster sizes by using *mspec* or, less commonly, the sample sizes by using *nspec*. The most common method is to specify the `k1()`, `k2()`, `m1()`, and `m2()` options. In addition, power and control-group proportion must be specified. You must also decide on the direction of the effect, which is specified in the `direction()` option. For the default, upper, meaning $p_2 > p_1$, `power twoproportions, cluster` uses `direction(upper)`. For lower, meaning $p_2 < p_1$, specify `direction(lower)`.

▷ Example 7: Effect size for a two-sample proportions test in a CRD

Continuing with example 5, we may also be interested in finding the minimum value of the participation rate in the after-school club that can be detected with a sample of 20 schools per group, 50 students per school, and 80% power. To compute this, we specify the control-group rate of 0.4 as the command argument and the required options `k1(20)`, `k2(20)`, `m1(50)`, `m2(50)`, and `power(0.8)` and continue to use `rho(0.2)`.

```
. power twoproportions 0.4, k1(20) k2(20) m1(50) m2(50) power(0.8) rho(0.2)

Performing iteration ...

Estimated experimental-group proportion for a two-sample proportions test
Cluster randomized design, Pearson's chi-squared test
H0: p2 = p1  versus  Ha: p2 != p1; p2 > p1

Study parameters:

        alpha =    0.0500
        power =    0.8000
           p1 =    0.4000

Cluster design:

           K1 =        20
           K2 =        20
           M1 =        50
           M2 =        50
           N1 =     1,000
           N2 =     1,000
          rho =    0.2000

Estimated effect size and experimental-group proportion:

        delta =    0.2046  (difference)
           p2 =    0.6046
```

Given the proportion of 0.4 in the control group, the smallest (in absolute value) difference that can be detected is about 0.2, corresponding to a proportion of about 0.6 in the experimental group.

◁

Testing hypotheses about two proportions in a CRD

There are different ways to account for a CRD or for clustered data when performing hypothesis tests that compare proportions in two groups. With large samples or when you know intraclass correlation, the simplest way is to use a large-sample test that accounts for clustered data; see [R] **prtest** for details. More commonly, two-level binary models such as `melogit` are used because they also allow adjusting for covariates.

In this section, we briefly demonstrate the `prtest` command for comparing proportions of two groups with clustered data.

Example 8: Two-sample proportions test with clustered data

Consider example 4 from [R] **prtest** that compared the proportions of bacterial pneumonia episodes in the two groups of infants from 18 clusters in each group. Each group received a different type of vaccine: the control group received a meningococcal C conjugate vaccine (MnCC) and the experimental group received the seven-valent pneumococcal conjugate vaccince (PnCRM7). The two groups are identified by the `vaccine` variable, and the `pneumonia` variable records 1 if an infant had at least one bacterial pneumonia episode and 0 otherwise. See example 4 in [R] **prtest** for other details. We replicate the analysis from that example below.

For clustered data, `prtest` requires that we specify the cluster identifier in the `cluster()` option and population intraclass correlation in the `rho()` option.

```
. use https://www.stata-press.com/data/r19/pneumoniacrt
(Bacterial pneumonia episodes data from CRT (Hayes and Moulton 2009))
. prtest pneumonia, by(vaccine) cluster(cluster) rho(0.02)

Two-sample test of proportions
Cluster variable: cluster

Group: MnCC                                  Group: PnCRM7
  Number of obs      =       238               Number of obs      =       211
  Number of clusters =        18               Number of clusters =        18
  Avg. cluster size  =     13.22               Avg. cluster size  =     11.72
  CV cluster size    =    0.9605               CV cluster size    =    0.7976
  Intraclass corr.   =    0.0200               Intraclass corr.   =    0.0200

------------------------------------------------------------------------------
       Group |       Mean   Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
       MnCC |   .2226891   .0329017                      .1582029    .2871753
     PnCRM7 |   .1658768   .0299027                      .1072686     .224485
-------------+----------------------------------------------------------------
       diff |   .0568123    .04446                     -.0303278    .1439524
            |  under H0:   .0447641     1.27   0.204
------------------------------------------------------------------------------
       diff = prop(MnCC) - prop(PnCRM7)                          z =   1.2691
   H0: diff = 0

   Ha: diff < 0                 Ha: diff != 0                 Ha: diff > 0
Pr(Z < z) = 0.8978         Pr(|Z| > |z|) = 0.2044          Pr(Z > z) = 0.1022
```

We do not have statistical evidence to reject the null hypothesis that the two group proportions are the same.

Suppose that we want to use the results of this study to design another study that compares the two vaccines in the same population. Specifically, we want to compute the required number of clusters given the average cluster sizes of 13.22 and 11.72 in the two groups, the intraclass correlation of 0.02, and the coefficient of variation of cluster sizes of 0.96, as shown in the output above. The coefficients of variation of cluster sizes are slightly different between the two groups, so we use the larger value to obtain conservative results from `power twoproportions`, which assumes a common coefficient of variation of cluster sizes. We also use the observed proportion estimates of 0.22 and 0.17 in the computation.

```
. power twoproportions 0.22 0.17, m1(13.22) m2(11.72) rho(0.02) cvcluster(0.96)
Performing iteration ...
Estimated numbers of clusters for a two-sample proportions test
Cluster randomized design, Pearson's chi-squared test
H0: p2 = p1  versus  Ha: p2 != p1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =   -0.0500  (difference)
           p1 =    0.2200
           p2 =    0.1700
Cluster design:
   Average M1 =   13.2200
   Average M2 =   11.7200
          rho =    0.0200
        CV_cl =    0.9600
Estimated numbers of clusters and sample sizes:
           K1 =       115
           K2 =       115
           N1 =     1,521
           N2 =     1,348
```

The required number of clusters for each group is 115. Given varying cluster sizes, we need to have a total of 1,521 infants in the control group and a total of 1,348 infants in the experimental group.

The observed difference of roughly 0.05 between the two proportions may be too small for the purpose of PSS computations. Suppose that we want to use a larger difference (in absolute value), for example, from 0.22 to 0.1, between the two proportions.

```
. power twoproportions 0.22 0.1, m1(13.22) m2(11.72) rho(0.02) cvcluster(0.96)
Performing iteration ...
Estimated numbers of clusters for a two-sample proportions test
Cluster randomized design, Pearson's chi-squared test
H0: p2 = p1  versus  Ha: p2 != p1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =   -0.1200  (difference)
           p1 =    0.2200
           p2 =    0.1000
Cluster design:
   Average M1 =   13.2200
   Average M2 =   11.7200
          rho =    0.0200
        CV_cl =    0.9600
Estimated numbers of clusters and sample sizes:
           K1 =        17
           K2 =        17
           N1 =       225
           N2 =       200
```

In this case, we need only 17 clusters and a total of 225 and 200 infants in the control and experimental groups, respectively.

◁

Stored results

power twoproportions, cluster stores the following in r():

Scalars

r(alpha)	significance level
r(power)	power
r(beta)	probability of a type II error
r(delta)	effect size
r(K1)	number of clusters in the control group
r(K2)	number of clusters in the experimental group
r(kratio)	ratio of numbers of clusters, K2/K1
r(M1)	cluster size of the control group
r(M2)	cluster size of the experimental group
r(mratio)	ratio of cluster sizes, M2/M1
r(N)	total sample size
r(N1)	sample size of the control group
r(N2)	sample size of the experimental group
r(nratio)	ratio of sample sizes, N2/N1
r(nfractional)	1 if nfractional is specified, 0 otherwise
r(onesided)	1 for a one-sided test, 0 otherwise
r(p1)	control-group proportion
r(p2)	experimental-group proportion
r(diff)	difference between the experimental- and control-group proportions
r(ratio)	ratio of the experimental-group proportion to the control-group proportion
r(rdiff)	risk difference
r(rrisk)	relative risk
r(oratio)	odds ratio
r(rho)	intraclass correlation
r(CV_cluster)	coefficient of variation for cluster sizes
r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise
r(init)	initial value for estimated parameter
r(maxiter)	maximum number of iterations
r(iter)	number of iterations performed
r(tolerance)	requested parameter tolerance
r(deltax)	final parameter tolerance achieved
r(ftolerance)	requested distance of the objective function from zero
r(function)	final distance of the objective function from zero
r(converged)	1 if iteration algorithm converged, 0 otherwise

Macros

r(type)	test
r(method)	twoproportions
r(design)	CRD
r(test)	chi2
r(effect)	specified effect: diff, ratio, etc.
r(direction)	upper or lower
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results

Methods and formulas

The computation in a CRD is based on the Pearson's χ^2 test under the large-sample normal approximation, adjusted by the cluster design; see *Pearson's χ^2 test* under *Methods and formulas* in [PSS-2] **power twoproportions** for the common notation for a two-sample proportions test.

In a CRD, let K_1 and K_2 be the numbers of clusters in the control and experimental groups, respectively, and M_1 and M_2 be the cluster sizes of the control and experimental groups, respectively. We have $n_1 = K_1M_1$ and $n_2 = K_2M_2$. Let R_k be the ratio of the numbers of clusters, K_2/K_1, and R_m be the ratio of the cluster sizes, M_2/M_1. Let ρ be the intraclass correlation coefficient and DE_1 and DE_2 be the design effect in the control and experimental groups, with

$$\text{DE}_1 = 1 + \rho(M_1 - 1) \qquad \text{and} \qquad \text{DE}_2 = 1 + \rho(M_2 - 1)$$

For unequal cluster sizes, we assume that the cluster sizes are independent and identically distributed and are small relative to the number of clusters; see Ahn, Heo, and Zhang (2015) for details. Let the coefficient of variation of the cluster sizes be CV_{cl}. According to van Breukelen, Candel, and Berger (2007) and Campbell and Walters (2014), to adjust for varying cluster sizes, define the relative efficiency of unequal versus equal cluster sizes as

$$\text{RE}_i = 1 - \lambda_i(1 - \lambda_i)\text{CV}_{\text{cl}}^2$$

where $\lambda_i = \rho M_i/(\rho M_i + 1 - \rho)$, where $i = 1$ corresponds to the control group and $i = 2$ corresponds to the experimental group.

When cluster sizes are equal, we use $\text{RE}_1 = \text{RE}_2 = 1$ in the formulas.

For a large sample size, a binomial process can be approximated by a normal distribution. Similarly to the discussion for the two-sample proportions test in the individual-level design, the asymptotic sampling distribution of the test statistic

$$z = \frac{(\hat{p}_2 - \hat{p}_1) - (p_2 - p_1)}{\sqrt{\overline{p}(1 - \overline{p})\left(\frac{\text{DE}_1}{n_1\text{RE}_1} + \frac{\text{DE}_2}{n_2\text{RE}_2}\right)}}$$

is standard normal, where the pooled proportion $\overline{p}$ is defined as

$$\overline{p} = \frac{n_1p_1\text{RE}_1/\text{DE}_1 + n_2p_2\text{RE}_2/\text{DE}_2}{n_1\text{RE}_1/\text{DE}_1 + n_2\text{RE}_2/\text{DE}_2}$$

The square of this statistic, z^2, has an approximate χ^2 distribution with one degree of freedom, and the corresponding test is known as Pearson's χ^2 test.

Let α be the significance level, β be the probability of a type II error, and $z_{1-\alpha}$ and z_β be the $(1-\alpha)$th and the βth quantiles of the standard normal distribution.

The power $\pi = 1 - \beta$ is computed using

$$\pi = \begin{cases} \Phi\left\{\frac{(p_2-p_1)-z_{1-\alpha}\sigma_p}{\sigma_D}\right\} & \text{for an upper one-sided test} \\ \Phi\left\{\frac{-(p_2-p_1)-z_{1-\alpha}\sigma_p}{\sigma_D}\right\} & \text{for a lower one-sided test} \\ \Phi\left\{\frac{(p_2-p_1)-z_{1-\alpha/2}\sigma_p}{\sigma_D}\right\} + \Phi\left\{\frac{-(p_2-p_1)-z_{1-\alpha/2}\sigma_p}{\sigma_D}\right\} & \text{for a two-sided test} \end{cases} \tag{1}$$

where $\Phi(\cdot)$ is the c.d.f. of the standard normal distribution, σ_p is the pooled standard deviation, and σ_D is the standard deviation of the difference between proportions, both defined below.

$$\sigma_p = \sqrt{\overline{p}(1-\overline{p})\{\mathrm{DE}_1/(n_1\mathrm{RE}_1) + \mathrm{DE}_2/(n_2\mathrm{RE}_2)\}}$$

$$\sigma_D = \sqrt{p_1(1-p_1)\mathrm{DE}_1/(n_1\mathrm{RE}_1) + p_2(1-p_2)\mathrm{DE}_2/(n_2\mathrm{RE}_2)}$$

The above definitions of σ_p and σ_D are based on the "minimum variance weights" method in Ahn, Heo, and Zhang (2015, 32). The relative efficiencies RE_1 and RE_2 are used to adjust for varying cluster sizes approximately.

Given the cluster sizes M_1 and M_2, and intraclass correlation ρ, we can compute R_m, DE_1, DE_2, RE_1, and RE_2. With these parameters and the ratio of the numbers of clusters R_k, the numbers of clusters K_1 and K_2 for a one-sided test are computed as follows. K_1 is computed by inverting a one-sided power equation from (1),

$$K_1 = \frac{\left\{z_{1-\alpha}\sqrt{\overline{p}(1-\overline{p})} - z_{\beta}\sqrt{w_1p_1(1-p_1) + w_2p_2(1-p_2)}\right\}^2}{w_1w_2\left(p_2-p_1\right)^2\left(\mathrm{RE}_1/\mathrm{DE}_1 + R_kR_m\mathrm{RE}_2/\mathrm{DE}_2\right)}$$

where $w_1 = 1 - 1/\{1 + R_kR_m\mathrm{DE}_1\mathrm{RE}_2/(\mathrm{DE}_2\mathrm{RE}_1)\}$ and $w_2 = 1 - w_1$. Then, K_2 is computed using $K_2 = R_kK_1$.

In all other cases, parameters are computed iteratively using the power equations in (1).

References

Ahn, C., M. Heo, and S. Zhang. 2015. *Sample Size Calculations for Clustered and Longitudinal Outcomes in Clinical Research.* Boca Raton, FL: CRC Press. https://doi.org/10.1201/b17822.

Campbell, M. J., and S. J. Walters. 2014. *How to Design, Analyse and Report Cluster Randomised Trials in Medicine and Health Related Research.* Chichester, UK: Wiley. https://doi.org/10.1002/9781118763452.

Gallis, J. A., F. Li, H. Yu, and E. L. Turner. 2018. cvcrand and cptest: Commands for efficient design and analysis of cluster randomized trials using constrained randomization and permutation tests. *Stata Journal* 18: 357–378.

van Breukelen, G. J. P., M. J. J. M. Candel, and M. P. F. Berger. 2007. Relative efficiency of unequal versus equal cluster sizes in cluster randomized and multicentre trials. *Statistics in Medicine* 26: 2589–2603. https://doi.org/10.1002/sim.2740.

Also see

[PSS-2] **power twoproportions** — Power analysis for a two-sample proportions test

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[ME] **melogit** — Multilevel mixed-effects logistic regression

[R] **prtest** — Tests of proportions

power pairedproportions — Power analysis for a two-sample paired-proportions test

Description

`power pairedproportions` computes sample size, power, or target discordant proportions for a two-sample paired-proportions test, also known as McNemar's test. By default, it computes sample size for given power and the values of the discordant or marginal proportions. Alternatively, it can compute power for given sample size and the values of the discordant or marginal proportions, or it can compute the target discordant proportions for given sample size and power. Also see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Quick start

Using discordant proportions

Sample size for two-sided McNemar's test of marginal homogeneity H_0: $\pi_{12} = \pi_{21}$ when $p_{12} = 0.1$ and $p_{21} = 0.2$ with default power of 0.8 and significance level $\alpha = 0.05$

```
power pairedproportions .1 .2
```

Same as above, specified as $p_{12} = 0.1$ and difference between proportions $p_{21} - p_{12} = 0.1$

```
power pairedproportions .1, diff(.1)
```

Same as above, specified as $p_{12} = 0.1$ and ratio $p_{21}/p_{12} = 2$

```
power pairedproportions .1, ratio(2)
```

Same as above, but for ratios of 1.8, 1.9, 2.0, 2.1, and 2.2

```
power pairedproportions .1, ratio(1.8(.1)2.2)
```

Same as above, but display results as a graph of sample size versus ratio

```
power pairedproportions .1, ratio(1.8(.1)2.2) graph
```

For a one-sided test with $\alpha = 0.01$ and sample size of 300

```
power pairedproportions .1 .2, alpha(0.01) onesided n(300)
```

Effect size and target discordant proportions for sample size of 200, power of 0.9, and sum of discordant proportions $p_{12} + p_{21} = 0.4$

```
power pairedproportions, n(200) power(.9) prdiscordant(.4)
```

Using marginal proportions

Sample size using marginal proportions $p_{+1} = 0.6$ and $p_{1+} = 0.7$ with a correlation of 0.35 between paired observations for default power of 0.8 and $\alpha = 0.05$

```
power pairedproportions .6 .7, corr(.35)
```

Same as above, specified as $p_{+1} = 0.6$ and difference $p_{1+} - p_{+1} = 0.1$

```
power pairedproportions .6, corr(.35) diff(.1)
```

Same as above, specified as $p_{+1} = 0.6$ and relative risk $p_{1+}/p_{+1} = 1.167$

```
power pairedproportions .6, corr(.35) rrisk(1.167)
```

Same as above, specified as $p_{+1} = 0.6$ and odds ratio $\{p_{1+}/(1-p_{1+}\}/\{p_{+1}/(1-p_{+1})\} = 1.556$

```
power pairedproportions .6, corr(.35) oratio(1.556)
```

Power for a sample size of 250

```
power pairedproportions .6 .7, corr(.35) n(250)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

Specify discordant proportions

```
power pairedproportions p12 p21 [ , power(numlist) discordopts ]
```

Specify marginal proportions

```
power pairedproportions p1+ p+1 , corr(numlist) [ power(numlist) margopts ]
```

Compute power

Specify discordant proportions

```
power pairedproportions p12 p21 , n(numlist) [ discordopts ]
```

Specify marginal proportions

```
power pairedproportions p1+ p+1 , corr(numlist) n(numlist) [ margopts ]
```

Compute effect size and target discordant proportions

```
power pairedproportions, n(numlist) power(numlist) prdiscordant(numlist)
  [ discordopts ]
```

where p_{12} is the probability of a success at occasion 1 and a failure at occasion 2, and p_{21} is the probability of a failure at occasion 1 and a success at occasion 2. Each represents the probability of a discordant pair. p_{1+} is the marginal probability of a success for occasion 1, and p_{+1} is the marginal probability of a success for occasion 2. Each may be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

discordopts	Description
Main	
* alpha(*numlist*)	significance level; default is alpha(0.05)
* power(*numlist*)	power; default is power(0.8)
* beta(*numlist*)	probability of type II error; default is beta(0.2)
* n(*numlist*)	sample size; required to compute power or effect size
nfractional	allow fractional sample size
* prdiscordant(*numlist*)	sum of the discordant proportions, $p_{12} + p_{21}$
* sum(*numlist*)	synonym for prdiscordant()
* diff(*numlist*)	difference between the discordant proportions, $p_{21} - p_{12}$
* ratio(*numlist*)	ratio of the discordant proportions, p_{21}/p_{12}
effect(*effect*)	specify the type of effect to display; default is effect(diff)
direction(upper|lower)	direction of the effect for effect-size determination; default is direction(upper), which means that the postulated value of the parameter is larger than the hypothesized value
onesided	one-sided test; default is two sided
parallel	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[no]table[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
saving(*filename* [, replace])	save the table data to *filename*; use replace to overwrite existing *filename*
Graph	
graph[(*graphopts*)]	graph results; see [PSS-2] **power, graph**
Iteration	
init(#)	initial value for sample size or difference between discordant proportions
iterate(#)	maximum number of iterations; default is iterate(500)
tolerance(#)	parameter tolerance; default is tolerance(1e-12)
ftolerance(#)	function tolerance; default is ftolerance(1e-12)
[no]log	suppress or display iteration log
[no]dots	suppress or display iterations as dots
notitle	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see [U] **11.1.8 numlist**. Also see the parallel option.

notitle does not appear in the dialog box.

margopts	Description
Main	
* `alpha(`*numlist*`)`	significance level; default is `alpha(0.05)`
* `power(`*numlist*`)`	power; default is `power(0.8)`
* `beta(`*numlist*`)`	probability of type II error; default is `beta(0.2)`
* `n(`*numlist*`)`	sample size; required to compute power or effect size
`nfractional`	allow fractional sample size
* `corr(`*numlist*`)`	correlation between the paired observations
* `diff(`*numlist*`)`	difference between the marginal proportions, $p_{+1} - p_{1+}$
* `ratio(`*numlist*`)`	ratio of the marginal proportions, p_{+1}/p_{1+}
* `rrisk(`*numlist*`)`	relative risk, p_{+1}/p_{1+}
* `oratio(`*numlist*`)`	odds ratio, $\{p_{+1}(1-p_{1+})\}/\{p_{1+}(1-p_{+1})\}$
`effect(`*effect*`)`	specify the type of effect to display; default is `effect(diff)`
`direction(upper|lower)`	direction of the effect for effect-size determination; default is `direction(upper)`, which means that the postulated value of the parameter is larger than the hypothesized value
`onesided`	one-sided test; default is two sided
`parallel`	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[`no`]`table`[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
`saving(`*filename* [`, replace`]`)`	save the table data to *filename*; use `replace` to overwrite existing *filename*
Graph	
`graph`[(*graphopts*)]	graph results; see [PSS-2] **power, graph**
`notitle`	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see [U] **11.1.8 numlist**. Also see the `parallel` option.

`notitle` does not appear in the dialog box.

effect	Description
`diff`	difference between the discordant proportions, $p_{21} - p_{12}$, or marginal proportions, $p_{+1} - p_{1+}$; the default
`ratio`	ratio of the discordant proportions, p_{21}/p_{12}, or of the marginal proportions, p_{+1}/p_{1+}
`rrisk`	relative risk, p_{+1}/p_{1+}; may only be specified with marginal proportions
`oratio`	odds ratio, $\{p_{+1}(1-p_{1+})\}/\{p_{1+}(1-p_{+1})\}$; may only be specified with marginal proportions

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`N`	number of subjects	N
`delta`	effect size	δ
`p12`	success–failure proportion	p_{12}
`p21`	failure–success proportion	p_{21}
`pmarg1`	success proportion in occasion 1	p_{1+}
`pmarg2`	success proportion in occasion 2	p_{+1}
`corr`	correlation between paired observations	ρ
`prdiscordant`	proportion of discordant pairs	$p_{12}+p_{21}$
`sum`	sum of discordant proportions	$p_{12}+p_{21}$
`diff`	difference between discordant proportions	$p_{21}-p_{12}$
	difference between marginal proportions	$p_{+1}-p_{1+}$
`ratio`	ratio of discordant proportions	p_{21}/p_{12}
	ratio of marginal proportions	p_{+1}/p_{1+}
`rrisk`	relative risk for marginal proportions	p_{+1}/p_{1+}
`oratio`	odds ratio for marginal proportions	θ
`target`	target parameter; synonym for `p12`	
`_all`	display all supported columns	

Column `beta` is shown in the default table in place of column `power` if specified.

Columns `p12` and `p21` are shown in the default table if discordant proportions are specified.

Columns `pmarg1`, `pmarg2`, and `corr` are shown in the default table if marginal proportions are specified.

Columns `pmarg1`, `pmarg2`, `corr`, `rrisk`, and `oratio` are available only if marginal proportions are specified.

Columns `diff`, `ratio`, `prdiscordant`, `sum`, `rrisk`, and `oratio` are shown in the default table if specified.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

Options

Main

`alpha()`, `power()`, `beta()`, `n()`, `nfractional`; see [PSS-2] **power**. The `nfractional` option is allowed only for sample-size determination.

`prdiscordant(`*numlist*`)` specifies the proportion of discordant pairs or the sum of the discordant proportions, $p_{12}+p_{21}$. See *Alternative ways of specifying effect* for details about the specification of this option.

`sum(`*numlist*`)` is a synonym for `prdiscordant()`. See *Alternative ways of specifying effect* for details about the specification of this option.

`corr(`*numlist*`)` specifies the correlation between paired observations. This option is required if marginal proportions are specified.

`diff(`*numlist*`)` specifies the difference between the discordant proportions, $p_{21} - p_{12}$, or the marginal proportions, $p_{+1} - p_{1+}$. See *Alternative ways of specifying effect* for details about the specification of this option.

`ratio(`*numlist*`)` specifies the ratio of the discordant proportions, p_{21}/p_{12}, or the marginal proportions, p_{+1}/p_{1+}. See *Alternative ways of specifying effect* for details about the specification of this option.

`rrisk(`*numlist*`)` specifies the relative risk of the marginal proportions, p_{+1}/p_{1+}. See *Alternative ways of specifying effect* for details about the specification of this option.

`oratio(`*numlist*`)` specifies the odds ratio of the marginal proportions, $\{p_{+1}(1-p_{1+})\}/\{p_{1+}(1-p_{+1})\}$. See *Alternative ways of specifying effect* for details about the specification of this option.

`effect(`*effect*`)` specifies the type of the effect size to be reported in the output as `delta`. *effect* is one of `diff` or `ratio` for discordant proportions and one of `diff`, `ratio`, `rrisk`, or `oratio` for marginal proportions. By default, the effect size `delta` is the difference between proportions. If `diff()`, `ratio()`, `rrisk()`, or `oratio()` is specified, the effect size `delta` will contain the effect corresponding to the specified option. For example, if `ratio()` is specified, `delta` will contain the ratio of the proportions. See *Alternative ways of specifying effect* for details about the specification of this option.

`direction()`, `onesided`, `parallel`; see [PSS-2] **power**.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(`#`)` specifies the initial value for the estimated parameter. The estimated parameter is sample size for sample-size determination or the difference between the discordant proportions for the effect-size determination.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following option is available with `power pairedproportions` but is not shown in the dialog box:

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `power pairedproportions` command and the methodology for power and sample-size analysis for a two-sample paired-proportions test. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis and [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Introduction

The analysis of paired proportions is used to compare two dependent binomial populations. Dependent binomial data arise from matched case–control studies, where the cases are matched to the controls on the basis of similar demographic characteristics, or from longitudinal studies, where the same cases serve as their own controls over time or for different treatments. In all cases, each observation represents a pair of correlated binary outcomes.

There are many examples of studies where a researcher would like to compare two dependent proportions. For example, a state highway department might be concerned that tollbooth workers may experience hearing loss because of chronic exposure to traffic noise. It wants to test whether the proportion of workers with moderate to severe hearing loss is the same between a sample of workers exposed to traffic noise and a sample of workers sheltered by the quieter interior of the booth. Or a pediatrician might conduct a study to compare the proportions of males and females with a particular food allergy in a study of male/female fraternal twins.

This entry describes power and sample-size analysis for correlated binary outcomes in a two-way contingency table. Consider a 2×2 table from a study where the outcome of interest is a pair of results from "occasion 1" and "occasion 2", each either a "success" or a "failure".

	Occasion 2		
Occasion 1	Success	Failure	Total
Success	n_{11}	n_{12}	n_{1+}
Failure	n_{21}	n_{22}	n_{2+}
Total	n_{+1}	n_{+2}	n

n is the total number of pairs; n_{11} is the number of pairs for which the response is a success for both occasions; n_{12} is the number of success–failure pairs for which the response is a success on occasion 1 and a failure on occasion 2; n_{21} is the number of failure–success pairs for which the response is a failure on occasion 1 and a success on occasion 2; and n_{22} is the number of pairs for which the response is a failure for both occasions. The success–failure and failure–success pairs form discordant pairs, and the remaining pairs form concordant pairs.

The above table can also be expressed in terms of the proportions.

	Occasion 2		
Occasion 1	Success	Failure	Total
Success	p_{11}	p_{12}	p_{1+}
Failure	p_{21}	p_{22}	$1-p_{1+}$
Total	p_{+1}	$1-p_{+1}$	1

p_{1+} is the success probability for occasion 1, and p_{+1} is the success probability for occasion 2. The marginal probabilities, p_{1+} and p_{+1}, are used to compare the outcomes between occasion 1 and occasion 2.

The null hypothesis for the test of equality of marginal proportions, also known as the test of marginal homogeneity, is H_0: $p_{+1} = p_{1+}$. The null hypothesis can be formulated in terms of the discordant probabilities, the failure–success probability, p_{21}, and the success–failure probability, p_{12}, using the relationships $p_{+1} = p_{11}+p_{21}$ and $p_{1+} = p_{11}+p_{12}$. The considered null hypothesis is then H_0: $p_{21} = p_{12}$ versus the two-sided alternative hypothesis H_a: $p_{21} \neq p_{12}$, the upper one-sided alternative H_a: $p_{21} > p_{12}$, or the lower one-sided alternative H_a: $p_{21} < p_{12}$. For a 2×2 table, the test of marginal homogeneity is also called a "test of symmetry".

A large-sample McNemar's test is commonly used for testing the above hypotheses. Under the null hypothesis, the test statistic is distributed as a χ^2_1 distribution with 1 degree of freedom.

`power pairedproportions` provides power and sample-size analysis for McNemar's test of two correlated proportions.

Using power pairedproportions

`power pairedproportions` computes sample size, power, or target discordant proportions for a two-sample paired-proportions test. All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option. You can specify the `onesided` option to request a one-sided test.

For sample-size and power determinations, `power pairedproportions` provides a number of ways of specifying the magnitude of an effect desired to be detected by the test. Below we describe the use of the command, assuming that the desired effect is expressed by the values of the two discordant proportions; see *Alternative ways of specifying effect* for other specifications.

To compute sample size, you must specify the discordant proportions, p_{12} and p_{21}, and, optionally, the power of the test in option `power()`. The default power is set to 0.8.

To compute power, you must specify the sample size in option `n()` and the discordant proportions, p_{12} and p_{21}.

The effect-size determination is available only for discordant proportions. To compute effect size and target discordant proportions, you must specify the sample size in option `n()`, the power in option `power()`, the sum of the discordant proportions in option `prdiscordant()`, and, optionally, the direction of the effect. The direction is upper by default, `direction(upper)`, which means that the failure–success proportion, p_{21}, is assumed to be larger than the specified success–failure proportion, p_{12}. You can change the direction to lower, which means that p_{21} is assumed to be smaller than p_{12}, by specifying the `direction(lower)` option.

There are multiple definitions of effect size for a two-sample paired-proportions test. The `effect()` option specifies what definition `power pairedproportions` should use when reporting the effect size, which is labeled as `delta` in the output of the `power` command.

When you specify the discordant proportions, the available definitions are the difference $p_{21} - p_{12}$ between the discordant proportions, `effect(diff)`, or the ratio p_{21}/p_{12} of the discordant proportions, `effect(ratio)`.

When you specify the marginal proportions, the available definitions are the difference $p_{+1} - p_{1+}$ between the marginal proportions, `effect(diff)`; the relative risk or ratio p_{+1}/p_{1+} of the marginal proportions, `effect(rrisk)` or `effect(ratio)`; or the odds ratio $\{p_{+1}(1-p_{1+})\}/\{p_{1+}(1-p_{+1})\}$ of the marginal proportions, `effect(oratio)`.

When `effect()` is specified, the effect size `delta` in the output of the `power` command contains the estimate of the corresponding effect and is labeled accordingly. By default, `delta` corresponds to the difference between proportions. If any one of the options `diff()`, `ratio()`, `rrisk()`, or `oratio()` is specified and `effect()` is not specified, `delta` will contain the effect size corresponding to the specified option.

Some of `power pairedproportions`'s computations require iteration. For example, a sample size for a two-sided test is obtained by iteratively solving a nonlinear power equation. The default initial value for the sample size for the iteration procedure is obtained using a closed-form one-sided formula. If you desire, you may change it by specifying the `init()` option. See [PSS-2] **power** for the descriptions of other options that control the iteration procedure.

Alternative ways of specifying effect

To compute power or sample size, you must also specify the magnitude of the effect that is desired to be detected by the test. You can do this by specifying either the discordant proportions, p_{12} and p_{21},

`power pairedproportions` p_{12} p_{21} `,` ...

or the marginal proportions, p_{1+} and p_{+1}:

`power pairedproportions` p_{1+} p_{+1} `, corr(`*numlist*`)` ...

When you specify marginal proportions, you must also specify the correlation between paired observations in option `corr()`.

Below we describe other alternative specifications separately for discordant proportions and marginal proportions.

Effect specifications for discordant proportions

Instead of specifying p_{21}, you may specify the discordant proportion p_{12} as the argument to the command and the sum of the discordant proportions, $p_{12}+p_{21}$, in option `prdiscordant()` or option `sum()`,

`power pairedproportions` p_{12} `, prdiscordant(`*numlist*`)` ...

`power pairedproportions` p_{12} `, sum(`*numlist*`)` ...

the difference between the discordant proportions, $p_{21} - p_{12}$, in option `diff()`,

`power pairedproportions` p_{12} `, diff(`*numlist*`)` ...

or the ratio of the discordant proportions, p_{21}/p_{12}, in option `ratio()`:

`power pairedproportions` p_{12} `, ratio(`*numlist*`)` ...

You may omit both command arguments p_{12} and p_{21} altogether and specify options `prdiscordant()` or `sum()`, `diff()`, and `ratio()` in pairs.

For example, you can specify the sum $p_{12}+p_{21}$ and difference $p_{21}-p_{12}$ of the discordant proportions:

`power pairedproportions , prdiscordant(`*numlist*`) diff(`*numlist*`)` ...

Or you can specify the sum $p_{12}+p_{21}$ and ratio p_{21}/p_{12} of the discordant proportions:

`power pairedproportions , sum(`*numlist*`) ratio(`*numlist*`)` ...

Or you can specify the difference $p_{21}-p_{12}$ and ratio p_{21}/p_{12} of the discordant proportions:

`power pairedproportions , diff(`*numlist*`) ratio(`*numlist*`)` ...

When discordant proportions are specified, the effect size may be expressed as the difference between discordant proportions, $p_{21}-p_{12}$, or the ratio of discordant proportions, p_{21}/p_{12}. You may choose what effect to compute by specifying the `effect()` option.

By default, effect size is the difference between the discordant proportions. For example, for the specification below, the effect size δ is the difference between the discordant proportions.

`power pairedproportions` p_{12} p_{21} `,` ...

The above specification is equivalent to

`power pairedproportions` p_{12} p_{21} `, effect(diff)` ...

Alternatively, you may request the effect size to be the ratio instead of the difference.

`power pairedproportions` p_{12} p_{21} `, effect(ratio)` ...

Likewise, if you specify the `ratio()` option, the effect size is the ratio of the proportions.

Effect specifications for marginal proportions

Instead of specifying p_{+1}, you may specify the marginal proportion, p_{1+}, as the argument to the command and the difference between the marginal proportions, $p_{+1}-p_{1+}$, in option `diff()`,

`power pairedproportions` p_{1+} `, corr(`*numlist*`) diff(`*numlist*`)` ...

the ratio of the marginal proportions or relative risk, p_{+1}/p_{1+}, in option `ratio()` or option `rrisk()`,

`power pairedproportions` p_{1+} `, corr(`*numlist*`) ratio(`*numlist*`)` ...

`power pairedproportions` p_{1+} `, corr(`*numlist*`) rrisk(`*numlist*`)` ...

or the odds ratio, $\{p_{+1}(1-p_{1+})\}/\{p_{1+}(1-p_{+1})\}$, in option `oratio()`:

`power pairedproportions` p_{1+} `, corr(`*numlist*`) oratio(`*numlist*`)` ...

Alternatively, you may omit both command arguments p_{1+} and p_{+1} and specify one of the combinations of `diff()` and `rrisk()` or `ratio()`, or `oratio()` and `rrisk()` or `ratio()`. You may not combine `diff()` and `oratio()`, because marginal proportions cannot be identified uniquely from this combination.

For example, you can specify the difference $p_{+1}-p_{1+}$ and ratio p_{+1}/p_{1+} of the marginal proportions,

```
power pairedproportions , corr(numlist) diff(numlist) ratio(numlist) ...
```

or the odds ratio, $\{p_{+1}(1-p_{1+})\}/\{p_{1+}(1-p_{+1})\}$, and the relative risk, p_{+1}/p_{1+}:

```
power pairedproportions , corr(numlist) oratio(numlist) rrisk(numlist) ...
```

The effect size for marginal proportions may be defined as one of the difference between marginal proportions, $p_{+1}-p_{1+}$, the odds ratio, $\{p_{+1}(1-p_{1+})\}/\{p_{1+}(1-p_{+1})\}$, or the relative risk or, equivalently, the ratio p_{+1}/p_{1+}. The `effect()` option for marginal proportions may contain one of `diff`, `oratio`, `rrisk`, or `ratio`.

By default, effect size is defined as the difference between the marginal proportions. For example, the following specification,

```
power pairedproportions p1+ p+1 , corr(numlist) ...
```

is equivalent to

```
power pairedproportions p1+ p+1 , corr(numlist) effect(diff) ...
```

You may request other measures of effect size for marginal proportions such as the risk ratio,

```
power pairedproportions p1+ p+1 , corr(numlist) effect(rrisk) ...
```

or odds ratio:

```
power pairedproportions p1+ p+1 , corr(numlist) effect(oratio) ...
```

In the following sections, we describe the use of `power pairedproportions` accompanied by examples for computing sample size, power, and target discordant proportion.

Computing sample size

To compute sample size, you must specify the discordant proportions, p_{12} and p_{21}, and, optionally, the power of the test in option `power()`. The default power is set to 0.8. Instead of the discordant proportions, you can specify an effect of interest as shown in *Alternative ways of specifying effect*.

▷ Example 1: Sample size for a two-sample paired-proportions test

Consider a study from Agresti (2013, 413) where the same group of subjects was asked who they voted for in the 2004 and 2008 presidential elections. In 2008, we witnessed a shift from the Republican President George W. Bush, who was finishing his second term, to Democratic President Barack Obama, who was beginning his first term. Suppose that we would like to conduct another survey for the 2012 and 2016 elections to see whether a similar shift—this time from a Democrat to a Republican—would occur in 2016, when President Obama finishes his second term. We are interested in testing the hypothesis whether the proportions of votes for a Democratic president in 2012 will be the same as in 2016.

Consider the following 2×2 table:

	2016 Election		
2012 Election	Democratic	Republican	Total
Democratic	p_{11}	p_{12}	p_{1+}
Republican	p_{21}	p_{22}	$1-p_{1+}$
Total	p_{+1}	$1-p_{+1}$	1

The test of marginal homogeneity that the proportion of Democratic votes in 2012 will be the same in 2016 is given by the null hypothesis H_0: $p_{1+} = p_{+1}$. This is equivalent to testing whether the proportion p_{12} of voters who changed parties from Democratic to Republican is the same as the proportion p_{21} of voters who changed parties from Republican to Democratic between 2012 and 2016. The corresponding null hypothesis that tests these discordant proportions is H_0: $p_{12} = p_{21}$.

Suppose that the previous survey reported that the proportion of respondents who voted for a Democratic president in 2012 is $p_{1+} = 0.53$. Using this empirical evidence, a political expert believes that the odds of a candidate being elected as the president when his or her party has already served two consecutive terms as the president are low. According to the expert's opinion, the odds of the population voting Democrat in 2016 to the odds of the population voting Democrat in 2012 are 2:3; that is, the corresponding odds ratio is $\theta = 2/3 = 0.667$. Using the relationship between the odds ratio and marginal probabilities, we compute the marginal probability p_{+1} to be $= [0.667\times\{0.53/(1-0.53)\}]/(1+[(0.667\times\{0.53/(1-0.53)\}]) = 0.4293$.

It is generally believed that voting behavior is positively correlated, which means that a person voting for one party in an election year is very likely to vote for the same party in the next election year. Suppose the expert posits a correlation of 0.8.

We wish to compute the sample size required for our survey to detect the difference between the considered marginal proportions. To compute the minimum sample size, we specify the marginal proportions after the command name and the correlation in option `corr()`:

```
. power pairedproportions 0.53 0.4293, corr(0.8)

Performing iteration ...

Estimated sample size for a two-sample paired-proportions test
Large-sample McNemar's test
H0: p+1 = p1+  versus  Ha: p+1 != p1+

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =   -0.1007  (difference)
          p1+ =    0.5300
          p+1 =    0.4293
         corr =    0.8000

Estimated sample size:

            N =        82
```

We find that 82 subjects are required in our survey for a 5%-level two-sided McNemar's test to detect a change in the proportion voting Democrat from 0.53 in 2012 to 0.4293 in 2016, which corresponds to the difference of $\delta = 0.4293 - 0.53 = -0.1007$, with 80% power.

◁

▷ Example 2: Reporting odds ratio

By default, as in example 1, the effect size δ is the difference between the marginal proportions. Alternatively, we can request that the effect size be reported as the odds ratio by specifying option `effect(oratio)`.

```
. power pairedproportions 0.53 0.4293, corr(0.8) effect(oratio)
Performing iteration ...
Estimated sample size for a two-sample paired-proportions test
Large-sample McNemar's test
H0: p+1 = p1+  versus  Ha: p+1 != p1+
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.6671  (odds ratio)
          p1+ =    0.5300
          p+1 =    0.4293
         corr =    0.8000
Estimated sample size:
            N =        82
```

The effect size `delta` now contains the odds ratio estimated from the specified marginal proportions. Also see *Alternative ways of specifying effect* for other available measures of effect.

◁

▷ Example 3: Specifying odds ratio

In example 1, we computed the second marginal proportion, p_{+1}, using the postulated values of the first marginal proportion, p_{1+}, and the odds ratio, θ, and we specified the two marginal proportions with `power pairedproportions`. We can instead specify the first marginal proportion and the odds ratio directly:

```
. power pairedproportions 0.53, corr(0.8) oratio(0.667)
Performing iteration ...
Estimated sample size for a two-sample paired-proportions test
Large-sample McNemar's test
H0: p+1 = p1+  versus  Ha: p+1 != p1+
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.6670  (odds ratio)
          p1+ =    0.5300
          p+1 =    0.4293
         corr =    0.8000
   odds ratio =    0.6670
Estimated sample size:
            N =        82
```

When the `oratio()` option is specified, the reported effect size `delta` corresponds to the odds ratio, and the value of the odds ratio specified in `oratio()` is also reported in the output.

Also see *Alternative ways of specifying effect* for other ways of specifying an effect.

◁

▷ Example 4: Specifying discordant proportions

Instead of marginal proportions, as in example 1, we can specify discordant proportions. We compute discordant proportions using (1) and the estimates of marginal proportions and correlation in this study: $p_{12} = 0.53(1-0.4293) - 0.8\sqrt{0.53 \times (1-0.53) \times 0.4293 \times (1-0.4293)} = 0.105$ and $p_{21} = 0.105 + 0.4293 - 0.53 = 0.004$.

```
. power pairedproportions 0.105 0.004
Performing iteration ...
Estimated sample size for a two-sample paired-proportions test
Large-sample McNemar's test
H0: p21 = p12  versus  Ha: p21 != p12
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =   -0.1010  (difference)
          p12 =    0.1050
          p21 =    0.0040
Estimated sample size:
            N =        82
```

We obtain the same sample size of 82 as in example 1. The reported effect size `delta` corresponds to the difference of -0.1010 between the discordant proportions.

Also see *Alternative ways of specifying effect* for other ways of specifying the effect desired to be detected by the test.

◁

Computing power

To compute power, you must specify the sample size in option `n()` and the discordant proportions, p_{12} and p_{21}. Instead of the discordant proportions, you can specify an effect of interest as shown in *Alternative ways of specifying effect*.

▷ Example 5: Power of a two-sample paired-proportions test

Continuing with example 4, we will suppose that we anticipate to obtain a sample of 100 subjects and want to compute the power corresponding to this sample size.

In addition to the discordant proportions, we specify the sample size of 100 in option `n()`:

```
. power pairedproportions 0.105 0.004, n(100)
Estimated power for a two-sample paired-proportions test
Large-sample McNemar's test
H0: p21 = p12  versus  Ha: p21 != p12
Study parameters:
        alpha =    0.0500
            N =       100
        delta =   -0.1010  (difference)
          p12 =    0.1050
          p21 =    0.0040
Estimated power:
        power =    0.8759
```

As expected, with a larger sample size, this example achieves a larger power, about 88%, compared with example 4.

◁

▷ Example 6: Alternative specification of an effect for power determination

Continuing with example 5, we can specify the first discordant proportion, p_{12}, and one of the options `prdiscordant()`, `diff()`, or `ratio()` instead of specifying both discordant proportions.

For example, let's specify the sum of the discordant proportions instead of the discordant proportion p_{21}:

```
. power pairedproportions 0.105, n(100) prdiscordant(0.109)

Estimated power for a two-sample paired-proportions test
Large-sample McNemar's test
H0: p21 = p12  versus  Ha: p21 != p12

Study parameters:

        alpha =    0.0500
            N =       100
        delta =   -0.1010  (difference)
          p12 =    0.1050
          p21 =    0.0040
    p12 + p21 =    0.1090

Estimated power:

        power =    0.8759
```

We obtain results identical to those in example 5.

◁

▷ Example 7: Power determination with marginal proportions

We can compute the power corresponding to the sample of 100 subjects for the specification using marginal proportions from example 1 by additionally specifying option `n(100)`:

```
. power pairedproportions 0.53 0.4293, corr(0.8) n(100)

Estimated power for a two-sample paired-proportions test
Large-sample McNemar's test
H0: p+1 = p1+  versus  Ha: p+1 != p1+

Study parameters:

        alpha =    0.0500
            N =       100
        delta =   -0.1007  (difference)
          p1+ =    0.5300
          p+1 =    0.4293
         corr =    0.8000

Estimated power:

        power =    0.8739
```

As expected, the estimated power of 0.8739 is very close to the estimated power of 0.8759 in example 5. If we had used input values for discordant proportions with more precision in example 5, we would have obtained nearly identical results.

◁

▷ Example 8: Multiple values of study parameters

Continuing with example 7, we would like to assess the effect of varying correlation on the power of our study.

We specify a range of correlations between 0.2 and 0.8 with a step size of 0.1 in option `corr()`:

```
. power pairedproportions 0.53 0.4293, corr(0.2(0.1)0.8) n(100)
Estimated power for a two-sample paired-proportions test
Large-sample McNemar's test
H0: p+1 = p1+  versus  Ha: p+1 != p1+
```

alpha	power	N	delta	pmarg1	pmarg2	corr
.05	.3509	100	-.1007	.53	.4293	.2
.05	.3913	100	-.1007	.53	.4293	.3
.05	.4429	100	-.1007	.53	.4293	.4
.05	.5105	100	-.1007	.53	.4293	.5
.05	.6008	100	-.1007	.53	.4293	.6
.05	.7223	100	-.1007	.53	.4293	.7
.05	.8739	100	-.1007	.53	.4293	.8

For a given sample size, the power increases as the correlation increases, which means that for a given power, the required sample size decreases as the correlation increases. This demonstrates that a paired design can improve the precision of the statistical inference compared with an independent design, which corresponds to a correlation of zero.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

Computing effect size and target discordant proportions

The effect-size determination is available only for discordant proportions. As we describe in detail in *Alternative ways of specifying effect*, there are multiple definitions of the effect size for a paired-proportions test. The default is the difference between the failure–success proportion, p_{21}, and the success–failure proportion, p_{12}.

Sometimes, we may be interested in determining the smallest effect and the corresponding discordant proportions that yield a statistically significant result for prespecified sample size and power. In this case, power, sample size, and the sum of the discordant proportions must be specified. In addition, you must also decide on the direction of the effect: upper, meaning $p_{21} > p_{12}$, or lower, meaning $p_{21} < p_{12}$. The direction may be specified in the `direction()` option; `direction(upper)` is the default.

▷ Example 9: Compute effect size and target proportions

Suppose that we want to compute the corresponding discordant proportions for given sample size and power. To compute the discordant proportions, we must specify the proportion of discordant pairs (the sum of the discordant proportions) in addition to sample size and power.

Continuing with example 4, we will compute the corresponding effect size and target proportions for the sample size of 82 and the power of 0.8 using 0.109 for the proportion of discordant pairs. We also specify the `direction(lower)` option because $p_{21} < p_{12}$ in our example.

```
. power pairedproportions, prdiscordant(.109) n(82) power(0.8) direction(lower)
Performing iteration ...
Estimated discordant proportions for a two-sample paired-proportions test
Large-sample McNemar's test
H0: p21 = p12  versus  Ha: p21 != p12; p21 < p12
Study parameters:
        alpha =    0.0500
        power =    0.8000
            N =        82
    p12 + p21 =    0.1090
Estimated effect size and discordant proportions:
        delta =   -0.1007  (difference)
          p12 =    0.1048
          p21 =    0.0042
```

The estimated discordant proportions of 0.1048 and 0.0042 are very close to the respective original estimates of the discordant proportions of 0.105 and 0.004 from example 4.

◁

Testing a hypothesis about two correlated proportions

Suppose we collected data from two paired binomial samples and wish to test whether the two proportions of an outcome of interest are the same. We wish to use McNemar's test to test this hypothesis. We can use the `mcc` command to perform McNemar's test; see [R] **Epitab** for details.

▷ Example 10: Testing for paired proportions

We use data provided in table 11.1 of Agresti (2013, 414) that present the results of a General Social Survey, which asked males who they voted for in the 2004 and 2008 presidential elections.

	2008 Election		
2004 Election	Democrat	Republican	Total
Democrat	175	16	191
Republican	54	188	42
Total	229	204	433

We wish to test whether there was a change in the voting behavior of males in 2008 compared with 2004 using McNemar's test. We use `mcci`, the immediate form of `mcc`, to perform this test.

```
. mcci 175 16 54 188
```

Cases	Controls: Exposed	Controls: Unexposed	Total
Exposed	175	16	191
Unexposed	54	188	242
Total	229	204	433

```
McNemar's chi2(1) =     20.63    Prob > chi2 = 0.0000
Exact McNemar significance probability       = 0.0000

Proportion with factor
        Cases       .4411085
        Controls    .5288684     [95% conf. interval]
                    ---------    --------------------
        difference -.0877598     -.1270274  -.0484923
        ratio       .8340611      .7711619   .9020907
        rel. diff. -.1862745     -.2738252  -.0987238
        odds ratio  .2962963      .1582882   .5254949   (exact)
```

McNemar's test statistic is 20.63 with the corresponding two-sided p-value less than 10^{-4}, which provides strong evidence of a shift in the Democratic direction among male voters in 2008.

We use the estimates of this study to perform a sample-size analysis we would have conducted before a new study. The discordant proportions are $p_{12} = 16/433 = 0.037$ and $p_{21} = 54/433 = 0.125$.

```
. power pairedproportions 0.037 0.125

Performing iteration ...

Estimated sample size for a two-sample paired-proportions test
Large-sample McNemar's test
H0: p21 = p12  versus  Ha: p21 != p12

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.0880  (difference)
          p12 =    0.0370
          p21 =    0.1250

Estimated sample size:

            N =       162
```

We find that we need a sample of 162 respondents to detect a difference of 0.0880 between discordant proportions of 0.037 and 0.125 with 80% power using a 5%-level two-sided test.

◁

Stored results

power pairedproportions stores the following in r():

Scalars

r(alpha)	significance level
r(power)	power
r(beta)	probability of a type II error
r(delta)	effect size
r(N)	sample size
r(nfractional)	1 if nfractional is specified, 0 otherwise
r(onesided)	1 for a one-sided test, 0 otherwise
r(p12)	success–failure proportion (first discordant proportion)
r(p21)	failure–success proportion (second discordant proportion)
r(pmarg1)	success proportion for occasion 1 (first marginal proportion)
r(pmarg2)	success proportion for occasion 2 (second marginal proportion)
r(corr)	correlation between paired observations
r(diff)	difference between proportions
r(ratio)	ratio of proportions
r(prdiscordant)	proportion of discordant pairs
r(sum)	sum of discordant proportions
r(rrisk)	relative risk
r(oratio)	odds ratio
r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise
r(init)	initial value for sample size or difference between discordant proportions
r(maxiter)	maximum number of iterations
r(iter)	number of iterations performed
r(tolerance)	requested parameter tolerance
r(deltax)	final parameter tolerance achieved
r(ftolerance)	requested distance of the objective function from zero
r(function)	final distance of the objective function from zero
r(converged)	1 if iteration algorithm converged, 0 otherwise

Macros

r(effect)	diff, ratio, oratio, or rrisk
r(type)	test
r(method)	pairedproportions
r(direction)	upper or lower
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results

Methods and formulas

Consider a 2×2 contingency table formed with n pairs of observations. The first subscript $i = 1, 2$ denotes the success of failure in occasion 1, and the second subscript $j = 1, 2$ denotes the success of failure in occasion 2.

	Occasion 2		
Occasion 1	Success	Failure	Total
Success	p_{11}	p_{12}	p_{1+}
Failure	p_{21}	p_{22}	$1 - p_{1+}$
Total	p_{+1}	$1 - p_{+1}$	1

Each element in the above table denotes the probability of observing a specific pair. For example, p_{11} is the probability of jointly observing a success on occasion 1 and occasion 2, and p_{12} is the probability of observing a success on occasion 1 and a failure on occasion 2. p_{1+} is the marginal probability of a success on occasion 1, and p_{+1} is the marginal probability of a success on occasion 2. The off-diagonal proportions p_{12} and p_{21} are referred to as "discordant proportions". The relationship between the discordant proportions and the marginal proportions is given by

$$\begin{aligned} p_{12} &= p_{1+}(1-p_{+1}) - \rho\sqrt{p_{1+}(1-p_{1+})\,p_{+1}(1-p_{+1})} \\ p_{21} &= p_{12} + p_{+1} - p_{1+} \end{aligned} \tag{1}$$

where ρ is the correlation between the paired observations.

A two-sample paired proportions test involves testing the null hypothesis H_0: $p_{+1} = p_{1+}$ versus the two-sided alternative hypothesis H_a: $p_{+1} \neq p_{1+}$, the upper one-sided alternative H_a: $p_{+1} > p_{1+}$, or the lower one-sided alternative H_a: $p_{+1} < p_{1+}$. Using the relationship $p_{1+} = p_{11}+p_{12}$ and $p_{+1} = p_{11}+p_{21}$, test hypotheses may be stated in terms of the discordant proportions, for example, H_0: $p_{21} = p_{12}$ versus H_a: $p_{21} \neq p_{12}$.

McNemar's test statistic is:

$$\chi^2 = (n_{12} - n_{21})^2/(n_{12} + n_{21})$$

where n_{ij} is the number of successes ($i = 1$) or failures ($i = 2$) on occasion 1 and the number of successes ($j = 1$) or failures ($j = 2$) on occasion 2; see Lachin (2011, chap. 5) for details. This test statistic has an approximately χ^2 distribution with 1 degree of freedom under the null hypothesis. The square root of the χ^2 test statistic is approximately normal with zero mean and variance of one.

Let α be the significance level, β be the probability of a type II error, and $z_{1-\alpha/k}$ and z_β be the $(1-\alpha/k)$th and the βth quantiles of the standard normal distribution.

The power $\pi = 1 - \beta$ is computed using

$$\pi = \begin{cases} \Phi\left\{\dfrac{p_{\text{diff}}\sqrt{n} - z_{1-\alpha}\sqrt{p_{\text{disc}}}}{\sqrt{p_{\text{disc}} - p_{\text{diff}}^2}}\right\} & \text{for an upper one-sided test} \\ \Phi\left\{\dfrac{-p_{\text{diff}}\sqrt{n} - z_{1-\alpha}\sqrt{p_{\text{disc}}}}{\sqrt{p_{\text{disc}} - p_{\text{diff}}^2}}\right\} & \text{for a lower one-sided test} \\ \Phi\left\{\dfrac{p_{\text{diff}}\sqrt{n} - z_{1-\alpha/2}\sqrt{p_{\text{disc}}}}{\sqrt{p_{\text{disc}} - p_{\text{diff}}^2}}\right\} + \Phi\left\{\dfrac{-p_{\text{diff}}\sqrt{n} - z_{1-\alpha/2}\sqrt{p_{\text{disc}}}}{\sqrt{p_{\text{disc}} - p_{\text{diff}}^2}}\right\} & \text{for a two-sided test} \end{cases} \tag{2}$$

where $\Phi(\cdot)$ is the cdf of a standard normal distribution, $p_{\text{diff}} = p_{21} - p_{12}$, and $p_{\text{disc}} = p_{12} + p_{21}$.

The sample size n for a one-sided test is computed using

$$n = \left(\frac{z_\alpha\sqrt{p_{\text{disc}}} + z_{1-\beta}\sqrt{p_{\text{disc}} - p_{\text{diff}}^2}}{p_{\text{diff}}}\right)^2 \tag{3}$$

See Connor (1987) for details.

For a two-sided test, sample size is computed iteratively from the two-sided power equation in (2). The default initial value is obtained from the corresponding one-sided formula (3) with the significance level $\alpha/2$.

The effect size $\delta = p_{21} - p_{12}$ is computed iteratively from the corresponding power equation in (2). The default initial value is $p_{\text{disc}}/2$.

References

Agresti, A. 2013. *Categorical Data Analysis*. 3rd ed. Hoboken, NJ: Wiley.

Connor, R. J. 1987. Sample size for testing differences in proportions for the paired-sample design. *Biometrics* 43: 207–211. https://doi.org/10.2307/2531961.

Lachin, J. M. 2011. *Biostatistical Methods: The Assessment of Relative Risks*. 2nd ed. Hoboken, NJ: Wiley.

McNemar, Q. 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika* 12: 153–157. https://doi.org/10.1007/BF02295996.

Schork, M. A., and G. W. Williams. 1980. Number of observations required for the comparison of two correlated proportions. *Communications in Statistics—Simulation and Computation* 9: 349–357. https://doi.org/10.1080/03610918008812161.

Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[R] **Epitab** — Tables for epidemiologists

power onevariance — Power analysis for a one-sample variance test

Description

`power onevariance` computes sample size, power, or target variance for a one-sample variance test. By default, it computes sample size for given power and the values of the variance parameters under the null and alternative hypotheses. Alternatively, it can compute power for given sample size and values of the null and alternative variances or the target variance for given sample size, power, and the null variance. The results can also be obtained for an equivalent standard deviation test, in which case standard deviations are used instead of variances. Also see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

For precision and sample-size analysis for a CI for a population variance, see [PSS-3] **ciwidth onevariance**.

Quick start

Sample size for test of H_0: $\sigma^2 = 4$ versus H_a: $\sigma^2 \neq 4$ with null variance $v_0 = 4$ and alternative variance $v_a = 9$ with default power of 0.8 and significance level $\alpha = 0.05$

```
power onevariance 4 9
```

Same as above, but for H_0: $\sigma = 2$ versus H_a: $\sigma \neq 2$ with null standard deviation $s_0 = 2$ and alternative standard deviation $s_a = 3$

```
power onevariance 2 3, sd
```

Same as above, but for $\alpha = 0.1$

```
power onevariance 2 3, sd alpha(0.1)
```

Sample sizes for v_a equal to 7, 8, 9, 10, and 11

```
power onevariance 4 (7(1)11)
```

Same as above, but display results in a graph showing sample size versus alternative variance

```
power onevariance 4 (7(1)11), graph
```

Specify v_0 and the ratio of variances

```
power onevariance 4, ratio(2.25)
```

Power for a sample size of 30

```
power onevariance 4 9, n(30)
```

Same as above, but specify standard deviations rather than variances

```
power onevariance 2 3, n(30) sd
```

Same as above, but specify a one-sided test

```
power onevariance 2 3, sd n(30) onesided
```

Effect size and target variance for $v_0 = 4$ with a sample size of 20 and power of 0.8

```
power onevariance 4, n(20) power(.8)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

Variance scale

```
power onevariance v0 va [ , power(numlist) options ]
```

Standard deviation scale

```
power onevariance s0 sa , sd [power(numlist) options ]
```

Compute power

Variance scale

```
power onevariance v0 va , n(numlist) [options ]
```

Standard deviation scale

```
power onevariance s0 sa , sd n(numlist) [options ]
```

Compute effect size and target parameter

Target variance

```
power onevariance v0 , n(numlist) power(numlist) [options ]
```

Target standard deviation

```
power onevariance s0 , sd n(numlist) power(numlist) [options ]
```

where v_0 and s_0 are the null (hypothesized) variance and standard deviation or the value of the variance and standard deviation under the null hypothesis, and v_a and s_a are the alternative (target) variance and standard deviation or the value of the variance and standard deviation under the alternative hypothesis. Each argument may be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

options	Description
`sd`	request computation using the standard deviation scale; default is the variance scale
Main	
* `alpha(`*numlist*`)`	significance level; default is `alpha(0.05)`
* `power(`*numlist*`)`	power; default is `power(0.8)`
* `beta(`*numlist*`)`	probability of type II error; default is `beta(0.2)`
* `n(`*numlist*`)`	sample size; required to compute power or effect size
`nfractional`	allow fractional sample size
* `ratio(`*numlist*`)`	ratio of variances, v_a/v_0 (or ratio of standard deviations, s_a/s_0, if option `sd` is specified); specify instead of the alternative variance v_a (or standard deviation s_a)
`direction(upper|lower)`	direction of the effect for effect-size determination; default is `direction(upper)`, which means that the postulated value of the parameter is larger than the hypothesized value
`onesided`	one-sided test; default is two sided
`parallel`	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[`no`]`table`[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
`saving(`*filename* [`, replace`]`)`	save the table data to *filename*; use `replace` to overwrite existing *filename*
Graph	
`graph`[(*graphopts*)]	graph results; see [PSS-2] **power, graph**
Iteration	
`init(#)`	initial value for sample size or variance
`iterate(#)`	maximum number of iterations; default is `iterate(500)`
`tolerance(#)`	parameter tolerance; default is `tolerance(1e-12)`
`ftolerance(#)`	function tolerance; default is `ftolerance(1e-12)`
[`no`]`log`	suppress or display iteration log
[`no`]`dots`	suppress or display iterations as dots
`notitle`	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the `parallel` option.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`sd` does not appear in the dialog box; specification of `sd` is done automatically by the dialog box selected.

`notitle` does not appear in the dialog box.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`N`	number of subjects	N
`delta`	effect size	δ
`v0`	null variance	σ_0^2
`va`	alternative variance	σ_a^2
`s0`	null standard deviation	σ_0
`sa`	alternative standard deviation	σ_a
`ratio`	ratio of the alternative variance to the null variance	σ_a^2/σ_0^2
	or ratio of the alternative standard deviation to the null standard deviation (if `sd` is specified)	σ_a/σ_0
`target`	target parameter; synonym for `va`	
`_all`	display all supported columns	

Column `beta` is shown in the default table in place of column `power` if specified.

Columns `s0` and `sa` are displayed in the default table in place of the `v0` and `va` columns when the `sd` option is specified.

Column `ratio` is shown in the default table if specified. If the `sd` option is specified, this column contains the ratio of standard deviations. Otherwise, this column contains the ratio of variances.

Options

`sd` specifies that the computation be performed using the standard deviation scale. The default is to use the variance scale.

Main

`alpha()`, `power()`, `beta()`, `n()`, `nfractional`; see [PSS-2] **power**. The `nfractional` option is allowed only for sample-size determination.

`ratio(`*numlist*`)` specifies the ratio of the alternative variance to the null variance, v_a/v_0, or the ratio of standard deviations, s_a/s_0, if the `sd` option is specified. You can specify either the alternative variance v_a as a command argument or the ratio of the variances in `ratio()`. If you specify `ratio(#)`, the alternative variance is computed as $v_a = v_0 \times \#$. This option is not allowed with the effect-size determination.

`direction()`, `onesided`, `parallel`; see [PSS-2] **power**.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies an initial value for the iteration procedure. Iteration is used to compute variance for a two-sided test and to compute sample size. The default initial value for the sample size is obtained from a closed-form normal approximation. The default initial value for the variance is obtained from a closed-form solution for a one-sided test with the significance level of $\alpha/2$.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following option is available with `power onevariance` but is not shown in the dialog box:

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `power onevariance` command and the methodology for power and sample-size analysis for a one-sample variance test. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis and [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Introduction

The study of variance arises in cases where investigators are interested in making an inference on the variability of a process. For example, the precision of a thermometer in taking accurate measurements, the variation in the weights of potato chips from one bag to another, the variation in mileage across automobiles of the same model. Before undertaking the actual study, we may want to find the optimal sample size to detect variations that exceed the tolerable limits or industry-wide standards.

This entry describes power and sample-size analysis for the inference about the population variance performed using hypothesis testing. Specifically, we consider the null hypothesis H_0: $\sigma^2 = \sigma_0^2$ versus the two-sided alternative hypothesis H_a: $\sigma^2 \neq \sigma_0^2$, the upper one-sided alternative H_a: $\sigma^2 > \sigma_0^2$, or the lower one-sided alternative H_a: $\sigma^2 < \sigma_0^2$.

Hypothesis testing of variances relies on the assumption of normality of the data. For a random sample of size n from a normal distribution, the distribution of the sample variance s^2 is scaled χ^2. The χ^2 test statistic $(n-1)s^2/\sigma_0^2$, which has a χ^2 distribution χ^2_{n-1} with $n-1$ degrees of freedom, is used to test hypotheses on variance, and the corresponding test is known as a χ^2 test.

The test of a variance is equivalent to the test of a standard deviation with the null hypothesis H_0: $\sigma = \sigma_0$. The standard deviation test uses the same χ^2 test statistic. The only difference between the two tests is the scale or metric of the variability parameter: variance for the variance test and standard deviation for the standard deviation test. In some cases, standard deviations may provide a more meaningful interpretation than variances. For example, standard deviations of test scores or IQ have the same scale as the mean and provide information about the spread of the observations around the mean.

The `power onevariance` command provides power and sample-size analysis for the χ^2 test of a one-sample variance or a one-sample standard deviation.

Using power onevariance

`power onevariance` computes sample size, power, or target variance for a one-sample variance test. If the `sd` option is specified, `power onevariance` computes sample size, power, or target standard deviation for an equivalent one-sample standard-deviation test. All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option. You can specify the `onesided` option to request a one-sided test.

In what follows, we describe the use of `power onevariance` in a variance metric. The corresponding use in a standard deviation metric, when the `sd` option is specified, is the same except variances v_0 and v_a should be replaced with the respective standard deviations s_0 and s_a. Note that computations using the variance and standard deviation scales yield the same results; the difference is only in the specification of the parameters.

To compute sample size, you must specify the variances under the null and alternative hypotheses, v_0 and v_a, respectively, and, optionally, the power of the test in the `power()` option. A default power of 0.8 is assumed if `power()` is not specified.

To compute power, you must specify the sample size in the `n()` option and the variances under the null and alternative hypotheses as arguments v_0 and v_a, respectively.

Instead of the null and alternative variances v_0 and v_a, you can specify the null variance v_0 and the ratio of the alternative variance to the null variance in the `ratio()` option.

To compute effect size, the ratio of the alternative to the null variances, and target variance, you must specify the sample size in the `n()` option, the power in the `power()` option, the null variance v_0, and, optionally, the direction of the effect. The direction is upper by default, `direction(upper)`, which means that the target variance is assumed to be larger than the specified null value. You can change the direction to lower, which means that the target variance is assumed to be smaller than the specified null value, by specifying the `direction(lower)` option.

By default, the computed sample size is rounded up. You can specify the `nfractional` option to see the corresponding fractional sample size; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. The `nfractional` option is allowed only for sample-size determination.

The test statistic for a one-sample variance test follows a χ^2 distribution. Its degrees of freedom depends on the sample size; therefore, sample-size computations require iteration. The effect-size determination for a two-sided test also requires iteration. The default initial value of the sample size is obtained using a closed-form normal approximation. The default initial value of the variance for the effect-size determination is obtained by using the corresponding computation for a one-sided test with the significance level $\alpha/2$. The default initial values may be changed by specifying the `init()` option. See [PSS-2] **power** for the descriptions of other options that control the iteration procedure.

In the following sections, we describe the use of `power onevariance` accompanied by examples for computing sample size, power, and target variance.

Computing sample size

To compute sample size, you must specify the variances under the null and alternative hypotheses, v_0 and v_a, respectively, and, optionally, the power of the test in the `power()` option. A default power of 0.8 is assumed if `power()` is not specified.

▷ Example 1: Sample size for a one-sample variance test

Consider a study where interest lies in testing whether the variability in mileage (measured in miles per gallon) of automobiles of a certain car manufacturer equals a specified value. Industry-wide standards maintain that a variation of at most two miles per gallon (mpg) from an average value is acceptable for commercial production.

The process engineer suspects that a faulty assembly line has been producing the variation higher than the acceptable standard. He or she wishes to test the null hypothesis of H_0: $\sigma = 2$ versus a two-sided alternative H_a: $\sigma \neq 2$ or, equivalently, H_0: $\sigma^2 = 4$ versus H_0: $\sigma^2 \neq 4$. The engineer wants to find the minimum number of cars so that the 5%-level two-sided test achieves the power of 80% to detect the alternative variance of 9 (or standard deviation of 3 mpg) given the null variance of 4 (or standard deviation of 2 mpg). To obtain the sample size, we specify the null and alternative values of the variance in v_0 and v_a after the command name:

```
. power onevariance 4 9

Performing iteration ...

Estimated sample size for a one-sample variance test
Chi-squared test
H0: v = v0  versus  Ha: v != v0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    2.2500
           v0 =    4.0000
           va =    9.0000

Estimated sample size:

            N =        24
```

We find that a sample of 24 subjects is required for this study.

As we mentioned in the previous section, sample-size computation requires iteration. By default, `power onevariance` suppresses the iteration log, which may be displayed by specifying the `log` option.

◁

▷ Example 2: Specifying ratio of variances

Instead of the alternative variance as in example 1, we can specify the ratio of the alternative and null variances of $9/4 = 2.25$ in the `ratio()` option:

```
. power onevariance 4, ratio(2.25)
Performing iteration ...
Estimated sample size for a one-sample variance test
Chi-squared test
H0: v = v0  versus  Ha: v != v0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    2.2500
           v0 =    4.0000
           va =    9.0000
        ratio =    2.2500
Estimated sample size:
            N =        24
```

We obtain the same results as in example 1. The `ratio` of the variances is now also displayed in the output.

◁

▷ Example 3: Standard deviation test

We can use the `sd` option to perform calculations in the standard deviation metric. We reproduce results from example 1 using the corresponding null and standard deviations of 2 and 3.

```
. power onevariance 2 3, sd
Performing iteration ...
Estimated sample size for a one-sample standard-deviation test
Chi-squared test
H0: s = s0  versus  Ha: s != s0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    1.5000
           s0 =    2.0000
           sa =    3.0000
Estimated sample size:
            N =        24
```

The results are the same, except the output reports standard deviations instead of variances.

◁

Computing power

To compute power, you must specify the sample size in the `n()` option and the variances under the null and alternative hypotheses, v_0 and v_a, respectively.

Example 4: Power of a one-sample variance test

Continuing with example 1, we will suppose that we are designing a new study and anticipate obtaining a sample of 30 cars. To compute the power corresponding to this sample size given the study parameters from example 1, we specify the sample size of 30 in the `n()` option:

```
. power onevariance 4 9, n(30)

Estimated power for a one-sample variance test
Chi-squared test
H0: v = v0  versus  Ha: v != v0

Study parameters:

        alpha =    0.0500
            N =        30
        delta =    2.2500
           v0 =    4.0000
           va =    9.0000

Estimated power:

        power =    0.8827
```

With a larger sample size, the power of the test increases to about 88%.

Example 5: Multiple values of study parameters

Suppose we would like to assess the effect of increasing the alternative variance on the power of the test. We do this by specifying a range of values in parentheses in the argument for the alternative variance:

```
. power onevariance 4 (4.5(0.5)10), n(30)

Estimated power for a one-sample variance test
Chi-squared test
H0: v = v0  versus  Ha: v != v0
```

alpha	power	N	delta	v0	va
.05	.08402	30	1.125	4	4.5
.05	.1615	30	1.25	4	5
.05	.2694	30	1.375	4	5.5
.05	.391	30	1.5	4	6
.05	.511	30	1.625	4	6.5
.05	.6189	30	1.75	4	7
.05	.7098	30	1.875	4	7.5
.05	.7829	30	2	4	8
.05	.8397	30	2.125	4	8.5
.05	.8827	30	2.25	4	9
.05	.9147	30	2.375	4	9.5
.05	.9382	30	2.5	4	10

The power is an increasing function of the effect size, which is measured by the ratio of the alternative variance to the null variance.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

Computing effect size and target variance

Effect size δ for a one-sample variance test is defined as the ratio of the alternative variance to the null variance $\delta = v_a/v_0$ or the ratio of the alternative standard deviation to the null standard deviation $\delta = s_a/s_0$ when the `sd` option is specified.

Sometimes, we may be interested in determining the smallest effect that yields a statistically significant result for prespecified sample size and power. In this case, power, sample size, and the null variance or the null standard deviation must be specified. In addition, you must also decide on the direction of the effect: upper, meaning $v_a > v_0$ ($s_a > s_0$), or lower, meaning $v_a < v_0$ ($s_a < s_0$). The direction may be specified in the `direction()` option; `direction(upper)` is the default.

▷ Example 6: Minimum detectable value of the variance

Continuing with example 4, we may also be interested to find the minimum effect size that can be detected with a power of 80% given a sample of 30 subjects. To compute the smallest effect size and the corresponding target variance, after the command name, we specify the null variance of 4, sample size `n(30)`, and power `power(0.8)`:

```
. power onevariance 4, n(30) power(0.8)

Performing iteration ...

Estimated target variance for a one-sample variance test
Chi-squared test
H0: v = v0  versus  Ha: v != v0; va > v0

Study parameters:

        alpha =    0.0500
        power =    0.8000
            N =        30
           v0 =    4.0000

Estimated effect size and target variance:

        delta =    2.0343
           va =    8.1371
```

The smallest detectable value of the effect size, the ratio of the variances, is 2.03, which corresponds to the alternative variance of 8.14. Compared with example 1, for the same power of 80%, this example shows a smaller variance with a larger sample of 30 subjects.

Above we assumed the effect to be in the upper direction. The effect size and target variance in the lower direction can be obtained by specifying `direction(lower)`.

```
. power onevariance 4, n(30) power(0.8) direction(lower)
Performing iteration ...
Estimated target variance for a one-sample variance test
Chi-squared test
H0: v = v0  versus  Ha: v != v0; va < v0
Study parameters:
        alpha =    0.0500
        power =    0.8000
            N =        30
           v0 =    4.0000
Estimated effect size and target variance:
        delta =    0.4567
           va =    1.8267
```

The smallest detectable value of the effect size is 0.46, which corresponds to the alternative variance of 1.83.

◁

Performing a hypothesis test on variance

In this section, we demonstrate the use of the `sdtest` command for testing hypotheses about variances; see [R] **sdtest** for details. Suppose we wish to test the hypothesis that the variance or standard deviation is different from a specific null value on the collected data. We can use the `sdtest` command to do this.

▷ Example 7: Testing for variance

We use `auto.dta` to demonstrate the use of `sdtest`. We have data on mileage ratings of 74 automobiles and wish to test whether the overall standard deviation is different from 3 miles per gallon (mpg).

```
. use https://www.stata-press.com/data/r19/auto
(1978 automobile data)
. sdtest mpg == 3
One-sample test of variance
```

Variable	Obs	Mean	Std. err.	Std. dev.	[95% conf.	interval]
mpg	74	21.2973	.6725511	5.785503	19.9569	22.63769

```
    sd = sd(mpg)                                     c = chi2 = 271.4955
H0: sd = 3                         Degrees of freedom =       73
     Ha: sd < 3                 Ha: sd != 3                 Ha: sd > 3
  Pr(C < c) = 1.0000         2*Pr(C > c) = 0.0000         Pr(C > c) = 0.0000
```

We find statistical evidence to reject the null hypothesis of H_0: $\sigma_{\text{mpg}} = 3$ versus a two-sided alternative H_a: $\sigma_{\text{mpg}} \neq 3$ at the 5% significance level; the p-value < 0.0000.

We use the estimates of this study to perform a sample-size analysis we would have conducted before the study.

```
. power onevar 3 5.78, sd
Performing iteration ...
Estimated sample size for a one-sample standard-deviation test
Chi-squared test
H0: s = s0  versus  Ha: s != s0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    1.9267
           s0 =    3.0000
           sa =    5.7800
Estimated sample size:
            N =        10
```

We find that the sample size required to detect a standard deviation of 5.78 mpg given the null value of 3 mpg with 80% power using a 5%-level two-sided test is only 10.

◁

Stored results

`power onevariance` stores the following in `r()`:

Scalars

`r(alpha)`	significance level
`r(power)`	power
`r(beta)`	probability of a type II error
`r(delta)`	effect size
`r(N)`	sample size
`r(nfractional)`	1 if `nfractional` is specified, 0 otherwise
`r(onesided)`	1 for a one-sided test, 0 otherwise
`r(v0)`	variance under the null hypothesis (for variance scale, default)
`r(va)`	variance under the alternative hypothesis (for variance scale, default)
`r(s0)`	standard deviation under the null hypothesis (if option `sd` is specified)
`r(sa)`	standard deviation under the alternative hypothesis (if option `sd` is specified)
`r(ratio)`	ratio of the alternative variance to the null variance (or the ratio of standard deviations if option `sd` is specified)
`r(separator)`	number of lines between separator lines in the table
`r(divider)`	1 if `divider` is requested in the table, 0 otherwise
`r(init)`	initial value for sample size or variance
`r(maxiter)`	maximum number of iterations
`r(iter)`	number of iterations performed
`r(tolerance)`	requested parameter tolerance
`r(deltax)`	final parameter tolerance achieved
`r(ftolerance)`	requested distance of the objective function from zero
`r(function)`	final distance of the objective function from zero
`r(converged)`	1 if iteration algorithm converged, 0 otherwise

Macros

`r(type)`	`test`
`r(method)`	`onevariance`
`r(direction)`	`upper` or `lower`
`r(columns)`	displayed table columns
`r(labels)`	table column labels
`r(widths)`	table column widths
`r(formats)`	table column formats
`r(scale)`	`variance` or `standard deviation`

Matrices

`r(pss_table)`	table of results

Methods and formulas

Consider a random sample of size n from a normal population with mean μ and variance σ^2. Let σ_0^2 and σ_a^2 denote the null and alternative values of the variance parameter, respectively.

A one-sample variance test involves testing the null hypothesis H_0: $\sigma^2 = \sigma_0^2$ versus the two-sided alternative hypothesis H_a: $\sigma^2 \neq \sigma_0^2$, the upper one-sided alternative H_a: $\sigma^2 > \sigma_0^2$, or the lower one-sided alternative H_a: $\sigma^2 < \sigma_0^2$.

The sampling distribution of the test statistic $\chi^2 = (n-1)s^2/\sigma^2$ under the null hypothesis follows a χ^2 distribution with $n-1$ degrees of freedom, where s^2 is the sample variance. The corresponding test is known as a χ^2 test.

The following formulas are based on Dixon and Massey (1983, 110–112).

Let α be the significance level, β be the probability of a type II error, and $\chi^2_{n-1,1-\alpha}$ and $\chi^2_{n-1,\beta}$ be the $(1-\alpha)$th and the βth quantiles of the χ^2 distribution with $n-1$ degrees of freedom.

The following equality holds at the critical value of the accept/reject boundary for H_0:

$$\frac{\chi^2_{n-1,1-\alpha}}{n-1}\sigma_0^2 = \frac{\chi^2_{n-1,\beta}}{n-1}\sigma_a^2$$

The power $\pi = 1-\beta$ is computed using

$$\pi = \begin{cases} 1 - \chi^2_{n-1}\left(\frac{\sigma_0^2}{\sigma_a^2}\chi^2_{n-1,1-\alpha}\right) & \text{for an upper one-sided test} \\ \chi^2_{n-1}\left(\frac{\sigma_0^2}{\sigma_a^2}\chi^2_{n-1,\alpha}\right) & \text{for a lower one-sided test} \\ 1 - \chi^2_{n-1}\left(\frac{\sigma_0^2}{\sigma_a^2}\chi^2_{n-1,1-\alpha/2}\right) + \chi^2_{n-1}\left(\frac{\sigma_0^2}{\sigma_a^2}\chi^2_{n-1,\alpha/2}\right) & \text{for a two-sided test} \end{cases} \quad (1)$$

where $\chi^2_{n-1}(\cdot)$ is the cdf of a χ^2 distribution with $n-1$ degrees of freedom.

Sample size n is obtained by iteratively solving the corresponding power equation from (1) for n. The default initial value for the sample size is obtained by using a large-sample normal approximation.

For a large n, the log-transformed sample variance is approximately normal with mean $2\ln(\sigma)$ and standard deviation $\sqrt{2/n}$. The approximate sample size is then given by

$$n = \frac{1}{2}\left\{\frac{z_{1-\alpha/k} - z_\beta}{\ln\left(\frac{\sigma_a}{\sigma_0}\right)}\right\}^2$$

where $k = 1$ for a one-sided test and $k = 2$ for a two-sided test.

For a one-sided test, the minimum detectable value of the variance is computed as follows:

$$\sigma_a^2 = \begin{cases} \sigma_0^2 \frac{\chi^2_{n-1,1-\alpha}}{\chi^2_{n-1,\beta}} & \text{for an upper one-sided test} \\ \sigma_0^2 \frac{\chi^2_{n-1,\alpha}}{\chi^2_{n-1,1-\beta}} & \text{for a lower one-sided test} \end{cases} \quad (2)$$

For a two-sided test, the minimum detectable value of the variance is computed by iteratively solving the two-sided power equation from (1) for σ_a^2. The default initial value is obtained from (2) with α replaced by $\alpha/2$.

If the `nfractional` option is not specified, the computed sample size is rounded up.

Reference

Dixon, W. J., and F. J. Massey, Jr. 1983. *Introduction to Statistical Analysis*. 4th ed. New York: McGraw–Hill.

Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-3] **ciwidth onevariance** — Precision analysis for a one-variance CI

[PSS-5] **Glossary**

[R] **sdtest** — Variance-comparison tests

power twovariances — Power analysis for a two-sample variances test

Description

`power twovariances` computes sample size, power, or the experimental-group variance (or standard deviation) for a two-sample variances test. By default, it computes sample size for given power and the values of the control-group and experimental-group variances. Alternatively, it can compute power for given sample size and values of the control-group and experimental-group variances or the experimental-group variance for given sample size, power, and the control-group variance. Also see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Quick start

Sample size for a test of $H_0\colon \sigma_1^2 = \sigma_2^2$ versus $H_a\colon \sigma_1^2 \neq \sigma_2^2$ with control-group variance $v_1 = 25$, experimental-group variance $v_2 = 36$, default power of 0.8, and significance level $\alpha = 0.05$

```
power twovariances 25 36
```

Same as above, but specified as standard deviations $s_1 = 5$ and $s_2 = 6$

```
power twovariances 5 6, sd
```

Sample size for $v_1 = 25$ and v_2 equals to 36, 38, 40, and 42

```
power twovariances 25 (36(2)42)
```

Same as above, but display results in a graph of sample size versus v_2

```
power twovariances 25 (36(2)42), graph
```

Save results to the dataset `mydata.dta`

```
power twovariances 25 (36(2)42), saving(mydata)
```

Power for a total sample size of 300

```
power twovariances 25 36, n(300)
```

Same as above, but specify sample sizes of 200 and 100 for groups 1 and 2, respectively

```
power twovariances 25 36, n1(200) n2(100)
```

Effect size and experimental-group standard deviation given control-group standard deviation of 5, sample size of 200, and power of 0.8

```
power twovariances 5, sd n(200) power(0.8)
```

Same as above, but calculate experimental-group variance given control-group variance of 25

```
power twovariances 25, n(200) power(0.8)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

Variance scale

```
power twovariances v1 v2 [ , power(numlist) options ]
```

Standard deviation scale

```
power twovariances s1 s2 , sd [power(numlist) options]
```

Compute power

Variance scale

```
power twovariances v1 v2 , n(numlist) [options]
```

Standard deviation scale

```
power twovariances s1 s2 , sd n(numlist) [options]
```

Compute effect size and target parameter

Experimental-group variance

```
power twovariances v1 , n(numlist) power(numlist) [options]
```

Experimental-group standard deviation

```
power twovariances s1 , sd n(numlist) power(numlist) [options]
```

where v_1 and s_1 are the variance and standard deviation, respectively, of the control (reference) group and v_2 and s_2 are the variance and standard deviation of the experimental (comparison) group. Each argument may be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

options	Description
`sd`	request computation using the standard deviation scale; default is the variance scale
Main	
* `alpha(`*numlist*`)`	significance level; default is `alpha(0.05)`
* `power(`*numlist*`)`	power; default is `power(0.8)`
* `beta(`*numlist*`)`	probability of type II error; default is `beta(0.2)`
* `n(`*numlist*`)`	total sample size; required to compute power or effect size
* `n1(`*numlist*`)`	sample size of the control group
* `n2(`*numlist*`)`	sample size of the experimental group
* `nratio(`*numlist*`)`	ratio of sample sizes, `N2/N1`; default is `nratio(1)`, meaning equal group sizes
`compute(N1 | N2)`	solve for `N1` given `N2` or for `N2` given `N1`
`nfractional`	allow fractional sample sizes
* `ratio(`*numlist*`)`	ratio of variances, v_2/v_1 (or ratio of standard deviations, s_2/s_1, if option `sd` is specified); specify instead of the experimental-group variance v_2 (or standard deviation s_2)
`direction(upper|lower)`	direction of the effect for effect-size determination; default is `direction(upper)`, which means that the postulated value of the parameter is larger than the hypothesized value
`onesided`	one-sided test; default is two sided
`parallel`	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[`no`]`table`[`(`*tablespec*`)`]	suppress table or display results as a table; see [PSS-2] **power, table**
`saving(`*filename* [`, replace`]`)`	save the table data to *filename*; use `replace` to overwrite existing *filename*
Graph	
`graph`[`(`*graphopts*`)`]	graph results; see [PSS-2] **power, graph**
Iteration	
`init(#)`	initial value for sample sizes or experimental-group variance
`iterate(#)`	maximum number of iterations; default is `iterate(500)`
`tolerance(#)`	parameter tolerance; default is `tolerance(1e-12)`
`ftolerance(#)`	function tolerance; default is `ftolerance(1e-12)`
[`no`]`log`	suppress or display iteration log
[`no`]`dots`	suppress or display iterations as dots
`notitle`	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see [U] 11.1.8 numlist. Also see the parallel option.

collect is allowed; see [U] 11.1.10 Prefix commands.

sd does not appear in the dialog box; specification of sd is done automatically by the dialog box selected.

notitle does not appear in the dialog box.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
alpha	significance level	α
power	power	$1-\beta$
beta	type-II-error probability	β
N	total number of subjects	N
N1	number of subjects in the control group	N_1
N2	number of subjects in the experimental group	N_2
nratio	ratio of sample sizes, experimental to control	N_2/N_1
delta	effect size	δ
v1	control-group variance	σ_1^2
v2	experimental-group variance	σ_2^2
s1	control-group standard deviation	σ_1
s2	experimental-group standard deviation	σ_2
ratio	ratio of the experimental-group variance to the control-group variance or ratio of the experimental-group standard deviation to the control-group standard deviation (if sd is specified)	σ_2^2/σ_1^2 σ_2/σ_1
target	target parameter; synonym for v2	
_all	display all supported columns	

Column beta is shown in the default table in place of column power if specified.

Columns s1 and s2 are displayed in the default table in place of the v1 and v2 columns when the sd option is specified.

Column ratio is shown in the default table if specified. If the sd option is specified, this column contains the ratio of standard deviations. Otherwise, this column contains the ratio of variances.

Options

sd specifies that the computation be performed using the standard deviation scale. The default is to use the variance scale.

Main

`alpha()`, `power()`, `beta()`, `n()`, `n1()`, `n2()`, `nratio()`, `compute()`, `nfractional`; see [PSS-2] **power**.

`ratio(`*numlist*`)` specifies the ratio of the experimental-group variance to the control-group variance, v_2/v_1, or the ratio of the standard deviations, s_2/s_1, if the `sd` option is specified. You can specify either the experimental-group variance v_2 as a command argument or the ratio of the variances in `ratio()`. If you specify `ratio(#)`, the experimental-group variance is computed as $v_2 = v_1 \times \#$. This option is not allowed with the effect-size determination.

`direction()`, `onesided`, `parallel`; see [PSS-2] **power**.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies the initial value for the estimated parameter. For sample-size determination, the estimated parameter is either the control-group size n_1 or, if `compute(N2)` is specified, the experimental-group size n_2. For the effect-size determination, the estimated parameter is the experimental-group variance v_2 or, if the `sd` option is specified, the experimental-group standard deviation s_2. The default initial values for the variance and standard deviation for a two-sided test are obtained as a closed-form solution for the corresponding one-sided test with the significance level $\alpha/2$. The default initial values for sample sizes for a χ^2 test are obtained from the corresponding closed-form normal approximation.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following option is available with `power twovariances` but is not shown in the dialog box:

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `power twovariances` command and the methodology for power and sample-size analysis for a two-sample variances test. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis and [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Introduction

Investigators are often interested in comparing the variances of two populations, such as comparing variances in yields of corn from two plots, comparing variances of stock returns from two companies, comparing variances of the alcohol concentrations from two different yeast strains, and so on. Before conducting the actual study, the investigators need to find the optimal sample size to detect variations that are beyond tolerable limits or industry-wide standards.

This entry describes power and sample-size analysis for the inference about two population variances performed using hypothesis testing. Specifically, we consider the null hypothesis H_0: $\sigma_2^2 = \sigma_1^2$ versus the two-sided alternative hypothesis H_a: $\sigma_2^2 \neq \sigma_1^2$, the upper one-sided alternative H_a: $\sigma_2^2 > \sigma_1^2$, or the lower one-sided alternative H_a: $\sigma_2^2 < \sigma_1^2$.

Hypothesis testing of variances relies on the assumption of normality. If two independent processes are assumed to follow a normal distribution, then the ratio of their sample variances follows an F distribution, and the corresponding test is known as an F test.

The test of variances is equivalent to the test of standard deviations with the null hypothesis H_0: $\sigma_1 = \sigma_2$. The standard deviation test uses the same F test statistic. The only difference between the two tests is the scale or metric of the variability parameters: variances for the variance test and standard deviations for the standard deviation test. In some cases, standard deviations may provide a more meaningful interpretation than variances. For example, standard deviations of test scores or IQ have the same scale as the mean and provide information about the spread of the observations around the mean.

The `power twovariances` command provides power and sample-size analysis for the F test of two-sample variances or standard deviations.

Using power twovariances

`power twovariances` computes sample size, power, or experimental-group variance for a two-sample variances test. All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option. You can specify the `onesided` option to request a one-sided test. By default, all computations assume a balanced- or equal-allocation design; see [PSS-4] **Unbalanced designs** for a description of how to specify an unbalanced design.

In what follows, we describe the use of `power twovariances` in a variance metric. The corresponding use in a standard deviation metric, when the `sd` option is specified, is the same except variances v_1 and v_2 should be replaced with the respective standard deviations s_1 and s_2. Note that computations using the variance and standard deviation scales yield the same results; the difference is only in the specification of the parameters.

To compute the total sample size, you must specify the control- and experimental-group variances, v_1 and v_2, respectively, and, optionally, the power of the test in the `power()` option. The default power is set to 0.8.

Instead of the total sample size, you can compute one of the group sizes given the other one. To compute the control-group sample size, you must specify the `compute(N1)` option and the sample size of the experimental group in the `n2()` option. Likewise, to compute the experimental-group sample size, you must specify the `compute(N2)` option and the sample size of the control group in the `n1()` option.

To compute power, you must specify the total sample size in the `n()` option and the control and the experimental-group variances, v_1 and v_2, respectively.

Instead of the experimental-group variance v_2, you may specify the ratio v_2/v_1 of the experimental-group variance to the control-group variance in the `ratio()` option when computing sample size or power.

To compute effect size, the ratio of the experimental-group variance to the control-group variance, and the experimental-group variance, you must specify the total sample size in the `n()` option, the power in the `power()` option, the control-group variance v_1, and, optionally, the direction of the effect. The direction is upper by default, `direction(upper)`, which means that the experimental-group variance is assumed to be larger than the specified control-group value. You can change the direction to be lower, which means that the experimental-group variance is assumed to be smaller than the specified control-group value, by specifying the `direction(lower)` option.

Instead of the total sample size `n()`, you can specify individual group sizes in `n1()` and `n2()`, or specify one of the group sizes and `nratio()` when computing power or effect size. Also see *Two samples* in [PSS-4] **Unbalanced designs** for more details.

In the following sections, we describe the use of `power twovariances` accompanied by examples for computing sample size, power, and experimental-group variance.

Computing sample size

To compute sample size, you must specify the control- and the experimental-group variances, v_1 and v_2, respectively, and, optionally, the power of the test in the `power()` option. A default power of 0.8 is assumed if `power()` is not specified.

▷ Example 1: Sample size for a two-sample variances test

Consider a study whose goal is to investigate whether the variability in weights (measured in ounces) of bags of potato chips produced by a machine at a plant A, the control group, differs from that produced by a similar machine at a new plant B, the experimental group. The considered null hypothesis is H_0: $\sigma_A = \sigma_B$ versus a two-sided alternative hypothesis H_a: $\sigma_A \neq \sigma_B$ or, equivalently, H_0: $\sigma_A^2 = \sigma_B^2$ versus H_a: $\sigma_A^2 \neq \sigma_B^2$. The standard deviation of weights from plant A is 2 ounces. The standard deviation of weights from the new plant B is expected to be lower, 1.5 ounces. The respective variances of weights from plants A and B are 4 and 2.25. Investigators wish to obtain the minimum sample size required to detect the specified change in variability with 80% power using a 5%-level two-sided test assuming equal-group allocation. To compute sample size for this study, we specify the control- and experimental-group variances after the command name:

```
. power twovariances 4 2.25

Performing iteration ...

Estimated sample sizes for a two-sample variances test
F test
H0: v2 = v1  versus  Ha: v2 != v1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.5625
           v1 =    4.0000
           v2 =    2.2500

Estimated sample sizes:

            N =       194
  N per group =        97
```

A total sample of 194 bags, 97 in each plant, must be obtained to detect the specified ratio of variances in the two plants with 80% power using a two-sided 5%-level test.

◁

▷ Example 2: Standard deviation scale

We can also specify standard deviations instead of variances, in which case we must also specify the `sd` option:

```
. power twovariances 2 1.5, sd
Performing iteration ...
Estimated sample sizes for a two-sample standard-deviations test
F test
H0: s2 = s1  versus  Ha: s2 != s1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.7500
           s1 =    2.0000
           s2 =    1.5000
Estimated sample sizes:
            N =       194
  N per group =        97
```

We obtain the same sample sizes as in example 1.

◁

▷ Example 3: Specifying ratio of variances or standard deviations

Instead of the experimental-group variance of 2.25 as in example 1, we can specify the ratio of variances $2.25/4 = 0.5625$ in the `ratio()` option.

```
. power twovariances 4, ratio(0.5625)
Performing iteration ...
Estimated sample sizes for a two-sample variances test
F test
H0: v2 = v1  versus  Ha: v2 != v1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.5625
           v1 =    4.0000
           v2 =    2.2500
        ratio =    0.5625
Estimated sample sizes:
            N =       194
  N per group =        97
```

The results are identical to those from example 1.

Similarly, instead of the experimental-group standard deviation of 1.5 as in example 2, we can specify the ratio of standard deviations $1.5/2 = 0.75$ in the `ratio()` option and obtain the same results:

```
. power twovariances 2, sd ratio(0.75)
Performing iteration ...
Estimated sample sizes for a two-sample standard-deviations test
F test
H0: s2 = s1  versus  Ha: s2 != s1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.7500
           s1 =    2.0000
           s2 =    1.5000
        ratio =    0.7500
Estimated sample sizes:
            N =       194
  N per group =        97
```

◁

▷ Example 4: Computing one of the group sizes

Continuing with example 1, we will suppose that investigators anticipate a sample of 100 bags from plant A and wish to compute the required number of bags from plant B. To compute sample size for plant B using the study parameters of example 1, we use a combination of the `n1()` and `compute(N2)` options.

```
. power twovariances 4 2.25, n1(100) compute(N2)
Performing iteration ...
Estimated sample sizes for a two-sample variances test
F test
H0: v2 = v1  versus  Ha: v2 != v1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.5625
           v1 =    4.0000
           v2 =    2.2500
           N1 =       100
Estimated sample sizes:
            N =       194
           N2 =        94
```

A slightly smaller sample of 94 bags is needed from plant B given a slightly larger sample of bags from plant A to achieve the same 80% power as in example 1.

If the sample size for plant B is known a priori, you can compute the sample size for plant A by specifying the `n2()` and `compute(N1)` options.

◁

▷ Example 5: Unbalanced design

By default, `power twovariances` computes sample size for a balanced- or equal-allocation design. If we know the allocation ratio of subjects between the groups, we can compute the required sample size for an unbalanced design by specifying the `nratio()` option.

Continuing with example 1, we will suppose that the new plant B is more efficient and more cost effective in producing bags of chips than plant A. Investigators anticipate twice as many bags from plant B than from plant A; that is, $n_2/n_1 = 2$. We compute the required sample size for this unbalanced design by specifying the `nratio()` option:

```
. power twovariances 4 2.25, nratio(2)
Performing iteration ...
Estimated sample sizes for a two-sample variances test
F test
H0: v2 = v1  versus  Ha: v2 != v1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.5625
           v1 =    4.0000
           v2 =    2.2500
        N2/N1 =    2.0000
Estimated sample sizes:
            N =       225
           N1 =        75
           N2 =       150
```

The requirement on the total sample size increases for an unbalanced design compared with the balanced design from example 1. Investigators must decide whether the decrease of 22 from 97 to 75 in the number of bags from plant A covers the cost of the additional 53 ($150 - 97 = 53$) bags from plant B.

Also see *Two samples* in [PSS-4] **Unbalanced designs** for more examples of unbalanced designs for two-sample tests.

◁

Computing power

To compute power, you must specify the total sample size in the `n()` option and the control- and experimental-group variances, v_1 and v_2, respectively.

▷ Example 6: Power of a two-sample variances test

Continuing with example 1, we will suppose that the investigators can afford a total sample of 250 bags, 125 from each plant, and want to find the power corresponding to this larger sample size.

To compute the power corresponding to this sample size, we specify the total sample size in the n() option:

```
. power twovariances 4 2.25, n(250)
Estimated power for a two-sample variances test
F test
H0: v2 = v1  versus  Ha: v2 != v1
Study parameters:
        alpha =    0.0500
            N =       250
  N per group =       125
        delta =    0.5625
           v1 =    4.0000
           v2 =    2.2500
Estimated power:
        power =    0.8908
```

With a total sample of 250 bags, 125 per plant, we obtain a power of roughly 89%.

◁

▷ Example 7: Multiple values of study parameters

In this example, we assess the effect of varying the variances of weights obtained from plant B on the power of our study. Continuing with example 6, we vary the experimental-group variance from 1.5 to 3 in 0.25 increments. We specify the corresponding *numlist* in parentheses:

```
. power twovariances 4 (1.5(0.25)3), n(250)
Estimated power for a two-sample variances test
F test
H0: v2 = v1  versus  Ha: v2 != v1
```

alpha	power	N	N1	N2	delta	v1	v2
.05	.9997	250	125	125	.375	4	1.5
.05	.9956	250	125	125	.4375	4	1.75
.05	.9701	250	125	125	.5	4	2
.05	.8908	250	125	125	.5625	4	2.25
.05	.741	250	125	125	.625	4	2.5
.05	.5466	250	125	125	.6875	4	2.75
.05	.3572	250	125	125	.75	4	3

The power decreases from 99.97% to 35.72% as the experimental-group variance gets closer to the control-group variance of 4.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

Computing effect size and experimental-group variance

Effect size δ for a two-sample variances test is defined as the ratio of the experimental-group variance to the control-group variance, $\delta = v_2/v_1$. If the sd option is specified, effect size δ is the ratio of the experimental-group standard deviation to the control-group standard deviation, $\delta = s_2/s_1$.

Sometimes, we may be interested in determining the smallest effect and the corresponding experimental-group variance that yield a statistically significant result for prespecified sample size and power. In this case, power, sample size, and control-group variance must be specified. In addition, you must also decide on the direction of the effect: upper, meaning $v_2 > v_1$, or lower, meaning $v_2 < v_1$. The direction may be specified in the `direction()` option; `direction(upper)` is the default. If the `sd` option is specified, the estimated parameter is the experimental-group standard deviation instead of the variance.

▷ Example 8: Minimum detectable value of the experimental-group variance

Continuing with example 6, we will compute the minimum plant B variance that can be detected given a total sample of 250 bags and 80% power. To find the variance, after the command name, we specify the plant A variance of 4, total sample size `n(250)`, and power `power(0.8)`:

```
. power twovariances 4, n(250) power(0.8)

Performing iteration ...

Estimated experimental-group variance for a two-sample variances test
F test
H0: v2 = v1  versus  Ha: v2 != v1; v2 > v1

Study parameters:

        alpha =    0.0500
        power =    0.8000
            N =       250
  N per group =       125
           v1 =    4.0000

Estimated effect size and experimental-group variance:

        delta =    1.6573
           v2 =    6.6291
```

We find that the minimum value of the experimental-group variance that would yield a statistically significant result in this study is 6.63, and the corresponding effect size is 1.66.

In this example, we computed the variance assuming an upper direction, or a ratio greater than 1, $\delta > 1$. To request a lower direction, or a ratio less than 1, we can specify the `direction(lower)` option.

◁

Testing a hypothesis about two independent variances

In this section, we demonstrate the use of the `sdtest` command for testing a hypothesis about two independent variances; see [R] **sdtest** for details.

▷ Example 9: Comparing two variances

Consider the `fuel` dataset analyzed in [R] **sdtest**. Suppose we want to investigate the effectiveness of a new fuel additive on the mileage of cars. We have a sample of 12 cars, where each car was run without the additive and later with the additive. The results of each run are stored in variables `mpg1` and `mpg2`.

```
. use https://www.stata-press.com/data/r19/fuel
. sdtest mpg1==mpg2
Variance ratio test
```

Variable	Obs	Mean	Std. err.	Std. dev.	[95% conf.	interval]
mpg1	12	21	.7881701	2.730301	19.26525	22.73475
mpg2	12	22.75	.9384465	3.250874	20.68449	24.81551
Combined	24	21.875	.6264476	3.068954	20.57909	23.17091

```
    ratio = sd(mpg1) / sd(mpg2)                                 f =   0.7054
H0: ratio = 1                             Degrees of freedom =   11, 11

    Ha: ratio < 1               Ha: ratio != 1                 Ha: ratio > 1
  Pr(F < f) = 0.2862         2*Pr(F < f) = 0.5725           Pr(F > f) = 0.7138
```

`sdtest` uses the ratio of the control-group standard deviation to the experimental-group standard deviation as its test statistic. We do not have sufficient evidence to reject the null hypothesis of H_0: $\sigma_1 = \sigma_2$ versus the two-sided alternative H_a: $\sigma_1 \neq \sigma_2$; the p-value > 0.5725.

We use the estimates of this study to perform a sample-size analysis we would have conducted before the study. We assume an equal-group design and power of 80%.

```
. power twovariances 2.73 3.25, sd power(0.8)
Performing iteration ...
Estimated sample sizes for a two-sample standard-deviations test
F test
H0: s2 = s1  versus  Ha: s2 != s1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    1.1905
           s1 =    2.7300
           s2 =    3.2500
Estimated sample sizes:
            N =       522
  N per group =       261
```

The total sample size required by the test to detect the difference between the two standard deviations of 2.73 in the control group and of 3.25 in the experimental group is 522, 261 for each group, which is significantly larger than the sample of 12 cars in our `fuel` dataset.

◁

Stored results

power twovariances stores the following in r():

Scalars

r(alpha)	significance level
r(power)	power
r(beta)	probability of a type II error
r(delta)	effect size
r(N)	total sample size
r(N_a)	actual sample size
r(N1)	sample size of the control group
r(N2)	sample size of the experimental group
r(nratio)	ratio of sample sizes, N2/N1
r(nratio_a)	actual ratio of sample sizes
r(nfractional)	1 if nfractional is specified, 0 otherwise
r(onesided)	1 for a one-sided test, 0 otherwise
r(v1)	control-group variance
r(v2)	experimental-group variance
r(ratio)	ratio of the experimental- to the control-group variances (or standard deviations if sd is specified)
r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise
r(init)	initial value for sample sizes, experimental-group variance, or standard deviation
r(maxiter)	maximum number of iterations
r(iter)	number of iterations performed
r(tolerance)	requested parameter tolerance
r(deltax)	final parameter tolerance achieved
r(ftolerance)	requested distance of the objective function from zero
r(function)	final distance of the objective function from zero
r(converged)	1 if iteration algorithm converged, 0 otherwise

Macros

r(type)	test
r(method)	twovariances
r(scale)	variance or standard deviation
r(direction)	upper or lower
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results

Methods and formulas

Consider two independent samples from a normal population with means μ_1 and μ_2 and variances σ_1^2 and σ_2^2. The ratio $(s_1/\sigma_1)^2/(s_2/\sigma_2)^2$ has an F distribution with n_1-1 numerator and n_2-1 denominator degrees of freedom. s_1^2 and s_2^2 are the sample variances, and n_1 and n_2 are the sample sizes.

Let σ_1^2 and σ_2^2 be the control-group and experimental-group variances, respectively.

A two-sample variances test involves testing the null hypothesis H_0: $\sigma_2^2 = \sigma_1^2$ versus the two-sided alternative hypothesis H_a: $\sigma_2^2 \neq \sigma_1^2$, the upper one-sided alternative H_a: $\sigma_2^2 > \sigma_1^2$, or the lower one-sided alternative H_a: $\sigma_2^2 < \sigma_1^2$.

Equivalently, the hypotheses can be expressed in terms of the ratio of the two variances: H_0: $\sigma_2^2/\sigma_1^2 = 1$ versus the two-sided alternative H_a: $\sigma_2^2/\sigma_1^2 \neq 1$, the upper one-sided alternative H_a: $\sigma_2^2/\sigma_1^2 > 1$, or the lower one-sided alternative H_a: $\sigma_2^2/\sigma_1^2 < 1$.

The following formulas are based on Dixon and Massey (1983, 116–119).

Let α be the significance level, β be the probability of a type II error, and $F_\alpha = F_{n_1-1,n_2-1,\alpha}$ and $F_{n_1-1,n_2-1,\beta}$ be the αth and the βth quantiles of an F distribution with $n_1 - 1$ numerator and $n_2 - 1$ denominator degrees of freedom.

The power $\pi = 1 - \beta$ is computed using

$$\pi = \begin{cases} 1 - F_{n_1-1,n_2-1}\left(\frac{\sigma_1^2}{\sigma_2^2}F_{1-\alpha}\right) & \text{for an upper one-sided test} \\ F_{n_1-1,n_2-1}\left(\frac{\sigma_1^2}{\sigma_2^2}F_{\alpha}\right) & \text{for a lower one-sided test} \\ 1 - F_{n_1-1,n_2-1}\left(\frac{\sigma_1^2}{\sigma_2^2}F_{1-\alpha/2}\right) + F_{n_1-1,n_2-1}\left(\frac{\sigma_1^2}{\sigma_2^2}F_{\alpha/2}\right) & \text{for a two-sided test} \end{cases} \quad (1)$$

where $F_{n_1-1,n_2-1}(\cdot)$ is the cdf of an F distribution with $n_1 - 1$ numerator and $n_2 - 1$ denominator degrees of freedom.

Let $R = n_2/n_1$ denote the allocation ratio. Then $n_2 = R \times n_1$ and power can be viewed as a function of n_1. Therefore, for sample-size determination, the control-group sample size n_1 is computed first. The experimental-group size n_2 is then computed as $R \times n_1$, and the total sample size is computed as $n = n_1 + n_2$. By default, sample sizes are rounded to integer values; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for details.

If either n_1 or n_2 is known, the other sample size is computed iteratively from the corresponding power equation in (1).

The initial values for the sample sizes are obtained from closed-form large-sample normal approximations; see, for example, Mathews (2010, 68).

For a one-sided test, the minimum detectable value of the experimental-group variance is computed as follows:

$$\sigma_2^2 = \begin{cases} \sigma_1^2 \frac{F_{n_1-1,n_2-1,1-\alpha}}{F_{n_1-1,n_2-1,\beta}} & \text{for an upper one-sided test} \\ \sigma_1^2 \frac{F_{n_1-1,n_2-1,\alpha}}{F_{n_1-1,n_2-1,1-\beta}} & \text{for a lower one-sided test} \end{cases} \quad (2)$$

For a two-sided test, the minimum detectable value of the experimental-group variance is computed iteratively using the two-sided power equation from (1). The default initial value is obtained from (2) with α replaced by $\alpha/2$.

References

Dixon, W. J., and F. J. Massey, Jr. 1983. *Introduction to Statistical Analysis*. 4th ed. New York: McGraw–Hill.

Mathews, P. 2010. *Sample Size Calculations: Practical Methods for Engineers and Scientists*. Fairport Harbor, OH: Mathews Malnar and Bailey.

Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[R] **sdtest** — Variance-comparison tests

power onecorrelation — Power analysis for a one-sample correlation test

Description

`power onecorrelation` computes sample size, power, or target correlation for a one-sample correlation test. By default, it computes sample size for given power and the values of the correlation parameters under the null and alternative hypotheses. Alternatively, it can compute power for given sample size and values of the null and alternative correlations or the target correlation for given sample size, power, and the null correlation. Also see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Quick start

Sample size for a test of H_0: $\rho = 0$ versus H_a: $\rho \neq 0$ with null correlation $r_0 = 0$, alternative correlation $r_a = 0.3$, and default power of 0.8 and significance level $\alpha = 0.05$

```
power onecorrelation 0 .3
```

Same as above, for alternative correlations of 0.2, 0.25, 0.3, 0.35, and 0.4

```
power onecorrelation 0 (.2(.05).4)
```

Same as above, but display a graph showing sample size versus alternative correlation

```
power onecorrelation 0 (.2(.05).4), graph
```

Sample size for $r_0 = 0.1$, $r_a = 0.3$ with power of 0.85 and $\alpha = 0.01$

```
power onecorrelation .1 .3, power(.85) alpha(.01)
```

Power for a sample size of 30

```
power onecorrelation 0 .5, n(30)
```

Effect size and target correlation for a sample size of 20 and power of 0.8

```
power onecorrelation 0, n(20) power(.8)
```

Same as above, but for sample sizes of 20, 30, 40, and 50

```
power onecorrelation 0, n(20(10)50) power(.8)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
power onecorrelation r0 ra [ , power(numlist) options ]
```

Compute power

```
power onecorrelation r0 ra , n(numlist) [ options ]
```

Compute effect size and target correlation

```
power onecorrelation r0 , n(numlist) power(numlist) [ options ]
```

where r_0 is the null (hypothesized) correlation or the value of the correlation under the null hypothesis, and r_a is the alternative (target) correlation or the value of the correlation under the alternative hypothesis. r_0 and r_a may each be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

options	Description
Main	
* `alpha(`*numlist*`)`	significance level; default is `alpha(0.05)`
* `power(`*numlist*`)`	power; default is `power(0.8)`
* `beta(`*numlist*`)`	probability of type II error; default is `beta(0.2)`
* `n(`*numlist*`)`	sample size; required to compute power or effect size
`nfractional`	allow fractional sample size
* `diff(`*numlist*`)`	difference between the alternative correlation and the null correlation, $r_a - r_0$; specify instead of the alternative correlation r_a
`direction(upper|lower)`	direction of the effect for effect-size determination; default is `direction(upper)`, which means that the postulated value of the parameter is larger than the hypothesized value
`onesided`	one-sided test; default is two sided
`parallel`	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[`no`]`table`[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
`saving(`*filename* [`, replace`]`)`	save the table data to *filename*; use `replace` to overwrite existing *filename*
Graph	
`graph`[(*graphopts*)]	graph results; see [PSS-2] **power, graph**
Iteration	
`init(#)`	initial value for sample size or correlation
`iterate(#)`	maximum number of iterations; default is `iterate(500)`
`tolerance(#)`	parameter tolerance; default is `tolerance(1e-12)`
`ftolerance(#)`	function tolerance; default is `ftolerance(1e-12)`
[`no`]`log`	suppress or display iteration log
[`no`]`dots`	suppress or display iterations as dots
`notitle`	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the `parallel` option.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`notitle` does not appear in the dialog box.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`N`	number of subjects	N
`delta`	effect size	δ
`r0`	null correlation	ρ_0
`ra`	alternative correlation	ρ_a
`diff`	difference between the alternative and null correlations	$\rho_a - \rho_0$
`target`	target parameter; synonym for `ra`	
`_all`	display all supported columns	

Column `beta` is shown in the default table in place of column `power` if specified.

Options

Main

`alpha()`, `power()`, `beta()`, `n()`, `nfractional`; see [PSS-2] **power**. The `nfractional` option is allowed only for sample-size determination.

`diff(`*numlist*`)` specifies the difference between the alternative correlation and the null correlation, $r_a - r_0$. You can specify either the alternative correlation r_a as a command argument or the difference between the two correlations in the `diff()` option. If you specify `diff(#)`, the alternative correlation is computed as $r_a = r_0 + \#$. This option is not allowed with the effect-size determination.

`direction()`, `onesided`, `parallel`; see [PSS-2] **power**.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies the initial value for the iteration procedure. Iteration is used to compute sample size or target correlation for a two-sided test. The default initial value for the estimated parameter is obtained from the corresponding closed-form one-sided computation using the significance level $\alpha/2$.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following option is available with `power onecorrelation` but is not shown in the dialog box:

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `power onecorrelation` command and the methodology for power and sample-size analysis for a one-sample correlation test. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis and [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Introduction

Correlation analysis emanates from studies quantifying dependence between random variables, such as dependence between height and weight of individuals, between blood pressure and cholesterol levels, between the SAT scores of students and their first-year grade point average, between the number of minutes per week customers spend using a new fitness program and the number of pounds lost, and many more.

The correlation coefficient ρ is commonly used to measure the strength of such dependence. We consider Pearson's correlation obtained by standardizing the covariance between two random variables. As a result, the correlation coefficient has no units and ranges between -1 and 1.

This entry describes power and sample-size analysis for the inference about the population correlation coefficient performed using hypothesis testing. Specifically, we consider the null hypothesis H_0: $\rho = \rho_0$ versus the two-sided alternative hypothesis H_a: $\rho \neq \rho_0$, the upper one-sided alternative H_a: $\rho > \rho_0$, or the lower one-sided alternative H_a: $\rho < \rho_0$. In most applications, the null value of the correlation, ρ_0, is set to zero.

Statistical inference on the correlation coefficient requires a distributional assumption between two random variables—bivariate normality with correlation ρ. The distribution of the sample correlation coefficient is rather complicated except under the null hypothesis of the true correlation being zero, H_0: $\rho = 0$, under which it is a Student's t distribution (for example, see Graybill [1961, 209]). The general inference of H_0: $\rho = \rho_0$ is based on the asymptotic approximation.

One common approach in testing hypotheses concerning the correlation parameter is to use an inverse hyperbolic tangent transformation, $\tanh^{-1}(x) = 0.5\ln(1+x)/\ln(1-x)$, also known as Fisher's z transformation when applied to the correlation coefficient (Fisher 1915). Specifically, if $\hat{\rho}$ is the sample correlation coefficient and n is the sample size, Fisher (1915) showed that

$$\sqrt{n-3}\left\{\tanh^{-1}(\hat{\rho}) - \tanh^{-1}(\rho)\right\} \sim N(0,1)$$

for n as small as 10, although the approximation tends to perform better for $n > 25$. The null hypothesis H_0: $\rho = \rho_0$ is equivalent to H_0: $\tanh^{-1}(\rho) = \tanh^{-1}(\rho_0)$. The latter test is referred to as Fisher's z test.

`power onecorrelation` performs computations based on the asymptotic Fisher's z test.

Using power onecorrelation

`power onecorrelation` computes sample size, power, or target correlation for a one-sample correlation test. All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option. You can specify the `onesided` option to request a one-sided test.

To compute sample size, you must specify the correlations under the null and alternative hypotheses, r_0 and r_a, respectively, and, optionally, the power of the test in the `power()` option. The default power is set to 0.8.

To compute power, you must specify the sample size in the `n()` option and the correlations under the null and alternative hypotheses, r_0 and r_a, respectively.

Instead of the alternative correlation, r_a, you may specify the difference $r_a - r_0$ between the alternative correlation and the null correlation in the `diff()` option when computing sample size or power.

To compute effect size, the difference between the alternative and null correlations, and target correlation, you must specify the sample size in the `n()` option, the power in the `power()` option, the null correlation r_0, and, optionally, the direction of the effect. The direction is upper by default, `direction(upper)`, which means that the target correlation is assumed to be larger than the specified null value. This is also equivalent to the assumption of a positive effect size. You can change the direction to lower, which means that the target correlation is assumed to be smaller than the specified null value, by specifying the `direction(lower)` option. This is equivalent to assuming a negative effect size.

By default, the computed sample size is rounded up. You can specify the `nfractional` option to see the corresponding fractional sample size; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. The `nfractional` option is allowed only for sample-size determination.

The sample-size and effect-size determinations for a two-sided test require iteration. The default initial value for the estimated parameter is obtained from the corresponding closed-form one-sided computation using the significance level $\alpha/2$. The default initial value may be changed by specifying the `init()` option. See [PSS-2] **power** for the descriptions of other options that control the iteration procedure.

In the following sections, we describe the use of `power onecorrelation` accompanied by examples for computing sample size, power, and target correlation.

Computing sample size

To compute sample size, you must specify the correlations under the null and alternative hypotheses, r_0 and r_a, respectively, and, optionally, the power of the test in the `power()` option. A default power of 0.8 is assumed if `power()` is not specified.

▷ Example 1: Sample size for a one-sample correlation test

Consider a study where the goal is to test the existence of a positive correlation between height and weight of individuals, that is, H_0: $\rho = 0$ versus H_a: $\rho > 0$. Before conducting the study, we wish to determine the sample size required to detect a correlation of at least 0.5 with 80% power using a one-sided 5%-level test. We specify the values 0 and 0.5 of the null and alternative correlations after the command name. The only option we need to specify is `onesided`, which requests a one-sided test. We omit options `alpha(0.05)` and `power(0.8)` because the specified values are their defaults.

```
. power onecorrelation 0 0.5, onesided

Estimated sample size for a one-sample correlation test
Fisher's z test
H0: r = r0  versus  Ha: r > r0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.5000
           r0 =    0.0000
           ra =    0.5000

Estimated sample size:

            N =        24
```

A sample of 24 individuals must be obtained to detect a correlation of at least 0.5 between height and weight with 80% power using an upper one-sided 5%-level test.

If we anticipate a stronger correlation of, say, 0.7, we will need a sample of only 12 subjects:

```
. power onecorrelation 0 0.7, onesided

Estimated sample size for a one-sample correlation test
Fisher's z test
H0: r = r0  versus  Ha: r > r0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.7000
           r0 =    0.0000
           ra =    0.7000

Estimated sample size:

            N =        12
```

This is consistent with our expectation that in this example, as the alternative correlation increases, the required sample size decreases.

◁

Computing power

To compute power, you must specify the sample size in the `n()` option and the correlations under the null and alternative hypotheses, r_0 and r_a, respectively.

▷ Example 2: Power of a one-sample correlation test

Continuing with example 1, we will suppose that we are designing a pilot study and would like to keep our initial sample small. We want to find out how much the power of the test would decrease if we sample only 15 individuals. To compute power, we specify the sample size in `n()`:

```
. power onecorrelation 0 0.5, n(15) onesided
Estimated power for a one-sample correlation test
Fisher's z test
H0: r = r0  versus  Ha: r > r0
Study parameters:
        alpha =    0.0500
            N =        15
        delta =    0.5000
           r0 =    0.0000
           ra =    0.5000
Estimated power:
        power =    0.6018
```

The power decreases to roughly 60%. The power to detect a correlation of 0.5 is low for a sample of 15 individuals. We should consider either a larger target correlation or a larger sample.

◁

▷ Example 3: Nonzero null hypothesis

To demonstrate power computation for a nonzero null hypothesis, we consider an example from Graybill (1961, 211) of a hypothetical study where the primary interest lies in the correlation between protein and fat for a particular breed of dairy cow. The sample contains 24 cows. Suppose we want to compute the power of a two-sided test with the null hypothesis H_0: $\rho = 0.5$ versus the alternative hypothesis H_a: $\rho \neq 0.5$. Suppose that the correlation between protein and fat was estimated to be 0.3 in an earlier pilot study. We can compute power by typing

```
. power onecorrelation 0.5 0.3, n(24)
Estimated power for a one-sample correlation test
Fisher's z test
H0: r = r0  versus  Ha: r != r0
Study parameters:
        alpha =    0.0500
            N =        24
        delta =   -0.2000
           r0 =    0.5000
           ra =    0.3000
Estimated power:
        power =    0.1957
```

The power of the test is estimated to be 0.1957 for the above study, which is unacceptably low for practical purposes. We should consider either increasing the sample size or choosing a different target correlation.

In fact, the author considers two values of the alternative correlation, 0.3 and 0.2. We can compute the corresponding powers for the multiple values of the alternative correlation by enclosing them in parentheses, as we demonstrate below.

```
. power onecorrelation 0.5 (0.3 0.2), n(24)
Estimated power for a one-sample correlation test
Fisher's z test
H0: r = r0  versus  Ha: r != r0
```

alpha	power	N	delta	r0	ra
.05	.1957	24	-.2	.5	.3
.05	.3552	24	-.3	.5	.2

The power increases from 20% to 36% as the alternative correlation decreases from 0.3 to 0.2. It may seem counterintuitive that the power increased as the value of the alternative correlation decreased. It is important to remember that the power of the correlation test is an increasing function of the magnitude of the effect size, $\delta = \rho_a - \rho_0$, the difference between the alternative and null values of the correlation. So the power of 20% corresponds to $|\delta| = |0.3 - 0.5| = 0.2$, and the power of 36% corresponds to $|\delta| = 0.3$. ◁

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

Computing effect size and target correlation

In a one-sample correlation analysis, the effect size is commonly defined as the difference between the alternative correlation and the null correlation, $\delta = \rho_a - \rho_0$. For a zero-null hypothesis, the effect size is the alternative correlation.

The distribution of the sample correlation coefficient is symmetric when the population correlation ρ is zero and becomes negatively skewed as ρ approaches 1 and positively skewed as ρ approaches -1. Also, the sampling variance approaches zero as $|\rho|$ approaches 1. Clearly, the power of a correlation test depends on the magnitude of the true correlation; therefore, δ alone is insufficient for determining the power. In other words, for the same difference $\delta = 0.1$, the power to detect the difference between $\rho_a = 0.2$ and $\rho_0 = 0.1$ is not the same as that between $\rho_a = 0.3$ and $\rho_0 = 0.2$; the former difference is associated with the larger sampling variability and thus has lower power to be detected.

Sometimes, we may be interested in determining the smallest effect (for a given null correlation) and the corresponding target correlation that yield a statistically significant result for prespecified sample size and power. In this case, power, sample size, and null correlation must be specified. In addition, you must also decide on the direction of the effect: upper, meaning $\rho_a > \rho_0$, or lower, meaning $\rho_a < \rho_0$. The direction may be specified in the `direction()` option; `direction(upper)` is the default.

▷ Example 4: Minimum detectable value of the correlation

Continuing with example 2, we will compute the minimum positive correlation that can be detected given a sample of 15 individuals and 80% power. To solve for target correlation, we specify the null correlation of `0` after the command, sample size `n(15)`, and power `power(0.8)`:

```
. power onecorrelation 0, n(15) power(0.8) onesided
Estimated target correlation for a one-sample correlation test
Fisher's z test
H0: r = r0  versus  Ha: r > r0; ra > r0
Study parameters:
        alpha =    0.0500
        power =    0.8000
            N =        15
           r0 =    0.0000
Estimated effect size and target correlation:
        delta =    0.6155
           ra =    0.6155
```

We find that the minimum value of the correlation as well as of the effect size that would yield a statistically significant result in this study is 0.6155.

In this example, we computed the correlation assuming an upper direction, $\rho_a > \rho_0$, or a positive effect, $\delta > 0$. To request a lower direction, or a negative effect, we can specify the `direction(lower)` option.

◁

Performing hypothesis tests on correlation

After the data are collected, you can use the `pwcorr` command to test for correlation; see [R] **correlate**.

▷ Example 5: Testing for correlation

For example, consider `auto.dta`, which contains various characteristics of 74 cars. Suppose that our study goal is to investigate the existence of a correlation between miles per gallon and weights of cars. The corresponding variables in the dataset are `mpg` and `weight`. We use `pwcorr` to perform the test.

```
. use https://www.stata-press.com/data/r19/auto
(1978 automobile data)
. pwcorr mpg weight, sig
             |      mpg   weight
-------------+------------------
         mpg |   1.0000
             |
             |
      weight |  -0.8072   1.0000
             |   0.0000
             |
```

The test finds a statistically significant negative correlation of -0.8 between `mpg` and `weight`; the p-value is less than 0.0001.

We use the parameters of this study to perform a sample-size analysis we would have conducted before the study.

```
. power onecorrelation 0 -0.8, power(0.9) alpha(0.01)

Performing iteration ...

Estimated sample size for a one-sample correlation test
Fisher's z test
H0: r = r0  versus  Ha: r != r0

Study parameters:

        alpha =    0.0100
        power =    0.9000
        delta =   -0.8000
           r0 =    0.0000
           ra =   -0.8000

Estimated sample size:

            N =        16
```

We find that the sample size required to detect a correlation of -0.8 with 90% power using a 1%-level two-sided test is 16. The current sample contains many more cars (74), which would allow us to detect a potentially smaller (in absolute value) correlation.

We should point out that `pwcorr` performs an exact one-sample correlation test of $H_0: \rho = 0$, which uses a Student's t distribution. `power onecorrelation` performs computation for an approximate Fisher's z test, which uses the normal distribution.

◁

Stored results

`power onecorrelation` stores the following in `r()`:

Scalars

`r(alpha)`	significance level
`r(power)`	power
`r(beta)`	probability of a type II error
`r(delta)`	effect size
`r(N)`	sample size
`r(nfractional)`	1 if `nfractional` is specified, 0 otherwise
`r(onesided)`	1 for a one-sided test, 0 otherwise
`r(r0)`	correlation under the null hypothesis
`r(ra)`	correlation under the alternative hypothesis
`r(diff)`	difference between the alternative and null correlations
`r(separator)`	number of lines between separator lines in the table
`r(divider)`	1 if `divider` is requested in the table, 0 otherwise
`r(init)`	initial value for sample size or correlation
`r(maxiter)`	maximum number of iterations
`r(iter)`	number of iterations performed
`r(tolerance)`	requested parameter tolerance
`r(deltax)`	final parameter tolerance achieved
`r(ftolerance)`	requested distance of the objective function from zero
`r(function)`	final distance of the objective function from zero
`r(converged)`	1 if iteration algorithm converged, 0 otherwise

Macros

`r(type)`	`test`
`r(method)`	`onecorrelation`
`r(direction)`	`upper` or `lower`
`r(columns)`	displayed table columns
`r(labels)`	table column labels

r(widths)	table column widths
r(formats)	table column formats
Matrices	
r(pss_table)	table of results

Methods and formulas

Let ρ denote Pearson's correlation and $\hat{\rho}$ its estimator, the sample correlation coefficient.

A one-sample correlation test involves testing the null hypothesis H_0: $\rho = \rho_0$ versus the two-sided alternative hypothesis H_a: $\rho \neq \rho_0$, the upper one-sided alternative H_a: $\rho > \rho_0$, or the lower one-sided alternative H_a: $\rho < \rho_0$.

The exact distribution of the sample correlation coefficient $\hat{\rho}$ is complicated for testing a general hypothesis H_0: $\rho = \rho_0 \neq 0$. An alternative is to apply an inverse hyperbolic tangent, $\tanh^{-1}(\cdot)$, or Fisher's z transformation to $\hat{\rho}$ to approximate its sampling distribution by a normal distribution (Fisher 1915). The hypothesis H_0: $\rho = \rho_0$ is then equivalent to H_0: $\tanh^{-1}(\rho) = \tanh^{-1}(\rho_0)$.

Let $Z = \tanh^{-1}(\hat{\rho}) = 0.5\ln\{(1+\hat{\rho})/(1-\hat{\rho})\}$ denote Fisher's z transformation of $\hat{\rho}$. Then Z follows a normal distribution with mean $\mu_z = \tanh^{-1}(\rho)$ and standard deviation $\sigma_z = 1/\sqrt{n-3}$, where n is the sample size (for example, see Graybill [1961, 211]; Anderson [2003, 134]).

Let α be the significance level, β be the probability of a type II error, and $z_{1-\alpha/k}$ and z_β be the $(1-\alpha/k)$th and the βth quantiles of the standard normal distribution. Denote $\delta_z = \mu_a - \mu_0 = \tanh^{-1}(\rho_a) - \tanh^{-1}(\rho_0)$.

The power $\pi = 1-\beta$ is computed using

$$\pi = \begin{cases} \Phi\left(\frac{\delta_z}{\sigma_z} - z_{1-\alpha}\right) & \text{for an upper one-sided test} \\ \Phi\left(-\frac{\delta_z}{\sigma_z} - z_{1-\alpha}\right) & \text{for a lower one-sided test} \\ \Phi\left(\frac{\delta_z}{\sigma_z} - z_{1-\alpha/2}\right) + \Phi\left(-\frac{\delta_z}{\sigma_z} - z_{1-\alpha/2}\right) & \text{for a two-sided test} \end{cases} \tag{1}$$

where $\Phi(\cdot)$ is the cdf of a standard normal distribution.

The sample size n for a one-sided test is computed using

$$n = 3 + \left(\frac{z_{1-\alpha} - z_\beta}{\delta_z}\right)^2$$

See, for example, Lachin (1981) for the case of $\rho_0 = 0$.

The minimum detectable value of the correlation is obtained by first finding the corresponding minimum value of μ_a and then applying the inverse Fisher's z transformation to the μ_a:

$$\rho_a = \frac{e^{2\mu_a} - 1}{e^{2\mu_a} + 1}$$

For a one-sided test, $\mu_a = \mu_0 + \sigma_z(z_{1-\alpha} - z_\beta)$ when $\mu_a > \mu_0$, and $\mu_a = \mu_0 - \sigma_z(z_{1-\alpha} - z_\beta)$ when $\mu_a < \mu_0$.

For a two-sided test, the sample size n and μ_a are obtained by iteratively solving the two-sided power equation in (1) for n and μ_a, respectively. The initial values are obtained from the respective formulas for the one-sided computation with the significance level $\alpha/2$.

If the `nfractional` option is not specified, the computed sample size is rounded up.

References

Anderson, T. W. 2003. *An Introduction to Multivariate Statistical Analysis.* 3rd ed. New York: Wiley.

Fisher, R. A. 1915. Frequency distribution of the values of the correlation coefficient in samples from an indefinitely large population. *Biometrika* 10: 507–521. https://doi.org/10.2307/2331838.

Graybill, F. A. 1961. *An Introduction to Linear Statistical Models.* Vol. 1. New York: McGraw–Hill.

Guenther, W. C. 1977. Desk calculation of probabilities for the distribution of the sample correlation coefficient. *American Statistician* 31: 45–48. https://doi.org/10.1080/00031305.1977.10479195.

Lachin, J. M. 1981. Introduction to sample size determination and power analysis for clinical trials. *Controlled Clinical Trials* 2: 93–113. https://doi.org/10.1016/0197-2456(81)90001-5.

Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[R] **correlate** — Correlations of variables

power twocorrelations — Power analysis for a two-sample correlations test

Description

`power twocorrelations` computes sample size, power, or the experimental-group correlation for a two-sample correlations test. By default, it computes sample size for given power and the values of the control-group and experimental-group correlations. Alternatively, it can compute power for given sample size and values of the control-group and experimental-group correlations or the experimental-group correlation for given sample size, power, and the control-group correlation. Also see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Quick start

Sample size for a test of H_0: $\rho_1 = \rho_2$ versus H_a: $\rho_1 \neq \rho_2$ when control-group correlation $r_1 = 0.4$ and experimental-group correlation $r_2 = 0.1$ with default power of 0.8 and significance level $\alpha = 0.05$

```
power twocorrelations .4 .1
```

Same as above, for r_2 equal to -0.05, 0, 0.05, and 0.1

```
power twocorrelations .4 (-.05(.05).1)
```

Same as above, but for a one-sided test

```
power twocorrelations .4 (-.05(.05).1), onesided
```

Specify r_1 and difference $r_2 - r_1 = -0.3$

```
power twocorrelations .4, diff(-.3)
```

Same as above, but specify a sample size of 200 for group 1 and compute sample size for group 2

```
power twocorrelations 0.4, diff(-0.3) n1(200) compute(N2)
```

Power for $r_1 = 0.4$ and $r_2 = -0.15$ with a sample size of 100 and default $\alpha = 0.05$

```
power twocorrelations .4 -.15, n(100)
```

Same as above, but specify sample size of 50 and 65 for groups 1 and 2, respectively

```
power twocorrelations .4 -.15 , n1(50) n2(65)
```

Power for sample sizes of 60, 70, 80, 90, and 100

```
power twocorrelations .4 -.15, n(60(10)100)
```

Same as above, but display results in a graph of power versus sample size

```
power twocorrelations .4 -.15, n(60(10)100) graph
```

Effect size and experimental-group correlation for $r_1 = 0.4$ with sample size of 150 and power of 0.85

```
power twocorrelations .4, n(150) power(.85)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
power twocorrelations r1 r2 [ , power(numlist) options ]
```

Compute power

```
power twocorrelations r1 r2 , n(numlist) [ options ]
```

Compute effect size and experimental-group correlation

```
power twocorrelations r1 , n(numlist) power(numlist) [ options ]
```

where r_1 is the correlation in the control (reference) group and r_2 is the correlation in the experimental (comparison) group. r_1 and r_2 may each be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

options	Description
Main	
* `alpha(`*numlist*`)`	significance level; default is `alpha(0.05)`
* `power(`*numlist*`)`	power; default is `power(0.8)`
* `beta(`*numlist*`)`	probability of type II error; default is `beta(0.2)`
* `n(`*numlist*`)`	total sample size; required to compute power or effect size
* `n1(`*numlist*`)`	sample size of the control group
* `n2(`*numlist*`)`	sample size of the experimental group
* `nratio(`*numlist*`)`	ratio of sample sizes, `N2/N1`; default is `nratio(1)`, meaning equal group sizes
`compute(N1 | N2)`	solve for `N1` given `N2` or for `N2` given `N1`
`nfractional`	allow fractional sample sizes
* `diff(`*numlist*`)`	difference between the experimental-group and control-group correlations, $r_2 - r_1$; specify instead of the experimental-group correlation r_2
`direction(upper|lower)`	direction of the effect for effect-size determination; default is `direction(upper)`, which means that the postulated value of the parameter is larger than the hypothesized value
`onesided`	one-sided test; default is two sided
`parallel`	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
`[no]table[(`*tablespec*`)]`	suppress table or display results as a table; see [PSS-2] **power, table**
`saving(`*filename*`[, replace])`	save the table data to *filename*; use `replace` to overwrite existing *filename*
Graph	
`graph[(`*graphopts*`)]`	graph results; see [PSS-2] **power, graph**
Iteration	
`init(#)`	initial value for sample sizes or experimental-group correlation
`iterate(#)`	maximum number of iterations; default is `iterate(500)`
`tolerance(#)`	parameter tolerance; default is `tolerance(1e-12)`
`ftolerance(#)`	function tolerance; default is `ftolerance(1e-12)`
`[no]log`	suppress or display iteration log
`[no]dots`	suppress or display iterations as dots
`notitle`	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the `parallel` option.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`notitle` does not appear in the dialog box.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`N`	total number of subjects	N
`N1`	number of subjects in the control group	N_1
`N2`	number of subjects in the experimental group	N_2
`nratio`	ratio of sample sizes, experimental to control	N_2/N_1
`delta`	effect size	δ
`r1`	control-group correlation	ρ_1
`r2`	experimental-group correlation	ρ_2
`diff`	difference between the experimental-group correlation and the control-group correlation	$\rho_2-\rho_1$
`target`	target parameter; synonym for `r2`	
`_all`	display all supported columns	

Column `beta` is shown in the default table in place of column `power` if specified.

Columns `diff` and `nratio` are shown in the default table if specified.

Options

Main

`alpha()`, `power()`, `beta()`, `n()`, `n1()`, `n2()`, `nratio()`, `compute()`, `nfractional`; see [PSS-2] **power**.

`diff(`*numlist*`)` specifies the difference between the experimental-group correlation and the control-group correlation, $r_2 - r_1$. You can specify either the experimental-group correlation r_2 as a command argument or the difference between the two correlations in `diff()`. If you specify `diff(#)`, the experimental-group correlation is computed as $r_2 = r_1 + \#$. This option is not allowed with the effect-size determination.

`direction()`, `onesided`, `parallel`; see [PSS-2] **power**.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies the initial value for the estimated parameter. For sample-size determination, the estimated parameter is either the control-group size n_1 or, if `compute(N2)` is specified, the experimental-group size n_2. For the effect-size determination, the estimated parameter is the experimental-group correlation r_2. The default initial values for a two-sided test are obtained as a closed-form solution for the corresponding one-sided test with the significance level $\alpha/2$.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following option is available with `power twocorrelations` but is not shown in the dialog box:

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `power twocorrelations` command and the methodology for power and sample-size analysis for a two-sample correlations test. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis and [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Introduction

There are many examples of studies where a researcher may want to compare two correlations. A winemaker may want to test the null hypothesis that the correlation between fermentation time and alcohol level is the same for pinot noir grapes and merlot grapes. An education researcher may want to test the null hypothesis that the correlation of verbal and math SAT scores is the same for males and females. Or a genetics researcher may want to test the null hypothesis that the correlation of the cholesterol levels in identical twins raised together is equal to the correlation of the cholesterol levels in identical twins raised apart.

Correlation analysis emanates from studies quantifying dependence between random variables. The correlation coefficient ρ is commonly used to measure the strength of such dependence. We consider Pearson's correlation obtained by standardizing the covariance between two random variables. As a result, the correlation coefficient has no units and ranges between -1 and 1.

This entry describes power and sample-size analysis for the inference about two population correlation coefficients performed using hypothesis testing. Specifically, we consider the null hypothesis H_0: $\rho_2 = \rho_1$ versus the two-sided alternative hypothesis H_a: $\rho_2 \neq \rho_1$, the upper one-sided alternative H_a: $\rho_2 > \rho_1$, or the lower one-sided alternative H_a: $\rho_2 < \rho_1$.

One common approach in testing hypotheses concerning the correlation parameter is to use an inverse hyperbolic tangent transformation, $\tanh^{-1}(x) = 0.5\ln(1+x)/\ln(1-x)$, also known as Fisher's z transformation when applied to the correlation coefficient (Fisher 1915). Specifically, if $\hat{\rho}$ is the sample correlation coefficient and n is the sample size, Fisher (1915) showed that

$$\sqrt{n-3}\left\{\tanh^{-1}(\hat{\rho}) - \tanh^{-1}(\rho)\right\} \sim N(0,1)$$

for n as small as 10, although the approximation tends to perform better for $n > 25$. For a two-sample correlations test, the null hypothesis H_0: $\rho_1 = \rho_2$ is equivalent to H_0: $\tanh^{-1}(\rho_1) = \tanh^{-1}(\rho_2)$. The latter test is referred to as the two-sample Fisher's z test.

`power twocorrelations` performs computations based on the asymptotic two-sample Fisher's z test.

Using power twocorrelations

`power twocorrelations` computes sample size, power, or experimental-group correlation for a two-sample correlations test. All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option. You can specify the `onesided` option to request a one-sided test. By default, all computations assume a balanced or equal-allocation design; see [PSS-4] **Unbalanced designs** for a description of how to specify an unbalanced design.

To compute the total sample size, you must specify the control- and experimental-group correlations, r_1 and r_2, respectively, and, optionally, the power of the test in the `power()` option. The default power is set to 0.8.

Instead of the total sample size, you can compute one of the group sizes given the other one. To compute the control-group sample size, you must specify the `compute(N1)` option and the sample size of the experimental group in the `n2()` option. Likewise, to compute the experimental-group sample size, you must specify the `compute(N2)` option and the sample size of the control group in the `n1()` option.

To compute power, you must specify the total sample size in the `n()` option and the control- and experimental-group correlations, r_1 and r_2, respectively.

Instead of the experimental-group correlation r_2, you may specify the difference $r_2 - r_1$ between the experimental-group correlation and the control-group correlation in the `diff()` option when computing sample size or power.

To compute effect size, the difference between the experimental-group and the control-group correlation, and the experimental-group correlation, you must specify the total sample size in the `n()` option, the power in the `power()` option, the control-group correlation r_1, and optionally, the direction of the effect. The direction is upper by default, `direction(upper)`, which means that the experimental-group correlation is assumed to be larger than the specified control-group value. You can change the direction to be lower, which means that the experimental-group correlation is assumed to be smaller than the specified control-group value, by specifying the `direction(lower)` option.

Instead of the total sample size `n()`, you can specify individual group sizes in `n1()` and `n2()`, or specify one of the group sizes and `nratio()` when computing power or effect size. Also see *Two samples* in [PSS-4] **Unbalanced designs** for more details.

In the following sections, we describe the use of `power twocorrelations` accompanied with examples for computing sample size, power, and experimental-group correlation.

Computing sample size

To compute sample size, you must specify the control- and experimental-group correlations, r_1 and r_2, respectively, and, optionally, the power of the test in the `power()` option. A default power of 0.8 is assumed if `power()` is not specified.

Example 1: Sample size for a two-sample correlations test

Consider a study in which investigators are interested in testing whether the correlation between height and weight differs for males and females. The null hypothesis H_0: $\rho_F = \rho_M$ is tested against the alternative hypothesis H_a: $\rho_F \neq \rho_M$, where ρ_F is the correlation between height and weight for females and ρ_M is the correlation between height and weight for males.

Before conducting the study, investigators wish to determine the minimum sample size required to detect a difference between the correlation of 0.3 for females and a correlation of 0.5 for males with 80% power using a two-sided 5%-level test. We specify the values 0.3 and 0.5 as the control- and experimental-group correlations after the command name. We omit options `alpha(0.05)` and `power(0.8)` because the specified values are their defaults. To compute the total sample size assuming equal-group allocation, we type

```
. power twocorrelations 0.3 0.5
Performing iteration ...
Estimated sample sizes for a two-sample correlations test
Fisher's z test
H0: r2 = r1  versus  Ha: r2 != r1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.2000
           r1 =    0.3000
           r2 =    0.5000
Estimated sample sizes:
            N =       554
  N per group =       277
```

A total sample of 554 individuals, 277 in each group, must be obtained to detect the difference between correlations of females and males when the correlations are 0.3 and 0.5, respectively, with 80% power using a two-sided 5%-level test.

◁

Example 2: Computing one of the group sizes

Continuing with example 1, we will suppose that for some reason, we can enroll only 250 male subjects in our study. We want to know how many female subjects we need to recruit to maintain the 80% power for detecting the difference as described in example 1. To do so, we use the combination of `compute(N1)` and `n2()`:

```
. power twocorrelations 0.3 0.5, n2(250) compute(N1)
Performing iteration ...
Estimated sample sizes for a two-sample correlations test
Fisher's z test
H0: r2 = r1  versus  Ha: r2 != r1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.2000
           r1 =    0.3000
           r2 =    0.5000
           N2 =       250
Estimated sample sizes:
            N =       559
           N1 =       309
```

We need 309 females for a total sample size of 559 subjects, which is larger than the required total sample size for the corresponding balanced design from example 1.

◁

▷ Example 3: Unbalanced design

By default, `power twocorrelations` computes sample size for a balanced or equal-allocation design. If we know the allocation ratio of subjects between the groups, we can compute the required sample size for an unbalanced design by specifying the `nratio()` option.

Continuing with example 1, we will suppose that we anticipate to recruit twice as many males as females; that is, $n_2/n_1 = 2$. We specify the `nratio(2)` option to compute the required sample size for the specified unbalanced design.

```
. power twocorrelations 0.3 0.5, nratio(2)
Performing iteration ...
Estimated sample sizes for a two-sample correlations test
Fisher's z test
H0: r2 = r1  versus  Ha: r2 != r1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.2000
           r1 =    0.3000
           r2 =    0.5000
        N2/N1 =    2.0000
Estimated sample sizes:
            N =       624
           N1 =       208
           N2 =       416
```

Also see *Two samples* in [PSS-4] **Unbalanced designs** for more examples of unbalanced designs for two-sample tests.

◁

Computing power

To compute power, you must specify the sample size in the `n()` option and the control- and the experimental-group correlations, r_1 and r_2, respectively.

Example 4: Power of a two-sample correlations test

Continuing with example 1, we will suppose that we are designing a study and anticipate a total sample of 500 subjects. To compute the power corresponding to this sample size given the study parameters from example 1, we specify the sample size in `n()`:

```
. power twocorrelations 0.3 0.5, n(500)

Estimated power for a two-sample correlations test
Fisher's z test
H0: r2 = r1  versus  Ha: r2 != r1

Study parameters:

        alpha =    0.0500
            N =       500
  N per group =       250
        delta =    0.2000
           r1 =    0.3000
           r2 =    0.5000

Estimated power:

        power =    0.7595
```

With a smaller sample of 500 subjects compared with example 1, we obtain a power of roughly 76%.

Example 5: Multiple values of study parameters

In this example, we assess the effect of varying the value of the correlation of the male group on the power of our study. We supply a list of correlations in parentheses for the second command argument:

```
. power twocorrelations 0.3 (0.4(0.1)0.9), n(500)

Estimated power for a two-sample correlations test
Fisher's z test
H0: r2 = r1  versus  Ha: r2 != r1
```

alpha	power	N	N1	N2	delta	r1	r2
.05	.2452	500	250	250	.1	.3	.4
.05	.7595	500	250	250	.2	.3	.5
.05	.9894	500	250	250	.3	.3	.6
.05	1	500	250	250	.4	.3	.7
.05	1	500	250	250	.5	.3	.8
.05	1	500	250	250	.6	.3	.9

From the table, we see that the power increases from 25% to 100% as the correlation increases from 0.4 to 0.9.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

Computing effect size and experimental-group correlation

Effect size δ for a two-sample correlations test is defined as the difference between the experimental-group and control-group correlations, $\delta = \rho_2 - \rho_1$.

Sometimes, we may be interested in determining the smallest effect (for a given control-group correlation) that yields a statistically significant result for prespecified sample size and power. In this case, power, sample size, and control-group correlation must be specified. In addition, you must also decide on the direction of the effect: upper, meaning $\rho_2 > \rho_1$, or lower, meaning $\rho_2 < \rho_1$. The direction may be specified in the `direction()` option; `direction(upper)` is the default.

▷ Example 6: Minimum detectable value of the experimental-group correlation

Continuing with example 4, we will compute the smallest positive correlation of the male group that can be detected given a total sample of 500 individuals and a power of 80%. To solve for the experimental-group correlation, after the command, we specify the control-group correlation of `0.3`, total sample size `n(500)`, and power `power(0.8)`:

```
. power twocorrelations 0.3, n(500) power(0.8)

Performing iteration ...

Estimated experimental-group correlation for a two-sample correlations test
Fisher's z test
H0: r2 = r1  versus  Ha: r2 != r1; r2 > r1

Study parameters:

        alpha =    0.0500
        power =    0.8000
            N =       500
  N per group =       250
           r1 =    0.3000

Estimated effect size and experimental-group correlation:

        delta =    0.2092
           r2 =    0.5092
```

We find that the minimum value of the experimental-group correlation that would yield a statistically significant result in this study is 0.5092, and the corresponding effect size is 0.2092.

In this example, we computed the correlation assuming an upper direction, $\rho_2 > \rho_1$, or a positive effect, $\delta > 0$. To request a lower direction, or a negative effect, we can specify the `direction(lower)` option.

◁

Testing a hypothesis about two independent correlations

After data are collected, we can use the `mvtest` command to test the equality of two independent correlations using an asymptotic likelihood-ratio test; see [MV] **mvtest** for details. We can also manually perform a two-sample Fisher's z test on which `power twocorrelations` bases its computations. We demonstrate our examples using `genderpsych.dta` from [MV] **mvtest correlations**.

▷ Example 7: Comparing two correlations using mvtest

Consider a sample of 64 individuals with equal numbers of males and females. We would like to know whether the correlation between the pictorial inconsistencies (variable `y1`) and vocabulary (variable `y4`) is the same between males and females.

```
. use https://www.stata-press.com/data/r19/genderpsych
(Four psychological test scores, Rencher and Christensen (2012))
. mvtest correlations y1 y4, by(gender)
Test of equality of correlation matrices across samples
      Jennrich chi2(1) =      2.16
           Prob > chi2 =    0.1415
```

The reported p-value is 0.1415, so we do not have sufficient evidence to reject the null hypothesis about the equality of the two correlations.

◁

▷ Example 8: Two-sample Fisher's z test

To compute a two-sample Fisher's z test manually, we perform the following steps. We first compute the estimates of correlation coefficients for each group by using the `correlate` command; see [R] **correlate**. We then compute Fisher's z test statistic and its corresponding p-value.

We compute and store correlation estimates for males in the `r1` scalar and the corresponding sample size in the `N1` scalar.

```
. /* compute and store correlation and sample size for males */
. correlate y1 y4 if gender==1
(obs=32)
             |       y1       y4
-------------+------------------
          y1 |   1.0000
          y4 |   0.5647   1.0000

. scalar r1 = r(rho)
. scalar N1 = r(N)
```

We store the corresponding results for females in `r2` and `N2`.

```
. /* compute and store correlation and sample size for females */
. correlate y1 y4 if gender==2
(obs=32)
             |       y1       y4
-------------+------------------
          y1 |   1.0000
          y4 |   0.2596   1.0000

. scalar r2 = r(rho)
. scalar N2 = r(N)
```

We now compute the z test statistic, stored in the `Z` scalar, by applying Fisher's z transformation to the obtained correlation estimates `r1` and `r2` and using the cumulative function of the standard normal distribution `normal()` to compute the p-value.

```
. /* compute Fisher's z statistic and p-value and display results */
. scalar mu_Z    = atanh(r2) - atanh(r1)
. scalar sigma_Z = sqrt(1/(N1-3)+1/(N2-3))
. scalar Z       = mu_Z/sigma_Z
. scalar pvalue  = 2*normal(-abs(Z))
. display "Z statistic = " %8.4g Z _n "P-value     = " %8.4g pvalue
Z statistic =   -1.424
P-value     =    .1543
```

The p-value is 0.1543 and is close to the p-value reported by `mvtest` in example 7.

The estimates of the correlations obtained from `correlate` are 0.5647 for males and 0.2596 for females. We may check how many subjects we need to detect the difference between these two correlation values.

```
. power twocorrelations 0.5647 0.2596
Performing iteration ...
Estimated sample sizes for a two-sample correlations test
Fisher's z test
H0: r2 = r1  versus  Ha: r2 != r1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =   -0.3051
           r1 =    0.5647
           r2 =    0.2596
Estimated sample sizes:
            N =       232
  N per group =       116
```

We need a total of 232 subjects, 116 per group in a balanced design, to have a power of 80% to detect the difference between the given values of correlations for males and females.

◁

Stored results

`power twocorrelations` stores the following in `r()`:

Scalars

`r(alpha)`	significance level
`r(power)`	power
`r(beta)`	probability of a type II error
`r(delta)`	effect size
`r(N)`	total sample size
`r(N_a)`	actual sample size
`r(N1)`	sample size of the control group
`r(N2)`	sample size of the experimental group
`r(nratio)`	ratio of sample sizes, `N2/N1`
`r(nratio_a)`	actual ratio of sample sizes
`r(nfractional)`	1 if `nfractional` is specified, 0 otherwise
`r(onesided)`	1 for a one-sided test, 0 otherwise
`r(r1)`	control-group correlation
`r(r2)`	experimental-group correlation
`r(diff)`	difference between the experimental- and control-group correlations
`r(separator)`	number of lines between separator lines in the table
`r(divider)`	1 if `divider` is requested in the table, 0 otherwise
`r(init)`	initial value for sample sizes or experimental-group correlation
`r(maxiter)`	maximum number of iterations
`r(iter)`	number of iterations performed
`r(tolerance)`	requested parameter tolerance
`r(deltax)`	final parameter tolerance achieved
`r(ftolerance)`	requested distance of the objective function from zero
`r(function)`	final distance of the objective function from zero
`r(converged)`	1 if iteration algorithm converged, 0 otherwise

Macros

r(type)	test
r(method)	twocorrelations
r(direction)	upper or lower
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results

Methods and formulas

Let ρ_1 and ρ_2 denote Pearson's correlation for the control and the experimental groups, respectively. Let $\hat{\rho}_1$ and $\hat{\rho}_2$ denote the corresponding estimators, the sample correlation coefficients.

A two-sample correlations test involves testing the null hypothesis H_0: $\rho_2 = \rho_1$ versus the two-sided alternative hypothesis H_a: $\rho_2 \neq \rho_1$, the upper one-sided alternative H_a: $\rho_2 > \rho_1$, or the lower one-sided alternative H_a: $\rho_2 < \rho_1$.

The exact distribution of the difference between the sample correlation coefficients $\hat{\rho}_1$ and $\hat{\rho}_2$ is complicated for testing the hypothesis H_0: $\rho_1 - \rho_2 \neq 0$. An alternative is to apply an inverse hyperbolic tangent, $\tanh^{-1}(\cdot)$, or Fisher's z transformation to the estimators to approximate their sampling distribution by a normal distribution (Fisher 1915). The hypothesis H_0: $\rho_1 = \rho_2$ is then equivalent to H_0: $\tanh^{-1}(\rho_1) = \tanh^{-1}(\rho_2)$.

Let $Z_1 = \tanh^{-1}(\hat{\rho}_1) = 0.5\ln\{(1+\hat{\rho}_1)/(1-\hat{\rho}_1)\}$ denote Fisher's z transformation of $\hat{\rho}_1$, and let Z_2 denote the corresponding transformation of $\hat{\rho}_2$. Then the difference $\delta_z = Z_2 - Z_1$ follows a normal distribution with mean $\mu_z = \mu_2 - \mu_1 = \tanh^{-1}(\rho_2) - \tanh^{-1}(\rho_1)$ and standard deviation $\sigma_z = \sqrt{1/(n_1-3) + 1/(n_2-3)}$, where n_1 and n_2 are the sample sizes of the control and the experimental groups, respectively (for example, see Graybill [1961, 211] and Anderson [2003, 134]).

Let α be the significance level, β be the probability of a type II error, and $z_{1-\alpha}$ and z_β be the $(1-\alpha)$th and the βth quantiles of the standard normal distribution.

The power $\pi = 1 - \beta$ is computed using

$$\pi = \begin{cases} \Phi\left(\frac{\delta_z}{\sigma_z} - z_{1-\alpha}\right) & \text{for an upper one-sided test} \\ \Phi\left(-\frac{\delta_z}{\sigma_z} - z_{1-\alpha}\right) & \text{for a lower one-sided test} \\ \Phi\left(\frac{\delta_z}{\sigma_z} - z_{1-\alpha/2}\right) + \Phi\left(-\frac{\delta_z}{\sigma_z} - z_{1-\alpha/2}\right) & \text{for a two-sided test} \end{cases} \tag{1}$$

where $\Phi(\cdot)$ is the cdf of a standard normal distribution.

Let $R = n_2/n_1$ denote the allocation ratio. Then $n_2 = R \times n_1$ and power can be viewed as a function of n_1. Therefore, for sample-size determination, the control-group sample size n_1 is computed first. The experimental-group size n_2 is then computed as $R \times n_1$, and the total sample size is computed as $n = n_1 + n_2$. By default, sample sizes are rounded to integer values; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for details.

For a one-sided test, the control-group sample size n_1 is computed as a positive root of the following quadratic equation:

$$\frac{(R+1)n_1 - 6}{(n_1-3)(Rn_1-3)} = \left(\frac{\delta_z}{z_{1-\alpha} - z_\beta}\right)^2$$

For a one-sided test, if one of the group sizes is known, the other one is computed using the following formula. For example, to compute n_1 given n_2 we use

$$n_1 = 3 + \frac{1}{\left(\frac{\delta_z}{z_{1-\alpha}-z_\beta}\right)^2 - \frac{1}{n_2-3}}$$

The minimum detectable value of the experimental-group correlation is obtained by first finding the corresponding minimum value of μ_2 and then applying the inverse Fisher's z transformation to that μ_2:

$$\rho_2 = \frac{e^{2\mu_2}-1}{e^{2\mu_2}+1}$$

For a one-sided test, $\mu_2 = \mu_1 + \sigma_z(z_{1-\alpha} - z_\beta)$ when $\mu_2 > \mu_1$, and $\mu_2 = \mu_1 - \sigma_z(z_{1-\alpha} - z_\beta)$ when $\mu_2 < \mu_1$.

For a two-sided test, the sample sizes and μ_2 are obtained by iteratively solving the two-sided power equation in (1) for the respective parameters. The initial values are obtained from the respective formulas for the one-sided computation with the significance level $\alpha/2$.

References

Anderson, T. W. 2003. *An Introduction to Multivariate Statistical Analysis*. 3rd ed. New York: Wiley.

Fisher, R. A. 1915. Frequency distribution of the values of the correlation coefficient in samples from an indefinitely large population. *Biometrika* 10: 507–521. https://doi.org/10.2307/2331838.

Graybill, F. A. 1961. *An Introduction to Linear Statistical Models*. Vol. 1. New York: McGraw–Hill.

Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[MV] **mvtest** — Multivariate tests

[R] **correlate** — Correlations of variables

power oneway — Power analysis for one-way analysis of variance

Description

`power oneway` computes sample size, power, or effect size for one-way analysis of variance (ANOVA). By default, it computes sample size for given power and effect size. Alternatively, it can compute power for given sample size and effect size or compute effect size for given sample size, power, and number of groups. Also see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Quick start

Sample size for a test of H_0: $\mu_1 = \mu_2 = \mu_3$ for cell means of 21, 19, and 18 and within-group variance of 16 with default power of 0.8 and significance level $\alpha = 0.05$

```
power oneway 21 19 18, varerror(16)
```

Same as above, but specify a between-group variance of 1.55 for three groups

```
power oneway, varerror(16) varmeans(1.55) ngroups(3)
```

Same as above, but specify that delta (Cohen's f), given by $\sqrt{\sigma_m^2/\sigma_e^2}$, equals 0.31

```
power oneway, ngroups(3) delta(.31)
```

Same as above, but specify delta equal to 0.2, 0.25, 0.3, 0.35, and 0.4

```
power oneway, ngroups(3) delta(.2(.05).4)
```

Same as above, but show results in a graph of sample size versus effect size (delta)

```
power oneway, ngroups(3) delta(.2(.05).4) graph
```

Sample size for contrast of H_0: $\mu_1 = (\mu_2 + \mu_3)/2$ or, equivalently, H_0: $-\mu_1 + 0.5\mu_2 + 0.5\mu_3 = 0$

```
power oneway 21 19 18, varerror(16) contrast(-1 0.5 0.5)
```

For an unbalanced design where the sample size in group 1 is 1.5 times the sample sizes for groups 2 and 3

```
power oneway 21 19 18, varerror(16) grweights(3 2 2)
```

Specify group means for 5 groups using matrix `means` and within-group variance of 34

```
matrix means = (21, 19, 19, 22, 27)
power oneway means, varerror(34)
```

Power for sample sizes of 99, 108, 117, and 126 with balanced group sizes

```
power oneway 21 19 18, varerror(16) n(99(9)126)
```

Same as above, but with sample sizes of 25, 25, and 50 for groups 1, 2 and 3, respectively

```
power oneway 21 19 18, varerror(16) n1(25) n2(25) n3(50)
```

Same as above, sample size allocations treated as parallel sets, each with total sample size 100

```
power oneway 21 19 18, varerror(16) n1(50 25 25) n2(25 50 25) ///
  n3(25 25 50) parallel
```

Effect size and target between-group variance for a sample size of 150 for three groups, power of 0.9, and $\alpha = 0.01$

```
power oneway, n(150) ngroups(3) power(.9) alpha(0.1)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
power oneway meanspec [ , power(numlist) options ]
```

Compute power

```
power oneway meanspec, n(numlist) [ options ]
```

Compute effect size and target between-group variance

```
power oneway, n(numlist) power(numlist) ngroups(#) [ varerror(numlist) options ]
```

where *meanspec* is either a matrix *matname* containing group means or individual group means specified in a matrix form:

$m_1\ m_2\ [m_3\ \ldots\ m_J]$

m_j, where $j = 1, 2, \ldots, J$, is the alternative group mean or the group mean under the alternative hypothesis for the jth group. Each m_j may be specified either as one number or as a list of values in parentheses; see **[U] 11.1.8 numlist**.

matname is the name of a Stata matrix with J columns containing values of alternative group means. Multiple rows are allowed, in which case each row corresponds to a different set of J group means or, equivalently, column j corresponds to *numlist* for the jth group mean.

options	Description
Main	
* alpha(*numlist*)	significance level; default is alpha(0.05)
* power(*numlist*)	power; default is power(0.8)
* beta(*numlist*)	probability of type II error; default is beta(0.2)
* n(*numlist*)	total sample size; required to compute power or effect size
nfractional	allow fractional sample sizes
* npergroup(*numlist*)	number of subjects per group; implies balanced design
* n#(*numlist*)	number of subjects in group #
grweights(*wgtspec*)	group weights; default is one for each group, meaning equal group sizes
ngroups(#)	number of groups
* varmeans(*numlist*)	variance of the group means or between-group variance
* varerror(*numlist*)	error (within-group) variance; default is varerror(1)
contrast(*contrastspec*)	contrast specification for group means
parallel	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[no]table[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
saving(*filename* [, replace])	save the table data to *filename*; use replace to overwrite existing *filename*
Graph	
graph[(*graphopts*)]	graph results; see [PSS-2] **power, graph**
Iteration	
init(#)	initial value for sample size or effect size; default is to use a bisection algorithm to bound the solution
iterate(#)	maximum number of iterations; default is iterate(500)
tolerance(#)	parameter tolerance; default is tolerance(1e-12)
ftolerance(#)	function tolerance; default is ftolerance(1e-12)
[no]log	suppress or display iteration log
[no]dots	suppress or display iterations as dots
notitle	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the parallel option.

collect is allowed; see **[U] 11.1.10 Prefix commands**.

notitle does not appear in the dialog box.

wgtspec	Description
$\#_1\ \#_2\ \ldots\ \#_J$	J group weights. Weights must be positive and must be integers unless option `nfractional` is specified. Multiple values for each group weight $\#_j$ can be specified as a *numlist* enclosed in parentheses.
matname	matrix with J columns containing J group weights. Multiple rows are allowed, in which case each row corresponds to a different set of J weights or, equivalently, column j corresponds to a *numlist* for the jth weight.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`N`	total number of subjects	N
`N_per_group`	number of subjects per group	N/N_g
`N_avg`	average number of subjects per group	N_{avg}
`N#`	number of subjects in group #	$N_{\#}$
`delta`	effect size	δ
`N_g`	number of groups	N_g
`m#`	group mean #	$\mu_{\#}$
`Cm`	mean contrast	$C\cdot\mu$
`c0`	null mean contrast	c_0
`Var_m`	group means (between-group) variance	σ^2_m
`Var_Cm`	contrast variance	$\sigma^2_{C\mu}$
`Var_e`	error (within-group) variance	σ^2_e
`grwgt#`	group weight #	$w_{\#}$
`target`	target parameter; synonym for `Var_m` or `Var_Cm`	
`_all`	display all supported columns	

Column `beta` is shown in the default table in place of column `power` if specified.

Column `N_per_group` is available and is shown in the default table only for balanced designs.

Columns `N_avg` and `N#` are shown in the default table only for unbalanced designs.

Columns `m#` are shown in the default table only if group means are specified.

Column `Var_m` is not shown in the default table if the `contrast()` option is specified.

Columns `Cm`, `c0`, and `Var_Cm` are shown in the default table only if the `contrast()` option is specified.

Columns `grwgt#` are not shown in the default table.

Options

Main

`alpha()`, `power()`, `beta()`, `n()`, `nfractional`; see [PSS-2] **power**.

`npergroup(`*numlist*`)` specifies the group size. Only positive integers are allowed. This option implies a balanced design. `npergroup()` cannot be specified with `n()`, `n#()`, or `grweights()`.

`n#(`*numlist*`)` specifies the size of the #th group. Only positive integers are allowed. All group sizes must be specified. For example, all three options `n1()`, `n2()`, and `n3()` must be specified for a design with three groups. `n#()` cannot be specified with `n()`, `npergroup()`, or `grweights()`.

`grweights(`*wgtspec*`)` specifies J group weights for an unbalanced design. The weights may be specified either as a list of values or as a matrix, and multiple sets of weights are allowed; see *wgtspec* for details. The weights must be positive and must also be integers unless the `nfractional` option is specified. `grweights()` cannot be specified with `npergroup()` or `n#()`.

`ngroups(#)` specifies the number of groups. At least two groups must be specified. This option is required if *meanspec* is not specified. This option is also required for effect-size determination unless `grweights()` is specified.

`varmeans(`*numlist*`)` specifies the variance of the group means or the between-group variance. `varmeans()` cannot be specified with *meanspec* or `contrast()`, nor is it allowed with effect-size determination.

`varerror(`*numlist*`)` specifies the error (within-group) variance. The default is `varerror(1)`.

`contrast(`*contrastspec*`)` specifies a contrast for group means containing J contrast coefficients that must sum to zero. *contrastspec* is

$\#_1\ \#_2\ [\ \#_3\ \ldots \#_J\ ][\ ,$ `null(`*numlist*`)` `onesided` $]$

`null(`*numlist*`)` specifies the null or hypothesized value of the mean contrast. The default is `null(0)`.

`onesided` requests a one-sided t test. The default is F test.

`parallel`; see [PSS-2] **power**.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies the initial value of the sample size for the sample-size determination or the initial value of the effect size δ for the effect-size determination. The default uses a bisection algorithm to bracket the solution.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following option is available with `power oneway` but is not shown in the dialog box:

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `power oneway` command and the methodology for power and sample-size analysis for one-way ANOVA. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis and [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Introduction

The comparison of multiple group means using one-way ANOVA models is a commonly used approach in a wide variety of statistical studies. The term "one way" refers to a single factor containing an arbitrary number of groups or levels. In what follows, we will assume that the factor levels are fixed. For two groups, the ANOVA model is equivalent to an unpaired two-sample t test; see [PSS-2] **power twomeans** for the respective power and sample-size analysis. One-way ANOVA uses an F test based on the ratio of the between-group variance to the within-group variance to compare means of multiple groups.

For example, consider a type of drug with three levels of dosage in treating a medical condition. An investigator may wish to test whether the mean response of the drug is the same across all levels of dosage. This is equivalent to testing the null hypothesis H_0: $\mu_1 = \mu_2 = \mu_3$ versus the alternative hypothesis H_a: $\mu_1 \neq \mu_2$ or $\mu_1 \neq \mu_3$ or $\mu_2 \neq \mu_3$; that is, at least one of the three group means is different from all the others. Rejection of the null hypothesis, however, does not provide any specific information about the individual group means. Therefore, in some cases, investigators may want to form a hypothesis for a mean contrast, $c = \sum_{j=1}^{k} c_j\mu_j$, a linear combination of group means where weights c_j sum to zero, and compare individual means by testing a hypothesis H_0: $c = c_0$ versus H_a: $c \neq c_0$.

This entry describes power and sample-size analysis for the inference about multiple population means using hypothesis testing based on one-way ANOVA. Specifically, we consider the null hypothesis H_0: $\mu_1 = \cdots = \mu_J$, which tests the equality of J group means against the alternative hypothesis H_a: $\mu_i \neq \mu_j$for some i, j. The test statistic for this hypothesis is based on the ratio of the between-group variance to the within-group variance and has an F distribution under the null hypothesis. The corresponding test is known as an overall F test, which tests the equality of multiple group means. This test is nondirectional.

For testing a single mean contrast, H_0: $c = c_0$ versus H_a: $c \neq c_0$, a test statistic is a function of the ratio of the contrast variance to the error (or within-group) variance, and either an F test or a t test can be used for a two-sided alternative. For a one-sided alternative, H_a: $c > c_0$ or H_a: $c < c_0$, only a t test can be used.

Power and sample-size computations use the distribution of the test statistic under the alternative hypothesis, which is a noncentral F distribution for the considered tests. Power is a function of the noncentrality parameter, and the noncentrality parameter is a function of the ratio of the standard deviation of the tested effect to the standard deviation of the errors. As such, the effect size for the overall F test is defined as the square root of the ratio of the between-group variance to the within-group variance. For testing a mean contrast, the effect size is defined as the square root of the contrast variance to the error or within-group variance.

Using power oneway

power oneway computes sample size, power, or effect size and target variance of the effect for a one-way fixed-effects analysis of variance. All computations are performed assuming a significance level of 0.05. You may change the significance level by specifying the alpha() option.

By default, the computations are performed for an overall F test testing the equality of all group means. The within-group or error variance for this test is assumed to be 1 but may be changed by specifying the varerror() option.

To compute the total sample size, you must specify the alternative *meanspec* and, optionally, the power of the test in the power() option. The default power is set to 0.8.

To compute power, you must specify the total sample size in the n() option and the alternative *meanspec*.

Instead of the alternative group means, you can specify the number of groups in the ngroups() option and the variance of the group means (or the between-group variance) in the varmeans() option when computing sample size or power.

To compute effect size, the square root of the ratio of the between-group variance to the error variance, and the target between-group variance, you must specify the total sample size in the n() option, the power in the power() option, and the number of groups in the ngroups() option.

To compute sample size or power for a test of a mean contrast, in addition to the respective options power() or n() as described above, you must specify the alternative *meanspec* and the corresponding contrast coefficients in the contrast() option. A contrast coefficient must be specified for each of the group means, and the specified coefficients must sum to zero. The null value for the specified contrast is assumed to be zero but may be changed by specifying the null() suboption within contrast(). The default test is an F test. You can instead request a one-sided t test by specifying the onesided suboption within contrast(). Effect-size determination is not available when testing a mean contrast.

For all the above computations, the error or within-group variance is assumed to be 1. You can change this value by specifying the varerror() option.

By default, all computations assume a balanced- or equal-allocation design. You can use the grweights() option to specify an unbalanced design for power, sample-size, or effect-size computations. For power and effect-size computations, you can specify individual group sizes in options n1(), n2(), and so on instead of a combination of n() and grweights() to accommodate an unbalanced design. For a balanced design, you can also specify the npergroup() option to specify a group size instead of a total sample size in n().

In a one-way analysis of variance, sample size and effect size depend on the noncentrality parameter of the F distribution, and their estimation requires iteration. The default initial values are obtained from a bisection search that brackets the solution. If you desire, you may change this by specifying your own value in the init() option. See [PSS-2] **power** for the descriptions of other options that control the iteration procedure.

Alternative ways of specifying effect

To compute power or sample size, you must specify the magnitude of the effect desired to be detected by the test. With `power oneway`, you can do this by specifying either the individual alternative *meanspec*, for example,

`power oneway` $m_1\ m_2 \ldots m_J$`, ...`

or the variance of J group means (between-group variance) and the number of groups J:

`power oneway, varmeans(#) ngroups(#)` [...]

You can also specify multiple values for variances in `varmeans()`.

There are multiple ways in which you can supply the group means to `power oneway`.

As we showed above, you may specify group means following the command line as

`power oneway` $m_1\ m_2 \ldots m_J$`, ...`

At least two means must be specified.

Alternatively, you can define a Stata matrix as a row or a column vector and use it with `power oneway`. The dimension of the Stata matrix must be at least 2. For example,

`matrix define` *meanmat* `=` (m_1`,` m_2`, . . .,` m_J)

`power oneway` *meanmat*`, ...`

You can also specify multiple values or *numlist* for each of the group means in parentheses:

`power oneway` ($m_{1,1}\ m_{1,2} \ldots m_{1,K_1}$) ($m_{2,1}\ m_{2,2} \ldots m_{2,K_2}$) `...,` ...

Each of the *numlist*s may contain different numbers of values, $K_1 \neq K_2 \neq \ldots \neq K_J$. `power oneway` will produce results for all possible combinations of values across *numlist*s. If instead you would like to treat each specification as a separate scenario, you can specify the `parallel` option.

Similarly, you can accommodate multiple sets of group means in a matrix form by adding a row for each specification. The columns of a matrix with multiple rows correspond to J group means, and values within each column j correspond to multiple specifications of the jth group mean or *numlist* for the jth group mean.

For example, the following two specifications are the same:

`power oneway` ($m_{1,1}\ m_{1,2}\ m_{1,3}$) ($m_{2,1}\ m_{2,2}\ m_{2,3}$)`, ...`

and

`matrix define` *meanmat* `=` ($m_{1,1}$`,` $m_{2,1}$ `\` $m_{1,2}$`,` $m_{2,2}$ `\` $m_{1,3}$`,` $m_{2,3}$)

`power oneway` *meanmat*`, ...`

In the above specification, if you wish to specify a *numlist* only for the first group, you may define your matrix as

`matrix define` *meanmat* `=` ($m_{1,1}$`,` $m_{2,1}$ `\` $m_{1,2}$`, . \` $m_{1,3}$`, .`)

and the results of

`power oneway` *meanmat*`, ...`

will be the same as the results of

power oneway ($m_{1,1}$ $m_{1,2}$ $m_{1,3}$) $m_{2,1}$, ...

If you wish to treat the rows of *meanmat* as separate scenarios, you must specify the parallel option.

In the following sections, we describe the use of power oneway accompanied by examples for computing sample size, power, and effect size.

Computing sample size

To compute sample size, you must specify the alternative group means or their variance and, optionally, the power of the test in the power() option. A power of 0.8 is assumed if power() is not specified.

▷ Example 1: Sample size for a one-way ANOVA

Consider an example from van Belle et al. (2004, 367), where the authors report the results of a study of the association between cholesterol level and the number of diseased blood vessels, which indicate the presence of coronary artery disease, in patients undergoing coronary bypass surgery. Suppose we wish to plan a similar new study in which a cholesterol level is considered a risk factor that may be associated with the number of diseased blood vessels, our grouping variable. We consider three groups of subjects with 1, 2, or 3 numbers of diseased blood vessels. We would like to know how many subjects we need to observe in each group to detect differences between cholesterol levels across groups. Our projected cholesterol levels in three groups are 260, 289, and 295 mg/dL, respectively. Suppose that we anticipate the within-group or error variance is 4,900. To compute the total sample size for power oneway's default setting of a balanced design with 5% significance level and 80% power, we type

```
. power oneway 260 289 295, varerror(4900)

Performing iteration ...

Estimated sample size for one-way ANOVA
F test for group effect
H0: delta = 0  versus  Ha: delta != 0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.2183
          N_g =         3
           m1 =  260.0000
           m2 =  289.0000
           m3 =  295.0000
        Var_m =  233.5556
        Var_e = 4900.0000

Estimated sample sizes:

            N =       207
  N per group =        69
```

We find that a total sample of 207 subjects, 69 subjects per group, is required to detect a change in average cholesterol levels in at least 1 of the 3 groups for this study.

In addition to the specified and implied study parameters, power oneway reports the value of the effect size, delta $= \sqrt{233.556/4900} = 0.2183$, computed as a square root of the ratio between the variance of the group means Var_m and the error variance Var_e. The two variances are also often referred to as the between-group variance and the within-group variance, respectively. The effect size δ

provides a unitless measure of the magnitude of an effect with a lower bound of zero meaning no effect. It corresponds to Cohen's effect-size measure f (Cohen 1988). Cohen's convention is that $f = 0.1$ means small effect size, $f = 0.25$ means medium effect size, and $f = 0.4$ means large effect size. According to this convention, the effect size considered in our example is medium.

◁

▷ Example 2: Alternative ways of specifying effect

Instead of specifying the alternative means as in example 1, we can specify the variance between them and the number of groups. From example 1, the variance between the group means was computed to be 233.5556. We specify this value in `varmeans()` as well as the number of groups in `ngroups()`:

```
. power oneway, varmeans(233.5556) ngroups(3) varerror(4900)

Estimated sample size for one-way ANOVA
F test for group effect
H0: delta = 0  versus  Ha: delta != 0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.2183
          N_g =         3
        Var_m =  233.5556
        Var_e = 4900.0000

Estimated sample sizes:

            N =       207
  N per group =        69
```

We obtain the exact same results as in example 1.

Instead of specifying alternative means directly following the command line, we can define a matrix, say, `means`, containing these means and use it with `power oneway`:

```
. matrix define means = (260,289,295)

. power oneway means, varerror(4900)
  (output omitted)
```

You can verify that the results are identical to the previous results.

◁

▷ Example 3: Computing sample size for a mean contrast

Continuing with example 1, suppose we would like to test whether the average of the first two cholesterol levels is different from the cholesterol level of the third group. We construct the following contrast $(0.5, 0.5, -1)$ to test this hypothesis. To compute sample size, we specify the contrast coefficients in the `contrast()` option:

```
. power oneway 260 289 295, varerror(4900) contrast(.5 .5 -1)
Performing iteration ...
Estimated sample size for one-way ANOVA
F test for contrast of means
H0: Cm = 0  versus  Ha: Cm != 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.1381
          N_g =         3
           m1 =  260.0000
           m2 =  289.0000
           m3 =  295.0000
          C*m =  -20.5000
           c0 =    0.0000
       Var_Cm =   93.3889
        Var_e = 4900.0000
Estimated sample sizes:
            N =       414
  N per group =       138
```

The required sample size to achieve this study objective is 414 with 138 subjects per group.

For a test of a mean contrast, we can also test a directional hypothesis by specifying the `onesided` option within `contrast()`. In this case, the computation is based on the t test instead of the F test.

For example, to test whether the average of the first two cholesterol levels is less than the cholesterol level of the third group, we type

```
. power oneway 260 289 295, varerror(4900) contrast(.5 .5 -1, onesided)
Performing iteration ...
Estimated sample size for one-way ANOVA
t test for contrast of means
H0: Cm = 0  versus  Ha: Cm < 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =   -0.1381
          N_g =         3
           m1 =  260.0000
           m2 =  289.0000
           m3 =  295.0000
          C*m =  -20.5000
           c0 =    0.0000
       Var_Cm =   93.3889
        Var_e = 4900.0000
Estimated sample sizes:
            N =       327
  N per group =       109
```

The results show that the required sample size reduces to a total of 327 subjects for this lower one-sided hypothesis.

The default null value for the contrast is zero, but you can change this by specifying `contrast()`'s suboption `null()`.

▷ Example 4: Unbalanced design

Continuing with example 1, suppose we anticipate that the first group will have twice as many subjects as the second and the third groups. We can accommodate this unbalanced design by specifying the corresponding group weights in the `grweights()` option:

```
. power oneway 260 289 295, varerror(4900) grweights(2 1 1)
Performing iteration ...
Estimated sample size for one-way ANOVA
F test for group effect
H0: delta = 0  versus  Ha: delta != 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.2306
          N_g =         3
           m1 =  260.0000
           m2 =  289.0000
           m3 =  295.0000
        Var_m =  260.5000
        Var_e = 4900.0000
Estimated sample sizes:
            N =       188
    Average N =   62.6667
           N1 =         94
           N2 =         47
           N3 =         47
```

The required total sample size for this unbalanced design is 188 with 94 subjects in the first group and 47 subjects in the second and third groups. The average number of subjects per group is 62.67.

We can compute results for multiple sets of group weights. The specification of group weights within `grweights()` is exactly the same as the specification of group means described in *Alternative ways of specifying effect*. Suppose that we would like to compute sample sizes for two unbalanced designs. The first design has twice as many subjects in the first group as the other two groups. The second design has the first two groups with twice as many subjects as the third group. We specify multiple group weights for the first and second groups in parentheses. We also specify the `parallel` option to treat multiple weight values in parallel instead of computing results for all possible combinations of these values, which would have been done by default.

```
. local tabcols alpha power N N1 N2 N3 grwgt1 grwgt2 grwgt3 Var_m Var_e
. power oneway 260 289 295, varerror(4900) grweights((2 2) (1 2) 1) parallel
> table(‘tabcols’, formats("%6.0g"))
Performing iteration ...
Estimated sample size for one-way ANOVA
F test for group effect
H0: delta = 0  versus  Ha: delta != 0
```

alpha	power	N	N1	N2	N3	grwgt1	grwgt2	grwgt3	Var_m	Var_e
.05	.8	188	94	47	47	2	1	1	260.5	4900
.05	.8	205	82	82	41	2	2	1	235.4	4900

The default table does not include group weights, so we request a table with custom columns containing group weights via `table()`. We also request a smaller format to make the table more compact.

◁

Computing power

To compute power, you must specify the total sample size in the `n()` option and the alternative group means or their variance.

▷ Example 5: Power for a one-way ANOVA

Continuing with example 1, suppose that we anticipate obtaining a total sample of 300 subjects. To compute the corresponding power, we specify the sample size of 300 in `n()`:

```
. power oneway 260 289 295, n(300) varerror(4900)

Estimated power for one-way ANOVA
F test for group effect
H0: delta = 0  versus  Ha: delta != 0

Study parameters:

        alpha =    0.0500
            N =       300
  N per group =       100
        delta =    0.2183
          N_g =         3
           m1 =  260.0000
           m2 =  289.0000
           m3 =  295.0000
       Var_m =  233.5556
       Var_e = 4900.0000

Estimated power:

        power =    0.9308
```

The power increases to 93.08% for a larger sample of 300 subjects.

◁

▷ Example 6: Multiple values of study parameters

Continuing with example 5, we may want to check powers for several sample sizes. We simply list multiple sample-size values in `n()`:

```
. power oneway 260 289 295, n(100 200 300) varerror(4900)
> table(, labels(N_per_group "N/N_g") formats("%6.2g"))

Estimated power for one-way ANOVA
F test for group effect
H0: delta = 0  versus  Ha: delta != 0
```

alpha	power	N	N/N_g	delta	N_g	m1	m2	m3	Var_m	Var_e
.05	.47	100	33	.22	3	260	289	295	234	4900
.05	.78	200	66	.22	3	260	289	295	234	4900
.05	.93	300	100	.22	3	260	289	295	234	4900

To shorten our default table, we specified a shorter label for the `N_per_group` column and reduced the default display format for all table columns by specifying the corresponding options within the `table()` option.

We can compute results for multiple values of group means. For example, to see how power changes when the first group mean takes values of 245, 260, and 280, we specify these values in parentheses:

```
. power oneway (245 260 280) 289 295, n(300) varerror(4900)
> table(, labels(N_per_group "N/N_g") formats("%6.2g"))

Estimated power for one-way ANOVA
F test for group effect
H0: delta = 0  versus  Ha: delta != 0
```

alpha	power	N	N/N_g	delta	N_g	m1	m2	m3	Var_m	Var_e
.05	1	300	100	.32	3	245	289	295	497	4900
.05	.93	300	100	.22	3	260	289	295	234	4900
.05	.25	300	100	.088	3	280	289	295	38	4900

We can compute results for a combination of multiple sample sizes and multiple mean values or a combination of multiple values of other study parameters.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

Computing effect size and between-group variance

Sometimes, we may be interested in determining the smallest effect that yields a statistically significant result for prespecified sample size and power. In this case, power, sample size, and the number of groups must be specified.

The effect size is defined as a square root of the ratio of the variance of the tested effect, for example, the between-group variance, to the error variance. Both the effect size and the target between-group variance are computed.

The effect-size determination is not available for testing a mean contrast.

▷ Example 7: Effect size for a one-way analysis of variance

Continuing with example 5, we now want to compute the effect size that can be detected for a sample of 300 subjects and a power of 80%. We specify both parameters in the respective options. For the effect-size determination, we must also specify the number of groups in `ngroups()`:

```
. power oneway, varerror(4900) n(300) power(0.80) ngroups(3)

Performing iteration ...

Estimated between-group variance for one-way ANOVA
F test for group effect
H0: delta = 0  versus  Ha: delta != 0

Study parameters:

        alpha =    0.0500
        power =    0.8000
            N =       300
  N per group =       100
          N_g =         3
        Var_e = 4900.0000

Estimated effect size and between-group variance:

        delta =    0.1801
        Var_m =  158.9648
```

For a larger sample size, given the same power, we can detect a smaller effect size, 0.18, compared with the effect size of 0.22 from example 1. The corresponding estimate of the between-group variance is 158.96, given the error variance of 4,900.

◁

Testing hypotheses about multiple group means

There are several ways in which you can compare group means of a single factor on the collected data. Two commonly used commands to do this are `oneway` (or `anova`) and `contrast`. We demonstrate a quick use of these commands here using the systolic blood pressure example; see [R] **oneway** and [R] **contrast** for more examples. Also see [R] **anova** for general ANOVA models.

▷ Example 8: One-way ANOVA

Consider `systolic.dta` containing 58 patients undergoing 4 different drug treatments for reducing systolic blood pressure. The `systolic` variable records the change in systolic blood pressure and the `drug` variable records four treatment levels. We would like to test whether the average change in systolic blood pressure is the same for all treatments. We use `oneway` to do this:

```
. use https://www.stata-press.com/data/r19/systolic
(Systolic blood pressure data)
. oneway systolic drug
                        Analysis of variance
    Source              SS         df      MS            F     Prob > F
------------------------------------------------------------------------
Between groups      3133.23851      3   1044.41284      9.09     0.0001
 Within groups      6206.91667     54   114.942901
------------------------------------------------------------------------
    Total           9340.15517     57   163.862371
Bartlett's equal-variances test: chi2(3) =   1.0063    Prob>chi2 = 0.800
```

We reject the null hypothesis that all treatment means are equal at the 5% significance level; the p-value is less than 0.0001.

Suppose we wish to design a new similar study. We use the estimates from this study to perform a sample-size analysis for our new study. First, we estimate the means of systolic blood pressure for different treatment levels:

```
. mean systolic, over(drug)
Mean estimation                                 Number of obs = 58
------------------------------------------------------------------
                |       Mean   Std. err.      [95% conf. interval]
----------------+-------------------------------------------------
c.systolic@drug |
              1 |   26.06667   3.014989       20.02926    32.10408
              2 |   25.53333   2.999788       19.52636    31.54031
              3 |       8.75   2.892323       2.958224    14.54178
              4 |       13.5   2.330951       8.832351    18.16765
------------------------------------------------------------------
```

From the `oneway` output, the estimate of the error variance is roughly 115. Second, we specify the means and the error variance with `power oneway` and compute the required sample size for a balanced design assuming 5% significance level and 90% power.

```
. power oneway 26.07 25.53 8.75 13.5, varerror(115) power(.9)
Performing iteration ...
Estimated sample size for one-way ANOVA
F test for group effect
H0: delta = 0  versus  Ha: delta != 0
Study parameters:
        alpha =    0.0500
        power =    0.9000
        delta =    0.7021
          N_g =         4
           m1 =   26.0700
           m2 =   25.5300
           m3 =    8.7500
           m4 =   13.5000
        Var_m =   56.6957
        Var_e =  115.0000
Estimated sample sizes:
            N =        36
  N per group =         9
```

The effect size for this ANOVA model specification is rather large, 0.7021. So we need only 36 subjects, 9 per group, to detect the effect of this magnitude with 90% power.

Suppose that in addition to testing the overall equality of treatment means, we are interested in testing a specific hypothesis of whether the average of the first two treatment means is equal to the average of the last two treatment means.

To perform this test on the collected data, we can use the `contrast` command. The `contrast` command is not available after `oneway`, so we repeat our one-way ANOVA analysis using the `anova` command before using `contrast`; see [R] **contrast** for details.

```
. anova systolic i.drug
                          Number of obs =         58    R-squared     =  0.3355
                          Root MSE      =    10.7211    Adj R-squared =  0.2985
```

Source	Partial SS	df	MS	F	Prob>F
Model	3133.2385	3	1044.4128	9.09	0.0001
drug	3133.2385	3	1044.4128	9.09	0.0001
Residual	6206.9167	54	114.9429		
Total	9340.1552	57	163.86237		

```
. contrast {drug .5 .5 -.5 -.5}
Contrasts of marginal linear predictions
Margins: asbalanced
```

	df	F	P>F
drug	1	26.85	0.0000
Denominator	54		

	Contrast	Std. err.	[95% conf.	interval]
drug (1)	14.675	2.832324	8.996533	20.35347

As with the overall equality test, we find statistical evidence to reject this hypothesis as well.

To compute the required sample size for this hypothesis, we specify the contrast coefficients in the `contrast()` option of `power oneway`:

```
. power oneway 26.07 25.53 8.75 13.5, varerror(115) power(.9)
> contrast(.5 .5 -.5 -.5)
Performing iteration ...
Estimated sample size for one-way ANOVA
F test for contrast of means
H0: Cm = 0  versus  Ha: Cm != 0
Study parameters:
        alpha =    0.0500
        power =    0.9000
        delta =    0.6842
          N_g =         4
           m1 =   26.0700
           m2 =   25.5300
           m3 =    8.7500
           m4 =   13.5000
          C*m =   14.6750
           c0 =    0.0000
       Var_Cm =   53.8389
        Var_e =  115.0000
Estimated sample sizes:
            N =        28
  N per group =         7
```

The required sample size is 28 subjects with 7 subjects per group, which is smaller than the required sample size computed earlier for the overall test of the equality of means.

◁

Video examples

Sample-size calculation for one-way analysis of variance

Power calculation for one-way analysis of variance

Minimum detectable effect size for one-way analysis of variance

Stored results

power oneway stores the following in r():

Scalars

r(alpha)	significance level
r(power)	power
r(beta)	probability of a type II error
r(delta)	effect size
r(N)	total sample size
r(N_a)	actual sample size
r(N_avg)	average sample size
r(N#)	number of subjects in group #
r(N_per_group)	number of subjects per group
r(N_g)	number of groups
r(nfractional)	1 if nfractional is specified, 0 otherwise
r(balanced)	1 for a balanced design, 0 otherwise
r(grwgt#)	group weight #
r(onesided)	1 for a one-sided test of a mean contrast, 0 otherwise
r(m#)	group mean #
r(Cm)	mean contrast
r(c0)	null mean contrast
r(Var_m)	group-means (between-group) variance
r(Var_Cm)	contrast variance
r(Var_e)	error (within-group) variance
r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise
r(init)	initial value for the sample size or effect size
r(maxiter)	maximum number of iterations
r(iter)	number of iterations performed
r(tolerance)	requested parameter tolerance
r(deltax)	final parameter tolerance achieved
r(ftolerance)	requested distance of the objective function from zero
r(function)	final distance of the objective function from zero
r(converged)	1 if iteration algorithm converged, 0 otherwise

Macros

r(type)	test
r(method)	oneway
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results

Methods and formulas

Consider a single factor A with J groups or levels, where each level comprises n_j observations for $j = 1, \ldots, J$. The total number of observations is $n = \sum_{j=1}^{J} n_j$. Let Y_{ij} denote the response for the jth level of the ith individual and μ_j denote the factor-level or group means. Individual responses from J populations are assumed to be normally distributed with mean μ_j and a constant variance σ_e^2.

The hypothesis of an overall F test of the equality of group means is

$$H_0\colon \mu_1 = \cdots = \mu_J$$

versus

$$H_a\colon \mu_j\text{'s are not all equal} \tag{1}$$

The hypothesis for a test of a mean contrast is

$$H_0\colon \sum_{j=1}^{J} c_j\mu_j = c_0 \tag{2}$$

versus a two-sided $H_a\colon \sum_{j=1}^{J} c_j\mu_j \neq c_0$, upper one-sided $H_a\colon \sum_{j=1}^{J} c_j\mu_j > c_0$, and lower one-sided $H_a\colon \sum_{j=1}^{J} c_j\mu_j < c_0$, where c_j's are contrast coefficients such that $\sum_{i=1}^{J} c_j = 0$ and c_0 is a null value of a mean constant. The two-sided hypothesis is tested using an F test, and one-sided hypotheses are tested using a t test.

Hypotheses (1) and (2) can be tested in a general linear model framework. Consider a linear model

$$\mathbf{y} = \mathbf{Xb} + \boldsymbol{\epsilon}$$

where $\mathbf{y}$ is an $n \times 1$ vector of observations, $\mathbf{X}$ is an $n \times p$ matrix of predictors, $\mathbf{b}$ is a $p \times 1$ vector of unknown and fixed coefficients, and $\boldsymbol{\epsilon}$ is an $n \times 1$ vector of error terms that are independent and identically distributed as $N(0, \sigma_e^2)$.

For the one-way model, $p = J$ and the contents of $\mathbf{b} = (b_1, \ldots, b_J)'$ are the μ_j, $j = 1, \ldots, J$.

A general linear hypothesis in this framework is given by

$$\mathbf{Cb} = \mathbf{c_0}$$

where $\mathbf{C}$ is a $\nu \times p$ matrix with $\text{rank}(\mathbf{C}) = \nu \leq p$, and $\mathbf{c_0}$ is a vector of constants. For an overall test of the means in (1), $\mathbf{c_0} = 0$. The estimates of $\mathbf{b}$ and σ_e^2, respectively, are given by

$$\hat{\mathbf{b}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$$
$$\hat{\sigma}_e^2 = (\mathbf{y} - \mathbf{X}\hat{\mathbf{b}})'(\mathbf{y} - \mathbf{X}\hat{\mathbf{b}})/(n-p)$$

A general test statistic for testing hypotheses (1) and (2) is given by

$$F_C = \frac{SS_C}{(p-1)\hat{\sigma}_e^2} \tag{3}$$

where $SS_C = (\mathbf{C}\hat{\mathbf{b}} - \mathbf{c_0})'\{\mathbf{C}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{C}'\}^{-1}(\mathbf{C}\hat{\mathbf{b}} - \mathbf{c_0})$. Let α be the significance level, β be the probability of a type II error, and $F_{p-1,n-p,1-\alpha}$ be the $(1-\alpha)$th quantile of an F distribution with $p-1$ numerator and $n-p$ denominator degrees of freedom. We reject the null hypothesis if we observe a statistic $F_C > F_{p-1,n-p,1-\alpha}$.

The test statistic in (3) under the alternative hypothesis is distributed as a noncentral F distribution with $p-1$ numerator and $n-p$ denominator degrees of freedom with a noncentrality parameter λ given by

$$\begin{aligned}\lambda &= (\mathbf{Cb}-\mathbf{c_0})'\{\mathbf{C}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{C}'\}^{-1}(\mathbf{Cb}-\mathbf{c_0})/\sigma_e^2 \\ &= n(\mathbf{Cb}-\mathbf{c_0})'\{\mathbf{C}(\ddot{\mathbf{X}}'\mathbf{W}\ddot{\mathbf{X}})^{-1}\mathbf{C}'\}^{-1}(\mathbf{Cb}-\mathbf{c_0})/\sigma_e^2 \\ &= n\delta^2\end{aligned}$$

where the matrix $\ddot{\mathbf{X}}$ contains the unique rows of $\mathbf{X}$ and $\mathbf{W} = \text{diag}(w_1, \ldots, w_p)$. We define δ as the effect size.

For the one-way design, the dimension of $\ddot{\mathbf{X}}$ is $J \times J$. The weights are formally $w_j = n_j/n$ but can also be expressed in terms of the group weights (specified in `grweights()`), normalized by the sum of the group weights, making w_j independent of n. Specifically, let the group weights be denoted $\widetilde{w}_j$, then define a cell sample size multiplier as $n_c = n/\sum_j \widetilde{w}_j$ so that $n_j = n_c\widetilde{w}_j$. The cell-means parameterization simplifies $\ddot{\mathbf{X}}$ to the identity matrix, $\mathbf{I}_J$.

See O'Brien and Muller (1993) for details.

The power of the overall F test in (1) is given by

$$1-\beta = F_{\nu,n-p,\lambda}\left(F_{\nu,n-p,1-\alpha}\right) \tag{4}$$

where $F_{\cdot,\cdot,\lambda}(\cdot)$ is the cdf of a noncentral F distribution.

Total sample size and effect size are obtained by iteratively solving the nonlinear equation (4). When the `grweights()` option is specified, a constant multiplier n_c is computed and rounded to an integer unless the `nfractional` option is specified. The group sizes are then computed as $\widetilde{w}_j n_c$. The actual sample size, `N_a`, is the sum of the group sizes.

See Kutner et al. (2005) for details.

The power of the test (2) for a mean contrast is given by

$$1-\beta = \begin{cases} 1-T_{n-1,\tilde{\lambda}}\left(t_{n-1,1-\alpha}\right) & \text{for an upper one-sided test} \\ T_{n-1,\tilde{\lambda}}\left(-t_{n-1,1-\alpha}\right) & \text{for a lower one-sided test} \\ F_{1,n-p,\lambda}\left(F_{1,n-p,1-\alpha}\right) & \text{for a two-sided test} \end{cases} \tag{5}$$

where $T_{\cdot,\tilde{\lambda}}(\cdot)$ is the cumulative of a noncentral Student's t distribution with the noncentrality parameter $\tilde{\lambda}$ given by

$$\begin{aligned}\tilde{\lambda} &= \sqrt{n}(\mathbf{Cb}-\mathbf{c_0})'\sqrt{\left\{\mathbf{C}(\ddot{\mathbf{X}}'\mathbf{W}\ddot{\mathbf{X}})^{-1}\mathbf{C}'\right\}^{-1}/\sigma_e^2} \\ &= \sqrt{n}\tilde{\delta}\end{aligned}$$

Sample size is obtained by iteratively solving the nonlinear equation (5).

References

Cohen, J. 1988. *Statistical Power Analysis for the Behavioral Sciences*. 2nd ed. Hillsdale, NJ: Erlbaum.

Kutner, M. H., C. J. Nachtsheim, J. Neter, and W. Li. 2005. *Applied Linear Statistical Models*. 5th ed. New York: McGraw–Hill.

O'Brien, R. G., and K. E. Muller. 1993. "Unified power analysis for t-tests through multivariate hypotheses". In *Applied Analysis of Variance in Behavioral Science*, edited by L. K. Edwards, 297–344. New York: Dekker.

van Belle, G., L. D. Fisher, P. J. Heagerty, and T. S. Lumley. 2004. *Biostatistics: A Methodology for the Health Sciences*. 2nd ed. New York: Wiley.

Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power repeated** — Power analysis for repeated-measures analysis of variance

[PSS-2] **power twomeans** — Power analysis for a two-sample means test

[PSS-2] **power twoway** — Power analysis for two-way analysis of variance

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[R] **anova** — Analysis of variance and covariance

[R] **contrast** — Contrasts and linear hypothesis tests after estimation

[R] **oneway** — One-way analysis of variance

power twoway — Power analysis for two-way analysis of variance

Description

`power twoway` computes sample size, power, or effect size for two-way analysis of variance (ANOVA). By default, it computes sample size for given power and effect size. Alternatively, it can compute power for given sample size and effect size or compute effect size for given sample size, power, and number of cells. You can choose between testing for main row or column effect or their interaction. Also see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Quick start

Sample size for the main effect of the row factor for a 2×3 design specified using cell means with a within-cell variance of 27 and default power of 0.8 and significance level $\alpha = 0.05$

```
power twoway 19 18 32 \ 23 25 26, varerror(27)
```

Same as above, but specify cell means in the matrix `cm`

```
matrix cm = (19, 18, 32 \ 23, 25, 26)
power twoway cm, varerror(27)
```

Same as above

```
power twoway cm, varerror(27) factor(row)
```

Same as above, but calculate sample size for the main effect of the column factor

```
power twoway cm, varerror(27) factor(column)
```

Same as above, but calculate sample size for the interaction of the row and column factors

```
power twoway cm, varerror(27) factor(rowcol)
```

Same as above, but for within-cell variances of 20, 25, 30, and 35

```
power twoway cm, varerror(20(5)35) factor(rowcol)
```

Sample size for the row factor with power of 0.85 and $\alpha = 0.01$

```
power twoway cm, varerror(27) power(.85) alpha(.01)
```

Specify that the groups in the second row have twice the sample size as those in the first row

```
power twoway cm, varerror(27) cellweights(1 1 1\2 2 2) showcellsizes
```

Sample size when variance of the main effect of the row factor equals 1.2 in a 2×3 design

```
power twoway, varerror(27) factor(row) vareffect(1.2) ///
  nrows(2) ncols(3)
```

Power for the main effect of the row factor with a sample size of 180 for cell means stored in matrix `cm`

```
power twoway cm, varerror(27) factor(row) n(180)
```

Same as above, but specify that the sample size per cell is 40

```
power twoway cm, varerror(27) factor(row) npercell(40)
```

Same as above, but specify cell sample sizes of 40, 45, 50, and 55

```
power twoway cm, varerror(27) factor(row) npercell(40(5)55)
```

Same as above, but display results as a graph of power versus sample size

```
power twoway cm, varerror(27) factor(row) npercell(40(5)55) graph
```

Effect size and target between-group variance for three groups, sample size of 150, and power of 0.8

```
power twoway, varerror(27) nrows(2) ncols(3) n(150) power(.8)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

`power twoway` *meanspec* [, `power(`*numlist*`)` *options*]

Compute power

`power twoway` *meanspec*`, n(`*numlist*`)` [*options*]

Compute effect size and target effect variance

`power twoway, n(`*numlist*`) power(`*numlist*`) nrows(#) ncols(#)` [*options*]

where *meanspec* is either a matrix *matname* containing cell means or individual cell means in a matrix form:

$m_{1,1}\ m_{1,2}\ [\ldots]\ \backslash\ m_{2,1}\ m_{2,2}\ [\ldots]\ [\backslash\ \ldots \backslash\ m_{J,1}\ \ldots m_{J,K}\,]$

m_{jk}, where $j = 1, 2, \ldots, J$ and $k = 1, 2, \ldots, K$, is the alternative cell mean or the cell mean of the jth row and kth column under the alternative hypothesis.

matname is the name of a Stata matrix with J rows and K columns containing values of alternative cell means.

options	Description
Main	
* alpha(*numlist*)	significance level; default is alpha(0.05)
* power(*numlist*)	power; default is power(0.8)
* beta(*numlist*)	probability of type II error; default is beta(0.2)
* n(*numlist*)	total sample size; required to compute power or effect size
nfractional	allow fractional sample sizes
* npercell(*numlist*)	number of subjects per cell; implies balanced design
cellweights(*wgtspec*)	cell weights; default is one for each cell, meaning equal cell sizes
nrows(#)	number of rows
ncols(#)	number of columns
factor(row | column | rowcol)	tested effect
* vareffect(*numlist*)	variance explained by the tested effect in factor()
* varrow(*numlist*)	variance explained by the row effect; synonym for factor(row) and vareffect(*numlist*)
* varcolumn(*numlist*)	variance explained by the column effect; synonym for factor(column) and vareffect(*numlist*)
* varrowcolumn(*numlist*)	variance explained by the row–column interaction effect; synonym for factor(rowcol) and vareffect(*numlist*)
* varerror(*numlist*)	error variance; default is varerror(1)
showmatrices	display cell means and sample sizes as matrices
showmeans	display cell means
showcellsizes	display cell sizes
parallel	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[no]table[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
saving(*filename* [, replace])	save the table data to *filename*; use replace to overwrite existing *filename*
Graph	
graph[(*graphopts*)]	graph results; see [PSS-2] **power, graph**
Iteration	
init(#)	initial value for sample size or effect size; default is to use a bisection algorithm to bound the solution
iterate(#)	maximum number of iterations; default is iterate(500)
tolerance(#)	parameter tolerance; default is tolerance(1e-12)
ftolerance(#)	function tolerance; default is ftolerance(1e-12)
[no]log	suppress or display iteration log
[no]dots	suppress or display iterations as dots
notitle	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see [U] 11.1.8 numlist. Also see the parallel option.

collect is allowed; see [U] 11.1.10 Prefix commands.

notitle does not appear in the dialog box.

wgtspec	Description
$\#_{1,1}\ \ldots\ \#_{1,K}\ \backslash\ \ldots\ \backslash\ \#_{J,1}\ \ldots\ \#_{J,K}$	$J \times K$ cell weights; weights must be positive and must be integers unless option nfractional is specified
matname	$J \times K$ matrix containing cell weights

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
alpha	significance level	α
power	power	$1-\beta$
beta	type-II-error probability	β
N	total number of subjects	N
N_per_cell	number of subjects per cell	N/N_{rc}
N_avg	average number of subjects per cell	N_{avg}
N$\#_1$_$\#_2$	number of subjects in cell ($\#_1$, $\#_2$)	$N_{\#_1,\#_2}$
delta	effect size	δ
N_rc	number of cells	N_{rc}
N_r	number of rows	N_r
N_c	number of columns	N_c
m$\#_1$_$\#_2$	cell mean ($\#_1$, $\#_2$)	$\mu_{\#_1,\#_2}$
Var_r	variance explained by the row effect	σ^2_r
Var_c	variance explained by the column effect	σ^2_c
Var_rc	variance explained by the row–column interaction	σ^2_{rc}
Var_e	error variance	σ^2_e
cwgt$\#_1$_$\#_2$	cell weight ($\#_1$, $\#_2$)	$w_{\#_1,\#_2}$
target	target parameter; synonym for target effect variance	
_all	display all supported columns	

Column beta is shown in the default table in place of column power if specified.

Column N_per_cell is available and is shown in the default table only for balanced designs.

Column N_avg is shown in the default table only for unbalanced designs.

Columns N $\#_1$_$\#_2$, N_rc, m $\#_1$_$\#_2$, and cwgt $\#_1$_$\#_2$ are not shown in the default table.

Options

Main

`alpha()`, `power()`, `beta()`, `n()`, `nfractional`; see [PSS-2] **power**.

`npercell(`*numlist*`)` specifies the cell size. Only positive integers are allowed. This option implies a balanced design. `npercell()` cannot be specified with `n()` or `cellweights()`.

`cellweights(`*wgtspec*`)` specifies $J \times K$ cell weights for an unbalanced design. The weights must be positive and must also be integers unless the `nfractional` option is specified. `cellweights()` cannot be specified with `npercell()`.

`nrows(`#`)` specifies the number of rows or the number of levels of the row factor in a two-way ANOVA. At least two rows must be specified. This option is required if *meanspec* is not specified. This option is also required for effect-size determination unless `cellweights()` is specified.

`ncols(`#`)` specifies the number of columns or the number of levels of the column factor in a two-way ANOVA. At least two columns must be specified. This option is required if *meanspec* is not specified. This option is also required for effect-size determination unless `cellweights()` is specified.

`factor(row` | `column` | `rowcol)` specifies the effect of interest for which power and sample-size analysis is to be performed. In a two-way ANOVA, the tested effects include the main effects of a row factor (row effect), the main effects of a column factor (column effect), or the interaction effects between the row and column factors (row–column effect). The default is `factor(row)`.

`vareffect(`*numlist*`)` specifies the variance explained by the tested effect specified in `factor()`. For example, if `factor(row)` is specified, `vareffect()` specifies the variance explained by the row factor. This option is required if the `factor()` option is specified and cell means are not specified. This option is not allowed with the effect-size determination. Only one of `vareffect()`, `varrow()`, `varcolumn()`, or `varrowcolumn()` may be specified.

`varrow(`*numlist*`)` specifies the variance explained by the row factor. This option is equivalent to specifying `factor(row)` and `vareffect(`*numlist*`)` and thus cannot be combined with `factor()`. This option is not allowed with the effect-size determination. Only one of `vareffect()`, `varrow()`, `varcolumn()`, or `varrowcolumn()` may be specified.

`varcolumn(`*numlist*`)` specifies the variance explained by the column factor. This option is equivalent to specifying `factor(column)` and `vareffect(`*numlist*`)` and thus cannot be combined with `factor()`. This option is not allowed with the effect-size determination. Only one of `vareffect()`, `varrow()`, `varcolumn()`, or `varrowcolumn()` may be specified.

`varrowcolumn(`*numlist*`)` specifies the variance explained by the interaction between row and column factors. This option is equivalent to specifying `factor(rowcol)` and `vareffect(`*numlist*`)` and thus cannot be combined with `factor()`. This option is not allowed with the effect-size determination. Only one of `vareffect()`, `varrow()`, `varcolumn()`, or `varrowcolumn()` may be specified.

`varerror(`*numlist*`)` specifies the error (within-cell) variance. The default is `varerror(1)`.

`showmatrices` specifies that the matrices of cell means and cell sizes be displayed, when applicable. The cell means will be displayed only if specified. The cell sizes will be displayed only for an unbalanced design.

`showmeans` specifies that the cell means be reported. For a text or graphical output, this option is equivalent to `showmatrices` except only the cell-mean matrix will be reported. For a tabular output, the columns containing cell means will be included in the default table.

showcellsizes specifies that the cell sizes be reported. For a text or graphical output, this option is equivalent to showmatrices except only the cell-sizes matrix will be reported. For a tabular output, the columns containing cell sizes will be included in the default table.

parallel; see [PSS-2] **power**.

Table

table, table(), notable; see [PSS-2] **power, table**.

saving(); see [PSS-2] **power**.

Graph

graph, graph(); see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

init(#) specifies the initial value of the sample size for the sample-size determination or the initial value of the effect size δ for the effect-size determination. The default uses a bisection algorithm to bracket the solution.

iterate(), tolerance(), ftolerance(), log, nolog, dots, nodots; see [PSS-2] **power**.

The following option is available with power twoway but is not shown in the dialog box:

notitle; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the power twoway command and the methodology for power and sample-size analysis for two-way ANOVA. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis and [PSS-2] **power** for a general introduction to the power command using hypothesis tests.

Introduction

ANOVA has been one of the most widely used statistical tools in many scientific applications. Two-way ANOVA models allow analysts to study the effects of two factors simultaneously. The term "two way" refers to two factors each containing an arbitrary number of groups or levels.

For example, consider a type of drug with three levels of dosage in reducing blood pressure for males and females. In this case, three interesting hypotheses arise: an investigator may wish to test whether the average change in blood pressure is the same for both genders, whether the average change in blood pressure is the same across all levels of dosage regardless of gender, or whether there is any interaction between dosage levels and gender.

This entry describes power and sample-size analysis for the inference about main and interaction effects of two factors based on hypothesis testing. Let μ_{jk} be the cell mean of the jth row and the kth column in a two-way cell-means ANOVA model, $\mu_{j\cdot}$ be the marginal mean of the jth row, $\mu_{\cdot k}$ be the marginal mean of the kth column, and $\mu_{\cdot\cdot}$ be the grand mean. The jth-row-by-kth-column interaction effect is then $(ab)_{jk} = \mu_{jk} - \mu_{j\cdot} - \mu_{\cdot k} + \mu_{\cdot\cdot}$. We consider the null hypotheses 1) H_0: $\mu_{1\cdot} = \cdots = \mu_{J\cdot}$, for testing the main row effect; 2) H_0: $\mu_{\cdot 1} = \cdots = \mu_{\cdot K}$, for testing the main column effect; and 3) H_0: all $(ab)_{jk} = 0$, for testing the row-by-column interaction effect.

The test statistic for each of the three hypotheses is based on the ratio of the variance explained by the tested effect to the error variance. Under the null hypothesis, the test statistics used for items 1, 2, and 3 above have an F distribution. We will refer to the corresponding tests as F tests for row, column, and row-by-column effects. For power analysis, we consider the distribution of the test statistic under the alternative hypothesis. This distribution is a noncentral F distribution for all the considered tests. Power is a function of the noncentrality parameter, and the noncentrality parameter is a function of the ratio of the standard deviation of the tested effect to the standard deviation of the errors. As such, the effect size for each of the F tests is defined as the square root of the ratio of the variance explained by the tested effect to the error variance.

`power twoway` performs power and sample-size computation for a two-way fixed-effects ANOVA model based on an F test of the effect of interest.

Using power twoway

`power twoway` computes sample size, power, or effect size and target variance of the effect for a two-way fixed-effects ANOVA. All computations are performed assuming a significance level of 0.05. You may change the significance level by specifying the `alpha()` option.

By default, the computations are performed for an F test of the main row effects; `factor(row)` is assumed. You can instead request a test of the main column effects by specifying `factor(column)` or a test of the row-by-column interaction effects by specifying `factor(rowcol)`. The error variance for all tests is assumed to be 1 but may be changed by specifying the `vararror()` option.

To compute the total sample size, you must specify the alternative *meanspec* and, optionally, the power of the test in the `power()` option. The default power is set to 0.8.

To compute power, you must specify the total sample size in the `n()` option and the alternative *meanspec*.

Instead of the alternative cell means, you can specify the number of rows in the `nrows()` option, the number of columns in the `ncols()` option, and the variance explained by the tested effect in the `vareffect()` option when computing sample size or power. See *Alternative ways of specifying effect*.

To compute effect size, the square root of the ratio of the variance explained by the tested factor to the error variance, and the target variance explained by the tested factor, you must specify the total sample size in the `n()` option, the power in the `power()` option, and the number of rows and columns in `nrows()` and `ncols()`, respectively.

By default, all computations assume a balanced- or an equal-allocation design. You can use the `cellweights()` option to specify an unbalanced design for power, sample-size, or effect-size computations. For power and effect-size computations of a balanced design, you can also specify the `npercell()` option to specify a cell size instead of a total sample size in `n()`.

In a two-way ANOVA, sample size and effect size depend on the noncentrality parameter of the F distribution, and their estimation requires iteration. The default initial values are obtained from a bisection search that brackets the solution. If you desire, you may change this by specifying your own value in the `init()` option. See [PSS-2] **power** for the descriptions of other options that control the iteration procedure.

In a two-way ANOVA, all computations depend on the noncentrality parameter of the F distribution. The sample-size and effect-size computation requires a nonlinear search algorithm where the default initial value is obtained using a bisection search algorithm that brackets the solution.

`power twoway` provides several ways of specifying the study parameters that we discuss next.

Alternative ways of specifying effect

To compute power or sample size, you must specify the magnitude of the effect desired to be detected by the test. With `power twoway`, you can do this in several ways. For example, consider a two-way model with $J = 2 \geq 2$ row-factor levels and $K = 3 \geq 2$ column-factor levels. You can specify either the individual alternative *meanspec*,

`power twoway` $m_{1,1}\ m_{1,2}\ m_{1,3} \setminus m_{2,1}\ m_{2,2}\ m_{2,3}$ [`, factor() ...`]

or the variance of the tested effect and the number of rows J and columns K:

`power twoway, factor() vareffect(#) nrows(2) ncols(3)` [`...`]

You can also replace `vareffect()` and `factor()` in the above with the `varrow()`, `varcolumn()`, or `varrowcolumn()` option. And you can specify multiple values of the variances in these options.

As an alternative to directly specifying alternative cell means following the command name, you can define a Stata matrix containing these means and use it with `power twoway`. For example,

`matrix define` *meanmat* `=` ($m_{1,1}$`,` $m_{1,2}$`,` $m_{1,3} \setminus m_{2,1}$`,` $m_{2,2}$`,` $m_{2,3}$)

The matrix must have at least two rows and two columns.

`power twoway` *meanmat*`, ...`

In the following sections, we describe the use of `power twoway` accompanied by examples for computing sample size, power, and effect size.

Computing sample size

To compute sample size, you must specify the alternative cell means or the variance of the tested effect and, optionally, the power of the test in the `power()` option. A power of 0.8 is assumed if `power()` is not specified.

▷ Example 1: Sample size for a two-way ANOVA—row effect

van Belle et al. (2004, 376) provide an example of an experimental study that investigates the effect of an automobile emission pollutant, nitrogen dioxide (NO_2). The experiment considers the effect of NO_2 exposure on protein leakage in the lungs of mice. In the experimental group, mice were exposed to 0.5 part per million (ppm) NO_2 for 10, 12, and 14 days. Measurements on the response variable, serum fluorescence, were taken on mice in the experimental (exposed) and control (unexposed) groups.

The analysis of these data used a two-way ANOVA model with the exposure status as a row factor and the number of days of exposure to NO_2 as a column factor. The row factor has two levels, exposed or unexposed, and the column factor has three levels: 10, 12, and 14 days.

Suppose that investigators are planning to conduct another similar study. They would like to know how many subjects, mice, they need for the experiment. We will use the estimates of parameters from the above study to answer this question.

The estimated cell means from this study over the number of days are 134.4, 143, and 91.3 in the control group and 106.4, 173.2, and 145.5 in the experimental group. From the ANOVA table on page 379 (van Belle et al. 2004), the estimated residual variance is 1417.35. For convenience, we round these numbers down to the nearest integers in our computations.

We begin by computing sample size for testing the main treatment (exposure) effects using `power twoway`'s defaults for other aspects of the study: a balanced design, a 5% significance level, and 80% power. (Option `factor(row)` is assumed by default.)

```
. power twoway 134 143 91 \ 106 173 145, varerror(1417)

Performing iteration ...

Estimated sample size for two-way ANOVA
F test for row effect
H0: delta = 0  versus  Ha: delta != 0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.2479
          N_r =         2
          N_c =         3
        means =   <matrix>
        Var_r =   87.1111
        Var_e = 1417.0000

Estimated sample sizes:

            N =       132
   N per cell =        22
```

Assuming a balanced design, we need a total of 132 mice with 22 mice per cell to detect the effect of exposure to NO_2 on the protein leakage of mice.

Like all other `power` methods, `power twoway` reports study parameters first and the estimated parameters next. The reported study parameters include the specified and implied parameters such as significance level, power, number of rows, number of columns, and so on. `power twoway` does not display the specified cell means by default but indicates in the output that the means are specified. You can specify the `showmeans` option to display cell means as a matrix.

In addition to the specified and implied study parameters, `power twoway` reports the value of the effect size, `delta` $= \sqrt{87.1111/1417} = 0.2479$, computed as a square root of the ratio between the variance of the row effect `Var_r` and the error variance `Var_e`. As for the one-way ANOVA models, the effect size δ provides a unitless measure of the magnitude of an effect with a lower bound of zero, meaning no effect. It corresponds to Cohen's effect-size measure f (Cohen 1988). Cohen's convention is that $f = 0.1$ means small effect size, $f = 0.25$ means medium effect size, and $f = 0.4$ means large effect size. According to this convention, the effect size considered in our example is medium.

◁

Example 2: Sample size for a two-way ANOVA—column effect

Continuing with example 1, we can compute the required sample size for the main column effects by specifying the `factor(column)` option:

```
. power twoway 134 143 91 \ 106 173 145, varerror(1417) factor(column)
Performing iteration ...
Estimated sample size for two-way ANOVA
F test for column effect
H0: delta = 0  versus  Ha: delta != 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.4889
          N_r =         2
          N_c =         3
        means =   <matrix>
        Var_c =  338.6667
        Var_e = 1417.0000
Estimated sample sizes:
            N =        48
   N per cell =         8
```

Assuming a balanced design, we need a total of 48 mice with 8 mice per cell to detect the effect of the length of exposure to NO_2 on the protein leakage of mice.

Similarly to the row effect, the effect size for the column effect, `delta` $= \sqrt{338.6667/1417} = 0.4889$, is computed as a square root of the ratio between the variance of the column effect `Var_c` and the error variance `Var_e`. The interpretation remains the same but with respect to the main column effects. According to Cohen's scale, the effect size corresponding to the test of the main column effects is large, so we need fewer subjects to detect the column effect than we need to detect the previous row effect.

◁

Example 3: Sample size for a two-way ANOVA—row-by-column effect

Continuing with example 2, we can also compute the required sample size for the row-by-column effects interaction by specifying the `factor(rowcol)` option:

```
. power twoway 134 143 91 \ 106 173 145, varerror(1417) factor(rowcol)
Performing iteration ...
Estimated sample size for two-way ANOVA
F test for row-by-column effect
H0: delta = 0  versus  Ha: delta != 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.4572
          N_r =         2
          N_c =         3
        means =   <matrix>
       Var_rc =  296.2222
        Var_e = 1417.0000
Estimated sample sizes:
            N =        54
   N per cell =         9
```

For a balanced design, we need a total of 54 mice with 9 mice per cell to detect the joint effects of exposure and the length of exposure to NO_2 on the protein leakage of mice.

Similarly to the row-by-column effects, the effect size for the row-by-column effect, `delta` $= \sqrt{296.2222/1417} = 0.4572$, is computed as a square root of the ratio between the variance of the row-by-column effect `Var_rc` and the error variance `Var_e`. The interpretation is again the same but with respect to the interaction of row-by-column effects. According to Cohen's scale, the effect size corresponding to the test of the row-by-column effects is also large, so we need fewer subjects to detect this effect than we need to detect the row effect. The effect size is similar to the column effect size, so the required numbers of subjects are comparable for the two tests. As a final sample size, we would choose the largest of the three sizes to ensure that we have enough subjects to detect any of the considered effects.

◁

Example 4: Alternative ways of specifying effect

Instead of specifying the alternative cell means as in previous examples, we can specify the variance explained by the corresponding tested effect and the numbers of rows and columns.

For instance, from example 2, the variance explained by the column effect was computed to be 338.6667. We specify this value in `vareffect()` as well as the number of rows in `nrows()` and the number of columns in `ncols()`:

```
. power twoway, varerror(1417) factor(column) vareffect(338.6667)
> nrows(2) ncols(3)

Performing iteration ...

Estimated sample size for two-way ANOVA
F test for column effect
H0: delta = 0  versus  Ha: delta != 0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.4889
          N_r =         2
          N_c =         3
        Var_c =  338.6667
        Var_e = 1417.0000

Estimated sample sizes:

            N =        48
   N per cell =         8
```

We obtain the exact same results as in example 2.

A shorthand for the specification of `factor(column)` and `vareffect()` is the `varcolumn()` option. You can verify that the specification

```
. power twoway, varerror(1417) varcolumn(338.6667) nrows(2) ncols(3)
  (output omitted)
```

produces results identical to the results above.

You can also use similar alternative specifications for the tests of row and row-by-column effects with intuitive modifications to the syntax.

`power twoway` also provides another alternative specification of the cell means. Instead of specifying alternative cell means directly following the command line, as in example 2, we can define a matrix, say, `means`, containing these means and use it with `power twoway`:

```
. matrix define means = (134, 143, 91 \ 106, 173, 145)
. power twoway means, varerror(1417) factor(column)
  (output omitted)
```

You can again verify that the results are identical to the previous results.

◁

▷ Example 5: Unbalanced design

Continuing with example 1, let's compute the required sample size for an unbalanced design. For instance, consider a design in which the control group (the first row) contains twice as many subjects as the experimental group (the second row) for each level of the other factor. We use the `cellweights()` option to specify weights for each cell.

```
. power twoway 134 143 91 \ 106 173 145, varerror(1417) cellweights(2 2 2 \ 1 1 1)
> showcellsizes

Performing iteration ...

Estimated sample size for two-way ANOVA
F test for row effect
H0: delta = 0  versus  Ha: delta != 0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.2338
          N_r =         2
          N_c =         3
        means =   <matrix>
        Var_r =   77.4321
        Var_e = 1417.0000

Estimated sample sizes:

            N =       153
    Average N =   25.5000

  Cell sample sizes

                |  columns
                |        1          2          3
----------------+-----------------------------------
  rows          |
              1 |       34         34         34
              2 |       17         17         17
```

The required total sample size for this unbalanced design is 153 with the average number of subjects in a cell of 25.5. We specified the `showcellsizes` option to display the number of subjects for each cell along with the total and average sample sizes that are displayed by default for an unbalanced design.

You can alternatively specify cell weights as a matrix.

Computing power

To compute power, you must specify the total sample size in the `n()` option and the desired effect size, expressed using alternative cell means, for example. See *Alternative ways of specifying effect*.

▷ Example 6: Power for a two-way ANOVA

Continuing with example 1, suppose that we anticipate a sample of 90 mice. To compute the corresponding power, we specify the sample size of 90 in `n()`.

```
. power twoway 134 143 91 \ 106 173 145, varerror(1417) n(90)
Estimated power for two-way ANOVA
F test for row effect
H0: delta = 0  versus  Ha: delta != 0
Study parameters:
        alpha =    0.0500
            N =        90
   N per cell =        15
        delta =    0.2479
          N_r =         2
          N_c =         3
        means =   <matrix>
        Var_r =   87.1111
        Var_e = 1417.0000
Estimated power:
        power =    0.6426
```

For this smaller sample size, the power for detecting the effect size of 0.25 is only 64%.

◁

▷ Example 7: Multiple values of study parameters

Continuing with example 6, we may want to check powers for several sample sizes. We simply list multiple sample-size values in `n()`:

```
. power twoway 134 143 91 \ 106 173 145, varerror(1417) n(90 114 126)
Estimated power for two-way ANOVA
F test for row effect
H0: delta = 0  versus  Ha: delta != 0
   means = <matrix>
```

alpha	power	N	N_per_cell	delta	N_r	N_c	Var_r	Var_e
.05	.6426	90	15	.2479	2	3	87.11	1417
.05	.7466	114	19	.2479	2	3	87.11	1417
.05	.7884	126	21	.2479	2	3	87.11	1417

The larger the sample size, the larger the power.

We can even compute results for multiple sample sizes and, for example, multiple values of error variances.

```
. power twoway 134 143 91 \ 106 173 145, varerror(1000 1800) n(90 114 126)
> table(, sep(3))
Estimated power for two-way ANOVA
F test for row effect
H0: delta = 0  versus  Ha: delta != 0
   means = <matrix>
```

alpha	power	N	N_per_cell	delta	N_r	N_c	Var_r	Var_e
.05	.7904	90	15	.2951	2	3	87.11	1000
.05	.8776	114	19	.2951	2	3	87.11	1000
.05	.9076	126	21	.2951	2	3	87.11	1000
.05	.5411	90	15	.22	2	3	87.11	1800
.05	.6436	114	19	.22	2	3	87.11	1800
.05	.6878	126	21	.22	2	3	87.11	1800

We specified `table()`'s suboption `separator()`, abbreviated to `sep()`, to improve readability of the table.

We can also compute results for combinations of multiple values of other study parameters.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

Computing effect size and target variance explained by the tested effect

Sometimes, we may be interested in determining the smallest effect that yields a statistically significant result for prespecified sample size and power. In this case, power, sample size, and the numbers of rows and columns must be specified.

The effect size is defined as a square root of the ratio of the variance explained by the tested effect to the error variance. The effect size and the target variance explained by the tested effect are computed.

▷ Example 8: Effect size for a two-way ANOVA—row effect

Continuing with example 6, we now want to compute the effect size that can be detected for a sample of 90 subjects and a power of 80%. We specify both parameters in the respective options. For the effect-size determination, we must also specify the number of rows in `nrows()` and the number of columns in `ncols()`:

```
. power twoway, varerror(1417) n(90) power(0.8) nrows(2) ncols(3)
Performing iteration ...
Estimated row variance for two-way ANOVA
F test for row effect
H0: delta = 0  versus  Ha: delta != 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
            N =        90
   N per cell =        15
          N_r =         2
          N_c =         3
        Var_e = 1417.0000
Estimated effect size and row variance:
        delta =    0.2987
        Var_r =  126.4634
```

With a smaller sample size, given the same power, we can only detect a larger effect size of 0.2987, compared with the effect size of 0.2479 from example 1. The corresponding estimate of the variance explained by the row effect is 126.46, given the error variance of 1417.

◁

Testing hypotheses about means from multiple populations

▷ Example 9: Two-way ANOVA

After the initial power and sample-size planning, we can use Stata's `anova` command to perform inference for two-way ANOVA based on the collected sample. We show a quick example of how to do this here; see [R] **anova** for more examples and details.

We use data on systolic blood pressure. Consider a sample of 58 patients, each suffering from 1 of 3 different diseases, who were randomly assigned to 1 of 4 different drug treatments and whose change in systolic blood pressure was recorded. To test for the effects of the drug and the disease and their interaction, we type the following:

```
. use https://www.stata-press.com/data/r19/systolic
(Systolic blood pressure data)
. anova systolic drug disease drug#disease
                          Number of obs =         58    R-squared     =  0.4560
                          Root MSE      =    10.5096    Adj R-squared =  0.3259
                 Source | Partial SS         df         MS         F    Prob>F
------------------------+----------------------------------------------------
                  Model |  4259.3385         11   387.21259      3.51  0.0013
                        |
                   drug |  2997.4719          3   999.15729      9.05  0.0001
                disease |  415.87305          2   207.93652      1.88  0.1637
           drug#disease |  707.26626          6   117.87771      1.07  0.3958
                        |
               Residual |  5080.8167         46   110.45254
------------------------+----------------------------------------------------
                  Total |  9340.1552         57   163.86237
```

We find that only the main effect of the drug is significant.

Suppose that we would like to conduct a similar study. We use the estimates from the study above to compute the required sample size for our new study. We are particularly interested in testing the interaction between the drug and disease, so we would like to compute the sample size for this test.

First, we estimate the cell means of systolic blood pressure for different treatment and disease levels:

```
. table drug disease, statistic(mean systolic) nformat(%9.0f) nototals
```

	Patient's disease		
	1	2	3
Drug used			
1	29	28	20
2	28	34	18
3	16	4	8
4	14	13	14

From the `twoway` output, the estimate of the error variance is roughly 110. Second, we specify the means and the error variance with `power twoway` and compute the required sample size for a balanced design assuming 5% significance level and 80% power for the test of interaction effects.

```
. power twoway 29 28 20 \ 28 34 18 \ 16 4 8 \ 14 13 14, vareror(110) f(rowcol)
Performing iteration ...
Estimated sample size for two-way ANOVA
F test for row-by-column effect
H0: delta = 0  versus  Ha: delta != 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.3465
          N_r =         4
          N_c =         3
        means =   <matrix>
       Var_rc =   13.2083
        Var_e =  110.0000
Estimated sample sizes:
            N =       132
   N per cell =        11
```

We need a total of 132 subjects with 11 subjects per cell to detect the drug-by-disease effect size of 0.3465 for this design.

To determine the final sample size, you may want to repeat the same computations for the tests of the main effects of drug and the main effects of disease and select the sample size based on the three tests.

◁

Stored results

power twoway stores the following in r():

Scalars

r(alpha)	significance level
r(power)	power
r(beta)	probability of a type II error
r(delta)	effect size
r(N)	total sample size
r(N_a)	actual sample size
r(N_avg)	average sample size
r(N#$_1$_#$_2$)	number of subjects in cell (#$_1$, #$_2$)
r(N_per_cell)	number of subjects per cell
r(N_rc)	number of cells
r(nfractional)	1 if nfractional is specified, 0 otherwise
r(balanced)	1 for a balanced design, 0 otherwise
r(cwgt#$_1$_#$_2$)	cell weight (#$_1$, #$_2$)
r(N_r)	number of rows
r(N_c)	number of columns
r(m#$_1$_#$_2$)	cell mean (#$_1$, #$_2$)
r(Var_r)	row variance
r(Var_c)	column variance
r(Var_rc)	row-by-column variance
r(Var_e)	error variance
r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise
r(init)	initial value for sample size or effect size
r(maxiter)	maximum number of iterations
r(iter)	number of iterations performed
r(tolerance)	requested parameter tolerance
r(deltax)	final parameter tolerance achieved
r(ftolerance)	requested distance of the objective function from zero
r(function)	final distance of the objective function from zero
r(converged)	1 if iteration algorithm converged, 0 otherwise

Macros

r(type)	test
r(method)	twoway
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results
r(Nij)	cell-sizes matrix
r(means)	cell-means matrix
r(cwgt)	cell-weights matrix

Methods and formulas

Consider factor A with J levels and factor B with K levels. Let μ_{jk} be the mean of cell (j, k) in a table formed by the levels of factors A and B. For example, let $J = 3$ and $K = 3$; then the following cell-means table summarizes the experiment.

	Factor B			
Factor A	$k=1$	$k=2$	$k=3$	Total
$j=1$	μ_{11}	μ_{12}	μ_{13}	$\mu_{1\cdot}$
$j=2$	μ_{21}	μ_{22}	μ_{23}	$\mu_{2\cdot}$
$j=3$	μ_{31}	μ_{32}	μ_{33}	$\mu_{3\cdot}$
Total	$\mu_{\cdot 1}$	$\mu_{\cdot 2}$	$\mu_{\cdot 3}$	$\mu_{\cdot\cdot}$

Methods and formulas are presented under the following headings:

Main effects

Main effects measure the deviation of the factor-level means from the overall or grand mean. The larger the main effect, the more likely you can detect the effect. From the above table, the main effect of factor A at the jth level is $a_j = \mu_{j\cdot} - \mu_{\cdot\cdot}$. Similarly, the main effect of factor B at the kth level is $b_k = \mu_{\cdot k} - \mu_{\cdot\cdot}$. The overall mean can be expressed as

$$\mu_{\cdot\cdot} = \frac{\sum_{j=1}^{J}\sum_{k=1}^{K}\mu_{jk}}{JK} = \frac{\sum_{j=1}^{J}\mu_{j\cdot}}{J} = \frac{\sum_{k=1}^{K}\mu_{\cdot k}}{K}$$

This implies that

$$\sum_{j=1}^{J} a_j = 0 \qquad \text{and} \qquad \sum_{k=1}^{K} b_k = 0$$

Interaction effects

Unlike main effects that measure the effect of individual factors on the dependent variable, interaction effects measure the effect of the two factors jointly on the dependent variable. For example, the interaction effect of factor A at the jth level and factor B at the kth level is $(ab)_{jk} = \mu_{jk} - \mu_{j\cdot} - \mu_{\cdot k} + \mu_{\cdot\cdot}$.

The sum of interaction effects is zero:

$$\sum_{j=1}^{J}(ab)_{jk} = 0 \text{ at each level of } k = 1, \ldots, K$$

$$\sum_{k=1}^{K}(ab)_{jk} = 0 \text{ at each level of } j = 1, \ldots, J$$

This implies

$$\sum_{j=1}^{J}\sum_{k=1}^{K}(ab)_{jk} = 0$$

Hypothesis testing

Let n denote the total sample size and y_{ijk} denote the response of ith individual at the jth level of factor A and kth level of factor B for $i = 1, \ldots, n_{jk}$ such that $n = \sum_{j,k} n_{jk}$. ANOVA models assume that responses y_{ijk} within each cell are independent and identically distributed random normal with constant variance σ_e^2. Using the definitions of a_j, b_k, and $(ab)_{jk}$ from the previous sections, our linear model is expressed as

$$\begin{aligned} y_{ijk} &= \mu_{..} + a_j + b_k + (ab)_{jk} + e_{ijk} \\ &= \mu_{jk} + e_{ijk} \end{aligned}$$

where $\mu_{..}$ is the overall mean and e_{ijk}'s are the independent error terms that have the standard normal distribution. The first equation corresponds to the formulation of an ANOVA model using effects, and the second formulation corresponds to the cell-means formulation.

The variance explained by the row effects is $\sigma_a^2 = \sum_j a_j^2/J$, by the column effects is $\sigma_b^2 = \sum_k b_k^2/K$, and by the row-by-column effects is $\sigma_{(ab)}^2 = \sum_{j,k}(ab)_{jk}^2/JK$.

The following sets of hypotheses are of interest in a two-way ANOVA:

$$H_0\text{: all } a_j = 0 \quad \text{versus} \quad H_a\text{: at least one } a_j \neq 0 \tag{1}$$

$$H_0\text{: all } b_k = 0 \quad \text{versus} \quad H_a\text{: at least one } b_k \neq 0 \tag{2}$$

$$H_0\text{: all } (ab)_{jk} = 0 \quad \text{versus} \quad H_a\text{: at least one } (ab)_{jk} \neq 0 \tag{3}$$

Hypotheses (1) and (2) test the main effects of factors A and B, respectively, and hypothesis (3) tests the interaction effects between A and B.

To test the above hypotheses, we can use the general linear model framework discussed in *Methods and formulas* of [PSS-2] **power oneway**. We recapitulate it here with application to the two-way model.

A general test statistic for testing hypotheses like (1), (2), and (3) is given by

$$F_C = \frac{SS_C}{\nu\hat{\sigma}_e^2} \tag{4}$$

where $SS_C = (\mathbf{C}\hat{\mathbf{b}})'\{\mathbf{C}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{C}'\}^{-1}(\mathbf{C}\hat{\mathbf{b}})$. The $n \times JK$ matrix $\mathbf{X}$ specifies the coding for the two-way design. Matrix $\mathbf{C}$ is $\nu \times JK$ and contains the contrasts for the means that are used to test each of the three hypotheses. For the two-way design, v is $J-1$, $K-1$, or $(J-1)(K-1)$ for hypotheses (1), (2), and (3), respectively.

Let α be the significance level and $F_{\nu,n-JK,1-\alpha}$ be the $(1-\alpha)$th quantile of an F distribution with ν numerator and $n-JK$ denominator degrees of freedom. We reject the null hypothesis if we observe a statistic $F_C > F_{\nu,n-JK,1-\alpha}$.

Under the alternative hypothesis, the test statistic (4) is distributed as a noncentral F distribution with ν numerator and $n-JK$ denominator degrees of freedom and a noncentrality parameter λ given by

$$\begin{aligned} \lambda &= n(\mathbf{C}\mathbf{b})'\{\mathbf{C}(\ddot{\mathbf{X}}'\mathbf{W}\ddot{\mathbf{X}})^{-1}\mathbf{C}'\}^{-1}(\mathbf{C}\mathbf{b})/\sigma_e^2 \\ &= n\delta^2 \end{aligned}$$

where the matrix $\ddot{\mathbf{X}}$ contains the unique rows of $\mathbf{X}$ such that $\boldsymbol{\mu} = \ddot{\mathbf{X}}\mathbf{b}$, $\mathbf{W} = \text{diag}(w_1, \ldots, w_{JK})$, and δ is the effect size. For a two-way design, the dimension of $\ddot{\mathbf{X}}$ is $JK \times JK$, and the weights are $w_i = n_{jk}/n$, $i = (k-1)J + j$. The cell-means parameterization simplifies $\ddot{\mathbf{X}}$ to the identity matrix, $\mathbf{I}_{JK}$.

The rows of $\ddot{\mathbf{X}}$ and weights w_i are associated with the column-major, `vec()`, order of the two-way table with factor A indexed on the rows and factor B indexed on the columns. (See the 3×3 table in the beginning of this section, and scan each column $k = 1, 2, 3$.) The weight w_i can be reexpressed as a cell weight $\widetilde{w}_i$, which is independent of the sample size n; see *Methods and formulas* of [PSS-2] **power oneway** for details.

When the `cellweights()` option is specified, a constant cell-size multiplier n_c is computed and rounded to an integer unless the `nfractional` option is specified. The cell sizes are then computed as $\widetilde{w}_j n_c$. The actual sample size, `N_a`, is the sum of the cell sizes.

References

Cohen, J. 1988. *Statistical Power Analysis for the Behavioral Sciences*. 2nd ed. Hillsdale, NJ: Erlbaum.

van Belle, G., L. D. Fisher, P. J. Heagerty, and T. S. Lumley. 2004. *Biostatistics: A Methodology for the Health Sciences*. 2nd ed. New York: Wiley.

Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power oneway** — Power analysis for one-way analysis of variance

[PSS-2] **power repeated** — Power analysis for repeated-measures analysis of variance

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[R] **anova** — Analysis of variance and covariance

power repeated — Power analysis for repeated-measures analysis of variance

Description

`power repeated` computes sample size, power, or effect size for one-way or two-way repeated-measures analysis of variance (ANOVA). By default, it computes sample size for given power and effect size. Alternatively, it can compute power for given sample size and effect size or compute effect size for given sample size, power, and number of groups. Also see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Quick start

Sample size for a repeated measures design with one 3-level within-subject factor, a correlation of 0.3 between measurements, and an error variance of 42 with default power of 0.8 and significance level $\alpha = 0.05$

```
power repeated 25 27 22, varerror(42) corr(.3)
```

Same as above, specified as 3 measurements on 1 group with within-subject variance of 4.22

```
power repeated, varerror(42) corr(.3) nrepeated(3) ngroups(1) ///
   varwithin(4.22)
```

Same as above, specified as cell means in matrix `cm`

```
matrix cm = (25,27,22)
power repeated cm, corr(.3) varerror(42)
```

Same as above, and show the mean and covariance matrices in the output

```
power repeated cm, corr(.3) varerror(42) showmatrices
```

Sample size for the between effect in a design with a 3-level within-subject factor and a 2-level between-subject factor

```
power repeated 18 14 12\14 13 10, covmatrix(24 9 9\9 24 9\9 9 24)
```

Same as above, specified as cell means in matrix `cm2` and covariances in matrix `cov`

```
matrix cm2 = (18,14,12\14,13,10)
matrix cov = (24,9,9\9,24,9\9,9,24)
power repeated cm2, covmatrix(cov)
```

Same as above, but for the within effect

```
power repeated cm2, covmatrix(cov) factor(within)
```

Same as above, but for the between–within effect

```
power repeated cm2, covmatrix(cov) factor(bwithin)
```

Power for a design with one within-subject factor, a sample size of 25, and $\alpha = 0.01$

```
power repeated 25 27 22, varerror(42) corr(.3) n(25) alpha(.01)
```

Same as above, but for sample sizes of 20, 24, 28, and 32

```
power repeated 25 27 22, varerror(42) corr(.3) n(20(4)32)
```

Same as above, but show results in a graph of sample size versus power

```
power repeated 25 27 22, varerror(42) corr(.3) n(20(4)32) graph
```

Power for the between effect of a design with a 3-level within-subject factor and a 2-level between-subject factor with a sample size of 160

```
power repeated cm2, covmatrix(cov) n(160)
```

Same as above, but specify sample sizes of 100 and 140 for groups 1 and 2, respectively

```
power repeated cm2, covmatrix(cov) n1(100) n2(140)
```

Effect size for a one-group repeated-measures design

```
power repeated, varerror(42) corr(.3) n(24) ngroups(1) ///
   nrepeated(3) power(.8)
```

Effect size for the within-subject effect of a design with a 3-level within-subject factor, a 2-level between-subject factor, and a sample size of 160

```
matrix cov = (24,9,9\9,24,9\9,9,24)
power repeated, covmatrix(cov) n(160) power(.8) ngroups(2) ///
   factor(within)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
power repeated meanspec, corrspec [power(numlist) options]
```

Compute power

```
power repeated meanspec, n(numlist) corrspec [options]
```

Compute effect size

```
power repeated, n(numlist) power(numlist) ngroups(#) corrspec [options]
```

where *meanspec* is either a matrix *matname* containing cell means or individual cell means in a matrix form:

$m_{1,1}\ m_{1,2}\ [\ \ldots\ m_{1,K}\ ]\ [\ \backslash\ \ldots [\ \backslash\ m_{J,1}\ m_{J,2}\ [\ \ldots\ m_{J,K}\ ]\ ]\ ]$

m_{jk}, where $j = 1, 2, \ldots, J$ and $k = 1, 2, \ldots, K$, is the alternative cell mean or the cell mean of the jth row (group) and kth column (repeated measure) under the alternative hypothesis.

matname is the name of a Stata matrix with J rows and K columns containing values of alternative cell means.

At least one group, $J = 1$, and two repeated measures, $K = 2$, must be specified.

where *corrspec* for computing power and sample size is {`corr(`*numlist*`)` | `covmatrix(`*matname*`)`}, and *corrspec* for computing effect size is {`nrepeated(#) corr(`*numlist*`)` | `covmatrix(`*matname*`)`}.

options	Description
Main	
* `alpha(`*numlist*`)`	significance level; default is `alpha(0.05)`
* `power(`*numlist*`)`	power; default is `power(0.8)`
* `beta(`*numlist*`)`	probability of type II error; default is `beta(0.2)`
* `n(`*numlist*`)`	total sample size; required to compute power or effect size
`nfractional`	allow fractional sample sizes
* `npergroup(`*numlist*`)`	number of subjects per group; implies balanced design
* `n#(`*numlist*`)`	number of subjects in group #
`grweights(`*wgtspec*`)`	group weights; default is one for each group, meaning equal group sizes
`ngroups(`#`)`	number of groups
`nrepeated(`#`)`	number of repeated measures
* `corr(`*numlist*`)`	correlation between repeated measures; one of `corr()` or `covmatrix()` is required
`covmatrix(`*matname*`)`	covariance between repeated measures; one of `corr()` or `covmatrix()` is required
`factor(between | within | bwithin)`	tested effect: between, within, or between–within; default is `factor(between)`
* `vareffect(`*numlist*`)`	variance explained by the tested effect specified in `factor()`
* `varbetween(`*numlist*`)`	variance explained by the between-subjects effect; synonym for `factor(between)` and `vareffect(`*numlist*`)`
* `varwithin(`*numlist*`)`	variance explained by the within-subject effect; synonym for `factor(within)` and `vareffect(`*numlist*`)`
* `varbwithin(`*numlist*`)`	variance explained by the between–within effect; synonym for `factor(bwithin)` and `vareffect(`*numlist*`)`
* `varerror(`*numlist*`)`	error variance; default is `varerror(1)` when `corr()` is specified; not allowed with `covmatrix()`
`showmatrices`	display cell-means matrix and covariance matrix
`showmeans`	display cell means
`parallel`	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
`[no]table[ (`*tablespec*`) ]`	suppress table or display results as a table; see [PSS-2] **power, table**
`saving(`*filename* `[ , replace ])`	save the table data to *filename*; use `replace` to overwrite existing *filename*
Graph	
`graph[ (`*graphopts*`) ]`	graph results; see [PSS-2] **power, graph**

Iteration

`init(#)`	initial value for sample size or effect size; default is to use a bisection algorithm to bound the solution
`iterate(#)`	maximum number of iterations; default is `iterate(500)`
`tolerance(#)`	parameter tolerance; default is `tolerance(1e-12)`
`ftolerance(#)`	function tolerance; default is `ftolerance(1e-12)`
`[no]log`	suppress or display iteration log
`[no]dots`	suppress or display iterations as dots
`notitle`	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the `parallel` option.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`notitle` does not appear in the dialog box.

wgtspec	Description
$\#_1\ \#_2 \ldots \#_J$	J group weights. Weights must be positive and must be integers unless option `nfractional` is specified. Multiple values for each group weight $\#_j$ can be specified as a *numlist* enclosed in parentheses.
matname	matrix with J columns containing J group weights. Multiple rows are allowed, in which case each row corresponds to a different set of J weights or, equivalently, column j corresponds to *numlist* for the jth weight.

where *tablespec* is

column[:*label*] [*column*[:*label*] [. . .]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`N`	total number of subjects	N
`N_per_group`	number of subjects per group	N/N_g
`N_avg`	average number of subjects per group	N_{avg}
`N#`	number of subjects in group #	$N_{\#}$
`delta`	effect size	δ
`N_g`	number of groups	N_g
`N_rep`	number of repeated measurements	N_{rep}
`m#1_#2`	cell mean (#1, #2): group #1, occasion #2	$\mu_{\#_1,\#_2}$
`Var_b`	between-subjects variance	σ^2_b
`Var_w`	within-subject variance	σ^2_w
`Var_bw`	between–within (group-by-occasion) variance	σ^2_{bw}
`Var_be`	between-subjects error variance	σ^2_{be}
`Var_we`	within-subject error variance	σ^2_{we}
`Var_bwe`	between–within (group-by-occasion) error variance	σ^2_{bwe}
`Var_e`	error variance	σ^2_e
`corr`	correlation between repeated measures	ρ
`grwgt#`	group weight #	$w_{\#}$
`target`	target parameter; synonym for target effect variance	
`_all`	display all supported columns	

Column `beta` is shown in the default table in place of column `power` if specified.

Column `N_per_group` is available and is shown in the default table only for balanced designs.

Columns `N_avg` and `N#` are shown in the default table only for unbalanced designs.

Columns `m#1_#2` are not shown in the default table.

Columns `Var_b` and `Var_be` are shown in the default table for the between-subjects test, `Var_w` and `Var_we` for the within-subjects test, and `Var_bw` and `Var_bwe` for the between–within test.

Columns `grwgt#` are not shown in the default table.

Options

Main

`alpha()`, `power()`, `beta()`, `n()`, `nfractional`; see [PSS-2] **power**.

`npergroup(`*numlist*`)` specifies the group size. Only positive integers are allowed. This option implies a balanced design. `npergroup()` cannot be specified with `n()`, `n#()`, or `grweights()`.

`n#(`*numlist*`)` specifies the size of the #th group. Only positive integers are allowed. All group sizes must be specified. For example, all three options `n1()`, `n2()`, and `n3()` must be specified for a design with three groups. `n#()` cannot be specified with `n()`, `npergroup()`, or `grweights()`.

`grweights(`*wgtspec*`)` specifies *J* group weights for an unbalanced design. The weights may be specified either as a list of values or as a matrix, and multiple sets of weights are allowed; see *wgtspec* for details. The weights must be positive and must also be integers unless the `nfractional` option is specified. `grweights()` cannot be specified with `npergroup()` or `n#()`.

`ngroups(#)` specifies the number of groups. This option is required if *meanspec* is not specified. This option is also required for effect-size determination unless `grweights()` is specified. For a one-way repeated-measures ANOVA, specify `ngroups(1)`.

`nrepeated(#)` specifies the number of repeated measurements within each subject. At least two repeated measurements must be specified. This option is required if the `corr()` option is specified and *meanspec* is not specified. This option is also required for effect-size determination unless `covmatrix()` is specified.

`corr(`*numlist*`)` specifies the correlation between repeated measurements. `corr()` cannot be specified with `covmatrix()`. This option requires the `nrepeated()` option unless *meanspec* is specified.

`covmatrix(`*matname*`)` specifies the covariance matrix between repeated measurements. `covmatrix()` cannot be specified with `corr()` or `varerror()`.

`factor(between | within | bwithin)` specifies the effect of interest for which power and sample-size analysis is to be performed. For a one-way repeated-measures ANOVA, only `factor(within)` is allowed and is implied when only one group is specified. In a two-way repeated-measures ANOVA, the tested effects include the between effect or main effect of a between-subjects factor, the within effect or main effect of a within-subject factor, and the between–within effect or interaction effect of the between-subjects factor and the within-subject factor. The default for a two-way repeated design is `factor(between)`.

`vareffect(`*numlist*`)` specifies the variance explained by the tested effect specified in `factor()`. For example, if `factor(between)` is specified, `vareffect()` specifies the variance explained by the between-subjects factor. This option is required if the `factor()` option is specified and *meanspec* is not specified. This option is not allowed with the effect-size determination. Only one of `vareffect()`, `varbetween()`, `varwithin()`, or `varbwithin()` may be specified.

`varbetween(`*numlist*`)` specifies the variance explained by the between-subjects factor. This option is equivalent to specifying `factor(between)` and `vareffect(`*numlist*`)` and thus cannot be combined with `factor()`. This option is not allowed with the effect-size determination. Only one of `vareffect()`, `varbetween()`, `varwithin()`, or `varbwithin()` may be specified. This option is not allowed when only one group is specified.

`varwithin(`*numlist*`)` specifies the variance explained by the within-subject factor. This option is equivalent to specifying `factor(within)` and `vareffect(`*numlist*`)` and thus cannot be combined with `factor()`. This option is not allowed with the effect-size determination. Only one of `vareffect()`, `varbetween()`, `varwithin()`, or `varbwithin()` may be specified.

`varbwithin(`*numlist*`)` specifies the variance explained by the interaction between a between-subjects factor and a within-subject factor. This option is equivalent to specifying `factor(bwithin)` and `vareffect(`*numlist*`)` and thus cannot be combined with `factor()`. This option is not allowed with the effect-size determination. Only one of `vareffect()`, `varbetween()`, `varwithin()`, or `varbwithin()` may be specified. This option is not allowed when only one group is specified.

`varerror(`*numlist*`)` specifies the error variance if `covmatrix()` is not specified. This option is allowed only if `corr()` is specified. When `corr()` is specified, the default is `varerror(1)`.

`showmatrices` specifies that the cell-means matrix and the covariance matrix be displayed, when applicable.

`showmeans` specifies that the cell means be reported. For a text or graphical output, this option is equivalent to `showmatrices` except only the cell-mean matrix will be reported. For a tabular output, the columns containing cell means will be included in the default table.

`parallel`; see [PSS-2] **power**.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies the initial value of the sample size for the sample-size determination or the initial value of the effect size δ for the effect-size determination. The default uses a bisection algorithm to bracket the solution.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following option is available with `power repeated` but is not shown in the dialog box:

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `power repeated` command and the methodology for power and sample-size analysis for one-way and two-way repeated-measures ANOVA. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis and [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Introduction

Repeated-measures ANOVA models are popular among experimenters because of their increased power compared with regular ANOVA models. Repeated-measures designs allow multiple measurements on the same subject. The repeated measurements often correspond to outcomes measured over time for each subject, but they can also correspond to different measurements of the same treatment or measurements of different treatments. The key point is that multiple measurements are made on the same subject.

One example of a repeated-measures design is a longitudinal study that offers an important alternative to a cross-sectional study because of its ability to establish a temporal relationship between the treatment and the outcome. For example, patients with hypertension might be randomized to receive a new experimental drug or standard care and have their systolic blood pressure measured at baseline and each year for two years.

	Baseline	Year 1	Year 2
Old drug	145	135	130
New drug	145	130	120

What makes repeated-measures designs more powerful? Using each subject as his or her own control reduces subject-to-subject variability that is explained by anything other than the effect of the treatment under study. This may dramatically increase power for detecting the effect of the treatment of interest.

Two classes of methods can be used to analyze repeated-measures data: univariate methods such as regular F tests and multivariate methods such as Wilks's lambda test, Pillai's trace test, and the Lawley–Hotelling trace test. The multivariate methods are more flexible in terms of the assumptions about the repeated-measures covariance structure, but they have lower power than regular F tests. In this entry, we concentrate on the univariate methods.

A repeated-measures design belongs to a class of within-subject designs, designs that contain one or more within-subject factors. A within-subject factor is a factor for which each subject receives several or all levels. A between-subjects factor, on the other hand, is any factor for which each subject receives only one level. In what follows, we consider designs with one fixed within-subject factor—one-way fixed-effects repeated-measures ANOVA models—or designs with one fixed between-subjects factor and one fixed within-subjects factor—two-way fixed-effects repeated-measures ANOVA models.

In a one-way repeated-measures model, the within-subject effect is the effect of interest. In a two-way repeated-measures model, you can choose between the three effects of interest: a main between-subjects effect or the between effect, a main within-subject effect or the within effect, and an interaction effect between the between-subjects factor and within-subject factor or the between–within effect. `power repeated` provides power and sample-size computations for the tests of all of these effects.

Repeated-measures ANOVA assumes that errors are normally distributed with zero mean and constant variance. The measurements between subjects are independent, but the measurements within a subject are assumed to be correlated. The within-subject covariance matrices must be constant between groups defined by the levels of between-subjects factors. The validity of the regular F test also relies on the so-called sphericity assumption (or, more generally, the circularity assumption). You can think of this assumption as all differences between levels of the within-subject factor having the same variances. A special case of this assumption is compound symmetry, a less stringent assumption. A covariance matrix is said to have a compound-symmetry structure if all the variances are equal and all the covariances are equal.

The assumption of sphericity is rather restrictive. When it is violated, the distribution of the test statistic of the regular F test of within and between–within effects is no longer an exact F distribution. To compensate for this, several nonsphericity corrections such as the Greenhouse–Geisser correction or Huynh–Feldt correction for the degrees of freedom of the regular F test are proposed (for example, Geisser and Greenhouse [1958]; Huynh and Feldt [1976]).

The distribution of the test statistic under the alternative hypothesis is a noncentral F distribution for all the considered tests. Thus power is a function of the noncentrality parameter, and the noncentrality parameter is a function of the ratio of the variance of the tested effect to the comparison error variance used in the denominator of the corresponding F test. For example, for a test of the within effect, the comparison error variance is the within-effect error variance. In what follows, by comparison error variance, we will imply one of the between-effect, within-effect, or between–within-effect error variance, whichever is appropriate for the considered test. The effect size for each of the F tests is defined as the square root of the ratio of the variance of the tested effect to the comparison error variance.

This entry describes power and sample-size analysis of repeated-measures ANOVA using the univariate F test with Greenhouse–Geisser correction for the nonsphericity.

Using power repeated

`power repeated` computes sample size, power, or effect size for one-way and two-way fixed-effects repeated-measures ANOVA models. A one-way repeated-measures ANOVA model includes one fixed within-subject factor. The supported two-way repeated-measures ANOVA includes one fixed between-subjects factor and one fixed within-subject factor. A one-way model is available as a special case of a two-way model with one group. At least one group and two repeated measures must be specified.

All computations are performed assuming a significance level of 0.05. You may change the significance level by specifying the `alpha()` option.

The computations are performed for an F test of the effect of interest. In a one-way model, the only effect of interest is a within-subject effect. In a two-way model, you can choose between the three effects of interest: between-subjects effect with `factor(between)` (the default), within-subject effect with `factor(within)`, and between–within effect with `factor(bwithin)`.

All computations require that you specify a residual covariance between repeated measures. You can either specify any unstructured covariance matrix in `covmatrix()` or specify the correlation between repeated measures in `corr()` and the error variance in `varerror()`. If `corr()` is specified, `varerror(1)` is assumed. The latter specification implies a residual covariance with compound-symmetry structure.

To compute the total sample size, you must also specify the alternative *meanspec* and, optionally, the power of the test in `power()`. The default power is set to 0.8.

To compute power, you must specify the total sample size in the `n()` option and the alternative *meanspec*.

Instead of the alternative cell means, you can specify the number of groups (rows) in the `ngroups()` option, the number of repeated measures (columns) in the `nrepeated()` option, and the variance explained by the tested effect in the `vareffect()` option when computing sample size or power. See *Alternative ways of specifying effect* in [PSS-2] **power twoway**; substitute `ngroups()` for `nrows()`, `nrepeated()` for `ncols()`, `varbetween()` for `varrow()`, `varwithin()` for `varcolumn()`, and `varbwithin()` for `varrowcolumn()`. If `covmatrix()` is specified, the `nrepeated()` option is not required—the number of repeated measures is determined by the dimensionality of the specified covariance matrix.

To compute effect size, the square root of the ratio of the variance explained by the tested factor to the comparison error variance, and the target variance explained by the tested factor, in addition to the residual covariance, you must specify the total sample size in `n()`, the power in `power()`, the number of groups in `ngroups()`, and the number of repeated measures in `nrepeated()` if `corr()` is specified.

By default, all computations assume a balanced- or an equal-allocation design. You can use `grweights()` to specify an unbalanced design for power, sample-size, or effect-size computations. For power and effect-size computations, you can specify individual group sizes in `n1()`, `n2()`, and so on, instead of a combination of `n()` and `grweights()` to accommodate an unbalanced design. For a balanced design, you can also specify `npergroup()` to specify a group size instead of a total sample size in `n()`.

In repeated-measures ANOVA, sample size and effect size depend on the noncentrality parameter of the F distribution, and their estimation requires iteration. The default initial values are obtained from a bisection search that brackets the solution. If you desire, you may change this by specifying your own value in the `init()` option. See [PSS-2] **power** for the descriptions of other options that control the iteration procedure.

Computing sample size

To compute sample size, you must specify a repeated-measures covariance, an alternative cell means, or the variance of the tested effect and, optionally, the power of the test in the `power()` option. A power of 0.8 is assumed if `power()` is not specified.

▷ Example 1: Sample size for a one-way repeated-measures ANOVA

Consider a version of the study described in Winer, Brown, and Michels (1991, 228). Suppose that researchers would like to conduct a similar study to investigate the effects of three drugs on reaction time to a series of standardized tasks. Per design, each subject will receive all three drugs, and a subject's score (mean reaction time to a task) will be recorded for each of the three drugs; that is, there will be three repeated measurements on each subject. This is a simple one-way repeated-measures design in which drug is the within-subject factor. See Winer, Brown, and Michels (1991) for other details of the design.

Before conducting the study, researchers would like to compute the required sample size to detect the effect of interest with 80% power and a 5% significance level. Suppose that the postulated means for the three drug levels are 26.4, 25.6, and 21; the correlation between repeated measurements is 0.6; and the error variance is 77. We use `power repeated` to compute the sample size:

```
. power repeated 26.4 25.6 21, corr(0.6) varerror(77)

Performing iteration ...

Estimated sample size for repeated-measures ANOVA
F test for within subject with Greenhouse-Geisser correction
H0: delta = 0  versus  Ha: delta != 0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.7426
          N_g =         1
        N_rep =         3
        means =   <matrix>
        Var_w =    5.6622
       Var_we =   10.2667
        Var_e =   77.0000
          rho =    0.6000

Estimated sample sizes:

            N =        20
  N per group =        20
```

We need to recruit 20 subjects to detect the effect size of $0.7426 = \sqrt{5.6622/10.2667}$ in this study.

Repeated-measures covariance in this study has a compound-symmetry structure by design, so the assumption of sphericity, underlying the F test of means for the within-subject factor, is automatically satisfied. Thus no correction to the degrees of freedom of the test is made.

◁

▷ Example 2: Alternative ways of specifying effect and repeated-measures covariance

Instead of specifying the alternative cell means as in example 1, we can specify the variance between them. Here we also need to specify the number of groups and the number of repeated measures. From example 1, the variance between the means was computed as 5.6622. We specify this value in `varwithin()`, the number of groups in `ngroups()`, and the number of repeated measures in `nrepeated()`:

```
. power repeated, ngroups(1) varwithin(5.6622) nrepeated(3) corr(0.6) vararror(77)
Performing iteration ...
Estimated sample size for repeated-measures ANOVA
F test for within subject with Greenhouse-Geisser correction
H0: delta = 0  versus  Ha: delta != 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.7426
          N_g =         1
        N_rep =         3
        Var_w =    5.6622
       Var_we =   10.2667
        Var_e =   77.0000
          rho =    0.6000
Estimated sample sizes:
            N =        20
  N per group =        20
```

We obtain the exact same results as in example 1.

Instead of specifying alternative means directly following the command line, we can define a matrix, say, `M`, containing these means and use it with `power repeated`:

```
. matrix M = (26.4,25.6,21)
. power repeated M, corr(0.6) varerror(77) showmatrices
Performing iteration ...
Estimated sample size for repeated-measures ANOVA
F test for within subject with Greenhouse-Geisser correction
H0: delta = 0  versus  Ha: delta != 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.7426
          N_g =         1
        N_rep =         3
        means =   <matrix>
        Var_w =    5.6622
       Var_we =   10.2667
        Var_e =   77.0000
          rho =    0.6000
Study matrices:
  Cell means
                    |          repeated
                    |        1        2        3
  ------------------+---------------------------
  groups            |
                  1 |     26.4     25.6       21
  Covariance
                    |          repeated
                    |        1        2        3
  ------------------+---------------------------
  repeated          |
                  1 |       77
                  2 |     46.2       77
                  3 |     46.2     46.2       77
Estimated sample sizes:
              N =        20
  N per group =        20
```

We used the `showmatrices` option to display the cell-means matrix and the covariance matrix.

We can also use the `covmatrix()` option to specify the repeated-measures covariance matrix. This option allows you to specify unstructured covariance matrices.

We could have typed the values of the covariance matrix displayed above, but instead, we simply retrieve it from the stored result `r(Cov)`. We then display the values of the covariance matrix to verify that we have the correct matrix.

```
. matrix Cov = r(Cov)
. matlist Cov
               | repeated
               |        1          2          3
---------------+---------------------------------
repeated       |
             1 |       77
             2 |     46.2         77
             3 |     46.2       46.2         77
```

We specify the covariance matrix in `covmatrix()`:

```
. power repeated M, covmatrix(Cov)
Performing iteration ...
Estimated sample size for repeated-measures ANOVA
F test for within subject with Greenhouse-Geisser correction
H0: delta = 0  versus  Ha: delta != 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.7426
          N_g =         1
        N_rep =         3
        means =   <matrix>
        Var_w =    5.6622
       Var_we =   10.2667
          Cov =   <matrix>
    spherical =      true
Estimated sample sizes:
            N =        20
  N per group =        20
```

We obtain the exact same results as before.

◁

▷ Example 3: Sample size for a two-way repeated-measures ANOVA—between effect

A group of researchers would like to design a study to determine whether a new antihypertension medication is more effective than the best medication currently available. They plan their study based on the design and results of the ALLHAT clinical trial (1996, 2002). Average systolic blood pressure (SBP) is assumed to be 145 mm/Hg at baseline in both treatment groups. Using the results of the ALLHAT study, the researchers expect a mean SBP of 135 at year 1 and 130 at year 2 in the old drug group. Using the results of pilot studies, the researchers expect a mean SBP of 130 at year 1 and 120 at year 2 in the new drug group.

	Baseline	Year 1	Year 2
Old drug	145	135	130
New drug	145	130	120

There are two factors in this experiment: treatment group is the between-subjects factor, and measurement time (baseline, year 1, and year 2) is the within-subject factor. Using the ALLHAT study and the pilot data, the researchers assume that the variance of SBP will be 225 for both groups at each of the three measurements. They also assume that the correlation between the repeated measurements is 0.7, so the covariance matrix is

$$\mathbf{\Sigma} = \begin{bmatrix} 225 & 157.5 & 157.5 \\ 157.5 & 225 & 157.5 \\ 157.5 & 157.5 & 225 \end{bmatrix}$$

There are potentially three tests of interest here: the test of the main effect of treatment, the test of the main effect of time, and the test of the interaction effect between treatment and time.

Let's compute the required sample size for the test of the between effect, treatment. This is the default test in `power repeated` when there is more than one group.

We begin by defining a matrix of means and a covariance matrix.

```
. matrix M   = (145,135,130\145,130,120)
. matrix Cov = (225,157.5,157.5\157.5,225,157.5\157.5,157.5,225)
```

We can use the `matlist` command to display these matrices to verify that we typed them correctly:

```
. matlist M
```

	c1	c2	c3
r1	145	135	130
r2	145	130	120

```
. matlist Cov
```

	c1	c2	c3
r1	225		
r2	157.5	225	
r3	157.5	157.5	225

For brevity, we use one of the alternative specifications from example 2 to compute sample size. We specify the cell-means matrix `M` following the command name and the covariance matrix `Cov` in `covmatrix()`:

```
. power repeated M, covmatrix(Cov)
Performing iteration ...
Estimated sample size for repeated-measures ANOVA
F test for between subjects
H0: delta = 0  versus  Ha: delta != 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.1863
          N_g =         2
        N_rep =         3
        means =   <matrix>
        Var_b =    6.2500
       Var_be =  180.0000
          Cov =   <matrix>
Estimated sample sizes:
            N =       228
  N per group =       114
```

To detect the treatment effect of the specified magnitude, $\delta = 0.1863 = \sqrt{6.25/180}$, we need to enroll 228 subjects with 114 subjects per treatment. Note that the sphericity requirement is not needed for the F test of between effects, so no correction is done to the degrees of freedom of the test.

By default, `power repeated` does not display the specified matrices. If desired, we can use the `showmatrices` option to display them:

```
. power repeated M, covmatrix(Cov) showmatrices
Performing iteration ...
Estimated sample size for repeated-measures ANOVA
F test for between subjects
H0: delta = 0  versus  Ha: delta != 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.1863
          N_g =         2
        N_rep =         3
        means =   <matrix>
        Var_b =    6.2500
       Var_be =  180.0000
          Cov =   <matrix>
Study matrices:
  Cell means
                |          repeated
                |         1         2         3
----------------+------------------------------
  groups        |
              1 |       145       135       130
              2 |       145       130       120
  Covariance
                |          repeated
                |         1         2         3
----------------+------------------------------
  repeated      |
              1 |       225
              2 |     157.5       225
              3 |     157.5     157.5       225
Estimated sample sizes:
            N =       228
  N per group =       114
```

Similarly to the alternative specifications discussed in example 2, all the specifications below will produce identical results:

```
. power repeated 145 135 130 \ 145 130 120, covmatrix(Cov)
```
(*output omitted*)

```
. power repeated M, corr(0.7) varerror(225)
```
(*output omitted*)

```
. power repeated, nrepeated(3) corr(0.7) varerror(225) ngroups(2) varbetween(6.25)
```
(*output omitted*)

◁

▷ Example 4: Sample size for a two-way repeated-measures ANOVA—within effect

Continuing with example 3, we now compute the required sample size for the test of the main effects of time, the within effects.

```
. power repeated M, covmatrix(Cov) factor(within)

Performing iteration ...

Estimated sample size for repeated-measures ANOVA
F test for within subject with Greenhouse-Geisser correction
H0: delta = 0  versus  Ha: delta != 0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    1.7392
          N_g =         2
        N_rep =         3
        means =   <matrix>
        Var_w =   68.0556
       Var_we =   22.5000
          Cov =   <matrix>
    spherical =      true

Estimated sample sizes:

            N =         6
  N per group =         3
```

We only need a total of 6 subjects, 3 per group, to detect the within effect in this study.

We can also obtain identical results by using the following alternative specification:

```
. power repeated, covmatrix(Cov) ngroups(2) varwithin(68.0556)
```
(*output omitted*)

◁

Example 5: Sample size for a two-way repeated-measures ANOVA—between–within effect

Continuing with example 3, we can also compute the required sample size for the test of the between–within interaction effects, interaction between treatment and time.

```
. power repeated M, covmatrix(Cov) factor(bwithin)

Performing iteration ...

Estimated sample size for repeated-measures ANOVA
F test for between-within subjects with Greenhouse-Geisser correction
H0: delta = 0  versus  Ha: delta != 0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.4303
          N_g =         2
        N_rep =         3
        means =   <matrix>
       Var_bw =    4.1667
      Var_bwe =   22.5000
          Cov =   <matrix>
    spherical =      true

Estimated sample sizes:

            N =        54
  N per group =        27
```

For this test, we need a total of 54 subjects with 27 subjects per group.

If we are interested in performing all three tests (between, within, and between–within) during our analysis, we should pick the largest of the three sample sizes as our final sample size. In our examples, the largest sample size is 228 for the test of between effects.

We can also obtain results identical to the above by using the following alternative specification:

```
. power repeated, covmatrix(Cov) ngroups(2) varbwithin(4.1667)
```
(*output omitted*)

▷ Example 6: Unbalanced design

Continuing with example 2, suppose we anticipate that the first group will have twice as many subjects as the second group. We can accommodate this unbalanced design by specifying the corresponding group weights in `grweights()`:

```
. power repeated M, covmatrix(Cov) grweights(2 1)

Performing iteration ...

Estimated sample size for repeated-measures ANOVA
F test for between subjects
H0: delta = 0  versus  Ha: delta != 0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.1757
          N_g =         2
        N_rep =         3
        means =   <matrix>
        Var_b =    5.5556
       Var_be =  180.0000
          Cov =   <matrix>

Estimated sample sizes:

            N =       258
    Average N =  129.0000
           N1 =       172
           N2 =        86
```

The required total sample size for this unbalanced design is 258 with 172 subjects in the first group and 86 subjects in the second group. The average number of subjects per group is 129.

We can compute results for multiple sets of group weights. The specification of group weights within `grweights()` is exactly the same as the specification of group means described in *Alternative ways of specifying effect*. Suppose that we would like to compute sample sizes for two unbalanced designs. The first design has twice as many subjects in the first group, and the second design has twice as many subjects in the second group. We specify multiple group weights for the first and second groups in parentheses. We also specify the `parallel` option to treat multiple weight values in parallel instead of computing results for all possible combinations of these values that would have been done by default.

```
. local columns alpha power N N1 N2 grwgt1 grwgt2 delta N_rep Var_b Var_be
. power repeated M, covmatrix(Cov) grweights((2 1) (1 2)) parallel
> table(‘columns’, formats("%6.0g"))
Performing iteration ...
Estimated sample size for repeated-measures ANOVA
F test for between subjects
H0: delta = 0  versus  Ha: delta != 0
   means = <matrix>
     Cov = <matrix>
```

alpha	power	N	N1	N2	grwgt1	grwgt2	delta	N_rep	Var_b	Var_be
.05	.8	258	172	86	2	1	.1757	3	5.556	180
.05	.8	258	86	172	1	2	.1757	3	5.556	180

The default table does not include group weights, so we request a table with custom columns containing group weights via `table()`. We also request a smaller format to make the table more compact.

Computing power

To compute power, you must specify a repeated-measures covariance, the total sample size in `n()`, and the alternative cell means or the variance of the tested effect.

▷ Example 7: Power for a two-way repeated-measures ANOVA

The team discovers that they are only able to recruit a maximum of $n = 200$ participants. They would like to calculate the statistical power for the between-subjects effect given this constraint and assuming a balanced design.

```
. power repeated M, covmatrix(Cov) n(200)
Estimated power for repeated-measures ANOVA
F test for between subjects
H0: delta = 0  versus  Ha: delta != 0
Study parameters:
        alpha =    0.0500
            N =       200
  N per group =       100
        delta =    0.1863
          N_g =         2
        N_rep =         3
        means =   <matrix>
        Var_b =    6.2500
       Var_be =  180.0000
          Cov =   <matrix>
Estimated power:
        power =    0.7462
```

The power corresponding to this design is 75%.

▷ Example 8: Multiple values of study parameters

Continuing with example 7, suppose that the researchers would like to know whether randomizing 60% of the participants to the new drug group and 40% to the old drug group will have an effect on statistical power. For comparison, we will also include the results from a balanced design.

To accommodate this unbalanced design, we could use `grweights()`, as we demonstrated in example 6. For variety, we instead use `n1()` and `n2()` to specify unequal group sizes directly. We also display only a subset of table columns, including power and sample sizes.

```
. power repeated M, covmat(Cov) n1(100 80) n2(100 120) parallel
> table(power N1 N2 N)

Estimated power for repeated-measures ANOVA
F test for between subjects
H0: delta = 0  versus  Ha: delta != 0

   means = <matrix>
     Cov = <matrix>
```

power	N1	N2	N
.7462	100	100	200
.7289	80	120	200

For the specified unbalanced design, the power decreases slightly to 73% from 75%.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

Computing effect size and target variance explained by the tested effect

Sometimes, we may be interested in determining the smallest effect that yields a statistically significant result for prespecified sample size and power. In this case, repeated-measures covariance, power, sample size, the numbers of groups, and possibly the number of repeated measurements must be specified.

The effect size in `power repeated` is defined as a square root of the ratio of the variance explained by the tested effect to the comparison error variance. The effect size and the target variance explained by the tested effect are computed.

▷ Example 9: Effect size for a two-way repeated-measures ANOVA

Continuing with example 7, suppose that researchers would like to know how large the between-subjects variance must be to achieve a power of 80% with a total sample size of 200 using a balanced design.

```
. power repeated, covmat(Cov) n(200) power(0.8) ngroups(2)
Performing iteration ...
Estimated between-subjects variance for repeated-measures ANOVA
F test for between subjects
H0: delta = 0  versus  Ha: delta != 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
            N =       200
  N per group =       100
          N_g =         2
        N_rep =         3
       Var_be =  180.0000
          Cov =   <matrix>
Estimated effect size and between-subjects variance:
        delta =    0.1991
        Var_b =    7.1331
```

We see that to achieve a power of at least 80%, the between-subjects variance must increase to 7.1331 from 6.250, which achieved a power of 0.7462 in example 7. The effect size increases from 0.1863 to 0.1991.

◁

Testing hypotheses about means from multiple dependent populations

After the data are collected, we can use Stata's `anova` command, for example, to perform inference for repeated-measures ANOVA. We show a quick example of how to do this here; see [R] **anova** for more examples and details.

▷ Example 10: One-way repeated-measures ANOVA

Suppose that researchers conduct their study and collect the data. Consider the data from Winer, Brown, and Michels (1991, 228), a version of which was discussed in example 1.

`t43.dta` contains 20 observations of `scores` of 4 repeated measurements identified by the `drug` variable from 5 people identified by the `person` variable. We use the `anova` command to fit a one-way repeated-measures model to these data.

```
. use https://www.stata-press.com/data/r19/t43
(T4.3 -- Winer, Brown, Michels)
. anova score person drug, repeated(drug)
                          Number of obs =         20    R-squared     =  0.9244
                          Root MSE      =    3.06594    Adj R-squared =  0.8803

                   Source | Partial SS         df          MS         F    Prob>F
--------------------------+----------------------------------------------------
                    Model |       1379          7         197     20.96  0.0000
                          |
                   person |      680.8          4       170.2     18.11  0.0001
                     drug |      698.2          3   232.73333     24.76  0.0000
                          |
                 Residual |      112.8         12         9.4
--------------------------+----------------------------------------------------
                    Total |     1491.8         19   78.515789

Between-subjects error term:  person
                     Levels:  5         (4 df)
       Lowest b.s.e. variable:  person
Repeated variable: drug
                                          Huynh-Feldt epsilon        =  1.0789
                                          *Huynh-Feldt epsilon reset to 1.0000
                                          Greenhouse-Geisser epsilon =  0.6049
                                          Box's conservative epsilon =  0.3333

                                                    ------------ Prob > F ------------
                   Source |      df       F      Regular     H-F      G-G        Box
--------------------------+----------------------------------------------------------
                     drug |       3     24.76     0.0000    0.0000   0.0006     0.0076
                 Residual |      12
--------------------------+----------------------------------------------------------
```

We are interested in the test of the effect of `drug`. The regular F test reports a significant result. The `anova` output for the repeated variable `drug`, however, indicates that the sphericity assumption is not met in these data; for example, the Greenhouse–Geisser epsilon of 0.6049 is different from 1.

When the sphericity assumption is not met, the degrees of freedom of a regular F test must be adjusted. Even after the adjustment, the effect of a drug is still significant according to all tests, at least at the 1% level.

To design a new study based on the results of this experiment, we can use `power repeated` to compute the required sample size. To perform this computation, we will need the estimates of the repeated-measures covariance and within-drug score means.

`anova` saves the estimated repeated-measures covariance in `e(Srep)`. We save it to a new matrix `Cov` and display it:

```
. mat Cov = e(Srep)
. matlist Cov
------------------------------------------------------------
             |        c1         c2         c3         c4
-------------+----------------------------------------------
          r1 |      76.8
          r2 |      53.2       42.8
          r3 |      29.2       15.8       14.8
          r4 |        69         47         27         64
------------------------------------------------------------
```

We now use the mean command to estimate means for each of the four drug levels. We store the resulting matrix of means in M:

```
. mean score, over(drug)
Mean estimation                                 Number of obs = 20
```

	Mean	Std. err.	[95% conf.	interval]
c.score@drug				
1	26.4	3.919184	18.19705	34.60295
2	25.6	2.925748	19.47634	31.72366
3	15.6	1.720465	11.99903	19.20097
4	32	3.577709	24.51177	39.48823

```
. mat M = e(b)
```

We now specify the obtained matrices with power repeated to compute the sample size:

```
. power repeated M, covmatrix(Cov)
Performing iteration ...
Estimated sample size for repeated-measures ANOVA
F test for within subject with Greenhouse-Geisser correction
H0: delta = 0  versus  Ha: delta != 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    3.8543
          N_g =         1
        N_rep =         4
        means =   <matrix>
        Var_w =   34.9100
       Var_we =    2.3500
          Cov =   <matrix>
    spherical =     false
Estimated sample sizes:
            N =         4
  N per group =         4
```

We only need 4 subjects to detect the effect of a drug in a study with 80% power and a 5% significance level.

◁

Stored results

power repeated stores the following in r():

Scalars

r(alpha)	significance level
r(power)	power
r(beta)	probability of a type II error
r(delta)	effect size
r(N)	total sample size
r(N_a)	actual sample size
r(N_avg)	average sample size
r(N#)	number of subjects in group #
r(N_per_group)	number of subjects per group
r(N_g)	number of groups
r(nfractional)	1 if nfractional is specified, 0 otherwise
r(balanced)	1 for a balanced design, 0 otherwise
r(grwgt#)	group weight #
r(N_rep)	number of rows
r(m#$_1$_#$_2$)	cell mean (#$_1$, #$_2$)
r(Var_b)	between-subjects variance
r(Var_w)	within-subject variance
r(Var_bw)	between–within subjects, interaction variance
r(Var_be)	between-subjects error variance
r(Var_we)	within-subject error variance
r(Var_bwe)	between–within subjects, interaction error variance
r(Var_e)	error variance
r(spherical)	1 covariance is spherical, 0 otherwise
r(epsilon)	nonsphericity correction
r(epsilon_m)	mean nonsphericity correction
r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise
r(init)	initial value for sample size or effect size
r(maxiter)	maximum number of iterations
r(iter)	number of iterations performed
r(tolerance)	requested parameter tolerance
r(deltax)	final parameter tolerance achieved
r(ftolerance)	requested distance of the objective function from zero
r(function)	final distance of the objective function from zero
r(converged)	1 if iteration algorithm converged, 0 otherwise

Macros

r(type)	test
r(method)	repeated
r(factor)	between, within, or bwithin
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results
r(means)	cell-means matrix
r(Cov)	repeated-measures covariance

Methods and formulas

Consider a sample of n units where each observation comprises q responses based on p predictors. A general linear multivariate model can then be expressed as

$$\mathbf{Y} = \mathbf{XB} + \mathbf{E} \tag{1}$$

where $\mathbf{Y}$ is an $n \times q$ matrix of dependent variables, $\mathbf{X}$ is an $n \times p$ matrix of fixed predictor variables, $\mathbf{B}$ is a $p \times q$ matrix of coefficients, and the error $\mathbf{E}$ is an $n \times q$ matrix where each row is an independent and identically distributed random variable drawn from a q-dimensional multivariate normal with mean $\mathbf{0}$ and a variance–covariance matrix $\mathbf{\Sigma}$. In our repeated measures design, $q = K$ is the number of repeated measures within $p = J$ treatments or groups.

For expositional purposes, consider a two-way repeated-measures design with one between-subjects factor, treatment, and one within-subject factor, time. Suppose we wish to test the effect of a treatment with three levels. The response of each individual is measured at the beginning of the experiment and at three time periods after one of the three treatments is administered. To put this into perspective, we see that $K = 4$ is the number of repeated measures and that $J = 3$ is the number of treatment levels. We can express this model as

$$\begin{bmatrix} y_{1,1} & y_{1,2} & y_{1,3} & y_{1,4} \\ y_{2,1} & y_{2,2} & y_{2,3} & y_{2,4} \\ . & . & . & . \\ . & . & . & . \\ y_{n,1} & y_{n,2} & y_{n,3} & y_{n,4} \end{bmatrix} = \begin{bmatrix} 1 & x_{1,1} & x_{1,2} \\ 1 & x_{2,1} & x_{2,2} \\ . & . & . \\ . & . & . \\ 1 & x_{n,1} & x_{n,2} \end{bmatrix} \begin{bmatrix} \mu_1 & \mu_2 & \mu_3 & \mu_4 \\ \alpha_{1,1} & \alpha_{1,2} & \alpha_{1,3} & \alpha_{1,4} \\ \alpha_{2,1} & \alpha_{2,2} & \alpha_{2,3} & \alpha_{2,4} \end{bmatrix} + \begin{pmatrix} \boldsymbol{\epsilon}_1' \\ \boldsymbol{\epsilon}_2' \\ . \\ . \\ \boldsymbol{\epsilon}_n' \end{pmatrix}$$

where $y_{i,k}$ is the response of the ith individual at time period $k = 1, 2, 3, 4$ and

$$x_{i1} = \begin{cases} 1 & \text{if subject } i \text{ received treatment 1} \\ 0 & \text{if subject } i \text{ received treatment 2} \\ -1 & \text{if subject } i \text{ received treatment 3} \end{cases}$$

$$x_{i2} = \begin{cases} 0 & \text{if subject } i \text{ received treatment 1} \\ 1 & \text{if subject } i \text{ received treatment 2} \\ -1 & \text{if subject } i \text{ received treatment 3} \end{cases}$$

represent the effects of a treatment for individual i. The elements in the coefficient matrix $\mathbf{B}$ have the following interpretation: μ_k is the mean-treatment response at time period k, $\alpha_{j,k}$ is the jth treatment effect, $j = 1, 2$, at time period k, and $\alpha_{3,k} = -\alpha_{1,k} - \alpha_{2,k}$. The treatment-by-time means are $\boldsymbol{\mu} = \mathbf{XB}$. The $\boldsymbol{\epsilon}_i$ are independent normal vectors of length K with mean $\mathbf{0}$ and variance–covariance $\mathbf{\Sigma}$.

Methods and formulas are presented under the following headings:

Hypothesis testing

A hypothesis test for a general linear multivariate model can be formed as

$$H_0\colon \mathbf{\Theta} = \mathbf{0} \qquad H_a\colon \mathbf{\Theta} \neq \mathbf{0}$$

where $\mathbf{\Theta} = \mathbf{CBU}$ is a $d_c \times d_u$ matrix with arbitrary dimensions d_c and d_u that depend on the specified contrast matrices $\mathbf{C}$ and $\mathbf{U}$. $\mathbf{C}$ is a $d_c \times p$ matrix of full rank, where $\text{rank}(\mathbf{C}) = d_c \leq p$, and $\mathbf{U}$ is a $q \times d_u$ matrix of full rank, where $\text{rank}(\mathbf{U}) = d_u \leq q$.

Each row of $\mathbf{C}$ corresponds to the row of $\mathbf{\Theta}$ and forms a contrast to test the between-subjects effects. Similarly, each column of $\mathbf{U}$ corresponds to the column of $\mathbf{\Theta}$ and forms a contrast to test the within-subject effect. Together, $\mathbf{C}$ and $\mathbf{U}$ can also be used to test for interaction effects.

The estimates are given by

$$\widehat{\mathbf{B}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y}$$

$$\widehat{\mathbf{\Theta}} = \mathbf{C}\widehat{\mathbf{B}}\mathbf{U} \tag{2}$$

$$\widehat{\mathbf{H}} = \widehat{\mathbf{\Theta}}'\{\mathbf{C}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{C}'\}^{-1}\widehat{\mathbf{\Theta}} \tag{3}$$

Define $\widehat{\mathbf{E}} = \mathbf{U}'\widehat{\mathbf{\Sigma}}\mathbf{U}(n-p)$. Then, under the assumption of sphericity, the test statistic is given by

$$F_{C,U} = \frac{\text{tr}(\widehat{\mathbf{H}})/d_c d_u}{\text{tr}(\widehat{\mathbf{E}})/\{d_u(n-p)\}} \tag{4}$$

where the statistic follows an F distribution with $d_c d_u$ numerator and $d_u(n-p)$ denominator degrees of freedom. However, if the assumption is not met, then the test statistic follows an F distribution with $d_c d_u \varepsilon$ numerator and $d_u(n-p)\varepsilon$ denominator degrees of freedom, where

$$\varepsilon = \frac{\text{tr}^2(\widehat{\mathbf{\Sigma}})}{d_u \text{ tr}(\widehat{\mathbf{\Sigma}}^2)} = \frac{\left(\sum_{k=1}^{d_u}\lambda_k\right)^2}{d_u \sum_{k=1}^{d_u}\lambda_k^2}$$

Under the alternative hypothesis, the power is obtained using a noncentral F distribution with noncentrality parameter equal to

$$\lambda = d_c d_u \epsilon F_{C,U}$$

Computing power

To compute power, we make conjectures about the parameters of interest, $\mathbf{B}$ and $\mathbf{\Sigma}$, and rewrite (2), (3), and (4) as

$$\begin{aligned}
\mathbf{\Theta} &= \mathbf{CBU} \\
\mathbf{H} &= n\mathbf{\Theta}'\{\mathbf{C}(\ddot{\mathbf{X}}'\mathbf{W}\ddot{\mathbf{X}})^{-1}\mathbf{C}'\}^{-1}\mathbf{\Theta} \\
&= n\mathbf{H}_* \\
\mathbf{E} &= \mathbf{U}'\mathbf{\Sigma}\mathbf{U}(n-p) \\
&= \mathbf{\Sigma}_*(n-p)
\end{aligned}$$

where $\ddot{\mathbf{X}}$ is the $p \times p$ model matrix containing all the unique rows of $\mathbf{X}$ in a special order and $\mathbf{W}$ is a diagonal matrix containing n_j/n, the sample size for the jth treatment divided by the total sample size. In our three-treatment example, the matrix $\ddot{\mathbf{X}}$ is

$$\ddot{\mathbf{X}} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & -1 & -1 \end{pmatrix}$$

The $F_{C,U}$ statistic using the parameter matrices $\mathbf{H}$ and $\mathbf{E}$ is

$$\begin{aligned} F_{C,U} &= \frac{\text{tr}(\mathbf{H})/d_c d_u}{\text{tr}(\mathbf{E})/\{d_u(n-p)\}} \\ &= \frac{n}{d_c}\frac{\text{tr}(\mathbf{H}_*)}{\text{tr}(\mathbf{\Sigma}_*)} \end{aligned}$$

from which we obtain the noncentrality parameter as

$$\begin{aligned} \lambda &= d_u d_c \epsilon F_{C,U} \\ &= n\epsilon\delta^2 \end{aligned} \tag{5}$$

where the effect size δ is defined as $\delta = \sqrt{d_u \text{tr}(\mathbf{H}_*)/\text{tr}(\mathbf{\Sigma}_*)}$.

The effect variance (`Var_b`, `Var_w`, or `Var_bw`) reported by `power repeated` is computed as $\text{tr}(\mathbf{H}_*)/d_c$. The effect error variance (`Var_be`, `Var_we`, or `Var_bwe`) is computed as $\text{tr}(\mathbf{\Sigma}_*)/(d_c d_u)$.

Under the alternative hypothesis, the test statistic in (4) is distributed as a noncentral F distribution with $d_c d_u \epsilon$ numerator and $d_u(n-p)\epsilon$ denominator degrees of freedom and noncentrality parameter λ from (5).

The power of the overall F test is

$$1 - \beta = F_{d_c d_u \epsilon, d_u(n-p)\epsilon, \lambda}\left(F_{d_c d_u \epsilon_m, d_u(n-p)\epsilon_m, 1-\alpha}\right) \tag{6}$$

where $F_{\cdot,\cdot,\lambda}(\cdot)$ is the cdf of a noncentral F distribution, and $\epsilon_m = E(\epsilon)$ is computed as described in Muller and Barton (1989, 551).

Total sample size and effect size are obtained by iteratively solving the nonlinear equation (6). When the `grweights()` option is specified, a constant multiplier n_c is computed and rounded to an integer unless the `nfractional` option is specified. The group sizes are then computed as $\widetilde{w}_j n_c$, where $\widetilde{w}$ is a standardized weight; see *Methods and formulas* of [PSS-2] **power oneway** for details. The actual sample size, `N_a`, is the sum of the group sizes.

See Muller et al. (1992) for details.

References

ALLHAT Officers and Coordinators for the ALLHAT Collaborative Research Group. 2002. Major outcomes in high-risk hypertensive patients randomized to angiotensin-converting enzyme inhibitor or calcium channel blocker vs diuretic: The antihypertensive and lipid-lowering treatment to prevent heart attack trial (ALLHAT). *Journal of the American Medical Association* 288: 2981–2997. https://doi.org/10.1001/jama.288.23.2981.

Davis, B. R., J. A. Cutler, D. J. Gordon, C. D. Furberg, J. T. Wright, Jr., W. C. Cushman, R. H. Grimm, J. LaRosa, P. K. Whelton, H. M. Perry, M. H. Alderman, C. E. Ford, S. Oparil, C. Francis, M. Proschan, S. Pressel, H. R. Black, and C. M. Hawkins, for the ALLHAT Research Group. 1996. Rationale and design for the antihypertensive and lipid lowering treatment to prevent heart attack trial (ALLHAT). *American Journal of Hypertension* 9: 342–360. https://doi.org/10.1016/0895-7061(96)00037-4.

Geisser, S., and S. W. Greenhouse. 1958. An extension of Box's results on the use of the F distribution in multivariate analysis. *Annals of Mathematical Statistics* 29: 885–891. https://doi.org/10.1214/aoms/1177706545.

Grayling, M. J., J. M. S. Wason, and A. P. Mander. 2018. Group sequential clinical trial designs for normally distributed outcome variables. *Stata Journal* 18: 416–431.

Huynh, H., and L. S. Feldt. 1976. Estimation of the Box correction for degrees of freedom from sample data in randomized block and split-plot designs. *Journal of Educational Statistics* 1: 69–82. https://doi.org/10.2307/1164736.

Muller, K. E., and C. N. Barton. 1989. Approximate power for repeated-measures ANOVA lacking sphericity. *Journal of the American Statistical Association* 84: 549–555. https://doi.org/10.2307/2289941.

Muller, K. E., L. M. LaVange, S. Landesman Ramey, and C. T. Ramey. 1992. Power calculations for general linear multivariate models including repeated measures applications. *Journal of the American Statistical Association* 87: 1209–1226. https://doi.org/10.1080/01621459.1992.10476281.

Winer, B. J., D. R. Brown, and K. M. Michels. 1991. *Statistical Principles in Experimental Design*. 3rd ed. New York: McGraw–Hill.

Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power oneway** — Power analysis for one-way analysis of variance

[PSS-2] **power pairedmeans** — Power analysis for a two-sample paired-means test

[PSS-2] **power twoway** — Power analysis for two-way analysis of variance

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[R] **anova** — Analysis of variance and covariance

power oneslope — Power analysis for a slope test in a simple linear regression

Description

`power oneslope` computes sample size, power, or the target slope coefficient for a test of a slope in a simple linear regression. By default, it computes sample size given power and the slope coefficient. Alternatively, it computes power given sample size and the slope coefficient, or it computes the target slope coefficient given sample size, power, and the coefficient under the null. See [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Quick start

Sample size for a two-sided test of H_0: $b = 1.0$ versus H_a: $b \neq 1.0$ with null slope $b_0 = 1.0$, alternative slope $b_a = 1.2$, covariate standard deviation of 3, and error standard deviation of 1.5 using default power of 0.8 and significance level $\alpha = 0.05$

```
power oneslope 1.0 1.2, sdx(3) sderror(1.5)
```

Same as above, but for a one-sided test with power of 0.9

```
power oneslope 1.0 1.2, sdx(3) sderror(1.5) power(.9) onesided
```

Specify b_0 and the difference between b_a and b_0 instead of specifying $b_a = 1.2$

```
power oneslope 1.0, sdx(3) sderror(1.5) diff(0.2)
```

Same as above, but specify correlation between the dependent variable and the covariate instead of the error standard deviation

```
power oneslope 1.0, sdx(3) corr(.923) diff(0.2)
```

Power for sample size of 75

```
power oneslope 1.0 1.2, sdx(3) sderror(1.5) n(75)
```

Power for sample sizes of 50, 60, 70, and 80

```
power oneslope 1.0 1.2, sdx(3) sderror(1.5) n(50(10)80)
```

Same as above, but display results in a graph of power versus sample size

```
power oneslope 1.0 1.2, sdx(3) sderror(1.5) n(50(10)80) graph
```

Effect size and target slope for sample size of 40, power of 0.9, and $\alpha = 0.01$

```
power oneslope 1.0, sdx(3) sderror(1.5) n(40) power(.9) alpha(0.01)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
power oneslope b0 ba [ , power(numlist) options ]
```

Compute power

```
power oneslope b0 ba , n(numlist) [ options ]
```

Compute effect size and target slope

```
power oneslope b0 , n(numlist) power(numlist) [ options ]
```

where b_0 is the null (hypothesized) slope or the value of the slope coefficient under the null hypothesis and b_a is the alternative (target) slope or the value of the slope coefficient under the alternative hypothesis. b_0 and b_a may each be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

options	Description
Main	
* alpha(*numlist*)	significance level; default is alpha(0.05)
* power(*numlist*)	power; default is power(0.8)
* beta(*numlist*)	probability of type II error; default is beta(0.2)
* n(*numlist*)	sample size; required to compute power or effect size
nfractional	allow fractional sample size
* diff(*numlist*)	difference between the alternative slope and the null slope coefficients, $b_a - b_0$; specify instead of the alternative slope b_a
* sdx(*numlist*)	standard deviation of the covariate of interest; default is sdx(1)
* sderror(*numlist*)	standard deviation of the error term of the regression model; may not be combined with sdy() or corr(); default is sderror(1)
* sdy(*numlist*)	standard deviation of the dependent variable; may not be combined with sderror() or corr()
* corr(*numlist*)	correlation between the dependent variable and the covariate of interest; may not be combined with sderror() or sdy()
direction(upper|lower)	direction of the effect for effect-size determination; default is direction(upper), which means that the postulated value of the parameter is larger than the hypothesized value
onesided	one-sided test; default is two sided
parallel	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[no]table[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
saving(*filename* [, replace])	save the table data to *filename*; use replace to overwrite existing *filename*
Graph	
graph[(*graphopts*)]	graph results; see [PSS-2] **power, graph**
Iteration	
init(#)	initial value for sample size or slope
iterate(#)	maximum number of iterations; default is iterate(500)
tolerance(#)	parameter tolerance; default is tolerance(1e-12)
ftolerance(#)	function tolerance; default is ftolerance(1e-12)
[no]log	suppress or display iteration log
[no]dots	suppress or display iterations as dots
notitle	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the parallel option.

collect is allowed; see **[U] 11.1.10 Prefix commands**.

notitle does not appear in the dialog box.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`N`	number of subjects	N
`delta`	effect size	δ
`b0`	null slope coefficient	b_0
`ba`	alternative slope coefficient	b_a
`diff`	difference between alternative and null slope coefficients	$b_a - b_0$
`sdx`	standard deviation of covariate	σ_x
`sderror`	standard deviation of error term	σ
`sdy`	standard deviation of dependent variable	σ_y
`corr`	correlation between dependent variable and covariate	ρ
`target`	target parameter; synonym for `ba`	
`_all`	display all supported columns	

Column `beta` is shown in the default table in place of column `power` if specified.

Columns `diff`, `sdy`, and `corr` are shown in the default table if specified.

Options

Main

`alpha()`, `power()`, `beta()`, `n()`, `nfractional`; see [PSS-2] **power**. The `nfractional` option is allowed only for sample-size determination.

`diff(`*numlist*`)` specifies the difference between the alternative slope and the null slope coefficients, $b_a - b_0$. You can specify either the alternative slope b_a as a command argument or the difference between the two slopes in `diff()`. If you specify `diff(#)`, the alternative slope is computed as $b_a = b_0 + \#$. This option is not allowed with effect-size determination.

`sdx(`*numlist*`)` specifies the standard deviation for the covariate of interest. The default is `sdx(1)`.

`sderror(`*numlist*`)` specifies the standard deviation of the error term of the regression model. The default is `sderror(1)`. This option may not be combined with `sdy()` or `corr()`.

`sdy(`*numlist*`)` specifies the standard deviation of the dependent variable in the regression model. This option may not be combined with `sderror()` or `corr()`.

`corr(`*numlist*`)` specifies the correlation between the covariate of interest and the dependent variable. This option may not be combined with `sderror()` or `sdy()`.

`direction()`, `onesided`, `parallel`; see [PSS-2] **power**.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies the initial value of the sample size for the sample-size determination or the initial value of the slope for the effect-size determination. The default is to use a closed-form normal approximation to compute an initial value for the sample size and a bisection search method to compute an initial value for the effect size.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following option is available with `power oneslope` but is not shown in the dialog box:

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

`power oneslope` computes sample size, power, and the target slope coefficient for a linear regression slope test. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis, and see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Introduction

In a simple linear regression, researchers estimate the relationship between a covariate and the outcome without controlling for other covariates. Consider a study where a researcher would like to compare the slope from a simple linear regression with a hypothesized slope. For example, health researchers may want to know whether the effect of time spent exercising on body mass index (BMI) has changed compared with 10 years ago. Or pharmacological researchers may be interested in knowing whether the slope of the dose–response curve reaches a given magnitude.

We can test whether the covariate of interest, x, has an effect on the outcome, y, by using a t test. In the case of the dose–response study, for example, we may have a model,

$$y_i = a + b_{\text{dose}} x_i + \varepsilon_i \qquad i = 1, 2, \ldots, n$$

where the errors ε_i's are independently and normally distributed with mean 0 and standard deviation σ. The null hypothesis would be H_0: $b_{\text{dose}} = b_0$, where b_{dose} is the slope coefficient for the dose of the drug that was administered and b_0 is the hypothesized value of the slope. In general, for a slope coefficient b, we consider the null hypothesis H_0: $b = b_0$ versus the two-sided alternative hypothesis H_a: $b \neq b_0$, the upper one-sided alternative H_a: $b > b_0$, or the lower one-sided alternative H_a: $b < b_0$.

The power oneslope command provides power and sample-size analysis for the comparison of a slope coefficient with a reference value using a t test in a simple linear regression. For power and sample-size analysis in a multiple linear regression, see [PSS-2] **power rsquared** and [PSS-2] **power pcorr**.

Using power oneslope

power oneslope computes sample size, power, or the target slope, b_a, for a test of slope in a simple linear regression. By default, all computations are performed for a two-sided hypothesis test where the significance level is set to 0.05. You can change the significance level by specifying the alpha() option. You can request a one-sided test by specifying the onesided option.

By default, all computations use one as the standard deviation of the covariate of interest and as the standard deviation of the error. You can change these values by specifying the sdx() and sderror() options. Instead of the sderror() option, you can combine the sdx() option with the sdy() option to specify the standard deviation of the dependent variable or combine the sdx() option with the corr() option to specify the correlation between the covariate and the dependent variable.

To compute sample size, you must specify the slope under the null hypothesis (b_0) and the slope under the alternative hypothesis (b_a), and you may specify the power of the test in the power() option. The default power is set to 0.8.

To compute power, you must specify the sample size in the n() option, the null slope b_0, and the alternative slope b_a.

When computing sample size or power, you may specify the difference between the alternative slope and null slope, $b_a - b_0$, in the diff() option instead of specifying the alternative slope.

To compute the effect size δ, you must specify the sample size in the n() option, the power in the power() option, and the null slope b_0, and you may specify the direction of the effect. δ is defined as the difference between the alternative and null values of the slope multiplied by the ratio of standard deviation of the covariate to that of the error term, $\delta = (b_a - b_0)\sigma_x/\sigma$. The direction is upper by default, direction(upper), which means that the target slope is assumed to be larger than the specified null value. This is also equivalent to the assumption of a positive effect size. You may change the direction to lower by specifying the direction(lower) option, which means that the target slope is assumed to be smaller than the specified null value. This is equivalent to assuming a negative effect size.

By default, the computed sample size is rounded up. You can specify the nfractional option to see the corresponding fractional sample size; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. The nfractional option is allowed only for sample-size determination.

power oneslope's computations of sample size and effect size require iteration. A noncentral Student's t distribution is used for the computations. The degrees of freedom depends on the sample size, and the noncentrality parameter depends on the sample size and effect size. The default initial value of the sample size is obtained using a closed-form normal approximation. The default initial value of the effect size is obtained using a bisection search method. The default initial values may be changed by specifying the init() option. See [PSS-2] **power** for the descriptions of other options that control the iteration procedure.

Computing sample size

To compute sample size, you must specify the slope coefficients under the null, b_0, and alternative, b_a, hypotheses, and you may specify the power of the test in the power() option. A default power of 0.8 is assumed if power() is not specified.

Example 1: Sample size for a linear regression slope test

Consider an example from Dupont and Plummer (1998) that discusses the effectiveness of a dieting program in encouraging patients to exercise regularly. Suppose we are conducting a similar study and we want to know the effect of average time spent per day exercising on BMI, measured in kg/m^2, after six months in the program. The parameter of interest is the slope coefficient b, which measures the effect of exercising on BMI, and our null hypothesis is H_0: $b = 0$ versus a two-sided H_a: $b \neq 0$.

We wish to compute the sample size required to detect a drop in BMI of $0.0667\ \text{kg}/\text{m}^2$ per minute of exercise, the effect reported in Dupont and Plummer, with 80% power using a 5%-level two-sided test. Using values obtained from previous studies, we specify a standard deviation of 7.5 minutes for time spent exercising in `sdx()` and $4.0\ \text{kg}/\text{m}^2$ for BMI in `sdy()`.

```
. power oneslope 0 -0.0667, sdx(7.5) sdy(4)

Performing iteration ...

Estimated sample size for a linear regression slope test
t test
H0: b = b0  versus  Ha: b != b0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =   -0.1261
           b0 =    0.0000
           ba =   -0.0667
          sdx =    7.5000
      sderror =    3.9686
          sdy =    4.0000

Estimated sample size:

            N =       496
```

We find that a sample of 496 subjects is required to detect a drop of $0.0667\ \text{kg}/\text{m}^2$ BMI per minute with 80% power using a 5%-level two-sided test. The effect size (`delta`) and standard deviation of the error term (`sderror`) are calculated using the given information about the null and alternative slopes and the standard deviations of the dependent and independent variables; see *Methods and formulas*.

As we mentioned in *Using power oneslope*, sample-size computation requires iteration. The iteration log is suppressed by default, but you can display it by specifying the `log` option.

◁

Example 2: Specifying difference between slopes

Instead of specifying the alternative slope as in example 1, we can obtain the same results by specifying the difference between the slope coefficient under the alternative and null hypotheses in the `diff()` option. We specify `diff(-0.0667)` and the same standard deviation of the covariate of interest in the `sdx()` option and standard deviation of the outcome in the `sdy()` option that we used in example 1.

In this case, the difference is equal to b_a, specified in example 1. If we wanted to test for a null value other than zero, then b_a would not equal the difference.

```
. power oneslope 0, diff(-0.0667) sdx(7.5) sdy(4)
Performing iteration ...
Estimated sample size for a linear regression slope test
t test
H0: b = b0  versus  Ha: b != b0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =   -0.1261
           b0 =    0.0000
           ba =   -0.0667
         diff =   -0.0667
          sdx =    7.5000
      sderror =    3.9686
          sdy =    4.0000
Estimated sample size:
            N =       496
```

Although the sample size estimate is the same, the difference between the alternative and null values is also reported in the output when we specify the `diff()` option. Notice that the command computed the value of b_a.

◁

▷ Example 3: Specifying standard deviation of the error

Instead of σ_y, the standard deviation of the dependent variable, we can specify the standard deviation of the error term in the regression model using the `sderror()` option. The value of the standard deviation is obtained from the relation $\sigma = \sqrt{\sigma_y^2 - b_a^2\sigma_x^2} = \sqrt{4^2 - \{(-0.0667)^2 \times 7.5^2\}} \approx 3.9686$.

```
. power oneslope 0 -0.0667, sdx(7.5) sderror(3.9686)
Performing iteration ...
Estimated sample size for a linear regression slope test
t test
H0: b = b0  versus  Ha: b != b0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =   -0.1261
           b0 =    0.0000
           ba =   -0.0667
          sdx =    7.5000
      sderror =    3.9686
Estimated sample size:
            N =       496
```

◁

▷ Example 4: Specifying correlation

We can also specify the correlation between time per day spent exercising and BMI using the `corr()` option. The value of the correlation ρ is obtained from the relation $\rho = b_a \times (\sigma_x/\sigma_y) = -0.0667 \times (7.5/4) \approx -0.1251$.

```
. power oneslope 0 -0.0667, sdx(7.5) corr(-0.1251)
Performing iteration ...
Estimated sample size for a linear regression slope test
t test
H0: b = b0  versus  Ha: b != b0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =   -0.1261
           b0 =    0.0000
           ba =   -0.0667
          sdx =    7.5000
      sderror =    3.9674
         corr =   -0.1251
Estimated sample size:
            N =       496
```

The sample sizes in the two examples now are the same, but the standard deviation of the errors is slightly different because of limited precision of the specified correlation.

◁

Computing power

To compute power, you must specify the sample size in the `n()` option and the slopes under the null, b_0, and alternative, b_a, hypotheses.

▷ Example 5: Power of a linear regression slope test

Continuing with example 1, suppose that we are designing a new study and anticipate that we will observe a similar effect but that we will be able to recruit only a sample of 400 subjects. Given the study parameters from example 1, we compute the power by specifying the sample size of 400 in the `n()` option:

```
. power oneslope 0 -0.0667, n(400) sdx(7.5) sdy(4)
Estimated power for a linear regression slope test
t test
H0: b = b0  versus  Ha: b != b0
Study parameters:
        alpha =    0.0500
            N =       400
        delta =   -0.1261
           b0 =    0.0000
           ba =   -0.0667
          sdx =    7.5000
      sderror =    3.9686
          sdy =    4.0000
Estimated power:
        power =    0.7106
```

With a smaller sample size, the power of the test decreases to about 71%.

◁

▷ Example 6: Multiple values of study parameters

To investigate the effect of sample size on power, we can specify a list of sample sizes in the `n()` option:

```
. power oneslope 0 -0.0667, n(50 100 200 400 800) sdx(7.5) sdy(4)
Estimated power for a linear regression slope test
t test
H0: b = b0  versus  Ha: b != b0
```

alpha	power	N	delta	b0	ba	sdx	sderror	sdy
.05	.141	50	-.1261	0	-.0667	7.5	3.969	4
.05	.239	100	-.1261	0	-.0667	7.5	3.969	4
.05	.4263	200	-.1261	0	-.0667	7.5	3.969	4
.05	.7106	400	-.1261	0	-.0667	7.5	3.969	4
.05	.9453	800	-.1261	0	-.0667	7.5	3.969	4

As expected, when the sample size increases, the power tends to get closer to 1.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

Computing effect size and target slope

The effect size δ for a linear regression slope test is defined as the difference between the alternative and null values of the slope multiplied by the ratio of standard deviations of the covariate to the error term, $\delta = (b_a - b_0)\sigma_x/\sigma$.

Sometimes, we may want to determine the minimum detectable effect and the corresponding alternative or target slope coefficient that yield a statistically significant result for a prespecified sample size and power. In this case, we must specify power, sample size, and the null slope coefficient. In addition, we must also decide on the direction of the effect: upper, which means $b_a > b_0$, or lower, which means $b_a < b_0$. The default direction is upper but may be changed by specifying `direction(lower)`.

▷ Example 7: Minimum detectable value of the slope coefficient

Continuing with example 5, we may also be interested in finding the minimum value of the slope coefficient that can be detected with a sample of 400 subjects and 80% power. To compute this, we specify the null value of 0 as the command argument and the required options `n(400)` and `power(0.8)` and continue to use `sdx(7.5)` and `sdy(4)`.

```
. power oneslope 0, n(400) power(0.8) sdx(7.5) sdy(4)
Performing iteration ...
Estimated target slope for a linear regression slope test
t test
H0: b = b0  versus  Ha: b != b0; ba > b0
Study parameters:
        alpha =    0.0500
        power =    0.8000
            N =       400
           b0 =    0.0000
          sdx =    7.5000
      sderror =    3.9611
          sdy =    4.0000
Estimated effect size and target slope:
        delta =    0.1404
           ba =    0.0742
```

The minimum detectable value of the slope coefficient is 0.0742, which corresponds to an effect size of 0.1404. Compared with example 1, this example uses a smaller sample size of 400 subjects, so the minimum detectable difference for the slope coefficient is larger than the absolute value of the alternative slope specified in example 1.

In this example, we assumed the effect to be in the upper direction. By symmetry, the effect size in the lower direction will be −0.1404, which can be obtained by specifying `direction(lower)`.

◁

Performing hypothesis tests on the slope coefficient

Suppose we wish to test the hypothesis that the slope coefficient is different from a reference value on the collected data. We can use the `regress` command to estimate the slope and perform a hypothesis test.

▷ Example 8: Testing for the slope coefficient

Suppose our study goal is to investigate the effect that the weight of cars (`weight`) has on their mileage (`mpg`). We use `regress` with the `beta` option to estimate the standardized slope using data from `auto.dta`:

```
. use https://www.stata-press.com/data/r19/auto
(1978 automobile data)
. regress mpg weight, beta
```

Source	SS	df	MS	Number of obs	=	74
				F(1, 72)	=	134.62
Model	1591.9902	1	1591.9902	Prob > F	=	0.0000
Residual	851.469256	72	11.8259619	R-squared	=	0.6515
				Adj R-squared	=	0.6467
Total	2443.45946	73	33.4720474	Root MSE	=	3.4389

mpg	Coefficient	Std. err.	t	P>|t|	Beta
weight	-.0060087	.0005179	-11.60	0.000	-.8071749
_cons	39.44028	1.614003	24.44	0.000	.

The standardized slope is about -0.81. The p-value < 0.001 for the test of the slope gives us statistical evidence to reject the null hypothesis of H_0: $b_{\text{weight}} = 0$ versus a two-sided alternative H_a: $b_{\text{weight}} \neq 0$ at the 1% significance level.

Suppose we wish to design a new similar study. We use the estimated standardized slope from this study to perform a sample-size analysis for our new study. Because we use the standardized slope, we can use the default value of 1 for `sdx()`, and we specify `sdy(1)`.

```
. power oneslope 0 -0.81, sdy(1)

Performing iteration ...

Estimated sample size for a linear regression slope test
t test
H0: b = b0  versus  Ha: b != b0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =   -1.3812
           b0 =    0.0000
           ba =   -0.8100
          sdx =    1.0000
      sderror =    0.5864
          sdy =    1.0000

Estimated sample size:

            N =         7
```

We find that a sample size of only 7 is required to detect a slope coefficient of -0.81 with 80% power using a 5%-level two-sided test.

◁

Stored results

`power oneslope` stores the following in `r()`:

Scalars

`r(alpha)`	significance level
`r(power)`	power
`r(beta)`	probability of a type II error
`r(delta)`	effect size
`r(N)`	sample size
`r(nfractional)`	1 if `nfractional` is specified, 0 otherwise
`r(onesided)`	1 for a one-sided test, 0 otherwise
`r(b0)`	slope coefficient under the null hypothesis
`r(ba)`	slope coefficient under the alternative hypothesis
`r(diff)`	difference between the alternative and null slopes
`r(sdx)`	standard deviation of the covariate of interest
`r(sderror)`	standard deviation of the error term of the regression model
`r(sdy)`	standard deviation of the dependent variable
`r(corr)`	correlation between the covariate of interest and the dependent variable
`r(separator)`	number of lines between separator lines in the table
`r(divider)`	1 if `divider` is requested in the table, 0 otherwise
`r(init)`	initial value for sample size or for slope
`r(maxiter)`	maximum number of iterations
`r(iter)`	number of iterations performed
`r(tolerance)`	requested parameter tolerance
`r(deltax)`	final parameter tolerance achieved
`r(ftolerance)`	requested distance of the objective function from zero

r(function)	final distance of the objective function from zero
r(converged)	1 if iteration algorithm converged, 0 otherwise
Macros	
r(type)	test
r(method)	oneslope
r(direction)	upper or lower
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats
Matrices	
r(pss_table)	table of results

Methods and formulas

Consider a simple linear regression model of a dependent variable y on a single covariate x,

$$y_i = a + bx_i + \varepsilon_i$$

for $i = 1, \ldots, n$, where i denotes an individual observation in a sample of n subjects, a is the intercept, b is the slope coefficient, and ε_i is the error term. ε_i's are independently and normally distributed with mean 0 and standard deviation σ. Let b_0 denote the null value of the slope coefficient and b_a denote the alternative value of the slope coefficient.

Testing a linear regression slope involves testing the null hypothesis H_0: $b = b_0$ versus the two-sided alternative hypothesis H_a: $b \neq b_0$, the upper one-sided alternative H_a: $b > b_0$, or the lower one-sided alternative H_a: $b < b_0$.

Let σ_y denote the standard deviation of y, σ_x denote the standard deviation of x, and ρ denote the correlation between y and x. These terms have the following relations with the standard deviation of the error term σ (Dupont and Plummer 1998):

$$\sigma = b\sigma_x\sqrt{\frac{1}{\rho^2} - 1} = \sqrt{\sigma_y^2 - b^2\sigma_x^2}$$

Let $\hat{b}$ denote the estimator of the population slope coefficient b, $\hat{\sigma}$ denote the estimated standard deviation of the error term, and $s = \hat{\sigma}/\sigma_x$ denote the estimated standard error of $\sqrt{n}\ \hat{b}$. The sampling distribution of the test statistic $t = \sqrt{n}(\hat{b} - b_0)/s$ under the null hypothesis follows a Student's t distribution with $n - 2$ degrees of freedom.

Let α be the significance level, β be the probability of a type II error, and $t_{n-2,\alpha}$ denote the αth quantile of a Student's t distribution with $n - 2$ degrees of freedom. Let $\delta = (b_a - b_0)\sigma_x/\hat{\sigma}$ denote the effect size. Under the alternative hypothesis, the test statistic follows a noncentral Student's t distribution.

The power $\pi = 1 - \beta$ is computed using

$$\pi = \begin{cases} 1 - T_{n-2,\lambda}\left(t_{n-2,1-\alpha}\right) & \text{for an upper one-sided test} \\ T_{n-2,\lambda}\left(-t_{n-2,1-\alpha}\right) & \text{for a lower one-sided test} \\ 1 - T_{n-2,\lambda}\left(t_{n-2,1-\alpha/2}\right) + T_{n-2,\lambda}\left(-t_{n-2,1-\alpha/2}\right) & \text{for a two-sided test} \end{cases} \quad (1)$$

where $T_{n-2,\lambda}(\cdot)$ is the cumulative noncentral Student's t distribution with a noncentrality parameter $\lambda = \sqrt{n}\delta$.

Sample size and minimum detectable value of the slope are obtained by iteratively solving nonlinear equations in (1) for n and δ. The default initial value for the sample size is calculated using the following closed-form expression,

$$n = \left(\frac{z_{1-\alpha/k} - z_{\beta}}{\delta}\right)^2$$

where $k = 1$ for a one-sided and $k = 2$ for a two-sided test, respectively, and $z_{1-\alpha/k}$ and z_{β} are the $(1-\alpha/k)$th and the βth quantiles of the standard normal distribution. If the `nfractional` option is not specified, the computed sample size is rounded up.

References

Dupont, W. D., and W. D. Plummer, Jr. 1998. Power and sample size calculations for studies involving linear regression. *Controlled Clinical Trials* 19: 589–601. https://doi.org/10.1016/S0197-2456(98)00037-3.

Grayling, M. J., J. M. S. Wason, and A. P. Mander. 2018. Group sequential clinical trial designs for normally distributed outcome variables. *Stata Journal* 18: 416–431.

Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power pcorr** — Power analysis for a partial-correlation test in a multiple linear regression

[PSS-2] **power rsquared** — Power analysis for an R^2 test in a multiple linear regression

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[R] **regress** — Linear regression

[R] **test** — Test linear hypotheses after estimation

power rsquared — Power analysis for an R^2 test in a multiple linear regression

Description

`power rsquared` computes sample size, power, or target R^2 for an R^2 test in a multiple linear regression. An R^2 test is an F test for the coefficient of determination, R^2, which is used to test the significance of all coefficients or of a subset of coefficients in a regression model.

By default, `power rsquared` computes sample size for a test of all coefficients given power and the R^2 of the tested model, R^2_T. Instead of the sample size, it can compute power given sample size and R^2_T or the target R^2_T given sample size and power.

If the number of control covariates is provided, `power rsquared` computes sample size for a test of a subset of coefficients given power, the R^2 of the full model, R^2_F, and the R^2 of the reduced model, R^2_R. It can also compute power given sample size, R^2_R, and R^2_F or the target R^2_F given sample size, power, and R^2_R.

See [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Quick start

Testing all coefficients

Sample size for a test of H_0: $R^2_T = 0$ versus H_a: $R^2_T \neq 0$ given alternative R^2_T of 0.10 and 2 tested covariates using default power 0.8 and significance level $\alpha = 0.05$

```
power rsquared 0.10, ntested(2)
```

Also use values of R^2_T equal to 0.11, 0.12, 0.13, and 0.14, and display results in a table

```
power rsquared (0.10(0.01)0.14), ntested(2)
```

Same as above, but display results in a graph of sample size versus R^2_T

```
power rsquared (0.10(0.01)0.14), ntested(2) graph
```

Power for a sample size of 80

```
power rsquared 0.10, ntested(2) n(80)
```

Effect size and target R^2_T for a sample size of 80 with power 0.9

```
power rsquared, ntested(2) n(80) power(0.9)
```

Testing a subset of coefficients

Sample size for a test of H_0: $R^2_F = R^2_R$ versus H_a: $R^2_F \neq R^2_R$ given R^2 of the reduced model of 0.10, the hypothesized R^2 of the full model of 0.15, 2 tested covariates, and 3 control covariates using default power 0.8 and significance level $\alpha = 0.05$

```
power rsquared 0.10 0.15, ntested(2) ncontrol(3)
```

Also use values of R^2_F equal to 0.11, 0.12, 0.13, 0.14, and 0.15, and display results in a table

```
power rsquared 0.10 (0.11(0.01)0.15), ntested(2) ncontrol(3)
```

Same as above, but display results in a graph of sample size versus R^2_F

```
power rsquared 0.10 (0.11(0.01)0.15), ntested(2) ncontrol(3) graph
```

Power for a sample size of 80

```
power rsquared 0.10 0.15, ntested(2) ncontrol(3) n(80)
```

Effect size and target R^2 for a sample size of 80 with power of 0.9

```
power rsquared 0.10, ntested(2) ncontrol(3) n(80) power(0.9)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

Test all coefficients

```
power rsquared R2T [ , power(numlist) options ]
```

Test a subset of coefficients

```
power rsquared R2R R2F , ncontrol(numlist) [power(numlist) options ]
```

Compute power

Test all coefficients

```
power rsquared R2T, n(numlist) [options ]
```

Test a subset of coefficients

```
power rsquared R2R R2F , ncontrol(numlist) n(numlist) [options ]
```

Compute effect size and target R^2

Test all coefficients

```
power rsquared, n(numlist) power(numlist) [options ]
```

Test a subset of coefficients

```
power rsquared R2R , ncontrol(numlist) n(numlist) power(numlist) [options ]
```

where R^2_T is the hypothesized R^2 of the tested model under the alternative hypothesis when testing all coefficients in the model, R^2_R is the R^2 of the reduced model, and R^2_F is the hypothesized R^2 of the full model when testing a subset of coefficients in the model.

R^2_T, R^2_R, and R^2_F may each be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

options	Description
Main	
* `alpha(`*numlist*`)`	significance level; default is `alpha(0.05)`
* `power(`*numlist*`)`	power; default is `power(0.8)`
* `beta(`*numlist*`)`	probability of type II error; default is `beta(0.2)`
* `n(`*numlist*`)`	sample size; required to compute power or effect size
`nfractional`	allow fractional sample size
* `ntested(`*numlist*`)`	number of tested covariates
* `ncontrol(`*numlist*`)`	number of control covariates; required for testing a subset of coefficients
* `diff(`*numlist*`)`	difference between the R^2 of the full and the reduced model, $R^2_F - R^2_R$; specify instead of the R^2 of the full model, R^2_F, when testing a subset of coefficients
`parallel`	treat number lists in starred options as parallel when multiple values per option are specified (do not enumerate all possible combinations of values)
Table	
[`no`]`table`[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
`saving(`*filename* [`, replace`]`)`	save the table data to *filename*; use `replace` to overwrite existing *filename*
Graph	
`graph`[(*graphopts*)]	graph results; see [PSS-2] **power, graph**
Iteration	
`init(`#`)`	initial value for sample size or R^2 of tested model in the case of testing all coefficients and R^2 difference in the case of testing a subset of coefficients
`iterate(`#`)`	maximum number of iterations; default is `iterate(500)`
`tolerance(`#`)`	parameter tolerance; default is `tolerance(1e-12)`
`ftolerance(`#`)`	function tolerance; default is `ftolerance(1e-12)`
[`no`]`log`	suppress or display iteration log
[`no`]`dots`	suppress or display iterations as dots
`notitle`	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the `parallel` option.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`notitle` does not appear in the dialog box.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`N`	number of subjects	N
`delta`	effect size	δ
`R2_T`	R^2 of the tested model	R^2_T
`R2_R`	R^2 of the reduced model	R^2_R
`R2_F`	R^2 of the full model	R^2_F
`R2_diff`	difference of R^2 between full and reduced models	R^2_D
`ntested`	number of tested covariates	N_T
`ncontrol`	number of control covariates	N_C
`target`	target parameter; synonym for `R2_T` or `R2_diff`	
`_all`	display all supported columns	

Column `beta` is shown in the default table in place of column `power` if specified.

Column `R2_T` is shown in the default table for a test of all coefficients and is not available if `ncontrol()` is specified.

Columns `R2_R`, `R2_F`, `R2_diff`, and `ncontrol` are shown in the default table for a test of a subset of coefficients and only available if `ncontrol()` is specified.

For a test of all coefficients, `target` is `R2_T`. For a test of a subset of coefficients, `target` is `R2_diff`.

Options

Main

`alpha()`, `power()`, `beta()`, `n()`, `nfractional`; see [PSS-2] **power**. The `nfractional` option is allowed only for sample-size determination.

`ntested(`*numlist*`)` specifies the number of tested covariates. The default is `ntested(1)`.

`ncontrol(`*numlist*`)` specifies the number of control covariates or the number of the covariates in the reduced model. This option is required for testing a subset of coefficients. If the option is not specified, all coefficients are assumed to be tested.

`diff(`*numlist*`)` specifies the difference between the R^2 of the full and reduced models, $R^2_F - R^2_R$, when computing sample size or power for a test of a subset of coefficients. You may specify either the R^2 of the full model, R^2_F, as a command argument or the difference $R^2_F - R^2_R$ in the `diff()` option. This option is not allowed with effect-size computation.

`parallel`; see [PSS-2] **power**.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies the initial value of the sample size for the sample-size determination or the initial value of the R^2 of the tested model in the case of testing all coefficients and the difference between the R^2 of the full and reduced models in the case of testing a subset of coefficients for the effect-size determination. The default is to use a bisection search method to compute an initial value.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following option is available with `power rsquared` but is not shown in the dialog box:

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

`power rsquared` computes sample size, power, and the target R^2 for a multiple linear regression R^2 test. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis, and see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Introduction

In contrast to a simple linear regression, a multiple regression framework allows researchers to control for additional variables that may better predict or explain the variation in the dependent variable of interest. The fit of the model, as measured by the R^2 statistic, sheds light on the efficacy of the multiple regression model in explaining the variation of the dependent variable.

Several scenarios arise for testing the fit of a multiple regression model. Consider an example where a researcher is interested in the effect of gender and education on wage,

$$y_{\text{wage}} = \beta_0 + \beta_{\text{edu}} x_{\text{edu}} + \beta_{\text{gender}} x_{\text{gender}} + \varepsilon \tag{1}$$

where the error term ε is independently and normally distributed with mean zero and constant standard deviation σ.

An F test may be performed for testing the joint significance of the coefficients on education and gender. The null hypothesis may be stated as H_0: $\beta_{\text{educ}} = \beta_{\text{gender}} = 0$; x_{edu} and x_{gender} are the tested covariates. This is equivalent to the null hypothesis H_0: $R^2_T = 0$, where R^2_T is the coefficient of determination, or alternatively, the variation of the dependent variable explained by the tested model.

Alternatively, a researcher may be interested in whether experience x_{exp} and location x_{loc} add further information in explaining wage variation after controlling for x_{edu} and x_{gender}:

$$y_{\text{wage}} = \beta_0 + \beta_{\text{exp}} x_{\text{exp}} + \beta_{\text{loc}} x_{\text{loc}} + \beta_{\text{edu}} x_{\text{edu}} + \beta_{\text{gender}} x_{\text{gender}} + \varepsilon \tag{2}$$

An F test may also be performed for testing the joint significance of the coefficients on experience and location. The null hypothesis may be stated as H_0: $\beta_{\text{exp}} = \beta_{\text{loc}} = 0$; x_{exp} and x_{loc} are now the tested covariates, and x_{edu} and x_{gender} are the control covariates. An equivalent test can be constructed based on the R^2. The null hypothesis is then H_0: $R^2_F = R^2_R$, where R^2_F is the R^2 of the full model (2) and R^2_R is the R^2 of the reduced model (1).

The `power rsquared` command provides power and sample-size analysis for the test of R^2 using an F test. For power analysis for a partial-correlation test in a multiple linear regression, see [PSS-2] **power pcorr**. For power analysis for a slope test in a simple linear regression, see [PSS-2] **power oneslope**.

Using power rsquared

`power rsquared` computes sample size, power, or target R^2 for an R^2 test in a multiple linear regression. By default, all computations are performed at the significance level of 0.05. You may change the significance level by specifying the `alpha()` option.

By default, the number of tested covariates is set to 1. You can change the number of tested covariates with the `ntested()` option. All computations assume that the model includes a constant. To test a subset of coefficients, you must also specify the `ncontrol()` option.

To compute sample size for testing all coefficients in the model, you must specify the R^2 of the tested model, R^2_T. For testing a subset of coefficients, you must specify the R^2 of the reduced model, R^2_R, and the R^2 of the full model, R^2_F. For either test, you can specify the power of the test in the `power()` option. The default power is set to 0.8.

To compute power, you must specify the sample size in the `n()` option. To test all coefficients in the model, you must also specify R^2_T. To test a subset of coefficients, you must also specify R^2_R and R^2_F.

When computing sample size or power for a subset of coefficients, you can specify the difference between the R^2 of the full and reduced models in the `diff()` option instead of R^2_F.

To compute effect size, which is defined as the ratio of R^2 explained by the tested covariates to the variance explained by the model error, you must specify the sample size in the `n()` option and the power of the test in the `power()` option. For a test of all coefficients, `power rsquared` reports the effect size and R^2_T. For a test of a subset of coefficients, you must also specify R^2_R to obtain the effect size. For this test, `power rsquared` reports the effect size and the difference between the R^2 statistics of the full and reduced models.

By default, the computed sample size is rounded up. You can specify the `nfractional` option to see the corresponding fractional sample size; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. The `nfractional` option is allowed only for sample-size determination.

`power rsquared`'s computations of sample size and effect size require iteration because the denominator degrees of freedom of the noncentral F distribution depends on the sample size, and the noncentrality parameter depends on the sample size and effect size. The default initial values are obtained using a bisection search method. You can use the `init()` option to specify your own value. The initial value of the sample size must be greater than the number of parameters in the multiple regression model. See [PSS-2] **power** for the descriptions of other options that control the iteration procedure.

In the following sections, we describe the use of `power rsquared` accompanied by examples for computing sample size, power, and target R^2.

Computing sample size

To compute sample size for testing all coefficients in the model, you must specify the R^2 of the tested model, R^2_T. For testing a subset of coefficients, you must specify the R^2 of the reduced model, R^2_R, the R^2 of the full model, R^2_F, and the number of control covariates in `ncontrol()`. For either test, a default power of 0.8 is assumed if `power()` is not specified, and one tested covariate is assumed if `ntested()` is not specified.

▷ Example 1: Sample size for testing all coefficients

Consider an example from Cohen (1988, 424) where a psychologist investigates a selection procedure based on job candidates' demographic characteristics used to predict success in a sales position. The five variables are age, education, prior experience, verbal aptitude, and extraversion.

Previous studies found that the addition of all five variables accounted for 10% of the variance in the dependent variable, that is to say, an R^2 of 0.1. We are designing a new study and want to determine the required sample size for detecting this previously observed R^2 so that we achieve 80% power at a 5% significance level. We do this by typing

```
. power rsquared 0.1, ntested(5)
Performing iteration ...
Estimated sample size for multiple linear regression
F test for R2 testing all coefficients
H0: R2_T = 0  versus  Ha: R2_T != 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.1111
         R2_T =    0.1000
      ntested =         5
Estimated sample size:
            N =       122
```

We find that a sample of 122 subjects is required to detect an R^2 of 0.1 with 80% power using a 5%-level test. The effect size (`delta`) is calculated using the given R^2 of the model; see *Methods and formulas* for details.

As we mentioned in *Using power rsquared*, sample-size computation requires iteration. The iteration log is suppressed by default, but you can display it by specifying the `log` option.

◁

▷ Example 2: Sample size for testing a subset of coefficients

Continuing with example 1, suppose that data for three of the variables—age, education, and prior experience—are readily available to the investigator. However, data for the other two variables—verbal aptitude and extraversion—are costly to obtain.

From previous studies, age, education, and prior experience explain about 6% of the variance in the dependent variable. The decision to include verbal aptitude and extraversion is deemed important only if their addition explains an additional 4% of the variance in the dependent variable; together the five variables should explain about 10% of the variance in the dependent variable. We will construct a test-case study to see the minimum sample size required to detect a 4% change in the R^2. We compute the minimum sample size required with 80% power at a 5% significance level:

```
. power rsquared 0.06 0.1, ntested(2) ncontrol(3)
Performing iteration ...
Estimated sample size for multiple linear regression
F test for R2 testing subset of coefficients
H0: R2_F = R2_R  versus  Ha: R2_F != R2_R
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.0444
         R2_R =    0.0600
         R2_F =    0.1000
      R2_diff =    0.0400
     ncontrol =         3
      ntested =         2
Estimated sample size:
            N =       220
```

We find that a sample of 220 subjects is required to detect an increase in R^2 by 4% (`R2_diff`) after adding verbal aptitude and extraversion with 80% power using a 5% level test.

◁

▷ Example 3: Specifying difference between the R^2 of the full and reduced models

Instead of using the R^2 of the full model 0.1 as in example 2, we can specify the difference between the R^2 of the full and reduced models, 0.04, in the `diff()` option.

```
. power rsquared 0.06, ntested(2) ncontrol(3) diff(0.04)
Performing iteration ...
Estimated sample size for multiple linear regression
F test for R2 testing subset of coefficients
H0: R2_F = R2_R  versus  Ha: R2_F != R2_R
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.0444
         R2_R =    0.0600
         R2_F =    0.1000
      R2_diff =    0.0400
     ncontrol =         3
      ntested =         2
Estimated sample size:
            N =       220
```

We obtain the same results as in example 2.

◁

Computing power

To compute power, you must specify the sample size in the `n()` option. To test all coefficients in the model, you must also specify the R^2 of the tested model, R^2_T. To test a subset of coefficients, you must also specify the R^2 of the reduced, R^2_R, and full models, R^2_F, and the number of control covariates in `ncontrol()`. The number of tested covariates is assumed to be one if `ntested()` is not specified.

Example 4: Power for testing all coefficients in a multiple regression

Continuing with example 1, suppose that we are designing a new study and anticipate obtaining a sample of 100 subjects. To compute the power corresponding to this sample size given the study parameters from example 1, we specify the sample size of 100 in the `n()` option:

```
. power rsquared 0.1, ntested(5) n(100)

Estimated power for multiple linear regression
F test for R2 testing all coefficients
H0: R2_T = 0  versus  Ha: R2_T != 0

Study parameters:

        alpha =    0.0500
            N =       100
        delta =    0.1111
         R2_T =    0.1000
      ntested =         5

Estimated power:

        power =    0.7014
```

For the smaller sample size, we achieve a lower power of about 70%.

Example 5: Power for testing a subset of coefficients

Continuing with example 2, suppose that we are designing a new study and anticipate a sample of 200 subjects. To compute the power corresponding to this sample size given the study parameters from example 2, we specify the sample size of 200 in the `n()` option:

```
. power rsquared 0.06 0.1, ntested(2) ncontrol(3) n(200)

Estimated power for multiple linear regression
F test for R2 testing subset of coefficients
H0: R2_F = R2_R  versus  Ha: R2_F != R2_R

Study parameters:

        alpha =    0.0500
            N =       200
        delta =    0.0444
         R2_R =    0.0600
         R2_F =    0.1000
      R2_diff =    0.0400
     ncontrol =         3
      ntested =         2

Estimated power:

        power =    0.7583
```

With a smaller sample size, the power of the test decreases to about 76%.

Example 6: Multiple values of study parameters

Continuing with example 5, we want to investigate the effect of sample size on power, so we specify a list of sample sizes in the `n()` option:

```
. power rsquared 0.06 0.1, n(50 100 200 400 800) ntested(2) ncontrol(3)
Estimated power for multiple linear regression
F test for R2 testing subset of coefficients
H0: R2_F = R2_R  versus  Ha: R2_F != R2_R
```

alpha	power	N	delta	R2_R	R2_F	R2_diff	ntested	ncontrol
.05	.2328	50	.04444	.06	.1	.04	2	3
.05	.4431	100	.04444	.06	.1	.04	2	3
.05	.7583	200	.04444	.06	.1	.04	2	3
.05	.9719	400	.04444	.06	.1	.04	2	3
.05	.9999	800	.04444	.06	.1	.04	2	3

As expected, when the sample size increases, the power tends to get closer to 1.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

Computing effect size and target R^2

For a test of all coefficients, effect size δ is defined as the ratio of the R^2 of the tested model to the variance explained by the model error term, $\delta = R_T^2/(1 - R_T^2)$. For a test of a subset of coefficients, effect size δ is defined as the ratio of the difference between the R^2 of the full and reduced models to the variance explained by the model error term, $\delta = (R_F^2 - R_R^2)/(1 - R_F^2)$.

Sometimes, we may be interested in determining the minimum detectable effect and the corresponding target R^2 that yield a statistically significant result for a prespecified sample size and power. In this case, we must specify power and sample size. To test a subset of coefficients, we must also specify the R^2 of the reduced model and the number of control covariates.

▷ Example 7: Minimum detectable value for the R^2 of the tested model

Continuing with example 4, we may also be interested in finding the minimum value for the R^2 of the tested model that can be detected with a power of 80% given a sample of 100 subjects. To compute this, we specify the sample size of 100 in the `n()` option and power of 0.8 in the `power()` option. As in example 4, we use the same value of 5 for the number of tested covariates.

```
. power rsquared, n(100) power(0.8) ntested(5)
Performing iteration ...
Estimated R-squared for multiple linear regression
F test for R2 testing all coefficients
H0: R2_T = 0  versus  Ha: R2_T != 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
            N =       100
      ntested =         5
Estimated effect size and R-squared:
        delta =    0.1360
         R2_T =    0.1197
```

The minimum detectable value for the R^2 of the tested model is 0.1197, which corresponds to the effect size of 0.1360. Compared with example 4, we would detect a slightly larger value for the R^2 of the tested model.

◁

▷ Example 8: Minimum detectable value of the R^2 difference

Continuing with example 5, we may also be interested in finding the minimum difference between the R^2 of the full and reduced models that can be detected with a power of 80% given a sample of 200 subjects. To compute this, we specify the R^2 of the reduced model of 0.06 as the command argument and also specify the sample size of 200 in the `n()` option and a power of 0.8 in the `power()` option. As in example 5, we use 2 tested covariates and 3 control covariates.

```
. power rsquared 0.06, n(200) power(0.8) ntested(2) ncontrol(3)

Performing iteration ...

Estimated R-squared for multiple linear regression
F test for R2 testing subset of coefficients
H0: R2_F = R2_R  versus  Ha: R2_F != R2_R

Study parameters:

        alpha =    0.0500
        power =    0.8000
            N =       200
         R2_R =    0.0600
     ncontrol =         3
      ntested =         2

Estimated effect size and R-squared:

        delta =    0.0489
      R2_diff =    0.0438
         R2_F =    0.1038
```

The minimum detectable value for the R^2 difference is 0.0438, which corresponds to an effect size of 0.0489. In example 2, we assumed the same power and significance levels, and we found that we need 220 subjects to detect an R^2 difference of 0.04. With the smaller sample of 200, the minimum detectable R^2 difference is slightly larger.

◁

Performing hypothesis tests on the coefficients

Suppose we wish to test the hypothesis that, controlling for other variables, a group of variables has no effect in explaining the variance of the dependent variable. We can use the `regress` command to estimate the coefficients and the `test` command to perform a hypothesis test.

▷ Example 9: Joint test of the coefficients

Consider `auto.dta`, which contains various characteristics of 74 cars. Suppose that our study goal is to investigate whether headroom (`headroom`) and trunk space (`trunk`) have any effect on the price of cars after controlling for their weight (`weight`) and mileage (`mpg`). We can fit the full linear model and obtain the R^2 using `regress`.

```
. use https://www.stata-press.com/data/r19/auto
(1978 automobile data)
. regress price headroom trunk weight mpg
```

Source	SS	df	MS	Number of obs	=	74
				F(4, 69)	=	8.21
Model	204838391	4	51209597.9	Prob > F	=	0.0000
Residual	430227005	69	6235173.98	R-squared	=	0.3225
				Adj R-squared	=	0.2833
Total	635065396	73	8699525.97	Root MSE	=	2497

price	Coefficient	Std. err.	t	P>|t|	[95% conf.	interval]
headroom	-726.5434	462.0322	-1.57	0.120	-1648.272	195.1856
trunk	23.04248	108.3649	0.21	0.832	-193.1396	239.2246
weight	2.011936	.7036432	2.86	0.006	.6082062	3.415666
mpg	-54.79153	85.91635	-0.64	0.526	-226.19	116.6069
_cons	3114.94	3648.08	0.85	0.396	-4162.779	10392.66

The R^2 of the full model is around 0.32. We can test the joint hypothesis H_0: $\beta_{\text{headroom}} = \beta_{\text{trunk}} = 0$ using the `test` command.

```
. test headroom trunk
 ( 1)  headroom = 0
 ( 2)  trunk = 0
       F(  2,    69) =    1.48
            Prob > F =    0.2337
```

We fail to reject the null hypothesis at the 5% significance level; the p-value > 0.05.

Suppose we wish to design a new similar study. We use the estimates from this study to perform a sample-size analysis. First, we need to fit the reduced model to obtain the estimate of its R^2.

```
. regress price weight mpg
```

Source	SS	df	MS	Number of obs	=	74
				F(2, 71)	=	14.74
Model	186321280	2	93160639.9	Prob > F	=	0.0000
Residual	448744116	71	6320339.67	R-squared	=	0.2934
				Adj R-squared	=	0.2735
Total	635065396	73	8699525.97	Root MSE	=	2514

price	Coefficient	Std. err.	t	P>|t|	[95% conf.	interval]
weight	1.746559	.6413538	2.72	0.008	.467736	3.025382
mpg	-49.51222	86.15604	-0.57	0.567	-221.3025	122.278
_cons	1946.069	3597.05	0.54	0.590	-5226.245	9118.382

The R^2 of the reduced model is around 0.29. Next, we specify this number as well as the R^2 of the full model, 0.32, with `power rsquared` to perform a sample-size analysis.

```
. power rsquared 0.29 0.32, ntested(2) ncontrol(2)
Performing iteration ...
Estimated sample size for multiple linear regression
F test for R2 testing subset of coefficients
H0: R2_F = R2_R  versus  Ha: R2_F != R2_R
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.0441
         R2_R =    0.2900
         R2_F =    0.3200
      R2_diff =    0.0300
     ncontrol =         2
      ntested =         2
Estimated sample size:
            N =       222
```

We find that a sample size of 222 is required to detect an increase in R^2 by 0.03 with 80% power using a 5%-level two-sided test.

◁

Stored results

`power rsquared` stores the following in `r()`:

Scalars

`r(alpha)`	significance level
`r(power)`	power
`r(beta)`	probability of a type II error
`r(delta)`	effect size
`r(N)`	sample size
`r(nfractional)`	1 if `nfractional` is specified, 0 otherwise
`r(R2_T)`	R^2 of the tested model
`r(R2_R)`	R^2 of the reduced model
`r(R2_F)`	R^2 of the full model
`r(R2_diff)`	difference between R^2 of the full and reduced models
`r(ntested)`	number of tested covariates
`r(ncontrol)`	number of control covariates
`r(separator)`	number of lines between separator lines in the table
`r(divider)`	1 if `divider` is requested in the table, 0 otherwise
`r(init)`	initial value for sample size or for R^2
`r(maxiter)`	maximum number of iterations
`r(iter)`	number of iterations performed
`r(tolerance)`	requested parameter tolerance
`r(deltax)`	final parameter tolerance achieved
`r(ftolerance)`	requested distance of the objective function from zero
`r(function)`	final distance of the objective function from zero
`r(converged)`	1 if iteration algorithm converged, 0 otherwise

Macros

`r(type)`	`test`
`r(method)`	`rsquared`
`r(columns)`	displayed table columns
`r(labels)`	table column labels
`r(widths)`	table column widths
`r(formats)`	table column formats

Matrices

`r(pss_table)`	table of results

Methods and formulas

Methods and formulas are presented under the following headings:

Introduction

This section subsumes the *Methods and formulas* for [PSS-2] **power pcorr**.

Consider a multiple linear regression model of a dependent variable y_i on $k+p$ fixed covariates x_i in a sample of n subjects,

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \cdots + \beta_k x_{ki} + \beta_{k+1} x_{(k+1)i} + \cdots + \beta_{k+p} x_{(k+p)i} + \varepsilon_i$$

where ε_i's are independently and normally distributed with mean zero and constant standard deviation σ. Suppose that we are interested in testing the significance of the coefficients in the model and we construct an F statistic using the effect size δ; see subsections below for the definition of the effect size specific to each test. Under the alternative hypothesis, the F statistic

$$F = \frac{\delta/\nu_1}{1/\nu_2}$$

follows a noncentral F distribution with noncentrality parameter $\lambda = n\delta$ and ν_1 numerator and ν_2 denominator degrees of freedom. The formulas of the F test are based on Cohen (1988).

Let α be the significance level, β be the probability of a type II error, and $F_{\nu_1,\nu_2,\alpha}$ denote the αth quantile of an F distribution with ν_1 numerator and ν_2 denominator degrees of freedom.

The power $\pi = 1-\beta$ is computed using

$$\pi = 1 - F_{\nu_1,\nu_2,\lambda}\left(F_{\nu_1,\nu_2,1-\alpha}\right) \tag{3}$$

where $F_{\nu_1,\nu_2,\lambda}(\cdot)$ is the cumulative noncentral F distribution with a noncentrality parameter λ.

Sample size and effect size are obtained by iteratively solving the nonlinear equations in (3) for n and δ. If the `nfractional` option is not specified, the computed sample size is rounded up.

Details of the effect size δ and the F statistics are given in the following sections.

Testing all coefficients

Consider first the hypothesis that all the coefficients are zero, H_0: $\beta_1 = \beta_2 = \cdots = \beta_{k+p} = 0$. Let R^2_T denote the proportion of the variance of y explained by all the $k+p$ covariates. The effect size is defined as $\delta = R^2_T/(1-R^2_T)$. The test statistic is then given by

$$F = \frac{\delta/\nu_1}{1/\nu_2} = \frac{R^2_T/\nu_1}{(1-R^2_T)/\nu_2}$$

where $\nu_1 = k+p$ and $\nu_2 = n-k-p-1$.

Testing a subset of coefficients: R^2 of full versus reduced models

Now suppose we are interested in testing whether the subset of p covariates accounts for any variation in the dependent variable after adjusting for the effects of the k covariates. The null hypothesis may be stated as H_0: $\beta_{k+1} = \cdots = \beta_{k+p} = 0$. Let R^2_F denote the R^2 of the full model with $k+p$ covariates and R^2_R denote the R^2 of the reduced model with k covariates. With effect size $\delta = (R^2_F - R^2_R)/(1 - R^2_F)$, the test statistic is given by

$$F = \frac{\delta/\nu_1}{1/\nu_2} = \frac{(R^2_F - R^2_R)/\nu_1}{(1 - R^2_F)/\nu_2} \tag{4}$$

where $\nu_1 = p$ and $\nu_2 = n - k - p - 1$.

Testing a subset of coefficients: Partial multiple correlation

The F test for testing the null hypothesis H_0: $\beta_{k+1} = \cdots = \beta_{k+p} = 0$ can also be constructed using partial multiple correlation. Let $\rho^2_p = (R^2_F - R^2_R)/(1 - R^2_R)$ denote the square of the partial multiple correlation. With effect size $\delta = \rho^2_p/(1 - \rho^2_p)$, the F statistic is given by

$$F = \frac{\delta/\nu_1}{1/\nu_2} = \frac{\rho^2_p/\nu_1}{(1 - \rho^2_p)/\nu_2}$$

where $\nu_1 = p$ and $\nu_2 = n - k - p - 1$. This is the same test statistic as in (4) for testing a subset of coefficients using R^2.

Reference

Cohen, J. 1988. *Statistical Power Analysis for the Behavioral Sciences.* 2nd ed. Hillsdale, NJ: Erlbaum.

Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power oneslope** — Power analysis for a slope test in a simple linear regression

[PSS-2] **power pcorr** — Power analysis for a partial-correlation test in a multiple linear regression

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[R] **regress** — Linear regression

[R] **test** — Test linear hypotheses after estimation

power pcorr — Power analysis for a partial-correlation test in a multiple linear regression

Description

`power pcorr` computes sample size, power, or target squared partial correlation for a partial-correlation test in a multiple linear regression. A partial-correlation test is an F test of the squared partial multiple correlation that is used to test the significance of a subset of coefficients in a regression model. By default, `power pcorr` computes sample size given power and the squared partial correlation. Alternatively, it computes power given sample size and the squared partial correlation, or it computes the squared partial correlation given sample size and power. See [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Quick start

Sample size for a test of H_0: $\rho_p^2 = 0$ versus H_a: $\rho_p^2 \neq 0$ given squared partial correlation of 0.1, 3 tested covariates, and 5 control covariates using default power of 0.8 and significance level $\alpha = 0.05$

```
power pcorr 0.1, ntested(3) ncontrol(5)
```

Power for sample size of 100

```
power pcorr 0.1, ntested(3) ncontrol(5) n(100)
```

Same as above, but for sample sizes of 45, 60, 75, and 90

```
power pcorr 0.1, ntested(3) ncontrol(5) n(45(15)90)
```

Same as above, but display results in a graph of power versus sample size

```
power pcorr 0.1, ntested(3) ncontrol(5) n(45(15)90) graph
```

Effect size and target squared partial correlation for sample size of 100 with power of 0.8

```
power pcorr, ntested(3) ncontrol(5) n(100) power(0.8)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
power pcorr rho2_p [ , power(numlist) options ]
```

Compute power

```
power pcorr rho2_p , n(numlist) [ options ]
```

Compute effect size and target squared partial correlation

```
power pcorr , n(numlist) power(numlist) [ options ]
```

where *rho2_p* is the hypothesized squared partial correlation in a multiple linear regression. *rho2_p* may be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

options	Description
Main	
* alpha(*numlist*)	significance level; default is alpha(0.05)
* power(*numlist*)	power; default is power(0.8)
* beta(*numlist*)	probability of type II error; default is beta(0.2)
* n(*numlist*)	sample size; required to compute power or effect size
nfractional	allow fractional sample size
* ntested(*numlist*)	number of tested covariates; default is ntested(1)
* ncontrol(*numlist*)	number of control covariates; default is ncontrol(1)
parallel	treat number lists in starred options as parallel when multiple values per option are specified (do not enumerate all possible combinations of values)
Table	
[no]table[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
saving(*filename* [, replace])	save the table data to *filename*; use replace to overwrite existing *filename*
Graph	
graph[(*graphopts*)]	graph results; see [PSS-2] **power, graph**
Iteration	
init(#)	initial value for sample size or squared partial correlation
iterate(#)	maximum number of iterations; default is iterate(500)
tolerance(#)	parameter tolerance; default is tolerance(1e-12)
ftolerance(#)	function tolerance; default is ftolerance(1e-12)
[no]log	suppress or display iteration log
[no]dots	suppress or display iterations as dots
notitle	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the parallel option.

collect is allowed; see **[U] 11.1.10 Prefix commands**.

notitle does not appear in the dialog box.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
alpha	significance level	α
power	power	$1-\beta$
beta	type-II-error probability	β
N	number of subjects	N
delta	effect size	δ
rho2_p	squared partial multiple correlation	ρ_p^2
ntested	number of tested covariates	N_T
ncontrol	number of control covariates	N_C
target	target parameter; synonym for rho2_p	
_all	display all supported columns	

Column beta is shown in the default table in place of column power if specified.

Options

Main

alpha(), power(), beta(), n(), nfractional; see [PSS-2] **power**. The nfractional option is allowed only for sample-size determination.

ntested(*numlist*) specifies the number of tested covariates. The default is ntested(1).

ncontrol(*numlist*) specifies the number of control covariates or the number of covariates in the reduced model. The default is ncontrol(1).

parallel; see [PSS-2] **power**.

Table

table, table(), notable; see [PSS-2] **power, table**.

saving(); see [PSS-2] **power**.

Graph

graph, graph(); see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

init(#) specifies the initial value of the sample size for the sample-size determination or the initial value of the squared partial correlation for the effect-size determination. The default is to use a bisection search method to compute an initial value.

iterate(), tolerance(), ftolerance(), log, nolog, dots, nodots; see [PSS-2] **power**.

The following option is available with power pcorr but is not shown in the dialog box:

notitle; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

`power pcorr` computes sample size, power, and the target squared partial correlation for a multiple regression partial-correlation test. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis, and see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Introduction

In contrast to a simple linear regression, a multiple regression framework allows researchers to control for additional predictors that may add information to better predict or explain the variation in the dependent variable of interest. There are several scenarios in which we may be interested in knowing whether additional predictors improve the model. Consider an example where a researcher is interested in whether variables such as experience, x_{exp}, and location, x_{loc}, add additional information in explaining variation in wage, y_{wage}, after controlling for education, x_{edu}, and gender, x_{gender},

$$y_{\text{wage}} = \beta_0 + \beta_{\text{exp}}x_{\text{exp}} + \beta_{\text{loc}}x_{\text{loc}} + \beta_{\text{edu}}x_{\text{edu}} + \beta_{\text{gender}}x_{\text{gender}} + \varepsilon$$

where ε is an independently and normally distributed error term with mean zero and constant standard deviation σ. One way to test whether experience and location add additional information is to test the joint significance of the coefficients on experience, β_{exp}, and location, β_{loc}. In this case, we perform an F test using the null H_0: $\beta_{\text{exp}} = \beta_{\text{loc}} = 0$. x_{exp} and x_{loc} are the tested covariates; x_{edu} and x_{gender} are the control covariates.

An equivalent test, what we refer to as a partial-correlation test, can be constructed based on the squared partial correlation, ρ_p^2. The null hypothesis is H_0: $\rho_p^2 = 0$ and, as in the joint test of coefficients, can be tested with an F test. ρ_p^2 is a function of the coefficient of determination, R^2, which is a measure of the variation of the dependent variable explained by the model that is used to assess overall model fit for a linear regression. Specifically, $\rho_p^2 = (R_F^2 - R_R^2)/(1 - R_R^2)$. R_F^2 is the R^2 of the full model that includes the tested and control covariates, and R_R^2 is the R^2 of the reduced model that includes only the control covariates.

The `power pcorr` command provides power and sample-size analysis for a partial-correlation test. For power analysis for an R^2 test in a multiple linear regression, see [PSS-2] **power rsquared**. For power analysis for a slope test in a simple linear regression, see [PSS-2] **power oneslope**.

Using power pcorr

`power pcorr` computes sample size, power, or the target squared partial correlation *rho2_p* for a partial-correlation test in a multiple linear regression. By default, all computations are performed at the significance level of 0.05. You may change the significance level by specifying the `alpha()` option.

By default, the numbers of tested covariates and of control covariates are set to 1. You may change the respective values with the `ntested()` and `ncontrol()` options.

To compute sample size, you must specify the squared partial correlation *rho2_p* and, optionally, the power of the test in the `power()` option. The default power is set to 0.8.

To compute power, you must specify the sample size in the `n()` option and the squared partial correlation *rho2_p*.

To compute the target partial correlation and effect size, which is defined in terms of the partial correlation as $\delta = \rho_p^2/(1-\rho_p^2)$, you must specify the sample size in the `n()` option and the power in the `power()` option.

By default, the computed sample size is rounded up. You can specify the `nfractional` option to see the corresponding fractional sample size; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. The `nfractional` option is allowed only for sample-size determination.

`power pcorr`'s computations of sample size and effect size require iteration because the denominator degrees of freedom of the noncentral F distribution depends on the sample size, and the noncentrality parameter depends on the sample size and effect size. The default initial values are obtained using a bisection search method. You may use the `init()` option to specify your own value. The initial value of the sample size must be greater than the number of parameters in the multiple regression model. See [PSS-2] **power** for the descriptions of other options that control the iteration procedure.

Computing sample size

To compute sample size, you must specify the squared partial correlation *rho2_p* and, optionally, the power of the test in the `power()` option. A default power of 0.8 is assumed if `power()` is not specified.

▷ Example 1: Sample size for a partial-correlation test

Consider an example from Cohen (1988, 436) where a psychologist investigates a selection procedure based on job candidates' demographic characteristics used to predict success in a sales position. Suppose we want to conduct a similar study. Data for age, education, and prior experience, our control covariates, are readily available. However, data on verbal aptitude and extraversion, the tested covariates, are costly to obtain, and we decide that these variables are worth including only if the squared partial correlation with the dependent variable is at least 0.0426.

We will conduct a test-case study to determine the minimum sample size required to detect a squared partial correlation of 0.0426 between the dependent variable and the two extra variables, verbal aptitude and extraversion. We compute the minimum sample size required with 80% power at a 5% significance level:

```
. power pcorr 0.0426, ntested(2) ncontrol(3)
Performing iteration ...
Estimated sample size for multiple linear regression
F test for partial correlation
H0: rho2_p = 0  versus  Ha: rho2_p != 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.0445
       rho2_p =    0.0426
     ncontrol =         3
      ntested =         2
Estimated sample size:
            N =       220
```

We find that a sample of 220 subjects is required to detect a squared partial correlation of 0.0426 associated with the two extra variables, verbal aptitude and extraversion, with 80% power using a 5% level test. The effect size `delta` is calculated using the given information about the hypothesized squared partial correlation; see *Methods and formulas* in [PSS-2] **power rsquared** for details.

As we mentioned in *Using power pcorr*, sample-size computation requires iteration. The iteration log is suppressed by default, but you can display it by specifying the `log` option.

Computing power

To compute power, you must specify the sample size in the `n()` option and the squared partial correlation *rho2_p*.

▷ Example 2: Power for a partial-correlation test

Continuing with example 1, suppose that we are designing a new study and anticipate a sample of 200 subjects. Given the study parameters from example 1, we compute the power by specifying the sample size of 200 in the `n()` option:

```
. power pcorr 0.0426, ntested(2) ncontrol(3) n(200)
Estimated power for multiple linear regression
F test for partial correlation
H0: rho2_p = 0  versus  Ha: rho2_p != 0
Study parameters:
        alpha =    0.0500
            N =       200
        delta =    0.0445
       rho2_p =    0.0426
     ncontrol =         3
      ntested =         2
Estimated power:
        power =    0.7588
```

With this smaller sample size, the power of the test decreases to about 76%.

▷ Example 3: Multiple values of study parameters

Continuing with example 2, we want to see the effect of sample size on power. We specify a list of sample sizes in the `n()` option:

```
. power pcorr 0.0426, n(50 100 200 400 800) ntested(2) ncontrol(3)
Estimated power for multiple linear regression
F test for partial correlation
H0: rho2_p = 0  versus  Ha: rho2_p != 0
```

alpha	power	N	delta	rho2_p	ntested	ncontrol
.05	.2331	50	.0445	.0426	2	3
.05	.4435	100	.0445	.0426	2	3
.05	.7588	200	.0445	.0426	2	3
.05	.972	400	.0445	.0426	2	3
.05	.9999	800	.0445	.0426	2	3

As expected, when the sample size increases, the power tends to get closer to 1.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

Computing effect size and target squared partial correlation

Effect size δ for a multiple regression, defined in terms of the partial correlation, is $\delta = \rho_p^2/(1-\rho_p^2)$. To compute the effect size and target squared partial correlation, you must specify the sample size in the `n()` option and the power in the `power()` option.

▷ Example 4: Minimum detectable squared partial correlation

Continuing with example 2, we may also be interested in finding the minimum value of the squared partial correlation that can be detected with a sample of 200 subjects and 80% power. To compute this, we specify `n(200)` and `power(0.8)`. As before, we use 2 tested covariates and 3 control covariates.

```
. power pcorr, n(200) power(0.8) ntested(2) ncontrol(3)
Performing iteration ...
Estimated squared partial correlation for multiple linear regression
F test for partial correlation
H0: rho2_p = 0  versus  Ha: rho2_p != 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
            N =       200
     ncontrol =         3
      ntested =         2
Estimated effect size and squared partial correlation:
        delta =    0.0489
       rho2_p =    0.0466
```

The minimum detectable squared partial correlation is 0.0466, which corresponds to an effect size of 0.0489. These values are slightly larger than the values of 0.0426 and 0.0445 that can be detected with the larger sample of 220 subjects from example 1 for the same 80% power.

◁

Performing hypothesis tests on the partial correlation

In this section, we briefly demonstrate the use of the pcorr command for estimating partial correlations.

▷ Example 5: Partial-correlation test for one variable

Suppose that our study goal is to investigate whether the mileage of a car (mpg) has an effect on its price (price) after controlling for headroom (headroom) and trunk space (trunk). We compute the partial correlation by using pcorr on data about cars from 1978 in auto.dta. This preliminary investigation will tell us how many modern cars we should select for our study.

```
. use https://www.stata-press.com/data/r19/auto
(1978 automobile data)

. pcorr price mpg headroom trunk
(obs=74)

Partial and semipartial correlations of price with
```

Variable	Partial corr.	Semipartial corr.	Partial corr.^2	Semipartial corr.^2	Significance value
mpg	-0.3800	-0.3576	0.1444	0.1279	0.0010
headroom	-0.1606	-0.1416	0.0258	0.0201	0.1779
trunk	0.1397	0.1228	0.0195	0.0151	0.2418

We obtain a squared partial correlation of around 0.14 for mpg. The significance value is less than 0.05.

Suppose we wish to design a new similar study. We use the estimated squared partial correlation from this study to perform a sample-size analysis. pcorr computes the partial correlation for only a single variable, not a group of variables, so our power calculation here uses only one tested covariate; see [R] **pcorr** for details. We use the default value of 1 for ntested() and specify 2 control covariates.

```
. power pcorr 0.14, ntested(1) ncontrol(2)

Performing iteration ...

Estimated sample size for multiple linear regression
F test for partial correlation
H0: rho2_p = 0  versus  Ha: rho2_p != 0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.1628
       rho2_p =    0.1400
     ncontrol =         2
      ntested =         1

Estimated sample size:

            N =        51
```

We find that a sample size of 51 is required to detect a squared partial correlation of 0.14 with 80% power using a 5%-level test.

◁

Stored results

power pcorr stores the following in r():

Scalars

r(alpha)	significance level
r(power)	power
r(beta)	probability of a type II error
r(delta)	effect size
r(N)	sample size
r(nfractional)	1 if nfractional is specified, 0 otherwise
r(rho2_p)	squared partial correlation
r(ntested)	number of tested covariates
r(ncontrol)	number of control covariates
r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise
r(init)	initial value for sample size or for squared partial correlation
r(maxiter)	maximum number of iterations
r(iter)	number of iterations performed
r(tolerance)	requested parameter tolerance
r(deltax)	final parameter tolerance achieved
r(ftolerance)	requested distance of the objective function from zero
r(function)	final distance of the objective function from zero
r(converged)	1 if iteration algorithm converged, 0 otherwise

Macros

r(type)	test
r(method)	pcorr
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results

Methods and formulas

See *Testing a subset of coefficients: Partial multiple correlation* under *Methods and formulas* in [PSS-2] **power rsquared**.

Reference

Cohen, J. 1988. *Statistical Power Analysis for the Behavioral Sciences*. 2nd ed. Hillsdale, NJ: Erlbaum.

Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power oneslope** — Power analysis for a slope test in a simple linear regression

[PSS-2] **power rsquared** — Power analysis for an R^2 test in a multiple linear regression

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[R] **pcorr** — Partial and semipartial correlation coefficients

power logistic — Power analysis for logistic regression[+]

[+]This command is part of StataNow.

Description Menu Syntax Remarks and examples Reference Also see

Description

`power logistic` computes sample size, power, or effect size for a test of one coefficient in logistic regression. There are three main uses of `power logistic`, each of which is described in its own manual entry. See [PSS-2] **power logistic onebin** for when the logistic regression has only one binary covariate. See [PSS-2] **power logistic twobin** for when the logistic regression has two binary covariates, one of which is the covariate of interest and the other is a nuisance covariate. See [PSS-2] **power logistic general** for a description of the general case that can accommodate covariates from any of 11 distributions while allowing for up to 20 nuisance covariates.

By default, `power logistic` computes sample size for a given power and effect size, where the effect size may be specified as a coefficient or an odds ratio. Alternatively, it can compute power given sample size and effect size, or it can compute the effect size given power and sample size.

Menu

Statistics > Power, precision, and sample size

Syntax

Sample-size computation with one binary covariate

```
power logistic oratio_X , { px(numlist) | oddsx(numlist) } [ power(numlist) onebinopts ]
```

Sample-size computation with two binary covariates

```
power logistic oratio_X , { px(numlist) | oddsx(numlist) } { pz(numlist) | oddsz(numlist) }
                          [ power(numlist) twobinopts ]
```

Sample-size computation with arbitrary covariates

```
power logistic oratio_X , x(xzspec) [ z1(xzspec) [ z2(xzspec) [ ... ] ]
                                      power(numlist) generalopts ]
```

oratio$_X$ is the odds ratio for covariate of interest X under the alternative hypothesis H_a. Argument *oratio*$_X$ may be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**). Power and effect-size calculations follow similar syntax; see [PSS-2] **power logistic onebin**, [PSS-2] **power logistic twobin**, and [PSS-2] **power logistic general** for details.

Remarks and examples

Remarks are presented under the following headings:

This entry provides an overview of the `power logistic` command and demonstrates how sample-size calculations can be specified for logistic regression with one or two binary covariates as well as the general syntax for logistic regression with arbitrary covariates. For further details and information about performing power and effect-size calculations, see the individual entries [PSS-2] **power logistic onebin**, [PSS-2] **power logistic twobin**, and [PSS-2] **power logistic general**. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis, and see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Introduction to power logistic

Logistic regression is a commonly used statistical method for analyzing binary outcome variables. Researchers often need to determine the appropriate sample size to ensure sufficient power for detecting an association between a binary outcome variable and a covariate of interest, potentially in the presence of nuisance covariates.

For example, consider a study that examines factors influencing whether migratory birds return to the same nesting site from one year to the next. Binary outcome Y is an indicator of whether the nesting site was reused, where the observed $y_i = 1$ if bird i returns to the nesting site it used last year and $y_i = 0$ if bird i does not. We will use logistic regression to test whether birds of one sex are more likely to return to the same nesting site than birds of the other sex. We define binary covariate of interest X as an indicator for female birds, where the observed $x_i = 1$ if bird i is female and $x_i = 0$ if bird i is male. Previous observations suggest that birds who mate with the same partner as last year are more likely to return to the same nesting site, as are heavier birds. We are not interested in studying the effect of a bird's mate or weight, but it would be foolhardy to ignore these effects. We include mate in our logistic regression as nuisance covariate Z_1, where the observed $z_{1i} = 1$ if bird i has the same mate as last year and $z_{1i} = 0$ if not. And we include weight as nuisance covariate Z_2, where the observed z_{2i} is the weight of bird i. Taken together, we write $\mathbf{Z} = (Z_1, Z_2)$. The logistic regression can be written as

$$\Pr(y_i = 1|x_i, \mathbf{z}_i) = H(\beta_X x_i + \zeta_0 + \zeta_1 z_{1i} + \zeta_2 z_{2i}) \qquad i = 1, 2, \ldots, n$$

where β_X is the coefficient quantifying the effect of a bird's sex on nesting-site reuse, ζ_0 is the logistic intercept, ζ_1 is the effect of partnering with the same mate, ζ_2 is the effect of the bird's weight, and n is the sample size. Function $H(\eta) = \{1 + \exp(-\eta)\}^{-1}$ is the logistic distribution function.

The effect of covariate X can also be expressed in terms of an odds ratio,

$$\text{OR}_X = \exp(\beta_X \text{U}_X) = \Pr(Y = 1|X = 1)\Pr(Y = 0|X = 0)/\{\Pr(Y = 0|X = 1)\Pr(Y = 1|X = 0)\}$$

where U_X is the unit change in X for the odds ratio and $\text{U}_X > 0$. In this example, U_X must equal 1 because X is a Bernoulli random variable. The null hypothesis is $H_0 : \beta_X = 0$, which can also be expressed as $H_0 : \text{OR}_X = 1$. The alternative hypothesis is $H_a : \beta_X \neq 0$, or equivalently, $H_a : \text{OR}_X \neq 1$.

The `power logistic` command provides power and sample-size analysis for the test of $\beta_X = 0$ in logistic regression. The formula for power, sample-size, and effect-size calculations is based on the likelihood-ratio test of $\beta_X = 0$. However, Bush (2015) demonstrates that sample-size requirements for Wald and score tests are generally equivalent to the sample-size requirement for the likelihood-ratio test.

Thus, these calculations may be used to plan studies that will be analyzed using the `logit` command, which conducts a Wald test of $\beta_X = 0$, or the `logistic` command, which conducts an equivalent Wald test of $\text{OR}_X = 1$. If you prefer a likelihood-ratio test, you can use the `lrtest` command.

Computing sample size

In the following examples, we perform sample-size calculations for the study of migratory birds. To compute sample size, you must specify the effect size and other parameters of the logistic regression, and, optionally, the power of the test using either the `power()` or the `beta()` option. A default power of 0.8 is assumed if `power()` is not specified. For further details and information about performing power and effect-size calculations, see the individual entries [PSS-2] **power logistic onebin**, [PSS-2] **power logistic twobin**, and [PSS-2] **power logistic general**.

▷ Example 1: Sample size with one binary covariate

To demonstrate, suppose we ignored the nuisance covariates in the study of migratory birds, leaving only the covariate of interest X. With a single binary covariate, we can use the one binary syntax of `power logistic`. We wish to compute the sample size required to detect an odds ratio of 1.5 for the effect of X on Y. We specify the effect size, $\text{OR}_X = 1.5$, as an argument. Female birds compose approximately 57% of our target population, which we specify in the `px()` option. A previous study of male birds found a 20% rate of nest-site reuse, so we specify $\Pr(Y = 1|X = 0) = 0.2$ using the `pycondx0()` option. Below, we calculate the sample size required to achieve 80% power with a 5% level test, using the default values of `alpha` and `power`.

```
. power logistic 1.5, px(0.57) pycondx0(0.2)

Estimated sample size for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    1.5000  (odds ratio)
      oratiox =    1.5000
           px =    0.5700
    pycondx0 =    0.2000

Estimated sample size:

            N =     1,092
```

When `power logistic` is called, the output lists the study parameters we specified, followed by the estimated sample size. A sample of 1,092 birds is required to detect an odds ratio of 1.5 with 80% power using a 5% level test.

The same sample-size calculation can be specified using the general syntax of `power logistic`. We still specify the effect size as an argument, but now we explicitly specify the distribution of X in the `x()` option. Covariate X follows a Bernoulli distribution with parameter $p_X = 0.57$. Instead of specifying $\Pr(Y = 1|X = 0)$ in the `pycondx0()` option, we specify $\Pr\{Y = 1|X = 0, \mathbf{Z} = E(\mathbf{Z})\}$ in the `pycondx0zm()` option. This logistic regression does not contain nuisance covariates, so in this case, $\Pr(Y = 1|X = 0) = \Pr\{Y = 1|X = 0, \mathbf{Z} = E(\mathbf{Z})\}$.

```
. power logistic 1.5, x(distribution(bernoulli 0.57)) pycondx0zm(0.2)
Estimated sample size for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    1.5000  (odds ratio)
   pycondx0zm =    0.2000
Covariate of interest X: Bernoulli(px), bins = 2
      oratiox =    1.5000
           px =    0.5700
Estimated sample size:
            N =     1,092
```

The result is the same as before: a sample of 1,092 birds is required to detect an odds ratio of 1.5 with 80% power using a 5% level test.

◁

▷ Example 2: Sample size with two binary covariates

Continuing with the study of migratory birds, we now include nuisance covariate Z as an indicator that a bird partners with the same mate. Previous research indicates that 40% of the target population has the same mate as the year before, 16% of male birds with new mates return to the same nesting site they used the previous year, and the odds that a bird with the same mate will return to the same nesting site is 1.8 times the odds that a bird with a new mate will do so. Written mathematically, this is, $\Pr(Z = 1) = 0.4$, $\Pr(Y = 1|X = 0, Z = 0) = 0.16$, and $\text{OR}_Z = 1.8$, respectively.

We can use the two binary syntax of `power logistic` to perform the sample-size calculation. Instead of specifying $\Pr(Y = 1|X = 0)$ in the `pycondx0()` option (as we did when we had only one binary covariate), we now include $\Pr(Y = 1|X = 0, Z = 0)$ using the `pycondx0z0()` option. We specify $\Pr(Z = 1)$ and OR_Z using the `pz()` and `oratioz()` options, respectively.

```
. power logistic 1.5, px(0.57) pz(0.4) oratioz(1.8) pycondx0z0(0.16)
Estimated sample size for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    1.5000  (odds ratio)
      oratiox =    1.5000
           px =    0.5700
      oratioz =    1.8000
           pz =    0.4000
   pycondx0z0 =    0.1600
Estimated sample size:
            N =     1,118
```

The output when using the two binary syntax is similar to the output we saw with the one binary syntax, but now it includes information about nuisance covariate Z. Including Z in the logistic regression brings the required sample size to 1,118 birds.

We now demonstrate how this sample-size calculation can be specified using the general syntax of `power logistic`. We still specify the effect size as an argument, but now we specify the distribution of X in the `x()` option and the distribution of Z in the `z1()` option. We specify the odds ratio for Z using the `oratio()` suboption of `z1()`. Instead of specifying $\Pr(Y = 1|X = 0, Z = 0)$, we now specify the logistic intercept ζ_0 directly using the `intercept()` option, where $\zeta_0 = H^{-1}\{\Pr(Y = 1|X = 0, Z = 0)\} = \texttt{logit(0.16)} = -1.658$.

```
. power logistic 1.5, x(distribution(bernoulli 0.57))
> z1(distribution(bernoulli 0.4) oratio(1.8)) intercept(-1.658)

Estimated sample size for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    1.5000  (odds ratio)
    intercept =   -1.6580

Covariate of interest X: Bernoulli(px), bins = 2
      oratiox =    1.5000
           px =    0.5700

Nuisance covariate Z1: Bernoulli(pz1), bins = 2
     oratioz1 =    1.8000
          pz1 =    0.4000

Estimated sample size:

            N =     1,118
```

The output of the general syntax is formatted differently than the output from the two binary syntax, but the parameters and estimated sample size are the same.

◁

▷ Example 3: Sample size with arbitrary covariates

Continuing with the study of migratory birds, we now include both nuisance covariates: Z_1 is an indicator that a bird has partnered with the same mate, and Z_2 is the bird's weight in grams. We model covariates X and Z_1 as before, and we model Z_2 as a normal random variable with mean $\mu_{Z_2} = 52$, standard deviation $\sigma_{Z_2} = 4$, and effect $\text{OR}_{Z_2} = 1.1$.

In example 1, which did not have nuisance covariates, we observed that $\Pr\{Y = 1|X = 0, \mathbf{Z} = E(\mathbf{Z})\}$ was equal to $\Pr(Y = 1|X = 0) = 0.2$. Now that we have nuisance covariates, the two probabilities are slightly different. $\Pr(Y = 1|X = 0)$ marginalizes out the effect of Z_1 and Z_2 by averaging over the $\mathbf{Z}$ values in the population, yielding the average probability of nesting-site reuse for male birds in the population. On the other hand, $\Pr\{Y = 1|X = 0, \mathbf{Z} = E(\mathbf{Z})\}$ is the probability of nesting-site reuse for a male bird with "average" characteristics. For a detailed description of the difference between the two probabilities, see *Average response versus response at average* in [R] **margins**. We do not know the precise value of $\Pr\{Y = 1|X = 0, \mathbf{Z} = E(\mathbf{Z})\}$, so we perform a sensitivity analysis by specifying multiple values of `pycondx0zm()`.

```
. power logistic 1.5, x(distribution(bernoulli 0.57))
> z1(distribution(bernoulli 0.4) oratio(1.8))
> z2(distribution(normal 52 4) oratio(1.1))
> pycondx0zm(0.18 0.2 0.22)

Estimated sample size for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1

Covariate of interest X: Bernoulli(px)
Nuisance covariates:
   Z1: Bernoulli(pz1)
   Z2: Normal(muz2, sigmaz2)
```

alpha	power	N	delta	oratiox	px	oratioz1	pz1	oratioz2	muz2
.05	.8	1,179	1.5	1.5	.57	1.8	.4	1.1	52
.05	.8	1,106	1.5	1.5	.57	1.8	.4	1.1	52
.05	.8	1,047	1.5	1.5	.57	1.8	.4	1.1	52

sigmaz2	pycondx0zm
4	.18
4	.2
4	.22

`power logistic` displays a table when we provide *numlist* of values for `pycondx0zm()`. We see that a sample of 1,179 birds would be necessary if $\Pr\{Y = 1|X = 0, \mathbf{Z} = E(\mathbf{Z})\} = 0.18$. To draw a power curve for a sample of 1,179, we add options `n(1179)` and `graph`.

```
. power logistic 1.5, x(distribution(bernoulli 0.57))
> z1(distribution(bernoulli 0.4) oratio(1.8))
> z2(distribution(normal 52 4) oratio(1.1))
> pycondx0zm(0.18 0.2 0.22) n(1179) graph

Covariate of interest X: Bernoulli(px)
Nuisance covariates:
   Z1: Bernoulli(pz1)
   Z2: Normal(muz2, sigmaz2)
```

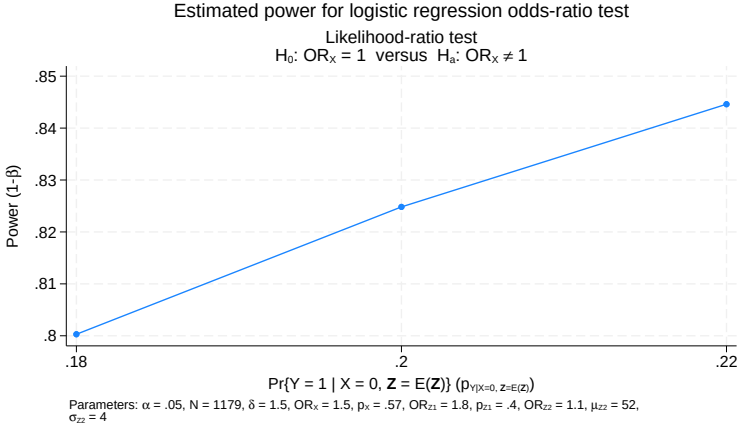


Figure 1. Power curve for a sample of 1,179

As the value of $\Pr\{Y = 1|X = 0, \mathbf{Z} = E(\mathbf{Z})\}$ increases, the power of the test increases as well, reaching a power of nearly 85% when $\Pr\{Y = 1|X = 0, \mathbf{Z} = E(\mathbf{Z})\} = 0.22$.

◁

Reference

Bush, S. 2015. Sample size determination for logistic regression: A simulation study. *Communications in Statistics—Simulation and Computation* 44: 360–373. https://doi.org/10.1080/03610918.2013.777458.

Also see

[PSS-2] **power logistic general** — Power analysis for logistic regression: General case+

[PSS-2] **power logistic onebin** — Power analysis for logistic regression with one binary covariate+

[PSS-2] **power logistic twobin** — Power analysis for logistic regression with two binary covariates+

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[R] **logistic** — Logistic regression, reporting odds ratios

[R] **logit** — Logistic regression, reporting coefficients

[R] **lrtest** — Likelihood-ratio test after estimation

power logistic onebin — Power analysis for logistic regression with one binary covariate[+]

[+]This command is part of StataNow.

Description

`power logistic` computes sample size, power, or effect size for a test of one coefficient in logistic regression. This entry describes how to use `power logistic` to plan a study that will be modeled using logistic regression with one binary covariate. For information about how to use `power logistic` with two binary covariates, see [PSS-2] **power logistic twobin**. To use `power logistic` with many covariates that can be continuous, discrete, or both, see [PSS-2] **power logistic general**.

By default, `power logistic` computes sample size for a given power and effect size, where the effect size may be specified as a coefficient or an odds ratio. Alternatively, it can compute power given sample size and effect size, or it can compute the effect size given power and sample size.

Quick start

Sample size for logistic regression of binary outcome Y on one binary covariate X, given an odds ratio of 1.5 for the effect of X on Y under the alternative hypothesis, population prevalences of X and Y of 0.22 and 0.13, and default power of 0.8 and significance level $\alpha = 0.05$

```
power logistic 1.5, px(0.22) py(0.13)
```

Same as above, but instead of using argument *oratio*$_X$ and the `py()` option to specify parameters for the logistic regression, we specify the respective `coefx()` and `intercept()` options

```
power logistic, px(0.22) coefx(0.4055) intercept(-2)
```

Same as above, but instead specify the odds that $X = 1$ and the outcome success probability conditional on $X = 1$ and $X = 0$

```
power logistic, oddsx(0.2821) pycondx1(0.1686) pycondx0(0.1191)
```

Sample-size calculation for all possible combinations of the specified values

```
power logistic (1.25 1.5 1.75 2), px(0.2 0.22 0.24) py(0.13)
```

Power given an odds ratio of 1.5, population prevalences of X and Y of 0.22 and 0.13, respectively, and a sample size of 2,000

```
power logistic 1.5, px(0.22) n(2000) py(0.13)
```

Same as above, but for different sample sizes and display results in a graph

```
power logistic 1.5, px(0.22) n(1500(250)2500) py(0.13) graph
```

Effect size given $\Pr(X = 1)$ of 0.22, a sample size of 2,000, power of 0.8, and $\Pr(Y = 1|X = 0)$ of 0.12

```
power logistic, px(0.22) n(2000) power(0.8) pycondx0(0.12)
```

Same as above, but display the effect size as a coefficient instead of an odds ratio

```
power logistic, px(0.22) n(2000) power(0.8) pycondx0(0.12) effect(coef)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
power logistic oratio_X , { px(numlist) | oddsx(numlist) } [ power(numlist) onebinopts ]
```

Compute power

```
power logistic oratio_X , { px(numlist) | oddsx(numlist) } n(numlist) [ onebinopts ]
```

Compute effect size

```
power logistic, { px(numlist) | oddsx(numlist) } n(numlist) power(numlist) [ onebinopts ]
```

oratio$_X$ is the odds ratio for binary covariate of interest X under the alternative hypothesis H_a. Argument *oratio*$_X$ may be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

onebinopts	Description
Main	
* alpha(*numlist*)	significance level; default is alpha(0.05)
* power(*numlist*)	power; default is power(0.8)
* beta(*numlist*)	probability of type II error; default is beta(0.2)
* n(*numlist*)	sample size; required to compute power or effect size
nfractional	allow fractional sample size
* coefx(*numlist*)	coefficient for X in logistic regression; specify instead of odds ratio $oratio_X$
effect(oratio | coefficient)	specify the type of effect to display; default is effect(oratio)
†* px(*numlist*)	success probability of X, $\Pr(X=1)$
†* oddsx(*numlist*)	odds of $X=1$
* py(*numlist*)	success probability of Y in the population, $\Pr(Y=1)$
* pycondx1(*numlist*)	success probability of Y given $X=1$, $\Pr(Y=1\|X=1)$
* pycondx0(*numlist*)	success probability of Y given $X=0$, $\Pr(Y=1\|X=0)$
* intercept(*numlist*)	intercept for logistic regression
direction(upper | lower)	direction of the effect for effect-size determination; default is direction(upper), which means that the postulated odds ratio for X is greater than 1 (thus, the coefficient is positive)
parallel	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[no]table[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
saving(*filename* [, replace])	save the table data to *filename*; use replace to overwrite existing *filename*
Graph	
graph[(*graphopts*)]	graph results; see [PSS-2] **power, graph**
Iteration	
init(#)	initial odds ratio for effect-size calculation
iterate(#)	maximum number of iterations; default is iterate(500)
tolerance(#)	parameter tolerance; default is tolerance(1e-12)
ftolerance(#)	function tolerance; default is ftolerance(1e-12)
[no]log	suppress or display iteration log
[no]dots	suppress or display iterations as dots
notitle	suppress the title

†Either px() or oddsx() is required.

*Specifying a list of values in at least two starred options, or in argument $oratio_X$ and at least one starred option, results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see parallel.

collect is allowed; see **[U] 11.1.10 Prefix commands**.

notitle does not appear in the dialog box.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`N`	number of subjects	N
`delta`	effect size	δ
`oratiox`	odds ratio for X	OR_X
`coefx`	coefficient for X	β_X
`px`	success probability of X, $\Pr(X=1)$	p_X
`oddsx`	odds of $X=1$	$p_X/(1-p_X)$
`py`	success probability of Y, $\Pr(Y=1)$	p_Y
`pycondx1`	success probability of Y given X is 1, $\Pr(Y=1\|X=1)$	$p_{Y\|X=1}$
`pycondx0`	success probability of Y given X is 0, $\Pr(Y=1\|X=0)$	$p_{Y\|X=0}$
`intercept`	intercept	ζ_0
`target`	target parameter; odds ratio or coefficient for X	
`_all`	display all supported columns	

Options

Main

`alpha()`, `power()`, `beta()`, `n()`, `nfractional`; see [PSS-2] **power**. The `nfractional` option is allowed only for sample-size determination.

`coefx(`*numlist*`)` specifies the coefficient for binary covariate X in the logistic regression, β_X, where $\beta_X \neq 0$. The `coefx()` option may be specified instead of argument *oratio*$_X$. This option is not allowed with effect-size determination.

`effect(oratio | coefficient)` specifies how to report the effect size in the output. If `coefx()` is specified, then by default the effect size is β_X, the coefficient for X. Otherwise, the default effect size is the odds ratio for X: $\Pr(Y=1|X=1)\Pr(Y=0|X=0)/\{\Pr(Y=0|X=1)\Pr(Y=1|X=0)\}$. The `effect()` option is used to override the default.

`px(`*numlist*`)` specifies the success probability of binary covariate X in the target population, $\Pr(X=1)$, where $0<\Pr(X=1)<1$. Either the `px()` or `oddsx()` option must be specified, but not both.

`oddsx(`*numlist*`)` specifies the odds of binary covariate X in the target population, $\Pr(X=1)/\Pr(X=0)$, where $\Pr(X=1)/\Pr(X=0)>0$. Either the `px()` or `oddsx()` option must be specified, but not both.

`py(`*numlist*`)` specifies the marginal success probability of Y in the target population, $\Pr(Y=1)$, where $0<\Pr(Y=1)<1$. This option is not allowed with effect-size determination.

`pycondx1(`*numlist*`)` specifies the conditional success probability of Y given X equals 1, $\Pr(Y=1|X=1)$, where $0<\Pr(Y=1|X=1)<1$. This option is not allowed with effect-size determination.

`pycondx0(`*numlist*`)` specifies the conditional success probability of Y given X equals 0, $\Pr(Y = 1|X = 0)$, where $0 < \Pr(Y = 1|X = 0) < 1$. Only one of the `intercept()` or `pycondx0()` option may be specified.

`intercept(`*numlist*`)` specifies the intercept for the logistic regression, ζ_0. Only one of the `intercept()` or `pycondx0()` option may be specified.

`direction()`, `parallel`; see [PSS-2] **power**.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies the initial value of the odds ratio for binary covariate X during effect-size determination.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following option is available with `power logistic` but is not shown in the dialog box:

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `power logistic` command and the methodology for power and sample-size analysis for logistic regression with one binary covariate. See [PSS-2] **power logistic** for power and sample-size analysis for logistic regression in other settings; [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis; and [PSS-2] **power** for a general introduction to using the `power` command for hypothesis tests.

Introduction

Logistic regression is a commonly used statistical method for analyzing binary outcome variables. Researchers often need to determine the appropriate sample size to ensure sufficient power for detecting an association between a binary covariate of interest and a binary outcome variable. For example, consider a study that examines whether migratory birds return to the same nesting site from one year to the next. Binary outcome Y is an indicator of whether the nesting site was reused, where the observed $y_i = 1$ if bird i returns to the nesting site it used last year and $y_i = 0$ if bird i does not. We will use

logistic regression to test whether birds of one sex are more likely to return to the same nesting site than birds of the other sex. We define binary covariate of interest X as an indicator for female birds, where the observed $x_i = 1$ if bird i is female and $x_i = 0$ if bird i is male.

The logistic regression can be written as

$$\Pr(y_i = 1|x_i) = H(\beta_X x_i + \zeta_0) \qquad i = 1, 2, \ldots, n$$

where β_X is the coefficient quantifying the effect of covariate X, a bird's sex, on nesting-site reuse; ζ_0 is the logistic intercept; and n is the sample size. Function $H(\eta) = \{1 + \exp(-\eta)\}^{-1}$ is the logistic distribution function. The effect of X can also be expressed in terms of an odds ratio,

$$\text{OR}_X = \exp(\beta_X) = \Pr(Y = 1|X = 1)\Pr(Y = 0|X = 0)/\{\Pr(Y = 0|X = 1)\Pr(Y = 1|X = 0)\}$$

The null hypothesis is $H_0 : \beta_X = 0$, which can also be expressed as $H_0 : \text{OR}_X = 1$, and the alternative hypothesis is $H_a : \beta_X \neq 0$ or, equivalently, $H_a : \text{OR}_X \neq 1$.

Logistic regression is commonly used to analyze binary outcomes in observational studies, such as the study of nesting-site reuse. But it can also be used to analyze data from a randomized controlled trial, where trial participants are randomly assigned to a treatment. For instance, a public health study might investigate whether attending a support group helps smokers quit smoking. In this example, participation in the support group is the binary covariate of interest: Some participants will be randomly assigned to attend support group meetings for, say, three months, whereas other participants will be randomized to the control group, which does not attend support group meetings. At the end of three months, smoking status is recorded; this is the binary outcome.

Here the observed $x_i = 1$ if participant i attends the support group, and $x_i = 0$ if participant i does not attend the support group. The observed $y_i = 1$ if participant i successfully quits smoking by the end of the three-month study, and $y_i = 0$ if participant i continues to smoke. The null and alternative hypotheses are $H_0 : \beta_X = 0$ and $H_a : \beta_X \neq 0$, respectively.

The `power logistic` command provides power and sample-size analysis for the test of $\beta_X = 0$ in logistic regression. The formula for power, sample-size, and effect-size calculations is based on the likelihood-ratio test of $\beta_X = 0$. However, Bush (2015) demonstrates that sample-size requirements for Wald and score tests are generally equivalent to the sample-size requirement for the likelihood-ratio test. Thus, these calculations may be used to plan studies that will be analyzed using the `logit` command, which conducts a Wald test of $\beta_X = 0$, or the `logistic` command, which conducts an equivalent Wald test of $\text{OR}_X = 1$. If you prefer a likelihood-ratio test, you can use the `lrtest` command.

Using power logistic with one binary covariate

`power logistic` computes sample size, power, or effect size for a test of one coefficient in logistic regression. This entry describes how to use `power logistic` when the logistic regression has a single binary covariate. All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option.

You must specify either the prevalence or the odds of a binary covariate X in the target population in the respective `px()` or `oddsx()` option. The `px()` option specifies $\Pr(X = 1)$, whereas the `oddsx()` option specifies $\Pr(X = 1)/\Pr(X = 0)$. When you collect observational data, $\Pr(X = 1)$ indicates the prevalence of X in the target population. In the observational study about nesting-site reuse, $\Pr(X = 1)$ is the proportion of female birds in the population. But when you collect data from a randomized controlled trial, the "target population" is restricted to study participants, and the meaning of $\Pr(X = 1)$ changes

accordingly. In the smoking-cessation trial, $\Pr(X = 1)$ is the proportion of participants who are randomly assigned to the support group. The ratio of participants in the experimental arm to the control arm is known as the allocation ratio.

Power and sample-size calculations require that you specify information about two parameters for logistic regression: the X coefficient (β_X) and the intercept (ζ_0). The information about β_X can be specified directly in the `coefx()` option or indirectly as an argument *oratio*$_X$, in the `py()` option, or in the `pycondx1()` option. The information about ζ_0 can be specified directly in the `intercept()` option or indirectly in the `pycondx0()`, `pycondx1()`, or `py()` option. See figure 1 for a graphical depiction of various ways to provide information about β_X and ζ_0.

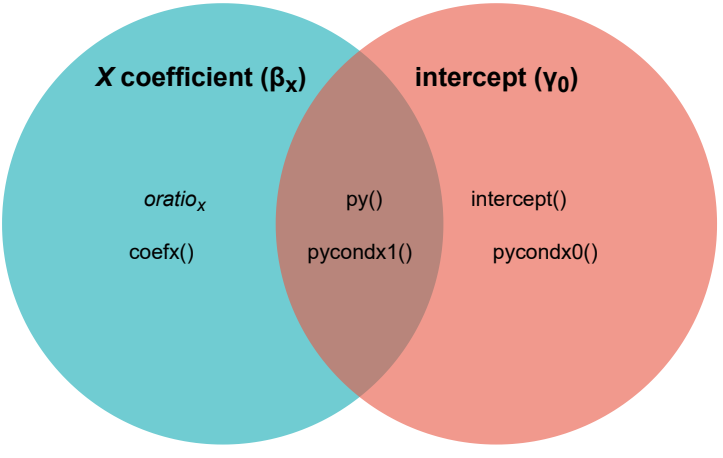


Figure 1. Specifying information about β_X and ζ_0

Valid ways of specifying the X coefficient and intercept include, for example, `coefx()` and `intercept()`, `py()` and `pycondx0()`, or `py()` and `pycondx1()`. However, the combination of `intercept()` and `pycondx0()` is invalid because no information is provided about the X coefficient. When you compute sample size, the power of the test may be specified using the `power()` option, which has a default of 0.8. When you compute power, the sample size must be specified using the `n()` option.

When `power logistic` is used to calculate effect size β_X, the command specification cannot include *oratio*$_X$ or options that provide information about the X coefficient, such as the `py()` and `pycondx1()` options. In this case, the information about the intercept must be specified using `intercept()` or `pycondx0()`. To calculate effect size, you must specify sample size using the `n()` option, power using the `power()` option, and, optionally, the direction of the effect using the `direction()` option. The default is `direction(upper)`, which means that coefficient β_X is assumed to be positive. This is equivalent to assuming that the odds ratio OR_X is greater than 1. You can change the direction to `lower`, which means that $\beta_X < 0$ or, equivalently, $\text{OR}_X < 1$.

The `effect()` option can be used to specify the type of the effect size to be reported in the output. Valid choices are `effect(oratio)` and `effect(coefficient)`. By default, the effect size is output as the odds ratio for X unless `coefx()` is specified, in which case it defaults to the coefficient for X. The `effect()` option is used to override the default.

By default, the computed sample size is rounded up. You can specify the `nfractional` option to see the corresponding fractional sample size; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. The `nfractional` option is allowed only for sample-size determination.

Some of the computations of power logistic require iteration, specifically, the computations used in effect-size determination. The default initial value of the estimated effect size is obtained using a bisection algorithm. This may be changed by specifying the init() option. See [PSS-2] **power** for descriptions of other options that control the iteration procedure.

In the following sections, we describe the use of power logistic with one binary covariate accompanied by examples for computing sample size, power, and effect size.

Computing sample size

To compute sample size, you must specify the prevalence or odds of X in the target population in the respective px() or oddsx() option; the information necessary to determine parameters β_X and ζ_0 (as described in *Using power logistic with one binary covariate* above); and, optionally, the power of the test using the power() option or the type-II-error probability using the beta() option. A default power of 0.8 is assumed if power() is not specified.

▷ Example 1: Sample size when OR_X and $\Pr(Y|X=0)$ are known

Consider an example from the seminal work on sample-size calculation for logistic regression by Whittemore (1981, 31) that describes the design of a study testing "the null hypothesis that risk of coronary heart disease (CHD) among white males aged 39–59 is unaffected by serum cholesterol levels". Here the X variable is an indicator for elevated serum cholesterol, where the observed $x_i = 1$ if participant i has elevated serum cholesterol and $x_i = 0$ otherwise. Based on data from Hulley et al. (1980), we assume a probability of 0.07 that an individual in the target population will develop CHD during an 18-month study period if he does not have elevated serum cholesterol; this will be entered using the pycondx0() option. We assume that 13% of our target population has elevated serum cholesterol, corresponding to px(0.13). Like Whittemore, we will calculate the sample size required to detect an odds ratio of 1.65 at the $\alpha = 0.05$ level, but we will use the default power of 80%.

```
. power logistic 1.65, px(0.13) pycondx0(0.07)

Estimated sample size for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    1.6500  (odds ratio)
      oratiox =    1.6500
           px =    0.1300
    pycondx0 =    0.0700

Estimated sample size:

            N =     3,256
```

The output of power logistic begins by displaying information about the test to be conducted and its null and alternative hypotheses. The study parameters we specified are listed next, followed by the estimated sample size. We find that a sample of 3,256 subjects is required to detect an odds ratio of 1.65 with 80% power using a 5% level test.

◁

▷ Example 2: Sample size when β_X and $\Pr(Y|X=0)$ are known

Instead of the odds ratio for X, as in example 1, we specify the effect as $\beta_X = \log(1.65) = 0.50077$. We leave the rest of the command specification unchanged.

```
. power logistic, px(0.13) coefx(0.50077) pycondx0(0.07)

Estimated sample size for logistic regression coefficient test
Likelihood-ratio test
H0: beta_X = 0  versus  Ha: beta_X != 0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.5008  (coefficient)
        coefx =    0.5008
           px =    0.1300
     pycondx0 =    0.0700

Estimated sample size:

            N =     3,256
```

The output of this command is identical to the output displayed in example 1, with the exception that the coefficient for X, `coefx`, is displayed instead of the odds ratio. If we wanted to see the odds ratio in the output despite inputting the coefficient for X, we could add the `effect(oratio)` option.

◁

▷ Example 3: Sample size when ζ_0 and $\Pr(Y)$ are known

Now we assume that we know $\Pr(Y = 1)$, the probability of observing CHD in a randomly selected individual from the study population. Recall from example 1 that the probability of observing CHD in an individual without elevated serum cholesterol was 7%; once we include individuals with elevated cholesterol, the overall probability of CHD rises to 7.5261%. This is calculated as $\Pr(Y = 1) = \text{invlogit}(\text{logit}(0.07) + 0.50077) \times 0.13 + 0.07 \times 0.87 = 0.075261$. For demonstration purposes, instead of specifying `pycondx0()`, we specify the intercept directly as $\zeta_0 = \texttt{logit(0.07)} = -2.5867$.

```
. power logistic, px(0.13) py(0.075261) intercept(-2.5867)

Estimated sample size for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    1.6500  (odds ratio)
      oratiox =    1.6500
           px =    0.1300
           py =    0.0753
    intercept =   -2.5867

Estimated sample size:

            N =     3,256
```

The output of this command is identical to the output displayed in example 1, with the exception that study parameters `intercept` and `py` are displayed instead of `pycondx0`. If we wanted to see `coefx` in the output instead of `oratiox`, we could add the `effect(coefficient)` option.

◁

Computing power

To compute power, you must specify the sample size using the `n()` option, the prevalence or odds of X in the target population using the respective `px()` or `oddsx()` option, and the information necessary to determine parameters β_X and ζ_0 (as described in *Using power logistic with one binary covariate*).

▷ Example 4: Power of a logistic regression odds-ratio test

Continuing with example 1, we will suppose that we are designing a new study and anticipate a sample of 3,000 subjects. To compute the power corresponding to this sample size given the study parameters from example 1, we specify the sample size of 3,000 in the `n()` option:

```
. power logistic 1.65, px(0.13) n(3000) pycondx0(0.07)

Estimated power for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1

Study parameters:

        alpha =    0.0500
            N =     3,000
        delta =    1.6500  (odds ratio)
      oratiox =    1.6500
           px =    0.1300
     pycondx0 =    0.0700

Estimated power:

        power =    0.7672
```

If the study recruits only 3,000 participants, the power to detect an odds ratio of 1.65 drops to 76.72%.

◁

▷ Example 5: Multiple values of study parameters

To investigate the effect of sample size on power, we can specify a list of sample sizes in the `n()` option:

```
. power logistic 1.65, px(0.13) n(3000 4000 5000) pycondx0(0.07)

Estimated power for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1
```

alpha	power	N	delta	oratiox	px	pycondx0
.05	.7672	3,000	1.65	1.65	.13	.07
.05	.874	4,000	1.65	1.65	.13	.07
.05	.9348	5,000	1.65	1.65	.13	.07

As expected, when the sample size increases, the power increases toward 1.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

Computing effect size

By default, effect size δ for a logistic regression odds-ratio test is defined as the odds ratio for X: $\delta = \text{OR}_X = \exp(\beta_X)$. Sometimes, we want to know the smallest effect that can be detected with a level α test at a prespecified power and sample size.

To compute the minimum detectable effect size, you must specify sample size and power, as well as the intercept parameter ζ_0. This may be done via the `intercept()` or `pycondx0()` option; for details about specifying logistic regression parameters, see *Using power logistic with one binary covariate*. In addition, you must pick the level of the test and the direction of the effect. The level of the test is specified using the `alpha()` option, with a default of `alpha(0.05)`. The direction of the effect is specified using the `direction()` option; the default is `direction(upper)`, which means that $\text{OR}_X > 1$ or, equivalently, $\beta_X > 0$. Specifying `direction(lower)` means that $\text{OR}_X < 1$ and $\beta_X < 0$. The estimated minimum detectable effect size is reported as an odds ratio by default. To display it as a coefficient, specify `effect(coefficient)`.

▷ Example 6: Minimum detectable odds ratio

We continue with example 4, where we learned that a size 0.05 test with 3,000 subjects would have 76.72% power to detect an odds ratio of 1.65. How much larger would the odds ratio need to be to detect it with 80% power? We use `power logistic` to find out.

```
. power logistic, px(0.13) n(3000) power(0.8) pycondx0(0.07)

Performing iteration ...

Estimated odds ratio for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
            N =     3,000
           px =    0.1300
     pycondx0 =    0.0700

Estimated effect size and odds ratio:

        delta =    1.6806  (odds ratio)
      oratiox =    1.6806
```

We see that a slightly larger odds ratio of 1.68 can be detected with 80% power.

In the above, we assumed the effect to be in the upper direction. There exists an effect size in the lower direction that can also be detected with 80% power. We specify `direction(lower)` to find it, and we add the `effect(coefficient)` option to display it as a coefficient instead of an odds ratio.

```
. power logistic, px(0.13) n(3000) power(0.8) pycondx0(0.07)
> direction(lower) effect(coefficient)

Performing iteration ...

Estimated coefficient for logistic regression coefficient test
Likelihood-ratio test
H0: beta_X = 0  versus  Ha: beta_X != 0

Study parameters:

        alpha =    0.0500
        power =    0.8000
            N =     3,000
           px =    0.1300
     pycondx0 =    0.0700

Estimated effect size and coefficient:

        delta =   -0.7213  (coefficient)
        coefx =   -0.7213
```

By specifying `direction(lower)`, we anticipate coefficient $\beta_X < 0$, which is what we see. Had we omitted the `effect(coefficient)` option, the odds ratio $\text{OR}_X = \exp(\beta_X)$ would have been displayed as $\exp(-0.7213) = 0.486$.

◁

Performing hypothesis tests with logistic regression

In this section, we briefly demonstrate the use of the `logistic` command for testing logistic regression coefficients; see [R] **logistic** for details. Alternatively, we could use the `logit` command to perform logistic regression because `logit` performs the same calculations as `logistic` but reports coefficients instead of odds ratios; see [R] **logit** for details and example 7 of [PSS-2] **power logistic twobin** for a demonstration of how `logit` can be used to analyze the results of a pilot study.

▷ Example 7: Analyzing a pilot study

`nlsw88.dta` contains employment data from the 1988 extract of the National Longitudinal Study of Young Women. We will treat this dataset as if it came from a pilot study investigating the relationship between college graduation and marital status and use it to plan a follow-up study. Our population of interest is American young women who are union members, so we will drop nonmembers from the dataset and then perform logistic regression of `collgrad` on `married`.

```
. use https://www.stata-press.com/data/r19/nlsw88
(NLSW, 1988 extract)
. keep if union == 1
(1,785 observations deleted)
. logistic collgrad married
Logistic regression                                     Number of obs =    461
                                                        LR chi2(1)    =   2.77
                                                        Prob > chi2   = 0.0959
Log likelihood = -287.96102                             Pseudo R2     = 0.0048

------------------------------------------------------------------------------
    collgrad | Odds ratio   Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
     married |   1.410769   .2936711     1.65   0.098     .9381378    2.121511
       _cons |   .3816794   .0634479    -5.79   0.000       .27555     .528685
------------------------------------------------------------------------------
Note: _cons estimates baseline odds.
```

The odds ratio OR_X of 1.41 suggests that married women are more likely to be college graduates than unmarried women, but the evidence is not strong enough to reject $H_0 : \text{OR}_X = 1$ at the 0.05 level. We use the parameter estimates from the pilot study to calculate the sample size for a follow-up study that has 80% power to detect an odds ratio of 1.41 with a 0.05-level test.

We will use `power logistic` with an effect size OR_X of 1.41, but what about the other parameters from the logistic regression? We also need to specify information about the intercept and $\Pr(X = 1)$.

The `logistic` command displays parameter estimates as odds ratios, so we take the natural logarithm of baseline odds parameter `_cons` to get the intercept from the logistic regression: $\log(0.38) \approx -0.96$. (Or we could have used `logistic, coef` to display coefficients instead of odds ratios.) To find $\Pr(X = 1)$, we use the `tabulate` command to calculate the prevalence of marriage in the target population; see [R] **tabulate oneway** for details.

```
. tabulate married
    Married |      Freq.     Percent        Cum.
------------+-----------------------------------
     Single |        181       39.26       39.26
    Married |        280       60.74      100.00
------------+-----------------------------------
      Total |        461      100.00
```

Approximately 61% of sampled women were married, so we will specify `px(0.61)`. We omit the `power()` option because we are designing our follow-up study to have 80% power, which is the default.

```
. power logistic 1.41, px(0.61) intercept(-0.96)
Estimated sample size for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    1.4100  (odds ratio)
      oratiox =    1.4100
           px =    0.6100
    intercept =   -0.9600

Estimated sample size:

            N =     1,310
```

We find that the sample size required to detect an odds ratio of 1.41 with 80% power using a 5% level test is 1,310. One detail that bears mentioning is that this sample-size calculation is for a likelihood-ratio test, but the `logistic` and `logit` commands report Wald tests of coefficients. Fortunately, an extensive simulation study by Bush (2015) demonstrates that sample-size requirements for Wald and likelihood-ratio tests of logistic regression coefficients are nearly identical. If you prefer a likelihood-ratio test, you can use the `lrtest` command; see [R] **lrtest** for details.

◁

Stored results

`power logistic` with one binary covariate stores the following in `r()`:

Scalars

`r(alpha)`	significance level
`r(power)`	power
`r(beta)`	probability of a type II error
`r(delta)`	effect size
`r(N)`	sample size
`r(nfractional)`	`1` if `nfractional` is specified, `0` otherwise
`r(px)`	success probability of X
`r(oddsx)`	odds that $X = 1$
`r(oratiox)`	odds ratio for X
`r(coefx)`	coefficient for X
`r(intercept)`	intercept from logistic regression
`r(py)`	success probability of Y in the target population
`r(pycondx1)`	success probability of Y given $X = 1$
`r(pycondx0)`	success probability of Y given $X = 0$
`r(separator)`	number of lines between separator lines in the table
`r(divider)`	`1` if `divider` is requested in the table, `0` otherwise
`r(init)`	initial value for odds ratio (if specified)
`r(maxiter)`	maximum number of iterations (for effect-size calculation)
`r(iter)`	number of iterations performed (for effect-size calculation)
`r(tolerance)`	requested parameter tolerance (for effect-size calculation)
`r(deltax)`	final parameter tolerance achieved (for effect-size calculation)
`r(ftolerance)`	requested distance of the objective function from zero (for effect-size calculation)
`r(function)`	final distance of the objective function from zero (for effect-size calculation)
`r(converged)`	`1` if iteration algorithm converged, `0` otherwise (for effect-size calculation)

Macros

`r(type)`	`test`
`r(method)`	`logistic`
`r(direction)`	`upper` or `lower` (for effect-size calculation)
`r(columns)`	displayed table columns
`r(labels)`	table column labels
`r(widths)`	table column widths
`r(formats)`	table column formats

Matrices

`r(pss_table)`	table of results

Methods and formulas

Methods and formulas are presented under the following headings:

Coefficient tests in logistic regression

Shieh (2000a) used simulation to compare the performance of two sample-size formulas for coefficient tests in logistic regression: the method of Whittemore (1981) and that of Self, Mauritsen, and Ohara (1992). Shieh generalized the superior of those two methods, that of Self, Mauritsen, and Ohara, in Shieh (2000b), which provides the formulas implemented in `power logistic` for power, sample-size, and effect-size calculations for likelihood-ratio tests in logistic regression.

In practice, it is more common to use the Wald test of logistic regression coefficients than the likelihood-ratio test, so there has been some concern about whether these calculations are appropriate for use with a Wald test (Demidenko 2007). Demidenko notes that the Wald and likelihood-ratio tests have asymptotically equivalent type I errors and that they are "*locally* equivalent, so that the power functions are close when the alternative approaches the null" (Demidenko 2007, 3385). Nevertheless, Demidenko raises the point that the two tests are not globally equivalent, so there is no theoretical guarantee that the power functions will be similar under the alternative hypothesis. Thankfully, an extensive simulation study by Bush (2015) found little difference between the power curves of the Wald, likelihood-ratio, and score tests over a range of scenarios. Additionally, Bush compared the performance of seven sample-size formulas for logistic regression and determined that the method of Shieh (2000b) was consistently accurate, regardless of the test that was used.

Logistic regression

The logistic regression with one binary covariate can be written as

$$\Pr(y_i = 1|x_i) = H(\beta_X x_i + \zeta_0) \qquad i = 1, 2, \ldots, n$$

where β_X is the coefficient for covariate X, ζ_0 is the logistic intercept, and n is the sample size. Function $H(\eta) = \{1 + \exp(-\eta)\}^{-1}$ is the logistic distribution function.

The effect of binary covariate X can also be expressed in terms of an odds ratio,

$$\text{OR}_X = \exp(\beta_X) = \Pr(Y = 1|X = 1)\Pr(Y = 0|X = 0)/\{\Pr(Y = 0|X = 1)\Pr(Y = 1|X = 0)\}$$

The null hypothesis is $H_0 : \beta_X = 0$, which can also be expressed as $H_0 : \text{OR}_X = 1$, and the alternative hypothesis is $H_a : \beta_X \neq 0$ or, equivalently, $H_a : \text{OR}_X \neq 1$.

Parameters β_X and ζ_0 may be directly specified by using the `coefx()` and `intercept()` options, respectively. But when the `py()` option is used to specify $\Pr(Y = 1)$, we need the equation

$$\Pr(Y = 1) = \Pr(X = 1) \times H(\beta_X + \zeta_0) + \Pr(X = 0) \times H(\zeta_0)$$

to solve for the unknown parameter, either β_X or ζ_0.

Power, sample-size, and effect-size calculations

Shieh (2000b) builds on the work of Self, Mauritsen, and Ohara (1992) and Self and Mauritsen (1988) to estimate the distribution of the likelihood-ratio statistic: $2\{l(\hat{\beta}_X, \hat{\mathbf{c}}) - l(0, \hat{\mathbf{c}}_0)\}$. Here $l(\cdot)$ is the log-likelihood function for the logistic regression, and $\mathbf{c}$ is a vector of nuisance parameters; for logistic regression with one binary covariate, the only nuisance term is the logistic intercept, so $\mathbf{c} = \zeta_0$. $\hat{\beta}_X$ and $\hat{\mathbf{c}}$ are the maximum likelihood estimates of β_X and $\mathbf{c}$ under the alternative hypothesis, and $\hat{\mathbf{c}}_0$ is the maximum likelihood estimate of $\mathbf{c}$ under the null hypothesis. If the null hypothesis is not true, $\hat{\mathbf{c}}_0$ is not a consistent estimate of $\mathbf{c}$ but instead converges to $\mathbf{c}_0^* = \zeta_0^* = \zeta_0 + \beta_X E(X)$, as described in Self and Mauritsen (1988, eq. 2.2). For a full decomposition of the likelihood-ratio statistic, see Shieh (2000b, 1193).

To calculate sample size, we begin by specifying the desired size of the test (also known as the type I error, α) and the desired power (which equals $1 - \beta$, where β is the desired type II error of the test). We equate the $(1 - \alpha)$100th percentile of a central χ^2 distribution with 1 degree of freedom to the β100th percentile of a noncentral χ^2 distribution with noncentrality parameter $\lambda = n\Delta^*$. We define Δ^* as

$$\Delta^* = 2E[H(\eta)(\eta - \eta^*) - \log\{1 + \exp(\eta)\} + \log\{1 + \exp(\eta^*)\}]$$

where $\eta = \beta_X X + \mathbf{c}$ and $\eta^* = \mathbf{c}_0^*$. Sample size is calculated as $n = \lambda/\Delta^*$. Power is computed similarly by starting with a known value of n and solving for the power required to yield the desired value of λ.

There is no closed-form expression that can be used to calculate effect size δ, either coefficient β_X or odds ratio OR_X. Effect size is estimated iteratively, and the default starting value for δ is calculated using a bisection algorithm, which is refined using a nonlinear solver; see [M-5] **solvenl( )** for details. To skip the bisection algorithm, you can use the `init()` option to specify a starting odds ratio for the nonlinear solver. You can control the iteration process with the `iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, and `nodots` options.

References

Bush, S. 2015. Sample size determination for logistic regression: A simulation study. *Communications in Statistics—Simulation and Computation* 44: 360–373. https://doi.org/10.1080/03610918.2013.777458.

Demidenko, E. 2007. Sample size determination for logistic regression revisited. *Statistics in Medicine* 26: 3385–3397. https://doi.org/10.1002/sim.2771.

Hulley, S. B., R. H. Rosenman, R. D. Bawol, and R. J. Brand. 1980. Epidemiology as a guide to clinical decisions—the association between triglyceride and coronary heart disease. *New England Journal of Medicine* 302: 1383–1389. https://doi.org/10.1056/nejm198006193022503.

Self, S. G., and R. H. Mauritsen. 1988. Power/sample size calculations for generalized linear models. *Biometrika* 44: 79–86. https://doi.org/10.2307/2531897.

Self, S. G., R. H. Mauritsen, and J. Ohara. 1992. Power calculations for likelihood ratio tests in generalized linear models. *Biometrika* 48: 31–39. https://doi.org/10.2307/2532736.

Shieh, G. 2000a. A comparison of two approaches for power and sample size calculations in logistic regression models. *Communications in Statistics—Simulation and Computation* 29: 763–791. https://doi.org/10.1080/03610910008813639.

———. 2000b. On power and sample size calculations for likelihood ratio tests in generalized linear models. *Biometrics* 56: 1192–1196. https://doi.org/10.1111/j.0006-341x.2000.01192.x.

Whittemore, A. S. 1981. Sample size for logistic regression with small response probability. *Journal of the American Statistical Association* 76: 27–32. https://doi.org/10.2307/2287036.

Also see

[PSS-2] **power logistic** — Power analysis for logistic regression+

[PSS-2] **power logistic general** — Power analysis for logistic regression: General case+

[PSS-2] **power logistic twobin** — Power analysis for logistic regression with two binary covariates+

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[R] **logistic** — Logistic regression, reporting odds ratios

[R] **logit** — Logistic regression, reporting coefficients

[R] **lrtest** — Likelihood-ratio test after estimation

power logistic twobin — Power analysis for logistic regression with two binary covariates[+]

[+]This command is part of StataNow.

Description

`power logistic` computes sample size, power, or effect size for a test of one coefficient in logistic regression. This entry describes how to use `power logistic` to plan a study that will be modeled using logistic regression with two binary covariates, one of which is the covariate of interest (X) and the other is a nuisance covariate (Z). For information about how to use `power logistic` with one binary covariate, see [PSS-2] **power logistic onebin**. To use `power logistic` with covariates that can be continuous, discrete, or both, see [PSS-2] **power logistic general**.

By default, `power logistic` computes sample size for a given power and effect size, where the effect size may be specified as a coefficient or an odds ratio. Alternatively, it can compute power given sample size and effect size, or it can compute the effect size given power and sample size.

Quick start

Sample size for logistic regression of binary outcome Y on binary covariate of interest X and nuisance binary covariate Z, given an odds ratio of 2 for the effect of X on Y under the alternative hypothesis, population prevalences of X and Z of 0.3 and 0.5, respectively, an odds ratio of 3 for the effect of Z on Y, population prevalence of outcome Y of 0.34, and default power of 0.8 and significance level $\alpha = 0.05$

```
power logistic 2, px(0.3) pz(0.5) oratioz(3) py(0.34)
```

Same as above, but X and Z are correlated, with a phi coefficient (correlation between X and Z) of 0.2

```
power logistic 2, px(0.3) pz(0.5) corrxz(0.2) oratioz(3) py(0.34)
```

Same as above, but instead use the `coefx()`, `coefz()` , and `intercept()` options to specify coefficients and the intercept for the logistic regression instead of odds ratios and outcome prevalence

```
power logistic, px(0.3) pz(0.5) corrxz(0.2) coefx(0.69) coefz(1.1)  ///
  intercept(-1.49)
```

Same as above, but instead use the `pycondx1z1()`, `pycondx1z0()`, and `pycondx0z1()` options to specify information about model parameters of logistic regression by using the outcome success probability conditional on X and Z

```
power logistic, px(0.3) pz(0.5) corrxz(0.2) pycondx1z1(0.57)          ///
  pycondx1z0(0.31) pycondx0z1(0.4)
```

Same as above, but instead of using the `px()` and `pz()` options, use the `oddsx()` and `oddsz()` options to specify the odds of X and Z in the population, and specify a list of values for the correlation

```
power logistic, oddsx(0.43) oddsz(1) corrxz(0 0.1 0.2 0.3)            ///
  pycondx1z1(0.57) pycondx1z0(0.31) pycondx0z1(0.4)
```

Power for sample size of 300

```
power logistic 2, px(0.3) pz(0.5) n(300) oratioz(3) py(0.34)
```

Same as above, but compute power for different sample sizes, and display results in a graph

```
power logistic 2, px(0.3) pz(0.5) n(200(50)400) oratioz(3) py(0.34) graph
```

Effect size for a sample size of 300 and power of 0.8

```
power logistic, px(0.3) pz(0.5) n(300) power(0.8) oratioz(3)          ///
   intercept(-1.49)
```

Same as above, but instead of specifying the intercept value, specify $\Pr(Y = 1|X = 0, Z = 0)$, and display the effect size as a coefficient instead of an odds ratio

```
power logistic, px(0.3) pz(0.5) n(300) power(0.8) oratioz(3)          ///
   pycondx0z0(0.18) effect(coefficient)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
power logistic oratio_X , { px(numlist) | oddsx(numlist) } { pz(numlist) | oddsz(numlist) }
                          [ power(numlist) twobinopts ]
```

Compute power

```
power logistic oratio_X , { px(numlist) | oddsx(numlist) } { pz(numlist) | oddsz(numlist) }
                          n(numlist) [ twobinopts ]
```

Compute effect size

```
power logistic, { px(numlist) | oddsx(numlist) } { pz(numlist) | oddsz(numlist) }
                n(numlist) power(numlist) [ twobinopts ]
```

oratio$_X$ is the odds ratio for covariate of interest X under the alternative hypothesis H_a. Argument *oratio*$_X$ may be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

twobinopts	Description
Main	
* alpha(*numlist*)	significance level; default is alpha(0.05)
* power(*numlist*)	power; default is power(0.8)
* beta(*numlist*)	probability of type II error; default is beta(0.2)
* n(*numlist*)	sample size; required to compute power or effect size
nfractional	allow fractional sample size
* coefx(*numlist*)	coefficient for X in logistic regression; specify instead of odds ratio $oratio_X$
effect(oratio | coefficient)	specify the type of effect to display; default is effect(oratio)
†* px(*numlist*)	success probability of X, $\Pr(X=1)$
†* oddsx(*numlist*)	odds of $X=1$
* oratioz(*numlist*)	odds ratio for Z
* coefz(*numlist*)	coefficient for Z in logistic regression
†* pz(*numlist*)	success probability of Z, $\Pr(Z=1)$
†* oddsz(*numlist*)	odds of $Z=1$
* corrxz(*numlist*)	phi coefficient for correlation between X and Z
* py(*numlist*)	success probability of Y in the population, $\Pr(Y=1)$
* pycondx1z1(*numlist*)	success probability of Y given $X=1$ and $Z=1$
* pycondx1z0(*numlist*)	success probability of Y given $X=1$ and $Z=0$
* pycondx0z1(*numlist*)	success probability of Y given $X=0$ and $Z=1$
* pycondx0z0(*numlist*)	success probability of Y given $X=0$ and $Z=0$
* intercept(*numlist*)	intercept for logistic regression
direction(upper | lower)	direction of the effect for effect-size determination; default is direction(upper), which means that the postulated odds ratio for X is greater than 1 (thus, the coefficient is positive)
parallel	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[no]table[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
saving(*filename* [, replace])	save the table data to *filename*; use replace to overwrite existing *filename*
Graph	
graph[(*graphopts*)]	graph results; see [PSS-2] **power, graph**
Iteration	
init(#)	initial odds ratio for effect-size calculation
iterate(#)	maximum number of iterations; default is iterate(500)
tolerance(#)	parameter tolerance; default is tolerance(1e-12)
ftolerance(#)	function tolerance; default is ftolerance(1e-12)
[no]log	suppress or display iteration log
[no]dots	suppress or display iterations as dots
notitle	suppress the title

†Either px() or oddsx() is required, and either pz() or oddsz() is required.

*Specifying a list of values in at least two starred options, or in argument *oratio*$_X$ and at least one starred option, results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see parallel.

collect is allowed; see **[U] 11.1.10 Prefix commands**.

notitle does not appear in the dialog box.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
alpha	significance level	α
power	power	$1-\beta$
beta	type-II-error probability	β
N	number of subjects	N
delta	effect size	δ
oratiox	odds ratio for X	OR_X
coefx	coefficient for X	β_X
px	success probability of X, $\Pr(X=1)$	p_X
oddsx	odds of $X=1$	$p_X/(1-p_X)$
oratioz	odds ratio for Z	OR_Z
coefz	coefficient for Z	ζ_1
pz	success probability of Z, $\Pr(Z=1)$	p_Z
oddsz	odds of $Z=1$	$p_Z/(1-p_Z)$
corrxz	correlation between X and Z	ϕ
py	success probability of Y, $\Pr(Y=1)$	p_Y
pycondx1z1	success probability of Y given X and Z are 1, $\Pr(Y=1\|X=1,Z=1)$	$p_{Y\|X=1,Z=1}$
pycondx1z0	success probability of Y given X is 1 and Z is 0, $\Pr(Y=1\|X=1,Z=0)$	$p_{Y\|X=1,Z=0}$
pycondx0z1	success probability of Y given X is 0 and Z is 1, $\Pr(Y=1\|X=0,Z=1)$	$p_{Y\|X=0,Z=1}$
pycondx0z0	success probability of Y given X and Z are 0, $\Pr(Y=1\|X=0,Z=0)$	$p_{Y\|X=0,Z=0}$
intercept	intercept	ζ_0
target	target parameter; odds ratio or coefficient for X	
_all	display all supported columns	

Options

Main

`alpha()`, `power()`, `beta()`, `n()`, `nfractional`; see [PSS-2] **power**. The `nfractional` option is allowed only for sample-size determination.

`coefx(`*numlist*`)` specifies the coefficient for binary covariate X in the logistic regression, β_X, where $\beta_X \neq 0$. `coefx()` may be specified instead of argument *oratio*$_X$. This option is not allowed with effect-size determination.

`effect(oratio | coefficient)` specifies how to report the effect size in the output. If `coefx()` is specified, then by default the effect size is β_X, the coefficient for X. Otherwise, the default effect size is the odds ratio for X: $\Pr(Y=1|X=1)\Pr(Y=0|X=0)/\{\Pr(Y=0|X=1)\Pr(Y=1|X=0)\}$. The `effect()` option is used to override the default.

`px(`*numlist*`)` specifies the success probability of binary covariate X in the target population, $\Pr(X=1)$, where $0<\Pr(X=1)<1$. Either the `px()` or `oddsx()` option must be specified, but not both.

`oddsx(`*numlist*`)` specifies the odds of binary covariate X in the target population, $\Pr(X=1)/\Pr(X=0)$, where $\Pr(X=1)/\Pr(X=0)>0$. Either the `px()` or `oddsx()` option must be specified, but not both.

`oratioz(`*numlist*`)` specifies the odds ratio for the effect of Z in the logistic regression, OR_Z, where $\text{OR}_Z>0$. The `oratioz()` option may be specified instead of the `coefz()` option.

`coefz(`*numlist*`)` specifies the coefficient for Z in the logistic regression, ζ_1. The `coefz()` option may be specified instead of the `oratioz()` option.

`pz(`*numlist*`)` specifies the success probability of Z in the target population, $\Pr(Z=1)$, where $0<\Pr(Z=1)<1$. Either the `pz()` or `oddsz()` option must be specified, but not both.

`oddsz(`*numlist*`)` specifies the odds of Z in the target population, $\Pr(Z=1)/\Pr(Z=0)$, where $\Pr(Z=1)/\Pr(Z=0)>0$. Either the `pz()` or `oddsz()` option must be specified, but not both.

`corrxz(`*numlist*`)` specifies the correlation between binary covariates X and Z, labeled ϕ, where $-1<\phi<1$. This value, called the ϕ coefficient, is a special case of Pearson's correlation coefficient between two binary variables. The default is `corrxz(0)`, which indicates no correlation between X and Z.

`py(`*numlist*`)` specifies the marginal success probability of Y in the target population, $\Pr(Y=1)$, where $0<\Pr(Y=1)<1$. This option is not allowed with effect-size determination.

`pycondx1z1(`*numlist*`)` specifies the conditional success probability of Y given X equals 1 and Z equals 1, $\Pr(Y=1|X=1,Z=1)$, where $0<\Pr(Y=1|X=1,Z=1)<1$. This option is not allowed with effect-size determination.

`pycondx1z0(`*numlist*`)` specifies the conditional success probability of Y given X equals 1 and Z equals 0, $\Pr(Y=1|X=1,Z=0)$, where $0<\Pr(Y=1|X=1,Z=0)<1$. This option is not allowed with effect-size determination.

`pycondx0z1(`*numlist*`)` specifies the conditional success probability of Y given X equals 0 and Z equals 1, $\Pr(Y=1|X=0,Z=1)$, where $0<\Pr(Y=1|X=0,Z=1)<1$.

`pycondx0z0(`*numlist*`)` specifies the conditional success probability of Y given X equals 0 and Z equals 0, $\Pr(Y=1|X=0,Z=0)$, where $0<\Pr(Y=1|X=0,Z=0)<1$. Only one of the `intercept()` or `pycondx0z0()` option may be specified.

`intercept(`*numlist*`)` specifies the intercept for the logistic regression, ζ_0. Only one of the `intercept()` or `pycondx0z0()` option may be specified.

`direction()`, `parallel`; see [PSS-2] **power**.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies the initial value of the odds ratio for binary covariate X during effect-size determination.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following option is available with `power logistic` but is not shown in the dialog box:

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `power logistic` command and the methodology for power and sample-size analysis for logistic regression with two binary covariates. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis, and see [PSS-2] **power** for a general introduction to using the `power` command for hypothesis tests.

Introduction

Logistic regression is a commonly used statistical method for analyzing binary outcome variables. Researchers often need to determine the appropriate sample size to ensure sufficient power for detecting the association between a covariate of interest (X) and a binary outcome variable (Y) while controlling for the effect of a nuisance variable (Z).

For example, consider a study that examines factors influencing whether migratory birds return to the same nesting site from one year to the next. Binary outcome Y is an indicator of whether the nesting site was reused, where the observed $y_i = 1$ if bird i returns to the nesting site it used last year and $y_i = 0$ if bird i does not. We will use logistic regression to test whether birds of one sex are more likely to return to the same nesting site than birds of the other sex. We define binary covariate of interest X as an indicator for female birds, where the observed $x_i = 1$ if bird i is female and $x_i = 0$ if bird i is male.

Previous observations suggest that birds that mate with the same partner as last year are more likely to return to the same nesting site. We are not interested in studying the effect of a bird's mate, but it would be foolhardy to ignore this effect. We include mate in our logistic regression as a nuisance covariate Z, where the observed $z_i = 1$ if bird i has the same mate as last year and $z_i = 0$ if not.

The logistic regression can be written as

$$\Pr(y_i = 1|x_i, z_i) = H(\beta_X x_i + \zeta_0 + \zeta_Z z_i) \qquad i = 1, 2, \ldots, n$$

where β_X is the coefficient quantifying the effect of a covariate X, bird's sex, on nesting-site reuse; ζ_0 is the logistic intercept; ζ_Z is the effect of a covariate Z, partnering with the same mate; and n is the sample size. Function $H(\eta) = \{1 + \exp(-\eta)\}^{-1}$ is the logistic distribution function. The effect of X can also be expressed in terms of an odds ratio,

$$\text{OR}_X = \exp(\beta_X) = \Pr(Y = 1|X = 1)\Pr(Y = 0|X = 0)/\{\Pr(Y = 0|X = 1)\Pr(Y = 1|X = 0)\}$$

The null hypothesis is $H_0 : \beta_X = 0$, which can also be expressed as $H_0 : \text{OR}_X = 1$, and the alternative hypothesis is $H_a : \beta_X \neq 0$ or, equivalently, $H_a : \text{OR}_X \neq 1$.

Logistic regression is commonly used to analyze binary outcomes in observational studies, such as the study of nesting-site reuse. But logistic regression can also be used to analyze data from a randomized controlled trial, where trial participants are randomly assigned to a treatment. For instance, a public health study might investigate whether attending a support group helps male and female smokers quit smoking. In this example, participation in the support group is binary covariate X: Some participants will be randomly assigned to attend support group meetings for, say, three months, whereas other participants will be randomized to the control group, which does not attend support group meetings. At the end of three months, smoking status is recorded; this is binary outcome Y. Based on previous research, we anticipate that males and females will have different success rates at quitting smoking. We are not interested in studying the effect of sex on smoking cessation, but we will include sex in our logistic regression as nuisance covariate Z.

The `power logistic` command provides power and sample-size analysis for the test of $\beta_X = 0$ in logistic regression. The formula for power, sample-size, and effect-size calculations is based on the likelihood-ratio test of $\beta_X = 0$. However, Bush (2015) demonstrates that sample-size requirements for Wald and score tests are generally equivalent to the sample-size requirement for the likelihood-ratio test. Thus, these calculations may be used to plan studies that will be analyzed using the `logit` command, which conducts a Wald test of $\beta_X = 0$, or the `logistic` command, which conducts an equivalent Wald test of $\text{OR}_X = 1$. If you prefer a likelihood-ratio test, you can use the `lrtest` command.

Using power logistic with two binary covariates

`power logistic` computes sample size, power, or effect size for a test of one coefficient in logistic regression. This entry describes how to use `power logistic` when the logistic regression has two binary covariates, one of which is of interest and one of which is a nuisance variable. All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option.

You must specify either the prevalence of a binary covariate X in the target population in the `px()` option or the odds of X in the `oddsx()` option. The `px()` option specifies $\Pr(X = 1)$, whereas the `oddsx()` option specifies $\Pr(X = 1)/\Pr(X = 0)$. When you collect observational data, $\Pr(X = 1)$ indicates the prevalence of X in the target population. In the observational study about nesting-site reuse,

$\Pr(X = 1)$ is the proportion of female birds in the population. But when you collect data from a randomized controlled trial, the "target population" is restricted to study participants, and the meaning of $\Pr(X = 1)$ changes accordingly. In the smoking-cessation trial, $\Pr(X = 1)$ is the proportion of participants who are randomly assigned to the support group. The ratio of participants in the experimental arm to the control arm is known as the allocation ratio. Additionally, you must specify either the prevalence of a nuisance binary covariate Z, $\Pr(Z = 1)$, in the `pz()` option or the odds of Z, $\Pr(Z = 1)/\Pr(Z = 0)$, in the `oddsz()` option.

Power and sample-size calculations require three parameters from the logistic regression: the X coefficient (β_X), the Z coefficient (ζ_1), and the intercept (ζ_0). The X coefficient can be specified directly in the `coefx()` option or indirectly as an argument *oratio*$_X$ or in the `pycondx1z0()`, `pycondx1z1()`, or `py()` option. The Z coefficient can be specified directly in the `coefz()` option or indirectly in the `oratioz()`, `pycondx0z1()`, `pycondx1z1()`, or `py()` option. The intercept can be specified directly in the `intercept()` option or indirectly in the `py()`, `pycondx1z1()`, `pycondx1z0()`, `pycondx0z1()`, or `pycondx0z0()` option. See figure 1 for a graphical depiction of various ways to provide information about the logistic regression parameters.

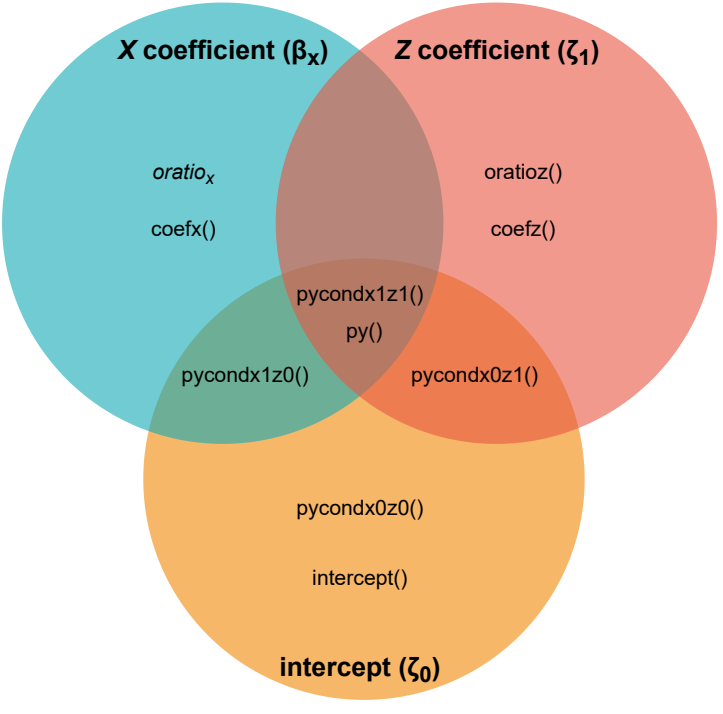


Figure 1. Specifying information about β_X, ζ_1, and ζ_0

For example, valid specifications for power and sample-size calculations include `coefx()`, `coefz()`, and `intercept()`, as well as `py()`, `pycondx1z1()`, and `pycondx0z1()`. However, the combination of *oratio*$_X$, `pycondx1z0()`, and `pycondx0z0()` is invalid because no information is provided about the Z coefficient. Additionally, the combination of `pycondx1z1()` and `pycondx0z1()` may not be specified together with *oratio*$_X$ or `coefx()`; the combination of `pycondx1z1()` and `pycondx1z0()` may not be specified together with `oratioz()` or `coefz()`; and the combination of `pycondx1z0()` and `pycondx0z1()` may not be specified together with `py()`.

When you compute sample size, the power of the test may be specified using the `power()` option, which has a default of 0.8. When you compute power, the sample size must be specified using the `n()` option.

When `power logistic` is used to calculate effect size β_X, the command specification cannot include *oratio*$_X$ or options that provide information about the X coefficient, such as the `py()` and `pycondx1z0()` options. In this case, only two options are required that provide information about the Z coefficient and intercept, such as `oratioz()` and `intercept()` or `pycondx0z1()` and `pycondx0z0()`.

To calculate effect size, you must specify sample size using the `n()` option, power using the `power()` option, and, optionally, the direction of the effect using the `direction()` option. The default is `direction(upper)`, which means that coefficient β_X is assumed to be positive. This is equivalent to assuming that the odds ratio OR_X is greater than 1. You can change the direction to `lower`, which means that $\beta_X < 0$ or, equivalently, $\text{OR}_X < 1$.

The `effect()` option can be used to specify the type of the effect size to be reported in the output. Valid choices are `effect(oratio)` and `effect(coefficient)`. By default, the effect size is output as the odds ratio for X unless `coefx()` is specified, in which case it defaults to the coefficient for X. The `effect()` option is used to override the default.

By default, the computed sample size is rounded up. You can specify the `nfractional` option to see the corresponding fractional sample size; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. The `nfractional` option is allowed only for sample-size determination.

Some of the computations of `power logistic` require iteration, specifically, the computations used in effect-size determination. The default initial value of the estimated effect size is obtained using a bisection algorithm. This may be changed by specifying the `init()` option. See [PSS-2] **power** for descriptions of other options that control the iteration procedure.

In the following sections, we describe the use of `power logistic` with two binary covariates accompanied by examples for computing sample size, power, and effect size.

Computing sample size

To compute sample size, you must specify the prevalence or odds of X in the target population in the respective `px()` or `oddsx()` option; the prevalence or odds of Z in the respective `pz()` or `oddsz()` option; the information necessary to determine parameters β_X, ζ_1, and ζ_0, as described in *Using power logistic with two binary covariates*; and, optionally, the power of the test using the `power()` option or the type-II-error probability using the `beta()` option. A default power of 0.8 is assumed if `power()` is not specified.

▷ Example 1: Sample size when OR_X, OR_Z, and $\Pr(Y|X=0,Z=0)$ are known

Consider an example from the seminal work on sample-size calculation for logistic regression by Whittemore (1981, 31) that describes the design of a study testing "the null hypothesis that risk of coronary heart disease (CHD) among white males aged 39–59 is unaffected by serum cholesterol levels". Here the X variable is an indicator for elevated serum cholesterol, where the observed $x_i = 1$ if participant i has elevated serum cholesterol and $x_i = 0$ otherwise. We want to control for the effects of triglyceride, which is a known risk factor for CHD. Our Z variable is an indicator of elevated triglyceride levels, where the observed $z_i = 1$ if participant i has elevated triglycerides and $z_i = 0$ otherwise.

The power logistic command will be used to conduct sample-size calculations for this scenario. We assume that 13% of our target population has elevated serum cholesterol, which we will enter as px(0.13), and 22% of participants have elevated triglycerides, corresponding to pz(0.22). The correlation (phi coefficient) between X and Z is estimated to be 0.4, which we specify as corrxz(0.4). Previous research has indicated that elevated triglycerides are associated with an odds ratio of 1.25, so we specify oratioz(1.25). Based on data from Hulley et al. (1980), we assume a probability of 0.07 that an individual in the target population will develop CHD during an 18-month study period if he does not have elevated serum cholesterol or elevated triglycerides; this will be entered as pycondx0z0(0.07).

Like Whittemore, we will calculate the sample size required to detect an odds ratio of 1.65 at the $\alpha = 0.05$ level, but we will use the default power of 80%.

```
. power logistic 1.65, px(0.13) pz(0.22) corrxz(0.4) oratioz(1.25) pycondx0z0(0.07)

Estimated sample size for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    1.6500  (odds ratio)
      oratiox =    1.6500
           px =    0.1300
      oratioz =    1.2500
           pz =    0.2200
       corrxz =    0.4000
   pycondx0z0 =    0.0700

Estimated sample size:

            N =     3,718
```

The output of power logistic begins by displaying information about the test to be conducted and its null and alternative hypotheses. The study parameters we specified are listed next, followed by the estimated sample size. We find that a sample of 3,718 subjects is required to detect an odds ratio of 1.65 with 80% power using a 5% level test.

◁

▷ Example 2: Sample size when β_X, OR_Z, and $\Pr(Y|X=0,Z=0)$ are known

Instead of the odds ratio for X, as in example 1, we specify the effect as $\beta_X = \log(1.65) = 0.5008$. We leave the rest of the command specification unchanged.

```
. power logistic, px(0.13) pz(0.22) corrxz(0.4) coefx(0.5008) oratioz(1.25)
> pycondx0z0(0.07)

Estimated sample size for logistic regression coefficient test
Likelihood-ratio test
H0: beta_X = 0  versus  Ha: beta_X != 0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.5008  (coefficient)
        coefx =    0.5008
           px =    0.1300
      oratioz =    1.2500
           pz =    0.2200
       corrxz =    0.4000
   pycondx0z0 =    0.0700

Estimated sample size:

            N =     3,718
```

The output of this command is identical to the output displayed in example 1, with the exception that the coefficient for X, `coefx`, is displayed instead of the odds ratio. If we wanted to see the odds ratio in the output despite inputting the coefficient for X, we could add the `effect(oratio)` option.

◁

▷ Example 3: Sample size when ζ_0, $\Pr(Y)$, and $\Pr(Y|X=1,Z=1)$ are known

Now we assume that we know $\Pr(Y=1)$, the probability that a randomly selected study participant develops CHD. Recall from example 1 that the chance of observing CHD in an individual without elevated serum cholesterol or triglycerides was 7%; once we include individuals with elevated cholesterol, triglycerides, or both, the overall chance of CHD rises to 7.9447%. The group at the highest risk of CHD is made up of participants whose serum cholesterol and triglycerides are both elevated, and they stand a 13.438% chance of developing CHD during the follow-up period. For demonstration purposes, instead of specifying `pycondx0z0()`, we specify the intercept directly as $\zeta_0 = \texttt{logit(0.07)} = -2.5867$.

```
. power logistic, px(0.13) pz(0.22) corrxz(0.4) py(0.079447)
> pycondx1z1(0.13438) intercept(-2.5867)

Estimated sample size for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    1.6500  (odds ratio)
      oratiox =    1.6500
           px =    0.1300
           pz =    0.2200
       corrxz =    0.4000
           py =    0.0794
   pycondx1z1 =    0.1344
    intercept =   -2.5867

Estimated sample size:

            N =     3,718
```

The output of this command is identical to the output displayed in example 1, with the exception that study parameters py, pycondx1z1, and intercept are displayed instead of oratioz and pycondx0z0. If we wanted to see coefx in the output instead of oratiox, we could add the effect(coefficient) option.

◁

Computing power

To compute power, you must specify the prevalence or odds of X in the target population using the respective px() or oddsx() option; the prevalence or odds of Z using the respective pz() or oddsz() option; the sample size using the n() option; and the information necessary to determine parameters β_X, ζ_1, and ζ_0, as described in *Using power logistic with two binary covariates*.

▷ Example 4: Power of a logistic regression odds-ratio test

Continuing with example 1, we will suppose that we are designing a new study and anticipate a sample of 4,000 subjects. To compute the power corresponding to this sample size, we specify the same study parameters we used in example 1, but now we use the n() option to specify a sample size of 4,000.

```
. power logistic 1.65, px(0.13) pz(0.22) n(4000) corrxz(0.4) oratioz(1.25)
> pycondx0z0(0.07)

Estimated power for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1

Study parameters:

        alpha =    0.0500
            N =     4,000
        delta =    1.6500  (odds ratio)
      oratiox =    1.6500
           px =    0.1300
      oratioz =    1.2500
           pz =    0.2200
       corrxz =    0.4000
   pycondx0z0 =    0.0700

Estimated power:

        power =    0.8279
```

If the study recruits 4,000 participants, the power to detect an odds ratio of 1.65 climbs to 82.79%.

◁

▷ Example 5: Multiple values of study parameters

To investigate the effect of sample size on power, we can specify a list of sample sizes in the `n()` option:

```
. power logistic 1.65, px(0.13) pz(0.22) n(3000 3500 4000 4500 5000)
> corrxz(0.4) oratioz(1.25) pycondx0z0(0.07)
Estimated power for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1
```

alpha	power	N	delta	oratiox	px	oratioz	pz	corrxz
.05	.7111	3,000	1.65	1.65	.13	1.25	.22	.4
.05	.7759	3,500	1.65	1.65	.13	1.25	.22	.4
.05	.8279	4,000	1.65	1.65	.13	1.25	.22	.4
.05	.8691	4,500	1.65	1.65	.13	1.25	.22	.4
.05	.9013	5,000	1.65	1.65	.13	1.25	.22	.4

pycondx0z0
.07
.07
.07
.07
.07

As expected, when the sample size increases, the power increases toward 1.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

Computing effect size

By default, effect size δ for a logistic regression odds-ratio test is defined as the odds ratio for X: $\delta = \text{OR}_X = \exp(\beta_X)$. Sometimes, we want to know the smallest effect that can be detected with a level α test at a prespecified power and sample size.

To compute the minimum detectable effect size, you must specify the prevalence or odds of X in the target population using the respective `px()` or `oddsx()` option; the prevalence or odds of Z using the respective `pz()` or `oddsz()` option; the information necessary to determine parameters ζ_1 and ζ_0, as described in *Using power logistic with two binary covariates*; the sample size using the `n()` option; and the power of the test using the `power()` option or the type-II-error probability in the `beta()` option. In addition, you must pick the level of the test and the direction of the effect. The level of the test is specified using the `alpha()` option, with a default of `alpha(0.05)`. The direction of the effect is specified using the `direction()` option; the default is `direction(upper)`, which means that $\text{OR}_X > 1$ or, equivalently, $\beta_X > 0$. Specifying `direction(lower)` means that $\text{OR}_X < 1$ and $\beta_X < 0$. The estimated minimum detectable effect size is reported as an odds ratio by default. To display it as a coefficient, specify `effect(coefficient)`.

Example 6: Minimum detectable odds ratio

We continue with example 4, where we learned that a size 0.05 test with 4,000 subjects would have 82.79% power to detect an odds ratio of 1.65. How much larger would the odds ratio need to be to detect it with 90% power? We use `power logistic` to find out.

```
. power logistic, px(0.13) pz(0.22) n(4000) power(0.9) corrxz(0.4)
> oratioz(1.25) pycondx0z0(0.07)

Performing iteration ...

Estimated odds ratio for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1

Study parameters:

        alpha =    0.0500
        power =    0.9000
            N =     4,000
           px =    0.1300
      oratioz =    1.2500
           pz =    0.2200
       corrxz =    0.4000
   pycondx0z0 =    0.0700

Estimated effect size and odds ratio:

        delta =    1.7356  (odds ratio)
      oratiox =    1.7356
```

We see that a slightly larger odds ratio of 1.7356 can be detected with 90% power.

In the above, we assumed the effect to be in the upper direction. By symmetry, there exists an effect size in the lower direction that can also be detected with 90% power. We specify `direction(lower)` to find it, and we add the `effect(coefficient)` option to display it as a coefficient instead of an odds ratio.

```
. power logistic, px(0.13) pz(0.22) n(4000) power(0.9) corrxz(0.4)
> oratioz(1.25) pycondx0z0(0.07) direction(lower)
> effect(coefficient)

Performing iteration ...

Estimated coefficient for logistic regression coefficient test
Likelihood-ratio test
H0: beta_X = 0  versus  Ha: beta_X != 0

Study parameters:

        alpha =    0.0500
        power =    0.9000
            N =     4,000
           px =    0.1300
      oratioz =    1.2500
           pz =    0.2200
       corrxz =    0.4000
   pycondx0z0 =    0.0700

Estimated effect size and coefficient:

        delta =   -0.7822  (coefficient)
        coefx =   -0.7822
```

By specifying `direction(lower)`, we anticipate coefficient $\beta_X < 0$, which is what we see. Had we omitted the `effect(coefficient)` option, the odds ratio $\text{OR}_X = \exp(\beta_X)$ would have been displayed as $\exp(-0.7822) = 0.457$.

Performing hypothesis tests with logistic regression

In this section, we briefly demonstrate the use of the `logit` command for testing logistic regression coefficients; see [R] **logit** for details. Alternatively, we could use the `logistic` command to perform logistic regression because `logistic` performs the same calculations as `logit` but reports odds ratios instead of coefficients; see [R] **logistic** for details, and see example 7 of [PSS-2] **power logistic onebin** for a demonstration of how `logistic` can be used to analyze the results of a pilot study.

▷ Example 7: Analyzing a pilot study

`nlsw88.dta` contains employment data from the 1988 extract of the National Longitudinal Study of Young Women. We will treat this dataset as if it came from a pilot study investigating the relationship between union membership and marital status and use it to plan a follow-up study. Our target population is American young women, some of whom are college graduates, so we will include college education as a nuisance covariate in our logistic regression.

```
. use https://www.stata-press.com/data/r19/nlsw88
(NLSW, 1988 extract)
. logit union married collgrad, nolog
Logistic regression                                     Number of obs =  1,878
                                                        LR chi2(2)    =  22.24
                                                        Prob > chi2   = 0.0000
Log likelihood = -1035.5051                             Pseudo R2     = 0.0106

------------------------------------------------------------------------------
       union | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
     married |  -.2484434   .1112893    -2.23   0.026    -.4665663   -.0303204
    collgrad |   .4992676   .1186018     4.21   0.000     .2668124    .7317229
       _cons |  -1.099668   .0942637   -11.67   0.000    -1.284421   -.9149144
------------------------------------------------------------------------------
```

Marital status is our covariate of interest X, so the output tells us that β_X, the coefficient for `married`, is -0.25. This suggests that married women are less likely to be union members than unmarried women, but the evidence is not strong enough to reject $H_0 : \beta_X = 0$ at the 0.01 level. We want to design a follow-up study that has 90% power to detect a coefficient of -0.25 with a 0.01-level test, and we will use the parameter estimates from our pilot study to do so.

We will use `power logistic` with effect size `coefx(-0.25)`, but what about the other parameters from the logistic regression? We also need to specify information about the effect of nuisance covariate Z, the logistic intercept, and the population prevalences of X and Z.

The logistic regression estimates the coefficient for `collgrad` to be 0.5, which we will specify using the `coefz()` option. The logistic intercept, reported as `_cons`, will be specified as `intercept(-1.1)`. To find the population prevalences of X and Z, we use the `tabulate` command to calculate the prevalence of marriage and college education in the target population; see [R] **tabulate twoway** for details. We include the expression `if e(sample)` to restrict the tabulation so that it includes only the 1,878 participants whose data were used to fit the logistic regression; see [U] **20.7 Specifying the estimation subsample** for details about `e(sample)`.

```
. tabulate collgrad married if e(sample)
```

College graduate	Married: Single	Married	Total
Not college grad	492	922	1,414
College grad	164	300	464
Total	656	1,222	1,878

Married women accounted for 1,222 of the 1,878 participants, or approximately 65% of the target population, while college graduates made up 464 of the 1,878 participants, or approximately 25% of the population. We specify these parameters using the `px()` and `pz()` options, respectively.

```
. power logistic, px(0.65) pz(0.25) power(0.9) alpha(0.01)
> coefx(-0.25) coefz(0.5) intercept(-1.1)

Estimated sample size for logistic regression coefficient test
Likelihood-ratio test
H0: beta_X = 0  versus  Ha: beta_X != 0

Study parameters:

        alpha =    0.0100
        power =    0.9000
        delta =   -0.2500  (coefficient)
        coefx =   -0.2500
           px =    0.6500
        coefz =    0.5000
           pz =    0.2500
    intercept =   -1.1000

Estimated sample size:

            N =     5,578
```

We find that a sample size of 5,578 is required to detect a β_X of -0.25 with 90% power using a 1% level test. One detail that bears mentioning is that this sample-size calculation is for a likelihood-ratio test, but the `logit` and `logistic` commands report Wald tests of coefficients. Fortunately, an extensive simulation study by Bush (2015) demonstrates that sample-size requirements for Wald and likelihood-ratio tests of logistic regression coefficients are nearly identical. If you prefer a likelihood-ratio test, you can use the `lrtest` command; see [R] **lrtest** for details.

◁

Stored results

`power logistic` with two binary covariates stores the following in `r()`:

Scalars

`r(alpha)`	significance level
`r(power)`	power
`r(beta)`	probability of a type II error
`r(delta)`	effect size
`r(N)`	sample size
`r(nfractional)`	1 if `nfractional` is specified, 0 otherwise
`r(px)`	success probability of X
`r(oddsx)`	odds that $X = 1$
`r(pz)`	success probability of Z
`r(oddsz)`	odds that $Z = 1$
`r(corrxz)`	correlation between X and Z
`r(oratiox)`	odds ratio for X
`r(coefx)`	coefficient for X

r(oratioz)	odds ratio for Z
r(coefz)	coefficient for Z
r(intercept)	intercept from logistic regression
r(py)	probability of Y in the target population
r(pycondx1z1)	success probability of Y given $X = 1$ and $Z = 1$
r(pycondx1z0)	success probability of Y given $X = 1$ and $Z = 0$
r(pycondx0z1)	success probability of Y given $X = 0$ and $Z = 1$
r(pycondx0z0)	success probability of Y given $X = 0$ and $Z = 0$
r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise
r(init)	initial value for odds ratio (if specified)
r(maxiter)	maximum number of iterations (for effect-size calculation)
r(iter)	number of iterations performed (for effect-size calculation)
r(tolerance)	requested parameter tolerance (for effect-size calculation)
r(deltax)	final parameter tolerance achieved (for effect-size calculation)
r(ftolerance)	requested distance of the objective function from zero (for effect-size calculation)
r(function)	final distance of the objective function from zero (for effect-size calculation)
r(converged)	1 if iteration algorithm converged, 0 otherwise (for effect-size calculation)

Macros

r(type)	test
r(method)	logistic
r(direction)	upper or lower (for effect-size calculation)
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results

Methods and formulas

Methods and formulas are presented under the following headings:

Coefficient tests in logistic regression

Shieh (2000a) used simulation to compare the performance of two sample-size formulas for coefficient tests in logistic regression: the method of Whittemore (1981) and that of Self, Mauritsen, and Ohara (1992). Shieh generalized the superior of those two methods, that of Self, Mauritsen, and Ohara, in Shieh (2000b), which provides the formulas implemented in `power logistic` for power, sample-size, and effect-size calculations for likelihood-ratio tests in logistic regression.

In practice, it is more common to use the Wald test of logistic regression coefficients than the likelihood-ratio test, so there has been some concern about whether these calculations are appropriate for use with a Wald test (Demidenko 2007). Demidenko notes that the Wald and likelihood-ratio tests have asymptotically equivalent type I errors and that they are "*locally* equivalent, so that the power functions are close when the alternative approaches the null" (Demidenko 2007, 3385). Nevertheless, Demidenko raises the point that the two tests are not globally equivalent, so there is no theoretical guarantee that the power functions will be similar under the alternative hypothesis. Thankfully, an extensive simulation study by Bush (2015) found little difference between the power curves of the Wald, likelihood-ratio, and

score tests over a range of scenarios. Additionally, Bush compared the performance of seven sample-size formulas for logistic regression and determined that the method of Shieh (2000b) was consistently accurate, regardless of the test that was used.

Logistic regression

The logistic regression with two binary covariates can be written as

$$\Pr(y_i = 1|x_i, z_i) = H(\beta_X x_i + \zeta_0 + \zeta_1 z_i) \qquad i = 1, 2, \ldots, n$$

where β_X is the coefficient for binary covariate of interest X, ζ_0 is the logistic intercept, ζ_1 is the coefficient for nuisance binary covariate Z, and n is the sample size. Function $H(\eta) = \{1 + \exp(-\eta)\}^{-1}$ is the logistic distribution function.

The effects of binary covariates X and Z can also be expressed in terms of odds ratios:

$$\text{OR}_X = \exp(\beta_X) = \Pr(Y = 1|X = 1)\Pr(Y = 0|X = 0)/\{\Pr(Y = 0|X = 1)\Pr(Y = 1|X = 0)\}$$

$$\text{OR}_Z = \exp(\zeta_1) = \Pr(Y = 1|Z = 1)\Pr(Y = 0|Z = 0)/\{\Pr(Y = 0|Z = 1)\Pr(Y = 1|Z = 0)\}$$

The null hypothesis is $H_0 : \beta_X = 0$, which can also be expressed as $H_0 : \text{OR}_X = 1$. The alternative hypothesis is $H_a : \beta_X \neq 0$ or, equivalently, $H_a : \text{OR}_X \neq 1$.

Parameters β_X, ζ_1, and ζ_0 may be specified directly with the `coefx()`, `coefz()`, and `intercept()` options, respectively. Alternatively, information about β_X, ζ_1, and ζ_0 may be specified by providing the marginal or conditional success probability of Y using one or more of the following options: `py()`, `pycondx1z1()`, `pycondx1z0()`, `pycondx0z1()`, and `pycondx0z0()`. If the marginal or conditional success probability of Y is specified, we solve for the unknown parameters β_X, ζ_1, and ζ_0 using the following equations:

$$\begin{aligned}
\Pr(Y = 1|X = 1, Z = 1) &= H(\beta_X + \zeta_0 + \zeta_1) \\
\Pr(Y = 1|X = 1, Z = 0) &= H(\beta_X + \zeta_0) \\
\Pr(Y = 1|X = 0, Z = 1) &= H(\zeta_1 + \zeta_0) \\
\Pr(Y = 1|X = 0, Z = 0) &= H(\zeta_0) \\
\Pr(Y = 1) &= \Pr(X = 1, Z = 1) \times \Pr(Y = 1|X = 1, Z = 1) \\
&\quad + \Pr(X = 1, Z = 0) \times \Pr(Y = 1|X = 1, Z = 0) \\
&\quad + \Pr(X = 0, Z = 1) \times \Pr(Y = 1|X = 0, Z = 1) \\
&\quad + \Pr(X = 0, Z = 0) \times \Pr(Y = 1|X = 0, Z = 0)
\end{aligned}$$

Power, sample-size, and effect-size calculations

Shieh (2000b) builds on the work of Self, Mauritsen, and Ohara (1992) and Self and Mauritsen (1988) to estimate the distribution of the likelihood-ratio statistic: $2\{l(\hat{\beta}_X, \hat{\mathbf{c}}) - l(0, \hat{\mathbf{c}}_0)\}$. Here $l(\cdot)$ is the log-likelihood function for the logistic regression, and $\mathbf{c} = (\zeta_0, \zeta_1)$ is a vector of nuisance parameters. $\hat{\beta}_X$ and $\hat{\mathbf{c}}$ are the maximum likelihood estimates of β_X and $\mathbf{c}$ under the alternative hypothesis, and $\hat{\mathbf{c}}_0$ is the maximum likelihood estimate of $\mathbf{c}$ under the null hypothesis. If the null hypothesis is not true, $\hat{\mathbf{c}}_0$ is not a consistent estimate of $\mathbf{c}$ but instead converges to $\mathbf{c}_0^* = (\zeta_0^*, \zeta_1)$, where $\zeta_0^* = \zeta_0 + \beta_X E(X)$, as described in Self and Mauritsen (1988, eq. 2.2). For a full decomposition of the likelihood-ratio statistic, see Shieh (2000b, 1193).

To calculate sample size, we begin by specifying the desired size of the test (also known as the type I error, α) and the desired power (which equals $1-\beta$, where β is the desired type II error of the test). We equate the $(1-\alpha)100$th percentile of a central χ^2 distribution with 1 degree of freedom to the $\beta100$th percentile of a noncentral χ^2 distribution with noncentrality parameter $\lambda = n\Delta^*$. We define Δ^* as

$$\Delta^* = 2E[H(\eta)(\eta-\eta^*) - \log\{1+\exp(\eta)\} + \log\{1+\exp(\eta^*)\}]$$

where $\eta = \beta_X X + \mathbf{cZ}$, $\eta^* = \mathbf{c}_0^*\mathbf{Z}$, and $\mathbf{Z} = (1, Z)'$. Sample size is calculated as $n = \lambda/\Delta^*(1-\phi^2)$, where ϕ is the phi coefficient, which is Pearson's correlation coefficient between two binary variables: $\phi = \mathrm{cov}(X,Z)/\{\mathrm{sd}(X)\mathrm{sd}(Z)\}$. Power is computed similarly by starting with a known value of n and solving for the power required to yield the desired value of λ.

There is no closed-form expression that can be used to calculate effect size δ, either coefficient β_X or odds ratio OR_X. Effect size is estimated iteratively, and the default starting value for δ is calculated using a bisection algorithm, which is refined using a nonlinear solver; see [M-5] **solvenl( )** for details. To skip the bisection algorithm, you can use the `init()` option to specify a starting odds ratio for the nonlinear solver. You can control the iteration process with the `iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, and `nodots` options.

References

Bush, S. 2015. Sample size determination for logistic regression: A simulation study. *Communications in Statistics—Simulation and Computation* 44: 360–373. https://doi.org/10.1080/03610918.2013.777458.

Demidenko, E. 2007. Sample size determination for logistic regression revisited. *Statistics in Medicine* 26: 3385–3397. https://doi.org/10.1002/sim.2771.

Hulley, S. B., R. H. Rosenman, R. D. Bawol, and R. J. Brand. 1980. Epidemiology as a guide to clinical decisions—the association between triglyceride and coronary heart disease. *New England Journal of Medicine* 302: 1383–1389. https://doi.org/10.1056/nejm198006193022503.

Self, S. G., and R. H. Mauritsen. 1988. Power/sample size calculations for generalized linear models. *Biometrika* 44: 79–86. https://doi.org/10.2307/2531897.

Self, S. G., R. H. Mauritsen, and J. Ohara. 1992. Power calculations for likelihood ratio tests in generalized linear models. *Biometrika* 48: 31–39. https://doi.org/10.2307/2532736.

Shieh, G. 2000a. A comparison of two approaches for power and sample size calculations in logistic regression models. *Communications in Statistics—Simulation and Computation* 29: 763–791. https://doi.org/10.1080/03610910008813639.

———. 2000b. On power and sample size calculations for likelihood ratio tests in generalized linear models. *Biometrics* 56: 1192–1196. https://doi.org/10.1111/j.0006-341x.2000.01192.x.

Whittemore, A. S. 1981. Sample size for logistic regression with small response probability. *Journal of the American Statistical Association* 76: 27–32. https://doi.org/10.2307/2287036.

Also see

[PSS-2] **power logistic** — Power analysis for logistic regression+

[PSS-2] **power logistic general** — Power analysis for logistic regression: General case+

[PSS-2] **power logistic onebin** — Power analysis for logistic regression with one binary covariate+

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[R] **logistic** — Logistic regression, reporting odds ratios

[R] **logit** — Logistic regression, reporting coefficients

[R] **lrtest** — Likelihood-ratio test after estimation

power logistic general — Power analysis for logistic regression: General case[+]

[+]This command is part of StataNow.

Description

`power logistic` computes sample size, power, or effect size for a test of one coefficient in logistic regression. This entry describes how to use `power logistic` to plan a study that will be modeled using logistic regression with one covariate of interest, X, and up to 20 nuisance covariates $\mathbf{Z} = (Z_1, Z_2, \ldots)$. Covariates X and $\mathbf{Z}$ can be continuous, discrete, or a combination of both. For information about how to use `power logistic` in the special cases of one or two binary covariates, see [PSS-2] **power logistic onebin** and [PSS-2] **power logistic twobin**, respectively.

By default, `power logistic` computes sample size for a given power and effect size, where the effect size may be specified as a coefficient or an odds ratio. Alternatively, it can compute power given sample size and effect size, or it can compute the effect size given power and sample size.

Quick start

Sample size for logistic regression with one binary covariate X, given a coefficient of 0.4055 for X under the alternative hypothesis H_a, population prevalence of X of 0.22, and an intercept of -2; using the default power of 0.8 and significance level $\alpha = 0.05$

```
power logistic, x(distribution(bernoulli 0.22) coefficient(0.4055))     ///
    intercept(-2)
```

Same as above, but use argument *oratio*$_X$ to specify the odds ratio for X of $\exp(0.4055) = 1.5$ instead of the coefficient

```
power logistic 1.5, x(distribution(bernoulli 0.22)) intercept(-2)
```

Same as above, but specify the success probability of Y conditional on the means of X and Z instead of argument *oratio*$_X$

```
power logistic, x(distribution(bernoulli 0.22)) intercept(-2)          ///
    pycondxmzm(0.128892)
```

Same as above, but specify different values for the prevalence of X and display the effect of X as a coefficient

```
power logistic, x(distribution(bernoulli (0.2 0.22 0.24 0.26)))        ///
    intercept(-2) pycondxmzm(0.128892) effect(coefficient)
```

Sample size for logistic regression with covariate of interest $X \sim$ normal(20, 5) and nuisance covariates $\mathbf{Z} = (Z_1, Z_2)$, where $Z_1 \sim$ exponential(12) and $Z_2 \sim$ binomial(3, 0.4), and given, under the alternative hypothesis H_a, an odds ratio of 1.8 for a 5-unit change in X, and odds ratios of 1.1 and 1.5 for 1-unit changes in Z_1 and Z_2; also, specify a correlation of 0.28 between X and $\mathbf{Z}$ and $\Pr\{Y = 1 | X = E(X), \mathbf{Z} = E(\mathbf{Z})\} = 0.56$

```
power logistic, x(distribution(normal 20 5) oratio(1.8, unit(5)))      ///
    z1(distribution(exponential 12) oratio(1.1))                       ///
    z2(distribution(binomial 3 0.4) oratio(1.5)) pycondxmzm(0.56) corrxz(0.28)
```

Same as above, but specify different standard deviations for X, and discretize Z_1 into 25 bins

```
power logistic, x(distribution(normal 20 (4 5 6)) oratio(1.8, unit(5)))  ///
   z1(distribution(exponential 12, nbins(25)) oratio(1.1))               ///
   z2(distribution(binomial 3 0.4) oratio(1.5)) pycondxmzm(0.56) corrxz(0.28)
```

Power given a sample size of 200 and previous values of other parameters

```
power logistic, x(distribution(normal 20 (4 5 6)) oratio(1.8, unit(5)))  ///
   z1(distribution(exponential 12, nbins(25)) oratio(1.1))               ///
   z2(distribution(binomial 3 0.4) oratio(1.5)) pycondxmzm(0.56)         ///
   corrxz(0.28) n(200)
```

Effect size given a sample of size 200, power of 90%, $\Pr\{Y = 1|X = 0, \mathbf{Z} = E(\mathbf{Z})\} = 0.108$, and previous values of other parameters

```
power logistic, x(distribution(normal 20 (4 5 6)))                       ///
   z1(distribution(exponential 12, nbins(25)) oratio(1.1))               ///
   z2(distribution(binomial 3 0.4) oratio(1.5)) pycondx0zm(0.108)        ///
   corrxz(0.28) n(200) power(0.9)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
power logistic oratio_X, x(xzspec) [z1(xzspec) [z2(xzspec) [...]]
                                   power(numlist) generalopts]
```

Compute power

```
power logistic oratio_X, x(xzspec) n(numlist) [z1(xzspec) [z2(xzspec) [...]]
                                              generalopts]
```

Compute effect size

```
power logistic, x(xzspec) n(numlist) power(numlist) [z1(xzspec) [z2(xzspec) [...]]
                                                    generalopts]
```

oratio$_X$ is the odds ratio for covariate of interest X under the alternative hypothesis H_a. Argument *oratio*$_X$ may be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**); argument *oratio*$_X$ does not appear in the dialog box.

generalopts	Description
Main	
* alpha(*numlist*)	significance level; default is alpha(0.05)
* power(*numlist*)	power; default is power(0.8)
* beta(*numlist*)	probability of type II error; default is beta(0.2)
* n(*numlist*)	sample size; required to compute power or effect size
nfractional	allow fractional sample size
† x(*xzspec*)	distribution and effect of covariate of interest X
effect(oratio | coefficient)	specify the type of effect to display; default is effect(oratio)
z[#](*xzspec*)	distribution and effect of nuisance covariate $Z_{\#}$
* corrxz(*numlist*)	coefficient of (multiple) correlation between covariate X and all covariates $\mathbf{Z}$; default is corrxz(0)
* pycondxmzm(*numlist*)	success probability of Y given mean values of X and $\mathbf{Z}$; $\Pr\{Y = 1 \mid X = E(X),\ \mathbf{Z} = E(\mathbf{Z})\}$
* pycondx0zm(*numlist*)	success probability of Y given $X = 0$ and mean values of covariates $\mathbf{Z}$; $\Pr\{Y = 1 \mid X = 0,\ \mathbf{Z} = E(\mathbf{Z})\}$
* intercept(*numlist*)	intercept for logistic regression
direction(upper | lower)	direction of the effect for effect-size determination; default is direction(upper), which means that the postulated odds ratio for X is greater than 1 (thus, the coefficient is positive)
parallel	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Discretization	
* minbins(*numlist*)	minimum product of bins for all covariates
* nbins(*numlist*)	number of bins to use for discretizing each binned covariate
Table	
[no]table[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
saving(*filename* [, replace])	save the table data to *filename*; use replace to overwrite existing *filename*
Graph	
graph[(*graphopts*)]	graph results; see [PSS-2] **power, graph**
Iteration	
init(#)	initial odds ratio for effect-size calculation
iterate(#)	maximum number of iterations; default is iterate(500)
tolerance(#)	parameter tolerance; default is tolerance(1e-12)
ftolerance(#)	function tolerance; default is ftolerance(1e-12)
[no]log	suppress or display iteration log
[no]dots	suppress or display iterations as dots
coefx(*numlist*)	coefficient for X in logistic regression; specify instead of odds ratio $oratio_X$
notitle	suppress the title

†`x()` is required.

*Specifying a list of values in at least two starred options, or in argument *oratio*$_X$ and at least one starred option, results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see `parallel`.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`notitle` and `coefx()` do not appear in the dialog box.

xzspec	Description
†`distribution(`*distspec* [`, *nbins(`*numlist*`)`]`)	covariate distribution and the number of bins for discretization
*`oratio(`*numlist*[`, unit(#|sd)`]`)	odds ratio for the covariate and the unit change; default is `unit(1)`
*`coefficient(`*numlist*`)`	coefficient for the covariate

†`distribution()` is required.

*Specifying a list of values in at least two starred options, suboptions, or arguments results in computations for all possible combinations of the values; also see the `parallel` option.

distspec	Distribution
`bernoulli` p^*	Bernoulli with success probability p; synonym for `binomial(1` p`)`
`beta` a^* b^*	beta with shape parameters a and b
`binomial` n p^*	binomial with n trials and success probability p
`exponential` b^*	exponential with scale parameter b
`laplace` m^* b^*	Laplace with mean m and scale parameter b
`logistic` m^* s^*	logistic with mean m and scale parameter s
`lognormal` μ^* σ^*	lognormal with mean μ and standard deviation σ
`normal` μ^* σ^*	normal with mean μ and standard deviation σ
`ordinal` $(v_1^*\ p_1)$ $(v_2^*\ p_2)$ [$(v_3^*\ p_3)$ [`...`]]	ordinal with values v_1^*, v_2^*, etc., and respective probabilities p_1, p_2, etc.
`poisson` m^*	Poisson with mean m
`uniform` a^* b^*	uniform on the interval $[a, b]$

*Starred parameters may be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**). Specifying a list of values in at least two starred parameters, options, suboptions, or arguments results in computations for all possible combinations of the values; also see the `parallel` option.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`N`	number of subjects	N
`delta`	effect size	δ
`oratiox`	odds ratio for X	OR_X
`unitx`	unit change in X for odds ratio	u_X
`coefx`	coefficient for X	β_X
`nbinsx`	number of bins for discretized X	B_X
`oratioz#`	odds ratio for $Z_\#$	$\text{OR}_{Z\#}$
`unitz#`	unit change in $Z_\#$ for odds ratio	$u_{Z\#}$
`coefz#`	coefficient for $Z_\#$	$\zeta_\#$
`nbinsz#`	number of bins for discretized $Z_\#$	$B_{Z\#}$
`corrxz`	multiple correlation between X and $\mathbf{Z}$	R
`pycondxmzm`	success probability of Y given mean values of X and $\mathbf{Z}$, $\Pr\{Y=1\|X=E(X),\ \mathbf{Z}=E(\mathbf{Z})\}$	$p_{Y\|X=E(X),\,Zs=E(Zs)}$
`pycondx0zm`	success probability of Y given X is 0 and mean value of $\mathbf{Z}$, $\Pr\{Y=1\|X=0,\ \mathbf{Z}=E(\mathbf{Z})\}$	$p_{Y\|X=0,\,Zs=E(Zs)}$
`intercept`	intercept	ζ_0
`nbins`	number of bins to use for discretizing each covariate	B_{all}
`minbins`	minimum product of bins for all covariates	B_{min}
`totalbins`	actual product of bins for all covariates	B_{tot}
`a[x|z#]`	parameter a from distribution of X or $Z_\#$ (if specified)	a_X or $a_{Z\#}$
`b[x|z#]`	parameter b from distribution of X or $Z_\#$ (if specified)	b_X or $b_{Z\#}$
`m[x|z#]`	mean m from distribution of X or $Z_\#$ (if specified)	m_X or $m_{Z\#}$
`n[x|z#]`	number of trials n from binomial distribution of X or $Z_\#$ (if specified)	n_X or $n_{Z\#}$
`p[x|z#]`	success probability from distribution of X or $Z_\#$ (if specified)	p_X or $p_{Z\#}$
`s[x|z#]`	scale s from logistic distribution of X or $Z_\#$ (if specified)	s_X or $s_{Z\#}$
`mu[x|z#]`	mean μ from distribution of X or $Z_\#$ (if specified)	μ_X or $\mu_{Z\#}$
`sigma[x|z#]`	standard deviation σ from distribution of X or $Z_\#$ (if specified)	σ_X or $\sigma_{Z\#}$
`v#[x|z#]`	level # ordinal value, $v_\#$, from ordinal distribution of X or $Z_\#$ (if specified)	$v\#_X$ or $v\#_{Z\#}$
`p#[x|z#]`	probability $p_\#$ that X or $Z_\#$equals $v_\#$ from ordinal distribution (if specified)	$p\#_X$ or $p\#_{Z\#}$
`nl[x|z#]`	number of levels from ordinal distribution of X or $Z_\#$ (if specified)	nl_X or $\text{nl}_{Z\#}$
`target`	target parameter; odds ratio or coefficient for X	
`_all`	display all supported columns	

Options

Main

`alpha()`, `power()`, `beta()`, `n()`, `nfractional`; see [PSS-2] **power**. The `nfractional` option is allowed only for sample-size determination.

`x(`*xzspec*`)` is a required option that specifies information about the covariate of interest, X. This includes the name of the X distribution and any parameters needed to specify that distribution, as well as the number of bins to use when discretizing X. When calculating power or sample size, *xzspec* specifies the effect of X as a coefficient or an odds ratio.

xzspec consists of the following suboptions: `distribution(`*distspec* [`, nbins(`*numlist*`)`]`)`, `oratio(`*numlist* [`, unit(#|sd)`]`)`, and `coefficient(`*numlist*`)`.

`distribution(`*distspec* [`, nbins(`*numlist*`)`]`)` is a required suboption, where *distspec* specifies the distribution of the covariate and `nbins()` specifies how it is to be discretized.

distspec consists of the distribution name and parameters. Starred parameters may be specified as a number or numlist in parentheses. *distspec* is one of the following:

`bernoulli` p^* specifies a Bernoulli distribution with parameter p, where $0 < p < 1$. The Bernoulli distribution describes a binary trial with outcomes 0 (failure) or 1 (success). Parameter p is the probability of success, and a Bernoulli random variable has mean p. Bernoulli covariates always use two bins during discretization, one for each possible outcome.

`beta` a^* b^* specifies a beta distribution with shape parameters a and b, where $a > 0$ and $b > 0$. A random variable following a beta distribution is defined only over the interval $[0, 1]$, and its mean is $a/(a + b)$.

`binomial` n p^* specifies a binomial distribution with parameters n and p, where n is a positive integer and $0 < p < 1$. A binomial random variable models the number of successes in n Bernoulli trials, each with success probability p, and its mean is $n \times p$. Binomial covariates always use $n + 1$ bins during discretization, one for each possible outcome.

`exponential` b^* specifies an exponential distribution with scale parameter b, where $b > 0$. A random variable following an exponential distribution can take only positive values, and its mean is b.

`laplace` m^* b^* specifies a Laplace distribution with mean m and scale parameter b, where $b > 0$. The Laplace distribution is symmetric around its mean and is defined for all real numbers.

`logistic` m^* s^* specifies a logistic distribution with mean m and scale parameter s, where $s > 0$. The logistic distribution is symmetric around its mean and is defined for all real numbers.

`lognormal` μ^* σ^* specifies a lognormal distribution with parameters μ and σ, where $\sigma > 0$. If random variable Q is lognormal with parameters μ and σ, its mean is $\exp(\mu + \sigma^2/2)$, and the natural logarithm of Q follows a normal distribution with mean μ and standard deviation σ.

`normal` μ^* σ^* specifies a normal distribution with mean μ and standard deviation σ, where $\sigma > 0$. The normal distribution is symmetric around its mean and is defined for all real numbers.

`ordinal` $(v_1^* \; p_1) \; (v_2^* \; p_2) \; [\,(v_3^* \; p_3) \; [\,\ldots\,]\,]$ specifies an ordinal distribution with parameters **v** and **p**, which are equal-length vectors. Parameter **v** is an ordered vector of values $(v_1, v_2, \ldots, v_J)$, where $v_1 < v_2 < \cdots < v_J$ and $J \leq 20$. Parameter **p** is a vector of probabilities corresponding to those values $(p_1, p_2, \ldots, p_J)$, where each probability is between 0 and 1 and $p_1 + p_2 + \cdots + p_J = 1$. To specify an ordinal covariate, enclose corresponding pairs of v_j and p_j values in parentheses, such as `x(ordinal (1 0.3) (2 0.5) (3 0.2))`. During discretization, ordinal covariates always use as many bins as they have values: one for each possible outcome. The mean of an ordinal random variable is $v_1 \times p_1 + v_2 \times p_2 + \cdots + v_J \times p_J$.

`poisson` m^* specifies a Poisson distribution with parameter m, where $m > 0$. The Poisson distribution is often used to model count data. A Poisson random variable can take only nonnegative integer values, and its mean is m.

`uniform` a^* b^* specifies a continuous uniform distribution over the interval $[a, b]$, where $a < b$. The mean of a uniformly distributed random variable is $(a + b)/2$.

`nbins(`*numlist*`)` specifies the number of bins to use when discretizing the covariate; the number of bins must be an integer between 2 and 100,000,000. For this covariate, the `nbins()` suboption overrides any value set by the `nbins()` global option. The `nbins()` suboption is not allowed with the `bernoulli`, `binomial`, or `ordinal` distribution.

`oratio(`*numlist*`[ , unit(#|sd) ])` specifies the odds ratio for the covariate in the logistic regression. The odds ratio must be positive, and only one of the `oratio()` or `coefficient()` suboption may be specified for each covariate. When you specify the covariate of interest X, the `oratio()` suboption may not take the value 1, and `oratio()` is not a valid suboption of `x()` when calculating effect size or when argument *oratio*$_X$ is specified.

`unit(#|sd)` specifies the unit change for the odds ratio. Specifying `unit(sd)` indicates that the odds ratio is for a 1-standard-deviation increase in the covariate. The relationship between the odds ratio and the coefficient is `oratio` = exp(`coefficient`×`unit`). The default is `unit(1)`.

`coefficient(`*numlist*`)` specifies the coefficient for the covariate in the logistic regression. Only one of the `coefficient()` or `oratio()` suboption may be specified for each covariate. When you specify the covariate of interest X, the `coefficient()` suboption may not take the value 0, and `coefficient()` is not a valid suboption of `x()` when calculating effect size or when argument *oratio*$_X$ is specified.

`effect(oratio|coefficient)` specifies how to report the effect size in the output. By default, the effect is output as the odds ratio for X unless the coefficient for X is specified, in which case it defaults to the coefficient. The `effect()` option is used to override the default.

`z[#](`*xzspec*`)` specifies information about nuisance covariate $Z_{\#}$. This includes the distribution and its parameters for $Z_{\#}$, the coefficient or odds ratio for $Z_{\#}$, and the number of bins to use when discretizing $Z_{\#}$. Up to 20 Z covariates may be specified, and Z covariates are assumed to be uncorrelated with each other (correlation with X is allowed; see the `corrxz()` option).

`corrxz(`*numlist*`)` specifies the correlation between X and **Z**, labeled R, where $-1 < R < 1$. If there is just one Z covariate, this is Pearson's correlation coefficient. If there are multiple Z covariates, `corrxz()` specifies the coefficient of multiple correlation, a generalization of Pearson's correlation coefficient. The default is `corrxz(0)`, which indicates no correlation between X and **Z**.

`pycondxmzm(`*numlist*`)` specifies the conditional success probability of Y given mean values of X and all $\mathbf{Z}$ covariates, $\Pr\{Y = 1|X = E(X), \mathbf{Z} = E(\mathbf{Z})\}$, where $0 < \Pr\{Y = 1|X = E(X), \mathbf{Z} = E(\mathbf{Z})\} < 1$. This option is not allowed with effect-size determination.

`pycondx0zm(`*numlist*`)` specifies the conditional success probability of Y given $X = 0$ and mean values of all $\mathbf{Z}$ covariates, $\Pr\{Y = 1|X = 0, \mathbf{Z} = E(\mathbf{Z})\}$, where $0 < \Pr\{Y = 1|X = 0, \mathbf{Z} = E(\mathbf{Z})\} < 1$. `pycondx0zm()` may not be combined with the `intercept()` option.

`intercept(`*numlist*`)` specifies the intercept for the logistic regression, ζ_0. The `intercept()` option may not be combined with the `pycondx0zm()` option.

`direction()`, `parallel`; see [PSS-2] **power**.

Discretization

`minbins(`*numlist*`)` specifies the minimum product of the bins for all covariates B_{min}, where B_{min} is an integer between 2 and 100,000,000. Covariates with Bernoulli, binomial, and ordinal distributions always use one bin for each value they can take, and the `nbins()` suboption of `x()` and `z#()` sets the number of bins for one covariate at a time. The value of `minbins()` is used when determining how many bins to allocate to the remaining covariates; they are discretized such that the product of the number of bins of each covariate exceeds `minbins()`. $B_{\text{min}} \leq B_X \times B_{Z_1} \times B_{Z_2} \ldots$, where B_X is the number of bins for X, B_{Z_1} is the number of bins for Z_1, and so on. The default is `minbins(10000)` for power and sample-size calculations and `minbins(1000)` for effect-size calculations. The `minbins()` option may not be combined with the `nbins()` global option (but it can be combined with the `nbins()` suboption of `x()` or `z#()`).

`nbins(`*numlist*`)` specifies the number of bins to use when discretizing each covariate; the number of bins must be an integer between 2 and 100,000,000. The `nbins()` option can be overridden on a per-covariate basis by specifying the `nbins()` suboption of `x()` or `z#()`. Covariates with Bernoulli, binomial, and ordinal distributions always use one bin for each value they can take, so they do not respect `nbins()`. The `nbins()` option may not be combined with the `minbins()` option (but the `nbins()` suboption of `x()` or `z#()` can be combined with `minbins()`). Note that the product of the number of bins for all covariates, B_{tot}, may not exceed 100,000,000. ($B_{\text{tot}} = B_X \times B_{Z_1} \times B_{Z_2} \cdots \leq$ 100,000,000, where B_X is the number of bins for X, B_{Z_1} is the number of bins for Z_1, and so on). Thus, the maximum of `nbins(100000000)` may be specified only if there is a single covariate of interest without any nuisance covariates.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies the initial value of the odds ratio for X during effect-size determination. The default is `init(1.5)` with `direction(upper)` and `init(0.67)` with `direction(lower)`.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following options are available with `power logistic` but are not shown in the dialog box:

`coefx(`*numlist*`)` specifies the coefficient for covariate X in the logistic regression, β_X, where $\beta_X \neq 0$. The `coefx()` option may be specified instead of argument *oratio*$_X$. This option is not allowed with effect-size determination.

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `power logistic` command and the methodology for power and sample-size analysis for logistic regression with one covariate of interest, X, and up to 20 nuisance covariates $\mathbf{Z} = (Z_1, Z_2, \ldots)$. Covariates X and $\mathbf{Z}$ can be continuous, discrete, or a combination of both. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis, and see [PSS-2] **power** for a general introduction to using the `power` command for hypothesis tests.

Introduction

Logistic regression is a commonly used statistical method for analyzing binary outcome variables. Researchers often need to determine the appropriate sample size to ensure sufficient power for detecting the association between a covariate of interest (X) and a binary outcome variable (Y) while controlling for the effect of nuisance covariates ($\mathbf{Z}$).

For example, consider a study that examines factors influencing whether migratory birds return to the same nesting site from one year to the next. Binary outcome Y is an indicator of whether the nesting site was reused, where the observed $y_i = 1$ if bird i returns to the nesting site it used last year and $y_i = 0$ if bird i does not. We will use logistic regression to test whether birds of one sex are more likely to return to the same nesting site than birds of the other sex. We define binary covariate of interest X as an indicator for female birds, where the observed $x_i = 1$ if bird i is female and $x_i = 0$ if bird i is male. Previous observations suggest that birds that mate with the same partner as last year are more likely to return to the same nesting site, as are heavier birds. We are not interested in studying the effect of a bird's mate or weight, but it would be foolhardy to ignore these effects. We include mate in our logistic regression as nuisance covariate Z_1, where the observed $z_{1i} = 1$ if bird i has the same mate as last year and $z_{1i} = 0$ if not. And we include weight as nuisance covariate Z_2, where the observed z_{2i} is the weight of bird i. The logistic regression can be written as

$$\Pr(y_i = 1 | x_i, \mathbf{z}_i) = H(\beta_X x_i + \zeta_0 + \zeta_1 z_{1i} + \zeta_2 z_{2i}) \qquad i = 1, 2, \ldots, n$$

where β_X is the coefficient quantifying the effect of covariate X, a bird's sex, on nesting-site reuse; ζ_0 is the logistic intercept; ζ_1 is the effect of covariate Z_1, partnering with the same mate; ζ_2 is the effect of covariate Z_2, weight; and n is the sample size. Function $H(\eta) = \{1 + \exp(-\eta)\}^{-1}$ is the logistic distribution function.

The effect of covariate X can also be expressed in terms of an odds ratio,

$$\text{OR}_X = \exp(\beta_X \text{u}_X) = \Pr(Y=1|X=1)\Pr(Y=0|X=0)/\{\Pr(Y=0|X=1)\Pr(Y=1|X=0)\}$$

where u_X is the unit change in X for the odds ratio and $\text{u}_X > 0$. In this example, u_X must equal 1 because X is a Bernoulli random variable. The null hypothesis is $H_0 : \beta_X = 0$, which can also be expressed as $H_0 : \text{OR}_X = 1$. The alternative hypothesis is $H_a : \beta_X \neq 0$ or, equivalently, $H_a : \text{OR}_X \neq 1$.

Logistic regression is commonly used to analyze binary outcomes in observational studies, such as the study of nesting-site reuse. But it can also be used to analyze data from a randomized controlled trial, where trial participants are randomly assigned to a treatment. For instance, a public health study might investigate whether attending a support group helps smokers quit smoking. In this example, participation in the support group is binary covariate X: Some participants will be randomly assigned to attend support group meetings for, say, three months, whereas other participants will be randomized to the control group, which does not attend support group meetings. At the end of three months, smoking status is recorded; this is binary outcome Y. Based on previous research, we anticipate males and females will have different success rates at quitting smoking, and we expect younger smokers to be more successful at quitting than older smokers. We are not interested in studying the effect of sex or age on smoking cessation, but it would be foolhardy to ignore these effects, so we include sex in our logistic regression as nuisance covariate Z_1, and we include age as nuisance covariate Z_2.

The `power logistic` command provides power and sample-size analysis for the test of $\beta_X = 0$ in logistic regression. The formula for power, sample-size, and effect-size calculations is based on the likelihood-ratio test of $\beta_X = 0$. However, Bush (2015) demonstrates that sample-size requirements for Wald and score tests are generally equivalent to the sample-size requirement for the likelihood-ratio test. Thus, these calculations may be used to plan studies that will be analyzed using the `logit` command, which conducts a Wald test of $\beta_X = 0$, or the `logistic` command, which conducts an equivalent Wald test of $\text{OR}_X = 1$. If you prefer a likelihood-ratio test, you can use the `lrtest` command.

Using power logistic with arbitrary covariates

`power logistic` computes sample size, power, or effect size for a test of one coefficient in logistic regression. This entry describes how to use `power logistic` when the logistic regression can have multiple discrete and continuous covariates, only one of which is of interest (X), while the others are nuisance covariates ($\mathbf{Z}$). All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option. You must specify the distribution of covariate X in the target population with the `x()` option. Nuisance covariates can be specified in `z1()`, `z2()`, and so on.

Power and sample-size calculations require that you specify information about three types of parameters from the logistic regression: the X coefficient (β_X), the Z coefficients ($\zeta_1, \zeta_2, \ldots$), and the intercept (ζ_0). For each nuisance covariate $Z_\#$ in the logistic regression, you must specify information about $\zeta_\#$ using either `z#(oratio())` or `z#(coefficient())`.

The information about the X coefficient (β_X) can be specified directly in the `x(coefficient())` option or indirectly as an argument *oratio*$_X$, in the `x(oratio())` option, or in the `pycondxmzm()` option. The information about the intercept (ζ_0) can be specified directly in the `intercept()` option or indirectly in the `pycondx0zm()` or `pycondxmzm()` option. See figure 1 for a graphical depiction of various ways to provide information about β_X and ζ_0.

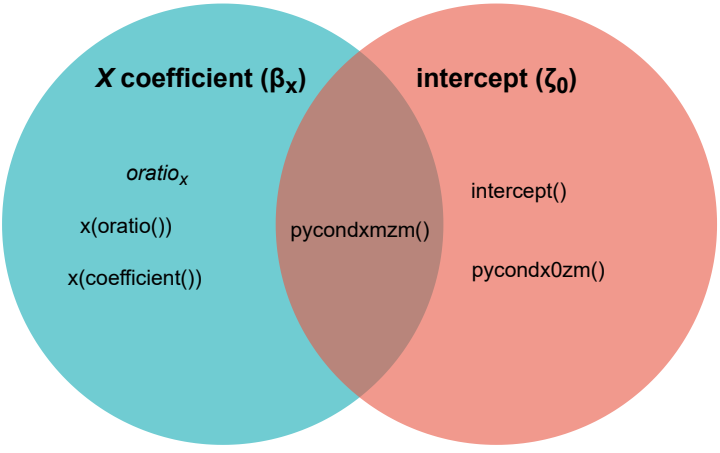


Figure 1. Specifying information about β_X and ζ_0

Valid ways of specifying the X coefficient and intercept include, for example, *oratio*$_X$ and `intercept()` or `pycondxmzm()` and `pycondx0zm()`. However, the combination of `intercept()` and `pycondx0zm()` is invalid because no information is provided about the X coefficient. When you compute sample size, the power of the test may be specified using the `power()` option, which has a default of 0.8. When you compute power, the sample size must be specified using the `n()` option.

When `power logistic` is used to calculate effect size β_X, the command specification cannot include *oratio*$_X$ or options that provide information about the X coefficient, such as the `pycondxmzm()` or `x(oratio())` option. In this case, the procedure for specifying the Z coefficient or Z coefficients is unchanged, but information about the intercept must be specified using `intercept()` or `pycondx0zm()`.

To calculate effect size, you must specify sample size using the `n()` option, power using the `power()` option, and, optionally, the direction of the effect using the `direction()` option. The default is `direction(upper)`, which means that coefficient β_X is assumed to be positive. This is equivalent to assuming that the odds ratio OR$_X$ is greater than 1. You can change the direction to `lower`, which means that $\beta_X < 0$ or, equivalently, OR$_X < 1$.

The `effect()` option can be used to specify the type of effect to be reported in the output. Valid choices are `effect(oratio)` and `effect(coefficient)`. If the coefficient for X is specified, the default output parameterizes the effect size as a coefficient; otherwise, the default is an odds ratio. The `effect()` option is used to override the default.

By default, the computed sample size is rounded up. You can specify the `nfractional` option to see the corresponding fractional sample size; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. The `nfractional` option is allowed only for sample-size determination.

Some of the computations of `power logistic` require iteration, specifically, the computations used in effect-size determination. The default initial value for OR$_X$ is 1.5 with `direction(upper)` and 0.67 with `direction(lower)`. This may be changed by specifying the `init()` option. See [PSS-2] **power** for descriptions of other options that control the iteration procedure.

In the following sections, we describe the use of `power logistic` to compute sample size, power, and effect size.

Computing sample size

To compute sample size, you must specify the distribution of the covariate of interest X using the `distribution()` suboption of `x()`; the distributions and coefficients or odds ratios of any nuisance covariates using `z1()`, `z2()`, and so on; the information necessary to determine parameters β_X and ζ_0, as described in *Using power logistic with arbitrary covariates*; and, optionally, the power of the test using the `power()` option or the type-II-error probability using the `beta()` option. A default power of 0.8 is assumed if the power of the test is not specified. The level of the test is specified using the `alpha()` option, with a default of `alpha(0.05)`.

Example 1: Sample size with a standard normal covariate of interest

Consider an example from the seminal work on sample-size calculation for logistic regression by Whittemore (1981, 31) that describes the design of a study testing "the null hypothesis that risk of coronary heart disease (CHD) among white males aged 39–59 is unaffected by serum cholesterol levels". In example 1 of [PSS-2] **power logistic onebin** and example 1 of [PSS-2] **power logistic twobin**, we approached this scenario using a binary indicator variable for elevated cholesterol, but here we model serum cholesterol as a normally distributed random variable, just as Whittemore did.

Normally distributed covariate of interest X is serum cholesterol, which has been standardized to have mean 0 and standard deviation 1. We want to control for the effects of triglyceride, nuisance covariate Z_1, which is a known risk factor for CHD. Log-triglyceride levels are standardized and modeled as a normal random variable with mean 0 and standard deviation 1. Following Whittemore (1981), we anticipate the odds ratio for a one-unit change in Z_1 to be 1.25, and the correlation between X and Z_1 is estimated to be 0.4. Based on data from Hulley et al. (1980), we assume a probability of 0.07 that an individual in the target population will develop CHD during an 18-month study period if he has average values for serum cholesterol and triglycerides; this will be entered as `pycondxmzm(0.07)`.

Like Whittemore, we will calculate the sample size required to detect an odds ratio of 1.65 at the $\alpha = 0.05$ level, but we will use the default power of 80%.

```
. power logistic 1.65, x(distribution(normal 0 1))
> z1(distribution(normal 0 1) oratio(1.25))
> corrxz(0.4) pycondxmzm(0.07)

Estimated sample size for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    1.6500  (odds ratio)
       corrxz =    0.4000
   pycondxmzm =    0.0700

Covariate of interest X: Normal(mux, sigmax), bins = 100
      oratiox =    1.6500
          mux =    0.0000
       sigmax =    1.0000

Nuisance covariate Z1: Normal(muz1, sigmaz1), bins = 100
     oratioz1 =    1.2500
         muz1 =    0.0000
      sigmaz1 =    1.0000

Estimated sample size:

            N =       521
```

The output of `power logistic` begins by displaying information about the test to be conducted and its null and alternative hypotheses. The study parameters we specified are listed next, followed by information about covariates X and Z_1. For both X and Z_1, we see the odds ratios we specified, along with means and standard deviations. We also see that these normal distributions were discretized into 100 bins each. We did not specify the number of bins to use, so `power logistic` used the default of `minbins(10000)` for sample-size calculations. The `minbins()` option specifies the minimum value for the product of bins for all covariates rather than directly specifying the number of bins per covariate (which can be done with the `nbins()` option or the `nbins()` suboption of the `x(distribution())` and `z#(distribution())` options). With 100 bins per covariate, we have a product of $100 \times 100 = 10{,}000$, satisfying the default `minbins(10000)`.

Finally, we find that a sample of 521 subjects is required to detect an odds ratio of 1.65 with 80% power using a 5% level test. But how sensitive is this sample-size estimate to our discretization process? We repeat the previous calculation, but this time, we specify different values for the minimum product of bins for all covariates. For demonstration purposes, we move the specification of OR_X from argument *oratio*$_X$ to suboption `x(oratio())`.

```
. power logistic, x(distribution(normal 0 1) oratio(1.65))
> z1(distribution(normal 0 1) oratio(1.25))
> corrxz(0.4) pycondxmzm(0.07)
> minbins(100 1000 10000 100000 1000000)

Estimated sample size for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1

Covariate of interest X: Normal(mux, sigmax)
  Nuisance covariate Z1: Normal(muz1, sigmaz1)
```

alpha	power	N	delta	oratiox	mux	sigmax	nbinsx	oratioz1
.05	.8	600	1.65	1.65	0	1	10	1.25
.05	.8	539	1.65	1.65	0	1	32	1.25
.05	.8	521	1.65	1.65	0	1	100	1.25
.05	.8	514	1.65	1.65	0	1	317	1.25
.05	.8	512	1.65	1.65	0	1	1000	1.25

muz1	sigmaz1	nbinsz1	corrxz	pycondxmzm	minbins	totalbins
0	1	10	.4	.07	100	100
0	1	32	.4	.07	1000	1024
0	1	100	.4	.07	10000	10000
0	1	317	.4	.07	1.0e+05	1.0e+05
0	1	1000	.4	.07	1.0e+06	1.0e+06

When we specify multiple values for one or more parameters or arguments, `power logistic` presents the results as a table. Above the table is information about the test to be conducted, with null and alternative hypotheses followed by the distributions of covariates X and Z_1. The table has columns for each of the logistic regression parameters we specified, as well as estimated sample size `N` and additional details about discretization (`nbinsx`, `nbinsz1`, and `totalbins`).

By increasing the minimum product of bins from 100 to 1,000,000 by successive orders of magnitude, we caused the number of bins for each covariate to increase from 10 all the way to 1,000. Using more bins to discretize the covariates increased the precision of our sample-size calculation, but it came at the expense of increased computational time, with diminishing returns above `minbins()` values of 10,000.

◁

▷ Example 2: Sample size with continuous covariates

Instead of standardizing serum cholesterol and log triglycerides as in example 1, we now input the means and standard deviations in the `x(distribution())` and `z1(distribution())` suboptions. To indicate that the odds ratios now refer to a 1-standard-deviation increase in the covariates, we include suboption `unit(sd)` when specifying the odds ratios of these covariates. We leave the rest of the command specification unchanged.

```
. power logistic, x(distribution(normal 212 38) oratio(1.65, unit(sd)))
> z1(distribution(normal 4.9 0.3) oratio(1.25, unit(sd)))
> corrxz(0.4) pycondxmzm(0.07)

Estimated sample size for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    1.0133  (odds ratio)
       corrxz =    0.4000
   pycondxmzm =    0.0700

Covariate of interest X: Normal(mux, sigmax), bins = 100
      oratiox =    1.6500
        unitx =   38.0000
          mux =  212.0000
       sigmax =   38.0000

Nuisance covariate Z1: Normal(muz1, sigmaz1), bins = 100
     oratioz1 =    1.2500
       unitz1 =    0.3000
         muz1 =    4.9000
      sigmaz1 =    0.3000

Estimated sample size:

            N =       521
```

The output of this command is similar to that of example 1, but now information about `unitx` and `unitz1` is included in the descriptions of those covariates. The estimated sample size, however, is unchanged.

◁

Example 3: Sample size with continuous and discrete covariates

We now include an additional nuisance covariate, smoking status, as Z_2. The probability that an individual in the target population is a smoker is 0.38, and the odds ratio for Z_2 is 3. Smoking status is uncorrelated with the covariate of interest, so the coefficient of multiple correlation remains 0.4. We leave the rest of the command specification unchanged except for the addition of the `effect(coefficient)` option to display the effect of X as a coefficient instead of an odds ratio.

```
. power logistic, x(distribution(normal 212 38) oratio(1.65, unit(sd)))
> z1(distribution(normal 4.9 0.3) oratio(1.25, unit(sd)))
> z2(distribution(bernoulli 0.38) oratio(3))
> corrxz(0.4) pycondxmzm(0.07) effect(coefficient)

Estimated sample size for logistic regression coefficient test
Likelihood-ratio test
H0: beta_X = 0  versus  Ha: beta_X != 0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.0132  (coefficient)
       corrxz =    0.4000
   pycondxmzm =    0.0700

Covariate of interest X: Normal(mux, sigmax), bins = 71
        coefx =    0.0132
          mux =  212.0000
       sigmax =   38.0000

Nuisance covariate Z1: Normal(muz1, sigmaz1), bins = 71
     oratioz1 =    1.2500
       unitz1 =    0.3000
         muz1 =    4.9000
      sigmaz1 =    0.3000

Nuisance covariate Z2: Bernoulli(pz2), bins = 2
     oratioz2 =    3.0000
          pz2 =    0.3800

Estimated sample size:

            N =       494
```

Under `Covariate of interest X:`, we now see `coefx` instead of `oratiox` and `unitx`. But the major change is the addition of covariate Z_2, which reduces the estimated sample size to 494. Bernoulli random covariate Z_2 can take only 2 values, which is why it is discretized into 2 bins. discretized into 71 bins each, yielding a product of $71 \times 71 \times 2 = 10{,}082$ bins.

◁

Computing power

To compute power, you must specify the distribution of covariate of interest X using the `x(distribution())` option; the distributions and coefficients or odds ratios of any nuisance covariates using `z1()`, `z2()`, and so on; the information necessary to determine parameters β_X and ζ_0, as described in *Using power logistic with arbitrary covariates* ; and the sample size using the `n()` option. The level of the test is specified using the `alpha()` option, with a default of `alpha(0.05)`.

▷ Example 4: Power of a logistic regression odds-ratio test

Continuing with example 3, we anticipate a sample of 600 subjects and would like to compute the power corresponding to this sample size. We specify the same study parameters we used in example 3, but now we use the `n()` option to specify a sample size of 600.

```
. power logistic, x(distribution(normal 212 38) oratio(1.65, unit(sd)))
> z1(distribution(normal 4.9 0.3) oratio(1.25, unit(sd)))
> z2(distribution(bernoulli 0.38) oratio(3))
> corrxz(0.4) pycondxmzm(0.07) n(600)

Estimated power for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1

Study parameters:

        alpha =    0.0500
            N =       600
        delta =    1.0133  (odds ratio)
       corrxz =    0.4000
   pycondxmzm =    0.0700

Covariate of interest X: Normal(mux, sigmax), bins = 71
      oratiox =    1.6500
        unitx =   38.0000
          mux =  212.0000
       sigmax =   38.0000

Nuisance covariate Z1: Normal(muz1, sigmaz1), bins = 71
     oratioz1 =    1.2500
       unitz1 =    0.3000
         muz1 =    4.9000
      sigmaz1 =    0.3000

Nuisance covariate Z2: Bernoulli(pz2), bins = 2
     oratioz2 =    3.0000
          pz2 =    0.3800

Estimated power:

        power =    0.8707
```

If the study recruits 600 participants, the power to detect an odds ratio of 1.65 climbs to 87.07%.

◁

▷ Example 5: Multiple values of study parameters

To investigate the effect of sample size on power, we can specify a list of sample sizes in the `n()` option:

```
. power logistic, x(distribution(normal 212 38) oratio(1.65, unit(sd)))
> z1(distribution(normal 4.9 0.3) oratio(1.25, unit(sd)))
> z2(distribution(bernoulli 0.38) oratio(3))
> corrxz(0.4) pycondxmzm(0.07) n(400 500 600 700)

Estimated power for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1

Covariate of interest X: Normal(mux, sigmax)
Nuisance covariates:
   Z1: Normal(muz1, sigmaz1)
   Z2: Bernoulli(pz2)
```

alpha	power	N	delta	oratiox	unitx	mux	sigmax	oratioz1
.05	.7132	400	1.013	1.65	38	212	38	1.25
.05	.8052	500	1.013	1.65	38	212	38	1.25
.05	.8707	600	1.013	1.65	38	212	38	1.25
.05	.9158	700	1.013	1.65	38	212	38	1.25

unitz1	muz1	sigmaz1	oratioz2	pz2	corrxz	pycondxmzm
.3	4.9	.3	3	.38	.4	.07
.3	4.9	.3	3	.38	.4	.07
.3	4.9	.3	3	.38	.4	.07
.3	4.9	.3	3	.38	.4	.07

As expected, when the sample size increases, the power increases toward 1.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

Computing effect size

By default, effect size δ for a logistic regression odds-ratio test is defined as the odds ratio for X: $\delta = \text{OR}_X = \exp(\beta_X/\text{u}_X)$. Sometimes, we want to know the smallest effect that can be detected with a level α test at a prespecified power and sample size.

To compute the effect size, you must specify the distribution of the covariate of interest X using the `x(distribution())` option; the distributions and coefficients or odds ratios of any nuisance covariates using `z1()`, `z2()`, and so on; parameter ζ_0 using the `pycondx0zm()` or `intercept()` option; the sample size using the `n()` option; and the power of the test using the `power()` option or the type II error probability using the `beta()` option. In addition, you must pick the level of the test and the direction of the effect. The level of the test is specified using the `alpha()` option, with a default of `alpha(0.05)`. The direction of the effect is specified using the `direction()` option; the default is `direction(upper)`, which means that $\text{OR}_X > 1$ or, equivalently, $\beta_X > 0$. Specifying `direction(lower)` means that $\text{OR}_X < 1$ and $\beta_X < 0$. The estimated minimum detectable effect size is reported as an odds ratio by default. To display it as a coefficient, specify `effect(coefficient)`.

▷ Example 6: Minimum detectable odds ratio

We continue with example 4, where we learned that a size 0.05 test with 600 subjects would have 87.07% power to detect an odds ratio of 1.65. How much larger would the odds ratio need to be to detect it with 90% power? We use `power logistic` to find out.

Instead of specifying the untransformed mean and standard deviation of serum cholesterol, we return to using standardized values as we did in example 1. The reason for this is twofold: First, there is no closed-form solution for the effect-size calculation, so it requires a nonlinear solver, which performs much more efficiently on the standardized values. Second, it allows us to use a trick to specify `pycondx0zm()` (which is allowed for effect-size calculations) instead of `pycondxmzm()` (which is not). The mean of X is 0 after standardizing, so $\Pr\{Y = 1|X = E(X), \mathbf{Z} = E(\mathbf{Z})\} = \Pr\{Y = 1|X = 0, \mathbf{Z} = E(\mathbf{Z})\}$, which means that we can specify `pycondx0zm(0.07)`.

```
. power logistic, x(distribution(normal 0 1))
> z1(distribution(normal 4.9 0.3) oratio(1.25, unit(sd)))
> z2(distribution(bernoulli 0.38) oratio(3))
> corrxz(0.4) pycondx0zm(.07) n(600) power(0.9)

Performing iteration ...

Estimated odds ratio for logistic regression odds-ratio test
Likelihood-ratio test
H0: OR_X = 1  versus  Ha: OR_X != 1

Study parameters:

        alpha =    0.0500
        power =    0.9000
            N =       600
       corrxz =    0.4000
   pycondx0zm =    0.0700

Covariate of interest X: Normal(mux, sigmax), bins = 23
          mux =    0.0000
       sigmax =    1.0000

Nuisance covariate Z1: Normal(muz1, sigmaz1), bins = 23
     oratioz1 =    1.2500
       unitz1 =    0.3000
         muz1 =    4.9000
      sigmaz1 =    0.3000

Nuisance covariate Z2: Bernoulli(pz2), bins = 2
     oratioz2 =    3.0000
          pz2 =    0.3800

Estimated effect size and odds ratio:

        delta =    1.7077  (odds ratio)
      oratiox =    1.7077
```

We see that a slightly larger odds ratio of 1.7077 can be detected with 90% power. The two normal covariates are discretized into only 23 bins each because the default value of `minbins()` for effect-size calculations is 1,000, and $23 \times 23 \times 2 = 1{,}058$.

In the above, we assumed the effect to be in the upper direction. By symmetry, there exists an effect size in the lower direction that can also be detected with 90% power. We specify `direction(lower)` to find it, and we add the `effect(coefficient)` option to display it as a coefficient instead of an odds ratio.

```
. power logistic, x(distribution(normal 0 1))
> z1(distribution(normal 4.9 0.3) oratio(1.25, unit(sd)))
> z2(distribution(bernoulli 0.38) oratio(3))
> corrxz(0.4) pycondx0zm(.07) n(600) power(0.9)
> direction(lower) effect(coefficient)

Performing iteration ...

Estimated coefficient for logistic regression coefficient test
Likelihood-ratio test
H0: beta_X = 0  versus  Ha: beta_X != 0

Study parameters:

        alpha =    0.0500
        power =    0.9000
            N =       600
       corrxz =    0.4000
   pycondx0zm =    0.0700

Covariate of interest X: Normal(mux, sigmax), bins = 23
          mux =    0.0000
       sigmax =    1.0000

Nuisance covariate Z1: Normal(muz1, sigmaz1), bins = 23
     oratioz1 =    1.2500
       unitz1 =    0.3000
         muz1 =    4.9000
      sigmaz1 =    0.3000

Nuisance covariate Z2: Bernoulli(pz2), bins = 2
     oratioz2 =    3.0000
          pz2 =    0.3800

Estimated effect size and coefficient:

        delta =   -0.5351  (coefficient)
        coefx =   -0.5351
```

By specifying `direction(lower)`, we anticipate coefficient $\beta_X < 0$, which is what we see. Had we omitted the `effect(coefficient)` option, the odds ratio $\text{OR}_X = \exp(\beta_X/1)$ would have been displayed as $\exp(-0.5351) = 0.5856$.

◁

Performing hypothesis tests with logistic regression

In this section, we briefly demonstrate the use of the `logit` command for testing logistic regression coefficients; see [R] **logit** for details. Alternatively, we could use the `logistic` command to perform logistic regression because `logistic` performs the same calculations as `logit` but reports odds ratios instead of coefficients; see [R] **logistic** for details, and see example 7 of [PSS-2] **power logistic onebin** for a demonstration of how `logistic` can be used to analyze the results of a pilot study.

▷ Example 7: Analyzing a pilot study

The `nlsw88` dataset contains employment data from the 1988 extract of the National Longitudinal Study of Young Women. We will treat this dataset as if it came from a pilot study investigating the relationship between union membership (`union`) and years of job experience (`ttl_exp`) and use it to plan a

follow-up study. Our target population is American young women, some of whom are married (`married`) and some of whom are college graduates (`collgrad`). These are both factors known to influence union membership, so we include them as nuisance covariates in our logistic regression.

```
. use https://www.stata-press.com/data/r19/nlsw88
(NLSW, 1988 extract)
. logit union ttl_exp married collgrad, nolog
Logistic regression                                     Number of obs =  1,878
                                                        LR chi2(3)    =  25.42
                                                        Prob > chi2   = 0.0000
Log likelihood = -1033.9131                             Pseudo R2     = 0.0121
```

union	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
ttl_exp	.0213728	.0120162	1.78	0.075	-.0021785	.0449241
married	-.2328041	.1117316	-2.08	0.037	-.4517941	-.0138141
collgrad	.4777123	.1192228	4.01	0.000	.24404	.7113846
_cons	-1.380919	.1854974	-7.44	0.000	-1.744487	-1.017351

Years of job experience is our covariate of interest X, so the output tells us that β_X, the coefficient for `ttl_exp`, is 0.02. This suggests that women are more likely to be union members as they accumulate more job experience, but the evidence is not strong enough to reject $H_0 : \beta_X = 0$ at the 0.05 level. We want to design a follow-up study that has 80% power to detect a coefficient of 0.02 with a 0.05-level test, and we will use the parameter estimates from our pilot study to do so.

In addition to an estimate of β_X, the output of the `logit` command provides estimates of the coefficients for `married` ($\zeta_1 = -0.23$), `collgrad` ($\zeta_2 = 0.48$), and the logistic intercept `_cons` ($\zeta_0 = -1.38$). To use `power logistic`, we need additional information about the distributions of our covariates.

To visually examine the distribution of X covariate `ttl_exp`, we use the `histogram` command with the `normal` option to draw a histogram of `ttl_exp` values with a normal density for reference; see [R] **histogram** for details. We include the expression `if e(sample)` to restrict the computation so that it includes only the 1,878 participants whose data were used to fit the logistic regression; see **[U] 20.7 Specifying the estimation subsample** for details about `e(sample)`.

```
. histogram ttl_exp if e(sample), normal
(bin=32, start=.11538462, width=.89903845)
```

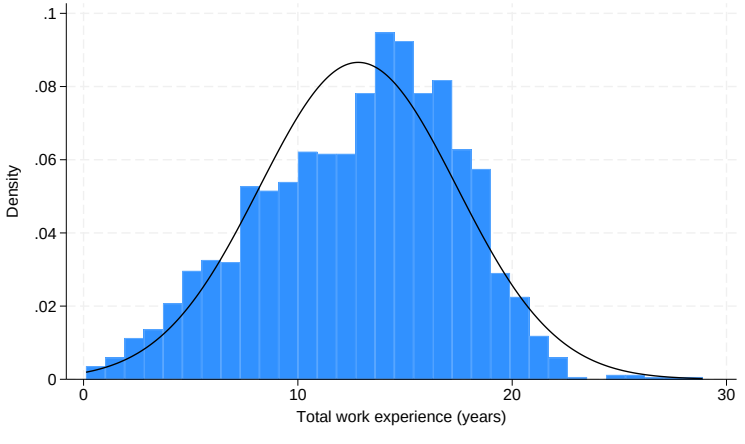


Figure 2. Histogram of total work experience

Total work experience is not quite normally distributed, but the fit is close enough to use a normal distribution to conduct a sensitivity analysis for sample-size calculations. Next we use the `summarize` command to calculate the means and standard deviations of our covariates; see [R] **summarize** for details.

```
. summarize ttl_exp married collgrad if e(sample)
```

Variable	Obs	Mean	Std. dev.	Min	Max
ttl_exp	1,878	12.81837	4.606392	.1153846	28.88461
married	1,878	.6506922	.4768783	0	1
collgrad	1,878	.2470714	.4314235	0	1

The mean of `ttl_exp` is 12.8, and the standard deviation is 4.6, but to account for uncertainty, we will perform a sensitivity analysis by specifying a numlist of values from 4 to 5 for the standard deviation of X. Binary covariates `married` and `collgrad` follow Bernoulli distributions where parameter p is equal to the mean, so we use 0.65 and 0.25 as their respective values of p.

The easiest way to calculate the multiple correlation coefficient between X and the **Z** covariates is to perform a linear regression of X on Z and take the square root of the coefficient of determination, R^2. We use the `regress` command to perform linear regression of `ttl_exp` on `married` and `collgrad` to calculate the coefficient of multiple correlation; see [R] **regress** for details.

```
. regress ttl_exp married collgrad if e(sample), notable
```

Source	SS	df	MS		
				Number of obs	= 1,878
				F(2, 1875)	= 14.73
Model	615.91641	2	307.958205	Prob > F	= 0.0000
Residual	39211.8662	1,875	20.9129953	R-squared	= 0.0155
				Adj R-squared	= 0.0144
Total	39827.7826	1,877	21.2188506	Root MSE	= 4.5731

```
. display "Multiple correlation coefficient: " sqrt(e(r2))
Multiple correlation coefficient: .12435631
```

Putting this all together, we calculate the required sample size for a 5% test with 80% power to detect a coefficient β_X of 0.02.

```
. power logistic, x(distribution(normal 12.8 (4(0.2)5)) coefficient(0.02))
> z1(distribution(bernoulli 0.65) coefficient(-0.23))
> z2(distribution(bernoulli 0.25) coefficient(0.48))
> corrxz(0.124) intercept(-1.38)

Estimated sample size for logistic regression coefficient test
Likelihood-ratio test
H0: beta_X = 0  versus  Ha: beta_X != 0

Covariate of interest X: Normal(mux, sigmax)
Nuisance covariates:
   Z1: Bernoulli(pz1)
   Z2: Bernoulli(pz2)
```

alpha	power	N	delta	coefx	mux	sigmax	coefz1	pz1
.05	.8	6,866	.02	.02	12.8	4	-.23	.65
.05	.8	6,228	.02	.02	12.8	4.2	-.23	.65
.05	.8	5,675	.02	.02	12.8	4.4	-.23	.65
.05	.8	5,192	.02	.02	12.8	4.6	-.23	.65
.05	.8	4,769	.02	.02	12.8	4.8	-.23	.65
.05	.8	4,395	.02	.02	12.8	5	-.23	.65

coefz2	pz2	corrxz	intercept
.48	.25	.124	-1.38
.48	.25	.124	-1.38
.48	.25	.124	-1.38
.48	.25	.124	-1.38
.48	.25	.124	-1.38
.48	.25	.124	-1.38

Depending on the standard deviation of X, the test will require between 4,395 and 6,866 participants to have 80% power. We have the budget to recruit 6,866 participants, so we perform a power analysis to see what the power of the test would be with a sample size of 6,866 over a range of standard deviations.

To display the result visually, we use the `graph` option.

```
. power logistic, x(distribution(normal 12.8 (4(0.2)5)) coefficient(0.02))
> z1(distribution(bernoulli 0.65) coefficient(-0.23))
> z2(distribution(bernoulli 0.25) coefficient(0.48))
> corrxz(0.124) intercept(-1.38) n(6866) graph
Covariate of interest X: Normal(mux, sigmax)
Nuisance covariates:
   Z1: Bernoulli(pz1)
   Z2: Bernoulli(pz2)
```

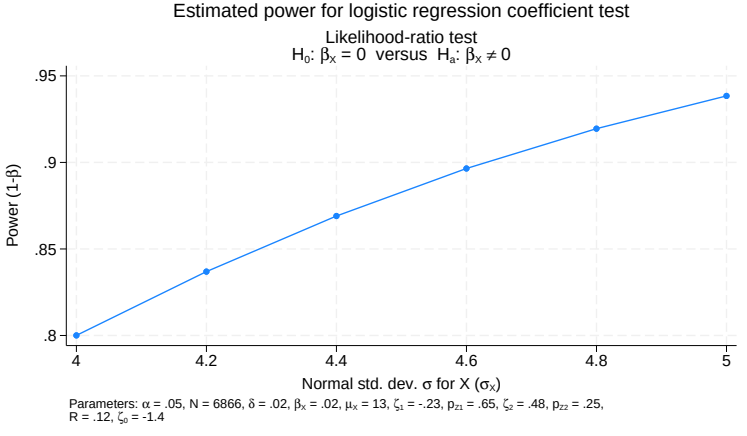


Figure 3. Power curve for a sample of 6,866

With 6,866 participants, the power of our test ranges from 80% when `ttl_exp` has a standard deviation of 4 to over 90% when the standard deviation is greater than 4.6.

Instead of specifying the intercept, we could calculate $\Pr\{Y = 1|X = E(X), \mathbf{Z} = E(\mathbf{Z})\} =$ `invlogit(`$\overline{X}\beta_X + \zeta_0 + \overline{Z}_1\zeta_1 + \overline{Z}_2\zeta_2$`) =` `invlogit(`$12.8 \times 0.02 - 1.38 + 0.65 \times -0.23 + 0.25 \times 0.48$`)` to specify `pycondxmzm(0.23985)`. Alternatively, we could calculate $\Pr\{Y = 1|X = 0, \mathbf{Z} = E(\mathbf{Z})\}$ = `invlogit(`$\zeta_0 + \overline{Z}_1\zeta_1 + \overline{Z}_2\zeta_2$`) =` `invlogit(`$-1.38 + 0.65 \times -0.23 + 0.25 \times 0.48$`)` to specify `pycondx0zm(0.19631)`. Either alternative parameterization will yield the same result; doing so is left as an exercise for the reader.

One detail that bears mentioning is that these power and sample-size calculations are for likelihood-ratio tests, but the `logit` and `logistic` commands report Wald tests of coefficients. Fortunately, an extensive simulation study by Bush (2015) demonstrates that sample-size requirements for Wald and likelihood-ratio tests of logistic regression coefficients are nearly identical. If you prefer a likelihood-ratio test, you can use the `lrtest` command; see [R] **lrtest** for details.

◁

Stored results

`power logistic` in the general case stores the following in `r()`:

Scalars

`r(alpha)`	significance level
`r(power)`	power
`r(beta)`	probability of a type II error
`r(delta)`	effect size
`r(N)`	sample size
`r(nfractional)`	`1` if `nfractional` is specified, `0` otherwise
`r(pycondxmzm)`	success probability of Y given mean values of X and $\mathbf{Z}$
`r(pycondx0zm)`	success probability of Y given $X = 0$ and mean values of covariates $\mathbf{Z}$
`r(intercept)`	intercept from logistic regression
`r(corrxz)`	correlation between X and $\mathbf{Z}$
`r(coefx)`	coefficient for X
`r(oratiox)`	odds ratio for X
`r(unitx)`	unit change in X for odds ratio
`r(nbinsx)`	number of bins for discretized X
`r(coefz#)`	coefficient for $Z_\#$ (if $Z_\#$ is specified)
`r(oratioz#)`	odds ratio for $Z_\#$
`r(unitz#)`	unit change in $Z_\#$ for odds ratio
`r(nbinsz#)`	number of bins for discretized $Z_\#$
`r(ax)`	parameter a of the distribution of X
`r(az#)`	parameter a of the distribution of $Z_\#$
`r(bx)`	parameter b of the distribution of X
`r(bz#)`	parameter b of the distribution of $Z_\#$
`r(mx)`	parameter m of the distribution of X
`r(mz#)`	parameter m of the distribution of $Z_\#$
`r(nx)`	parameter n of the distribution of X
`r(nz#)`	parameter n of the distribution of $Z_\#$
`r(px)`	parameter p of the distribution of X
`r(pz#)`	parameter p of the distribution of $Z_\#$
`r(sx)`	parameter s of the distribution of X
`r(sz#)`	parameter s of the distribution of $Z_\#$
`r(mux)`	parameter μ of the distribution of X
`r(muz#)`	parameter μ of the distribution of $Z_\#$
`r(sigmax)`	parameter σ of the distribution of X
`r(sigmaz#)`	parameter σ of the distribution of $Z_\#$
`r(v#x)`	parameter $v_\#$ of ordinal distribution of X
`r(v#z#)`	parameter $v_\#$ of ordinal distribution of $Z_\#$
`r(p#x)`	parameter $p_\#$ of ordinal distribution of X
`r(p#z#)`	parameter $p_\#$ of ordinal distribution of $Z_\#$
`r(nlx)`	number of levels of ordinal distribution of X
`r(nlz#)`	number of levels of ordinal distribution of $Z_\#$
`r(nbins)`	requested number of bins per covariate (if specified)
`r(minbins)`	minimum requested product of all bins
`r(totalbins)`	actual product of all bins
`r(separator)`	number of lines between separator lines in the table
`r(divider)`	`1` if `divider` is requested in the table, `0` otherwise

r(init)	initial value for odds ratio (if specified)
r(maxiter)	maximum number of iterations (for effect-size calculation)
r(iter)	number of iterations performed (for effect-size calculation)
r(tolerance)	requested parameter tolerance (for effect-size calculation)
r(deltax)	final parameter tolerance achieved (for effect-size calculation)
r(ftolerance)	requested distance of the objective function from zero (for effect-size calculation)
r(function)	final distance of the objective function from zero (for effect-size calculation)
r(converged)	1 if iteration algorithm converged, 0 otherwise (for effect-size calculation)
Macros	
r(type)	test
r(method)	logistic
r(direction)	upper or lower (for effect-size calculation)
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats
r(distx)	distribution of X
r(distz#)	distribution of $Z_{\#}$ (if $Z_{\#}$ is specified)
Matrices	
r(pss_table)	table of results
r(ordinalx)	values and probabilities for ordinal covariate X (if specified)
r(ordinalz#)	values and probabilities for ordinal covariate $Z_{\#}$ (if specified)

Methods and formulas

Methods and formulas are presented under the following headings:

Coefficient tests in logistic regression

Shieh (2000a) used simulation to compare the performance of two sample-size formulas for coefficient tests in logistic regression: the method of Whittemore (1981) and that of Self, Mauritsen, and Ohara (1992). Shieh generalized the superior of those two methods, that of Self, Mauritsen, and Ohara, in Shieh (2000b), which provides the formulas implemented in `power logistic` for power, sample-size, and effect-size calculations for likelihood-ratio tests in logistic regression.

In practice, it is more common to use the Wald test of logistic regression coefficients than the likelihood-ratio test, so there has been some concern about whether these calculations are appropriate for use with a Wald test (Demidenko 2007). Demidenko notes that the Wald and likelihood-ratio tests have asymptotically equivalent type I errors and that they are "*locally* equivalent, so that the power functions are close when the alternative approaches the null" (Demidenko 2007, 3385). Nevertheless, Demidenko raises the point that the two tests are not globally equivalent, so there is no theoretical guarantee that the power functions will be similar under the alternative hypothesis. Thankfully, an extensive simulation study by Bush (2015) found little difference between the power curves of the Wald, likelihood-ratio, and score tests over a range of scenarios. Additionally, Bush compared the performance of seven sample-size formulas for logistic regression and determined that the method of Shieh (2000b) was consistently accurate, regardless of the test that was used.

Logistic regression

The logistic regression can be written as

$$\Pr(y_i = 1|x_i, \mathbf{z}_i) = H(x_i\beta_X + \zeta_0 + \mathbf{z}_i\boldsymbol{\zeta}_Z) \qquad i = 1, 2, \ldots, n$$

where x_i is the observed value of covariate of interest X for subject i, β_X is the X coefficient, ζ_0 is the logistic intercept, $\mathbf{z}_i = (z_{1i}, z_{2i}, \ldots, z_{Ki})$ is the row vector of observed values of K nuisance covariates Z_1 through Z_K for subject i, $\boldsymbol{\zeta}_Z = (\zeta_1, \zeta_2, \ldots, \zeta_K)'$ is the column vector of K coefficients for the nuisance covariates, and n is the sample size. Function $H(\eta) = \{1+\exp(-\eta)\}^{-1}$ is the logistic distribution function.

The effect of covariate X can also be expressed in terms of an odds ratio, $\text{OR}_X = \exp(\beta_X \text{u}_X)$, where $\text{u}_X > 0$ is the unit change in X for the odds ratio. The null hypothesis is $H_0 : \beta_X = 0$, which can also be expressed as $H_0 : \text{OR}_X = 1$. The alternative hypothesis is $H_a : \beta_X \neq 0$ or, equivalently, $H_a : \text{OR}_X \neq 1$.

Parameters $\zeta_1, \zeta_2, \ldots, \zeta_K$ must be specified as coefficients or odds ratios in the respective `z#()` options. Intercept ζ_0 may be specified directly in the `intercept()` option or the information necessary to calculate ζ_0 may be specified using either the `pycondxmzm()` or `pycondx0zm()` option. Coefficient β_X may be specified as a coefficient or odds ratio, or the information necessary to calculate β_X may be specified using the `pycondxmzm()` option. When one or both of `pycondxmzm()` and `pycondx0zm()` are specified, we solve for the unknown parameters (β_X, ζ_0, or both) using the equations

$$\begin{aligned} \Pr\{Y = 1|X = E(X), \mathbf{Z} = E(\mathbf{Z})\} &= H\{E(X)\beta_X + \zeta_0 + E(\mathbf{Z})\boldsymbol{\zeta}_Z\} \\ \Pr\{Y = 1|X = 0, \mathbf{Z} = E(\mathbf{Z})\} &= H\{\zeta_0 + E(\mathbf{Z})\boldsymbol{\zeta}_Z\} \end{aligned}$$

where expected values $E(X)$ and $E(\mathbf{Z})$ are determined based on the specified distributions.

Power, sample-size, and effect-size calculations

Shieh (2000b) builds on the work of Self, Mauritsen, and Ohara (1992) and Self and Mauritsen (1988) to estimate the distribution of the likelihood-ratio statistic: $2\{l(\hat{\beta}_X, \hat{\mathbf{c}}) - l(0, \hat{\mathbf{c}}_0)\}$. Here $l(\cdot)$ is the log-likelihood function for the logistic regression, and $\mathbf{c} = (\zeta_0, \boldsymbol{\zeta}_Z{}')$ is a vector of nuisance parameters. $\hat{\beta}_X$ and $\hat{\mathbf{c}}$ are the maximum likelihood estimates of β_X and $\mathbf{c}$ under the alternative hypothesis, and $\hat{\mathbf{c}}_0$ is the maximum likelihood estimate of $\mathbf{c}$ under the null hypothesis. If the null hypothesis is not true, $\hat{\mathbf{c}}_0$ is not a consistent estimate of $\mathbf{c}$ but instead converges to $\mathbf{c}_0^* = (\zeta_0^*, \boldsymbol{\zeta}_Z{}')$, where $\zeta_0^* = \zeta_0 + E(X)\beta_X$, as described in Self and Mauritsen (1988, eq. 2.2). For a full decomposition of the likelihood-ratio statistic, see Shieh (2000b, 1193).

To calculate sample size, we begin by specifying the desired size of the test (also known as the type I error, α) and the desired power (which equals $1 - \beta$, where β is the desired type II error of the test). We equate the $(1 - \alpha)100$th percentile of a central χ^2 distribution with 1 degree of freedom to the $\beta 100$th percentile of a noncentral χ^2 distribution with noncentrality parameter $\lambda = n\Delta^*$. Here $\Delta^* = 2E(W^*)$, and we define W^* as

$$W^* = H(\eta)(\eta - \eta^*) - \log\{1 + \exp(\eta)\} + \log\{1 + \exp(\eta^*)\}$$

where $\eta = X\beta_X + (1, \mathbf{Z})\mathbf{c}'$ and $\eta^* = (1, \mathbf{Z})\mathbf{c}_0^{*\prime}$. Sample size is calculated as $n = \lambda/\{\Delta^*(1 - R^2)\}$, where R is the coefficient of (multiple) correlation between X and $\mathbf{Z}$. R is specified using the `corrxz()` option, with a default of `corrxz(0)`; the adjustment for correlated covariates is based on Whittemore (1981). Power is computed similarly by starting with a known value of n and solving for the power required to yield the desired value of λ.

There is no closed-form expression that can be used to calculate effect size δ, either coefficient β_X or odds ratio OR_X. Effect size is estimated iteratively, and the default starting value for OR_X is 1.5 with the default `direction(upper)` or 0.67 with `direction(lower)`; see [M-5] `solvenl()` for details. You can use the `init()` option to specify a starting odds ratio for the nonlinear solver. You can control the iteration process with the `iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, and `nodots` options.

Discretization

To approximate the expected value of W^* when calculating Δ^*, we discretize the covariates into bins. Bernoulli, binomial, and ordinal random variables have a fixed number of possible outcomes, so they always use one bin per outcome. For covariates with other distributions, the number of bins is determined as follows. If the `nbins()` option is specified, then the specified number of bins is used. If the `nbins()` suboption is specified in the `distribution()` option, this number overrides the number specified with the `nbins()` option for the specified covariate. All other covariates for which the number of bins was not specified are assigned an equal number of bins, such that the product of bins is greater than or equal to `minbins()`. The default value for `minbins()` is 10,000 for power and sample-size calculations and 1,000 for effect-size calculations. The formula for the total number of bins is $\text{B}_{\text{tot}} = \text{B}_X \prod_{k=1}^{K} \text{B}_{Z_k}$, where $\text{B}_{\text{tot}} \leq 100{,}000{,}000$ is the product of the bins for all covariates, B_X is the number of bins for discretized X, and B_{Z_k} is the number of bins for discretized Z_k.

For Bernoulli, binomial, and ordinal random variables, the probability of each outcome is defined by the distribution. For all other distributions, bins are assigned such that their midpoints yield quantiles of equal probability. To calculate the expectation, we use all B_{tot} possible combinations of binned variables, with each combination weighted by its probability: $E(W^*) \approx \sum_{c=1}^{\text{B}_{\text{tot}}} W_c^* \pi_c$, where W_c^* is W^* calculated at combination c and π_c is the probability of observing combination c, calculated under the assumption of independence between covariates.

References

Bush, S. 2015. Sample size determination for logistic regression: A simulation study. *Communications in Statistics—Simulation and Computation* 44: 360–373. https://doi.org/10.1080/03610918.2013.777458.

Demidenko, E. 2007. Sample size determination for logistic regression revisited. *Statistics in Medicine* 26: 3385–3397. https://doi.org/10.1002/sim.2771.

Hulley, S. B., R. H. Rosenman, R. D. Bawol, and R. J. Brand. 1980. Epidemiology as a guide to clinical decisions—the association between triglyceride and coronary heart disease. *New England Journal of Medicine* 302: 1383–1389. https://doi.org/10.1056/nejm198006193022503.

Self, S. G., and R. H. Mauritsen. 1988. Power/sample size calculations for generalized linear models. *Biometrika* 44: 79–86. https://doi.org/10.2307/2531897.

Self, S. G., R. H. Mauritsen, and J. Ohara. 1992. Power calculations for likelihood ratio tests in generalized linear models. *Biometrika* 48: 31–39. https://doi.org/10.2307/2532736.

Shieh, G. 2000a. A comparison of two approaches for power and sample size calculations in logistic regression models. *Communications in Statistics—Simulation and Computation* 29: 763–791. https://doi.org/10.1080/03610910008813639.

———. 2000b. On power and sample size calculations for likelihood ratio tests in generalized linear models. *Biometrics* 56: 1192–1196. https://doi.org/10.1111/j.0006-341x.2000.01192.x.

Whittemore, A. S. 1981. Sample size for logistic regression with small response probability. *Journal of the American Statistical Association* 76: 27–32. https://doi.org/10.2307/2287036.

Also see

[PSS-2] **power logistic** — Power analysis for logistic regression+

[PSS-2] **power logistic onebin** — Power analysis for logistic regression with one binary covariate+

[PSS-2] **power logistic twobin** — Power analysis for logistic regression with two binary covariates+

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[R] **logistic** — Logistic regression, reporting odds ratios

[R] **logit** — Logistic regression, reporting coefficients

[R] **lrtest** — Likelihood-ratio test after estimation

power cmh — Power and sample size for the Cochran–Mantel–Haenszel test

Description

`power cmh` computes sample size, power, or effect size (the minimum detectable odds ratio) for a Cochran–Mantel–Haenszel (CMH) test of association in stratified 2×2 tables. The command accommodates unbalanced stratum sizes and unbalanced group sizes within each stratum.

Quick start

Sample size for a two-sided CMH test with success probabilities in the control group of 0.2, 0.3, and 0.4 and hypothesized common odds ratio of 3.5 given default power of 0.8 and significance level $\alpha = 0.05$

```
power cmh .2 .3 .4, oratio(3.5)
```

Sample size for an unbalanced design with numbers of subjects in the experimental group as 25% of the subjects in each stratum

```
power cmh .2 .3 .4, oratio(3.5) grratios(.25 .25 .25)
```

Sample size for a one-sided test with the continuity correction applied

```
power cmh .2 .3 .4, oratio(3.5) onesided continuity
```

Power for a total sample size of 120 subjects

```
power cmh .2 .3 .4, oratio(3.5) n(120)
```

Plot of power against total sample size for samples of 100 to 150 in increments of 10

```
power cmh .2 .3 .4, oratio(3.5) n(100(10)150) graph
```

Same as above, but display results in a table instead of a graph

```
power cmh .2 .3 .4, oratio(3.5) n(100(10)150)
```

Minimum detectable odds ratio with 80% power

```
power cmh .2 .3 .4, power(.8) n(120)
```

Same as above, but with 50% of subjects from the first stratum, 30% of subjects from the second stratum, and 20% of subjects from the third stratum

```
power cmh .2 .3 .4, power(.8) n(120) strweights(50 30 20)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
power cmh probspec, oratio(numlist) [power(numlist) options]
```

Compute power

```
power cmh probspec, oratio(numlist) n(numlist) [options]
```

Compute target odds ratio

```
power cmh probspec, n(numlist) power(numlist) [options]
```

where *probspec* is either a matrix *matname* containing the probability of a success in a control group for each stratum or a list of individual stratum probabilities:

$p_{11}\ p_{12}\ \cdots\ p_{1K}$

p_{1k}, where $k = 1,\ 2,\ \ldots,\ K$, is the control-group probability of a success in the kth stratum. Each p_{1k} may be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

matname is the name of a Stata matrix with K columns containing control-group success probabilities. Multiple rows are allowed, in which case each row corresponds to a different set of K strata probabilities or, equivalently, column k corresponds to *numlist* for the control-group success probability in the kth stratum.

options	Description
Main	
* alpha(*numlist*)	significance level; default is alpha(0.05)
* power(*numlist*)	power; default is power(0.8)
* beta(*numlist*)	probability of type II error; default is beta(0.2)
* n(*numlist*)	total sample size; required to compute power or effect size
nfractional	allow fractional sample sizes
* nperstratum(*numlist*)	number of subjects per stratum; implies balanced design
* n#(*numlist*)	number of subjects in stratum #
strweights(*wgtspec*)	stratum weights; default is one for each stratum, meaning equal stratum sizes
* grratios(*grspec*)	stratum-specific group ratios of the experimental-group size to the stratum size, n_{2k}/n_k
* oratio(*numlist*)	common odds ratio of the experimental group to the control group; required to compute power or sample size
continuity	apply continuity correction; default is no continuity correction
direction(upper|lower)	direction of the effect for effect-size determination; default is direction(upper), which means that the postulated value of the parameter is larger than the hypothesized value
onesided	one-sided test; default is two sided
parallel	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[no]table[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
saving(*filename* [, replace])	save the table data to *filename*; use replace to overwrite existing *filename*
Graph	
graph[(*graphopts*)]	graph results; see [PSS-2] **power, graph**
Iteration	
init(#)	initial value for sample size or effect size
iterate(#)	maximum number of iterations; default is iterate(500)
tolerance(#)	parameter tolerance; default is tolerance(1e-12)
ftolerance(#)	function tolerance; default is ftolerance(1e-12)
[no]log	suppress or display iteration log
[no]dots	suppress or display iterations as dots
Reporting	
[no]showgrstrsizes	suppress or display group-per-stratum sizes
showasmatrix	display all sample sizes in a matrix
notitle	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see [U] 11.1.8 numlist. Also see the parallel option.

collect is allowed; see [U] 11.1.10 Prefix commands.

notitle does not appear in the dialog box.

wgtspec	Description
$\#_1\ \#_2\ \ldots\ \#_K$	K stratum weights. Weights must be positive and must be integers unless option nfractional is specified. Multiple values for each stratum weight $\#_k$ can be specified as a *numlist* enclosed in parentheses.
matname	matrix with K columns containing K stratum weights. Multiple rows are allowed, in which case each row corresponds to a different set of K weights or, equivalently, column k corresponds to a *numlist* for the kth weight.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
alpha	significance level	α
power	power	$1-\beta$
beta	type-II-error probability	β
N	total number of subjects	N
N_per_stratum	number of subjects per stratum	N/N_s
N_avg	average number of subjects per stratum	N_{avg}
N#	number of subjects in stratum #	$N_{\#}$
N_per_group	number of subjects per group	$N/2$
G1	number of subjects in control group	G_1
G2	number of subjects in experimental group	G_2
N_per_grstr	number of subjects per group and stratum	$N/(2N_s)$
G1_#	number of subjects in control group and stratum #	$G_{1,\#}$
G2_#	number of subjects in experimental group and stratum #	$G_{2,\#}$
delta	effect size	δ
N_s	number of strata	N_s
oratio	odds ratio	θ
p1_#	#th stratum control-group success probability	$p_{1,\#}$
strwgt#	weight for stratum #	$w_{\#}$
grratio#	ratio of experimental-group size to stratum size for stratum #	$G_{2,\#}/N_{\#}$
target	target parameter; synonym for oratio	
_all	display all supported columns	

Column beta is shown in place of column power in the default table if option beta() is specified.

Column N_per_stratum is shown in the default table only for equal-strata designs; otherwise, columns N# are displayed.

Column N_per_group is shown in the default table only when group sizes are the same; otherwise, columns G1 and G2 are displayed.

Column N_per_grstr is shown in the default table only for balanced designs.

Columns G1_# and G2_# are shown in the table only if requested or if option showgrstrsizes is specified.

Columns strwgt# are shown in the default table only if option strweights() is specified.

Columns grratio# are shown in the default table only if option grratios() is specified.

Options

Main

alpha(), power(), beta(), n(), nfractional; see [PSS-2] **power**.

nperstratum(*numlist*) specifies the stratum size. Only positive integers are allowed. This option implies a balanced-strata design with equal strata sizes. nperstratum() cannot be specified with n(), n#(), or strweights().

n#(*numlist*) specifies the size of the #th stratum. Only positive integers are allowed. All stratum sizes must be specified. For example, all three options n1(), n2(), and n3() must be specified for a design with three strata. n#() cannot be specified with n(), nperstratum(), or strweights().

strweights(*wgtspec*) specifies K stratum weights for an unequal-strata design. The weights may be specified either as a list of values or as a matrix, and multiple sets of weights are allowed; see *wgtspec* for details. The weights must be positive and must also be integers unless the nfractional option is specified. strweights() cannot be specified with nperstratum() or n#().

grratios(*grspec*) specifies K ratios, one for each stratum, of the number of subjects in the experimental group to the number of subjects in the corresponding stratum, n_{2k}/n_k. By default, a balanced group design (or equal numbers of subjects in the control and experimental groups in each stratum) is assumed, which corresponds to setting the K ratios to 0.5.

grspec is similar to *wgtspec* but allows noninteger numbers.

oratio(*numlist*) specifies the alternative value of the common odds ratio of the experimental group to the control group. This option specifies the magnitude of an effect size. It is required to compute power or sample size.

continuity requests that the continuity correction be applied. By default, no continuity correction is applied.

direction(), onesided, parallel; see [PSS-2] **power**.

Table

table, table(), notable; see [PSS-2] **power, table**.

saving(); see [PSS-2] **power**.

Graph

graph, graph(); see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies the initial value for the estimated sample size or effect size when an iterative search is required. When computing the sample size for the two-sided test, the sample-size estimate from the one-sided test is used. The initial estimate for computing the effect size is obtained from a bisection search.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

Reporting

`showgrstrsizes` and `noshowgrstrsizes` displays or suppresses the display of sample sizes in each group and stratum. The default for general output is to display group-per-stratum sizes in a matrix. The default for table output is to suppress the display of group-per-stratum sizes. If you specify this option with table output, group-per-stratum sizes will be displayed in a table as columns. This option has no effect on graphical output.

`showasmatrix` requests that reported sample sizes be displayed as a matrix containing group-per-stratum sizes as cells, and total strata sizes, total group sizes, and a total sample size as marginal totals. This option is not allowed with table or graphical output.

The following option is available with `power cmh` but is not shown in the dialog box:

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `power cmh` command and the methodology for power and sample-size analysis for a CMH test of association in $2 \times 2 \times K$ tables. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis and [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Introduction

Many studies are designed to ascertain the relationship between a binary exposure (exposed or unexposed) and a binary outcome (success of failure). Sometimes the relationship between the two binary variables is influenced by another variable (or variables). One way to adjust for such influence is to stratify on that variable and perform stratified analysis.

For example, a health researcher might conduct a case–control study of birth defects (cases) and mother's medication use during pregnancy (exposure) stratified by mother's age group. A law firm might want to know if there is a relationship between a defendant receiving the death penalty and the defendant's race stratified by the murder victim's race. An education researcher might like to know if students' promotion to the next grade level is more likely after participating in a remedial reading program after stratifying by grade level.

In stratified analysis, a separate 2×2 contingency table is constructed for each stratum $k = 1, 2, \ldots, K$.

	Group		
	Control	Experimental	Total
Failure	a_k	b_k	m_{0k}
Success	c_k	d_k	m_{1k}
Total	n_{1k}	n_{2k}	n_k

The corresponding stratum-specific probabilities of success are π_{1k} and π_{2k} in the control and experimental groups, respectively, and their respective estimates are $p_{1k} = c_k/n_{1k}$ and $p_{2k} = d_k/n_{2k}$. The strength of the association in each table is described by the stratum-specific odds ratio of the experimental group to control group θ_k, estimated as $p_{2k}/(1-p_{2k})/\{p_{1k}/(1-p_{1k})\} = (a_k d_k)/(b_k c_k)$. An overall measure of association between the exposure groups and the outcome across all strata is formed to provide stratified-adjusted inference about the association.

The CMH test (Cochran 1954; Mantel and Haenszel 1959) is commonly used to test for association in stratified 2×2 tables or, equivalently, in $2 \times 2 \times K$ tables. It forms a common odds ratio θ, which can be viewed as a weighted aggregate of stratum-specific odds ratios θ_k to quantify the strength of the association between the exposure and outcome in such stratified analysis. The null hypothesis for this test is $H_0: \pi_{1k} = \pi_{2k}$ for all $k = 1, 2, \ldots, K$. This is equivalent to $H_0: \theta_k = 1$ for all k. The alternative hypothesis is such that there is common odds ratio that is different from one, that is, $H_a: \theta_1 = \theta_2 = \cdots = \theta_K = \theta \neq 1$. Thus, we can rewrite our hypotheses simply as $H_0: \theta = 1$ versus the two-sided alternative $H_a: \theta \neq 1$, an upper one-sided alternative $H_a: \theta > 1$, or a lower one-sided alternative $H_a: \theta < 1$. Statistical significance is assessed using a weighted CMH χ^2 test statistic. Under the null, the distribution of the test statistic is approximately a χ^2 distribution with one degree of freedom. See, for example, Lachin (2011) for more details about this test.

`power cmh` provides power and sample-size computations based on the asymptotic distribution of the test statistic according to Nam (1992); see *Methods and formulas* for details.

Using power cmh

`power cmh` computes sample size, power, or effect size (the minimum detectable odds ratio) for the CMH test of association in $2 \times 2 \times K$ tables. All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying option `alpha()`. You can specify the `onesided` option to request a one-sided test.

To compute the total sample size, you must specify the probabilities of success p_{1k} in the control group in each of the K strata following the command name; the common odds ratio in the `oratio()` option; and, optionally, the power of the test in option `power()`. The default power is set to 0.8.

To compute power, you must specify the total sample size in the `n()` option, the common odds ratio in the `oratio()` option, and the control-group success probabilities p_{1k} following the command name.

To compute effect size, the minimum detectable odds ratio, or the target odds ratio, you must specify the total sample size in `n()`; the power in `power()`; the control-group success probabilities following the command name; and, optionally, the direction of the effect. The direction is upper by default, `direction(upper)`, which means that the common odds ratio is assumed to be larger than one. You can change the direction to be lower, which means that the common odds ratio is assumed to be smaller than one, by specifying the `direction(lower)` option.

There are multiple ways to specify the control-group success probabilities; see *Alternative ways of specifying probabilities*.

By default, all computations assume a design with equal numbers of subjects in each group and each stratum. To change group-specific proportions of subjects in each stratum, use the `grratios(`*grspec*`)` option. A common proportion may be used for all strata or you may specify different proportions for each stratum. Regardless of whether you choose a common proportion or choose to let the proportions vary across strata, you must specify K ratios of cases to the stratum sample size within `grratios()`.

To accommodate unequal stratum sizes for power and effect-size computations, you can specify either individual stratum sizes in options `n1()`, `n2()`, ..., `n`K`()` or a combination of the total sample size in `n()` and integer stratum weights in `strweights(`*wgtpec*`)`. For equal stratum sizes, you can also specify the `nperstratum()` option to specify a common stratum size instead of a total sample size in `n()`.

By default, all computations assume no continuity correction. Use the `continuity` option to change that.

`power cmh` reports group-per-stratum sizes as a matrix in the output. To suppress this matrix, use the `noshowgrstrsizes` option. Alternatively, for the table output, you can use the `showgrstrsizes` option to include columns containing group-per-stratum sizes in the default table.

To make the output of `power cmh` more compact, you may consider using the `showasmatrix` option to display all sample sizes in a matrix.

Sample-size determination for the two-sided test and effect-size determination require iteration. The default initial value for the sample size uses the estimate for the one-sided test. The initial estimate for the effect size is obtained using a bisection search. To specify a different starting value, you may use option `init()`. For more options used to control the iteration process, see [PSS-2] **power**.

Alternative ways of specifying probabilities

There are multiple ways in which you can supply the success probabilities in the control group to `power cmh`.

You may specify each p_{1k} following the command line as

`power cmh` $p_{11}\ p_{12}\ \ldots\ p_{1K}$`,` ...

At least two probabilities must be specified.

When you have many strata, you may find it more convenient to first define a Stata matrix as a row or column vector and use it with `power cmh`. The dimension of the matrix must be at least 2. For example,

`matrix define` *probmat* `=` $(p_{11}, p_{12}, \ldots, p_{1K})$

`power cmh` *probmat*`,` ...

In some cases, you may wish to examine multiple control-group success probabilities in one or more strata. To do this, you can specify multiple values or *numlist* for each of the stratum probabilities in parentheses:

`power cmh` $(p_{11,1}\ p_{11,2}\ \ldots\ p_{11,L1})\ (p_{12,1}\ p_{12,2}\ \ldots\ p_{12,L2})\ \ldots$ `,` ...

Each of the *numlists* may contain different numbers of values, $L_1 \neq L_2 \neq \cdots \neq L_K$. `power cmh` will produce results for all possible combinations of values across *numlists*. Results are presented in a table. If instead you would like to treat each specification as a separate scenario, you may specify the `parallel` option.

You can accommodate multiple sets of stratum probabilities in a matrix form by adding a row for each specification. The columns of a matrix with multiple rows correspond to K strata probabilities, and values within each column k correspond to multiple specifications of the kth stratum probability or a *numlist* for the kth stratum probability.

For example, the following two specifications for three strata with two scenarios each are the same:

`power cmh` $(p_{11,1}\ p_{11,2})\ (p_{12,1}\ p_{12,2})\ (p_{13,1}\ p_{13,2})$`, ...`

and

`matrix define` *probmat* `=` $(p_{11,1}$`,` $p_{12,1}$`,` $p_{13,1} \backslash p_{11,2}$`,` $p_{12,2}$`,` $p_{13,2})$

`power cmh` *probmat*`, ...`

In the above specification, if you wish to specify a *numlist* only for the first stratum, you may define your matrix as

`matrix define` *probmat* `=` $(p_{11,1}$`,` p_{12}`,` $p_{13} \backslash p_{11,2}$`, ., .)`

and the results of

`power cmh` *probmat*`, ...`

will be the same as the results of

`power cmh` $(p_{11,1}\ p_{11,2})\ p_{12}\ p_{13}$`, ...`

In the following sections, we describe the use of `power cmh` accompanied by examples for computing sample size, power, and the minimum detectable odds ratio.

Motivating example

Consider example 4.1 of a clinical trial in duodenal ulcers from Lachin (2011). Blum (1982) describes a hypothetical clinical trial studying the effectiveness of a new drug for treating an ulcer versus placebo. The outcome is whether an ulcer is healed or not, that is, whether the excretion of gastric juices that lead to ulceration of the duodenum is retarded. Three classes of ulcers are considered: acid-dependent (caused by excessive gastric secretion), drug-dependent (caused by excessive use of drugs), and ulcers of intermediate origin. The third class contains ulcers for which the cause is difficult to determine.

The research objective is to evaluate the effectiveness of the treatment on healing ulcers adjusted for ulcer types.

The data are summarized in three 2 × 2 tables, one for each ulcer type.

```
. use https://www.stata-press.com/data/r19/ulcer
(Duodenal ulcers data)
. table (ulcer healed) (treatment) [fw=weight]
```

	Treatment		
	Placebo	Drug	Total
Ulcer type			
Acid-dependent			
Healing status			
Not healed	27	26	53
Healed	20	16	36
Total	47	42	89
Drug-dependent			
Healing status			
Not healed	5	3	8
Healed	4	9	13
Total	9	12	21
Intermediate			
Healing status			
Not healed	28	18	46
Healed	16	28	44
Total	44	46	90
Total			
Healing status			
Not healed	60	47	107
Healed	40	53	93
Total	100	100	200

The stratum-specific proportions of healed ulcers in each group are as follows:

```
. table (ulcer treatment) [fw=weight], statistic(mean healed) nformat(%8.3g)
> nototals
```

	Mean
Ulcer type	
Acid-dependent	
Treatment	
Placebo	.426
Drug	.381
Drug-dependent	
Treatment	
Placebo	.444
Drug	.75
Intermediate	
Treatment	
Placebo	.364
Drug	.609

The common (marginal) odds ratio can be computed by using the cc command. (See [R] **Epitab** and Testing hypotheses about association in $2 \times 2 \times K$ tables for details about this command.)

```
. cc treatment healed [fw=weight]
```

	Exposed	Unexposed	Total	Proportion exposed	
Cases	53	47	100	0.5300	
Controls	40	60	100	0.4000	
Total	93	107	200	0.4650	
	Point estimate		[95% conf.	interval]	
Odds ratio	1.691489		.9299035	3.08095	(exact)
Attr. frac. ex.	.408805		-.0753804	.6754248	(exact)
Attr. frac. pop	.2166667				

chi2(1) = 3.40 Pr>chi2 = 0.0653

The estimate of the marginal odds ratio for these data is 1.691.

Suppose that we would like to conduct a similar study for evaluating a new treatment for healing ulcers. We would like to perform power and sample-size analysis for this new study. In what follows, we demonstrate how to use power cmh to perform these analyses, and we use the results from the current study as our pilot estimates.

Computing sample size

To compute sample size, you must specify the control-group probability of success in each stratum following the command name; the common odds ratio in option oratio(); and, optionally, the power of the test in option power(). A default power of 0.8 is assumed if power() is not specified.

▷ Example 1: Sample size for a two-sided CMH test

Consider a study of drug effectiveness for healing ulcers from *Motivating example*. The data are stratified by an ulcer type. We would like to conduct a new study to evaluate another ulcer treatment versus placebo. We need to compute the required sample size for this study. We use the estimates from the previous ulcer study as our pilot estimates.

The proportions of healed ulcers in the placebo (control) group for each of the three strata were estimated to be 0.426, 0.444, and 0.364, respectively. We anticipate an improvement in the new treatment, and we would like to detect an odds ratio of at least 2.5 with 80% power using a two-sided 5%-level test.

To compute the required sample size, we specify the control-group proportions after the command name and an odds ratio of 2.5 in option oratio(). We omit options alpha(0.05) and power(0.8), because the specified values are their defaults.

```
. power cmh 0.426 0.444 0.364, or(2.5)
Performing iteration ...
Estimated sample size for a test of independence in stratified 2x2 tables
Cochran-Mantel-Haenszel test
H0: OR = 1  versus  Ha: OR != 1;  OR = OR_1 = OR_2 = OR_3
Study parameters:
                alpha =    0.0500
                power =    0.8000
                delta =    2.5000
                  N_s =         3
                 p1_1 =    0.4260
                 p1_2 =    0.4440
                 p1_3 =    0.3640
               oratio =    2.5000
Estimated sample sizes:
                    N =       156
          N per stratum =        52
            N per group =        78
  N per group/stratum =        26
```

A total sample of 156 subjects (78 per group, 52 per stratum, and 26 per each group and stratum) must be obtained to detect an odds ratio of at least 2.5 with 80% power using a 5%-level two-sided CMH test.

Example 2: Unbalanced design, unequal stratum sizes

In example 1, we assumed equal numbers of subjects in each stratum. Based on our pilot data, there are 89 subjects in stratum 1, 21 subjects in stratum 2, and 90 subjects in stratum 3. In other words, the first and the third strata have about 4 times as many subjects as the second stratum.

Let's see how unequal stratum sizes affect our required sample size. We specify strata weights in the `strweights()` option.

```
. power cmh 0.426 0.444 0.364, or(2.5) strweights(4 1 4)
Performing iteration ...
Estimated sample size for a test of independence in stratified 2x2 tables
Cochran-Mantel-Haenszel test
H0: OR = 1  versus  Ha: OR != 1;  OR = OR_1 = OR_2 = OR_3
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    2.5000
          N_s =         3
         p1_1 =    0.4260
         p1_2 =    0.4440
         p1_3 =    0.3640
       oratio =    2.5000
Estimated sample sizes:
            N =       162
           N1 =        72
           N2 =        18
           N3 =        72
  N per group =        81
Group-per-stratum sample sizes:
                 | stratum
                 |         1         2         3
-----------------+---------------------------------
group            |
               1 |        36         9        36
               2 |        36         9        36
```

The required total sample size increases slightly from 156 to 162 with 81 subjects now in each group, which is a slightly larger number than before. Because of unbalanced stratum sizes, the number of subjects per group differs across strata with 36 subjects per group in the first and the third strata and 9 subjects per group in the second stratum.

◁

▷ Example 3: Unbalanced design, unequal group ratios

Continuing with example 2 and our *Motivating example*, we notice that our pilot study also had different numbers of subjects in each group across strata. The actual ratios of the experimental group sizes to the stratum sizes were $42/89 = 0.47$ in stratum 1, $12/21 = 0.57$ in stratum 2, and $46/90 = 0.51$ in stratum 3.

To accommodate unequal group sizes across strata, we can specify stratum-specific group ratios in the `grratios()` option.

```
. power cmh 0.426 0.444 0.364, or(2.5) strweights(4 1 4) grratios(0.47 0.57 0.51)
Performing iteration ...
Estimated sample size for a test of independence in stratified 2x2 tables
Cochran-Mantel-Haenszel test
H0: OR = 1  versus  Ha: OR != 1;  OR = OR_1 = OR_2 = OR_3
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    2.5000
          N_s =         3
         p1_1 =    0.4260
         p1_2 =    0.4440
         p1_3 =    0.3640
       oratio =    2.5000
Estimated sample sizes:
            N =       162
           N1 =        72
           N2 =        18
           N3 =        72
    N group 1 =        80
    N group 2 =        82
Group-per-stratum sample sizes:
```

	stratum		
	1	2	3
group			
1	38	7	35
2	34	11	37

Because all the group ratios are close enough to 0.5 (equal-sized groups), the estimate of the total sample size does not change, and the group sizes change only slightly.

We can investigate the impact of group ratios that are appreciably different from 0.5. For example, for the following three group ratios, the total sample size increases to 207 with 88 subjects in the control group and 119 subjects in the experimental group.

```
. power cmh 0.426 0.444 0.364, or(2.5) strweights(4 1 4) grratios(0.8 0.7 0.3)
Performing iteration ...
Estimated sample size for a test of independence in stratified 2x2 tables
Cochran-Mantel-Haenszel test
H0: OR = 1  versus  Ha: OR != 1;  OR = OR_1 = OR_2 = OR_3
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    2.5000
          N_s =         3
         p1_1 =    0.4260
         p1_2 =    0.4440
         p1_3 =    0.3640
       oratio =    2.5000
Estimated sample sizes:
            N =       207
           N1 =        92
           N2 =        23
           N3 =        92
    N group 1 =        88
    N group 2 =       119
Group-per-stratum sample sizes:
```

	stratum		
	1	2	3
group			
1	18	6	64
2	74	17	28

Computing power

To compute power, you must specify the total sample size in the `n()` option, the common odds ratio in option `oratio()`, and the probabilities of success in the control group following the command name.

▷ Example 4: Power of a two-sided CMH test

Returning to example 1, suppose that we have resources to recruit twice as many subjects, say, 300. To compute power, we add the total number of subjects, 300, in the `n()` option to the syntax from example 1:

```
. power cmh 0.426 0.444 0.364, or(2.5) n(300)
Estimated power for a test of independence in stratified 2x2 tables
Cochran-Mantel-Haenszel test
H0: OR = 1  versus  Ha: OR != 1;  OR = OR_1 = OR_2 = OR_3
Study parameters:
                alpha =    0.0500
                    N =       300
        N per stratum =       100
          N per group =       150
  N per group/stratum =        50
                delta =    2.5000
                  N_s =         3
                 p1_1 =    0.4260
                 p1_2 =    0.4440
                 p1_3 =    0.3640
               oratio =    2.5000
Estimated power:
                power =    0.9759
```

For such an increase in sample size, the power increases dramatically to 0.976 to detect an odds ratio of 2.5.

◁

▷ Example 5: Multiple values of study parameters

We may want to check powers for several sample sizes. Continuing with example 4, we simply list desired sample-size values in option `n(`*numlist*`)`; see **[U] 11.1.8 numlist**.

```
. power cmh 0.426 0.444 0.364, or(2.5) n(150(25)300) table(power N)
Estimated power for a test of independence in stratified 2x2 tables
Cochran-Mantel-Haenszel test
H0: OR = 1  versus  Ha: OR != 1;  OR = OR_1 = OR_2 = OR_3
```

power	N
.7904	150
.8473	175
.8902	200
.9253	225
.9475	250
.9634	275
.9759	300

To shorten our default table, we selected only two varying columns, `power` and total sample size `N`, by specifying them within the `table()` option.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

Computing effect size

Sometimes, we may be interested in determining the smallest odds ratio that yields a statistically significant result for prespecified sample size and power. In this case, we must specify power in option `power()`, sample size in option `n()`, and control-group success probabilities following the command

name. We may also choose the direction of the effect by specifying the direction() option. The default is direction(upper), meaning a common odds ratio greater than one, which corresponds to the improvement of the experimental group over the control group. You can use direction(lower) to request an odds ratio less than one.

▷ Example 6: Minimum detectable odds ratio

In example 4, we learned that with a sample of 300 subjects, the power to detect a common odds ratio of 2.5 is very high. We now want to identify the minimum detectable odds ratio for that study with 80% power.

```
. power cmh 0.426 0.444 0.364, power(0.8) n(300)

Performing iteration ...

Estimated odds ratio for a test of independence in stratified 2x2 tables
Cochran-Mantel-Haenszel test
H0: OR = 1  versus  Ha: OR != 1, OR > 1;  OR = OR_1 = OR_2 = OR_3

Study parameters:

                alpha =    0.0500
                power =    0.8000
                    N =       300
        N per stratum =       100
          N per group =       150
  N per group/stratum =        50
                  N_s =         3
                 p1_1 =    0.4260
                 p1_2 =    0.4440
                 p1_3 =    0.3640

Estimated effect size and odds ratio:

                delta =    1.9192
           odds ratio =    1.9192
```

With a sample of 300 subjects, we can detect an odds ratio of 1.92 with 80% power using a two-sided 5%-level CMH test.

◁

Testing hypotheses about association in $2 \times 2 \times K$ tables

You can test hypotheses about association in $2 \times 2 \times K$ tables using the cc command; see [R] **Epitab**. cc conducts a Mantel–Haenszel test, which is asymptotically equivalent to a Cochran test; see, for example, *Methods and formulas*.

▷ Example 7: Performing Mantel–Haenszel test

Consider the ulcer data from *Motivating example*. We would like to test for association between the treatment and healing of an ulcer stratified by a type of ulcer.

```
. use https://www.stata-press.com/data/r19/ulcer
(Duodenal ulcers data)
. describe
Contains data from https://www.stata-press.com/data/r19/ulcer.dta
 Observations:            12                  Duodenal ulcers data
    Variables:             4                  3 Mar 2024 21:40
```

Variable name	Storage type	Display format	Value label	Variable label
treatment	byte	%9.0g	treatlab	Treatment
healed	byte	%10.0g	heallab	Healing status
ulcer	byte	%14.0g	ulcerlab	Ulcer type
weight	byte	%9.0g		Frequency weights

```
Sorted by:
```

The binary variable `healed` indicates whether an ulcer healed. The binary variable `treatment` indicates whether a subject received a treatment (drug) or placebo. The categorical variable `ulcer` is a stratification variable, which records an ulcer type. The `weight` variable contains the numbers of subjects for each combination of values for `healed` and `treatment` for each of the three strata. This variable will be used as a frequency weight on our analysis.

Here are our data:

```
. list, sepby(ulcer) nolabel
```

	treatm~t	healed	ulcer	weight
1.	1	1	1	16
2.	1	0	1	26
3.	0	1	1	20
4.	0	0	1	27
5.	1	1	2	9
6.	1	0	2	3
7.	0	1	2	4
8.	0	0	2	5
9.	1	1	3	28
10.	1	0	3	18
11.	0	1	3	16
12.	0	0	3	28

We use the `cc` command to test for association between `healed` and `treatment` adjusted for `ulcer` types. We specify `healed` as the dependent variable and `treatment` as the exposure variable after the command name and use the `by(ulcer)` option to stratify on `ulcer`. We also specify the `weight` variable as the frequency weight.

```
. cc healed treatment [fweight=weight], by(ulcer)
```

Ulcer type	Odds ratio	[95% conf.	interval]	M-H weight	
Acid-dependent	.8307692	.32516	2.112608	5.842697	(exact)
Drug-dependent	3.75	.4283095	35.61362	.5714286	(exact)
Intermediate	2.722222	1.069202	6.992856	3.2	(exact)
Crude	1.691489	.9299035	3.08095		(exact)
M-H combined	1.633836	.934329	2.857044		

```
Test of homogeneity (M-H)      chi2(2) =     4.58  Pr>chi2 = 0.1012
                Test that combined odds ratio = 1:
                            Mantel-Haenszel chi2(1) =      3.00
                                              Pr>chi2 =    0.0830
```

The combined Mantel–Haenszel odds ratio equals 1.63, which is not significantly different from 1 at the 5% significance level; the *p*-value is 0.0830. Thus, we do not have sufficient evidence to reject the null hypothesis of no association between the treatment and healing of ulcer after adjusting for ulcer type, at least at the 5% significance level.

◁

Stored results

`power cmh` stores the following in `r()`:

Scalars

`r(alpha)`	significance level
`r(power)`	power
`r(beta)`	probability of a type II error
`r(delta)`	effect size
`r(N)`	total sample size
`r(N_a)`	actual sample size
`r(N_avg)`	average sample size
`r(N#)`	number of subjects in stratum #
`r(N_per_stratum)`	number of subjects per stratum
`r(N_s)`	number of strata
`r(nfractional)`	1 if `nfractional` is specified, 0 otherwise
`r(balanced)`	1 for a balanced design, 0 otherwise
`r(strwgt#)`	stratum weight #
`r(N_per_group)`	number of subjects per group
`r(G1)`	number of subjects in the control group
`r(G2)`	number of subjects in the experimental group
`r(N_per_grstr)`	number of subjects per group and stratum
`r(G1_#)`	number of subjects in the control group in stratum #
`r(G2_#)`	number of subjects in the experimental group in stratum #
`r(grratio#)`	ratio of the experimental-group size to stratum size for stratum #
`r(p1_#)`	control-group probability of success in stratum #
`r(oratio)`	odds ratio of the experimental group to control group
`r(continuity)`	1 if continuity correction is used, 0 otherwise
`r(c)`	continuity-correction value
`r(separator)`	number of lines between separator lines in the table
`r(divider)`	1 if `divider` is requested in the table, 0 otherwise
`r(init)`	initial value for sample size or effect size
`r(maxiter)`	maximum number of iterations
`r(iter)`	number of iterations performed
`r(tolerance)`	requested parameter tolerance
`r(deltax)`	final parameter tolerance achieved

r(ftolerance)	requested distance of the objective function from zero
r(function)	final distance of the objective function from zero
r(converged)	1 if iteration algorithm converged, 0 otherwise

Macros

r(type)	test
r(method)	cmh
r(direction)	upper or lower
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results

Methods and formulas

Consider definitions and a 2×2 contingency table for stratum $k = 1, 2, \ldots, K$ from *Introduction*.

Assume a common odds ratio among all the K tables; that is, $\theta_1 = \theta_2 = \cdots = \theta_K = \theta$ (Woolson, Bean, and Rojas 1986; Nam 1992). To test the hypothesis H_0: $\theta = 1$ versus H_a: $\theta > 1$, Cochran's test statistic may be formed as

$$C = \frac{\sum_{k=1}^{K} \omega_k (p_{2k} - p_{1k})}{\sqrt{\sum_{k=1}^{K} \omega_k \overline{p}_k \overline{q}_k}} = \frac{W}{\sqrt{\widehat{\mathrm{Var}_0(W)}}}$$

where

$$\omega_k = \frac{n_{1k} n_{2k}}{n_k}$$

$$\overline{p}_k = \frac{n_{1k} p_{1k} + n_{2k} p_{2k}}{n_k} \qquad \text{and} \qquad \overline{q}_k = 1 - \overline{p}_k$$

Under the null hypothesis, the statistic C converges to a standard normal distribution. We therefore reject the null hypothesis if $C > z_{1-\alpha}$, where $z_{1-\alpha}$ is the $(1-\alpha)$th quantile of a standard normal distribution.

A similar statistic is the Mantel–Haenszel statistic, which considers a hypergeometric distribution for all fixed marginals for the variance of W under the null hypothesis. Asymptotically, the Cochran test is equivalent to the Mantel–Haenszel test (Armitage, Berry, and Matthews 2002), and thus we refer to this test jointly as the Cochran–Mantel–Haenszel (CMH) test.

Following Nam (1992), we note that the expected value and the variance of W is given, respectively, by

$$E(W) = \sum_{k=1}^{K} \omega_k (\pi_{2k} - \pi_{1k})$$

$$\mathrm{Var}(W) = \sum_{k=1}^{K} \omega_k^2 \left\{ \frac{\pi_{1k}(1-\pi_{1k})}{n_{1k}} + \frac{\pi_{2k}(1-\pi_{2k})}{n_{2k}} \right\}$$

The variance under the null hypothesis is given by

$$\text{Var}(W|H_0) = \text{Var}_0(W) = \sum_{k=1}^{K} \omega_k \overline{\pi}_k (1 - \overline{\pi}_k)$$

For the upper one-sided alternative hypothesis, H_a: $\theta > 1$, the asymptotic power is given by

$$1 - \beta \approx P(C > z_{1-\alpha}) = 1 - \Phi(U_u) \tag{1}$$

where

$$U_u = \frac{z_{1-\alpha}\sqrt{\text{Var}_0(W)} - E(W)}{\sqrt{\text{Var}(W)}}$$

and $\Phi(\cdot)$ is the cumulative of the standard normal distribution.

For the lower one-sided alternative hypothesis, H_a: $\theta < 1$, the asymptotic power is given by

$$1 - \beta \approx P(C \leq z_{\alpha}) = \Phi(U_l) \tag{2}$$

where

$$U_l = \frac{z_{\alpha}\sqrt{\text{Var}_0(W)} - E(W)}{\sqrt{\text{Var}(W)}}$$

When the `onesided` option is used, the direction of the test is determined by the sign of $E(W)$—when $E(W) > 0$, then H_a: $\theta > 1$ and when $E(W) < 0$, then H_a: $\theta < 1$.

Then, the power of the test is given by

$$\pi = \begin{cases} 1 - \Phi(U_u) & \text{for an upper one-sided test} \\ \Phi(U_l) & \text{for a lower one-sided test} \\ 1 - \Phi(U_u) + \Phi(U_l) & \text{for a two-sided test} \end{cases}$$

The two-sided test uses $\alpha/2$ in the definitions of U_u and U_l.

The sample size for a two-sided test and the minimum detectable odds ratio θ must be computed iteratively from the corresponding power equations. The starting value for the sample-size computation is the sample-size estimate for the one-sided test. For the minimum detectable odds ratio, the starting value is obtained from a bisection algorithm.

When strata weights w_k are specified in the `strweights()` option, a constant multiplier n_c is computed and rounded to an integer unless the `nfractional` option is specified. The stratum sizes are then computed as $\widetilde{w}_j n_c$, where $\widetilde{w}_k$ are normalized strata weights. The actual sample size, `N_a`, is the sum of the stratum sizes. The numbers of subjects in each group and stratum are computed as $n_k/2$ for equal-group designs and as $n_{2k} = n_k \times \#_k$ and $n_{1k} = n_k - n_{2k}$ when group ratios $\#_k$s are supplied in `grratios()`. Unless the `nfractional` option is specified, the experimental-group sizes corresponding to unbalanced strata are rounded up. For strata that have equal numbers of subjects in each group, the group-per-stratum sizes are not rounded.

Woolson, Bean, and Rojas (1986) provide a formula for the sample size that is required to attain a specific power by substituting $U_u = z_\beta = \Phi^{-1}(\beta)$ and $U_l = z_{1-\beta} = \Phi^{-1}(1-\beta)$ in (1) and (2) and solving for n:

$$n = \begin{cases} \left(\frac{z_{1-\alpha}\sqrt{X}+z_{1-\beta}\sqrt{Y}}{Z}\right)^2 & \text{if } E(W) > 0 \\ \left(\frac{z_{\alpha}\sqrt{X}+z_{\beta}\sqrt{Y}}{Z}\right)^2 & \text{if } E(W) < 0 \end{cases}$$

where

$$X = \sum_{k=1}^{K} w_k s_k (1-s_k)\overline{\pi}_k(1-\overline{\pi}_k)$$

$$Y = \sum_{k=1}^{K} w_k s_k (1-s_k)\left\{(1-s_k)\pi_{2k}(1-\pi_{2k}) + s_k\pi_{1k}(1-\pi_{1k})\right\}$$

$$Z = \sum_{k=1}^{K} w_k s_k (1-s_k)(\pi_{2k}-\pi_{1k})$$

and $w_k = n_k/n$ is the fraction of the individual-stratum size to the total sample size, and $s_k = n_{2k}/n_k$ is the fraction of the number of subjects in the experimental group to the sample size of each stratum. Sample size for the two-sided alternative hypothesis is solved iteratively.

The continuity-corrected version of U_u and U_l is

$$U_u = \frac{z_{1-\alpha}\sqrt{\text{Var}_0(W)} - E(W) + \frac{1}{2}}{\sqrt{\text{Var}(W)}} \qquad \text{if } E(W) > 0$$

$$U_l = \frac{z_{\alpha}\sqrt{\text{Var}_0(W)} - E(W) - \frac{1}{2}}{\sqrt{\text{Var}(W)}} \qquad \text{if } E(W) < 0$$

Nam (1992) derives the sample size for the continuity-corrected version of the CMH test, n_c, as follows:

$$n_c = \begin{cases} \frac{n}{4}\left(1+\sqrt{1+\frac{2}{nZ}}\right)^2 & \text{if } E(W) > 0 \\ \frac{n}{4}\left(1+\sqrt{1-\frac{2}{nZ}}\right)^2 & \text{if } E(W) < 0 \end{cases}$$

where n is the uncorrected sample size.

For the computation, π_{1k} and π_{2k} are replaced with their estimates p_{1k} and p_{2k} in the above formulas. p_{1k}s are supplied directly to `power cmh`, and p_{2k}s are computed using the specified $p_{1k}s$ and odds ratio in `oratio()`.

References

Armitage, P., G. Berry, and J. N. S. Matthews. 2002. *Statistical Methods in Medical Research*. 4th ed. Oxford: Blackwell.

Blum, A. L. 1982. "Principles for selection and exclusion". In *The Randomized Clinical Trial and Therapeutic Decisions*, edited by N. Tygstrup, J. M. Lachin, and E. Juhl, 43–58. New York: Dekker.

Cochran, W. G. 1954. Some methods for strengthening the common chi-squared tests. *Biometrics* 10: 417–451. https://doi.org/10.2307/3001616.

Lachin, J. M. 2011. *Biostatistical Methods: The Assessment of Relative Risks*. 2nd ed. Hoboken, NJ: Wiley.

Mantel, N., and W. Haenszel. 1959. Statistical aspects of the analysis of data from retrospective studies of disease. *Journal of the National Cancer Institute* 22: 719–748. Reprinted in *Evolution of Epidemiologic Ideas: Annotated Readings on Concepts and Methods*, ed. S. Greenland, pp. 112–141. Newton Lower Falls, MA: Epidemiology Resources.

Nam, J. 1992. Sample size determination for case–control studies and the comparison of stratified and unstratified analyses. *Biometrics* 48: 389–395. https://doi.org/10.2307/2532298.

Woolson, R. F., J. A. Bean, and P. B. Rojas. 1986. Sample size for case–control studies using Cochran's statistic. *Biometrics* 42: 927–932. https://doi.org/10.2307/2530706.

Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[R] **Epitab** — Tables for epidemiologists

power mcc — Power analysis for matched case–control studies

Description

`power mcc` computes sample size, power, or effect size (the minimum detectable odds ratio) for a test of association between a risk factor and a disease in 1:M matched case–control studies.

Quick start

Number of cases for a test with exposure probability for controls of 0.2 and odds ratio for exposure of 1.4 from a 1:1 matched design using default power of 0.8 and significance level $\alpha = 0.05$

```
power mcc .2, oratio(1.4)
```

Same as above, but for a 1:2 matched design and compute the ratio of the number of cases for a 1:2 matched design relative to a 1:1 matched design

```
power mcc .2, oratio(1.4) m(2) compare
```

Number of cases when correlation of exposure between matched pairs is 0.3

```
power mcc .2, oratio(1.4) corr(.3)
```

Power for 500 cases and a 1:1 matched design

```
power mcc .2, oratio(1.4) n(500)
```

Plot of power against the number of cases for 450, 475, 500, 525, and 550 cases

```
power mcc .2, oratio(1.4) n(450(25)550) graph
```

Minimum detectable odds ratio with 80% power and 500 cases using a 1:1 matched design

```
power mcc .2, power(.8) n(500)
```

Same as above, but for an upper one-sided test

```
power mcc .2, power(.8) n(500) onesided direction(upper)
```

Same as above

```
power mcc .2, power(.8) n(500) onesided
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
power mcc p0, oratio(numlist) [power(numlist) options]
```

Compute power

```
power mcc p0, oratio(numlist) n(numlist) [options]
```

Compute target odds ratio

```
power mcc p0, power(numlist) n(numlist) [options]
```

where p_0 is the probability of exposure among control patients. p_0 must satisfy the condition $0 < p_0 < 1$ and may be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

options	Description
Main	
* alpha(*numlist*)	significance level; default is alpha(0.05)
* power(*numlist*)	power; default is power(0.8)
* beta(*numlist*)	probability of type II error; default is beta(0.2)
* n(*numlist*)	sample size; required to compute power or effect size
nfractional	allow fractional sample size
* oratio(*numlist*)	odds ratio of exposure in cases relative to controls; required to compute power or sample size
* m(*numlist*)	number of matched controls per case; default is m(1)
compare	ratio of the required number of cases for a 1:M design relative to a paired 1:1 design
* corr(*numlist*)	correlation of exposure between cases and controls; default is corr(0)
direction(upper|lower)	direction of the effect for effect-size determination; default is direction(upper), which means that the postulated value of the parameter is larger than the hypothesized value
onesided	one-sided test; default is two sided
parallel	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[no]table[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
saving(*filename* [, replace])	save the table data to *filename*; use replace to overwrite existing *filename*
Graph	
graph[(*graphopts*)]	graph results; see [PSS-2] **power, graph**
Iteration	
init(#)	initial value for sample size or effect size
iterate(#)	maximum number of iterations; default is iterate(500)
tolerance(#)	parameter tolerance; default is tolerance(1e-12)
ftolerance(#)	function tolerance; default is ftolerance(1e-12)
[no]log	suppress or display iteration log
[no]dots	suppress or display iterations as dots
notitle	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see [U] **11.1.8 numlist**. Also see the parallel option.

collect is allowed; see [U] **11.1.10 Prefix commands**.

notitle does not appear in the dialog box.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`N`	number of cases	N
`delta`	effect size	δ
`M`	number of matched controls	M
`F_M`	ratio of the number of cases with M controls relative to one control	F_M
`p0`	probability of exposure among controls	p_0
`p1`	probability of exposure among cases	p_1
`oratio`	odds ratio	θ
`corr`	correlation of exposure between cases and controls	ρ
`target`	target parameter; synonym for `oratio`	
`_all`	display all supported columns	

Column `beta` is shown in the default table in place of column `power` if option `beta()` is specified.

Column `F_M` is shown in the default table only if option `compare` is specified.

Column `p1` is not shown in the default table.

Options

Main

`alpha()`, `power()`, `beta()`, `n()`, `nfractional`; see [PSS-2] **power**. The sample size in `n()` is the number of matched case–control sets or, equivalently, the number of cases. The `nfractional` option is allowed only for sample-size determination.

`oratio(`*numlist*`)` specifies the odds ratio of exposure in cases relative to controls. This option is required for power or sample-size determination and may not be specified for effect-size determination.

`m(`*numlist*`)` specifies the number of matched controls per case. Only positive integers are allowed. The default is `m(1)`, which implies a paired design.

`compare` specifies that the ratio, F_M, of the required number of cases for a 1:M design relative to a paired 1:1 design be computed. `compare` can be specified only when computing sample size and when a value of 2 or greater is specified in option `m()`.

`corr(`*numlist*`)` specifies the correlation coefficient for exposure ρ between matched cases and controls. `corr()` must contain numbers between -1 and 1. The default is `corr(0)`, meaning no correlation between matched cases and controls. This assumption may not be realistic in practice; see example 3 for discussion.

`direction()`, `onesided`, `parallel`; see [PSS-2] **power**.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

graph, graph(); see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

init(#) specifies the initial value for the estimated sample size or effect size when an iterative search is required. When computing the sample size for the two-sided test, the closed-form sample-size computation for the one-sided test is used. The initial estimate for computing the minimum detectable odds ratio is obtained from a bisection search.

iterate(), tolerance(), ftolerance(), log, nolog, dots, nodots; see [PSS-2] **power**.

The following option is available with power mcc but is not shown in the dialog box:

notitle; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the power mcc command and the methodology for power and sample-size analysis for 1:M matched case–control studies. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis and [PSS-2] **power** for a general introduction to the power command using hypothesis tests.

Introduction

Matched case–control studies investigate the relationship between disease and exposure, controlling for the effect of confounding variables. Cases are observations that have the outcome of interest; controls are observations that do not. Cases are matched to the controls on the basis of similar values of the variable or variables thought to confound the relationship between exposure and disease.

Matched case–control studies are used to investigate a variety of outcomes. A pediatrician might be interested in the relationship between low birthweight (case) and mother's smoking status during pregnancy (exposure), where case and control mothers are matched on the basis of age, race, alcohol consumption, and history of hypertension (confounding variables). An oncologist might want to know if women with endometrial cancer are more likely to have taken estrogen, where cases and controls are matched on age, marital status, and time living in the community. A psychologist might design a study to see if suicide is more prevalent among patients who used a particular antidepressant, where cases and controls are matched on age, sex, race, severity of depression symptoms, and history of head injury.

This entry describes power and sample-size analysis for inference about correlated binary outcomes. In a 1:M matched case–control study, we first randomly select cases from a population of cases and observe their exposure status. The population is then stratified by the confounding (matching) variables, and for each selected case, M matched controls are randomly selected from the same stratum as the

selected case. This creates a series of 2×2 contingency tables within the stratum defined by the matching covariates. Each contingency table summarizes the probability of observing each exposure–outcome combination within a given stratum, and the probabilities are assumed to be equal across strata (or, equivalently, odds ratios are assumed to be equal across strata) so that for any study we need to analyze only one table.

	Control		
Case	Exposed	Unexposed	Total
Exposed	p_{11}	p_{10}	p_1
Unexposed	p_{01}	p_{00}	q_1
Total	p_0	q_0	1

The concordant probabilities lie on the diagonal; p_{11} is the probability that an exposed case subject is matched to an exposed control subject, and p_{00} is the probability that an unexposed case subject is matched to an unexposed control subject. The target parameter is the odds ratio θ of developing the disease in exposed and unexposed subjects who have equal values of matching variables. It can be calculated from the discordant probabilities as p_{10}/p_{01}.

The probability of exposure for controls, p_0, is the probability that the sampled control subject is exposed and is simply the sum of p_{11} and p_{01}. The probability of exposure for cases, p_1, is the probability that the sampled case subject is exposed and is simply the sum of p_{11} and p_{10}.

The two-sided hypothesis test for association between disease and exposure can be formally stated in terms of the odds ratio as H_0: $\theta = 1$ versus H_a: $\theta \neq 1$. We can equivalently state the hypothesis test in terms of marginal homogeneity: H_0: $p_0 = p_1$ versus H_a: $p_0 \neq p_1$.

In a matched case–control study, n cases are sampled and then matched to M controls. When $M = 1$, a 1:1 matched design or paired design, there are n matched pairs and $n \times 2$ total subjects. When $M > 1$, there are n matched sets and $n \times (M + 1)$ total subjects. Unlike a study without matching, a matched case–control study does not have $n \times 2$ independent observations [or $n \times (M + 1)$ for matched designs with multiples controls].

All calculations performed by `power mcc` treat n as the relevant sample size. Throughout the remainder of this entry, when we refer to the sample size, we mean n, the number of cases and thus the number of matched pairs (or sets).

Using power mcc

`power mcc` computes sample size, power, or effect size (the minimum detectable odds ratio) for 1:M matched case–control studies, in which one case is matched to M controls. All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option. You can specify the `onesided` option to request a one-sided test.

To compute sample size, you must specify the probability of exposure for the control group p_0; the odds ratio for exposure θ in option `oratio()`; and, optionally, the power of the test in the `power()` option. The default power is set to 0.8. The sample-size estimate returned is the number of matched pairs or, if option `m()` was specified, the number of matched sets. This is equivalent to the number of cases. Hereafter, we simply refer to the number of cases.

To compute power, you must specify the sample size in option `n()`, the probability of exposure for the control group p_0, and the odds ratio in option `oratio()`.

To compute the minimum detectable odds ratio, you must specify the sample size in option `n()`; the power in option `power()`; the probability of exposure for the control group p_0; and, optionally, the direction of the effect. The direction is upper by default, `direction(upper)`, which means that the probability of exposure among cases is assumed to be larger than the specified control-group value. You can change the direction to be lower, which means that the probability of exposure among cases is assumed to be lower than the specified control-group value, by specifying the `direction(lower)` option. `power mcc` defines the effect size as the target odds ratio.

By default, all computations assume a 1:1 or paired design, in which one case is matched to one control. You may specify the `m()` option to accommodate multiple matches per case.

The correlation between the matched case–control subjects is set to 0 by default but may be changed by specifying option `corr()`.

For sample-size determination, you can specify the `compare` option to compute the ratio of the required number of cases when using M matched controls rather than one.

Sample-size determination for the two-sided test and effect-size determination for $M > 1$ require iteration. The default initial sample-size value is set to the closed-form one-sided sample size. The initial value for the effect size is computed using a bisection algorithm. You can use the `init()` option to specify your own value. See [PSS-2] **power** for a description of other options that control the iteration process.

In the following sections, we describe the use of `power mcc` accompanied by examples for computing sample size, power, and the minimum detectable odds ratio.

Computing sample size

To compute sample size, you must specify the probability of exposure among control patients p_0 after the command name; the odds ratio θ in option `oratio()`; and, optionally, the power in option `power()`.

▷ Example 1: Sample size for a 1:1 matched case–control study

Consider a study comparing the odds of developing lung cancer among smokers with the odds among nonsmokers. Suppose that previous studies matching smokers and nonsmokers on the basis of age, gender, race, and alcohol consumption found the following proportions:

	No Lung Cancer (Control)		
Lung Cancer (Case)	Smoker	Nonsmoker	Total
Smoker	0.180	0.144	0.324
Nonsmoker	0.040	0.636	0.676
Total	0.220	0.780	1

If we wish to plan a new case–control study, we might assume that these proportions represent population probabilities and, therefore, let $p_0 = 0.22$. Under the assumption of no correlation of exposure in matched pairs, $\theta = (0.324 \times 0.78)/(0.22 \times 0.676) = 1.7$.

We would like to determine the number of case–control pairs that we will need to achieve 80% power to detect an odds ratio of 1.7 with a 5%-level two-sided test.

To compute the required sample size, we specify 0.22 as the probability of exposure for the control group after the command name and specify 1.7 as the odds ratio in option `oratio()`. We omit options `alpha(0.05)` and `power(0.8)` because the specified values are their defaults.

```
. power mcc .22, oratio(1.7)
Performing iteration ...
Estimated sample size for a matched case-control study
Asymptotic z test, 1:1 matched design
H0: OR = 1  versus  Ha: OR != 1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    1.7000
           p0 =    0.2200
       oratio =    1.7000
         corr =    0.0000
            M =         1
Estimated sample size:
      N cases =       285
```

Our calculation indicates that we will need a sample of 285 cases to detect an odds ratio of 1.7 with 80% power using a 5%-level test.

◁

▷ Example 2: Sample size for a 1:M matched case–control study

Multiple controls are often matched with one case to increase the efficiency of the study. Continuing with example 1, we note that we have access to many more control participants than case participants.

We specify option `m(2)` to recalculate our sample size assuming that we will match two controls with each case (a 1:2 matched design). We also specify the `compare` option to calculate the ratio of the number of required cases relative to the 1:1 paired design.

```
. power mcc .22, oratio(1.7) m(2) compare
Performing iteration ...
Estimated sample size for a matched case-control study
Asymptotic z test, 1:2 matched design
H0: OR = 1  versus  Ha: OR != 1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    1.7000
           p0 =    0.2200
       oratio =    1.7000
         corr =    0.0000
            M =         2
Estimated sample size:
      N cases =       210
          F_M =    0.7368
```

We obtain a new sample-size estimate of 210 cases, and the reduction in the number of cases relative to the paired design is approximately 26% ($1 - \texttt{F\_M} = 0.2632$).

◁

Example 3: Sample size with correlated exposures

Matching based on confounders will typically lead to correlation of the exposure between cases and controls. For example, smoking status is known to be correlated with alcohol consumption. Matching on alcohol consumption might, therefore, result in the cases and controls being more similar with regard to smoking status. Ignoring this correlation will lead to underestimation of the required sample size or overestimation of power.

The correlation coefficient ρ for exposure between cases and controls can be computed from the probabilities in our contingency table; see (1) in *Methods and formulas*.

Returning to example 1, we compute the correlation of exposure:

$$\rho = (0.180 \times 0.636 - 0.144 \times 0.040)/\sqrt{0.324 \times 0.676 \times 0.220 \times 0.780} = 0.56$$

We then specify its value in the `corr()` option.

```
. power mcc .22, oratio(1.7) corr(.56)
Performing iteration ...
Estimated sample size for a matched case-control study
Asymptotic z test, 1:1 matched design
H0: OR = 1  versus  Ha: OR != 1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    1.7000
           p0 =    0.2200
      oratio =    1.7000
        corr =    0.5600
           M =         1
Estimated sample size:
      N cases =       703
```

With such a high level of correlation between exposure and the matching variables, our sample size increases dramatically to 703 cases.

This examples demonstrates the importance of taking into account the correlation of exposure between matched cases and controls. For this reason, Dupont (1988) recommends using the value of, say, 0.2 in computations instead of making an independence assumption, which is unlikely to hold in practice.

◁

Example 4: Multiple values of study parameters

Continuing with example 3, suppose that we believe that the previously gathered data may provide a good estimate of exposure probability for controls p_0 and the odds ratio θ, but that the individual cell proportions are not precise enough estimates of the population probabilities to produce a good estimate of ρ.

In this case, we may want to use a range of plausible values and consider how it affects the sample size for our study. We can specify a range of correlations between 0.4 and 0.6 with a step size of 0.05 using standard *numlist* (see **[U] 11.1.8 numlist**) notation in option `corr()`.

```
. power mcc .22, oratio(1.7) corr(.4(.05).6)
Performing iteration ...
Estimated sample size for a matched case-control study
Asymptotic z test, 1:1 matched design
H0: OR = 1  versus  Ha: OR != 1
```

alpha	power	N	delta	M	p0	oratio	corr
.05	.8	503	1.7	1	.22	1.7	.4
.05	.8	553	1.7	1	.22	1.7	.45
.05	.8	613	1.7	1	.22	1.7	.5
.05	.8	687	1.7	1	.22	1.7	.55
.05	.8	779	1.7	1	.22	1.7	.6

For a given power, the required sample size increases as the correlation ρ increases. In this example, the choice of the correlation has a large effect on the required sample size, which suggests that we should carefully consider the choice of matching variables.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

Computing power

To compute power, you must specify the number of cases in option `n()`, the exposure probability among controls p_0 following the command name, and the odds ratio in option `oratio()`.

▷ Example 5: Power for matched case–control studies

Returning to example 1, we discover that we are able to recruit 300 cases for our study. To compute the corresponding power, we specify 300 as the sample size in option `n()`.

```
. power mcc .22, oratio(1.7) n(300)
Estimated power for a matched case-control study
Asymptotic z test, 1:1 matched design
H0: OR = 1  versus  Ha: OR != 1
Study parameters:
        alpha =    0.0500
      N cases =       300
        delta =    1.7000
           p0 =    0.2200
       oratio =    1.7000
         corr =    0.0000
            M =         1
Estimated power:
        power =    0.8204
```

As expected, with a larger sample size, this example has a higher power (about 82%) than example 1.

◁

▷ Example 6: Power for a one-sided test

Continuing with example 5, suppose that we are interested in testing whether the odds ratio is greater than 1 because we hypothesize that a history of smoking will lead to an increased incidence of lung cancer. In this case, we can specify the `onesided` option to calculate power for a one-sided test.

```
. power mcc .22, oratio(1.7) n(300) onesided

Estimated power for a matched case-control study
Asymptotic z test, 1:1 matched design
H0: OR = 1  versus  Ha: OR > 1

Study parameters:

        alpha =    0.0500
      N cases =       300
        delta =    1.7000
           p0 =    0.2200
       oratio =    1.7000
         corr =    0.0000
            M =         1

Estimated power:

        power =    0.8931
```

As expected, the power of the one-sided is higher (89%) than the power of the corresponding two-sided test.

◁

Computing target odds ratio

Sometimes, we may be interested in determining the smallest effect that will yield a statistically significant result for a prespecified sample size and power. In this case, power, sample size, and the exposure probability among controls must be specified. The effect size in `power mcc` is expressed as an odds ratio of exposure in cases relative to controls.

▷ Example 7: Minimum detectable odds ratio with 1:1 matching

Continuing with example 5, we now would like to calculate the minimum detectable odds ratio that we can identify with the study design we have planned and knowing that we will be able to recruit 300 cases. We again specify 300 cases in option `n()` and also specify 80% power by using option `power(0.8)`:

```
. power mcc .22, n(300) power(.8)

Performing iteration ...

Estimated odds ratio for a matched case-control study
Asymptotic z test, 1:1 matched design
H0: OR = 1  versus  Ha: OR != 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
      N cases =       300
           p0 =    0.2200
         corr =    0.0000
            M =         1

Estimated effect size and odds ratio:

        delta =    1.6783
   odds ratio =    1.6783
```

Our minimum detectable odds ratio is about 1.68 for this study design.

◁

▷ Example 8: Minimum detectable odds ratio with 1:M matching

We can specify other options to tailor our estimates of the minimum detectable odds ratio to accommodate other study design parameters. Continuing with example 7, we may wish to calculate the minimum detectable odds ratio if we adopt a 1:2 matched design.

```
. power mcc .22, n(300) power(.8) m(2)
Performing iteration ...
Estimated odds ratio for a matched case-control study
Asymptotic z test, 1:2 matched design
H0: OR = 1  versus  Ha: OR != 1
Study parameters:
        alpha =    0.0500
        power =    0.8000
      N cases =       300
           p0 =    0.2200
         corr =    0.0000
            M =         2
Estimated effect size and odds ratio:
        delta =    1.5656
   odds ratio =    1.5656
```

Our minimum detectable odds ratio estimate decreases to 1.57 from 1.68 when the number of cases is held constant at 300.

Comparing results with example 7, we see that increasing M while holding power and sample size constant decreases the minimum detectable odds ratio. Consequently, increasing M while holding sample size and the odds ratio constant increases power.

◁

Testing hypotheses in matched case–control studies

Matched case–control data can be organized in two ways: long and wide format. In long format, each row corresponds to a person; this is the format used by `clogit` and `mhodds` (see [R] **clogit** and [R] **Epitab**). In wide format, each row corresponds to a matched pair or set; this is the format used by `mcc` (see [R] **Epitab**).

▷ Example 9: Analysis of matched case–control data in long format

Hosmer, Lemeshow, and Sturdivant (2013) describe a study in which low birthweight infants were matched with normal weight infants on the basis of the age of the mother. The mothers were then asked whether or not they smoked during pregnancy. We will list a subset of these data to illustrate long format.

```
. use https://www.stata-press.com/data/r19/lowbirth2
(Applied Logistic Regression, Hosmer & Lemeshow)
. list pairid low smoke in 1/6, sepby(pairid)
```

	pairid	low	smoke
1.	1	0	0
2.	1	1	1
3.	2	0	0
4.	2	1	0
5.	3	0	0
6.	3	1	0

The first column is the case–control identifier for each pair of infants. The second column identifies each infant as a case (`low==1`, indicating low birthweight) or a control (`low==0`). The third column identifies each infant as exposed (`smoke==1`, indicating that the mother smoked) or not exposed (`smoke==0`). We can estimate the odds ratio and test the null hypothesis that it equals 1 by using `clogit`.

We must use option `group()` to specify the identifier for our case–control matched pairs. We also specify option `nolog` to suppress the iteration log and `or` to view the result as an odds ratio.

```
. clogit low smoke, group(pairid) nolog or
Conditional (fixed-effects) logistic regression         Number of obs =    112
                                                        LR chi2(1)    =   6.79
                                                        Prob > chi2   = 0.0091
Log likelihood = -35.419282                             Pseudo R2     = 0.0875
```

low	Odds ratio	Std. err.	z	P>|z|	[95% conf.	interval]
smoke	2.75	1.135369	2.45	0.014	1.224347	6.176763

The estimated odds ratio is 2.75. We reject the null hypothesis that mothers who smoke and mothers who do not smoke have equal odds of giving birth to an infant with low birthweight at the 5%-level ($p = 0.014$).

◁

▷ Example 10: Analysis of paired case–control data in wide format

Continuing with example 9, because the Hosmer, Lemeshow, and Sturdivant (2013) data are for a study with matched pairs, we could also conduct a classic McNemar's test. In Stata, McNemar's test is calculated by the `mcc` command; see [R] **Epitab**. The `mcc` command, however, requires that the data be in wide form. For details, see the technical note in [R] **clogit**.

We can reshape our low birthweight data using the `reshape` command.

```
. keep low smoke pairid
. reshape wide smoke, i(pairid) j(low 0 1)
Data                                Long   ->   Wide
-----------------------------------------------------------------------------
Number of observations               112   ->   56
Number of variables                    3   ->   3
j variable (2 values)                low   ->   (dropped)
xij variables:
                                   smoke   ->   smoke0 smoke1
-----------------------------------------------------------------------------
```

```
. list pairid smoke1 smoke0 in 1/3

     +-------------------------+
     | pairid   smoke1   smoke0 |
     |-------------------------|
  1. |      1        1        0 |
  2. |      2        0        0 |
  3. |      3        0        0 |
     +-------------------------+
```

The variable `smoke1` indicates whether or not the case mother smoked, and the variable `smoke0` indicates whether or not the control mother smoked. We can now use `mcc` to estimate the overall odds ratio.

```
. mcc smoke1 smoke0
                    |          Controls
Cases               |  Exposed   Unexposed  |      Total
--------------------+-----------------------+-----------
           Exposed  |         8          22 |         30
         Unexposed  |         8          18 |         26
--------------------+-----------------------+-----------
              Total |        16          40 |         56

McNemar's chi2(1) =      6.53    Prob > chi2 = 0.0106
Exact McNemar significance probability       = 0.0161

Proportion with factor
        Cases       .5357143
        Controls    .2857143     [95% conf. interval]
                    ---------    --------------------

        difference        .25      .0519726   .4480274
        ratio            1.875     1.148685   3.060565
        rel. diff.         .35      .1336258   .5663742

        odds ratio        2.75     1.179154   7.143667   (exact)
```

Again, the estimated odds ratio θ is 2.75, and McNemar's χ^2 is 6.53. As with the analysis using `clogit`, we reject the null hypothesis of equal odds at the 5% significance level ($p = 0.0106$). The confidence interval for the odds ratio calculated by `clogit` and `mcc` differ slightly due to different estimation methods.

We could also calculate the same test statistic based on symmetry and marginal homogeneity; see [R] **symmetry** for further details.

◁

Stored results

power mcc stores the following in r():

Scalars

r(alpha)	significance level
r(power)	power
r(beta)	probability of a type II error
r(delta)	effect size
r(N)	sample size
r(nfractional)	1 if nfractional is specified, 0 otherwise
r(onesided)	1 for a one-sided test, 0 otherwise
r(p0)	probability of exposure among controls
r(M)	number of matched controls per case
r(F_M)	ratio of the number of cases relative to the 1:1 paired design
r(oratio)	odds ratio
r(corr)	correlation of exposure between matched cases and controls
r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise
r(init)	initial value for sample size or effect size
r(maxiter)	maximum number of iterations
r(iter)	number of iterations performed
r(tolerance)	requested parameter tolerance
r(deltax)	final parameter tolerance achieved
r(ftolerance)	requested distance of the objective function from zero
r(function)	final distance of the objective function from zero
r(converged)	1 if iteration algorithm converged, 0 otherwise

Macros

r(type)	test
r(method)	mcc
r(direction)	upper or lower
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results

Methods and formulas

Consider a 1:M matched case–control study, where n cases with a disease are matched to M controls without the disease on the basis of similar values of confounding variables such as age, gender, etc. Some patients in the study have prior exposure to a certain risk factor of interest. Below we provide power and sample-size formulas based on Dupont (1988) . Also see Breslow and Day (1980, sec. 5.3) for background on the analysis of 1:M matched data.

Referring to the table in the *Introduction*, let p_0 denote the probability of exposure among the control patients. Under the null hypothesis, the odds ratio θ of developing the disease in exposed and unexposed subjects is constant for equal values of the confounding variables.

The value of p_{ij} for each cell represents the joint probability of exposure status of the matched case–control pair for $i, j = 0, 1$. (With M matches, the first control is used.) Let p_0 and p_1 represent the probability of exposure of the control and case patient, respectively. Let ρ denote the correlation coefficient for exposure in matched pairs of case–control patients. Let $q_0 = 1 - p_0$ and $q_1 = 1 - p_1$, then the odds ratio is given by $\theta = p_1 q_0 / p_0 q_1$ if $\rho = 0$ and $\theta = p_{10}/p_{01}$ if $\rho \neq 0$.

The correlation coefficient can be expressed as

$$\rho = \frac{p_{11}p_{00} - p_{10}p_{01}}{\sqrt{p_1 q_1 p_0 q_0}} \tag{1}$$

The individual cell probabilities can be expressed in terms of exposure probabilities as

$$\begin{aligned} p_{11} &= p_1 p_0 + \rho\sqrt{p_1 q_1 p_0 q_0} \\ p_{10} &= p_1 q_0 - \rho\sqrt{p_1 q_1 p_0 q_0} \\ p_{01} &= q_1 p_0 - \rho\sqrt{p_1 q_1 p_0 q_0} \\ p_{00} &= q_1 q_0 + \rho\sqrt{p_1 q_1 p_0 q_0} \end{aligned}$$

Let p_{0+} and p_{0-} denote the probability that a control patient is exposed given the corresponding matched case is or is not exposed, respectively. Then

$$\begin{aligned} p_{0+} &= \frac{p_{11}}{p_1} \\ p_{0-} &= \frac{p_{01}}{q_1} \\ q_{0+} &= 1 - p_{0+} \\ q_{0-} &= 1 - p_{0-} \end{aligned}$$

The probability of observing m exposed subjects among a case matched with M controls is given by

$$t_m = p_1 \binom{M}{m-1} p_{0+}^{m-1} q_{0+}^{M-m+1} + q_1 \binom{M}{m} p_{0-}^{m} q_{0-}^{M-m}$$

for $m = 1, \ldots, M$.

Let $n_{i,j}$ denote the number of matched sets such that $n_{1,j}$ is the number of exposed cases matched with j of M exposed controls and $n_{0,j}$ is the number of nonexposed cases matched with j of M exposed controls. Then, the number of matched sets in which m subjects were exposed is

$$T_m = n_{1,m-1} + n_{0,m}$$

Define a matched set as a discordant matched set if there is a least one discordant pair between the case and the M controls. Then, the number of discordant matched sets in which the case patients were exposed is

$$y = \sum_{m=1}^{M} n_{1,m-1}$$

Denote E_θ and s_θ the conditional mean and standard deviation of y, respectively, given $T_m = E(T_m) = nt_m$, for $m = 1, \ldots, M$. Then $E_\theta = ne_\theta$ and $s_\theta = \sqrt{n\nu_\theta}$, where

$$e_\theta = \sum_{m=1}^{M} \frac{mt_m\theta}{m\theta + M - m + 1} \qquad \text{and} \qquad \nu_\theta = \sum_{m=1}^{M} \frac{mt_m\theta(M-m+1)}{(m\theta + M - m + 1)^2}$$

(Breslow and Day 1980, eq. 5.19).

Let $L_\alpha = (E_1 - E_\theta - z_{1-\alpha}s_1)/s_\theta$ and $U_\alpha = (E_1 - E_\theta + z_{1-\alpha}s_1)/s_\theta$, where $z_{1-\alpha}$ is the quantile of a standard normal distribution such that $P(Z \geq z_{1-\alpha}) = \alpha$ and $\Phi(\cdot)$ is the cumulative of a standard normal distribution.

Then, the power of the test is given by

$$1-\beta = \begin{cases} \Phi\left(L_{\alpha/2}\right) + 1 - \Phi\left(U_{\alpha/2}\right) & \text{for a two-sided test} \\ \Phi\left(L_\alpha\right) & \text{for a lower one-sided test} \\ 1 - \Phi\left(U_\alpha\right) & \text{for an upper one-sided test} \end{cases} \tag{2}$$

The sample size for a one-sided test can be obtained from the inverse of the power computation, $n = (z_{1-\beta}\sqrt{\nu_\theta} + z_{1-\alpha}\sqrt{\nu_1})^2/(e_1 - e_\theta)^2$. The sample size for a two-sided test and the minimum detectable odds ratio θ must be computed iteratively from (2). The starting value for the sample-size computation is the sample-size estimate for the one-sided test. For the minimum detectable odds ratio, the starting value is obtained from a bisection algorithm.

Let F_M denote the ratio of the sample sizes for a study with a 1:M matched design relative to a study with a 1:1 matched design. If n_1 is the sample size required for a study with 1 control matched to 1 case and n_M is the sample size required for a study with M controls matched to 1 case, then $F_M = n_M/n_1$.

References

Breslow, N. E., and N. E. Day. 1980. *The Analysis of Case–Control Studies*. Vol. 1 of *Statistical Methods in Cancer Research*. Lyon: IARC.

Dupont, W. D. 1988. Power calculations for matched case–control studies. *Biometrics* 44: 1157–1168. https://doi.org/10.2307/2531743.

Hosmer, D. W., Jr., S. A. Lemeshow, and R. X. Sturdivant. 2013. *Applied Logistic Regression*. 3rd ed. Hoboken, NJ: Wiley.

Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power pairedproportions** — Power analysis for a two-sample paired-proportions test

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[R] **clogit** — Conditional (fixed-effects) logistic regression

[R] **Epitab** — Tables for epidemiologists

[R] **symmetry** — Symmetry and marginal homogeneity tests

power trend — Power analysis for the Cochran–Armitage trend test

Description

`power trend` computes sample size or power for the Cochran–Armitage trend test, a test for a linear trend in a probability of response in $J \times 2$ tables. It can accommodate unbalanced designs and unequally spaced exposure levels (doses). With equally spaced exposure levels, a continuity correction is available.

Quick start

Sample size for a test with alternative probabilities of 0.2, 0.3, and 0.4 using default power of 0.8 and significance level $\alpha = 0.05$

```
power trend .2 .3 .4
```

Same as above, but for power of 0.9

```
power trend .2 .3 .4, power(.9)
```

Same as above, but for power of 0.7, 0.75, 0.8, 0.85, and 0.9

```
power trend .2 .3 .4, power(.7(.05).9)
```

Sample size for a one-sided test

```
power trend .2 .3 .4, onesided
```

Same as above, and apply continuity correction

```
power trend .2 .3 .4, onesided continuity
```

Power for a total sample size of 240 subjects

```
power trend .2 .3 .4, n(240)
```

Same as above, specified as 3 groups of 80 subjects each

```
power trend .2 .3 .4, npergroup(80)
```

Power for 100 subjects in group 1, 80 in group 2, and 60 in group 3

```
power trend .2 .3 .4, n1(100) n2(80) n3(60)
```

Graph of power against group sample size for group sizes of 70, 80, 90, and 100

```
power trend .2 .3 .4, npergroup(70(10)100) graph
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
power trend probspec [ , power(numlist) options ]
```

Compute power

```
power trend probspec , n(numlist) [ options ]
```

where *probspec* is either a matrix *matname* containing group probabilities or a list of individual group probabilities:

$p_1\ p_2\ [\ p_3\ \cdots\ p_J\ ]$

p_j, where $j = 1, 2, \ldots, J$, is the alternative group probability of observing a success for subjects with the jth level of exposure. Each p_j may be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

matname is the name of a Stata matrix with J columns containing values of alternative group probabilities. Multiple rows are allowed, in which case each row corresponds to a different set of J group probabilities or, equivalently, column j corresponds to *numlist* for the jth group probabilities.

Alternative probabilities should be strictly monotonic: all increasing or all decreasing.

options	Description
Main	
* `alpha(`*numlist*`)`	significance level; default is `alpha(0.05)`
* `power(`*numlist*`)`	power; default is `power(0.8)`
* `beta(`*numlist*`)`	probability of type II error; default is `beta(0.2)`
* `n(`*numlist*`)`	total sample size; required to compute power
`nfractional`	allow fractional sample sizes
* `npergroup(`*numlist*`)`	number of subjects per group; implies balanced design
* `n#(`*numlist*`)`	number of subjects in group #
`grweights(`*wgtspec*`)`	group weights; default is one for each group, meaning equal group sizes
`exposure(`*exposspec*`)`	strictly increasing exposure levels; default is equally spaced ordinal values
`continuity`	apply the continuity correction; default is no continuity correction
`onesided`	one-sided test; default is two sided
`parallel`	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[`no`]`table`[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
`saving(`*filename* [`, replace`]`)`	save the table data to *filename*; use `replace` to overwrite existing *filename*
Graph	
`graph`[(*graphopts*)]	graph results; see [PSS-2] **power, graph**
Iteration	
`init(`#`)`	initial value for sample size for a two-sided test; default is to use a sample-size estimate for a one-sided test
`iterate(`#`)`	maximum number of iterations; default is `iterate(500)`
`tolerance(`#`)`	parameter tolerance; default is `tolerance(1e-12)`
`ftolerance(`#`)`	function tolerance; default is `ftolerance(1e-12)`
[`no`]`log`	suppress or display iteration log
[`no`]`dots`	suppress or display iterations as dots
`notitle`	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the `parallel` option.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`notitle` does not appear in the dialog box.

wgtspec	Description
$\#_1\ \#_2\ \ldots\ \#_J$	J group weights. Weights must be positive and must be integers unless option `nfractional` is specified. Multiple values for each group weight $\#_j$ can be specified as a *numlist* enclosed in parentheses.
matname	matrix with J columns containing J group weights. Multiple rows are allowed, in which case each row corresponds to a different set of J weights or, equivalently, column j corresponds to a *numlist* for the jth weight.

exposspec	Description
$\#_1\ \#_2\ \ldots\ \#_J$	J exposure levels. By default, equally spaced exposure levels of 1, 2, …, J are used. Multiple values for each exposure level $\#_j$ can be specified as a *numlist* enclosed in parentheses.
matname	matrix with J columns containing J exposure levels. Multiple rows are allowed; in which case each row corresponds to a different set of J exposure levels or, equivalently, column j corresponds to a *numlist* for the jth exposure level.

where *tablespec* is

column[:*label*] [*column*[:*label*] [. . .]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`N`	total number of subjects	N
`N_per_group`	number of subjects per group	N/N_g
`N_avg`	average number of subjects per group	N_{avg}
`N#`	number of subjects in group #	$N_{\#}$
`N_g`	number of groups	N_g
`p#`	probability of outcome for group #	$p_{\#}$
`x#`	exposure level #	$x_{\#}$
`grwgt#`	group weight #	$w_{\#}$
`_all`	display all supported columns	

Column `beta` is shown in the default table in place of column `power` if option `beta()` is specified.

Column `N_per_group` is shown in the default table only for balanced designs.

Columns `N_avg` and `N#` are shown in the default table only for unbalanced designs.

Columns `x#` are shown only when exposure levels are specified using the `exposure()` option.

Options

Main

`alpha()`, `power()`, `beta()`, `n()`, `nfractional`; see [PSS-2] **power**.

`npergroup(`*numlist*`)` specifies the group size. Only positive integers are allowed. This option implies a balanced design. `npergroup()` cannot be specified with `n()`, `n#()`, or `grweights()`.

`n#(`*numlist*`)` specifies the number of subjects in the #th group to be used for power determination. Only positive integers are allowed. All group sizes must be specified. `n#()` cannot be specified with `n()`, `npergroup()`, or `grweights()`.

`grweights(`*wgtspec*`)` specifies J group weights for an unbalanced design. The weights may be specified either as a list of values or as a matrix, and multiple sets of weights are allowed; see *wgtspec* for details. The weights must be positive and must also be integers unless the `nfractional` option is specified. `grweights()` cannot be specified with `npergroup()` or `n#()`.

`exposure(`*exposspec*`)` specifies the J strictly increasing exposure levels. The default is to use equally spaced values of $1, 2, \ldots, J$.

`continuity` requests that the continuity correction be applied. This option can be specified only for equally spaced exposure levels. By default, no continuity correction is applied.

`onesided`, `parallel`; see [PSS-2] **power**.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies the initial value of the sample size for the sample-size computation for a two-sided test. The default initial value is the sample size for the corresponding one-sided test.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following option is available with `power trend` but is not shown in the dialog box:

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `power trend` command and the methodology for power and sample-size analysis for Cochran–Armitage test for a linear trend in probability of response in $J \times 2$ tables. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis and [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Introduction

Studies that examine the relationship between an exposure and a binary outcome have many biomedical and social science applications. When exposure can be treated as ordinal levels, researchers are often interested in whether there is a trend, or dose–response relationship, in exposure and a binary outcome. The data are typically summarized in an ordered $J \times 2$ table,

	Binary response	
Exposure level	Success	Failure
x_1	m_1	$n_1 - m_1$
x_2	m_2	$n_2 - m_2$
$\vdots$	$\vdots$	$\vdots$
x_J	m_J	$n_J - m_J$

where x_j is an ordinal level (table score) or rank score associated with the exposure (dose) received by group j such that $x_{j-1} < x_j$, m_j is the number of successes in group j and n_j is the number of subjects in group j for each $j = 1, 2, \ldots, J$. For equally spaced exposure levels, the levels are often assigned ordinal numbers; $x_j = j,\ j = 1, 2, \ldots, J$.

A "success" simply means observing an event of interest. A dermatologist might wish to identify dosage of a topical antibiotic (ordinal exposure) necessary to cure a skin infection (binary outcome). A oncologist might conduct a case–control study to see if the number of first- and second-degree relatives with a BRCA1 gene mutation is associated with the occurrence of breast cancer. An education researcher might want to know if a higher number of unexcused absences from school is associated with failing a school grade.

This entry describes power and sample-size analysis for inference using hypothesis testing about the presence of a linear trend in probability of response in $J \times 2$ tables. The Cochran–Armitage trend test (Cochran 1954 and Armitage 1955) is commonly used to test for trend in $J \times 2$ tables. It is based on the linear logit model,

$$\text{logit}(p_j) = a + bx_j$$

where p_j is the hypothesized probability of a success in group j, and a and b are unknown coefficients. These group probabilities can be estimated from our contingency table as $\widehat{p}_j = m_j/n_j$.

The null hypothesis of interest is $H_0\colon p_1 = p_2 = \cdots = p_J$ against the one-sided increasing-trend alternative $H_a\colon p_1 < p_2 < \cdots < p_J$, the one-sided decreasing-trend alternative $H_a\colon p_1 > p_2 > \cdots > p_J$, or the two-sided alternative $H_a\colon p_1 < p_2 < \cdots < p_J$ or $p_1 > p_2 > \cdots > p_J$. For the linear logit model, these hypothesis are equivalent to the null $H_0\colon b = 0$ against the two-sided alternative $H_a\colon b \neq 0$. If we believe that the probability of a success increases with exposure, the alternative hypothesis is upper one-sided, $H_a\colon b > 0$. If we believe that the probability of a success decreases with exposure, the alternative hypothesis is lower one-sided, $H_a\colon b < 0$. The test statistic for testing $H_0\colon b = 0$ is asymptotically normal under the null hypothesis.

Power and sample-size computations are based on the asymptotic distribution of the test statistic.

Using power trend

`power trend` computes sample size or power for a Cochran–Armitage trend test in $J \times 2$ tables. All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option. You can specify the `onesided` option to request a one-sided test.

To compute the total and individual group sample sizes, you must specify the alternative probabilities of a success for J levels of exposure and, optionally, the power of the test in the `power()` option. The default power is set to 0.8.

To compute power, you must specify the total sample size in the `n()` option and the alternative probabilities.

There are multiple ways to specify the alternative probabilities; see *Alternative ways of specifying probabilities*.

By default, all computations assume a balanced- or equal-allocation design. You can use the `grweights()` option to specify an unbalanced design for power or sample-size computations. For power computations, you can specify individual group sizes in options `n1()`, `n2()`, ..., `nJ()` instead of a combination of `n()` and `grweights()` to accommodate an unbalanced design. For a balanced design, you can also specify the `npergroup()` option to specify a group size instead of a total sample size in `n()`.

Computations also assume that exposure levels are equally spaced and no continuity correction is applied. When the exposure levels are equally spaced, you can use option `continuity` to request that the continuity correction be applied. You may specify specific exposure levels in the `exposure()` option. There are multiple ways of specifying the levels; any method described in *Alternative ways of specifying probabilities* can also be applied to the specification of exposure levels.

Sample-size determination for a two-sided test requires iteration. The default initial values are sample-size estimates for the corresponding one-sided test. You can use the `init()` option to specify your own value. See [PSS-2] **power** for a description of other options that control the iteration process.

Alternative ways of specifying probabilities

There are multiple ways in which you can supply the group probabilities of success to `power trend`.

You may specify each p_j following the command line as

`power trend` $p_1\ p_2 \ldots p_J$ [, ...]

At least two probabilities must be specified.

When you have many groups, you may find it more convenient to first define a Stata matrix as a row or column vector and use it with `power trend`. The dimension of the matrix must be at least 2. For example,

`matrix define` *probmat* = (p_1, p_2,..., p_J)

`power trend` *probmat* [, ...]

In some cases, you may wish to examine multiple alternative probabilities for one or more groups. To do this, you can specify multiple values or a *numlist* for each of the group probabilities in parentheses.

`power trend` ($p_{1,1}\ p_{1,2} \ldots p_{1,K_1}$) ($p_{2,1}\ p_{2,2} \ldots p_{2,K_2}$) ... [, ...]

Each of the *numlist*s may contain different numbers of values, $K_1 \neq K_2 \neq \cdots \neq K_J$. `power trend` will produce results for all possible combinations of values across *numlist*s. Results are presented in a table. If instead you would like to treat each specification as a separate scenario, you may specify the `parallel` option.

You can accommodate multiple sets of group probabilities in a matrix form by adding a row for each specification. The columns of a matrix with multiple rows correspond to J group probabilities, and values within each column j correspond to multiple specifications of the jth group probability of a success or a *numlist* for the jth group probability.

For example, the following two specifications for three groups defined by their exposure levels with two scenarios each are the same:

`power trend` (*p*$_{1,1}$ *p*$_{1,2}$) (*p*$_{2,1}$ *p*$_{2,2}$) (*p*$_{3,1}$ *p*$_{3,2}$) [, ...]

and

`matrix define` *probmat* = (*p*$_{1,1}$, *p*$_{2,1}$, *p*$_{3,1}$ \ *p*$_{1,2}$, *p*$_{2,2}$, *p*$_{3,2}$)

`power trend` *probmat* [, ...]

In the above specification, if you wish to specify a *numlist* only for the first group, you may define your matrix as

`matrix define` *probmat* = (*p*$_{1,1}$, *p*$_{2,1}$, *p*$_{3,1}$ \ *p*$_{1,2}$, . , .)

and the results of

`power trend` *probmat* [, ...]

will be the same as the results of

`power trend` (*p*$_{1,1}$ *p*$_{1,2}$) *p*$_{2,1}$ *p*$_{3,1}$ [, ...]

In the following sections, we describe the use of `power trend` accompanied by examples for computing sample size and power.

Computing sample size

To compute sample size, you must specify the alternative probabilities of a success for each of the J exposure levels and, optionally, the power of the test in the `power()` option. A default power of 0.8 is assumed if `power()` is not specified.

Example 1: Sample size for a two-sided trend test

Consider a study investigating the effectiveness of a new topical antibiotic for the treatment of skin infections.

Suppose that in previous studies of the treatment, we observed the following proportions of successfully treated cases at different doses. We may hypothesize that these represent the probability of a successful treatment for each dose.

Doses/day	Proportion Successes
1	0.80
2	0.85
3	0.90

We wish to determine the minimum sample size required for a clinical trial designed to detect a dose–response trend with 80% power using a two-sided 5%-level test.

To compute the required sample size, we specify the values 0.80, 0.85, and 0.90 as the alternative probabilities after the command name. We omit options `alpha(0.05)` and `power(0.8)` because the specified values are their defaults.

```
. power trend .80 .85 .90
note: exposure levels are assumed to be equally spaced.

Performing iteration ...

Estimated sample size for a trend test
Cochran-Armitage trend test
H0: b = 0  versus  Ha: b != 0;  logit(p) = a + b*x

Study parameters:

        alpha =    0.0500
        power =    0.8000
          N_g =         3
           p1 =    0.8000
           p2 =    0.8500
           p3 =    0.9000

Estimated sample sizes:

            N =       597
  N per group =       199
```

A total sample of 597 individuals, 199 individuals per group, must be obtained to detect a linear trend in probability of a successful treatment with 80% power using a two-sided 5%-level Cochran–Armitage test.

◁

Example 2: Sample size for a one-sided test

Continuing with example 1, suppose that the relevant research question is whether the probability of a successful treatment increases with the number of doses. In this case, we would choose to use a one-sided test because we are only interested in an increasing trend.

By specifying the `onesided` option, we obtain the sample size needed to detect a positive trend with 80% power using a one-sided 5%-level Cochran–Armitage test.

```
. power trend 0.80 0.85 0.90, onesided
note: exposure levels are assumed to be equally spaced.

Estimated sample size for a trend test
Cochran-Armitage trend test
H0: b = 0  versus  Ha: b > 0;  logit(p) = a + b*x

Study parameters:

        alpha =    0.0500
        power =    0.8000
          N_g =         3
           p1 =    0.8000
           p2 =    0.8500
           p3 =    0.9000

Estimated sample sizes:

            N =       471
  N per group =       157
```

Switching to a one-sided hypothesis decreased our total sample-size requirement to 471 individuals. Because the doses are equally spaced, it is possible to also perform a continuity correction. If we wanted this correction, we would have also specified the `continuity` option.

◁

▷ Example 3: Unbalanced design

Continuing with example 1, we have assumed that the participants will be equally divided among the treatment groups. Thus, we would randomize 199 participants to each treatment level. Suppose that we instead plan to have twice as many subjects at the lowest treatment level. We can accommodate this unbalanced design by specifying the corresponding group weights in the `grweights()` option.

```
. power trend 0.80 0.85 0.90, grweights(2 1 1)
note: exposure levels are assumed to be equally spaced.

Performing iteration ...

Estimated sample size for a trend test
Cochran-Armitage trend test
H0: b = 0  versus  Ha: b != 0;  logit(p) = a + b*x

Study parameters:

        alpha =    0.0500
        power =    0.8000
          N_g =         3
           p1 =    0.8000
           p2 =    0.8500
           p3 =    0.9000

Estimated sample sizes:

            N =       600
    Average N =  200.0000
           N1 =       300
           N2 =       150
           N3 =       150
```

The required total sample size for this unbalanced design is 600 with 300 subjects in the group receiving a single dose per day and 150 in the groups receiving two and three doses per day. The average number of subjects per group is 200.

◁

▷ Example 4: Sample size for unequally spaced exposures

It is possible that the groups do not represent equally spaced exposures. Consider an example from Agresti (2013, 89) on the association between maternal drinking and congenital malformations, which shows data originally from Graubard and Korn (1987). The data are reported in ranges rather than actual values.

```
. use https://www.stata-press.com/data/r19/infants
(Congenital malformation data)
. list, noobs abbreviate(12)
```

consump	cscore	infants	cases	prmalform
0	0.0	17,114	48	0.0028
< 1	0.5	14,502	38	0.0026
1 to 2	1.5	793	5	0.0063
3 to 5	4.0	127	1	0.0079
>= 6	7.0	38	1	0.0263

Suppose we wish to use these data to design a new study to test whether there is an increasing trend in the probability of congenital malformation as the average number of alcoholic beverages consumed each week by the mother increases. We will use the `onesided` option as we did in example 2 to obtain sample size with 80% power using a one-sided 5%-level Cochran–Armitage test.

To accurately compute the sample size that we will need, we should use the score associated with the reported range. Agresti (2013) recommends using the midpoint of the range for a Cochran–Armitage trend test and adopts an arbitrary value of 7 for the last interval in this case.

We can relax the assumption that the exposure levels for alcohol consumption are evenly spaced by adding the `exposure()` option. Because we have 5 exposure levels, we use a matrix specification of alternative probabilities and exposure levels. We can use the `mkmat` command to create the matrices rather than retyping the values that we already have in our dataset; see [P] **matrix mkmat**.

```
. mkmat prmalform, matrix(p)
. matrix list p
p[5,1]
    prmalform
r1  .00280472
r2  .00262033
r3  .00630517
r4  .00787402
r5  .02631579
. mkmat cscore, matrix(exposed)
. matrix list exposed
exposed[5,1]
    cscore
r1       0
r2      .5
r3     1.5
r4       4
r5       7
```

Now, we specify the matrix of probabilities `p` after the command `power trend` and the matrix of exposure levels `exposed` in option `exposure()`.

```
. power trend p, onesided exposure(exposed)
Estimated sample size for a trend test
Cochran-Armitage trend test
H0: b = 0  versus  Ha: b > 0;  logit(p) = a + b*x
Study parameters:
        alpha =    0.0500
        power =    0.8000
          N_g =         5
           p1 =    0.0028          x1 =    0.0000
           p2 =    0.0026          x2 =    0.5000
           p3 =    0.0063          x3 =    1.5000
           p4 =    0.0079          x4 =    4.0000
           p5 =    0.0263          x5 =    7.0000
Estimated sample sizes:
            N =     1,030
  N per group =       206
Warning: Alternative probabilities are not monotonic.
```

To conduct this study, we would need to recruit a total sample of 1,030 mothers with 206 mothers at each level of consumption.

For this example, `power trend` reported a warning message that the specified alternative probabilities are not monotonic. The power for the test depends on the specific alternative, which is $p_1 < p_2 < \cdots < p_J$ in this example. Gross departures from the monotonicity assumption may lead to invalid results. In our example, the violation of this assumption is very mild—the offending probabilities are $p_1 = 0.0028$ and $p_2 = 0.0026$.

◁

Computing power

To compute power, you must specify the alternative probabilities after the command name and the total sample size in the `n()` option.

▷ Example 5: Power of a two-sided Cochran–Armitage trend test

Returning to example 1, suppose that we anticipate obtaining a sample size of only 540 participants. To compute the corresponding power, we specify the sample size of 540 in `n()`:

```
. power trend 0.80 0.85 0.90, n(540)
note: exposure levels are assumed to be equally spaced.
Estimated power for a trend test
Cochran-Armitage trend test
H0: b = 0  versus  Ha: b != 0;  logit(p) = a + b*x
Study parameters:
        alpha =    0.0500
            N =       540
  N per group =       180
          N_g =         3
           p1 =    0.8000
           p2 =    0.8500
           p3 =    0.9000
Estimated power:
        power =    0.7592
```

Power decreases to 75.9% with the smaller sample of 540 subjects.

◁

▷ Example 6: Multiple values of study parameters

We may want to check powers for several sample sizes. Continuing with example 5, we simply list sample-size values in the option `n(`*numlist*`)`; see **[U] 11.1.8 numlist**.

```
. power trend 0.80 0.85 0.90, n(540 570 600 630 660)
> table(, labels(N_per_group "N/N_g") formats("%6.2g"))
note: exposure levels are assumed to be equally spaced.

Estimated power for a trend test
Cochran-Armitage trend test
H0: b = 0  versus  Ha: b != 0;  logit(p) = a + b*x
```

alpha	power	N	N/N_g	N_g	p1	p2	p3
.05	.76	540	180	3	.8	.85	.9
.05	.78	570	190	3	.8	.85	.9
.05	.8	600	200	3	.8	.85	.9
.05	.82	630	210	3	.8	.85	.9
.05	.84	660	220	3	.8	.85	.9

To shorten our default table, we specified a shorter label for the `N_per_group` column and reduced the default display format for all table columns by specifying the corresponding options within the `table()` option.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

Testing hypotheses about a trend in J × 2 tables

There are several ways to conduct a trend test in Stata. We demonstrate one method here. For more examples showing the Cochran–Armitage trend test in Stata, see Sribney (1996).

▷ Example 7: Testing hypotheses about trends

Returning to example 4, let's test whether the proportion of infants with congenital malformations increases as maternal alcohol consumption per week increases using Agresti's data. (We converted these data from wide to long as needed by further analysis.)

```
. use https://www.stata-press.com/data/r19/infants2
(Congenital malformation data, long form)
. list, noobs sepby(consump)
```

consump	cscore	malform	n
0	0.0	Has malformation	17066
0	0.0	No malformation	48
< 1	0.5	Has malformation	14464
< 1	0.5	No malformation	38
1 to 2	1.5	Has malformation	788
1 to 2	1.5	No malformation	5
3 to 5	4.0	Has malformation	126
3 to 5	4.0	No malformation	1
>= 6	7.0	Has malformation	37
>= 6	7.0	No malformation	1

In Stata, we can conduct a trend test by using the `tabodds` command; see [R] **Epitab**.

```
. tabodds malform cscore [fweight=n]
```

cscore	Cases	Controls	Odds	[95% conf.	interval]
0	48	17066	0.00281	0.00212	0.00373
.5	38	14464	0.00263	0.00191	0.00361
1.5	5	788	0.00635	0.00263	0.01529
4	1	126	0.00794	0.00111	0.05678
7	1	37	0.02703	0.00371	0.19698

```
Test of homogeneity (equal odds): chi2(4)  =    12.08
                                  Pr>chi2  =   0.0168

Score test for trend of odds:     chi2(1)  =     6.57
                                  Pr>chi2  =   0.0104
```

`tabodds` reports a χ^2 value of 6.57 for the trend test. We reject the null hypothesis of no trend at the 5% significance level; the p-value is 0.0104.

Note that the χ^2 statistic reported by `tabodds` is for a score test for trend of odds. This test is asymptotically equivalent to the Cochran–Armitage trend test. See Lachin (2011) and Agresti (2013) for details.

◁

Stored results

power trend stores the following in r():

Scalars

r(alpha)	significance level
r(power)	power
r(beta)	probability of a type II error
r(delta)	effect size
r(N)	total sample size
r(N_a)	actual sample size
r(N_avg)	average sample size
r(N#)	number of subjects in group #
r(N_per_group)	number of subjects per group
r(N_g)	number of groups
r(nfractional)	1 if nfractional is specified, 0 otherwise
r(balanced)	1 for a balanced design, 0 otherwise
r(grwgt#)	group weight #
r(p#)	probability of a success in group #
r(x#)	exposure level for group #
r(continuity)	1 if continuity correction is used, 0 otherwise
r(c)	continuity-correction value
r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise
r(init)	initial value for sample size for a two-sided test
r(maxiter)	maximum number of iterations
r(iter)	number of iterations performed
r(tolerance)	requested parameter tolerance
r(deltax)	final parameter tolerance achieved
r(ftolerance)	requested distance of the objective function from zero
r(function)	final distance of the objective function from zero
r(converged)	1 if iteration algorithm converged, 0 otherwise

Macros

r(type)	test
r(method)	trend
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results

Methods and formulas

Assume that the probability of a success or a positive response, p_j, follows a linear trend on the logistic scale

$$p_j = \frac{e^{a+bx_j}}{1+e^{a+bx_j}}$$

such that $\text{logit}(p_j) = a + bx_j$, where x_j denotes the exposure level (dose) for each of the J groups and $x_{j-1} < x_j$ for $j = 1, 2, \ldots, J$. b is the trend parameter about which we form our hypotheses. Under the null hypothesis, H_0: $b = 0$.

Power and sample-size computations for this test are based on Nam (1987).

Methods and formulas are presented under the following headings:

Computing power

Let n_j be the sample size at each exposure level for $j = 1, 2, \ldots, J$ and $n = \sum_{j=1}^{J} n_j$ denote the total sample size. The observed number of successes m_j at each of the J exposure levels follows a binomial distribution; the m_js are assumed to be independent. Let $M = \sum_{j=1}^{J} m_j$ denote the total response. Then, the average response rate is given by $\overline{p} = M/n$ and the average nonresponse rate by $\overline{q} = 1 - \overline{p}$.

Let $U = \sum_{j=1}^{J} m_j x_j$ denote the total exposure-weighted response and $\overline{x} = \sum_{j=1}^{J} n_j x_j / n$ be the average exposure level in the sample. Denote the conditional mean of U given M as $E_0(U|M)$, where $E_0(U|M) = \overline{p}(\sum_{j=1}^{J} n_j x_j)$. Let $U' = U - E_0(U|M) = \sum_{j=1}^{J} m_j(x_j - \overline{x})$.

Given the significance level α and the probability of a type II error β, the power $\pi = 1-\beta$ is computed as

$$\pi = \begin{cases} \Phi(u_l) & \text{for a lower one-sided test} \\ 1 - \Phi(u_u) & \text{for an upper one-sided test} \\ 1 - \Phi(u_u) + \Phi(u_l) & \text{for a two-sided test} \end{cases} \tag{1}$$

where $\Phi(\cdot)$ is the cumulative distribution function of the standard normal distribution. For a one-sided test,

$$u_l = \frac{-E(U' + c) + z_\alpha \sqrt{\text{Var}_0(U')}}{\sqrt{\text{Var}(U')}} \tag{2a}$$

and

$$u_u = \frac{-E(U' - c) - z_\alpha \sqrt{\text{Var}_0(U')}}{\sqrt{\text{Var}(U')}} \tag{2b}$$

In (2a) and (2b), z_α is the (α)th quantile of the standard normal distribution and $c = (x_{j+1} - x_j)/2$ in the presence of continuity correction or $c = 0$ in the absence of continuity correction. The correction is not available with unequally spaced exposure levels. The power of the two-sided test uses $\alpha/2$ to determine the value of z when computing u_l in (2a) and u_u in (2b).

Under the null hypothesis, $p_1 = p_2 = \cdots = p_J$. Therefore, we can define a common $p = \sum_{j=1}^{J} n_j p_j / N$ and $q = 1 - p$. The variance of U' under the null is

$$\text{Var}_0(U') = \text{Var}(U'|H_0) = pq \sum_{j=1}^{J} n_j (x_j - \overline{x})^2$$

and the variance of U' given $p_1, p_2, \ldots, p_J$ is

$$\text{Var}(U') = \sum_{j=1}^{J} n_j p_j q_j (x_j - \overline{x})^2$$

Computing sample size

Let n_1 denote the sample size of the reference group, the group with the lowest exposure level, and $r_j = n_j/n_1$ be the known ratio of sample size of the jth group to that of the reference group. Define $A = \sum_{j=1}^{J} r_j p_j (x_j - \overline{x})$, and then sample size for a one-sided test without continuity correction is given by

$$n_1 = \frac{1}{A^2}\left[-z_\alpha\sqrt{pq\left\{\sum_{j=1}^{J} r_j (x_j-\overline{x})^2\right\}} + z_{1-\beta}\sqrt{\sum_{j=1}^{J} r_j p_j q_j (x_j-\overline{x})^2}\right]^2 \tag{3}$$

where z_α is the (α)th quantile of the standard normal distribution.

The total sample size is computed as $n = \sum_{j=1}^{J} n_j = n_1 \sum_{j=1}^{J} r_j$.

The sample-size estimate n_{1c} for the continuity-corrected statistic is found according to Nam (1987), as follows:

$$n_{1c} = \begin{cases} \frac{n_1}{4}\left(1+\sqrt{1-\frac{4c}{An_1}}\right)^2 & \text{for a lower one-sided test} \\ \frac{n_1}{4}\left(1+\sqrt{1+\frac{4c}{An_1}}\right)^2 & \text{for an upper one-sided test} \end{cases} \tag{4}$$

For a two-sided hypothesis, n is computed by iteratively solving the two-sided power equation given in (1) using the one-sided estimates as starting values. Without continuity correction, (3) is used to obtain the one-sided sample estimates for the starting values; with a continuity correction, (4) is used.

References

Agresti, A. 2013. *Categorical Data Analysis*. 3rd ed. Hoboken, NJ: Wiley.

Armitage, P. 1955. Tests for linear trends in proportions and frequencies. *Biometrics* 11: 375–386. https://doi.org/10.2307/3001775.

Cochran, W. G. 1954. Some methods for strengthening the common chi-squared tests. *Biometrics* 10: 417–451. https://doi.org/10.2307/3001616.

Graubard, B. I., and E. L. Korn. 1987. Choice of column scores for testing independence in ordered $2 \times K$ contingency tables. *Biometrics* 43: 471–476. https://doi.org/10.2307/2531828.

Lachin, J. M. 2011. *Biostatistical Methods: The Assessment of Relative Risks*. 2nd ed. Hoboken, NJ: Wiley.

Nam, J. 1987. A simple approximation for calculating sample sizes for detecting linear trend in proportions. *Biometrics* 43: 701–705. https://doi.org/10.2307/2532006.

Sribney, W. M. 1996. FAQ: Does Stata provide a test for trend? https://www.stata.com/support/faqs/statistics/test-for-trend.

Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[R] **Epitab** — Tables for epidemiologists

[R] **logit** — Logistic regression, reporting coefficients

power cox — Power analysis for the Cox proportional hazards model

Description

`power cox` computes sample size, power, or effect size for survival analyses that use Cox proportional hazards (PH) models. The results are obtained for the test of the effect of one covariate (binary or continuous) on time to failure adjusted for other predictors in a PH model. Effect size can be expressed as a regression coefficient (or log hazard-ratio) or as a hazard ratio. The command can account for the dependence between the covariate of interest and other model covariates, and it can adjust computations for censoring and for withdrawal of subjects for the study.

Quick start

Sample size for a test of $H_0\colon \beta_1 = 0$ versus $H_a\colon \beta_1 \neq 0$ of a binary covariate in a Cox PH model given alternative coefficient *b1* of 0.4, no censoring, and default power of 0.8 and significance level $\alpha = 0.05$

```
power cox .4
```

Same as above, specified using hazard ratio of 1.492

```
power cox, hratio(1.492)
```

Sample size for a test of a continuous independent variable with *b1* of 1.2 and standard deviation of 0.47

```
power cox 1.2, sd(.47)
```

Same as above, but for a model with multiple covariates and a squared multiple-correlation coefficient $R^2 = 0.2$ when the covariate of interest is regressed on all other covariates

```
power cox 1.2, sd(.47) r2(.2)
```

Sample size and estimated number of events for a study with censoring, assuming an 80% event rate

```
power cox .4, eventprob(.8)
```

Plot required sample size versus coefficient values of 0.2, 0.3, 0.4, 0.5, and 0.6

```
power cox (.2(.1).6), graph
```

Sample size assuming 12% dropout of subjects in the study

```
power cox .4, wdprob(.12)
```

Power of a test of a binary independent variable given sample size of 200

```
power cox .4, n(200)
```

Power of a one-sided test of a continuous independent variable with *b1* of 1.2 and standard deviation of 0.47

```
power cox 1.2, sd(.47) n(200) onesided
```

Effect size given power of 0.8 and sample sizes of 120, 130, 140, 150, and 160

```
power cox, power(.8) n(120(10)160)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
power cox [ b1 ] [ , power(numlist) options ]
```

Compute power

```
power cox [ b1 ], n(numlist) [ options ]
```

Compute effect size (target regression coefficient)

```
power cox, n(numlist) power(numlist) [ options ]
```

where *b1* is the hypothesized regression coefficient (effect size) of a covariate of interest in a Cox PH model desired to be detected by a test with a prespecified power. *b1* may be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

options	Description
Main	
* alpha(*numlist*)	significance level; default is alpha(0.05)
* power(*numlist*)	power; default is power(0.8)
* beta(*numlist*)	probability of type II error; default is beta(0.2)
* n(*numlist*)	sample size; required to compute power or effect size
nfractional	allow fractional sample size
* hratio(*numlist*)	hazard ratio (exponentiated *b1*) associated with a one-unit increase in covariate of interest; specify instead of the regression coefficient *b1*; default is hratio(0.5)
* sd(*numlist*)	standard deviation of covariate of interest; default is sd(0.5)
* r2(*numlist*)	squared coefficient of multiple correlation with other covariates; default is r2(0)
* eventprob(*numlist*)	overall probability of an event (failure) of interest; default is eventprob(1), meaning no censoring
* failprob(*numlist*)	synonym for eventprob()
* wdprob(*numlist*)	proportion of subjects anticipated to withdraw from the study; default is wdprob(0)
effect(*effect*)	specify the type of effect to display; default is effect(coefficient)
direction(lower|upper)	direction of the effect for effect-size determination; default is direction(lower), which means that the postulated value of the parameter is smaller than the hypothesized value
onesided	one-sided test; default is two sided
parallel	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[no]table[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
saving(*filename* [, replace])	save the table data to *filename*; use replace to overwrite existing *filename*
Graph	
graph[(*graphopts*)]	graph results; see [PSS-2] **power, graph**
notitle	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the parallel option.

collect is allowed; see **[U] 11.1.10 Prefix commands**.

notitle does not appear in the dialog box.

effect	Description
coefficient	regression coefficient, *b1*; the default
hratio	hazard ratio, exp(*b1*)
lnhratio	log hazard-ratio; synonym for coefficient

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`N`	number of subjects	N
`delta`	effect size	δ
`E`	total number of events (failures)	E
`b1`	regression coefficient	β_{1a}
`hratio`	hazard ratio	Δ
`lnhratio`	log hazard-ratio	$\ln(\Delta)$
`sd`	standard deviation	σ
`R2`	squared multiple-correlation coefficient	R^2
`Pr_E`	overall probability of an event (failure)	p_E
`Pr_w`	probability of withdrawals	p_w
`target`	target parameter; synonym for `b1`	
`_all`	display all supported columns	

Column `beta` is shown in the default table in place of column `power` if option `beta()` is specified.

Column `b1` is shown in the default table in place of column `hratio` when a regression coefficient is specified.

Columns `R2` and `Pr_w` are shown in the default table only if specified.

Options

Main

`alpha()`, `power()`, `beta()`, `n()`, `nfractional`; see [PSS-2] **power**. The `nfractional` option is allowed only for sample-size determination.

`hratio(`*numlist*`)` specifies the hazard ratio (or exponentiated regression coefficient) associated with a one-unit increase in the covariate of interest when other covariates are held constant. This value defines an effect size or the minimal clinically significant effect of a covariate on the response to be detected by a test with a certain power in a Cox PH model.

You can specify an effect size either as the regression coefficient *b1*, which is the command argument, or as the hazard ratio in `hratio()`. The default is `hratio(0.5)`. If you specify `hratio(#)`, the regression coefficient is computed as $b1 = \ln(\#)$. If you specify a regression coefficient *b1*, the hazard ratio is computed as exp(*b1*).

This option is not allowed with the effect-size determination.

`sd(`*numlist*`)` specifies the standard deviation of the covariate of interest. The default is `sd(0.5)`.

`r2(`*numlist*`)` specifies the squared multiple-correlation coefficient between the covariate of interest and other predictors in a Cox PH model. The default is `r2(0)`, meaning that the covariate of interest is independent of other covariates. This option defines the proportion of variance explained by the regression of the covariate of interest on other covariates used in the Cox model (see [R] **regress**).

`eventprob(`*numlist*`)` specifies the overall probability of a subject experiencing an event of interest (or failing, or not being censored) in the study. The default is `eventprob(1)`, meaning that all subjects experience an event (or fail) in the study; that is, no censoring of subjects occurs.

`failprob(`*numlist*`)` is a synonym for `eventprob()`.

`wdprob(`*numlist*`)` specifies the proportion of subjects anticipated to withdraw from a study. The default is `wdprob(0)`. `wdprob()` is allowed only with sample-size computation.

`effect(`*effect*`)` specifies the type of the effect size to be reported in the output as `delta`. *effect* is one of `coefficient`, `hratio`, or `lnhratio`. By default, the effect size `delta` is the regression coefficient, `effect(coefficient)`.

`direction()`, `onesided`, `parallel`; see [PSS-2] **power**. `direction(lower)` is the default.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

The following option is available with `power cox` but is not shown in the dialog box:

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `power cox` command and the methodology for power and sample-size analysis for survival analyses that use Cox PH models. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis, and see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests. See *Survival data* in [PSS-2] **Intro (power)** for an introduction to power and sample-size analysis for survival data.

Introduction

Consider a survival study for which the goal is to investigate the effect of a covariate of interest, x_1, on time to failure, possibly adjusted for other predictors, $x_2, x_3, \ldots, x_p$, using the Cox PH model (Cox 1972). The effect is commonly measured as a hazard ratio Δ associated with a one-unit increase in x_1 when the other covariates $x_2, x_3, \ldots, x_p$ are held constant. For a binary predictor x_1, hazard ratio Δ corresponds to the two categories of x_1 when the other covariates are held constant.

In a Cox PH model, the hazard function is assumed to be

$$h(t) = h_0(t)\exp(\beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p)$$

where no distributional assumption is made about the baseline hazard $h_0(t)$. Under this assumption, the regression coefficient β_1 is the log hazard-ratio $\ln(\Delta)$ associated with a one-unit increase in x_1 when the other predictors are held constant, and the exponentiated regression coefficient $\exp(\beta_1)$ is the hazard ratio Δ. Therefore, the effect of x_1 on time to failure can be investigated by performing an appropriate test based on the partial likelihood (Hosmer, Lemeshow, and May 2008; Klein and Moeschberger 2003) for the regression coefficient β_1 from a Cox model. Negative values of β_1 correspond to the reduction in hazard for a one-unit increase in x_1, and, conversely, positive values correspond to the increase in hazard for a one-unit increase in x_1.

`power cox` provides power and sample-size analysis for a test of the regression coefficient β_1 in a Cox model with the null hypothesis H_0: $(\beta_1, \beta_2, \ldots, \beta_p) = (0, \beta_2, \ldots, \beta_p)$ against the alternative H_a: $(\beta_1, \beta_2, \ldots, \beta_p) = (\beta_{1a}, \beta_2, \ldots, \beta_p)$. The postulated or null coefficient value is zero and a hypothesized or alternative coefficient value, specified as *b1* with `power cox`, is β_{1a}. (Similarly, Δ_a is a hypothesized or alternative value of the hazard ratio specified in the `hratio()` option.) The methods used are derived for the score test of H_0 versus H_a. In practice, however, the obtained results may be used in the context of the Wald test as well because the two tests usually lead to the same conclusions about the significance of the regression coefficient. Refer to *The conditional versus unconditional approaches* in [PSS-2] **power exponential** for more details about the results based on conditional and unconditional tests. From now on, we will refer to H_a as H_a: $\beta_1 = \beta_{1a}$ for simplicity.

`power cox` implements the method of Hsieh and Lavori (2000) for the sample-size and power computation, which reduces to the method of Schoenfeld (1983) for a binary covariate. The sample size is related to the power of a test through the number of events observed in the study; that is, for a fixed number of events, the power of a test is independent of the sample size. As a result, the sample size is estimated as the number of events divided by the overall probability of a subject failing in the study.

You can use `power cox` to

- compute required number of events and sample size when you know power and effect size expressed as a hazard ratio or regression coefficient (log hazard-ratio);
- compute power when you know sample size (number of events) and effect size expressed as a hazard ratio or regression coefficient (log hazard-ratio); or
- compute effect size and display it as a regression coefficient, a log hazard-ratio, or a hazard ratio when you know sample size (number of events) and power.

You can also account for the dependence between the covariate of interest and other model covariates in your analysis, adjust results for censoring, and adjust results for withdrawal of subjects from the study.

Using power cox

`power cox` computes sample size, power, or effect size for a test of one regression coefficient in a Cox PH model, holding coefficients of the other covariates constant. All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option. You can specify the `onesided` option to request a one-sided test.

To compute sample size, you specify an effect size and, optionally, power of the test in the `power()` option. The default power is set to 0.8. By default, the computed sample size is rounded up. You can specify the `nfractional` option to see the corresponding fractional sample size; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. The `nfractional` option is allowed only for sample-size determination.

To compute power, you must specify the sample size in the `n()` option and an effect size.

An effect size may be specified either as a regression coefficient supplied as the command argument *b1* or as a hazard ratio supplied in the `hratio()` option. If neither is specified, a hazard ratio of 0.5 is assumed.

To compute effect size, which may be expressed either as a regression coefficient (log hazard-ratio) or hazard ratio, you must specify the sample size in the `n()` option; the power in the `power()` option; and, optionally, the direction of the effect. The direction is lower by default, `direction(lower)`, which means, for example, that the target regression coefficient is assumed to be negative. This is equivalent to the hazard ratio being less than one. You can change the direction to upper, which means that the target regression coefficient is assumed to be positive, by specifying the `direction(upper)` option. This is equivalent to the hazard ratio being greater than one.

As we mentioned above, the effect size for `power cox` may be expressed as a regression coefficient or, equivalently, a log hazard-ratio, or as a hazard ratio. By default, the effect size, which is labeled as `delta` in the output, corresponds to the regression coefficient. You can change this by specifying the `effect()` option: `effect(coefficient)` (the default) reports the regression coefficient, `effect(hratio)` reports the hazard ratio, and `effect(lnhratio)` reports the log hazard-ratio.

The standard deviation of the covariate of interest is set to 0.5 by default and may be changed by specifying the `sd()` option. In the presence of additional covariates in a Cox model, you can use the `r2()` option to specify the correlation between the covariate of interest and other covariates in the model.

All computations assume no censoring. In the presence of censoring, you can use the `eventprob()` option to specify an overall probability of an event or failure. When computing sample size, you can also adjust for withdrawal of subjects from the study by specifying the anticipated proportion of withdrawals in the `wdprob()` option.

In the following sections, we describe the use of `power cox` accompanied by examples for computing sample size, power, and effect size.

Computing sample size

To compute sample size and number of events, you must specify an effect size (a regression coefficient or a hazard ratio) and, optionally, the power of the test in the `power()` option. A default power of 0.8 is assumed if `power()` is not specified. A hazard ratio of 0.5 is assumed if an effect size is not specified.

Computing sample size in the absence of censoring

First, consider a type I study in which all subjects fail by the end of the study (no censoring). Then the required sample size is the same as the number of events required to be observed in a study.

Example 1: Sample size for a model with a binary covariate of interest

Consider a survival study for which the goal is to investigate the effect of a treatment on survival times of subjects. The covariate of interest is binary with levels defining whether a subject receives the treatment (the experimental group) or a placebo (the control or placebo group). Prior to conducting the study, investigators need an estimate of the sample size that ensures that a ratio of hazards of the experimental group to the control group of 0.5 ($\beta_{1a} = \ln(0.5) = -0.6931$) can be detected with a power of 80% with a two-sided, 5%-level test. Under 1:1 randomization, a subject has a 50% chance of receiving the treatment. The corresponding binary covariate follows a Bernoulli distribution with the probability of a subject receiving a treatment, p, equal to 0.5. As such, the standard deviation of the covariate is $\{p(1-p)\}^{1/2} = 0.5$. Because these study parameters correspond to default values of `power cox`, to obtain the sample size for the above study, we simply type

```
. power cox

Estimated sample size for Cox PH regression
Wald test
H0: beta1 = 0  versus  Ha: beta1 != 0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =   -0.6931  (coefficient)
       hratio =    0.5000
           sd =    0.5000

Censoring:

         Pr_E =    1.0000

Estimated number of events and sample size:

            E =        66
            N =        66
```

Recall that, by default, a hazard ratio of 0.5, corresponding to the regression coefficient of $\ln(0.5) = -0.6931$, is assumed. From the output, we see that 66 events (failures) are required to be observed in the study to ensure a power of 80% to detect an alternative H_a: $\beta_1 = -0.6931$ using a two-sided test with a 0.05 significance level. Because we have no censoring (`Pr_E = 1.0000`), a total of 66 subjects is needed in the study to observe 66 events.

Example 2: Alternative ways of specifying effect

In example 1, the effect size `delta` corresponds to the regression coefficient, the default. We can use the `effect(hratio)` option to redefine `delta` as a hazard ratio.

```
. power cox, effect(hratio)
Estimated sample size for Cox PH regression
Wald test
H0: beta1 = 0  versus  Ha: beta1 != 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.5000  (hazard ratio)
       hratio =    0.5000
           sd =    0.5000
Censoring:
         Pr_E =    1.0000
Estimated number of events and sample size:
            E =        66
            N =        66
```

We can also obtain the same results as in example 1 by directly specifying the value of the coefficient.

```
. power cox -0.6931
Estimated sample size for Cox PH regression
Wald test
H0: beta1 = 0  versus  Ha: beta1 != 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =   -0.6931  (coefficient)
           b1 =   -0.6931
           sd =    0.5000
Censoring:
         Pr_E =    1.0000
Estimated number of events and sample size:
            E =        66
            N =        66
```

The specified regression coefficient, `b1`, is reported in the output instead of the hazard ratio `hratio`.

◁

Suppose now that the covariate of interest, x_1, is continuous. Hsieh and Lavori (2000) extend the formula of Schoenfeld (1983) for the number of events to the case when a covariate is continuous. They also relax the assumption of Schoenfeld (1983) about the independence of x_1 of other covariates and provide an adjustment to the estimate of the number of events for possible correlation.

▷ Example 3: Sample size for a model with a continuous covariate of interest

Consider an example from Hsieh and Lavori (2000) of a study of multiple-myeloma patients treated with alkylating agents (Krall, Uthoff, and Harley 1975). Although in the original study of multiple-myeloma patients, 17 of a total of 65 patients are censored; here we assume that all patients die by the end of the study (a type I study, no censoring). Suppose that the covariate of interest, x_1, is the log of the amount of blood urea nitrogen (BUN) measured in a patient. The sample size for a one-sided, 5%-level test to detect a coefficient (log hazard-ratio) of 1 for a unit increase in x_1 with a power of 80% is required. The standard deviation of x_1 is 0.3126. To obtain an estimate of the sample size, we supply *b1*, 1, as an argument, the `sd(0.3126)` option, and the `onesided` option to request a one-sided test.

```
. power cox 1, sd(0.3126) onesided
Estimated sample size for Cox PH regression
Wald test
H0: beta1 = 0  versus  Ha: beta1 > 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    1.0000  (coefficient)
           b1 =    1.0000
           sd =    0.3126
Censoring:
         Pr_E =    1.0000
Estimated number of events and sample size:
            E =        64
            N =        64
```

The estimate of the required number of events and the sample size is 64.

◁

Based on the derivation in Schoenfeld (1983) and Hsieh and Lavori (2000), sample-size estimates in the above examples may be used if other covariates are also present in the model as long as these covariates are independent of the covariate of interest. The independence assumption holds for randomized studies, but it is not true for nonrandomized studies often encountered in practice. Also, in many studies, the main covariate of interest will often be correlated with other covariates. For example, age and gender will often be confounded with the covariate of interest, such as smoking. Below we investigate the effect of the confounding factor on the estimate of the required number of events.

▷ Example 4: Sample size when covariates are not independent

Continuing with example 3, suppose that we want to adjust the effect of the covariate BUN for eight other covariates in the model. Hsieh and Lavori (2000) report the coefficient of determination of $R^2 = 0.1837$ from regression of the log of BUN, x_1, on the eight other covariates. We specify this value in the `r2()` option.

```
. power cox 1, sd(0.3126) onesided r2(0.1837)
Estimated sample size for Cox PH regression
Wald test
H0: beta1 = 0  versus  Ha: beta1 > 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    1.0000  (coefficient)
           b1 =    1.0000
           sd =    0.3126
           R2 =    0.1837
Censoring:
         Pr_E =    1.0000
Estimated number of events and sample size:
            E =        78
            N =        78
```

The number of events required to be observed in the study increases from 64 to 78 because of the correlation between BUN and the other covariates. The new estimate is equal to the original estimate multiplied by the variance inflation factor VIF $= 1/(1-R^2)$. Likewise, the number of subjects increases from 64 to 78.

◁

Computing sample size in the presence of censoring

In the previous section, we assumed that all subjects experience an event by the end of the study. In practice, the study often terminates after a fixed time, T. As a result, some subjects may not experience an event by the end of the study (a type II study). These subjects are censored. To obtain an estimate of the sample size in the presence of censoring, an estimate of the overall probability of a subject not being censored is required. The investigator may already have such an estimate from previous studies, or this probability may be computed as suggested in the literature (Schoenfeld [1983], Lachin and Foulkes [1986], Barthel et al. [2006], and Barthel, Royston, and Babiker [2005], also see [PSS-2] **power logrank** and [PSS-2] **power exponential**).

▷ Example 5: Sample size in the presence of censoring

Consider the study from example 3. In reality, as mentioned earlier, 17 of a total of 65 patients survived until the end of the study. The overall deathrate is estimated as $1 - 17/65 = 0.738$. We specify the overall probability of an event in the `eventprob()` option.

```
. power cox 1, sd(0.3126) onesided eventprob(0.738)

Estimated sample size for Cox PH regression
Wald test
H0: beta1 = 0  versus  Ha: beta1 > 0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    1.0000  (coefficient)
           b1 =    1.0000
           sd =    0.3126

Censoring:

         Pr_E =    0.7380

Estimated number of events and sample size:

            E =        64
            N =        86
```

In the presence of censoring, the number of subjects required in the study increases from 64 to 86. The number of events remains the same (64) because the only change in the study is the presence of censoring, and censoring is assumed to be independent of failure (event) times.

◁

▷ Example 6: Sample size in the presence of censoring adjusting for other covariates

Continuing with example 5, if we also adjust for the correlation between the log of BUN and other covariates, we obtain the estimate of the sample size to be 106.

```
. power cox, hratio(2.7182) sd(0.3126) onesided eventprob(0.738) r2(0.1837)
Estimated sample size for Cox PH regression
Wald test
H0: beta1 = 0  versus  Ha: beta1 > 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    1.0000  (coefficient)
       hratio =    2.7182
           sd =    0.3126
           R2 =    0.1837
Censoring:
         Pr_E =    0.7380
Estimated number of events and sample size:
            E =        78
            N =       106
```

In the above example, for no other reason than variety, rather than supplying the coefficient of 1, we used the `hratio()` option to specify the size of the effect expressed as the hazard ratio $\exp(1) = 2.7182$. ◁

❑ Technical note

Supplying the coefficient (log hazard-ratio) of 1 or -1 [or, respectively, the hazard ratio of $\exp(1) = 2.7182$ or $\exp(-1) = 1/2.7182 = 0.36788$] is irrelevant for sample-size and power determination because it results in the same estimates of sample size and power. However, the sign of the coefficient (or the value of the hazard ratio being larger or smaller than one) is important at the analysis stage because it determines the direction of the effect associated with a one-unit increase of a covariate value.

❑

Often, in practice, subjects may withdraw from a study before it terminates. As a result, the information about the subjects' response is lost. The proportion of subjects anticipated to withdraw from a study may be specified by using `wdprob()`. Refer to *Survival data* in [PSS-2] **Intro (power)** and *Withdrawal of subjects from the study* in [PSS-2] **power logrank** for a more detailed description and an example.

Link to the sample-size and power computation for the log-rank test

The score test of the regression coefficient of a binary covariate in a Cox model with one binary predictor is the same (in the absence of tied observations) as the log-rank test comparing survivor functions of two groups defined by this covariate. Powers of the two tests are the same and so are the formulas for the number of events (Schoenfeld 1983; Schoenfeld 1981). As such, the required number of events computed by `power cox` for the Cox model and the Schoenfeld method of `power logrank` for the log-rank test are the same.

▷ Example 7: Using power logrank for a binary covariate

Using `power logrank` (see [PSS-2] **power logrank**) for the study described in example 1 yields the same estimates of 66 for both the required number of events and the required sample size.

```
. power logrank, schoenfeld
Estimated sample sizes for two-sample comparison of survivor functions
Log-rank test, Schoenfeld method
H0: ln(HR) = 0  versus  Ha: ln(HR) != 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =   -0.6931  (log hazard-ratio)
       hratio =    0.5000
Censoring:
         Pr_E =    1.0000
Estimated number of events and sample sizes:
            E =        66
            N =        66
  N per group =        33
```

Schoenfeld (1983) demonstrates that the same formula can also be used to compute the required number of events when other covariates are present in a Cox model, provided that the binary covariate of interest is independent of these covariates and that the covariates are not extremely unbalanced. Although the formulas for the number of events are the same whether covariates are present or not, it is important to adjust for covariates when analyzing the data to avoid loss of power; see Schoenfeld (1983) for details.

Væth and Skovlund (2004) demonstrate that for a continuous covariate of interest in a Cox model, the sample-size formula for the log-rank test with the value of the hazard ratio equal to $\exp(2\beta_{1a}\sigma)$ and with equal-group allocation may be used to obtain the required sample size. Indeed, by substituting the above expression for the hazard ratio into the sample-size formula for the log-rank test, one obtains the sample-size formula derived in Hsieh and Lavori (2000) for a Cox model.

Example 8: Using power logrank for a continuous covariate

For example, we obtain the same estimate of the total number of events as computed in example 3 by using `power logrank` with the `schoenfeld` option and with the value of the hazard ratio equal to $\exp(2\beta_{1a}\sigma) = \exp(2 \times 1 \times 0.3126) = 1.8686$.

```
. power logrank, hratio(1.8686) onesided schoenfeld
Estimated sample sizes for two-sample comparison of survivor functions
Log-rank test, Schoenfeld method
H0: ln(HR) = 0  versus  Ha: ln(HR) > 0
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    0.6252  (log hazard-ratio)
       hratio =    1.8686
Censoring:
         Pr_E =    1.0000
Estimated number of events and sample sizes:
            E =        64
            N =        64
  N per group =        32
```

Computing power

To compute power, you must specify the sample size in the `n()` option and an effect size (a regression coefficient or a hazard ratio). A hazard ratio of 0.5 is assumed if an effect size is not specified.

▷ Example 9: Power determination

In example 6, the required number of patients was estimated to be 106 to ensure a power of 80% for a 0.05 one-sided test to detect a value of 1 in the regression coefficient. Suppose that we can recruit only 65 subjects. How does this reduction in sample size affect the power of the test to detect the alternative H_a: $\beta_1 = 1$?

```
. power cox 1, sd(0.3126) onesided r2(0.1837) eventprob(0.738) n(65)

Estimated power for Cox PH regression
Wald test
H0: beta1 = 0  versus  Ha: beta1 > 0

Study parameters:

        alpha =    0.0500
            N =        65
        delta =    1.0000  (coefficient)
           b1 =    1.0000
           sd =    0.3126
           R2 =    0.1837

Number of events and censoring:

            E =        48
         Pr_E =    0.7380

Estimated power:

        power =    0.6222
```

When the sample size decreases from 106 to 65, the corresponding number of events decreases from 78 to 48, and power decreases from 80% to 62%.

◁

▷ Example 10: Multiple values of study parameters

Suppose we want to investigate the effect of a correlation between a covariate of interest and other model covariates on power. Continuing with example 9, we can specify a list (see **[U] 11.1.8 numlist**) of correlations in the `r2()` option.

```
. power cox 1, sd(0.3126) onesided r2(0.1(0.1)0.5) eventprob(0.738) n(65)

Estimated power for Cox PH regression
Wald test
H0: beta1 = 0  versus  Ha: beta1 > 0
```

alpha	power	N	E	delta	b1	sd	R2	Pr_E
.05	.6588	65	48	1	1	.3126	.1	.738
.05	.6147	65	48	1	1	.3126	.2	.738
.05	.5662	65	48	1	1	.3126	.3	.738
.05	.5128	65	48	1	1	.3126	.4	.738
.05	.4547	65	48	1	1	.3126	.5	.738

As the correlation increases, the power decreases.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

Computing effect size

Effect size δ for a test of a coefficient in a Cox PH model is defined as a coefficient (or, equivalently, a log hazard-ratio) or a hazard ratio, corresponding to a one-unit change in the tested covariate holding other model covariates constant.

Sometimes, we may be interested in determining the smallest effect that yields a statistically significant result for prespecified sample size and power. In this case, both power and sample size must be specified in options `power()` and `n()`, respectively. Additionally, you may also choose the direction of the effect by specifying the `direction()` option. `direction(lower)`, the default, assumes $\beta_{1a} < 0$ (or $\Delta_a < 1$), corresponding to the reduction in hazard for a unit change in a covariate. `direction(upper)` assumes $\beta_{1a} > 0$ (or $\Delta_a > 1$).

▷ Example 11: Effect-size determination

Continuing with example 9, if a power of 62% is unacceptable to investigators, they may want to determine the smallest value of the regression coefficient that can be detected with a preserved power of 80%. To obtain this estimate, we specify both the `n()` and the `power()` options.

```
. power cox, sd(0.3126) onesided r2(0.1837) eventprob(0.738) n(65) power(0.8)
> direction(upper)

Estimated target coefficient for Cox PH regression
Wald test
H0: beta1 = 0  versus  Ha: beta1 > 0

Study parameters:

        alpha =    0.0500
        power =    0.8000
            N =        65
           sd =    0.3126
           R2 =    0.1837

Number of events and censoring:

            E =        48
         Pr_E =    0.7380

Estimated effect size and target coefficient:

        delta =    1.2711  (coefficient)
           b1 =    1.2711
```

With only 65 subjects, the smallest change in a coefficient (log hazards) for a one-unit increase in the log of BUN, which can be detected with a preserved 80% power, is roughly 1.27, corresponding to a 27% increase in the coefficient of 1 desired to be detected originally in example 6.

◁

Performing analyses using a Cox PH model

After the data are collected, one can use `stcox` and `test` to fit the Cox PH model and perform a Wald test, as we demonstrate below.

Example 12: Performing a Wald test

We demonstrate how to perform a Wald test for the regression coefficient of the log of BUN from a Cox model using the data from Krall, Uthoff, and Harley (1975) described in example 3. `myeloma.dta` consists of 11 variables, described below.

```
. use https://www.stata-press.com/data/r19/myeloma
(Multiple myeloma patients)
. describe
Contains data from https://www.stata-press.com/data/r19/myeloma.dta
 Observations:            65                  Multiple myeloma patients
    Variables:            11                  11 Feb 2024 19:26
-------------------------------------------------------------------------------
Variable      Storage   Display    Value
    name         type    format    label      Variable label
-------------------------------------------------------------------------------
time            float   %9.0g                 Survival time from diagnosis to
                                                nearest month + 1
died            byte    %9.0g                 0 - Alive, 1 - Dead
lnbun           float   %9.0g                 Log BUN at diagnosis
hemo            float   %9.0g                 Hemoglobin at diagnosis
platelet        byte    %9.0g      normal     Platelets at diagnosis
age             byte    %9.0g                 Age (complete years)
lnwbc           float   %9.0g                 Log WBC at diagnosis
fracture        byte    %9.0g      present    Fractures at diagnosis
lnbm            float   %9.0g                 Log % of plasma cells in bone
                                                marrow
protein         byte    %9.0g                 Proteinuria at diagnosis
scalcium        byte    %9.0g                 Serum calcium (mgm%)
-------------------------------------------------------------------------------
Sorted by:
```

Before using `stcox` to fit a Cox model, we need to set up the data by using `stset` (see [ST] **stset**). The analysis-time variable is `time`, and the failure variable is `died`.

```
. stset time, failure(died)
Survival-time data settings
         Failure event: died!=0 & died<.
Observed time interval: (0, time]
     Exit on or before: failure
------------------------------------------------------------------------------
         65  total observations
          0  exclusions
------------------------------------------------------------------------------
         65  observations remaining, representing
         48  failures in single-record/single-failure data
    1,560.5  total analysis time at risk and under observation
                                                At risk from t =         0
                                     Earliest observed entry t =         0
                                          Last observed exit t =        92
```

We include all nine covariates in the model and perform a fit by using `stcox`. Then we perform a Wald test of H_0: $\beta_1 = 1$ for the coefficient of `lnbun` using `test`.

```
. stcox lnbun hemo platelet age lnwbc fracture lnbm protein scalcium, nohr
        Failure _d: died
  Analysis time _t: time
Iteration 0:  Log likelihood = -154.85799
Iteration 1:  Log likelihood = -146.68114
Iteration 2:  Log likelihood = -146.29446
Iteration 3:  Log likelihood = -146.29404
Refining estimates:
Iteration 0:  Log likelihood = -146.29404
Cox regression with Breslow method for ties
No. of subjects =       65                              Number of obs =     65
No. of failures =       48
Time at risk    = 1,560.5
                                                        LR chi2(9)    =  17.13
Log likelihood = -146.29404                             Prob > chi2   = 0.0468
```

_t	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
lnbun	1.798354	.6483293	2.77	0.006	.5276519	3.069056
hemo	-.1263119	.0718333	-1.76	0.079	-.2671026	.0144789
platelet	-.2505915	.5074656	-0.49	0.621	-1.245206	.7440228
age	-.0127949	.019475	-0.66	0.511	-.0509653	.0253755
lnwbc	.3537259	.7131935	0.50	0.620	-1.044108	1.75156
fracture	.3378767	.4072774	0.83	0.407	-.4603722	1.136126
lnbm	.3589346	.4860298	0.74	0.460	-.5936663	1.311535
protein	.0130672	.0261696	0.50	0.618	-.0382243	.0643587
scalcium	.1259479	.1034015	1.22	0.223	-.0767153	.3286112

```
. test lnbun = 1
 ( 1)  lnbun = 1
           chi2(  1) =    1.52
         Prob > chi2 =    0.2182
```

By default, `stcox` reports estimates of hazard ratios and the two-sided tests of the equality of a coefficient to zero. We use the `nohr` option to request estimates of coefficients. From the output table, a one-sided test of $H_0\colon \beta_1 = 0$ versus $H_a\colon \beta_1 > 0$ is rejected at a 0.05 level (one-sided p-value is $0.006/2 = 0.003 < 0.05$). The estimate of the log-hazard difference associated with a one-unit increase of `lnbun` is $\widehat{\beta}_1 = 1.8$. From the `test` output, we cannot reject the hypothesis of $H_0\colon \beta_1 = 1$.

For these data, the observed effect size (coefficient) of 1.8 is large enough for the sample size of 65 to be sufficient to reject the null hypothesis of no effect of the BUN on the survival of subjects ($H_0\colon \beta_1 = 0$). However, if the goal of the study were to ensure that the test detects the effect size corresponding to the coefficient of at least 1 with 80% power, a sample of approximately 106 subjects would have been required.

◁

Stored results

power cox stores the following in r():

Scalars

r(alpha)	significance level
r(power)	power
r(beta)	probability of a type II error
r(delta)	effect size
r(N)	sample size
r(nfractional)	1 if nfractional is specified, 0 otherwise
r(onesided)	1 for a one-sided test, 0 otherwise
r(E)	total number of events (failures)
r(hratio)	hazard ratio under the alternative hypothesis
r(b1)	regression coefficient under the alternative hypothesis
r(sd)	standard deviation
r(R2)	squared multiple correlation (if specified)
r(Pr_E)	probability of an event (failure) (if specified)
r(Pr_w)	proportion of withdrawals (if specified)
r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise

Macros

r(type)	test
r(method)	cox
r(effect)	coefficient, hratio, or lnhratio
r(direction)	lower or upper
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results

Methods and formulas

Let β_1 denote the regression coefficient of the covariate of interest, x_1, from a Cox PH model, possibly in the presence of other covariates, $x_2, \ldots, x_p$; and let Δ denote the hazard ratio associated with a one-unit increase of x_1 when other covariates are held constant. Under the PH model, $\beta_1 = \ln(\Delta)$, where $\ln(\Delta)$ is the change in log hazards associated with a one-unit increase in x_1 when other covariates are held constant.

Define E and n to be the total number of events (failures) and the total number of subjects required in the study; σ to be the standard deviation of x_1; p_E to be the overall probability of an event (failure); R^2 to be the proportion of variance explained by the regression of x_1 on $x_2, \ldots, x_p$ (or squared multiple-correlation coefficient); p_w to be the proportion of subjects withdrawn from a study (lost to follow-up); α to be the significance level; β to be the probability of a type II error; and $z_{(1-\alpha/k)}$ and $z_{(1-\beta)}$ to be the $(1-\alpha/k)$th and the $(1-\beta)$th quantiles of the standard normal distribution, respectively, with $k = 1$ for the one-sided test and $k = 2$ for the two-sided test.

The total number of events required to be observed in a study to ensure a power of $1-\beta$ of a test to detect the regression coefficient, β_1, with a significance level α, according to Hsieh and Lavori (2000), is

$$E = \frac{(z_{1-\alpha/k} + z_{1-\beta})^2}{\sigma^2 \beta_1^2 (1 - R^2)}$$

For the case of randomized study and a binary covariate x_1, this formula was derived in Schoenfeld (1983). The formula is an approximation and relies on a set of assumptions such as distinct failure times, all subjects completing the course of the study (no withdrawal), and a local alternative under which $\ln(\Delta)$ is assumed to be of order $O(n^{-1/2})$. The formula is derived for the score test but may be applied to other tests (Wald, for example) that are based on the partial likelihood of a Cox model because all of these tests are asymptotically equivalent (Schoenfeld 1983; Hosmer, Lemeshow, and May 2008; Klein and Moeschberger 2003).

The total sample size required to observe the total number of events, E, is given by

$$n = \frac{E}{p_E}$$

If the `nfractional` option is not specified, the computed sample size is rounded up.

To account for a proportion of subjects, p_w, withdrawn from a study, a conservative adjustment to the total sample size suggested in the literature (Freedman 1982; Machin and Campbell 2005) is applied as follows:

$$n_w = \frac{n}{1 - p_w}$$

Withdrawal is assumed to be independent of administrative censoring and failure (event) times.

Power is estimated using the formula

$$1 - \beta = \Phi\left[|\beta_1|\sigma\{np_E(1 - R^2)\}^{1/2} - z_{1-\alpha/k}\right]$$

where $\Phi(\cdot)$ is the standard normal cumulative distribution function.

The estimate of the regression coefficient for a fixed power, $1 - \beta$, and a sample size, n, is computed as

$$\beta_1^2 = \frac{(z_{1-\alpha/k} + z_{1-\beta})^2}{\sigma^2 np_{\mathrm{E}}(1 - R^2)}$$

Either of the two values $|\beta_1|$ and $-|\beta_1|$ satisfy the above equation. By default or if `direction(lower)` is specified, `power cox` reports the negative of the two values, which corresponds to the reduction in a hazard of a failure for a one-unit increase in x_1. If `direction(upper)` is specified, `power cox` reports the positive of the two values, which corresponds to the increase in a hazard of a failure for a one-unit increase in x_1.

Similarly, if the `effect(hratio)` option is used, the corresponding value of the hazard ratio less than 1 for `direction(lower)` and greater than 1 for `direction(upper)` is reported to reflect, respectively, the reduction or increase in hazard for a one-unit increase in x_1.

References

Barthel, F. M.-S., A. G. Babiker, P. Royston, and M. K. B. Parmar. 2006. Evaluation of sample size and power for multi-arm survival trials allowing for non-uniform accrual, non-proportional hazards, loss to follow-up and cross-over. *Statistics in Medicine* 25: 2521–2542. https://doi.org/10.1002/sim.2517.

Barthel, F. M.-S., P. Royston, and A. G. Babiker. 2005. A menu-driven facility for complex sample size calculation in randomized controlled trials with a survival or a binary outcome: Update. *Stata Journal* 5: 123–129.

Cox, D. R. 1972. Regression models and life-tables (with discussion). *Journal of the Royal Statistical Society*, B ser., 34: 187–220. https://doi.org/10.1111/j.2517-6161.1972.tb00899.x.

Freedman, L. S. 1982. Tables of the number of patients required in clinical trials using the logrank test. *Statistics in Medicine* 1: 121–129. https://doi.org/10.1002/sim.4780010204.

Hosmer, D. W., Jr., S. A. Lemeshow, and S. May. 2008. *Applied Survival Analysis: Regression Modeling of Time to Event Data*. 2nd ed. New York: Wiley.

Hsieh, F. Y., and P. W. Lavori. 2000. Sample-size calculations for the Cox proportional hazards regression model with nonbinary covariates. *Controlled Clinical Trials* 21: 552–560. https://doi.org/10.1016/S0197-2456(00)00104-5.

Klein, J. P., and M. L. Moeschberger. 2003. *Survival Analysis: Techniques for Censored and Truncated Data*. 2nd ed. New York: Springer.

Krall, J. M., V. A. Uthoff, and J. B. Harley. 1975. A step-up procedure for selecting variables associated with survival. *Biometrics* 31: 49–57. https://doi.org/10.2307/2529709.

Lachin, J. M., and M. A. Foulkes. 1986. Evaluation of sample size and power for analyses of survival with allowance for nonuniform patient entry, losses to follow-up, noncompliance, and stratification. *Biometrics* 42: 507–519. https://doi.org/10.2307/2531201.

Machin, D., and M. J. Campbell. 2005. *Design of Studies for Medical Research*. Chichester, UK: Wiley.

Schoenfeld, D. A. 1981. The asymptotic properties of nonparametric tests for comparing survival distributions. *Biometrika* 68: 316–319. https://doi.org/10.2307/2335833.

———. 1983. Sample-size formula for the proportional-hazards regression model. *Biometrics* 39: 499–503. https://doi.org/10.2307/2531021.

Væth, M., and E. Skovlund. 2004. A simple approach to power and sample size calculations in logistic regression and Cox regression models. *Statistics in Medicine* 23: 1781–1792. https://doi.org/10.1002/sim.1753.

Also see [PSS-2] **Intro (power)** for more references.

Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power exponential** — Power analysis for a two-sample exponential test

[PSS-2] **power logrank** — Power analysis for the log-rank test

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[R] **test** — Test linear hypotheses after estimation

[ST] **stcox** — Cox proportional hazards model

[ST] **sts test** — Test equality of survivor functions

power exponential — Power analysis for a two-sample exponential test

Description

`power exponential` computes sample size or power for survival analysis comparing two exponential survivor functions by using parametric tests for the difference between hazards or, optionally, for the difference between log hazards. It accommodates unequal allocation between the two groups, flexible accrual of subjects into the study, and group-specific losses to follow-up. The accrual distribution may be chosen to be uniform or truncated exponential over a fixed accrual period. Losses to follow-up are assumed to be exponentially distributed. Also the computations may be performed using the conditional or the unconditional approach.

Quick start

Sample size for a test of exponential hazard rates H_0: $\lambda_2 = \lambda_1$ versus H_a: $\lambda_2 \neq \lambda_1$ given control-group hazard rate $h_1 = 0.4$, experimental-group hazard rate $h_2 = 0.2$, equal group sizes, and no censoring using default power of 0.8 and significance level $\alpha = 0.05$

```
power exponential .4 .2
```

Same as above, specified using a hazard ratio of 0.5 instead of the experimental-group hazard rate

```
power exponential .4, hratio(.5)
```

Same as above, but specify hazard ratios of 0.4, 0.45, 0.5, and 0.55

```
power exponential .4, hratio(.4(.05).55)
```

Same as above, but display results in a graph

```
power exponential .4, hratio(.4(.05).55) graph
```

Total and per group sample sizes given twice as many observations in the experimental group as in the control group

```
power exponential .4 .2, nratio(2)
```

Sample size with survival probabilities $s_1 = 0.65$ and $s_2 = 0.8$ and reference survival time 2

```
power exponential .65 .8, time(2)
```

Same as above, specified using survival probability s_1 and a hazard ratio

```
power exponential .65, time(2) hratio(.52)
```

Same as above, but for a one-sided test with power of 0.95

```
power exponential .65, time(2) hratio(.52) onesided power(.95)
```

Sample size for a test of a log hazard-ratio given $h_1 = 0.4$ and hazard ratio of 0.5

```
power exponential .4, hratio(.5) loghazard
```

Same as above, but specify corresponding survival probabilities s_1 and s_2 at reference time 2

```
power exponential .45 .67, time(2) loghazard
```

Sample size for a design with a 10-year follow-up period and a 1-year accrual period

```
power exponential .4 .2, fperiod(10) aperiod(1)
```

Power for a test of H_0: $\lambda_2 = \lambda_1$, with $h_1 = 0.4$, $h_2 = 0.2$, a sample size of 80, and default $\alpha = 0.05$

```
power exponential .4 .2, n(80)
```

Power for a test of the log hazard-ratio with $\alpha = 0.01$

```
power exponential .4, hratio(.5) loghazard n(200) alpha(.01)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

Specify hazard rates

```
power exponential [ h1 [ h2 ] ] [ , power(numlist) options ]
```

Specify survival probabilities

```
power exponential s1 [ s2 ] , time(#) [ power(numlist) options ]
```

Compute power

Specify hazard rates

```
power exponential [ h1 [ h2 ] ], n(numlist) [ options ]
```

Specify survival probabilities

```
power exponential s1 [ s2 ] , time(#) n(numlist) [ options ]
```

where

h_1 is the hazard rate in the control group;

h_2 is the hazard rate in the experimental group;

s_1 is the survival probability in the control group at reference (base) time t; and

s_2 is the survival probability in the experimental group at reference (base) time t.

h_1, h_2 and s_1, s_2 may each be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

options	Description
Main	
* time(*numlist*)	reference time t for survival probabilities s_1 and s_2
* alpha(*numlist*)	significance level; default is alpha(0.05)
* power(*numlist*)	power; default is power(0.8)
* beta(*numlist*)	probability of type II error; default is beta(0.2)
* n(*numlist*)	total sample size; required to compute power
* n1(*numlist*)	sample size of the control group
* n2(*numlist*)	sample size of the experimental group
* nratio(*numlist*)	ratio of sample sizes, N2/N1; default is nratio(1), meaning equal group sizes
nfractional	allow fractional sample sizes
* hratio(*numlist*)	hazard ratio (effect size) of the experimental to the control group; default is hratio(0.5); may not be combined with lnhratio() or hdifference()
* lnhratio(*numlist*)	log hazard-ratio (effect size) of the experimental to the control group; may not be combined with hratio() or hdifference()
* hdifference(*numlist*)	difference between the experimental-group and control-group hazard rates (effect size); may not be combined with hratio() or lnhratio()
loghazard	power or sample-size computation for the test of the difference between log hazards; default is the test of the difference between hazards
unconditional	power or sample-size computation using the unconditional approach
effect(*effect*)	specify the type of effect to display; default is method specific
onesided	one-sided test; default is two sided
parallel	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)

Accrual/Follow-up

* studytime(*numlist*)	duration of the study; if not specified, the study is assumed to continue until all subjects experience an event (fail)
* fperiod(*numlist*)	length of the follow-up period; if not specified, the study is assumed to continue until all subjects experience an event (fail)
* aperiod(*numlist*)	length of the accrual period; default is aperiod(0), meaning no accrual
* aprob(*numlist*)	proportion of subjects accrued by time t_a under truncated exponential accrual; default is aprob(0.5)
* aptime(*numlist*)	proportion of the accrual period, t_a/aperiod(), by which proportion of subjects in aprob() is accrued; default is aptime(0.5)
* atime(*numlist*)	reference accrual time t_a by which the proportion of subjects in aprob() is accrued; default value is 0.5×aperiod()
* ashape(*numlist*)	shape of the truncated exponential accrual distribution; default is ashape(0), meaning uniform accrual
* lossprob(*numlist*)	proportion of subjects lost to follow-up by time losstime() in the control and the experimental groups; default is lossprob(0), meaning no losses to follow-up
* lossprob1(*numlist*)	proportion of subjects lost to follow-up by time losstime() in the control group; default is lossprob1(0), meaning no losses to follow-up in the control group
* lossprob2(*numlist*)	proportion of subjects lost to follow-up by time losstime() in the experimental group; default is lossprob2(0), meaning no losses to follow-up in the experimental group
* losstime(*numlist*)	reference time t_L by which the proportion of subjects specified in lossprob(), lossprob1(), or lossprob2() is lost to follow-up; default is losstime(1)
* losshaz(*numlist*)	loss hazard rates in the control and the experimental groups; default is losshaz(0), meaning no losses to follow-up
* losshaz1(*numlist*)	loss hazard rates in the control group; default is losshaz1(0), meaning no losses to follow-up in the control group
* losshaz2(*numlist*)	loss hazard rates in the experimental group; default is losshaz2(0), meaning no losses to follow-up in the experimental group

Table	
[no]table[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
saving(*filename* [, replace])	save the table data to *filename*; use replace to overwrite existing *filename*
Graph	
graph[(*graphopts*)]	graph results; see [PSS-2] **power, graph**
Reporting	
show	display group-specific numbers of events and, in the presence of loss to follow-up, numbers of losses
show(*showspec*)	display group-specific numbers of events, numbers of losses, and event probabilities
notitle	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the parallel option.

collect is allowed; see **[U] 11.1.10 Prefix commands**.

notitle does not appear in the dialog box.

effect	Description
hratio	hazard ratio
lnhratio	log hazard-ratio
hdifference	difference between hazard rates
lnhdifference	difference between log hazard-rates (equivalent to log hazard-ratio)
difference	synonym for hdifference

showspec	Description
events	numbers of events
losses	numbers of losses
eventprobs	event probabilities
all	all the above

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`N`	total number of subjects	N
`N1`	number of subjects in the control group	N_1
`N2`	number of subjects in the experimental group	N_2
`nratio`	ratio of sample sizes, experimental to control	N_2/N_1
`delta`	effect size	δ
`s1`	survival probability in the control group	$S_1(t)$
`s2`	survival probability in the experimental group	$S_2(t)$
`time`	reference survival time	t
`h1`	hazard rate in the control group	λ_1
`h2`	hazard rate in the experimental group	λ_2
`hdiff`	difference between hazard rates	$\lambda_2-\lambda_1$
`hratio`	hazard ratio	Δ
`lnhratio`	log hazard-ratio	$\ln(\Delta)$
`studytime`	duration of a study	T
`fperiod`	follow-up period	f
`aperiod`	accrual period	r
`aprob`	proportion of subjects accrued by time `atime` (or by `aptime`$\times$100% of accrual period)	p_a
`aptime`	proportion of an accrual period by which `aprob`$\times$100% of subjects are accrued	t_a/r
`atime`	reference accrual time	t_a
`ashape`	shape of the accrual distribution	γ
`E0`	total number of events under H_0	E_0
`E01`	number of events in the control group under H_0	E_{01}
`E02`	number of events in the experimental group under H_0	E_{02}
`Ea`	total number of events under H_a	E_a
`Ea1`	number of events in the control group under H_a	E_{a1}
`Ea2`	number of events in the experimental group under H_a	E_{a2}
`Pr_E01`	control-group probability of an event under H_0	Pr_E_{01}
`Pr_E02`	experimental-group probability of an event under H_0	Pr_E_{02}
`Pr_Ea1`	control-group probability of an event under H_a	Pr_E_{a1}
`Pr_Ea2`	experimental-group probability of an event under H_a	Pr_E_{a2}
`lossprob`	proportion of subjects lost to follow-up in the control and experimental groups	$L(t_L)$
`lossprob1`	proportion of subjects lost to follow-up in the control group	$L_1(t_L)$
`lossprob2`	proportion of subjects lost to follow-up in the experimental group	$L_2(t_L)$

`losstime`	reference loss-to-follow-up time	t_L
`losshaz`	loss hazard rate in the control and experimental groups	η
`losshaz1`	loss hazard rate in the control group	η_1
`losshaz2`	loss hazard rate in the experimental group	η_2
`L0`	total number of losses under H_0	L_0
`L01`	number of losses in the control group under H_0	L_{01}
`L02`	number of losses in the experimental group under H_0	L_{02}
`La`	total number of losses under H_a	L_a
`La1`	number of losses in the control group under H_a	L_{a1}
`La2`	number of losses in the experimental group under H_a	L_{a2}
`target`	target parameter; synonym for `h2` or `hratio`	
`_all`	display all supported columns	

Column `beta` is shown in the default table in place of column `power` if option `beta()` is specified.

Column `hratio` is shown in the default table if option `hratio()` is specified or implied by the command.

Columns `nratio` and `lnhratio` are shown in the default table if the corresponding options are specified.

Columns `h1`, `h2`, `s1`, and `s2` are available and are shown in the default table when the corresponding command arguments are specified.

Columns `time`, `studytime`, `fperiod`, `aperiod`, `aprob`, `aptime`, `atime`, `ashape`, `losshaz`, `losshaz1`, `losshaz2`, `lossprob`, `lossprob1`, `lossprob2`, and `losstime` are available and are shown in the default table when the corresponding options are specified.

Columns containing numbers of events, numbers of losses, and probabilities of an event are displayed if specified or if respective options `show(events)`, `show(losses)`, or `show(eventprobs)` are specified. If `show` is specified, numbers of events and losses are displayed. If `show(all)` is specified, numbers of events, numbers of losses, and probabilities are displayed.

Options

Main

`time(#)` specifies a fixed time t (reference survival time) such that the proportions of subjects in the control and experimental groups still alive past this time point are as specified in s_1 and s_2. If this option is specified, the input parameters s_1 and s_2 are the survival probabilities $S_1(t)$ and $S_2(t)$. Otherwise, the input parameters are assumed to be hazard rates λ_1 and λ_2 given as h_1 and h_2, respectively.

`alpha()`, `power()`, `beta()`, `n()`, `n1()`, `n2()`, `nratio()`, `nfractional`; see [PSS-2] **power**.

`hratio(`*numlist*`)` specifies the hazard ratio (effect size) of the experimental group to the control group. The default is `hratio(0.5)`. This value typically defines the clinically significant improvement of the experimental procedure over the control procedure desired to be detected by a test with a certain power. If h_1 and h_2 (or s_1 and s_2) are given, `hratio()` is not allowed and the hazard ratio is computed as h_2/h_1 [or $\ln(s_2)/\ln(s_1)$]. Also see *Alternative ways of specifying effect* for various specifications of an effect size.

This option is not allowed with the effect-size determination and may not be combined with `lnhratio()` or `hdifference()`.

`lnhratio(`*numlist*`)` specifies the log hazard-ratio (effect size) of the experimental group to the control group. This value typically defines the clinically significant improvement of the experimental procedure over the control procedure desired to be detected by a test with a certain power. If h_1 and h_2 (or s_1 and s_2) are given, `lnhratio()` is not allowed and the log hazard-ratio is computed as $\ln(h_2/h_1)$ [or $\ln\{\ln(s_2)/\ln(s_1)\}$]. Also see *Alternative ways of specifying effect* for various specifications of an effect size.

This option is not allowed with the effect-size determination and may not be combined with `hratio()` or `hdifference()`.

`hdifference(`*numlist*`)` specifies the difference between the experimental-group hazard rate and the control-group hazard rate. It requires that the control-group hazard rate, the command argument h_1, is specified. `hdifference()` provides a way of specifying an effect size; see *Alternative ways of specifying effect* for details.

This option is not allowed with the effect-size determination and may not be combined with `hratio()` or `lnhratio()`.

`loghazard` requests sample-size or power computation for the test of the difference between log hazards (or the log hazard-ratio test). This option implies uniform accrual. By default, the test of the difference between hazards is assumed.

`unconditional` requests that the unconditional approach be used for sample-size or power computation; see *The conditional versus unconditional approaches* and *Methods and formulas* for details.

`effect(`*effect*`)` specifies the type of the effect size to be reported in the output as `delta`. *effect* is one of `hratio`, `lnhratio`, `hdifference`, or `lnhdifference`. By default, the effect size `delta` is a hazard ratio, `effect(hratio)`, for a hazard-ratio test and a log hazard-ratio, `effect(lnhratio)`, for a log hazard-ratio test (when `schoenfeld` is specified).

`onesided`, `parallel`; see [PSS-2] **power**.

Accrual/Follow-up

`studytime(`*numlist*`)` specifies the duration of the study, T. By default, it is assumed that subjects are followed up until the last subject experiences an event (fails). The (minimal) follow-up period is defined as the length of the period after the recruitment of the last subject to the study until the end of the study. If r is the length of an accrual period and f is the length of the follow-up period, then $T = r+f$. You can specify only two of the three options `studytime()`, `fperiod()`, and `aperiod()`.

`fperiod(`*numlist*`)` specifies the follow-up period of the study, f. By default, it is assumed that subjects are followed up until the last subject experiences an event (fails). The (minimal) follow-up period is defined as the length of the period after the recruitment of the last subject to the study until the end of the study. If T is the duration of a study and r is the length of an accrual period, then the follow-up period is $f = T - r$. You can specify only two of the three options `studytime()`, `fperiod()`, and `aperiod()`.

`aperiod(`*numlist*`)` specifies the accrual period, r, during which subjects are to be recruited into the study. The default is `aperiod(0)`, meaning no accrual. You can specify only two of the three options `studytime()`, `fperiod()`, and `aperiod()`.

`aprob(`*numlist*`)` specifies the proportion of subjects expected to be accrued by time t^* according to the truncated exponential distribution. The default is `aprob(0.5)`. This option is useful when the shape parameter is unknown but the proportion of accrued subjects at a certain time is known. `aprob()` is often used in conjunction with `aptime()` or `atime()`. This option may not be specified with `ashape()` or `loghazard` and requires specifying a nonzero accrual period in `aperiod()`.

`aptime(`*numlist*`)` specifies the proportion of the accrual period, t^*/r, by which the proportion of subjects specified in `aprob()` is expected to be accrued according to the truncated exponential distribution. The default is `aptime(0.5)`. This option may not be combined with `atime()`, `ashape()`, or `loghazard` and requires specifying a nonzero accrual period in `aperiod()`.

`atime(`*numlist*`)` specifies the time point t^*, reference accrual time, by which the proportion of subjects specified in `aprob()` is expected to be accrued according to the truncated exponential distribution. The default value is $0.5 \times r$. This option may not be combined with `aptime()`, `ashape()`, or `loghazard` and requires specifying a nonzero accrual period in `aperiod()`. The value in `atime()` may not exceed the value in `aperiod()`.

`ashape(`*numlist*`)` specifies the shape, γ, of the truncated exponential accrual distribution. The default is `ashape(0)`, meaning uniform accrual. This option is not allowed in conjunction with `loghazard` and requires specifying a nonzero accrual period in `aperiod()`.

`lossprob(`*numlist*`)` specifies the proportion of subjects lost to follow-up by time `losstime()` in the control and the experimental groups. The default is `lossprob(0)`, meaning no losses to follow-up. This option requires specifying `aperiod()` or `fperiod()` and may not be combined with `lossprob1()`, `lossprob2()`, `losshaz()`, `losshaz1()`, or `losshaz2()`.

`lossprob1(`*numlist*`)` specifies the proportion of subjects lost to follow-up by time `losstime()` in the control group. The default is `lossprob1(0)`, meaning no losses to follow-up in the control group. This option requires specifying `aperiod()` or `fperiod()` and may not be combined with `lossprob()`, `losshaz()`, `losshaz1()`, or `losshaz2()`.

`lossprob2(`*numlist*`)` specifies the proportion of subjects lost to follow-up by time `losstime()` in the experimental group. The default is `lossprob2(0)`, meaning no losses to follow-up in the experimental group. This option requires specifying `aperiod()` or `fperiod()` and may not be combined with `lossprob()`, `losshaz()`, `losshaz1()`, or `losshaz2()`.

`losstime(`*numlist*`)` specifies the time at which the proportion of subjects specified in `lossprob()` or `lossprob1()` and `lossprob2()` is lost to follow-up, also referred to as the reference loss to follow-up time. The default is `losstime(1)`. This option requires specifying `lossprob()`, `lossprob1()`, or `lossprob2()`.

`losshaz(`*numlist*`)` specifies an exponential hazard rate of losses to follow-up common to both the control and the experimental groups. The default is `losshaz(0)`, meaning no losses to follow-up. This option requires specifying `aperiod()` or `fperiod()` and may not be combined with `lossprob()`, `lossprob1()`, `lossprob2()`, `losshaz1()`, or `losshaz2()`.

`losshaz1(`*numlist*`)` specifies an exponential hazard rate of losses to follow-up, η_1, in the control group. The default is `losshaz1(0)`, meaning no losses to follow-up in the control group. This option requires specifying `aperiod()` or `fperiod()` and may not be combined with `lossprob()`, `lossprob1()`, `lossprob2()`, or `losshaz()`.

`losshaz2(`*numlist*`)` specifies an exponential hazard rate of losses to follow-up, η_2, in the experimental group. The default is `losshaz2(0)`, meaning no losses to follow-up in the experimental group. This option requires specifying `aperiod()` or `fperiod()` and may not be combined with `lossprob()`, `lossprob1()`, `lossprob2()`, or `losshaz()`.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Reporting

`show` and `show(`*showspec*`)` specify to display additional output containing the numbers of events, losses to follow-up, and event probabilities. If `show` is specified, group-specific numbers of events and, in the presence of losses to follow-up, group-specific numbers of losses to follow-up are displayed for the null and alternative hypotheses. With the table output, the numbers are displayed as additional columns.

showspec may contain any combination of `events`, `losses`, `eventprobs`, and `all`. `events` displays the group-specific numbers of events under the null and alternative hypotheses. `losses`, if present, displays group-specific numbers of losses under the null and alternative hypotheses. `eventprobs` displays group-specific event probabilities under the null and alternative hypotheses. `all` displays all the above.

The following option is available with `power exponential` but is not shown in the dialog box:

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `power exponential` command and the methodology for power and sample-size analysis for a two-sample comparison of exponential survivor functions. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis and [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests. See *Survival data* in [PSS-2] **Intro (power)** for an introduction to power and sample-size analysis for survival data.

Introduction

Let $S_1(t)$ and $S_2(t)$ be the exponential survivor functions with hazard rates λ_1 and λ_2 in the control and experimental groups, respectively. Define δ to be the treatment effect that can be expressed as a difference, $\psi = \lambda_2 - \lambda_1$, between hazard rates or as the log of the hazard ratio (a difference between log hazard-rates), $\ln(\Delta) = \ln(\lambda_2/\lambda_1) = \ln(\lambda_2) - \ln(\lambda_1)$. Negative values of the treatment effect δ imply the superiority of the experimental treatment over the standard (control) treatment. Denote r and T to be the length of the accrual period and the total duration of the study, respectively. Then, the follow-up period f is $f = T - r$.

Consider a study designed to compare the exponential survivor functions, $S_1(t) = e^{-\lambda_1 t}$ and $S_2(t) = e^{-\lambda_2 t}$, of the two treatment groups. The disparity in survivor functions may be tested using the hazards λ_1 and λ_2 for the exponential model. Depending on the definition of the treatment effect δ, two test statistics based on the difference and on the log ratio of the hazards may be used to conduct tests of the difference between survivor functions using respective null hypotheses, H_0: $\psi = 0$ and H_0: $\ln(\Delta) = 0$.

The basic formula for the sample-size and power calculations for the test of H_0: $\psi = 0$ is proposed by Lachin (1981). He also derives the equation relating the sample size and power allowing for uniform accrual of subjects into the study over the period from 0 to r. Lachin and Foulkes (1986) extend this formula to truncated exponential accrual over the interval 0 to r and exponential losses to follow-up over the interval 0 to T.

The simplest method for the sample-size and power calculations for the test of H_0: $\ln(\Delta) = 0$ is presented by George and Desu (1974). Rubinstein, Gail, and Santner (1981) extend their method to account for uniform accrual and exponential losses to follow-up and apply it to planning the duration of a survival study. The formula that relates the sample size and power for this test and takes into account the uniform accrual and exponential losses to follow-up is formulated by Lakatos and Lan (1992), based on the derivations of Rubinstein, Gail, and Santner (1981).

You can use `power exponential` to

- compute required number of events and sample size when you know power and effect size; or
- compute power when you know sample size (number of events) and effect size.

You can also supply effect size as hazard rates, survival probabilities, hazard ratio, or log hazard-ratio; adjust results for censoring; adjust results for uniform or exponential accrual; adjust results for group-specific exponentially distributed losses to follow-up; and compute results using the conditional or unconditional approach.

Using power exponential

`power exponential` computes sample size or power for a test comparing two exponential survivor functions. All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option. You can specify the `onesided` option to request a one-sided test. By default, all computations assume a balanced- or equal-allocation design; see [PSS-4] **Unbalanced designs** for a description of how to specify an unbalanced design.

To compute a total sample size, you specify an effect size and, optionally, the power of the test in the `power()` option. The default power is set to 0.8. By default, the computed sample size is rounded up. You can specify the `nfractional` option to see the corresponding fractional sample size; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. The `nfractional` option is allowed only for sample-size determination.

To compute power, you must specify the total sample size in the `n()` option and an effect size.

An effect size may be specified as a hazard ratio in option `hratio()`, as a log hazard-ratio in option `lnhratio()`, or as a difference between hazard rates in option `hdifference()`. By default, a hazard ratio of 0.5 is assumed. For a fixed-duration study, the control-group hazard rate h_1 or the control-group survival probability s_1 must also be specified. See *Alternative ways of specifying effect* below for details.

Instead of the total sample size `n()`, you can specify individual group sizes in `n1()` and `n2()` or specify one of the group sizes and `nratio()` when computing power or effect size. See *Two samples* in [PSS-4] **Unbalanced designs** for more details.

If the `time()` option is specified, the command's input parameters are the values of survival probabilities in the control (or the less favorable) group, $S_1(t)$, and in the experimental group, $S_2(t)$, at a fixed time, t (reference survival time), specified in `time()`, given as s_1 and s_2, respectively. Otherwise, the input parameters are assumed to be the values of the hazard rates in the control group, λ_1, and in the experimental group, λ_2, given as h_1 and h_2, respectively. If survival probabilities are specified, they are converted to hazard rates by using the formula for the exponential survivor function and the value of time t in `t()`.

By default, the estimates of sample sizes or power for the test of the difference between hazards are reported. This may be changed to the test versus the difference between log hazards by using the `loghazard` option. The default conditional approach may be replaced with the unconditional approach by using `unconditional`; see *The conditional versus unconditional approaches*.

If the duration of a study (T) in option `studytime()`, the length of a follow-up period (f) in option `fperiod()`, or the length of an accrual period (r) in option `aperiod()` is not specified, then the study is assumed to continue until all subjects experience an event (failure), regardless of how much time is required. If only `studytime()` is specified or only `fperiod()` is specified, the length of the accrual period is assumed to be zero and the follow-up period equals the duration of the study. If only `aperiod()` is specified, the length of the follow-up is assumed to be zero and the duration of the study equals the length of the accrual period (continuous accrual until the end of the study). If either `aperiod()` or `fperiod()` is specified with `studytime()`, the other one is computed using the relationship $T = r + f$. If both `aperiod()` or `fperiod()` are specified, a fixed-duration study of length $T = r + f$ is assumed.

If an accrual period of length r is specified in the `aperiod()` option, uniform accrual over the period $[0, r]$ is assumed. The accrual distribution may be changed to truncated exponential when the shape parameter is specified in `ashape()`. The combination of the `aprob()` and `aptime()` (or `atime()`) options may be used in place of the `ashape()` option to request the desired shape of the truncated exponential accrual. For examples, see *Nonuniform accrual*.

To take into account exponential losses to follow-up, the `losshaz()` or `lossprob()` and `losstime()` options may be used. Instead of specifying losses common to both groups, you can use options `losshaz1()` and `losshaz2()` or `lossprob1()` and `lossprob2()` to specify group-specific losses to follow-up. For examples, see *Exponential losses to follow-up*.

Alternative ways of specifying effect

`power exponential` provides several ways to specify the disparity between the control-group and experimental-group survivor functions for sample-size and power determinations. You can specify group hazard rates or group survival probabilities at a fixed time t directly. If survival probabilities are specified, they are converted to hazard rates by using the formula for the exponential survivor function and the value of time t. Alternatively, you can specify the control-group hazard rate or the control-group survival probability and an effect size expressed as a hazard ratio, a log hazard-ratio, or a difference between the two hazard rates. The corresponding experimental-group hazard rate will then be computed using the specified values of the control-group hazard rate and effect size.

By default, `power exponential` performs computation assuming a hazard ratio of 0.5. You can use the `hratio()` option to specify a different value for the hazard ratio or you can use the `lnhratio()` option to specify an effect size as a log hazard-ratio. If a control-group hazard rate or survival probability is specified, you can also specify an effect size as a difference between the experimental-group and control-group hazard rates in option `hdifference()`.

For a fixed-duration study when not all subjects experience an event by the end of the study, a control-group hazard rate or a control-group survival probability at time t must be specified in addition to an effect size.

You specify the control-group hazard rate h_1 following the command name. You can use any of the three options mentioned above to specify an effect size. The experimental-group hazard rate is then computed using the specified values of the control-group hazard rate and effect size.

`power exponential` h_1 [`, hratio()` | `lnhratio()` | `hdifference()` ...]

Alternatively, you can specify the experimental-group hazard rate h_2 directly.

`power exponential` h_1 h_2 [`, ...`]

Instead of the control-group hazard rate, you can specify the control-group survival probability s_1 at time t; the reference time t must be specified in option `time()`.

`power exponential` s_1 `, time(#)` [`hratio()` | `lnhratio()` | `hdifference()` ...]

Similarly to hazard rates, you can specify the experimental-group survival probability at time t instead of an effect size.

`power exponential` s_1 s_2 `, time(#)` [...]

The displayed effect size `delta` corresponds to the difference between hazard rates (or the hazard ratio if the control-group hazard is not specified) for the hazard-difference test and to the log hazard-ratio for the log hazard-ratio (or log hazard-difference) test when the `loghazard` option is specified. You can change this by specifying the `effect()` option: `effect(hratio)` reports the hazard ratio, `effect(lnhratio)` reports the log hazard-ratio, and `effect(hdifference)` reports the difference between the experimental-group and control-group hazard rates.

In the following sections, we describe the use of `power exponential` accompanied by examples for computing sample size and power.

Computing sample size

To compute sample size and number of events, you must specify an effect size and, optionally, the power of the test in the `power()` option. A default power of 0.8 is assumed if `power()` is not specified. A hazard ratio of 0.5 is assumed if an effect size is not specified. See *Alternative ways of specifying effect* for various ways of specifying an effect size.

Consider the following two types of survival studies: the first type, a type I study, is when investigators have enough resources to monitor the subjects until all of them experience an event (failure) and the second type, a type II study, is when the study terminates after a fixed period of time, regardless of whether all subjects experienced an event by that time.

Computing sample size in the absence of censoring

In this subsection we explore sample-size estimates using different approximations for a type I study. Examples of sample-size determination for a type II study are presented in the next subsection.

In survival studies, the requirement for the sample size is based on the requirement to observe a certain number of events (failures) to ensure a prespecified power of a test to detect a difference in survivor functions. For a type I study, the number of subjects required for the study is the same as the number of events required to be observed in the study because all subjects experience an event by the end of the study.

Example 1: Sample size using the Lachin method

Consider an example from Lachin (1981, 107). A clinical trial is to be conducted to compare the survivor functions in the control and the experimental groups with a one-sided exponential test, based on the difference between hazards, of the superiority of a new treatment (H_a: $\psi < 0$) for a disease with moderate levels of mortality. Subjects in the control group receive a standard treatment and subjects in the experimental group receive a new treatment. From previous studies the yearly hazard rate for the standard treatment was found to be $\lambda_1 = 0.3$, corresponding to 50% survival after 2.3 years. The investigators would like to know how many subjects are required to detect a reduction in hazard to $\lambda_2 = 0.2$ (H_a: $\psi = -0.1$), which corresponds to an increase in survival to 63% at 2.3 years, with 90% power, equal-sized groups, and a significance level, α, of 0.05.

To obtain the estimate of the sample size for the above study, we supply hazard rates 0.3 and 0.2 as arguments and specify the `power(0.9)` option for 90% power and the `onesided` option for a one-sided test.

```
. power exponential 0.3 0.2, power(0.9) onesided
note: input parameters are hazard rates.

Estimated sample sizes for two-sample comparison of survivor functions
Exponential test, hazard difference, conditional
H0: h2 = h1  versus  Ha: h2 < h1

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =   -0.1000  (hazard difference)

Survival information:

           h1 =    0.3000
           h2 =    0.2000

Estimated sample sizes:

            N =       218
  N per group =       109
```

From the output, a total of 218 events (subjects) must be observed (recruited) in a study to ensure a power of 90% of a one-sided exponential test to detect a 13% increase in survival probability of subjects in the experimental group with $\alpha = 0.05$. Our estimate of 218 of the total number of subjects (109 per group) required for the study is the same as the one reported in Lachin (1981, 107).

Example 2: Sample size using the George–Desu method

Example 1 reports the sample size obtained using the approximation of Lachin (1981) for the test based on the hazard difference. To obtain the sample size using the approximation of George and Desu (1974), for the equivalent alternative H_a: $\ln(\Delta) = -0.4055$ (a test based on the log of the hazard ratio), we need to specify the `loghazard` option.

```
. power exponential 0.3 0.2, power(0.9) onesided loghazard
note: input parameters are hazard rates.

Estimated sample sizes for two-sample comparison of survivor functions
Exponential test, log hazard-ratio, conditional
H0: ln(HR) = 0  versus  Ha: ln(HR) < 0

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =   -0.4055  (log hazard-ratio)

Survival information:

           h1 =    0.3000
           h2 =    0.2000

Estimated sample sizes:

            N =       210
  N per group =       105
```

The George–Desu method yields a slightly smaller estimate (210) of the total number of events (subjects). George and Desu (1974) studied the accuracy of the two approximations based on ψ and $\ln(\Delta)$ and concluded that the former is slightly conservative; that is, it gives slightly larger sample-size estimates. The latter was found to be accurate to one or two units of the exact solution for equal-sized groups.

◁

❑ Technical note

The approach from example 2 may also be used to obtain an approximation to the sample size or power for the exact F test of equality of two exponential mean analysis (life) times (using the relation between a mean and a hazard rate of the exponential distribution, $\mu = 1/\lambda$).

For example, the sample size of 210 obtained above may be used as an approximation to the number of subjects required in a study of which the goal is to detect an increase in a mean analysis (life) time of the experimental group from $3.33 = 1/0.3$ to $5 = 1/0.2$ by using the one-sided 5%-level F test with 90% power.

The test statistic of the F test is a ratio of two sample means from two exponential distributions that has an exact F distribution. The George–Desu method is based on the normal approximation of the distribution of the log of this test statistic. George and Desu (1974) studied this approximation for equal-sized groups and some common values of significance levels, powers, and hazard ratios and found it to be accurate to one or two units of the exact solution.

❑

▷ Example 3: Alternative ways of specifying effect

In *Alternative ways of specifying effect*, we described various ways in which the survival information of the groups can be supplied to `power exponential`. Here we demonstrate several examples.

In example 1, we specified the survival information by supplying the control-group and experimental-group hazard rates.

Instead of the experimental-group hazard rate, we can specify the difference between hazards in the `hdifference()` option and obtain identical results.

```
. power exponential 0.3, power(0.9) onesided hdifference(-0.1)
note: input parameters are hazard rates.

Estimated sample sizes for two-sample comparison of survivor functions
Exponential test, hazard difference, conditional
H0: h2 = h1  versus  Ha: h2 < h1

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =   -0.1000  (hazard difference)

Survival information:

           h1 =    0.3000
           h2 =    0.2000
      h2 - h1 =   -0.1000

Estimated sample sizes:

            N =       218
  N per group =       109
```

We can redisplay the effect size `delta` as a hazard ratio instead of the hazard difference:

```
. power exponential 0.3, power(0.9) onesided hdifference(-0.1) effect(hratio)
note: input parameters are hazard rates.

Estimated sample sizes for two-sample comparison of survivor functions
Exponential test, hazard difference, conditional
H0: h2 = h1  versus  Ha: h2 < h1

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =    0.6667  (hazard ratio)

Survival information:

           h1 =    0.3000
           h2 =    0.2000
      h2 - h1 =   -0.1000

Estimated sample sizes:

            N =       218
  N per group =       109
```

We can specify the hazard ratio of $0.2/0.3 = 0.66667$ in the `hratio()` option instead of `hdifference(-0.1)`.

```
. power exponential 0.3, power(0.9) onesided hratio(0.6667)
note: input parameters are hazard rates.

Estimated sample sizes for two-sample comparison of survivor functions
Exponential test, hazard difference, conditional
H0: h2 = h1  versus  Ha: h2 < h1

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =   -0.1000  (hazard difference)

Survival information:

           h1 =    0.3000
           h2 =    0.2000
       hratio =    0.6667

Estimated sample sizes:

            N =       218
  N per group =       109
```

We can obtain the same results from `power exponential` if we specify the control-group survival probability of 0.5 at time $t = 2.3$.

```
. power exponential 0.5, time(2.3) power(0.9) onesided hratio(0.6667)
note: input parameters are survival probabilities.

Estimated sample sizes for two-sample comparison of survivor functions
Exponential test, hazard difference, conditional
H0: h2 = h1  versus  Ha: h2 < h1

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =   -0.1004  (hazard difference)

Survival information:

           h1 =    0.3014          s1 =    0.5000
           h2 =    0.2009          s2 =    0.6299
       hratio =    0.6667           t =    2.3000

Estimated sample sizes:

            N =       218
  N per group =       109
```

We can also specify the experimental-group survival probability of 0.63 at time $t = 2.3$ directly instead of specifying the hazard ratio.

```
. power exponential 0.5 0.63, time(2.3) power(0.9) onesided
note: input parameters are survival probabilities.

Estimated sample sizes for two-sample comparison of survivor functions
Exponential test, hazard difference, conditional
H0: h2 = h1  versus  Ha: h2 < h1

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =   -0.1005  (hazard difference)

Survival information:

           h1 =    0.3014          s1 =    0.5000
           h2 =    0.2009          s2 =    0.6300
                                    t =    2.3000

Estimated sample sizes:

            N =       218
  N per group =       109
```

◁

▷ Example 4: Unbalanced design

By default, `power exponential` computes sample size for a balanced- or equal-allocation design. If we know the allocation ratio of subjects between the groups, we can compute the required sample size for an unbalanced design by specifying the `nratio()` option.

In example 1, we assumed the same numbers of subjects in the two groups. Suppose that we anticipate to recruit twice as many subjects in the experimental group, that is, $n_2/n_1 = 2$. We specify the `nratio(2)` option to compute the required sample size for the specified unbalanced design.

```
. power exponential 0.3 0.2, power(0.9) onesided nratio(2)
note: input parameters are hazard rates.

Estimated sample sizes for two-sample comparison of survivor functions
Exponential test, hazard difference, conditional
H0: h2 = h1  versus  Ha: h2 < h1

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =   -0.1000  (hazard difference)
        N2/N1 =    2.0000

Survival information:

           h1 =    0.3000
           h2 =    0.2000

Estimated sample sizes:

            N =       242
           N1 =        81
           N2 =       161
        N2/N1 =    1.9877
```

We need a total of 242 subjects—81 in the control group and 161 in the experimental group.

When different from the specified allocation rate, `power exponential` also displays the actual allocation rate corresponding to the reported rounded group sample sizes. If you wish, you can specify the `nfractional` option to see sample sizes without rounding; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for more information.

Also see *Two samples* in [PSS-4] **Unbalanced designs** for more examples of unbalanced designs for two-sample tests.

◁

Computing sample size in the presence of censoring

Often in practice, investigators may not have enough resources to continue a study until all subjects experience an event and, therefore, plan to terminate the study after a fixed period, T. Some subjects may not experience an event by the end of the study, in which case the (administrative) censoring of subjects occurs. In the presence of censoring, the number of subjects required in a study will be larger than the number of events required to be observed in the study.

We investigate how terminating the study after some fixed period, T, before all subjects experience an event affects the requirements for the sample size. The duration of a study is divided into two phases: an accrual phase of a length r, during which subjects are recruited to the study, and a follow-up phase of a length f, during which subjects are followed up until the end of the study and no new subjects enter the study. The duration of a study, T, is the sum of the lengths of the two phases.

Consider the following study designs. In the first study design, A, each subject is followed up for a length of time T. Here the minimum follow-up time f is equal to T, and, consequently, $r = 0$. In practice, however, subjects will often enter the study at random times and will be followed up until the end of a study at time T, in which case the subjects observed later will have a shorter follow-up than subjects who entered the study at the beginning. Therefore, the minimum follow-up time f will be less than T, and r will be equal to $T - f$. In this case the length of the accrual period, r, must be taken into account in the computations. In the presence of an accrual period, subjects may be recruited continuously during

a period of length T ($r = T$, $f = 0$) for the second study design, B. Or subjects may be recruited for a fixed period, r, and then followed up for a period of time, f, during which no new subjects enter the trial, so that the total duration of study is $T = r + f$ (the third design, C).

▷ Example 5: Sample size in the presence of accrual and follow-up periods

Continuing with example 1, assume that the investigators have resources to continue the study for only 5 years, $T = 5$. We specify the duration of the study in the `studytime()` option, and we tabulate sample-size values for different lengths of an accrual period specified as a list (see **[U] 11.1.8 numlist**) in `aperiod()`. For simplicity, we use the `table()` option to obtain a table containing only columns `power`, `N`, `aperiod`, `fperiod`, `h1`, `h2`, and `alpha`.

```
. power exponential 0.3 0.2, power(0.9) onesided aperiod(0(1)5) studytime(5)
> table(power N aperiod fperiod h1 h2 alpha)
note: input parameters are hazard rates.

Estimated sample sizes for two-sample comparison of survivor functions
Exponential test, hazard difference, conditional
H0: h2 = h1  versus  Ha: h2 < h1
```

power	N	aperiod	fperiod	h1	h2	alpha
.9	304	0	5	.3	.2	.05
.9	322	1	4	.3	.2	.05
.9	344	2	3	.3	.2	.05
.9	378	3	2	.3	.2	.05
.9	426	4	1	.3	.2	.05
.9	502	5	0	.3	.2	.05

```
Note: Uniform accrual; 50% accrued by 50% of accrual period.
```

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

The first and the last entries of the above table correspond to the extreme cases of no accrual (design A) and no follow-up (design B), respectively. When `aperiod()` is specified, a uniform accrual is assumed that implies, for example, that 50% of the subjects will be recruited once 50% of the accrual period has elapsed.

For design A, the estimate of the sample size, 304, is larger than the earlier estimate of 218 from example 1. That is, if the study in example 1 terminates after 5 years, the requirement for the sample size increases by 39% to ensure that the same number of 218 events is observed.

By trying different values of the follow-up period, we may find that a 30-year follow-up is required if the investigators can recruit no more than 218 subjects: 30 years is required to observe an event for all subjects in this study.

```
. power exponential 0.3 0.2, power(0.9) onesided fperiod(30)
note: input parameters are hazard rates.

Estimated sample sizes for two-sample comparison of survivor functions
Exponential test, hazard difference, conditional
H0: h2 = h1  versus  Ha: h2 < h1

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =   -0.1000  (hazard difference)

Accrual and follow-up information:

     duration =   30.0000
    follow-up =   30.0000

Survival information:

           h1 =    0.3000
           h2 =    0.2000

Estimated sample sizes:

            N =       218
  N per group =       109
```

Returning to our table, for design B, instead of being monitored for 5 years, subjects are continuously recruited throughout those 5 years; the total sample size increases from 304 to 502. The reason for such an increase is that the average analysis time (the time when a subject is at risk of a failure) decreases from 5 to 2.5 and, therefore, reduces the probability of a subject failing by the end of the study.

In general, the estimates of the total sample size steadily increase as the length of the follow-up decreases. That is, the presence of a follow-up period reduces the requirement for the number of subjects in the study. For example, a clinical trial with a 3-year uniform accrual and a 2-year follow-up needs a total of 378 subjects (189 per group) compared with the total of 502 subjects required for a study with no follow-up and a 5-year accrual.

◁

▷ Example 6: Uniform accrual

In example 5, we investigated the effect of the length of accrual on sample size for a type II study when not all subjects experience an event by the end of the study. We specified the length of the accrual period in option `aperiod()` and the duration of the study in option `studytime()`. When `aperiod()` is specified, the accrual distribution is assumed to be uniform, that is, 10% of the subjects are expected to be recruited once 10% of the accrual period has elapsed, 25% of subjects are expected to be recruited once 25% of the accrual period has elapsed, 50% of subjects are expected to be recruited once 50% of the accrual period has elapsed, and so on. Let's compute the sample size for a study with a 3-year uniform accrual and a 2-year follow-up. We use options `aperiod()` and `fperiod()` to specify the accrual and follow-up periods, respectively.

```
. power exponential 0.3 0.2, power(0.9) onesided aperiod(3) fperiod(2)
note: input parameters are hazard rates.

Estimated sample sizes for two-sample comparison of survivor functions
Exponential test, hazard difference, conditional
H0: h2 = h1  versus  Ha: h2 < h1

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =   -0.1000  (hazard difference)

Accrual and follow-up information:

     duration =    5.0000
    follow-up =    2.0000
      accrual =    3.0000  (uniform)

Survival information:

           h1 =    0.3000
           h2 =    0.2000

Estimated sample sizes:

            N =       378
  N per group =       189
```

The required total sample size is 378 with 189 subjects per group. This is the same sample size we obtained in the table from example 5 with the corresponding values of the accrual and follow-up periods. ◁

Nonuniform accrual

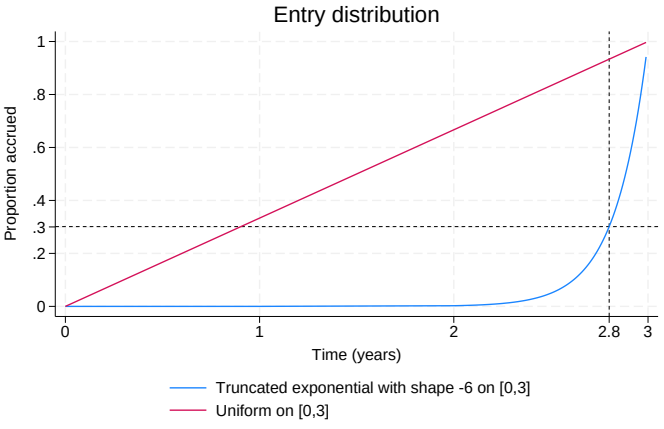
In the presence of an accrual period, `power exponential` performs computations assuming uniform accrual over the period of time r, specified in `aperiod()`. The assumption of uniform accrual may be relaxed by requesting a truncated exponential accrual over the interval 0 to r with shape γ as specified in `ashape(#)`. If an estimate of γ is unavailable, the proportion of subjects expected to be recruited, $G(t^*)$, may be specified in `aprob()` along with either the fixed time by which the subjects were recruited, t^*, in option `atime()` or the elapsed proportion of the accrual period, t^*/r, in option `aptime()`. This information is used to find the corresponding γ by using

$$G(t^*) = \{1 - \exp(-\gamma t^*)\}/\{1 - \exp(-\gamma r)\}$$

Also see Cleves, Gould, and Marchenko (2016, sec. 16.2) for more information, and see *Methods and formulas* for technical details.

▷ Example 7: Truncated exponential entry distribution

Continuing with example 6, we investigate the influence of nonuniform accrual on the estimate of the sample size for a study with a 3-year accrual and a 2-year follow-up. Suppose that the recruitment of subjects to the study is slow for most of the accrual period and increases rapidly toward the end of the recruitment. Consider an extreme case of such an accrual corresponding to shape parameter -6. The graph of a uniform entry distribution and an exponential entry distribution with shape -6 truncated over $[0, 3]$ is given below.



From the above graph, the accrual of subjects is extremely slow during most of the recruitment period, with 70% of subjects being recruited within the last few months of a 3-year accrual period. Stated another way, according to the graph, only 30% of subjects are expected to be recruited during the first 2.8 years.

To obtain the estimate of the sample size for this study, we type

```
. power exponential 0.3 0.2, power(0.9) onesided aperiod(3) fperiod(2) ashape(-6)
note: input parameters are hazard rates.

Estimated sample sizes for two-sample comparison of survivor functions
Exponential test, hazard difference, conditional
H0: h2 = h1  versus  Ha: h2 < h1

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =   -0.1000  (hazard difference)

Accrual and follow-up information:

     duration =    5.0000
    follow-up =    2.0000
      accrual =    3.0000  (exponential)
   accrual(%) =     50.00  (by time t*)
          t* =    2.8845  (96.15% of accrual)

Survival information:

           h1 =    0.3000
           h2 =    0.2000

Estimated sample sizes:

            N =       516
  N per group =       258
```

and conclude that 516 subjects have to be recruited to this study. This sample size ensures 90% power of a one-sided, 5%-level test to detect a reduction in hazard from 0.3 to 0.2 when the accrual of subjects follows the considered truncated exponential distribution. For this extreme case of a negative truncated exponential entry distribution (the concave entry distribution), the estimate of the sample, 516, increases substantially compared with an estimate of 378 from example 6, which assumes a uniform entry distribution. On the other hand, a truncated exponential distribution with positive values of the shape parameter (convex entry distribution) will reduce the requirement for the sample size when compared with uniform accrual.

Suppose that we do not know (or do not wish to guess) the value of the shape parameter. The only information available to us from the above graph is that 30% of the subjects are expected to be recruited in the first 2.8 years. We submit this information in the `aprob()` and `atime()` options, as shown below, and obtain the same estimate of 516 for sample size.

```
. power exponential 0.3 0.2, power(0.9) onesided aperiod(3) fperiod(2)
> aprob(0.3) atime(2.8)
note: input parameters are hazard rates.

Estimated sample sizes for two-sample comparison of survivor functions
Exponential test, hazard difference, conditional
H0: h2 = h1  versus  Ha: h2 < h1

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =   -0.1000  (hazard difference)

Accrual and follow-up information:

     duration =    5.0000
    follow-up =    2.0000
      accrual =    3.0000  (exponential)
   accrual(%) =     30.00  (by time t*)
           t* =    2.8000  (93.33% of accrual)

Survival information:

           h1 =    0.3000
           h2 =    0.2000

Estimated sample sizes:

            N =       516
  N per group =       258
```

Another way we can supply the information about accrual is by specifying a percentage of subjects expected to be recruited by a certain percentage of the accrual period. For example, and equivalent to the above specification, 30% of subjects are expected to be recruited after 93.33% of the accrual period has elapsed. We submit this information in the `aprob()` and `aptime()` options, and we again obtain the same estimate of 516 for sample size.

```
. power exponential 0.3 0.2, power(0.9) onesided aperiod(3) fperiod(2)
> aprob(0.3) aptime(0.9333)
note: input parameters are hazard rates.

Estimated sample sizes for two-sample comparison of survivor functions
Exponential test, hazard difference, conditional
H0: h2 = h1  versus  Ha: h2 < h1

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =   -0.1000  (hazard difference)

Accrual and follow-up information:

     duration =    5.0000
    follow-up =    2.0000
      accrual =    3.0000  (exponential)
   accrual(%) =     30.00  (by time t*)
          t* =    2.7999  (93.33% of accrual)

Survival information:

           h1 =    0.3000
           h2 =    0.2000

Estimated sample sizes:

            N =       516
  N per group =       258
```

◁

Exponential losses to follow-up

Apart from administrative censoring, subjects may not experience an event by the end of the study because of being lost to follow-up for various reasons. See *Survival data* in [PSS-2] **Intro (power)** and [PSS-5] **Glossary** for a more detailed description. Rubinstein, Gail, and Santner (1981) and Lachin and Foulkes (1986) extend sample-size and power computations to take into account exponentially distributed losses to follow-up. In addition to being exponentially distributed, losses to follow-up are assumed to be independent of the survival times.

▷ Example 8: Exponential losses to follow-up

Suppose that in example 6, in the study with a 3-year uniform accrual and a 2-year follow-up, yearly loss hazards in the control and the experimental groups are 0.2. A loss hazard rate common to both groups can be specified in option `losshaz()`.

```
. power exponential 0.3 0.2, power(0.9) onesided aperiod(3) fperiod(2)
> losshaz(0.2) show
note: input parameters are hazard rates.

Estimated sample sizes for two-sample comparison of survivor functions
Exponential test, hazard difference, conditional
H0: h2 = h1  versus  Ha: h2 < h1

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =   -0.1000  (hazard difference)

Accrual and follow-up information:

     duration =    5.0000
    follow-up =    2.0000
      accrual =    3.0000  (uniform)

Survival information:

           h1 =    0.3000
           h2 =    0.2000

Loss-to-follow-up information:

          lh1 =    0.2000
          lh2 =    0.2000

Estimated expected number of events:

         E|Ha =       213      E|H0 =       216
        E1|Ha =       121     E1|H0 =       108
        E2|Ha =        92     E2|H0 =       108

Estimated expected number of losses to follow-up:

         L|Ha =       173      L|H0 =       172
        L1|Ha =        81     L1|H0 =        86
        L2|Ha =        92     L2|H0 =        86

Estimated sample sizes:

            N =       500
  N per group =       250
```

The sample size required for a one-sided, 5%-level test to detect a reduction in hazard from 0.3 to 0.2 with 90% power increases from 378 (see example 6) to 500. We observe that for the extreme case of losses to follow-up, sample size increases significantly. A conservative adjustment commonly applied in practice is $n(1 + p_L)$, where p_L is the expected proportion of losses to follow-up in both groups combined. For this example, p_L may be computed as $0.5(0.369 + 0.324) \approx 0.35$ from table 2 of Lachin and Foulkes (1986). Then the conservative estimate of the sample size is $378(1 + 0.35) = 510$, which is slightly greater than 500, the actual required sample size.

We also requested that additional information about the expected number of events and losses to follow-up under the null and under the alternative hypothesis be displayed by using the `show` option. From the above output, a total of 173 subjects (81 from the control group and 92 from the experimental group) are expected to be lost in the study with exponentially distributed losses with yearly rates of 0.2 in each group under the alternative hypothesis.

If the proportion of subjects lost to follow-up by a fixed period in each group is available, it can be supplied by using the `lossprob()` and `losstime()` options rather than loss to follow-up rates. For example, in the above study approximately 33%, $1 - \exp(-0.2 \times 2) \approx 0.33$, of subjects in each group are lost at time 2 (years). We can obtain the same estimates of sample sizes by typing

```
. power exponential 0.3 0.2, power(0.9) onesided aperiod(3) fperiod(2)
> lossprob(0.33) losstime(2)
```
(*output omitted*)

◁

The conditional versus unconditional approaches

Denote δ to be the effect size, and denote $\hat{\lambda}_1$ and $\hat{\lambda}_2$ to be the maximum likelihood estimates of the respective hazard-rate parameters. Consider the two effect-size estimators based on the difference between the hazard rates, $\hat{\lambda}_2 - \hat{\lambda}_1$, and based on the log of the hazard ratio, $\ln(\hat{\lambda}_2/\hat{\lambda}_1)$. Both estimators are asymptotically normal under the null and under the alternative hypothesis.

We adopt Chow et al. (2018, 156) terminology when referring to the conditional and unconditional tests. The conditional test is the test that uses the constraint $\lambda_2 = \lambda_1$ (conditional on H_0) when computing the variance of the effect-size estimator under the null. The unconditional test is the test that does not use the above constraint when computing the variance of the effect-size estimator under the null. The score and the Wald tests are each one of the examples of conditional and unconditional tests, respectively. Chow et al. (2018) note that neither of the two tests (conditional or unconditional) is always more powerful than the other under the alternative hypothesis. Therefore, there is no definite recommendation of which one is preferable in practice.

The conditional approach relies on the following relationship between sample size and power, given in Lachin (1981), to compute estimates of required sample size or power,

$$|\delta| = z_{1-\alpha}\left\{\text{Var}(\delta, H_0)\right\}^{1/2} + z_{1-\beta}\left\{\text{Var}(\delta, H_a)\right\}^{1/2}$$

where $z_{1-\alpha}$ and $z_{1-\beta}$ are the $(1-\alpha)$th and the $(1-\beta)$th quantiles of the standard normal distribution, and $\text{Var}(\delta, H_0)$ and $\text{Var}(\delta, H_a)$ are the asymptotic variances under the null and under the alternative, respectively, of the effect-size estimator, $\hat{\delta}$. This approach uses the variance of the estimator conditional on the hypothesis type.

The unconditional approach replaces $\text{Var}(\delta, H_0)$ with $\text{Var}(\delta, H_a)$ in the above and uses the variance under the alternative to compute the estimates of sample size and power:

$$|\delta| = (z_{1-\alpha} + z_{1-\beta})\left\{\text{Var}(\delta, H_a)\right\}^{1/2}$$

Therefore, the resulting formulas based on the two approaches are different.

Lakatos and Lan (1992) formulate the sample-size formula for the log hazard-ratio test based on the method of Rubinstein, Gail, and Santner (1981). This formula is based on the unconditional approach. Lachin and Foulkes (1986) provide the sample-size formula for the test of the log of the hazard ratio that uses the conditional approach. They also present both conditional and unconditional versions of formulas for the test based on the difference between hazards. As noted by Lachin and Foulkes (1986), sample sizes estimated based on the unconditional approach will be larger than the estimates based on the conditional approach for equal-sized groups.

Both approaches are available with `power exponential`; the conditional is the default and the unconditional may be requested by specifying the `unconditional` option. Refer to *Methods and formulas* for the formulas underlying these approaches.

▷ Example 9: Sample size using the Rubinstein–Gail–Santner method

Consider the following scenario in Lakatos and Lan (1992, table I). A 10-year survival study with a 1-year accrual period and a 9-year follow-up is conducted to compare the survivor functions of the two groups by using a two-sided, 0.05 exponential test based on the log of the hazard ratio. The probability of surviving to the end of a study for subjects in the control group is 0.8 [$S_1(t) = 0.8$, $t = 10$]. Subjects are recruited uniformly over the interval $[0, 1]$. Lakatos and Lan (1992) report an estimate of 664 for the sample size required to detect a change in the hazard of the experimental group corresponding to the hazard ratio $\Delta = 0.5$ with 90% power by using the Rubinstein–Gail–Santner (1981) method. To obtain the estimates according to this method, we need to specify both `loghazard` and `unconditional`.

```
. power exponential 0.8, t(10) power(0.9) aperiod(1) fperiod(9) loghazard
> unconditional
note: input parameters are survival probabilities.

Estimated sample sizes for two-sample comparison of survivor functions
Exponential test, log hazard-ratio, unconditional
H0: ln(HR) = 0  versus  Ha: ln(HR) != 0

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =   -0.6931  (log hazard-ratio)

Accrual and follow-up information:

     duration =   10.0000
    follow-up =    9.0000
      accrual =    1.0000  (uniform)

Survival information:

           h1 =    0.0223          s1 =    0.8000
           h2 =    0.0112          s2 =    0.8944
       hratio =    0.5000           t =   10.0000

Estimated sample sizes:

            N =       664
  N per group =       332
```

Because the default value of the hazard ratio is 0.5, we omit the `hratio(0.5)` option in the above. From the output, we obtain the same estimate of 664 of the sample size as reported in Lakatos and Lan (1992).

◁

In the absence of censoring, the estimates of the sample size or power based on the test of log of the hazard ratio are the same for the conditional and the unconditional approaches. For example, both

```
. power exponential 0.8, t(10) power(0.9) loghazard
  (output omitted)
```

and

```
. power exponential 0.8, t(10) power(0.9) loghazard unconditional
  (output omitted)
```

produce the same estimate of the sample size (88). The asymptotic variance of maximum likelihood estimates of the log of the hazard ratio does not depend on hazard rates when there is no censoring and, therefore, does not depend on the type of hypothesis, $\text{Var}(\hat{\delta}, H_0) = \text{Var}(\hat{\delta}, H_a) = 2/N$.

Link to the sample-size and power computation for the log-rank test

▷ Example 10: Sample size using the Freedman and the Schoenfeld methods

Continuing with examples 1 and 2, Lachin (1981, 106) gives another approximation to obtain the estimate of the sample size under the equal-group allocation. This approximation coincides with the formula derived by Freedman (1982) for the number of events in the context of the log-rank test. We can obtain such an estimate by using `power logrank` and by specifying the hazard ratio of 0.66667 computed earlier.

```
. power logrank, hratio(0.66667) power(0.9) onesided

Estimated sample sizes for two-sample comparison of survivor functions
Log-rank test, Freedman method
H0: HR = 1  versus  Ha: HR < 1

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =    0.6667  (hazard ratio)
       hratio =    0.6667

Censoring:

         Pr_E =    1.0000

Estimated number of events and sample sizes:

            E =       216
            N =       216
  N per group =       108
```

The estimate, 216, of the sample size is the same as given in Lachin (1981, 107) and is slightly smaller than the estimate, 218, obtained in example 1 and larger than the estimate, 210, obtained using the George–Desu method in example 2.

The approximation due to George and Desu (1974) is the same as the approximation to the number of events derived by Schoenfeld (1981) in application to the log-rank test. We can confirm that by typing

```
. power logrank, hratio(0.66667) power(0.9) onesided schoenfeld

Estimated sample sizes for two-sample comparison of survivor functions
Log-rank test, Schoenfeld method
H0: ln(HR) = 0  versus  Ha: ln(HR) < 0

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =   -0.4055  (log hazard-ratio)
       hratio =    0.6667

Censoring:

         Pr_E =    1.0000

Estimated number of events and sample sizes:

            E =       210
            N =       210
  N per group =       105
```

We obtain the same estimate of 210 as when using `power exponential` with the `loghazard` option in example 2.

◁

Computing power

Sometimes the number of subjects available for enrollment into the study is limited. In such cases, the researchers may want to investigate with what power they can detect a desired treatment effect for a given sample size.

To compute power, you must specify the sample size in the `n()` option and an effect size. A hazard ratio of 0.5 is assumed if an effect size is not specified. Also see *Alternative ways of specifying effect* for various ways of specifying an effect size.

▷ Example 11: Power determination

We verify the power computation for the study from example 9. We expect the power estimate to be close to 0.9.

The only thing we change in the `power exponential` command from example 9 is replacing the `power(0.9)` option with the `n(664)` option.

```
. power exponential 0.8, t(10) n(664) aperiod(1) fperiod(9)
> loghazard unconditional
note: input parameters are survival probabilities.

Estimated power for two-sample comparison of survivor functions
Exponential test, log hazard-ratio, unconditional
H0: ln(HR) = 0  versus  Ha: ln(HR) != 0

Study parameters:

        alpha =    0.0500
            N =       664
  N per group =       332
        delta =   -0.6931  (log hazard-ratio)

Accrual and follow-up information:

     duration =   10.0000
    follow-up =    9.0000
      accrual =    1.0000  (uniform)

Survival information:

           h1 =    0.0223           s1 =    0.8000
           h2 =    0.0112           s2 =    0.8944
       hratio =    0.5000            t =   10.0000

Estimated power:

        power =    0.9000
```

We obtain the estimate of power 0.9.

◁

Testing hypotheses about two exponential survivor functions

▷ Example 12: Using streg to perform the log hazard-ratio test

In this example, we demonstrate the importance of sample-size computations to ensure a high power of a test to detect a difference between exponential survivor functions. We consider an asymptotic Wald (or normal z) test to test whether the log of the hazard ratio is zero.

Continuing with example 11, suppose that the investigators have only 100 subjects available for the study. As we see below, the power to detect a 50% risk reduction in a hazard of the experimental group (the hazard ratio of 0.5) decreases from 90% to 24%:

```
. power exponential 0.8, t(10) n(100) aperiod(1) fperiod(9)
> loghazard unconditional
note: input parameters are survival probabilities.

Estimated power for two-sample comparison of survivor functions
Exponential test, log hazard-ratio, unconditional
H0: ln(HR) = 0  versus  Ha: ln(HR) != 0

Study parameters:

        alpha =    0.0500
            N =       100
  N per group =        50
        delta =   -0.6931  (log hazard-ratio)

Accrual and follow-up information:

     duration =   10.0000
    follow-up =    9.0000
      accrual =    1.0000  (uniform)

Survival information:

           h1 =    0.0223           s1 =    0.8000
           h2 =    0.0112           s2 =    0.8944
       hratio =    0.5000            t =   10.0000

Estimated power:

        power =    0.2414
```

To demonstrate the implication of this reduction, consider the following example. We generate the data according to the study from example 11 with the following code:

```
program simdata
        args n h1 h2 r
        set obs `n'
        generate double entry = `r'*runiform()
        generate double u = runiform()
        /* random allocation to two groups of equal sizes */
        generate double u1 = runiform()
        generate double u2 = runiform()
        sort u1 u2, stable
        generate byte drug = (_n<=`n'/2)
        /* exponential failure times with rates h1 and h2 */
        generate double failtime = entry - ln(1-u)/`h1' if drug==0
        replace failtime = entry - ln(1-u)/`h2' if drug==1
end

. clear
. set seed 234
. quietly simdata 100 0.0223 0.0112 1
```

The entry times of subjects are generated from a uniform $[0, 1)$ distribution and stored in variable `entry`. The subjects are randomized to two groups of equal size of 50 subjects each. The survival times are generated from exponential distribution with the hazard rate of $-\ln(0.8)/10 = 0.0223$ in the control group, `drug` $= 0$, and the hazard rate of $0.5 \times 0.0223 = 0.0112$ in the experimental group, `drug` $= 1$, conditional on subjects' entry times in `entry`.

Before analyzing these survival data, we need to set up the data properly using `stset`. The failure-time variable is `failtime`. The study terminates at $t = 10$, so we use `exit(time 10)` with `stset` to specify that all failure times past 10 are to be treated as censored. Because subjects enter the study at random times (`entry`) and become at risk of a failure upon entering the study, we also specify the `origin(entry)` option to ensure that the analysis time is adjusted for the entry times. For more details, see [ST] **stset**.

```
. stset failtime, exit(time 10) origin(entry)
Survival-time data settings
         Failure event: (assumed to fail at time=failtime)
Observed time interval: (origin, failtime]
     Exit on or before: time 10
     Time for analysis: (time-origin)
               Origin: time entry
------------------------------------------------------------------------------
        100  total observations
          0  exclusions
------------------------------------------------------------------------------
        100  observations remaining, representing
          7  failures in single-record/single-failure data
    921.825  total analysis time at risk and under observation
                                            At risk from t =         0
                                 Earliest observed entry t =         0
                                      Last observed exit t =  9.990494
```

To perform the log hazard-ratio test, we fit an exponential regression model on `drug` by using `streg` (see [ST] **streg**). We can express the log of the hazard ratio in terms of regression coefficients as follows: $\ln(\Delta) = \ln(\lambda_2/\lambda_1) = \ln\{\exp(\beta_0 + \beta_1)/\exp(\beta_0)\} = \beta_1$, where β_0 and β_1 are the estimated coefficients for the constant and `drug` in the regression model. Then the test of H_0: $\ln(\Delta) = 0$ may be rewritten in terms of a coefficient on `drug` as H_0: $\beta_1 = 0$. This test is part of the standard output after `streg`.

```
. streg drug, distribution(exponential) nohr
        Failure _d: 1 (meaning all fail)
  Analysis time _t: (failtime-origin)
            Origin: time entry
  Exit on or before: time 10

Iteration 0:  Log likelihood = -29.718762
Iteration 1:  Log likelihood = -29.049311
Iteration 2:  Log likelihood = -29.014323
Iteration 3:  Log likelihood = -29.014222
Iteration 4:  Log likelihood = -29.014222

Exponential PH regression

No. of subjects =      100                  Number of obs =    100
No. of failures =        7
Time at risk    = 921.8249
                                            LR chi2(1)    =   1.41
Log likelihood = -29.014222                 Prob > chi2   = 0.2352

------------------------------------------------------------------------------
          _t | Coefficient  Std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
        drug |   .9428118     .83666     1.13   0.260    -.6970117    2.582635
       _cons |  -5.453234   .7071068    -7.71   0.000    -6.839137    -4.06733
------------------------------------------------------------------------------
```

From the output table above, the p-value for a two-sided test of the coefficient for `drug`, 0.260, is greater than 0.05. On that basis, we do not have evidence to reject the null hypothesis of no difference between the two exponential survivor functions. Therefore, we make an incorrect decision because we simulated the data with different group hazard rates. If we were to repeat this, say, 100 times, using different datasets simulated according to the alternative H_a: $\ln(\Delta) = \ln(0.5) = -0.6931$ (see [R] **simulate**), for roughly 76 of them we would have failed to reject the null hypothesis of no difference (a type II error). Therefore, more subjects are required to be able to detect the log of the hazard ratio of -0.4055 in this study.

◁

▷ Example 13: Using results from streg to perform the Wald test of hazard difference

We obtain the power of the test based on the difference between hazards for the study in example 12 (omit the `loghazard` option from the syntax of `power exponential`).

```
. power exponential 0.8, t(10) n(100) aperiod(1) fperiod(9) unconditional
note: input parameters are survival probabilities.

Estimated power for two-sample comparison of survivor functions
Exponential test, hazard difference, unconditional
H0: h2 = h1  versus  Ha: h2 != h1

Study parameters:

        alpha =    0.0500
            N =       100
  N per group =        50
        delta =   -0.0112  (hazard difference)

Accrual and follow-up information:

     duration =   10.0000
    follow-up =    9.0000
      accrual =    1.0000  (uniform)

Survival information:

           h1 =    0.0223           s1 =    0.8000
           h2 =    0.0112           s2 =    0.8944
       hratio =    0.5000            t =   10.0000

Estimated power:

        power =    0.2458
```

We obtain a power estimate of 0.2458, which is close to 0.2414 from example 12.

To test the difference between hazard rates by using the Wald test, we express this difference in terms of coefficients, $\lambda_2 - \lambda_1 = \exp(\beta_0)\{\exp(\beta_1) - 1\}$, and we use `testnl` ([R] **testnl**) after `streg` to perform the nonlinear hypothesis test of H_0: $\exp(\beta_0)\{\exp(\beta_1) - 1\} = 0$.

```
. testnl exp(_b[_cons])*(exp(_b[drug])-1) = 0
  (1)  exp(_b[_cons])*(exp(_b[drug])-1) = 0

               chi2(1) =        1.35
           Prob > chi2 =        0.2451
```

We obtain the same conclusions from the Wald test based on the difference between hazards as in example 12. That is, based on the p-value of 0.2451, we fail to reject the null hypothesis of no difference between hazards of two groups (or miss the alternative H_a: $\psi = -0.0112$ corresponding to reduction in hazard from roughly 0.02 to 0.01) for the data from example 12.

◁

Often in practice, to test the disparity in two exponential survivor functions, the log-rank test is used instead of the hazard-difference test. Also the Wald (or the score) test from the Cox model is used instead of the exponential log hazard-ratio test. Refer to [ST] **sts test** and [ST] **stcox** for examples on how to perform these tests (also see [PSS-2] **power logrank** and [PSS-2] **power cox**).

Sometimes the estimates of sample size and power obtained under the assumption of the exponential model are used as an approximation to the results used in a more general context of the log-rank test or the Cox proportional hazards model. Refer to Lachin (2011, 483–484) for the rationale behind this. Also see Lakatos and Lan (1992) for a discussion of the circumstances under which sample-size estimates obtained assuming the exponential model may be inaccurate when used with more general proportional hazards models.

Stored results

power exponential stores the following in r():

Scalars

r(alpha)	significance level
r(power)	power
r(beta)	probability of a type II error
r(delta)	effect size
r(N)	total sample size
r(N_a)	actual sample size
r(N1)	sample size of the control group
r(N2)	sample size of the experimental group
r(nratio)	ratio of sample sizes, N2/N1
r(nratio_a)	actual ratio of sample sizes
r(nfractional)	1 if nfractional is specified, 0 otherwise
r(onesided)	1 for a one-sided test, 0 otherwise
r(hratio)	hazard ratio
r(lnhratio)	log hazard-ratio
r(hdiff)	difference between hazard rates
r(h1)	hazard in the control group (if specified)
r(h2)	hazard in the experimental group
r(s1)	survival probability in the control group (if specified)
r(s2)	survival probability in the experimental group (if specified)
r(time)	reference survival time (if time() is specified)
r(aperiod)	length of the accrual period (if specified)
r(fperiod)	length of the follow-up period (if specified)
r(studytime)	duration of the study (if specified)
r(ashape)	shape parameter (if aperiod() is specified)
r(aprob)	shape parameter (if aprob() is specified)
r(aptime)	proportion of accrual period (if aptime() is specified)
r(atime)	reference accrual time (if atime() is specified)
r(losshaz)	loss hazard rate in both groups (if specified)
r(losshaz1)	loss hazard in the control group (if specified)
r(losshaz2)	loss hazard in the experimental group (if specified)
r(lossprob)	proportions of subjects lost to follow-up in both groups (if lossprob() is specified)
r(losstime)	reference loss to follow-up time (if losstime() is specified)
r(unconditional)	1 if unconditional is specified, 0 otherwise
r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise

Macros

r(type)	test
r(method)	exponential
r(test)	hazard difference or log-hazard difference
r(accrual)	uniform or exponential
r(effect)	hratio, lnhratio, hdifference, or lnhdifference
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results
r(Pr_vec)	1×4 matrix of probabilities of an event (when computed)
r(Ea_vec)	1×3 matrix of expected number of events under the alternative (when computed)
r(E0_vec)	1×3 matrix of expected number of events under the null (when computed)
r(La_vec)	1×3 matrix of expected number of losses under the alternative (when computed)
r(L0_vec)	1×3 matrix of expected number of losses under the null (when computed)

Methods and formulas

By default, `power exponential` computes the sample size required to achieve a specified power to detect a difference between hazard rates, $\psi_a = \lambda_{2a} - \lambda_{1a}$, using the method of Lachin (1981). If `loghazard` is specified, the sample size required to detect a log of the hazard ratio $\ln(\Delta_a) = \ln(\lambda_{2a}/\lambda_{1a})$ with specified power is reported using the formula derived by George and Desu (1974). In the presence of an accrual period, the methods of Lachin and Foulkes (1986) or (for uniform accrual only) Rubinstein, Gail, and Santner (1981) (if `loghazard` and `unconditional` are specified) are used.

In addition to the notation given in *Introduction*, denote n, n_1, and n_2 to be the total number of subjects required for the study, the number of subjects in the control group, and the number of subjects in the experimental group, respectively. Let $R = n_2/n_1$ denote the ratio of sample sizes of the experimental group to the control group. Let $p_1 = n_1/n = 1/(1+R)$ and $p_2 = n_2/n = 1 - p_1 = R/(1+R)$ be the proportions of subjects allocated to the control and the experimental groups; γ be the shape parameter of the truncated exponential distribution with p.d.f. $g(z) = \gamma\exp(-\gamma z)/\{1 - \exp(-\gamma r)\},\ 0 \le z \le r,\ \gamma \ne 0$; η_1 and η_2 be the loss hazards in the control and the experimental groups; and $z_{(1-\alpha/k)}$ and $z_{(1-\beta)}$ be the $(1-\alpha/k)$th and the $(1-\beta)$th quantiles of the standard normal distribution, with $k = 1$ for the one-sided test and $k = 2$ for the two-sided test. Denote $\overline{\lambda} = p_1\lambda_1 + p_2\lambda_2$. Recall that the difference between hazards is denoted by $\psi = \lambda_2 - \lambda_1$ and the hazard ratio is denoted by $\Delta = \lambda_2/\lambda_1$.

If survival probabilities $S_1(t)$ and $S_2(t)$ at a fixed time t are specified rather than hazard rates, the hazard rates are computed as $\lambda_i = -\ln\{S_i(t)\}/t,\ i = 1, 2$. If loss to follow-up probabilities $L_1(t_L)$ and $L_2(t_L)$ at a fixed time t_L are given instead of loss to follow-up hazard rates, the loss hazard rates are computed as $\eta_i = -\ln\{1 - L_i(t_L)\}/t_L,\ i = 1, 2$.

All formulas below are derived under the assumption of exponential survival distributions with hazard rates in the control and the experimental groups λ_1 and λ_2, respectively, and rely on large-sample properties of the maximum likelihood estimates of λ_1 and λ_2.

Denote $\xi_o = \zeta(\overline{\lambda}, \gamma, \eta_1)p_1^{-1} + \zeta(\overline{\lambda}, \gamma, \eta_2)p_2^{-1}$ and $\xi_a = \zeta(\lambda_1, \gamma, \eta_1)p_1^{-1} + \zeta(\lambda_2, \gamma, \eta_2)p_2^{-1}$.

The formula for the sample-size calculation using the conditional approach is

$$n = \frac{\left(z_{1-\alpha/k}\xi_o^{1/2} + z_{1-\beta}\xi_a^{1/2}\right)^2}{\delta^2}$$

and using the unconditional approach is

$$n = \frac{(z_{1-\alpha/k} + z_{1-\beta})^2\xi_a}{\delta^2}$$

where $\zeta(\lambda, \gamma, \eta) = \lambda^2/p_E$ if $\delta = \psi$, $\zeta(\lambda, \gamma, \eta) = 1/p_E$ if $\delta = \ln(\Delta)$, and p_E is to be defined later. λ and η denote a failure hazard rate and a loss to follow-up hazard rate.

In the absence of censoring, the overall probability of an event (failure), p_E, is set to 1. Here the resulting formula for the sample size for the log hazard-ratio test depends only on the ratio of hazards and not on the individual group hazard rates. The resulting sample size formula for the test of the difference may also be rewritten as a function of the ratio of hazards only. Therefore, under no censoring, for a fixed value of the hazard ratio $\Delta = \lambda_2/\lambda_1$, the estimates of the sample size (or power) will be constant with respect to varying hazard rates λ_1 and λ_2.

In the presence of censoring, when each subject is followed up for a fixed period $f = T$,

$$p_E = p_E(\lambda, \eta) = \frac{\lambda}{\lambda + \eta}[1 - \exp\{-(\lambda + \eta)T\}]$$

In the presence of an accrual period, the probability of an event is defined as

$$p_E = p_E(\lambda, \eta) = \frac{\lambda}{(\lambda + \eta)}\left[1 - \frac{\exp\{-(\lambda + \eta)(T - r)\} - \exp\{-(\lambda + \eta)T\}}{(\lambda + \eta)r}\right]$$

or

$$p_E = p_E(\lambda, \gamma, \eta) = \frac{\lambda}{(\lambda + \eta)}\left(1 + \frac{\gamma\exp\{-(\lambda + \eta)T\}[1 - \exp\{(\lambda + \eta - \gamma)r\}]}{(\lambda + \eta - \gamma)\{1 - \exp(-\gamma r)\}}\right)$$

under uniform or truncated exponential accrual with shape γ over $[0, r]$, respectively. Uniform accrual is assumed for $|\gamma| < 10^{-6}$.

The formulas are obtained from Lachin (1981), Lachin and Foulkes (1986), and Lakatos and Lan (1992). To avoid division by 0 in the case $\lambda + \eta = \gamma$, the probability of an event is taken to be the limit of the above expression, $p_E = \lim_{\lambda+\eta->\gamma} p_E(\lambda, \gamma, \eta)$.

The number of subjects required to be recruited in each group is obtained as $n_1 = n/(1 + R)$ and $n_2 = nR/(1 + R)$. If `nfractional` is not specified, sample sizes are rounded to integer values; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for details.

The expected number of events and losses to follow-up are computed as suggested by Lachin and Foulkes (1986). Under the null hypothesis,

$$E_{H_0} = n_1 p_E(\overline{\lambda}, \gamma, \eta_1) + n_2 p_E(\overline{\lambda}, \gamma, \eta_2)$$
$$L_{H_0} = n_1(\eta_1/\overline{\lambda}) p_E(\overline{\lambda}, \gamma, \eta_1) + n_2(\eta_2/\overline{\lambda}) p_E(\overline{\lambda}, \gamma, \eta_2)$$

and under the alternative hypothesis,

$$E_{H_a} = n_1 p_E(\lambda_1, \gamma, \eta_1) + n_2 p_E(\lambda_2, \gamma, \eta_2)$$
$$L_{H_a} = n_1(\eta_1/\lambda_1) p_E(\lambda_1, \gamma, \eta_1) + n_2(\eta_2/\lambda_2) p_E(\lambda_2, \gamma, \eta_2)$$

For unconditional tests, the expected number of events and losses to follow-up under the null is computed by setting $\overline{\lambda} = \lambda_1$. The estimates of the expected number of events and losses to follow-up in each group are rounded to the nearest integer.

To obtain the estimate of the power, $1 - \beta$, the formulas for the sample size are solved for $z_{(1-\beta)}$ and the normal cumulative distribution function is used to obtain the corresponding probability $1 - \beta$.

To obtain the unknown shape parameter, γ, of a truncated exponential entry distribution, an iterative procedure is used to solve the equation

$$p_a = G(t_a) = \frac{1 - \exp(-\gamma t_a)}{1 - \exp(-\gamma r)}$$

for a given proportion of subjects p_a recruited at a given time, t_a, for $t_a \in [0, r]$.

References

Chow, S.-C., J. Shao, H. Wang, and Y. Lokhnygina. 2018. *Sample Size Calculations in Clinical Research*. 3rd ed. Boca Raton, FL: CRC Press.

Cleves, M. A., W. W. Gould, and Y. V. Marchenko. 2016. *An Introduction to Survival Analysis Using Stata*. Rev. 3rd ed. College Station, TX: Stata Press.

Freedman, L. S. 1982. Tables of the number of patients required in clinical trials using the logrank test. *Statistics in Medicine* 1: 121–129. https://doi.org/10.1002/sim.4780010204.

George, S. L., and M. M. Desu. 1974. Planning the size and duration of a clinical trial studying the time to some critical event. *Journal of Chronic Diseases* 27: 15–24. https://doi.org/10.1016/0021-9681(74)90004-6.

Lachin, J. M. 1981. Introduction to sample size determination and power analysis for clinical trials. *Controlled Clinical Trials* 2: 93–113. https://doi.org/10.1016/0197-2456(81)90001-5.

———. 2011. *Biostatistical Methods: The Assessment of Relative Risks*. 2nd ed. Hoboken, NJ: Wiley.

Lachin, J. M., and M. A. Foulkes. 1986. Evaluation of sample size and power for analyses of survival with allowance for nonuniform patient entry, losses to follow-up, noncompliance, and stratification. *Biometrics* 42: 507–519. https://doi.org/10.2307/2531201.

Lakatos, E., and K. K. G. Lan. 1992. A comparison of sample size methods for the logrank statistic. *Statistics in Medicine* 11: 179–191. https://doi.org/10.1002/sim.4780110205.

Rubinstein, L. V., M. H. Gail, and T. J. Santner. 1981. Planning the duration of a comparative clinical trial with loss to follow-up and a period of continued observation. *Journal of Chronic Diseases* 34: 469–479. https://doi.org/10.1016/0021-9681(81)90007-2.

Schoenfeld, D. A. 1981. The asymptotic properties of nonparametric tests for comparing survival distributions. *Biometrika* 68: 316–319. https://doi.org/10.2307/2335833.

Also see [PSS-2] **Intro (power)** for more references.

Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power cox** — Power analysis for the Cox proportional hazards model

[PSS-2] **power logrank** — Power analysis for the log-rank test

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[R] **test** — Test linear hypotheses after estimation

[ST] **streg** — Parametric survival models

power logrank — Power analysis for the log-rank test

Description

`power logrank` computes sample size, power, or effect size for survival analysis comparing survivor functions in two groups by using the log-rank test. The results can be obtained using the Freedman or Schoenfeld approaches. Effect size can be expressed as a hazard ratio or as a log hazard-ratio. The command supports unbalanced designs, and provides options to account for administrative censoring, uniform accrual, and withdrawal of subjects from the study. For power and sample-size analysis in a cluster randomized design, see [PSS-2] **power logrank, cluster**.

Quick start

Sample size for the log-rank test of $H_0\colon \Delta = 0$ versus $H_a\colon \Delta \neq 0$ using the Freedman method for alternative hazard ratio $\Delta_a = 0.76$ without censoring and with default power of 0.8 and significance level $\alpha = 0.05$

```
power logrank, hratio(.76)
```

Same as above, but use Schoenfeld's method

```
power logrank, hratio(.76) schoenfeld
```

Sample size for censored design with survival probabilities $surv_1 = 0.3$ and $surv_2 = 0.4$

```
power logrank .3 .4
```

Same as above, specified as $surv_1 = 0.3$ and hazard ratio of 0.76

```
power logrank .3, hratio(.76)
```

Same as above, but for hazard ratios of 0.65, 0.7, 0.75, and 0.8

```
power logrank .3, hratio(.65(.05).8)
```

Same as above, but show results in a graph of hazard ratio versus sample size

```
power logrank .3, hratio(.65(.05).8) graph
```

Sample size for a one-sided test with $\alpha = 0.01$

```
power logrank .3, hratio(.76) onesided alpha(.01)
```

Sample size adjusted for 10% withdrawal from the study

```
power logrank .3, hratio(.76) wdprob(.1)
```

Power for a design with censoring and a sample size of 300

```
power logrank .3 .4, n(300)
```

Same as above, but specify twice as many observations in the experimental group

```
power logrank .3 .4, n(300) nratio(2)
```

Effect size for a design without censoring, sample size of 300, power of 0.8, and default $\alpha = 0.05$

```
power logrank, n(300) power(.8)
```

Same as above, but for a censored design with control-group survival probability of 0.3

```
power logrank .3, n(300) power(.8)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
power logrank [surv1 [surv2]] [, power(numlist) options]
```

Compute power

```
power logrank [surv1 [surv2]], n(numlist) [options]
```

Compute effect size

```
power logrank [surv1], n(numlist) power(numlist) [options]
```

where $surv_1$ is the survival probability in the control (reference) group at the end of the study t^* and $surv_2$ is the survival probability in the experimental (comparison) group at the end of the study t^*. $surv_1$ and $surv_2$ may each be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

options	Description
Main	
* alpha(*numlist*)	significance level; default is alpha(0.05)
* power(*numlist*)	power; default is power(0.8)
* beta(*numlist*)	probability of type II error; default is beta(0.2)
* n(*numlist*)	total sample size; required to compute power or effect size
* n1(*numlist*)	sample size of the control group
* n2(*numlist*)	sample size of the experimental group
* nratio(*numlist*)	ratio of sample sizes, N2/N1; default is nratio(1), meaning equal group sizes
nfractional	allow fractional sample sizes
* hratio(*numlist*)	hazard ratio of the experimental to the control group; default is hratio(0.5)
* lnhratio(*numlist*)	log hazard-ratio of the experimental to the control group
schoenfeld	use the formula based on the log hazard-ratio in calculations; default is to use the formula based on the hazard ratio
effect(*effect*)	specify the type of effect to display; default is method-specific
direction(lower|upper)	direction of the effect for effect-size determination; default is direction(lower), which means that the postulated value of the parameter is smaller than the hypothesized value
onesided	one-sided test; default is two sided
parallel	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Censoring	
simpson(# # # | *matname*)	survival probabilities in the control group at three specific time points to compute the probability of an event (failure), using Simpson's rule under uniform accrual
st1(*varname*$_s$ *varname*$_t$)	variables *varname*$_s$, containing survival probabilities in the control group, and *varname*$_t$, containing respective time points, to compute the probability of an event (failure), using numerical integration under uniform accrual
* wdprob(*numlist*)	proportion of subjects anticipated to withdraw from the study; default is wdprob(0)
Table	
[no]table[(*tablespec*)]	suppress table or display results as a table; see [PSS-2] **power, table**
saving(*filename* [, replace])	save the table data to *filename*; use replace to overwrite existing *filename*
Graph	
graph[(*graphopts*)]	graph results; see [PSS-2] **power, graph**

Iteration

`init(#)`	initial value for effect size
`iterate(#)`	maximum number of iterations; default is `iterate(500)`
`tolerance(#)`	parameter tolerance; default is `tolerance(1e-12)`
`ftolerance(#)`	function tolerance; default is `ftolerance(1e-12)`
`[no]log`	suppress or display iteration log
`[no]dots`	suppress or display iterations as dots
`cluster`	perform computations for a CRD; see [PSS-2] **power logrank, cluster**
`notitle`	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the `parallel` option.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`cluster` and `notitle` do not appear in the dialog box.

effect	Description
`hratio`	hazard ratio
`lnhratio`	log hazard-ratio

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`N`	total number of subjects	N
`N1`	number of subjects in the control group	N_1
`N2`	number of subjects in the experimental group	N_2
`nratio`	ratio of sample sizes, experimental to control	N_2/N_1
`delta`	effect size	δ
`E`	total number of events (failures)	E
`hratio`	hazard ratio	Δ
`lnhratio`	log hazard-ratio	$\ln(\Delta)$
`s1`	survival probability in the control group	$S_1(T)$
`s2`	survival probability in the experimental group	$S_2(T)$
`Pr_E`	overall probability of an event (failure)	p_E
`Pr_w`	probability of withdrawals	p_w
`target`	target parameter; `hratio` or `lnhratio`	
`_all`	display all supported columns	

Column beta is shown in the default table in place of column power if specified.

Column lnhratio is shown in the default table in place of column hratio if specified.

Columns s1 and s2 are available only when specified.

Columns nratio and Pr_w are shown in the default table if specified.

Options

Main

alpha(), power(), beta(), n(), n1(), n2(), nratio(), nfractional; see [PSS-2] **power**.

hratio(*numlist*) specifies the hazard ratio (effect size) of the experimental group to the control group. The default is hratio(0.5). This value typically defines the clinically significant improvement of the experimental procedure over the control procedure desired to be detected by the log-rank test with a certain power.

You can specify an effect size either as a hazard ratio in hratio() or as a log hazard-ratio in lnhratio(). The default is hratio(0.5). If both arguments $surv_1$ and $surv_2$ are specified, hratio() is not allowed and the hazard ratio is instead computed as $\ln(surv_2)/\ln(surv_1)$.

This option is not allowed with the effect-size determination and may not be combined with lnhratio().

lnhratio(*numlist*) specifies the log hazard-ratio (effect size) of the experimental group to the control group. This value typically defines the clinically significant improvement of the experimental procedure over the control procedure desired to be detected by the log-rank test with a certain power.

You can specify an effect size either as a hazard ratio in hratio() or as a log hazard-ratio in lnhratio(). The default is hratio(0.5). If both arguments $surv_1$ and $surv_2$ are specified, lnhratio() is not allowed and the log hazard-ratio is computed as $\ln\{\ln(surv_2)/\ln(surv_1)\}$.

This option is not allowed with the effect-size determination and may not be combined with hratio().

schoenfeld requests calculations using the formula based on the log hazard-ratio, according to Schoenfeld (1981). The default is to use the formula based on the hazard ratio, according to Freedman (1982).

effect(*effect*) specifies the type of the effect size to be reported in the output as delta. *effect* is one of hratio or lnhratio. By default, the effect size delta is a hazard ratio, effect(hratio), for a hazard-ratio test and a log hazard-ratio, effect(lnhratio), for a log hazard-ratio test (when schoenfeld is specified).

direction(), onesided, parallel; see [PSS-2] **power**. direction(lower) is the default.

Censoring

simpson(# # # | *matname*) specifies survival probabilities in the control group at three specific time points to compute the probability of an event (failure) using Simpson's rule under the assumption of uniform accrual. Either the actual values or a 1×3 matrix, *matname*, containing these values can be specified. By default, the probability of an event is approximated as an average of the failure probabilities $1-s_1$ and $1-s_2$; see *Methods and formulas*. simpson() may not be combined with st1() and may not be used if command argument $surv_1$ or $surv_2$ is specified. This option is not allowed with effect-size computation.

`st1(`*varname*$_s$ *varname*$_t$`)` specifies variables *varname*$_s$, containing survival probabilities in the control group, and *varname*$_t$, containing respective time points, to compute the probability of an event (failure) using numerical integration under the assumption of uniform accrual; see [R] **dydx**. The minimum and the maximum values of *varname*$_t$ must be the length of the follow-up period and the duration of the study, respectively. By default, the probability of an event is approximated as an average of the failure probabilities $1-s_1$ and $1-s_2$; see *Methods and formulas*. `st1()` may not be combined with `simpson()` and may not be used if command argument *surv*$_1$ or *surv*$_2$ is specified. This option is not allowed with effect-size computation.

`wdprob(`*numlist*`)` specifies the proportion of subjects anticipated to withdraw from the study. The default is `wdprob(0)`. `wdprob()` is allowed only with sample-size computation.

Table

`table`, `table()`, `notable`; see [PSS-2] **power, table**.

`saving()`; see [PSS-2] **power**.

Graph

`graph`, `graph()`; see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies an initial value for the estimated hazard ratio or, if `schoenfeld` is specified, for the estimated log hazard-ratio during the effect-size determination.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following options are available with `power logrank` but are not shown in the dialog box:

`cluster`; see [PSS-2] **power logrank, cluster**.

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `power logrank` command and the methodology for power and sample-size analysis for a two-sample comparison of survivor functions using the log-rank test. See [PSS-2] **Intro (power)** for a general introduction to power and sample-size analysis and [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests. See *Survival data* in [PSS-2] **Intro (power)** for an introduction to power and sample-size analysis for survival data. For power and sample-size analysis in a cluster randomized design, see [PSS-2] **power logrank, cluster**.

Introduction

Consider a survival study comparing the survivor functions in two groups using the log-rank test. Let $S_1(t)$ and $S_2(t)$ denote the survivor functions of the control and the experimental groups, respectively. The key assumption of the log-rank test is that the hazard functions are proportional. That is, $h_2(t) = \Delta h_1(t)$ for any t or, equivalently, $S_2(t) = \{S_1(t)\}^{\Delta}$, where Δ is the hazard ratio. If $\Delta < 1$, the survival in the experimental group is higher relative to the survival in the control group; the new treatment is superior to the standard treatment. If $\Delta > 1$, then the standard treatment is superior to the new treatment. Under the proportional-hazards assumption, the test of the equality of the two survivor functions H_0: $S_1(t) = S_2(t)$ versus H_a: $S_1(t) \neq S_2(t)$ is equivalent to the test H_0: $\Delta = 1$ versus H_a: $\Delta \neq 1$ or H_0: $\ln(\Delta) = 0$ versus H_a: $\ln(\Delta) \neq 0$.

The methods implemented in `power logrank` for power and sample-size analysis relate the power of the log-rank test directly to the number of events observed in the study. Depending on whether censoring occurs in a study, the required number of subjects is either equal to the number of events or is computed using the estimates of the number of events and the combined probability of an event (failure). Thus, in the presence of censoring, in addition to the number of events, the probability of a subject not being censored (failing) needs to be estimated to obtain the final estimate of the required number of subjects in the study.

To determine the required number of events, the investigator must specify the size or significance level, α, and the clinically significant difference between the two treatments (effect size) to be detected by the log-rank test, H_a: $\Delta = \Delta_a$, with prespecified power $1 - \beta$. The effect size, a difference between the two treatments, is usually expressed as a hazard ratio, Δ_a, using the `hratio()` option. Alternatively, you may specify an effect size as a log hazard-ratio, $\ln(\Delta_a)$, in the `lnhratio()` option.

When all subjects fail by the end of the study (no censoring), a type I study, the information above is sufficient to obtain the number of subjects required in the study. Often, in practice, not all subjects fail by the end of the study, in which case censoring of subjects occurs (a type II study). Here the estimates of the survival probabilities in the control and experimental groups are necessary to estimate an overall probability of an event and, then, the required sample size.

`power logrank` supports two methods, those of Freedman (1982) and Schoenfeld (1981), to obtain the estimates of the number of events or power (see also Marubini and Valsecchi [1995, 127, 134] and Collett [2015, 473, 479]). The final estimates of the sample size are based on the approximation of the probability of an event due to Freedman (1982), the default, or in the presence of uniform accrual, due to Schoenfeld (1983) (see also Collett 2015).

You can use `power logrank` to

- compute required number of events and sample size when you know power and effect size (expressed as a hazard ratio or log hazard-ratio);
- compute power when you know sample size (number of events) and effect size (expressed as a hazard ratio or log hazard-ratio); or
- compute effect size (hazard ratio or log hazard-ratio) and experimental-group survival when you know sample size (number of events) and power.

You can also choose between the Freedman or Schoenfeld computational approaches, adjust results for administrative censoring, adjust results for uniform accrual of subjects to the study, and adjust results for withdrawal of subjects from the study.

Using power logrank

power logrank computes sample size, power, or effect size for the log-rank test comparing the survivor functions in two groups. All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the alpha() option. You can specify the onesided option to request a one-sided test. By default, all computations assume a balanced- or equal-allocation design; see [PSS-4] **Unbalanced designs** for a description of how to specify an unbalanced design.

To compute a total sample size, you specify an effect size and optionally power of the test in the power() option. The default power is set to 0.8. By default, the computed sample size is rounded up. You can specify the nfractional option to see the corresponding fractional sample size; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. The nfractional option is allowed only for sample-size determination.

To compute power, you must specify the total sample size in the n() option and an effect size.

An effect size may be specified either as a hazard ratio supplied in the hratio() option or as a log hazard-ratio supplied in the lnhratio() option. If neither is specified, a hazard ratio of 0.5 is assumed.

To compute an effect size, which may be expressed either as a hazard ratio or as a log hazard-ratio, you must specify the total sample size in the n() option; the power in the power() option; and, optionally, the direction of the effect. The direction is lower by default, direction(lower), which means that the target hazard ratio is assumed to be less than one or that target log hazard-ratio is negative. In other words, the experimental treatment is presumed to be an improvement over the control treatment. If you want to change the direction to upper, corresponding to the target hazard ratio being greater than one, use direction(upper).

Instead of the total sample size n(), you can specify individual group sizes in n1() and n2(), or specify one of the group sizes and nratio() when computing power or effect size. Also see *Two samples* in [PSS-4] **Unbalanced designs** for more details.

As we mentioned earlier, the effect size for power logrank may be expressed as a hazard ratio or as a log hazard-ratio. By default, the effect size, which is labeled as delta in the output, corresponds to the hazard ratio for the Freedman method and to the log hazard-ratio for the Schoenfeld method. You can change this by specifying the effect() option: effect(hratio) (the default) reports the hazard ratio and effect(lnhratio) reports the log hazard-ratio.

By default, all computations assume no censoring. In the presence of administrative censoring, you must specify a survival probability at the end of the study in the control group as the first command argument. You can also specify a survival probability at the end of the study in the experimental group as the second command argument. Otherwise, it will be computed using the specified hazard ratio or log hazard-ratio and the control-group survival probability. To accommodate an accrual period under the assumption of uniform accrual, survival information may instead be supplied in option simpson() or in option st1(); see *Including information about subject accrual* for details.

When computing sample size, you can adjust for withdrawal of subjects from the study by specifying the anticipated proportion of withdrawals in the wdprob() option.

By default, power logrank performs computations for a hazard-ratio test. Use the schoenfeld option to request computations for a log-hazard-ratio test.

In the presence of censoring, effect-size determination requires iteration. The default initial value of the estimated hazard ratio or, if `schoenfeld` is specified, of log hazard-ratio is obtained based on the formula assuming no censoring. This value may be changed by specifying the `init()` option. See [PSS-2] **power** for the descriptions of other options that control the iteration procedure.

In the following sections, we describe the use of `power logrank` accompanied by examples for computing sample size, power, and effect size.

Computing sample size

To compute sample size and number of events, you must specify an effect size (a hazard ratio or a log hazard-ratio) and, optionally, the power of the test in the `power()` option. A default power of 0.8 is assumed if `power()` is not specified. A hazard ratio of 0.5 is assumed if an effect size is not specified.

Computing sample size in the absence of censoring

We demonstrate several examples of how to use `power logrank` to obtain the estimates of sample size and number of events using Freedman (1982) and Schoenfeld (1981) methods for uncensored data (a type I study when no censoring of subjects occurs).

▷ Example 1: Number of events (failures) using Freedman method

Consider a survival study to be conducted to compare the survivor function of subjects receiving a treatment (the experimental group) to the survivor function of those receiving a placebo or no treatment (the control group) using the log-rank test. Suppose that the study continues until all subjects fail (no censoring). The investigator wants to know how many events need to be observed in the study to achieve a power of 80% of a two-sided log-rank test with $\alpha = 0.05$ to detect a 50% reduction in the hazard of the experimental group ($\Delta_a = 0.5$). Because the default settings of `power logrank` are `power(0.8)`, `alpha(0.05)`, and `hratio(0.5)`, to obtain the estimate of the required number of events for the above study using the Freedman method (the default), we simply type

```
. power logrank

Estimated sample sizes for two-sample comparison of survivor functions
Log-rank test, Freedman method
H0: HR = 1  versus  Ha: HR != 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.5000  (hazard ratio)
       hratio =    0.5000

Censoring:

         Pr_E =    1.0000

Estimated number of events and sample sizes:

            E =        72
            N =        72
  N per group =        36
```

From the output, a total of 72 events (failures) must be observed to achieve the required power of 80%. Because all subjects experience an event by the end of the study (`Pr_E=1.0000`), the number of subjects required to be recruited to the study is equal to the number of events. That is, the investigator needs to recruit a total of 72 subjects (36 per group) to the study.

◁

▷ Example 2: Number of events (failures) using Schoenfeld method

Following example 1, we can request the Schoenfeld method by specifying the `schoenfeld` option.

```
. power logrank, schoenfeld

Estimated sample sizes for two-sample comparison of survivor functions
Log-rank test, Schoenfeld method
H0: ln(HR) = 0  versus  Ha: ln(HR) != 0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =   -0.6931  (log hazard-ratio)
       hratio =    0.5000

Censoring:

         Pr_E =    1.0000

Estimated number of events and sample sizes:

            E =        66
            N =        66
  N per group =        33
```

We obtain a slightly smaller estimate (66) of the total number of events and subjects.

◁

❑ Technical note

Freedman (1982) and Schoenfeld (1981) derive the formulas for the number of events based on the asymptotic distribution of the log-rank test statistic. Freedman (1982) uses the asymptotic mean and variance of the log-rank test statistic expressed as a function of the true hazard ratio, Δ, whereas Schoenfeld (1981) (see also Collett [2015, 474]) bases the derivation on the asymptotic mean of the log-rank test statistic as a function of the true log hazard-ratio, $\ln(\Delta)$. We label the corresponding approaches as "Freedman method" and "Schoenfeld method" in the output.

For values of the hazard ratio close to one, the two methods tend to give similar results. In general, the Freedman method gives higher estimates than the Schoenfeld method. The performance of the Freedman method was studied by Lakatos and Lan (1992) and was found to slightly overestimate the sample size under the assumption of proportional hazards. Hsieh (1992) investigated the performance of the two methods under unequal allocation and concluded that Freedman's formula predicts the highest power for the log-rank test when the sample-size ratio of the two groups equals the reciprocal of the hazard ratio. Schoenfeld's formula predicts highest powers when sample sizes in the two groups are equal.

❑

▷ Example 3: Unbalanced design

By default, `power logrank` computes sample size for a balanced- or equal-allocation design. If we know the allocation ratio of subjects between the groups, we can compute the required sample size and number of events for an unbalanced design by specifying the `nratio()` option.

Continuing with example 1, we anticipate being able to recruit twice as many subjects in the experimental group; that is, $n_2/n_1 = 2$. We specify the `nratio(2)` option to compute the required sample size for the specified unbalanced design.

```
. power logrank, nratio(2)
Estimated sample sizes for two-sample comparison of survivor functions
Log-rank test, Freedman method
H0: HR = 1  versus  Ha: HR != 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.5000  (hazard ratio)
       hratio =    0.5000
        N2/N1 =    2.0000

Censoring:

         Pr_E =    1.0000

Estimated number of events and sample sizes:

            E =        63
            N =        63
           N1 =        21
           N2 =        42
```

We need a total of 63 subjects—21 in the control group and 42 in the experimental group.

Also see *Two samples* in [PSS-4] **Unbalanced designs** for more examples of unbalanced designs for two-sample tests.

◁

Computing sample size in the presence of censoring

Because of constraints on costs and time, it is often infeasible to continue the study until all subjects experience an event. Instead, the study terminates at some prespecified point in time. As a result, some subjects may not experience an event by the end of the study; that is, administrative censoring of subjects occurs. This increases the requirement on the number of subjects in the study to ensure that a certain number of events is observed.

In the presence of censoring (for a type II study), Freedman (1982) assumes the following. The analysis occurs at a fixed time t^* after the last patient was accrued, and all information about subject follow-up beyond time t^* is excluded. To minimize an overestimation of the sample size because of neglecting this information, the author suggests choosing t^* as the minimum follow-up time, f, beyond which the frequency of occurrence of events is low (the time at which, say, 85% of the total events expected are observed). Under this assumption, the number of required subjects does not depend on the rates of accrual and occurrence of events but only on the proportions of patients in the two treatment groups, s_1 and s_2, surviving after f. See *Including information about subject accrual* about how to compute the sample size in the presence of a long accrual.

If censoring of subjects occurs, the probability of a subject not being censored needs to be estimated to obtain an accurate estimate of the required sample size. The assumption above justifies a simple procedure, suggested by Freedman (1982) and used by default by `power logrank`, to compute this probability using the estimates of survival probabilities at the end of the study in the control and the experimental groups. Therefore, for a type II study (under administrative censoring), these probabilities must be supplied to `power logrank`.

▷ Example 4: Sample size in the presence of censoring using Freedman method

Consider an example from Machin et al. (2009, 91) of a study of patients with resectable colon cancer. The goal of the study was to compare the efficacy of the drug levamisole against a placebo with respect to relapse-free survival, using a one-sided log-rank test with a significance level of 5%. The investigators anticipated a 10% increase (from 50% to 60%, with a respective hazard ratio of 0.737) in the survival of the experimental group with respect to the survival of the control (placebo) group at the end of the study. They wanted to detect this increase with a power of 80%. To obtain the required sample size, we enter the survival probabilities 0.5 and 0.6 as arguments and specify the `onesided` option to request a one-sided test.

```
. power logrank 0.5 0.6, onesided

Estimated sample sizes for two-sample comparison of survivor functions
Log-rank test, Freedman method
H0: HR = 1  versus  Ha: HR < 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.7370  (hazard ratio)
       hratio =    0.7370

Censoring:

           s1 =    0.5000
           s2 =    0.6000
         Pr_E =    0.4500

Estimated number of events and sample sizes:

            E =       270
            N =       600
  N per group =       300
```

From the above output, the investigators would have to observe a total of 270 events (relapses) to detect a 26% decrease in the hazard ($\Delta_a = 0.737$) of the experimental group relative to the hazard of the control group with a power of 80% using a one-sided log-rank test with $\alpha = 0.05$. They would have to recruit a total of 600 patients (300 per group) to observe that many events.

In contrast, in the absence of censoring, only 270 subjects would have been required to detect a decrease in hazard corresponding to $\Delta_a = 0.737$:

```
. power logrank, hratio(0.737) onesided

Estimated sample sizes for two-sample comparison of survivor functions
Log-rank test, Freedman method
H0: HR = 1  versus  Ha: HR < 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.7370  (hazard ratio)
       hratio =    0.7370

Censoring:

         Pr_E =    1.0000

Estimated number of events and sample sizes:

            E =       270
            N =       270
  N per group =       135
```

◁

▷ Example 5: Sample size in the presence of censoring using Schoenfeld method

If we wanted to use the Schoenfeld method to calculate sample size for the study described in example 4, we could type

```
. power logrank 0.5 0.6, onesided schoenfeld

Estimated sample sizes for two-sample comparison of survivor functions
Log-rank test, Schoenfeld method
H0: ln(HR) = 0  versus  Ha: ln(HR) < 0

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =   -0.3052  (log hazard-ratio)
       hratio =    0.7370

Censoring:

           s1 =    0.5000
           s2 =    0.6000
         Pr_E =    0.4500

Estimated number of events and sample sizes:

            E =       266
            N =       590
  N per group =       295
```

We find that 590 subjects are required in the study to observe a total of 266 events to ensure a power of 80%.

See the technical note in *Computing sample size in the absence of censoring* for a brief comparison of the Freedman and Schoenfeld methods.

◁

▷ Example 6: Alternative ways of specifying effect

If we wish, we can redefine effect size `delta` in example 4 to be a log hazard-ratio by specifying the `effect()` option.

```
. power logrank 0.5 0.6, onesided effect(lnhratio)

Estimated sample sizes for two-sample comparison of survivor functions
Log-rank test, Freedman method
H0: HR = 1  versus  Ha: HR < 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =   -0.3052  (log hazard-ratio)
       hratio =    0.7370

Censoring:

           s1 =    0.5000
           s2 =    0.6000
         Pr_E =    0.4500

Estimated number of events and sample sizes:

            E =       270
            N =       600
  N per group =       300
```

The effect size `delta` now contains the value of the log hazard-ratio.

Continuing with example 4, instead of the estimate of the survival probability in the experimental group, we may have an estimate of the hazard ratio Δ_a. For example, the estimate of the hazard ratio in this example is 0.737. We can specify the value of the hazard ratio in the `hratio()` option instead of specifying the experimental-group survival probability 0.6.

```
. power logrank 0.5, onesided hratio(0.737)

Estimated sample sizes for two-sample comparison of survivor functions
Log-rank test, Freedman method
H0: HR = 1  versus  Ha: HR < 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.7370  (hazard ratio)
       hratio =    0.7370

Censoring:

           s1 =    0.5000
           s2 =    0.6000
         Pr_E =    0.4500

Estimated number of events and sample sizes:

            E =       270
            N =       600
  N per group =       300
```

Alternatively, instead of the hazard ratio we can specify the log hazard-ratio in option `lnhratio()`.

```
. power logrank 0.5, onesided lnhratio(-0.3052)

Estimated sample sizes for two-sample comparison of survivor functions
Log-rank test, Freedman method
H0: HR = 1  versus  Ha: HR < 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.7370  (hazard ratio)
   ln(hratio) =   -0.3052

Censoring:

           s1 =    0.5000
           s2 =    0.6000
         Pr_E =    0.4500

Estimated number of events and sample sizes:

            E =       270
            N =       600
  N per group =       300
```

The results are identical to the prior results. The estimate of the log hazard-ratio is now displayed in the output instead of the hazard ratio.

◁

Withdrawal of subjects from the study

Under administrative censoring, the subject is known to have experienced either of the two outcomes by the end of the study: survival or failure. Often, in practice, subjects may withdraw from the study before it terminates and therefore may not experience an event by the end of the study (or be censored) for nonadministrative reasons. Withdrawal of subjects from a study may greatly affect the estimate of the sample size and must be accounted for in the computations. Refer to *Survival data* in [PSS-2] **Intro (power)** and [PSS-5] **Glossary** for a formal definition of withdrawal.

Freedman (1982) suggests a conservative adjustment for the estimate of the sample size in the presence of withdrawal, which is implemented in `power logrank`. Withdrawal is assumed to be independent of failure (event) times and administrative censoring.

The proportion of subjects anticipated to withdraw from a study may be specified by using `wdprob()`.

▷ Example 7: Withdrawal of subjects from the study

Continuing with example 4, suppose that a withdrawal rate of 10% is expected in the study of colon cancer patients. To account for this, we also specify `wdprob(0.1)`.

```
. power logrank 0.5 0.6, onesided wdprob(0.1)

Estimated sample sizes for two-sample comparison of survivor functions
Log-rank test, Freedman method
H0: HR = 1  versus  Ha: HR < 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    0.7370  (hazard ratio)
       hratio =    0.7370

Censoring and withdrawal:

           s1 =    0.5000
           s2 =    0.6000
         Pr_E =    0.4500
         Pr_w =    0.1000

Estimated number of events and sample sizes:

            E =       270
            N =       666
  N per group =       333
```

The estimate of the total sample size using the Freedman method increases from 600 to 666 when the withdrawal rate is assumed to be 10%. The adjustment of the estimate of the sample size for the withdrawal of subjects is conservative. It assumes equal withdrawals from each group; that is, 10% of subjects are lost by the end of the study in each group. This adjustment affects only the estimates of the sample sizes but not the number of events. This is because withdrawal is assumed to be independent of event times, and the ratio of subjects surviving until the end of the study in the two groups does not change under equal withdrawals.

◁

Including information about subject accrual

Many clinical studies have an accrual period of r, during which the subjects are recruited to the study, and a follow-up period of $f = T - r$, during which the subjects are followed until the end of the study, T, and no new subjects enter the study. The information about the duration of the accrual and follow-up periods affects the probability of a subject experiencing an event or failing during the study.

Freedman (1982) suggests approximating the combined event-free probability as an average of the survival probabilities in the control and the experimental groups at the minimum follow-up time, $t^* = f$ (the default approach used in `power logrank`). However, for a long accrual of subjects, this approach may overestimate the required number of subjects, often seriously, because it does not take into account the information about subject follow-up beyond time f. Here Freedman (1982) proposes to use the survival probabilities at the average follow-up time, defined as $t^* = (f + T)/2 = f + 0.5r$, instead of the minimum follow-up time, f.

Alternatively, Schoenfeld (1983) (see also Collett [2015, 479]) presents a formula for the required number of subjects allowing for uniform accrual (entry, recruitment) over $[0, r]$ and a follow-up period, f. This information is incorporated into the formula for the probability of an event (or failure). The formula involves the integrals of the survivor functions of the control and the experimental groups. Schoenfeld (1983) suggests approximating the integral by using Simpson's rule, which requires the estimates of the

survivor function at three specific time points: f, $0.5r + f$, and $T = r + f$. It is sufficient to provide the estimates of these three survival probabilities, $S_1(f)$, $S_1(0.5r+f)$, and $S_1(T)$, for the control group only. The corresponding survival probabilities of the experimental group are automatically computed using the value of the hazard ratio in `hratio()` (or log hazard-ratio in `lnhratio()`) and the proportional-hazards assumption.

The three estimates of the survival probabilities of the control group may be supplied by using the `simpson()` option to adjust the estimates of the sample size or power for uniform entry and a follow-up period. If the estimate of the survivor function over an array of values in the range $[f, T]$ is available from a previous study, it can be supplied using the `st1()` option to form a more accurate approximation of the probability of an event using numerical integration (see [R] **dydx**). Here the value of the length of the accrual period is needed for the computation. It is computed as the difference between the maximum and the minimum values of the time variable *varname*$_t$, supplied using `st1()`, that is, $r = T - f = \max(\textit{varname}_t) - \min(\textit{varname}_t)$.

For more information, see Cleves, Gould, and Marchenko (2016, sec. 16.2).

▷ Example 8: Sample size in the presence of accrual and follow-up periods

Consider an example described in Collett (2015, 482) of a survival study of chronic active hepatitis. A new treatment is to be compared with a standard treatment with respect to the survival times of the patients with this disease. The investigators want to detect a change in a hazard ratio of 0.57 with 90% power and a 5% two-sided significance level. Also subjects are to be entered into the study uniformly over a period of 18 months and then followed for 24 months. From the Kaplan–Meier estimate of the survivor function available for the control group, the survival probabilities at $f = 24$, $0.5r + f = 33$, and $T = 42$ months are 0.70, 0.57, and 0.45, respectively.

```
. power logrank, hratio(0.57) power(0.9) schoenfeld simpson(0.7 0.57 0.45)
note: probability of an event is computed using Simpson's rule with
      S1(f) = 0.70, S1(f+r/2) = 0.57, S1(T) = 0.45
      S2(f) = 0.82, S2(f+r/2) = 0.73, S2(T) = 0.63

Estimated sample sizes for two-sample comparison of survivor functions
Log-rank test, Schoenfeld method
H0: ln(HR) = 0  versus  Ha: ln(HR) != 0

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =   -0.5621  (log hazard-ratio)
       hratio =    0.5700

Censoring:

         Pr_E =    0.3514

Estimated number of events and sample sizes:

            E =       134
            N =       380
  N per group =       190
```

Collett (2015, 309) reports the required number of events to be 133, which, apart from rounding, agrees with our estimate of 134. In a later example, Collett (2003, 309) uses the number of events, rounded to 140, to compute the required sample size as $140/0.35 = 400$, where 0.35 is the estimate of the combined probability of an event. By hand, without rounding the number of events, we compute the required sample size as $133/0.35 = 380$ and obtain the same estimate of the total sample size as in the output.

Using the average follow-up time suggested by Freedman (1982), we obtain the following:

```
. power logrank 0.57, hratio(0.57) power(0.9) schoenfeld

Estimated sample sizes for two-sample comparison of survivor functions
Log-rank test, Schoenfeld method
H0: ln(HR) = 0  versus  Ha: ln(HR) != 0

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =   -0.5621  (log hazard-ratio)
       hratio =    0.5700

Censoring:

           s1 =    0.5700
           s2 =    0.7259
         Pr_E =    0.3521

Estimated number of events and sample sizes:

            E =       134
            N =       378
  N per group =       189
```

We specify the survival probability in the control group at $t^* = 0.5r + f = 0.5 \times 18 + 24 = 33$ as $S_1(33) = 0.57$ and the hazard ratio of 0.57 (coincidentally). The survival probability in the experimental group is $S_2(33) = S_1(33)^{\Delta} = 0.57^{0.57} = 0.726$. Here we obtain the estimate of the sample size, 378, which is close to the estimate of 380 computed using the more complicated approximation. In this example, the two approximations produce similar results, but this may not always be the case.

The approximation suggested by Schoenfeld (1983) and Collett (2015) is considered to be more accurate because it takes into account information about the patient survival beyond the average follow-up time. In general, the Freedman (1982) and Schoenfeld (1983) approximations tend to give similar results when $\{\tilde{S}(f) + \tilde{S}(T)\}/2 \approx \tilde{S}(0.5r + f)$; see *Methods and formulas* for a formal definition of $\tilde{S}(\cdot)$.

If we use the survival probability in the control group, $S_1(24) = 0.7$, at a follow-up time $t^* = f = 24$ instead of the average follow-up time $t^* = 33$ in the presence of an accrual period,

```
. power logrank 0.7, hratio(0.57) power(0.9) schoenfeld

Estimated sample sizes for two-sample comparison of survivor functions
Log-rank test, Schoenfeld method
H0: ln(HR) = 0  versus  Ha: ln(HR) != 0

Study parameters:

        alpha =    0.0500
        power =    0.9000
        delta =   -0.5621  (log hazard-ratio)
       hratio =    0.5700

Censoring:

           s1 =    0.7000
           s2 =    0.8160
         Pr_E =    0.2420

Estimated number of events and sample sizes:

            E =       134
            N =       550
  N per group =       275
```

we obtain the estimate of the total sample size of 550, which is substantially greater than the previously estimated sample sizes of 380 and 378.

◁

Computing power

Sometimes the number of subjects available for the enrollment into the study is limited. In such cases, the researchers may want to investigate with what power they can detect a desired treatment effect for a given sample size.

To compute power, you must specify the sample size in the `n()` option and an effect size (a hazard ratio or a log hazard-ratio). A hazard ratio of 0.5 is assumed if an effect size is not specified.

▷ Example 9: Power determination

Recall the colon cancer study described in example 4. Suppose that only 100 subjects are available to be recruited to the study. We find out how this affects the power to detect a hazard ratio of 0.737.

```
. power logrank 0.5, hratio(0.737) onesided n(100)
Estimated power for two-sample comparison of survivor functions
Log-rank test, Freedman method
H0: HR = 1  versus  Ha: HR < 1
Study parameters:
        alpha =    0.0500
            N =       100
  N per group =        50
        delta =    0.7370  (hazard ratio)
       hratio =    0.7370
Number of events and censoring:
            E =        46
           s1 =    0.5000
           s2 =    0.6000
         Pr_E =    0.4500
Estimated power:
        power =    0.2646
```

The power to detect an alternative H_a: $\Delta = 0.737$ decreased from 0.8 to 0.2646 when the sample size decreased from 600 to 100 (the number of events decreased from 270 to 46).

◁

▷ Example 10: Multiple values of study parameters

Continuing with example 9, suppose we want to consider a range of sample sizes. We can specify a list (see **[U] 11.1.8 numlist**) of sample sizes in the `n()` option. For simplicity, we display only power, sample size, and number of events in the table.

```
. power logrank 0.5, hratio(0.737) onesided n(100(100)600) table(power N E)
Estimated power for two-sample comparison of survivor functions
Log-rank test, Freedman method
H0: HR = 1  versus  Ha: HR < 1
```

power	N	E
.2646	100	46
.4174	200	91
.5455	300	136
.6505	400	181
.7344	500	226
.8004	600	271

As the sample size increases, the power increases. The decrease in sample size reduces the number of events observed in the study and therefore changes the estimates of the power. If the number of events were fixed, power would have been independent of the sample size, provided that all other parameters were held constant, because the formulas relate power directly to the number of events and not the number of subjects.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

◁

Computing effect size

Effect size δ for the log-rank test comparing two survivor functions is defined as a hazard ratio (or a log hazard-ratio) of the experimental group to the control group. This value typically defines the clinically significant improvement of the experimental procedure over the control procedure desired to be detected by the log-rank test with a certain power.

Sometimes, we may be interested in determining the smallest effect that yields a statistically significant result for prespecified sample size and power. In this case, both power and sample size must be specified in options `power()` and `n()`, respectively. Additionally, you may also choose the direction of the effect by specifying the `direction()` option. `direction(lower)` is the default, and it assumes $\Delta_a < 1$ [or $\ln(\Delta_a) < 0$]. You can use `direction(upper)` to compute $\Delta_a > 1$ [or $\ln(\Delta_a) > 0$].

▷ Example 11: Effect-size determination

Continuing with example 10, we can find that the value of the hazard ratio that can be detected for a fixed sample size of 100 with 80% power is approximately 0.42, corresponding to an increase in survival probability from 0.5 to roughly 0.75.

```
. power logrank 0.5, onesided n(100) power(0.8)
Performing iteration ...
Estimated hazard ratio for two-sample comparison of survivor functions
Log-rank test, Freedman method
H0: HR = 1  versus  Ha: HR < 1
Study parameters:
        alpha =    0.0500
        power =    0.8000
            N =       100
  N per group =        50
Number of events and censoring:
            E =        38
           s1 =    0.5000
           s2 =    0.7455
         Pr_E =    0.3772
Estimated effect size and hazard ratio:
        delta =    0.4237  (hazard ratio)
       hratio =    0.4237
```

Under the censoring information, `power logrank` also reports the experimental-group survival rate at the end of the study corresponding to the computed hazard ratio—`s2=0.7455` in our example.

◁

Testing a hypothesis about two survivor functions using the log-rank test

▷ Example 12: Using the log-rank test to detect a change in survival in two groups

Similarly to example 4, consider the generated dataset `drug.dta`, consisting of variables `drug` (a drug type) and `failtime` (a time to failure).

```
. use https://www.stata-press.com/data/r19/drug
(Patient survival in drug trial)
. tabulate drug
```

Treatment type	Freq.	Percent	Cum.
Placebo	50	33.33	33.33
Drug A	50	33.33	66.67
Drug B	50	33.33	100.00
Total	150	100.00	

```
. by drug, sort: summarize failtime
```

-> drug = Placebo

Variable	Obs	Mean	Std. dev.	Min	Max
failtime	50	1.03876	.5535538	.1687701	2.382302

-> drug = Drug A

Variable	Obs	Mean	Std. dev.	Min	Max
failtime	50	1.191802	.5927507	.2366922	2.277536

-> drug = Drug B

Variable	Obs	Mean	Std. dev.	Min	Max
failtime	50	1.717314	.8350659	.5511715	3.796102

Failure times of the control group (`Placebo`) were generated from the Weibull distribution with $\lambda_w = 0.693$ and $p = 2$ (see [ST] **streg**); failure times of the two experimental groups, `Drug A` and `Drug B`, were generated from Weibull distributions with hazard functions proportional to the hazard of the control group in ratios 0.737 and 0.42, respectively. The Weibull family of survival distributions is chosen arbitrarily, and the Weibull parameter, λ_w, is chosen such that the survival at 1 year, $t = 1$, is roughly equal to 0.5. Subjects are randomly allocated to one of the three groups in equal proportions. Subjects with failure times greater than $t = 1$ will be censored at $t = 1$.

Before analyzing these survival data, we need to set up the data using `stset`. After that, we can use `sts test, logrank` to test the survivor functions separately for `Drug A` against `Placebo` and `Drug B` against `Placebo` by using the log-rank test. See [ST] **stset** and [ST] **sts test** for more information about these two commands.

```
. stset failtime, exit(time 1)
Survival-time data settings
         Failure event: (assumed to fail at time=failtime)
Observed time interval: (0, failtime]
     Exit on or before: time 1
--------------------------------------------------------------------------
        150  total observations
          0  exclusions
--------------------------------------------------------------------------
        150  observations remaining, representing
         59  failures in single-record/single-failure data
    128.985  total analysis time at risk and under observation
                                                At risk from t =         0
                                     Earliest observed entry t =         0
                                          Last observed exit t =         1

. sts test drug if drug!=2, logrank
         Failure _d: 1 (meaning all fail)
   Analysis time _t: failtime
  Exit on or before: time 1

Equality of survivor functions
Log-rank test
          |  Observed       Expected
drug      |    events         events
----------+-------------------------
Placebo   |        25          22.17
 Drug A   |        21          23.83
----------+-------------------------
  Total   |        46          46.00
                 chi2(1) =   0.70
                 Pr>chi2 = 0.4028

. sts test drug if drug!=1, logrank
         Failure _d: 1 (meaning all fail)
   Analysis time _t: failtime
  Exit on or before: time 1

Equality of survivor functions
Log-rank test
          |  Observed       Expected
drug      |    events         events
----------+-------------------------
Placebo   |        25          16.61
 Drug B   |        13          21.39
----------+-------------------------
  Total   |        38          38.00
                 chi2(1) =   7.55
                 Pr>chi2 = 0.0060
```

From the results from `sts test` for the `Drug A` group, we fail to reject the null hypothesis of no difference between the survivor functions in the two groups; given our simulated data, the test made a type II error. On the other hand, for the `Drug B` group the one-sided p-value of 0.003, computed as $0.006/2 = 0.003$, suggests that the null hypothesis of nonsuperiority of the experimental treatment be rejected at the 0.005 significance level. We correctly conclude that the data provide the evidence that `Drug B` is superior to the `Placebo`.

Results from `sts test, logrank` for the two experimental groups agree with findings from examples 9 and 11. For the sample size of 100, the power of the log-rank test to detect the hazard ratio of 0.737 (10% increase in survival) is low (26%), whereas this sample size is sufficient for the test to detect a change in a hazard of 0.42 (25% increase in survival) with approximately 80% power.

Here we simulated our data from the alternative hypothesis and therefore can determine whether the correct decision or a type II error was made by the test. In practice, however, there is no way to determine the accuracy of the decision from the test. All we know is that in a long series of trials, there is a 5% chance that a particular test will incorrectly reject the null hypothesis, a 74% [$(1 - 0.25) \times 100\%$ given the power of 0.2646 obtained in example 9] chance that the test will miss the alternative H_a: $\Delta = 0.737$, and a 20% [$(1 - 0.8) \times 100\%$ given the power of 0.8 in example 11] chance that the test will miss the alternative H_a: $\Delta = 0.42$.

◁

Stored results

`power logrank` stores the following in `r()`:

Scalars

`r(alpha)`	significance level
`r(power)`	power
`r(beta)`	probability of a type II error
`r(delta)`	effect size
`r(N)`	total sample size
`r(N_a)`	actual sample size
`r(N1)`	sample size of the control group
`r(N2)`	sample size of the experimental group
`r(nratio)`	ratio of sample sizes, `N2/N1`
`r(nratio_a)`	actual ratio of sample sizes
`r(nfractional)`	1 if `nfractional` is specified, 0 otherwise
`r(onesided)`	1 for a one-sided test, 0 otherwise
`r(E)`	total number of events (failures)
`r(hratio)`	hazard ratio
`r(lnhratio)`	log hazard-ratio
`r(s1)`	survival probability in the control group (if specified)
`r(s2)`	survival probability in the experimental group (if specified)
`r(Pr_E)`	probability of an event (failure)
`r(Pr_w)`	proportion of withdrawals
`r(t_min)`	minimum time (if `st1()` is specified)
`r(t_max)`	maximum time (if `st1()` is specified)
`r(separator)`	number of lines between separator lines in the table
`r(divider)`	1 if `divider` is requested in the table, 0 otherwise
`r(init)`	initial value for hazard ratio or log hazard-ratio
`r(maxiter)`	maximum number of iterations
`r(iter)`	number of iterations performed
`r(tolerance)`	requested parameter tolerance
`r(deltax)`	final parameter tolerance achieved
`r(ftolerance)`	requested distance of the objective function from zero
`r(function)`	final distance of the objective function from zero
`r(converged)`	1 if iteration algorithm converged, 0 otherwise

Macros

`r(type)`	`test`
`r(method)`	`logrank`
`r(test)`	`Freedman` or `Schoenfeld`
`r(effect)`	`hratio` or `lnhratio`
`r(survvar)`	name of the variable containing survival probabilities (if `st1()` is specified)

`r(timevar)`	name of the variable containing time points (if `st1()` is specified)
`r(direction)`	`lower` or `upper`
`r(columns)`	displayed table columns
`r(labels)`	table column labels
`r(widths)`	table column widths
`r(formats)`	table column formats

Matrices

`r(pss_table)`	table of results
`r(simpmat)`	control-group survival probabilities (if `simpson()` is specified)

Methods and formulas

Let $S_1(t)$ and $S_2(t)$ denote the survivor functions of the control and the experimental groups and $\Delta(t) = \ln\{S_2(t)\}/\ln\{S_1(t)\}$ denote the hazard ratio at time t of the experimental to the control groups. Thus, for a given constant hazard ratio Δ, the survivor function of the experimental group at any time $t > 0$ may be computed as $S_2(t) = \{S_1(t)\}^{\Delta}$ under the assumption of proportional hazards. Define E and n to be the total number of events and the total number of subjects required for the study, respectively; p_w to be the proportion of subjects withdrawn from the study (lost to follow-up); and $z_{(1-\alpha/k)}$ and $z_{(1-\beta)}$ to be the $(1-\alpha/k)$th and the $(1-\beta)$th quantiles of the standard normal distribution, respectively, with $k = 1$ for the one-sided test and $k = 2$ for the two-sided test. Let R be the allocation ratio to the experimental group with respect to the control group, that is, $n_2 = Rn_1$.

The total number of events required to be observed in a study to ensure a power of $\pi = 1-\beta$ of the log-rank test to detect the hazard ratio Δ with significance level α, according to Freedman (1982), is

$$E = \frac{1}{R}(z_{1-\alpha/k} + z_{1-\beta})^2 \left(\frac{R\Delta + 1}{\Delta - 1}\right)^2$$

and, according to Schoenfeld (1983) and Collett (2015, 473), is

$$E = \frac{1}{R}(z_{1-\alpha/k} + z_{1-\beta})^2 \left\{\frac{1+R}{\ln(\Delta)}\right\}^2$$

Both formulas are approximations and rely on a set of assumptions such as distinct failure times, all subjects completing the course of the study (no withdrawal), and a constant ratio, R, of subjects at risk in two groups at each failure time.

The total sample size required to observe the total number of events, E, is given by

$$n = \frac{E}{p_E}$$

The number of subjects required to be recruited in each group is obtained as $n_1 = n/(1+R)$ and $n_2 = nR/(1+R)$. If `nfractional` is not specified, sample sizes and the number of events are rounded to integer values; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for details.

By default, the probability of an event (failure), p_E, is approximated as suggested by Freedman (1982),

$$p_E = 1 - \frac{S_1(t^*) + RS_2(t^*)}{1+R}$$

where t^* is the minimum follow-up time, f, or, in the presence of an accrual period, the average follow-up time, $(f+T)/2 = f + 0.5r$.

If `simpson()` is specified, the probability of an event is approximated using Simpson's rule as suggested by Schoenfeld (1983):

$$p_E = 1 - \frac{1}{6}\left\{\tilde{S}(f) + 4\tilde{S}(0.5r+f) + \tilde{S}(T)\right\}$$

where $\tilde{S}(t) = \{S_1(t) + RS_2(t)\}/(1+R)$ and f, r, and $T = f + r$ are the follow-up period, the accrual period, and the total duration of the study, respectively.

The methods do not incorporate time explicitly but rather use it to determine values of the survival probabilities $S_1(t)$ and $S_2(t)$ used in the computations.

If `st1()` is used, the integral in the expression for the probability of an event

$$p_E = 1 - \frac{1}{r}\int_f^T \tilde{S}(t)dt$$

is computed numerically using cubic splines (see [R] **dydx**). The value of r is computed as the difference between the maximum and the minimum values of *varname*$_t$ in `st1()`, $r = T - f = \max(\textit{varname}_t) - \min(\textit{varname}_t)$.

To account for the proportion of subjects, p_w, withdrawn from the study (lost to follow-up), a conservative adjustment to the total sample size is applied as follows:

$$n_w = \frac{n}{1-p_w}$$

Equal withdrawal rates are assumed in the adjustment of the group sample sizes for the withdrawal of subjects. Equal withdrawals do not affect the estimates of the number of events, provided that withdrawal is independent of event times and the ratio of subjects at risk in two groups remains constant at each failure time.

The power for each method is estimated using the formula

$$\pi = 1 - \beta = \Phi\{|\psi|^{-1}(Rnp_E)^{1/2} - z_{1-\alpha/k}\}$$

where $\Phi(\cdot)$ is the standard normal cumulative distribution function; $\psi = (R\Delta+1)/(\Delta-1)$ or $\psi = (1+R)/\ln(\Delta)$ if the `schoenfeld` option is specified.

The estimate of the hazard ratio (or log hazard-ratio) for fixed power and sample size is computed (iteratively for censoring) using the formulas for the sample size given above. The value of the hazard ratio (log hazard-ratio) corresponding to the reduction in a hazard of the experimental group relative to the control group is reported by default.

References

Cleves, M. A., W. W. Gould, and Y. V. Marchenko. 2016. *An Introduction to Survival Analysis Using Stata*. Rev. 3rd ed. College Station, TX: Stata Press.

Collett, D. 2015. *Modelling Survival Data in Medical Research*. 3rd ed. Boca Raton, FL: Chapman and Hall/CRC.

Freedman, L. S. 1982. Tables of the number of patients required in clinical trials using the logrank test. *Statistics in Medicine* 1: 121–129. https://doi.org/10.1002/sim.4780010204.

Hsieh, F. Y. 1992. Comparing sample size formulae for trials with unbalanced allocation using the logrank test. *Statistics in Medicine* 11: 1091–1098. https://doi.org/10.1002/sim.4780110810.

Lakatos, E., and K. K. G. Lan. 1992. A comparison of sample size methods for the logrank statistic. *Statistics in Medicine* 11: 179–191. https://doi.org/10.1002/sim.4780110205.

Machin, D., M. J. Campbell, S. B. Tan, and S. H. Tan. 2009. *Sample Size Tables for Clinical Studies*. 3rd ed. Chichester, UK: Wiley–Blackwell.

Marubini, E., and M. G. Valsecchi. 1995. *Analysing Survival Data from Clinical Trials and Observational Studies*. New York: Wiley.

Schoenfeld, D. A. 1981. The asymptotic properties of nonparametric tests for comparing survival distributions. *Biometrika* 68: 316–319. https://doi.org/10.2307/2335833.

———. 1983. Sample-size formula for the proportional-hazards regression model. *Biometrics* 39: 499–503. https://doi.org/10.2307/2531021.

Schoenfeld, D. A., and J. R. Richter. 1982. Nomograms for calculating the number of patients needed for a clinical trial with survival as an endpoint. *Biometrics* 38: 163–170. https://doi.org/10.2307/2530299.

Also see [PSS-2] **Intro (power)** for more references.

Also see

[PSS-2] **power logrank, cluster** — Power analysis for the log-rank test, CRD

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power cox** — Power analysis for the Cox proportional hazards model

[PSS-2] **power exponential** — Power analysis for a two-sample exponential test

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[ADAPT] **gsdesign logrank** — Group sequential design for a log-rank test

[ST] **stcox** — Cox proportional hazards model

[ST] **sts test** — Test equality of survivor functions

power logrank, cluster — Power analysis for the log-rank test, CRD

Description

`power logrank, cluster` computes group-specific numbers of clusters, group-specific cluster sizes, power, or the hazard ratio for the log-rank test comparing survivor functions in two groups in a cluster randomized design (CRD). Without censoring, the survival input parameter is the hazard ratio; otherwise, the survival input parameters are the control-group and experimental-group survival probabilities.

`power logrank, cluster` computes group-specific numbers of clusters given cluster sizes, power, and survival parameters. It also computes group-specific cluster sizes given numbers of clusters, power, and survival parameters. Alternatively, it computes power given numbers of clusters, cluster sizes, and survival parameters, or it computes the hazard ratio given numbers of clusters, cluster sizes, power, and, in the presence of censoring, the control-group survival probability. See [PSS-2] **power logrank** for a general discussion of power and sample-size analysis for the log-rank test. Also see [PSS-2] **power** for a general introduction to the `power` command using hypothesis tests.

Quick start

Numbers of clusters for uncensored design with alternative hazard ratio $\Delta_a = 0.76$ for the log-rank test of H_0: $\Delta = 1$ versus H_a: $\Delta \neq 1$ given a cluster size of 5 in both groups and using default intraclass correlation of 0.5, power of 0.8, and significance level $\alpha = 0.05$

```
power logrank, m1(5) m2(5) hratio(0.76)
```

Same as above, but use intraclass correlation of 0.4

```
power logrank, m1(5) m2(5) hratio(0.76) rho(0.4)
```

Assume cluster sizes vary with an average cluster size of 5 and a coefficient of variation of 0.6

```
power logrank, m1(5) m2(5) hratio(0.76) cvcluster(0.6)
```

Group-specific numbers of clusters using ratio of experimental clusters to control clusters of 0.5

```
power logrank, m1(5) m2(5) hratio(0.76) kratio(0.5)
```

Cluster sizes for censored design with control- and experimental-group survival probabilities of 0.3 and 0.4 for the log-rank test of H_0: $\Delta = 1$ versus H_a: $\Delta \neq 1$ with 200 clusters in both groups using default intraclass correlation of 0.5, power of 0.8, and significance level $\alpha = 0.05$

```
power logrank 0.3 0.4, k1(200) k2(200)
```

Power for 50 clusters of size 5 in the control group and 50, 100, 150, or 200 clusters of size 5 in the experimental group with results shown in a power plot

```
power logrank 0.3 0.4, k1(50) k2(50(50)200) m1(5) m2(5) graph
```

Hazard ratio and experimental-group survival probability with power of 0.8

```
power logrank 0.3, k1(50) k2(50) m1(5) m2(5) power(0.8)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute numbers of clusters

```
power logrank [surv1 [surv2]], {mspec | nspec cluster} [options]
```

Compute cluster sizes

```
power logrank [surv1 [surv2]], kspec [options]
```

Compute power

```
power logrank [surv1 [surv2]], kspec {mspec | nspec} [options]
```

Compute effect size

```
power logrank [surv1], kspec {mspec | nspec} power(numlist) [options]
```

where $surv_1$ is the survival probability in the control (reference) group at the end of the study t^* and $surv_2$ is the survival probability in the experimental (comparison) group at the end of the study t^*. $surv_1$ and $surv_2$ may each be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

mspec is one of

```
m1()  m2()
m1()  [mratio()]
m2()  [mratio()]
```

nspec is one of

```
n1()  n2()
n1()  [nratio()]
n2()  [nratio()]
```

kspec is one of

```
k1()  k2()
k1()  [kratio()]
k2()  [kratio()]
```

options	Description
Main	
`cluster`	perform computations for a CRD; implied by `k1()`, `k2()`, `m1()`, or `m2()`
* `alpha(`*numlist*`)`	significance level; default is `alpha(0.05)`
* `power(`*numlist*`)`	power; default is `power(0.8)`
* `beta(`*numlist*`)`	probability of type II error; default is `beta(0.2)`
* `k1(`*numlist*`)`	number of clusters in the control group
* `k2(`*numlist*`)`	number of clusters in the experimental group
* `kratio(`*numlist*`)`	cluster ratio, K2/K1; default is `kratio(1)`
* `m1(`*numlist*`)`	cluster size of the control group
* `m2(`*numlist*`)`	cluster size of the experimental group
* `mratio(`*numlist*`)`	cluster-size ratio, M2/M1; default is `mratio(1)`
* `n1(`*numlist*`)`	sample size of the control group
* `n2(`*numlist*`)`	sample size of the experimental group
* `nratio(`*numlist*`)`	sample-size ratio, N2/N1; default is `nratio(1)`
`nfractional`	allow fractional numbers of clusters, cluster sizes, and sample sizes
* `hratio(`*numlist*`)`	hazard ratio of the experimental to the control group; default is `hratio(0.5)`
* `lnhratio(`*numlist*`)`	log hazard-ratio of the experimental to the control group
* `rho(`*numlist*`)`	intraclass correlation; default is `rho(0.5)`
* `cvcluster(`*numlist*`)`	coefficient of variation for cluster sizes
`direction(lower|upper)`	direction of the effect for effect-size determination; default is `direction(lower)`, which means that the postulated value of the parameter is smaller than the hypothesized value
`onesided`	one-sided test; default is two sided
`parallel`	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
`[no]table[(`*tablespec*`)]`	suppress table or display results as a table; see [PSS-2] **power, table**
`saving(`*filename*` [, replace])`	save the table data to *filename*; use `replace` to overwrite existing *filename*
Graph	
`graph[(`*graphopts*`)]`	graph results; see [PSS-2] **power, graph**

Iteration

`init(#)`	initial value for hazard ratio
`iterate(#)`	maximum number of iterations; default is `iterate(500)`
`tolerance(#)`	parameter tolerance; default is `tolerance(1e-12)`
`ftolerance(#)`	function tolerance; default is `ftolerance(1e-12)`
[`no`]`log`	suppress or display iteration log
[`no`]`dots`	suppress or display iterations as dots
`notitle`	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the `parallel` option.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`notitle` does not appear in the dialog box.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`alpha`	significance level	α
`power`	power	$1-\beta$
`beta`	type-II-error probability	β
`K1`	number of clusters in the control group	K_1
`K2`	number of clusters in the experimental group	K_2
`kratio`	ratio of numbers of clusters, experimental to control	K_2/K_1
`M1`	cluster size of the control group	M_1
`M2`	cluster size of the experimental group	M_2
`mratio`	ratio of cluster sizes, experimental to control	M_2/M_1
`N`	total number of observations	N
`N1`	number of observations in the control group	N_1
`N2`	number of observations in the experimental group	N_2
`nratio`	ratio of sample sizes, experimental to control	N_2/N_1
`delta`	effect size	δ
`E`	total number of events (failures)	E
`hratio`	hazard ratio	Δ
`lnhratio`	log hazard-ratio	$\ln(\Delta)$
`s1`	survival probability in the control group	$S_1(T)$
`s2`	survival probability in the experimental group	$S_2(T)$
`Pr_E`	overall probability of an event (failure)	p_E
`rho`	intraclass correlation	ρ
`CV_cluster`	coefficient of variation for cluster sizes	$\mathrm{CV}_{\mathrm{cl}}$
`target`	target parameter; `hratio`	
`_all`	display all supported columns	

Column beta is shown in the default table in place of column power if specified.
Column N is shown in the table if specified.
Columns N1 and N2 are shown in the default table if n1() or n2() is specified.
Column lnhratio is shown in the default table in place of column hratio if specified.
Columns s1 and s2 are available only when specified.
Columns nratio and CV_cluster are shown in the default table if specified.

Options

Main

cluster specifies that computations should be performed for a CRD. This option is implied when the k1(), k2(), m1(), or m2() option is specified. cluster is required to compute the numbers of clusters when *nspec* is used to specify sample sizes instead of *mspec* for cluster sizes.

alpha(), power(), beta(); see [PSS-2] **power**.

k1(*numlist*) specifies the number of clusters in the control group.

k2(*numlist*) specifies the number of clusters in the experimental group.

kratio(*numlist*) specifies the ratio of the numbers of clusters of the experimental group relative to the control group, K2/K1. The default is kratio(1), meaning equal numbers of clusters in the two groups.

m1(*numlist*) specifies the cluster size of the control group. m1() may contain noninteger values.

m2(*numlist*) specifies the cluster size of the experimental group. m2() may contain noninteger values.

mratio(*numlist*) specifies the ratio of cluster sizes of the experimental group relative to the control group, M2/M1. The default is mratio(1), meaning equal cluster sizes in the two groups.

n1(), n2(), nratio(); see [PSS-2] **power**.

nfractional; see [PSS-2] **power**. The nfractional option displays fractional (without rounding) values of the numbers of clusters, cluster sizes, and sample sizes.

hratio(), lnhratio(); see [PSS-2] **power logrank**.

rho(*numlist*) specifies the intraclass correlation. The default is rho(0.5).

cvcluster(*numlist*) specifies the coefficient of variation for cluster sizes. This option is used with varying cluster sizes.

direction(), onesided, parallel; see [PSS-2] **power**. direction(lower) is the default.

Table

table, table(), notable; see [PSS-2] **power, table**.

saving(); see [PSS-2] **power**.

Graph

graph, graph(); see [PSS-2] **power, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies the initial value for the hazard ratio.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-2] **power**.

The following option is available with `power logrank, cluster` but is not shown in the dialog box:

`notitle`; see [PSS-2] **power**.

Remarks and examples

Remarks are presented under the following headings:

`power logrank, cluster` requests that computations for the `power logrank` command be done for a CRD. In a CRD, groups of subjects or clusters are randomized instead of individual subjects, so the sample size is determined by the numbers of clusters and the cluster sizes. The sample-size determination thus consists of the determination of the numbers of clusters given cluster sizes or the determination of cluster sizes given the numbers of clusters. For a general discussion of using `power logrank`, see [PSS-2] **power logrank**. The discussion below is specific to the CRD.

Using power logrank, cluster

If you specify the `cluster` option, include `k1()` or `k2()` to specify the number of clusters or include `m1()` or `m2()` to specify the cluster size, the `power logrank` command will perform computations for the log-rank test in a CRD. The computations for a CRD are based on the Freedman method; see *Introduction* in [PSS-2] **power logrank** for details.

All computations are performed for a two-sided hypothesis test where, by default, the significance level is set to 0.05. You may change the significance level by specifying the `alpha()` option. You can specify the `onesided` option to request a one-sided test. By default, all computations assume a balanced or equal-allocation design, meaning equal numbers of clusters and cluster sizes in both groups; see [PSS-4] **Unbalanced designs** for a description of how to specify an unbalanced design.

To compute the number of clusters in both groups, you must provide cluster sizes for both groups. There are multiple ways to supply cluster sizes, but the most common is to specify the cluster size of the control group in the `m1()` option and the cluster size of the experimental group in the `m2()` option. See *mspec* and *nspec* under *Syntax* for other specifications. When *nspec* is specified, the `cluster` option is also required to request that `power logrank` perform computations for a CRD. The number of clusters is assumed to be equal in the two groups, but you can change this by specifying the ratio of the numbers of clusters in the experimental to the control group in the `kratio()` option. Other parameters are specified as described in *Using power logrank* in [PSS-2] **power logrank**.

To compute the cluster sizes in both groups, you must provide the numbers of clusters in both groups. There are several ways to supply the numbers of clusters; see *kspec* under *Syntax*. The most common is to specify the numbers of clusters in the control group and the experimental group in the `k1()` and

k2() options, respectively. Equal cluster sizes are assumed in the two groups, but you can change this by specifying the ratio of the cluster sizes in the experimental to that of the control group in the mratio() option. Other parameters are specified as described in *Using power logrank* in [PSS-2] **power logrank**.

The power and effect-size determination is the same as described in *Using power logrank* in [PSS-2] **power logrank**, but the sample-size information is supplied as the numbers of clusters *kspec* and either cluster sizes using *mspec* or, less commonly, sample sizes using *nspec*.

All computations assume an intraclass correlation of 0.5. You can change this by specifying the rho() option. Also, all clusters are assumed to be of the same size unless the coefficient of variation for cluster sizes is specified in the cvcluster() option.

By default, the computed numbers of clusters, cluster sizes, and sample sizes are rounded up. However, you can specify the nfractional option to see the corresponding fractional values; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. If the cvcluster() option is specified when computing cluster sizes, then cluster sizes represent average cluster sizes and are thus not rounded. When sample sizes are specified using *nspec*, fractional cluster sizes may be reported to accommodate the specified numbers of clusters and sample sizes.

Some of power logrank, cluster's computations require iteration; see *Methods and formulas* for details and [PSS-2] **power** for the descriptions of options that control the iteration procedure.

Computing numbers of clusters

To compute the numbers of clusters in each group, you must either provide the cluster size for each group using *mspec* or specify the cluster option and provide the sample sizes of both groups using *nspec*. The most common method is to use *mspec* of m1() and m2(). A hazard ratio of 0.5 is assumed but may be changed by specifying the hratio() option. If there is censoring, the control-group survival probability $surv_1$ must be specified, and the experimental-group survival probability $surv_2$ may be specified instead of the hazard ratio.

▷ Example 1: Numbers of clusters for the log-rank test with no censoring in a CRD

Consider an example from Xie and Waksman (2003). A researcher would like to study the time it takes diabetic foot ulcers to heal completely (the event of interest). Patients enrolled in the study may have multiple ulcers, and all existing ulcers will be treated. Suppose that at the end of the study, all ulcers are expected to be healed. Patients, the clustering unit, are randomly assigned equally to the control group with the usual treatment and the experimental group with the new treatment. The researcher is interested in detecting a hazard ratio, experimental group to control group, of 1.79 for healing. For the planned study, the researcher assumes about 3 ulcers per patient in each group with an intraclass correlation of 0.3.

To compute the numbers of patients in each group required to detect a hazard ratio of 1.79 with 80% power using a 5%-level two-sided test, we type

```
. power logrank, hratio(1.79) m1(3) m2(3) rho(0.3)

Estimated numbers of clusters for two-sample comparison of survivor functions
Cluster randomized design, log-rank test, Freedman method
H0: HR = 1  versus  Ha: HR != 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    1.7900  (hazard ratio)
       hratio =    1.7900

Cluster design:

           M1 =         3
           M2 =         3
          rho =    0.3000

Number of events and censoring:

            E =       157
         Pr_E =    1.0000

Estimated numbers of clusters and sample sizes:

           K1 =        27
           K2 =        27
           N1 =        81
           N2 =        81
```

We find that 27 patients per group are required to detect a hazard ratio of 1.79 with 80% power using a 5%-level two-sided test.

◁

▷ Example 2: Numbers of clusters for the log-rank test with censoring in a CRD

Unlike example 1, suppose that 70% of the ulcers in the control group and 50% in the experimental group are not healed at the end of the study. Because there is censoring, we specify the survival probabilities of 0.7 and 0.5 as command arguments.

```
. power logrank 0.7 0.5, m1(3) m2(3) rho(0.3)

Estimated numbers of clusters for two-sample comparison of survivor functions
Cluster randomized design, log-rank test, Freedman method
H0: HR = 1  versus  Ha: HR != 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    1.9434  (hazard ratio)
       hratio =    1.9434

Cluster design:

           M1 =         3
           M2 =         3
          rho =    0.3000

Number of events and censoring:

            E =       123
           s1 =    0.7000
           s2 =    0.5000
         Pr_E =    0.4000

Estimated numbers of clusters and sample sizes:

           K1 =        51
           K2 =        51
           N1 =       153
           N2 =       153
```

We find that 51 patients per group are required to detect a decrease in the survival rate or an increase in the healing rate of ulcers from 0.7 to 0.5 with 80% power using a 5%-level two-sided test.

◁

▷ Example 3: Numbers of clusters for the log-rank test in a CRD, varying cluster sizes

Continuing with example 2, we now want to account for the fact that the numbers of ulcers may vary among patients. We assume a coefficient of variation of 0.4. To compute the numbers of clusters when cluster sizes vary, we specify the coefficient of variation of the numbers of ulcers of 0.4 in the `cvcluster()` option.

```
. power logrank 0.7 0.5, m1(3) m2(3) rho(0.3) cvcluster(0.4)

Estimated numbers of clusters for two-sample comparison of survivor functions
Cluster randomized design, log-rank test, Freedman method
H0: HR = 1  versus  Ha: HR != 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    1.9434  (hazard ratio)
       hratio =    1.9434

Cluster design:

   Average M1 =    3.0000
   Average M2 =    3.0000
          rho =    0.3000
        CV_cl =    0.4000

Number of events and censoring:

            E =       134
           s1 =    0.7000
           s2 =    0.5000
         Pr_E =    0.4000

Estimated numbers of clusters and sample sizes:

           K1 =        56
           K2 =        56
           N1 =       168
           N2 =       168
```

The required number of patients in each group is 56, which is larger than the required number of patients of 51 in example 2. When the numbers of ulcers vary among patients, we need more patients to achieve the same power.

◁

Computing cluster sizes

To compute cluster sizes in both groups, you must provide the numbers of clusters in both groups by using *kspec*. The most common method is to specify the numbers of clusters in the control and experimental groups in the `k1()` and `k2()` options, respectively. As for the determination of the numbers of clusters, a hazard ratio of 0.5 is assumed but may be changed by specifying the `hratio()` option. If there is censoring, the control-group survival probability $surv_1$ must be specified, and the experimental-group survival probability $surv_2$ may be specified instead of the hazard ratio.

▷ Example 4: Cluster sizes for the log-rank test in a CRD

Continuing with example 2, suppose that we are designing a new study and planning to recruit 100 patients, with 50 patients in each group. Given other study parameters from example 2, we compute the number of ulcers we would expect to see from each patient to achieve the power of 80% by specifying 50 clusters in the `k1()` and `k2()` options.

```
. power logrank 0.7 0.5, k1(50) k2(50) rho(0.3)

Estimated cluster sizes for two-sample comparison of survivor functions
Cluster randomized design, log-rank test, Freedman method
H0: HR = 1  versus  Ha: HR != 1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    1.9434  (hazard ratio)
       hratio =    1.9434

Cluster design:

           K1 =        50
           K2 =        50
          rho =    0.3000

Number of events and censoring:

            E =        77
           s1 =    0.7000
           s2 =    0.5000
         Pr_E =    0.4000

Estimated cluster sizes and sample sizes:

           M1 =         4
           M2 =         4
           N1 =       200
           N2 =       200
```

To achieve the desired power of 80%, with 100 patients, we will need to observe 4 ulcers per patient.

◁

Computing power

To compute power in a CRD, you supply the sample-size information as the numbers of clusters by using *kspec* along with either the cluster sizes by using *mspec* or, less commonly, the sample sizes by using *nspec*. The most common method is to specify the `k1()`, `k2()`, `m1()`, and `m2()` options. A hazard ratio of 0.5 is assumed but may be changed by specifying the `hratio()` option. If there is censoring, the control-group survival probability $surv_1$ must be specified, and the experimental-group survival probability $surv_2$ may be specified instead of the hazard ratio.

▷ Example 5: Power for the log-rank test in a CRD

Continuing with example 2, suppose that we can recruit 100 patients (50 patients per group) and observe 3 ulcers per patient and we want to compute power for this design. Given other study parameters from example 2, we compute power by specifying 50 clusters in the `k1()` and `k2()` options and 3 as the cluster size in the `m1()` and `m2()` options.

```
. power logrank 0.7 0.5, k1(50) k2(50) m1(3) m2(3) rho(0.3)

Estimated power for two-sample comparison of survivor functions
Cluster randomized design, log-rank test, Freedman method
H0: HR = 1  versus  Ha: HR != 1

Study parameters:

        alpha =    0.0500
        delta =    1.9434  (hazard ratio)
       hratio =    1.9434

Cluster design:

           K1 =        50
           K2 =        50
           M1 =         3
           M2 =         3
           N1 =       150
           N2 =       150
          rho =    0.3000

Number of events and censoring:

            E =       120
           s1 =    0.7000
           s2 =    0.5000
         Pr_E =    0.4000

Estimated power:

        power =    0.7927
```

The computed power is about 79%.

Example 6: Multiple values of study parameters

To investigate the effect of the number of clusters in the experimental group on power, we can specify a list of numbers of clusters in the `k2()` option:

```
. power logrank 0.7 0.5, k1(50) k2(10(20)90) m1(3) m2(3) rho(0.3) table(power K2)

Estimated power for two-sample comparison of survivor functions
Cluster randomized design, log-rank test, Freedman method
H0: HR = 1  versus  Ha: HR != 1
```

power	K2
.4603	10
.7157	30
.7927	50
.8276	70
.8472	90

In this example, we also specified the `table(power K2)` option to list only the two columns that vary. As expected, as the number of clusters increases, the power tends to get closer to 1.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-2] **power, table**. If you wish to produce a power plot, see [PSS-2] **power, graph**.

Computing effect size

Effect size δ for the log-rank test is the hazard ratio. To compute effect size in a CRD, you supply the sample-size information as the numbers of clusters by using *kspec* along with either the cluster sizes by using *mspec* or, less commonly, the sample sizes by using *nspec*. The most common method is to specify the `k1()`, `k2()`, `m1()`, and `m2()` options. In addition, power and, in the presence of censoring, control-group survival probability must be specified. You must also decide on the direction of the effect, which is specified in the `direction()` option. For the default, lower, meaning $\Delta < 1$, `power logrank, cluster` uses `direction(lower)`. For upper, meaning $\Delta > 1$, specify `direction(upper)`.

Example 7: Effect size for the log-rank test in a CRD

Continuing with example 5, we may be interested in finding the minimum value of the hazard ratio that can be detected with a sample of 50 patients per group, with about 3 ulcers per patient, and with 80% power. To compute this, we specify the control-group survival probability of 0.7 as the command argument and required options `k1(50)`, `k2(50)`, `m1(3)`, `m2(3)`, and `power(0.8)`. We continue to use `rho(0.3)`. In this example, we want to detect an effect in the upper direction. That is, we believe the rate of healing in the experimental group is greater than in the control group, so we expect a hazard ratio greater than 1. Therefore, we need to specify the `direction(upper)` option.

```
. power logrank 0.7, k1(50) k2(50) m1(3) m2(3) power(0.8) rho(0.3)
> direction(upper)

Performing iteration ...

Estimated hazard ratio for two-sample comparison of survivor functions
Cluster randomized design, log-rank test, Freedman method
H0: HR = 1  versus  Ha: HR != 1; HR > 1

Study parameters:

        alpha =    0.0500
        power =    0.8000

Cluster design:

           K1 =        50
           K2 =        50
           M1 =         3
           M2 =         3
           N1 =       150
           N2 =       150
          rho =    0.3000

Number of events and censoring:

            E =       121
           s1 =    0.7000
           s2 =    0.4980
         Pr_E =    0.4010

Estimated effect size and hazard ratio:

        delta =    1.9546  (hazard ratio)
       hratio =    1.9546
```

The minimum detectable value of the hazard ratio is about 1.95.

◁

Compare two survivor functions with clustered data

To compare two survivor functions nonparametrically, we can use the `sts test` command to perform, for example, the log-rank test. `sts test`, however, assumes independent data. We can compare the survivor functions of the two groups semiparametrically while accounting for correlated observations by using a Cox proportional hazards model with clustered standard errors. The Cox model assumes proportional hazards, so testing the equality of two survivor functions reduces to testing that the ratio of the corresponding hazard functions, the hazard ratio, is equal to one. So we can include an identifier of the two groups in the Cox model as the covariate and test the corresponding hazard ratio against one or, equivalently, the log hazard-ratio (coefficient) against zero.

In this section, we briefly demonstrate the `stcox` command with clustered standard errors to compare two groups with survival-time clustered data.

▷ Example 8: Comparing two survivor functions with clustered data using stcox

Consider `eartubes.dta` containing durations of ventilating tubes from Le and Lindgren (1996). To prevent frequent ear infections in infants and young children, a surgical procedure of inserting ventilating tubes in the ears is often recommended. In this study, children with ventilating tubes in both ears were assigned to either a treatment group (receiving a steroid treatment postsurgery) or a control group (receiving no postsurgery intervention). We want to compare the duration of ventilating tubes for children in the two groups. Because durations of ventilating tubes within a child, from the left and right ears, will be correlated, we cluster on the child identifier, `childid`, in our analysis.

We first declare our survival-time data with the `stset` command and then use the `stcox` command with the `vce(cluster childid)` option to fit a Cox model with a treatment variable identifying the two groups as the covariate.

```
. use https://www.stata-press.com/data/r19/eartubes
(Ventilating tubes life data (Le and Lindgren (1996)))
. stset time, failure(status)
Survival-time data settings
         Failure event: status!=0 & status<.
Observed time interval: (0, time]
     Exit on or before: failure
──────────────────────────────────────────────────────────────────────────
        156  total observations
          0  exclusions
──────────────────────────────────────────────────────────────────────────
        156  observations remaining, representing
        144  failures in single-record/single-failure data
    1,373.8  total analysis time at risk and under observation
                                                At risk from t =         0
                                     Earliest observed entry t =         0
                                          Last observed exit t =        33
```

```
. stcox i.treatment, vce(cluster childid)
        Failure _d: status
  Analysis time _t: time
Iteration 0:  Log pseudolikelihood = -590.59936
Iteration 1:  Log pseudolikelihood = -588.43049
Iteration 2:  Log pseudolikelihood = -588.42903
Refining estimates:
Iteration 0:  Log pseudolikelihood = -588.42903
Cox regression with Breslow method for ties
No. of subjects =     156                                 Number of obs =    156
No. of failures =     144
Time at risk    = 1,373.8
                                                          Wald chi2(1)  =   3.55
Log pseudolikelihood = -588.42903                         Prob > chi2   = 0.0594
                              (Std. err. adjusted for 78 clusters in childid)
------------------------------------------------------------------------------
             |               Robust
          _t | Haz. ratio   std. err.      z    P>|z|     [95% conf. interval]
-------------+----------------------------------------------------------------
   treatment |
 treatment   |   .7026351   .1315461    -1.89   0.059      .486821    1.014122
------------------------------------------------------------------------------
```

The estimated hazard ratio is 0.7, and the test of log hazard-ratio (or coefficient) of `treatment` against zero has a p-value $= 0.059$. We do not have sufficient statistical evidence to conclude that the durations of ventilating tubes of the two groups are different, at least at the 5% significance level

Suppose that we want to use the results of this study to design a new study. We want to determine how many children we need in the study. To compute the required number of children or clusters, we need the number of observations per cluster in each group and the estimates of the hazard ratio, of the intraclass correlation, and of the survival rate in the control group at the end of the study.

We can use the estimate of 0.7 for the hazard ratio from `stcox`. Suppose that we are planning to stop our new study after 12 months. We can use the `sts list` command to estimate the control-group survival rate at 12 months based on these data,

```
. sts list if treatment==0, at(12 12)
        Failure _d: status
  Analysis time _t: time
Kaplan-Meier survivor function
                Beg.              Survivor      Std.
    Time       total      Fail     function     error     [95% conf. int.]
-------------------------------------------------------------------------
      12          15        56       0.2035    0.0484     0.1184    0.3049
      12          15         0       0.2035    0.0484     0.1184    0.3049
-------------------------------------------------------------------------
Note: Survivor function is calculated over full data and evaluated at
      indicated times; it is not calculated from aggregates shown at left.
```

The estimated survival rate after 12 months is about 0.2.

To account for clustered data, we need to specify an intraclass correlation in our PSS computations. We can compute a rough estimate of the intraclass correlation assuming a parametric Weibull model as described by Canette (2016). Also see Xie and Waksman (2003) for a nonparametric estimation of the intraclass correlation. For these data, the two methods provide similar values of 0.12 and 0.14, respectively.

We can compute the required number of clusters for a range of intraclass correlation values between 0.04 and 0.2, for example. We specify this range of intraclass correlation values to straddle the rough estimates of 0.12 and 0.14. We specify the control-group survival rate of 0.2 as the command argument, the hazard ratio of 0.7 in the `hratio()` option, and because each child has ventilating tubes in both ears, we specify 2 in the `m1()` and `m2()` options in the `power logrank` command below. We also specify the `table(rho K1)` option to list only the columns that vary in the table.

```
. power logrank 0.2, hratio(0.7) m1(2) m2(2) rho(0.04(0.02)0.2) table(rho K1)

Estimated numbers of clusters for two-sample comparison of survivor functions
Cluster randomized design, log-rank test, Freedman method
H0: HR = 1  versus  Ha: HR != 1
```

rho	K1
.04	89
.06	91
.08	93
.1	94
.12	96
.14	98
.16	100
.18	101
.2	103

The required number of clusters varies between 89 and 103, with the values of 96 and 98 corresponding to our rough estimates of the intraclass correlation of 0.12 and 0.14.

◁

Stored results

`power logrank, cluster` stores the following in `r()`:

Scalars

`r(alpha)`	significance level
`r(power)`	power
`r(beta)`	probability of a type II error
`r(delta)`	effect size
`r(K1)`	number of clusters in the control group
`r(K2)`	number of clusters in the experimental group
`r(kratio)`	ratio of numbers of clusters, `K2/K1`
`r(M1)`	cluster size of the control group
`r(M2)`	cluster size of the experimental group
`r(mratio)`	ratio of cluster sizes, `M2/M1`
`r(N)`	total sample size
`r(N1)`	sample size of the control group
`r(N2)`	sample size of the experimental group
`r(nratio)`	ratio of sample sizes, `N2/N1`
`r(nfractional)`	1 if `nfractional` is specified, 0 otherwise
`r(onesided)`	1 for a one-sided test, 0 otherwise
`r(E)`	total number of events (failures)
`r(hratio)`	hazard ratio
`r(lnhratio)`	log hazard-ratio
`r(s1)`	survival probability in the control group (if specified)
`r(s2)`	survival probability in the experimental group (if specified)
`r(Pr_E)`	probability of an event (failure)
`r(rho)`	intraclass correlation

r(CV_cluster)	coefficient of variation for cluster sizes
r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise
r(init)	initial value for estimated parameter
r(maxiter)	maximum number of iterations
r(iter)	number of iterations performed
r(tolerance)	requested parameter tolerance
r(deltax)	final parameter tolerance achieved
r(ftolerance)	requested distance of the objective function from zero
r(function)	final distance of the objective function from zero
r(converged)	1 if iteration algorithm converged, 0 otherwise
Macros	
r(type)	test
r(method)	logrank
r(design)	CRD
r(test)	Freedman
r(direction)	lower or upper
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats
Matrices	
r(pss_table)	table of results

Methods and formulas

The computation in a CRD uses the Freedman method based on the asymptotic distribution of the log-rank test statistic. See *Methods and formulas* in [PSS-2] **power logrank** for the common notation.

In a CRD, let K_1 and K_2 be the numbers of clusters in the control and experimental groups, respectively, and M_1 and M_2 be the cluster sizes of the control and experimental groups, respectively. For unequal cluster sizes, we assume that the cluster sizes are independent and identically distributed and are small relative to the number of clusters. We have $n_1 = K_1 M_1$ and $n_2 = K_2 M_2$. Let R be the ratio of the numbers of observations, R_k be the ratio of the numbers of clusters, K_2/K_1, and R_m be the ratio of the cluster sizes, M_2/M_1. Let $n = n_1 + n_2$ be the total number of observations, $K = K_1 + K_2$ be the total number of clusters, ρ be the intraclass correlation, CV_{cl} be the coefficient of variation, and $\overline{M}$ be the average cluster size

$$\overline{M} = \frac{M_1 K_1 + M_2 K_2}{K}$$

Define ψ as

$$\psi = (R\Delta + 1)/(\Delta - 1)$$

The total number of events required to be observed in a CRD study to ensure a power of $\pi = 1 - \beta$ of the log-rank test to detect the hazard ratio Δ with significance level α, according to Xie and Waksman (2003), is

$$E = \frac{1}{R}(z_{1-\alpha/k} + z_{1-\beta})^2 \left(\frac{R\Delta + 1}{\Delta - 1}\right)^2 \left[1 + \rho\{\overline{M}(1 + \text{CV}_{\text{cl}}^2) - 1\}\right] \tag{1}$$

where $k = 1$ for the one-sided test and $k = 2$ for the two-sided test.

The required total number of clusters is approximately

$$K = \frac{E}{p_E \overline{M}} \tag{2}$$

where $p_E = 1 - (S_1 + RS_2)/(1 + R)$ is the event probability with S_1 and S_2 defined as the survival probabilities in the control and the experimental groups at the end of the study. Without censoring, $p_E = 1$. Given R_k, we can compute $K_1 = K/(1 + R_k)$ and $K_2 = KR_k/(1 + R_k)$.

Given K_1 and K_2, we can compute the average cluster size $\overline{M}$ as

$$\overline{M} = \frac{1 - \rho}{RKp_E/\{(z_{1-\alpha/k} - z_\beta)\psi\}^2 - \rho(1 + \mathrm{CV}_{\mathrm{cl}}^2)}$$

We can then compute $M_1 = K\overline{M}/(K_1 + R_m K_2)$ and $M_2 = M_1 R_m$.

The power is estimated using the formula

$$\pi = 1 - \beta = \Phi\{|\psi|^{-1}(Rnp_E/[1 + \rho\{\overline{M}(1 + \mathrm{CV}_{\mathrm{cl}}^2) - 1\}])^{1/2} - z_{1-\alpha/k}\}$$

where $\Phi(\cdot)$ is the standard normal cumulative distribution.

Without censoring, the hazard ratio Δ is computed as

$$\Delta = \begin{cases} 1 - \dfrac{R+1}{\sqrt{Rn/(\{z_{1-\alpha/k} - z_\beta\}^2[1+\rho\{\overline{M}(1+\mathrm{CV}_{\mathrm{cl}}^2)-1\}])+R}} & \text{when } \Delta < 1 \\ 1 + \dfrac{R+1}{\sqrt{Rn/(\{z_{1-\alpha/k} - z_\beta\}^2[1+\rho\{\overline{M}(1+\mathrm{CV}_{\mathrm{cl}}^2)-1\}])-R}} & \text{when } \Delta > 1 \end{cases}$$

With censoring, the hazard ratio Δ is computed iteratively based on (1) and (2).

References

Canette, I. 2016. In the spotlight: Intraclass correlations after multilevel survival models. *Stata News*, vol. 31, no. 2. https://www.stata.com/stata-news/news31-2/intraclass-correlations/.

Gallis, J. A., F. Li, H. Yu, and E. L. Turner. 2018. cvcrand and cptest: Commands for efficient design and analysis of cluster randomized trials using constrained randomization and permutation tests. *Stata Journal* 18: 357–378.

Le, C. T., and B. R. Lindgren. 1996. Duration of ventilating tubes: A test for comparing two clustered samples of censored data. *Biometrics* 52: 328–334. https://doi.org/10.2307/2533170.

Xie, T., and J. Waksman. 2003. Design and sample size estimation in clinical trials with clustered survival times as the primary endpoint. *Statistics in Medicine* 22: 2835–2846. https://doi.org/10.1002/sim.1536.

Also see

[PSS-2] **power logrank** — Power analysis for the log-rank test

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-2] **power, graph** — Graph results from the power command

[PSS-2] **power, table** — Produce table of results from the power command

[PSS-5] **Glossary**

[ST] **stcox** — Cox proportional hazards model

[ST] **sts test** — Test equality of survivor functions

[PSS-3] Precision and sample-size analysis

Description Remarks and examples References Also see

Description

Precision and sample-size (PrSS) analysis is essential for designing a statistical study that uses confidence intervals (CIs) for inference. It investigates the optimal allocation of study resources to increase the likelihood of the successful achievement of a study objective. PrSS analysis provides an estimate of the sample size required to achieve the desired precision of a CI in a future study.

For power and sample-size analysis for hypothesis tests, see [PSS-2] **Intro (power)**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the statistical methodology for PrSS and its terminology that will be used throughout the manual. For a list of supported PrSS methods and the description of the software, see [PSS-3] **ciwidth**. For more information about PrSS analysis, see Meeker, Hahn, and Escobar (2017), Dixon and Massey (1983), Zar (2010), and Chow et al. (2018), to name a few.

For power and sample-size analysis for hypothesis tests, see [PSS-2] **Intro (power)**.

Precision and sample-size analysis

Precision and sample-size (PrSS) analysis is a key component in designing a statistical study that uses confidence intervals (CIs) for inference. It investigates the optimal allocation of study resources to increase the likelihood of the successful achievement of a study objective.

How many subjects do we need in a study to achieve a CI of a desired width? A study with too few subjects may have a CI that is too wide to be useful in practice. A study with too many subjects may offer little gain and will thus waste time and resources. Given limited resources, what is the largest CI width that can be expected for an anticipated number of subjects? PrSS analysis helps answer these questions and more. In what follows, when we refer to PrSS analysis, we imply any of these goals.

We consider PrSS analysis of a future study as opposed to a study that has already happened.

In the context of PrSS analysis, a CI is the inferential method used to evaluate research objectives of a study. PrSS analysis is provided for CIs for one mean, the difference between two means, and more. See [PSS-3] **ciwidth** for a full list of methods.

Before we discuss the components of PrSS analysis, let us first revisit the basics of CIs.

Confidence intervals

Consider a study in which the research objective is to estimate a population parameter of interest θ. For example, suppose we want to estimate the mean lifetime of all the bulbs in a factory. The entire population is rarely available in a study. Instead, a sample $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ of a particular size n is randomly selected from a population, and the parameter of interest is estimated from this sample. In our bulb lifetime example, we can measure only the lifetime of a few bulbs to make inferences about the mean lifetime of all the bulbs in the factory. Continuing with our example, we can use the sample mean $\widehat{\theta}$ as an estimate of the mean lifetime of all bulbs. But what if we select a different random sample? Would we obtain the same sample-mean estimate? Given the random nature of the sample, our new estimate will most likely be different. But how much different? In other words, we want to know how precise our sample-mean estimate is given the stochastic nature of a random sample.

In contrast to a single point estimate such as a sample mean, a CI provides an interval estimate for θ that incorporates the precision of the single estimate. Specifically, a $100(1-\alpha)\%$ CI for θ is formed as an interval $[ll(\mathbf{x}), ul(\mathbf{x})]$ such that $\Pr\{ll(\mathbf{x}) \leq \theta \leq ul(\mathbf{x})|\theta\} \geq 1-\alpha$, for $0 \leq \alpha \leq 1$. $1-\alpha$, or expressed as a percentage $100(1-\alpha)\%$, is known as a confidence coefficient or confidence level. $ll = ll(\mathbf{x})$ and $ul = ul(\mathbf{x})$ are confidence limits or confidence bounds, and they depend on the observed data $\mathbf{x}$. A $100(1-\alpha)\%$ CI is constructed such that, in a repeated independent sampling, it is guaranteed to contain the true parameter value $100(1-\alpha)\%$ of the times. Note that for the observed sample $\mathbf{x}$, the probability that the observed CI $[ll, ul]$ contains the true parameter value is either 0 or 1, and we do not know which. Therefore, a CI is often interpreted as a plausible range of values for θ.

CIs can be two sided or one sided. A two-sided CI has finite confidence limits. For a one-sided CI, one of the confidence limits is infinite. Specifically, an upper one-sided CI has $ll = -\infty$ and is of the form $(-\infty, ul]$. A lower one-sided CI has $ul = \infty$ and is of the form $[ll, \infty)$.

CIs are formed around a point estimate of the parameter of interest. For instance, a normal-based $100(1-\alpha)\%$ two-sided CI for one population mean has the form

$$\left[\overline{\theta} - z_{1-\alpha/2}\frac{\sigma}{\sqrt{n}}, \overline{\theta} + z_{1-\alpha/2}\frac{\sigma}{\sqrt{n}}\right]$$

where $\overline{\theta}$ is the sample mean, σ is the population standard deviation, and $z_{1-\alpha/2}$ is the $(1-\alpha/2)$th quantile of the standard normal distribution. The width of the CI can be used to measure the degree of precision of the point estimate.

For a two-sided CI, the width w is defined as the difference between the upper and lower limits, $w = ul - ll$. For an upper one-sided CI, the width is defined as the difference between the upper confidence limit and the point estimate, $w = ul - \widehat{\theta}$. For a lower one-sided CI, the width is defined as the difference between the point estimate and the lower confidence limit, $w = \widehat{\theta} - ll$. For instance, for the above normal-based mean CI, $w = 2z_{\alpha/2}\sigma/\sqrt{n}$ for a two-sided CI and $w = z_{\alpha/2}\sigma/\sqrt{n}$ for a one-sided CI. A half-width $w/2$, also known as a margin of error, is also used as a measure of precision for a symmetric CI. In our PrSS analysis, we will use the CI width as a measure of CI precision and will use the two terms (CI precision and CI width) interchangeably.

Typically, CIs depend on the sample size, confidence level, point estimate of the parameter of interest, and potentially on other method-dependent parameters such as the population standard deviation in our normal-based CI. As the sample size increases, the CI width becomes smaller. The CI becomes narrower and thus has a higher precision. As the confidence level increases, the CI width becomes larger and the CI becomes wider.

To compute CIs in Stata, use `ci`, `mean`, `proportion`, `cc`, to name a few.

There is a strong correspondence between CIs and hypothesis tests. A $100(1-\alpha)\%$ CI can be obtained by inverting the acceptance region of the corresponding level α test. In other words, a $100(1-\alpha)\%$ CI provides the entire range of hypothetical values for a parameter of interest that cannot be rejected by the test at a significance level of α. Despite the strong correspondence between CIs and hypothesis tests, PrSS analysis and PSS analysis will not necessarily lead to the same requirements for the sample size. A hypothesis test compares the parameter of interest with a single value, whereas a CI provides a range of plausible values. Thus, for the same significance level, the sample-size requirements for the CI will generally be larger than for the hypothesis test.

Next, we review concepts specific to PrSS analysis.

Components of PrSS analysis

The main goal of PrSS analysis is to help plan a study such that the chosen CI method has a desired precision. When the CI precision depends on unknown parameters that need to be estimated from the data, its value will vary from one random sample to another. To ensure that a CI in a future sample achieves at least the desired precision, PrSS analysis must account for the sampling variability of the CI precision. To address this, PrSS incorporates the probability of CI width. This is the probability that a future CI will have a width no larger than the target width. In what follows, we will use the CI width to measure the CI precision.

You can think of PrSS analysis analogously to power and sample-size analysis with the confidence level playing the role of the significance level, the probability of CI width playing the role of power, and the CI width playing the role of the effect size.

The goal of PrSS analysis can be achieved in several ways. You can

- compute sample size directly given the specified confidence level, CI width, probability of CI width, and other study parameters;
- evaluate the CI width that can be achieved for a specific sample size given a confidence level, probability of CI width, and other study parameters;
- evaluate the probability of CI width that can be achieved for a specific sample size and CI width given a confidence level and other study parameters; or
- evaluate the sensitivity of the sample-size requirements to various study parameters.

The main components of PrSS analysis are

- confidence level, $100(1-\alpha)$;
- CI width, w;
- probability of CI width, $\Pr(w)$; and
- sample size, n.

Below we describe each of the main components of PrSS analysis in more detail.

Confidence level

We denote the confidence level by $100(1-\alpha)$, where α is between 0 and 1, inclusively. You can think of α as the significance level for the corresponding hypothesis test. Researchers typically set the significance level to a small value such as 0.01 or 0.05 to protect the null hypothesis, which usually

represents a state for which an incorrect decision is more costly. Similarly, the typical confidence levels are 90%, 95%, etc. The most frequently used level is 95%, and it is also the default confidence level for the `ciwidth` command.

CI width

For a two-sided CI, the CI width is the distance from the upper limit to the lower limit. For an upper one-sided CI, the CI width is the distance from the upper limit to the point estimate. For a lower one-sided CI, the CI width is the distance from the point estimate to the lower limit. CI width is commonly used as a measure of the CI precision. The larger the CI width, the wider and less precise the CI.

For some methods, the lower and upper limits of a two-sided CI are symmetric around the point estimate. In this case, the half-width, also called the margin of error, is used to measure the CI precision.

The CI width is an increasing function of the confidence level. So, as we increase the confidence level, the CI becomes wider. The CI width is generally a decreasing function of the sample size.

Probability of CI width

One of the main goals of PrSS analysis is to estimate the required sample size such that, in a new study, the estimated CI will have a certain width. During PrSS analysis, a researcher specifies the desired CI width, and the sample size is computed based on this width for a given confidence level. The CI width often depends on other study parameters such as a standard deviation in a CI for one mean. Unless we know the values of all study parameters that affect the CI width, the width will generally vary from one sample to another.

For example, the population standard deviation is used to define the width of a one-mean CI. When the population standard deviation is unknown, its estimate, a sample standard deviation, is used in the computations. This estimate will be different between different random samples. Thus, the CI width will be different. To obtain a reliable estimate of the required sample size, the sampling variability of the CI width must be accounted for in the computations.

Kupper and Hafner (1989) recommend that PrSS analysis incorporates what we call a probability of CI width, denoted as $\Pr(w)$. This is a probability that the width of a CI in a future study will not exceed a prespecified target value. You can think of the probability of CI width as the analog of power in power and sample-size analysis. As with power, the values of probability of CI width close to 1 such as 0.9 and 0.95 are commonly used in PrSS analysis.

See Kupper and Hafner (1989) and Meeker, Hahn, and Escobar (2017) for more information.

Sample size

Sample size is usually the main component of interest in PrSS analysis. The sample size required to successfully achieve the objective of a study is determined given a specified confidence level, CI width, probability of CI width, and other study parameters. The higher the desired confidence level, the larger the sample size required, with everything else being equal. To achieve a narrower CI, a larger sample size is required, with everything else being equal. The higher the probability of CI width, the larger the required sample size.

When you compute sample size, the actual probability of CI width (probability of CI width corresponding to the obtained sample size) will most likely be different from the probability of CI width you requested because sample size is an integer. In the computation, the resulting fractional sample size that

corresponds to the requested probability of CI width is usually rounded to the nearest integer. To be conservative and to ensure that the actual probability of CI width is at least as large as the requested one, the sample size is rounded up. For sample-size computations that do not use probability of CI width, such as the case of a known population standard deviation for a one-mean CI, the above applies to the specified CI width.

For two-sample designs, fractional sample sizes may arise when you specify a given sample size to compute CI precision. For example, to accommodate an odd total sample size of, say, 51 in a balanced two-sample design, each individual sample size must be 25.5. To be conservative, sample sizes are rounded down on input. The actual sample sizes in our example would be 25 and 25 for a total of 50. See *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for details about sample-size rounding.

For two samples, the allocation of subjects between groups also affects the estimates of CI width and probability of CI width.

One-sided versus two-sided CIs

Two-sided CIs are commonly used in analysis. But there are scenarios where the major interest is only in the lower or upper limit. For example, we may want to know the lower limit for the mean lifetime of bulbs to ensure that, with high certainty, customers will not get bulbs that last less than a year on average, given a 95% confidence level.

For a symmetric two-sided CI, the true value of the parameter of interest has an equal probability of being located outside the lower and upper limits. In this case, by simply changing the confidence level, you can get two-sided confidence limits when computing a one-sided CI and vice versa. For example, the 90% upper confidence limit for a two-sided CI is the upper limit for the 95% one-sided CI.

Sensitivity analysis

Because of limited resources, it may not always be feasible to conduct a study under the original ideal specification. In this case, you may vary study parameters to find an appropriate balance between the desired CI width, probability of CI width, sample size, available resources, and the objective of the study. For example, a researcher may decide to lower the desired confidence level or probability of CI width to decrease the required sample size. In some situations, it may not be possible to reduce the required sample size, in which case more resources must be acquired before the study can be conducted.

CI precision is a function of all the components we described in the previous section—none of the components can be viewed in isolation. For this reason, it is important to perform sensitivity analysis, which investigates CI precision for various specifications of study parameters, and refine the sample-size requirements based on the findings prior to conducting a study. Tables of sample-size values (see [PSS-3] **ciwidth, table**) and graphs of sample-size and CI precision curves (see [PSS-3] **ciwidth, graph**) may be useful for this purpose.

An example of PrSS analysis in Stata

Consider a study of math scores from the SAT exam, similar to the one we discussed in [PSS-2] **Intro (power)**. Suppose that now the investigators would like to measure the average SAT math score. Prior to conducting the study, investigators would like to estimate the sample size required for the two-sided 95% CI for the average SAT score to have a width no larger than 20 points. We can use the `ciwidth onemean` command to estimate the sample size for this study; see [PSS-3] **ciwidth onemean** for more examples.

The investigators do not know the actual value of the standard deviation of the scores, but they do not anticipate it to be larger than the national value of 117 points. As we mentioned in *Probability of CI width*, the CI width of a one-mean CI depends on the standard deviation. Because we assume an unknown standard deviation, the CI width may vary from one study to another. We must account for this sampling variability when estimating the sample size. To ensure that the CI will have a width of at most 20 points, we must also specify the probability of achieving the target CI width.

Below we demonstrate PrSS analysis of this example interactively by typing the commands; see [PSS-3] **GUI (ciwidth)** for point-and-click analysis of this example.

We specify the values of the standard deviation, probability of CI width, and CI width in the corresponding `sd()`, `probwidth()`, and `width()` options. `ciwidth onemean` assumes a 95%-level two-sided CI, so we do not need to specify any additional options.

```
. ciwidth onemean, probwidth(.9) width(20) sd(117)

Performing iteration ...

Estimated sample size for a one-mean CI
Student's t two-sided CI

Study parameters:

        level =     95.00
     Pr_width =    0.9000
        width =   20.0000
           sd =  117.0000

Estimated sample size:

            N =       569
```

We find that a sample of 569 subjects is required to obtain a two-sided 95% CI for the mean SAT score that with a 90% probability will have a width no larger than 20 points.

Suppose investigators can enroll 600 subjects, and they would like to estimate the corresponding probability of obtaining the same target CI width as before. To compute the probability of CI width, we need to specify the sample size, so we modify the command above by replacing the `probwidth(.9)` option with the `n(600)` option.

```
. ciwidth onemean, n(600) width(20) sd(117)

Estimated probability of width for a one-mean CI
Student's t two-sided CI

Study parameters:

        level =     95.00
            N =       600
        width =   20.0000
           sd =  117.0000

Estimated probability of width:

     Pr_width =    0.9887
```

For a larger sample of 600 subjects, the probability of obtaining a CI width no larger than 20 increases to 99%.

Investigators would also like to estimate how narrow a CI width they can obtain with a 90% certainty, given a sample of 600 subjects. To compute the CI width, we specify both the probability of CI width in the `probwidth()` option and the sample size in the `n()` option.

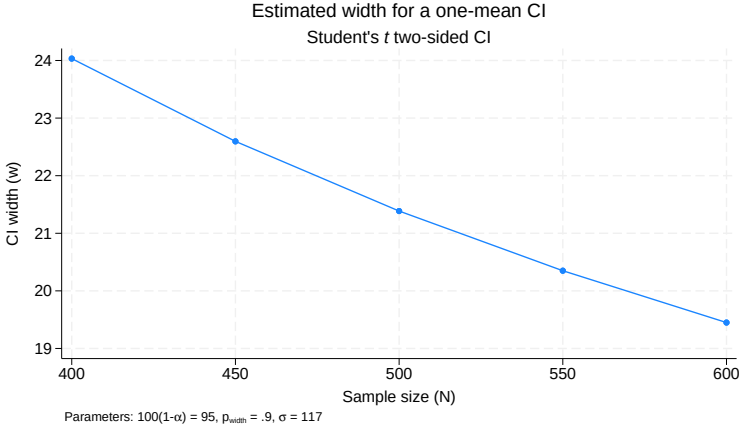
```
. ciwidth onemean, probwidth(.9) n(600) sd(117)
Estimated width for a one-mean CI
Student's t two-sided CI
Study parameters:
        level =     95.00
            N =       600
     Pr_width =    0.9000
           sd =  117.0000
Estimated width:
        width =   19.4499
```

We obtain a CI width of about 19 points.

Continuing their analysis, investigators want to assess the impact of different sample sizes on CI width. They wish to estimate CI width for a range of sample sizes between 400 and 600, in increments of 50.

To display the results graphically, we specify the `graph` option. We abbreviate the range of sample sizes as *numlist* in the `n()` option.

```
. ciwidth onemean, probwidth(.9) n(400(50)600) sd(117) graph
```



The default graph plots the estimated CI width on the y axis and the requested sample size on the x axis. Note how CI width decreases as the sample size increases.

For multiple values of parameters, the results are automatically displayed in a table. However, when we specify the `graph` option, as we did above, the tabular output is suppressed. We can choose to display the results in a table as well by specifying the `table` option:

```
. ciwidth onemean, probwidth(.9) n(400(50)600) sd(117) graph table
  (output omitted)
```

The `ciwidth` command performs PrSS analysis for a number of CIs; see [PSS-3] **ciwidth** and method-specific entries for more examples. You can also add your own methods to the `ciwidth` command as described in [PSS-3] ***ciwidth usermethod***.

References

Chow, S.-C., J. Shao, H. Wang, and Y. Lokhnygina. 2018. *Sample Size Calculations in Clinical Research*. 3rd ed. Boca Raton, FL: CRC Press.

Dixon, W. J., and F. J. Massey, Jr. 1983. *Introduction to Statistical Analysis*. 4th ed. New York: McGraw–Hill.

Kupper, L. L., and K. B. Hafner. 1989. How appropriate are popular sample size formulas? *American Statistician* 43: 101–105. https://doi.org/10.2307/2684511.

Meeker, W. Q., G. J. Hahn, and L. A. Escobar. 2017. *Statistical Intervals: A Guide for Practitioners and Researchers*. 2nd ed. Hoboken, NJ: Wiley.

Zar, J. H. 2010. *Biostatistical Analysis*. 5th ed. Upper Saddle River, NJ: Pearson.

Also see

[PSS-3] **ciwidth** — Precision and sample-size analysis for CIs

[PSS-3] **ciwidth, table** — Produce table of results from the ciwidth command

[PSS-3] **ciwidth, graph** — Graph results from the ciwidth command

[PSS-3] **GUI (ciwidth)** — Graphical user interface for precision and sample-size analysis

[PSS-2] **Intro (power)** — Introduction to power and sample-size analysis for hypothesis tests

[PSS-4] **Unbalanced designs** — Specifications for unbalanced designs

[PSS-5] **Glossary**

GUI (ciwidth) — Graphical user interface for precision and sample-size analysis

Description Menu Remarks and examples Also see

Description

This entry describes the graphical user interface (GUI) for the `ciwidth` command. See [PSS-3] **ciwidth** for a general introduction to the `ciwidth` command.

Menu

Statistics > Power, precision, and sample size

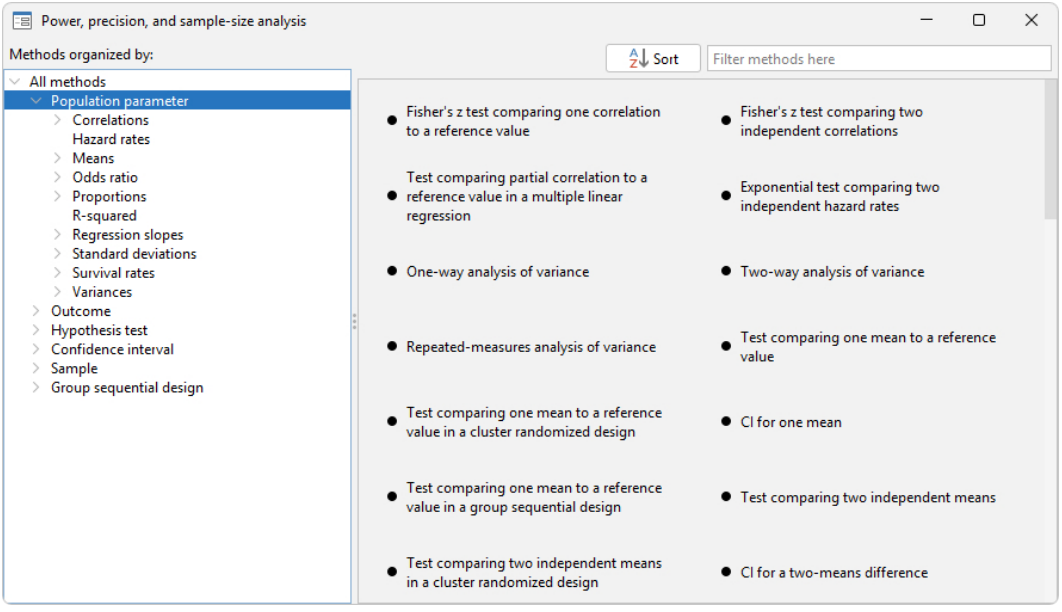
Remarks and examples

Remarks are presented under the following headings:

PSS Control Panel

You can perform PrSS analysis interactively by typing the `ciwidth` command or by using a point-and-click GUI available via the PSS Control Panel.

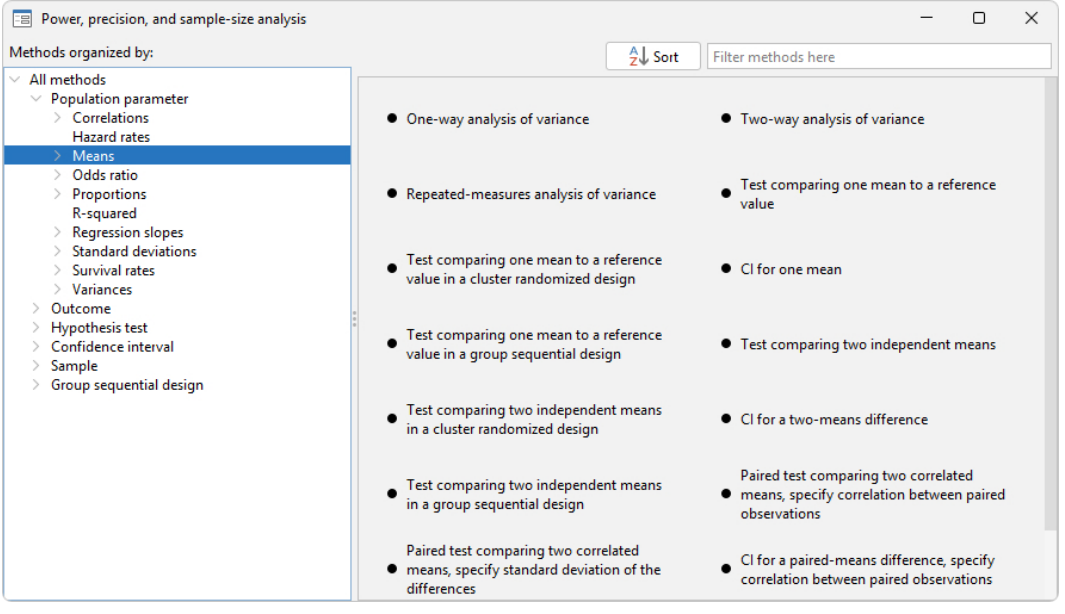
The PSS Control Panel can be accessed by selecting **Statistics** > **Power, precision, and sample size** from the Stata menu. It includes a tree-view organization of the PSS, PrSS, and group sequential design methods.



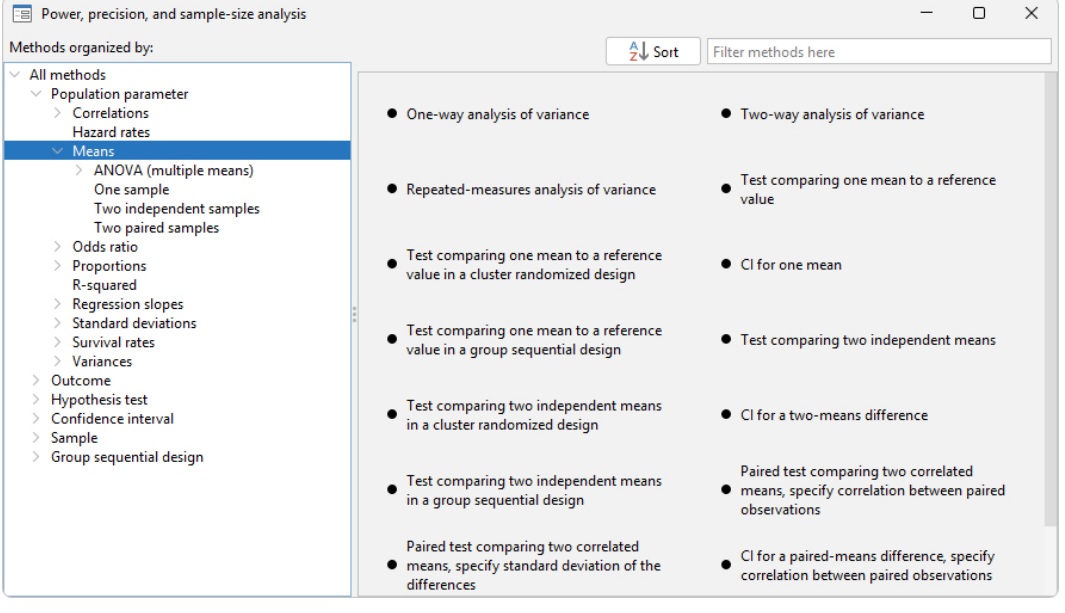
The left pane organizes the methods, and the right pane displays the methods corresponding to the selection in the left pane. On the left, the methods are organized by the type of population parameter, such as mean or proportion; the type of outcome, such as continuous or binary; the type of analysis, such as hypothesis test or confidence interval; and the type of sample, such as one sample or two samples. You click on one of the methods shown in the right pane to launch the dialog box for that method.

By default, methods are organized by **Population parameter**. We can find the method we want to use by looking for it in the right pane, or we can narrow down the type of method we are looking for by selecting one of the expanded categories in the left pane.

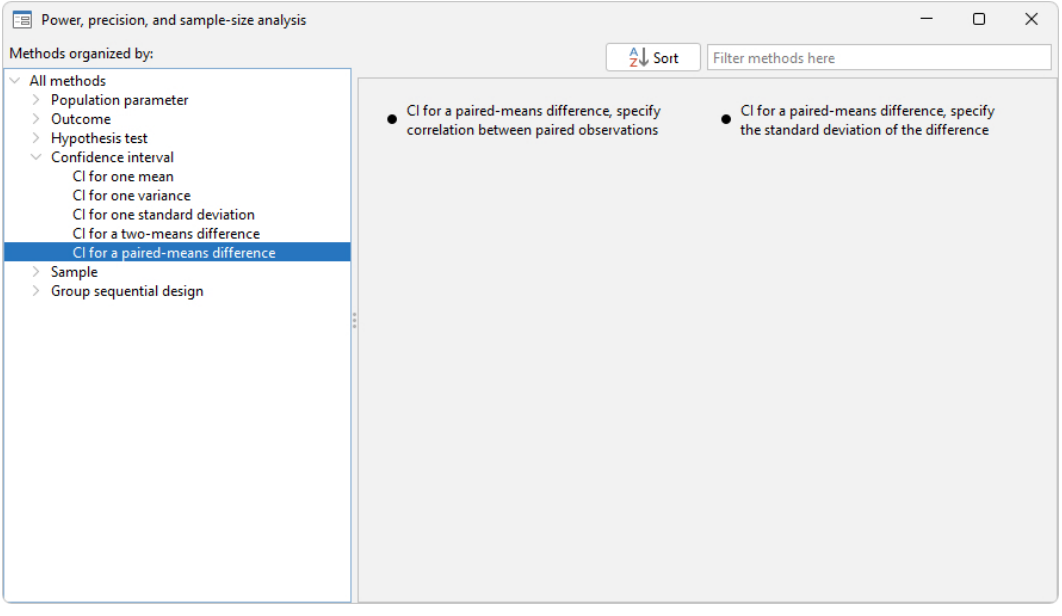
For example, if we are interested in means, we can click on **Means** within **Population parameter** to see all methods for means in the right pane.



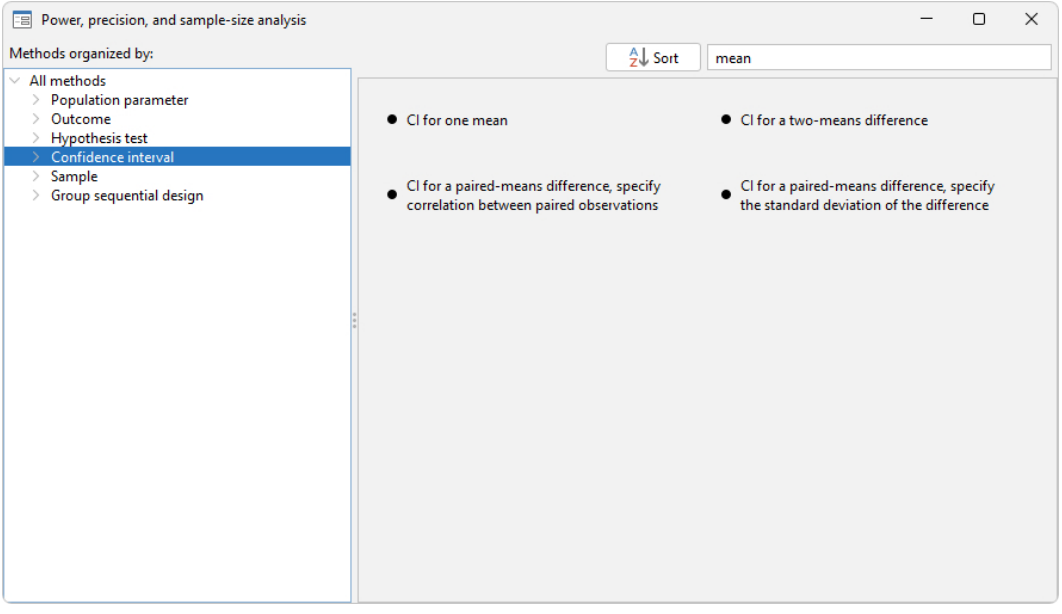
We can expand **Means** to further narrow down the choices by clicking on the symbol to the left of **Means**.



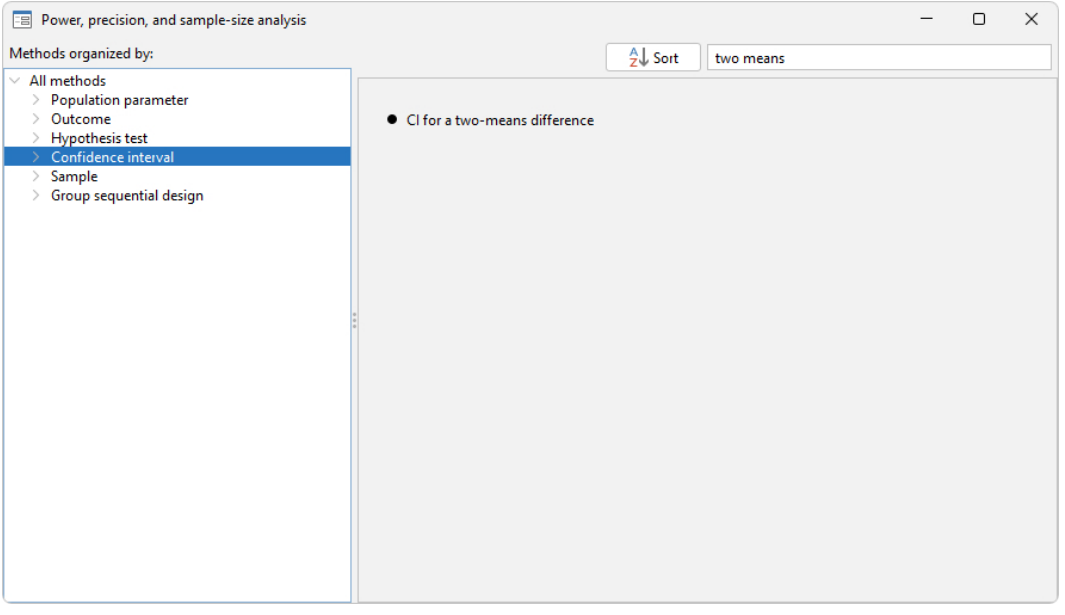
Or we can choose a method by the type of analysis by expanding **Confidence interval** and selecting, for example, **CI for a paired-means difference**:



We can also locate methods by searching the titles of methods. You specify the search string of interest in the *Filter* box at the top right of the PSS Control Panel. For example, if we type "mean" in the *Filter* box while keeping the focus on **Confidence interval**, only CI methods with a title containing "mean" will be listed in the right pane.



We can specify multiple words in the *Filter* box, and only methods with all the specified words in their titles will appear. For example, if we type "two means", only methods with the words "two" and "means" in their titles will be shown:

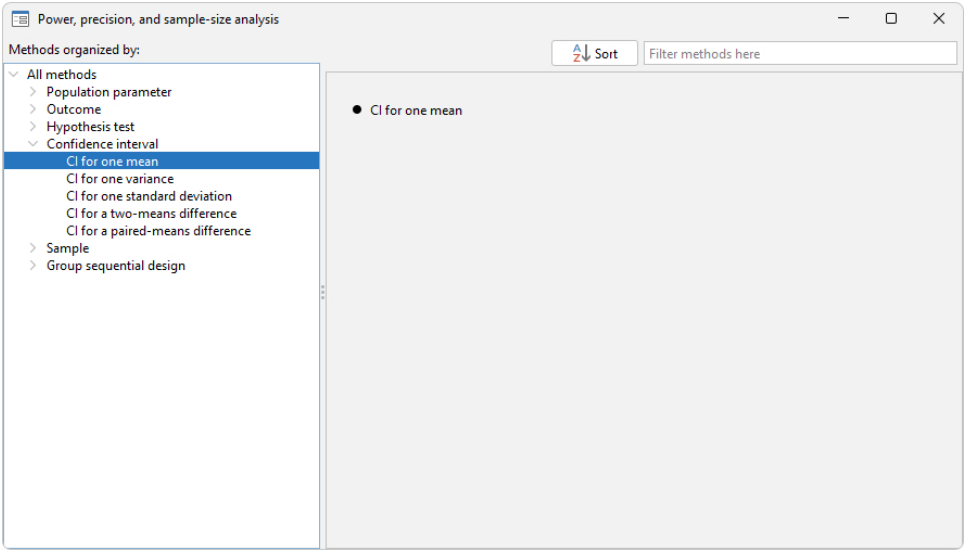


The search is performed within the group of methods selected by the choice in the left pane. In the above example, the search was done within **Confidence interval**. When you search all methods, whether you select **Population parameter**, **Outcome**, or **Sample** in the left pane, the same set of methods appears in the right pane but in the order determined by the selected category.

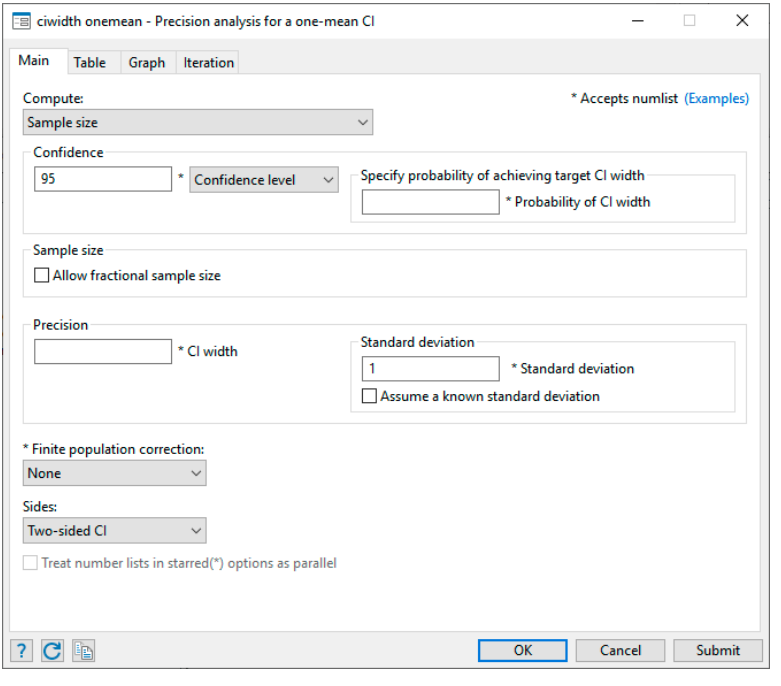
Example using PSS Control Panel

In *An example of PrSS analysis in Stata* of [PSS-3] **Intro (ciwidth)**, we performed PrSS analysis interactively by typing commands. We replicate the analysis by using the PSS Control Panel and dialog boxes.

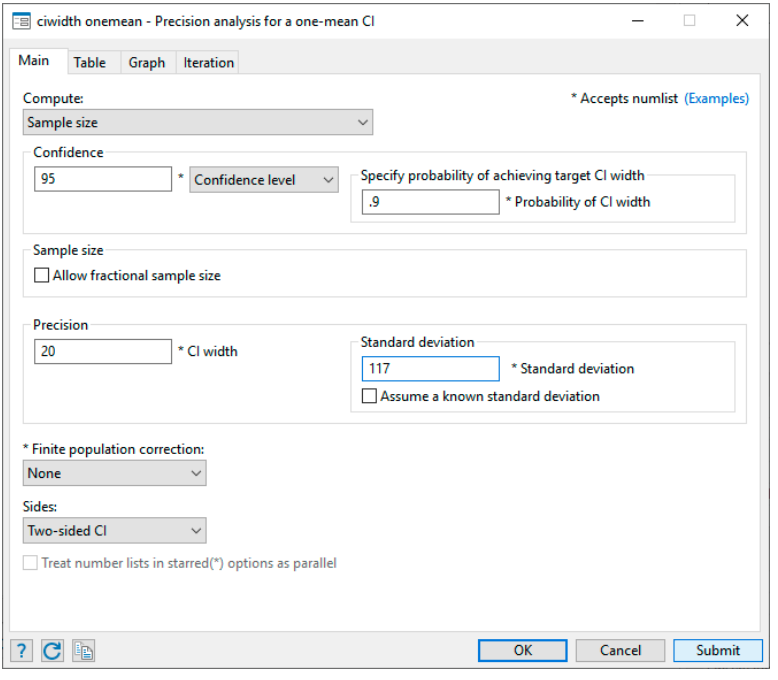
We first launch the PSS Control Panel from the **Statistics** > **Power, precision, and sample size** menu. We then narrow down to the desired dialog box by first choosing **Confidence interval** in the left pane and then choosing **CI for one mean**. In the right pane, we see the corresponding CI method.



We invoke the dialog box by clicking on the corresponding method title in the right pane. The following appears:



Following the example from *An example of PrSS analysis in Stata* in [PSS-3] **Intro (ciwidth)**, we now compute sample size. The first step is to choose which parameter to compute. The *Compute* drop-down box specifies `Sample size`, so we leave it unchanged. The next step is to specify the confidence level. The default confidence level is already set to our desired value of 95%, so we leave it unchanged. We fill the *Probability of CI width* box with the value 0.9 and the *CI width* box with the value 20. We then specify a standard deviation of 117. We leave everything else unchanged and click on the **Submit** button to obtain results.

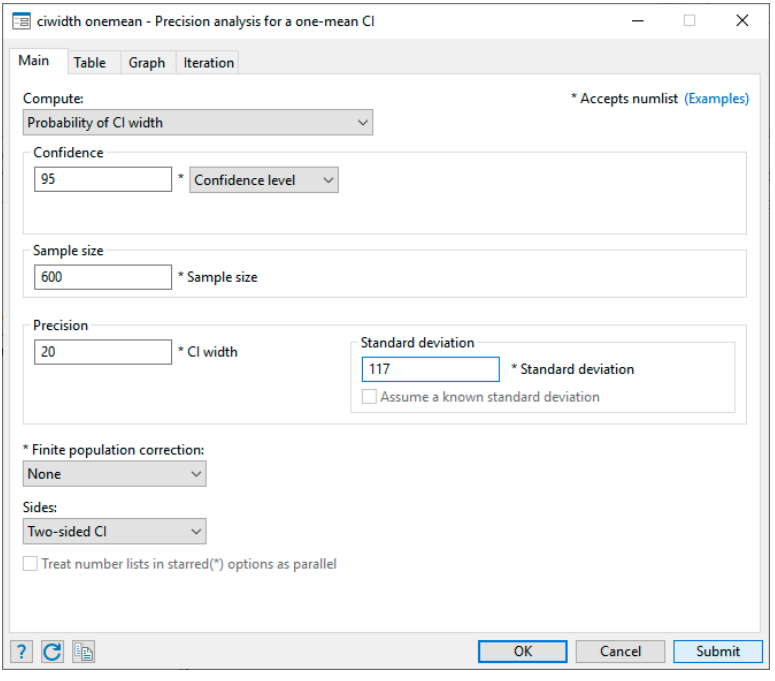


The following command is displayed in the Results window and executed:

```
. ciwidth onemean, probwidth(.9) width(20) sd(117)
Performing iteration ...
Estimated sample size for a one-mean CI
Student's t two-sided CI
Study parameters:
        level =     95.00
     Pr_width =    0.9000
        width =   20.0000
           sd =  117.0000
Estimated sample size:
            N =       569
```

We can verify that the command and results are exactly the same as what we specified in *An example of PrSS analysis in Stata* of [PSS-3] **Intro (ciwidth)**.

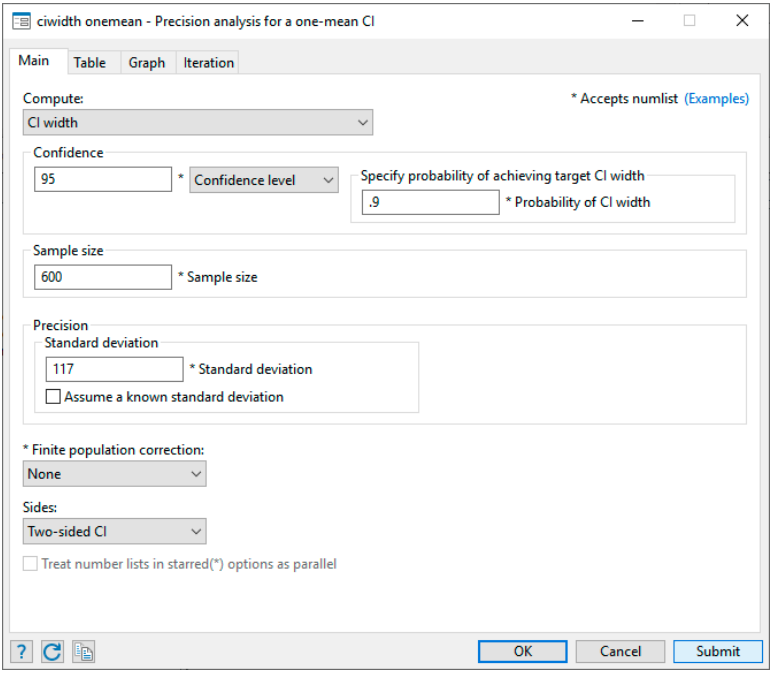
Continuing our PrSS analysis, we can enroll 600 subjects and would like to estimate the corresponding probability of CI width given the same CI width. We return to the dialog box and select `Probability of CI width` under *Compute*. To compute the probability of CI width, we need to specify the sample size of 600 and leave the other specifications unchanged.



The following command is issued after we click on the **Submit** button:

```
. ciwidth onemean, n(600) width(20) sd(117)

Estimated probability of width for a one-mean CI
Student's t two-sided CI

Study parameters:

        level =     95.00
            N =       600
        width =   20.0000
           sd =  117.0000

Estimated probability of width:

     Pr_width =    0.9887
```

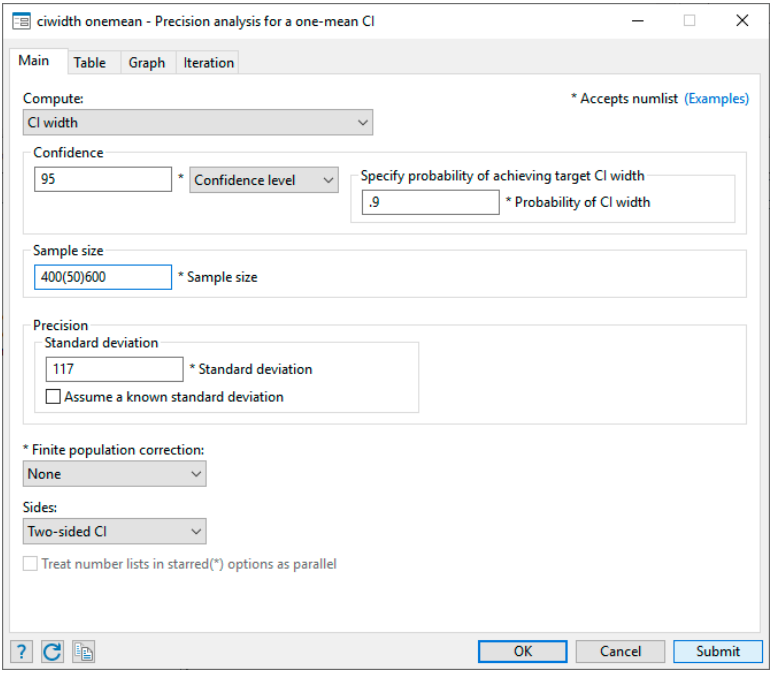
Instead of the probability of CI width, we can also compute the CI width given the same sample size of 600 and the earlier probability of CI width of 0.9. We return to our dialog box and simply select `CI width` under **Compute**.



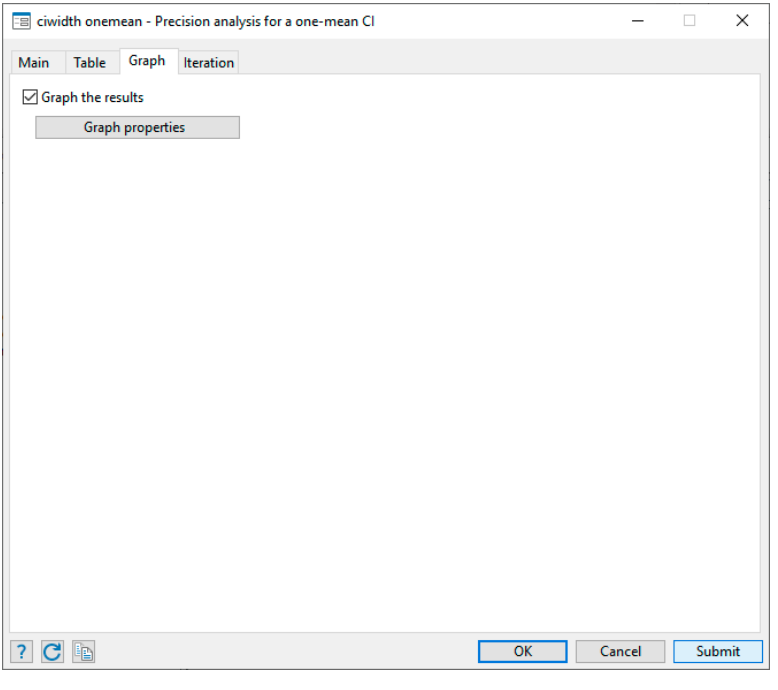
The following command is issued after we click on the **Submit** button:

```
. ciwidth onemean, probwidth(.9) n(600) sd(117)

Estimated width for a one-mean CI
Student's t two-sided CI

Study parameters:

        level =     95.00
            N =       600
     Pr_width =    0.9000
           sd =  117.0000

Estimated width:

        width =   19.4499
```

To produce the graph from *An example of PrSS analysis in Stata*, we first select `CI width` under *Compute*. Then we specify the *numlist* for sample size in the respective box:

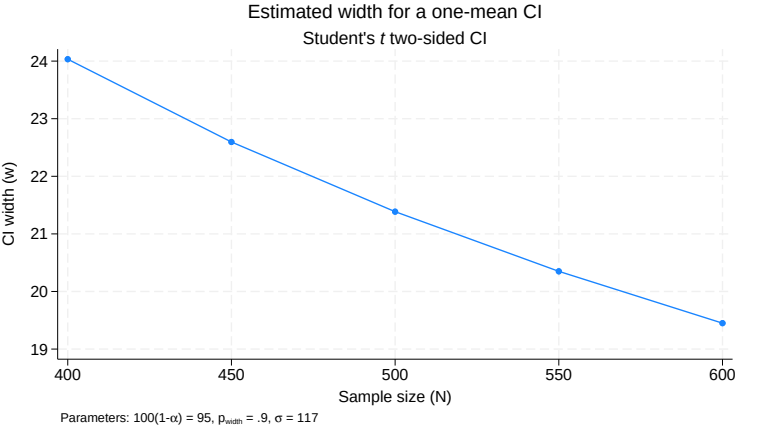


Then we select the **Graph** tab and check the *Graph the results* box:



We click on the **Submit** button and obtain the following command and graph:

```
. ciwidth onemean, probwidth(.9) n(400(50)600) sd(117) graph
```



Also see

[PSS-3] **ciwidth** — Precision and sample-size analysis for CIs

[PSS-3] **Intro (ciwidth)** — Introduction to precision and sample-size analysis for confidence intervals

[PSS-5] **Glossary**

ciwidth — Precision and sample-size analysis for CIs

Description

The `ciwidth` command performs precision and sample-size analysis (PrSS) for CIs. You can compute sample size given CI width (or precision) and probability of CI width. Alternatively, you can compute CI width given sample size and probability of CI width. You can also compute probability of CI width given sample size and CI width. You can display results in a table ([PSS-3] **ciwidth, table**) and on a graph ([PSS-3] **ciwidth, graph**).

For power and sample-size analysis for hypothesis tests, see [PSS-2] **power**.

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
ciwidth method ..., width(numlist) probwidth(numlist) [ciwidth_options]
```

Compute CI width

```
ciwidth method ..., probwidth(numlist) n(numlist) [ciwidth_options]
```

Compute probability of CI width

```
ciwidth method ..., width(numlist) n(numlist) [ciwidth_options]
```

method	Description
One sample	
`onemean`	CI for one mean
`onevariance`	CI for one variance
Two independent samples	
`twomeans`	CI for comparing two means from independent samples
Two paired samples	
`pairedmeans`	CI for comparing two means from paired samples
User-defined methods	
usermethod	Add your own method to `ciwidth`

ciwidth_options	Description
Main	
* `level(`*numlist*`)`	confidence level; default is `level(95)`
* `alpha(`*numlist*`)`	significance level; default is `alpha(0.05)`
* `probwidth(`*numlist*`)`	probability of CI width; required to compute sample size and CI width
* `width(`*numlist*`)`	CI width; required to compute sample size and probability of CI width
* `n(`*numlist*`)`	total sample size; required to compute CI width and probability of CI width
* `n1(`*numlist*`)`	sample size of the control group
* `n2(`*numlist*`)`	sample size of the experimental group
* `nratio(`*numlist*`)`	ratio of sample sizes, `N2/N1`; default is `nratio(1)`, meaning equal group sizes
`compute(N1 | N2)`	solve for `N1` given `N2` or for `N2` given `N1`
`nfractional`	allow fractional sample sizes
`lower`	lower one-sided CI; default is two-sided CI
`upper`	upper one-sided CI; default is two-sided CI
`onesided`	synonym for option `upper`
`parallel`	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[`no`]`table`[(*tablespec*)]	suppress table or display results as a table; see [PSS-3] **ciwidth, table**
`saving(`*filename* [`, replace`]`)`	save the table data to *filename*; use `replace` to overwrite existing *filename*
Graph	
`graph`[(*graphopts*)]	graph results; see [PSS-3] **ciwidth, graph**
Iteration	
`init(#)`	initial value for sample size; default is to use a closed-form normal approximation
`iterate(#)`	maximum number of iterations; default is `iterate(500)`
`tolerance(#)`	parameter tolerance; default is `tolerance(1e-12)`
`ftolerance(#)`	function tolerance; default is `ftolerance(1e-12)`
[`no`]`log`	suppress or display iteration log
[`no`]`dots`	suppress or display iterations as dots
`notitle`	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the `parallel` option.

Options `n1()`, `n2()`, `nratio()`, and `compute()` are available only for two-independent-samples methods.

Iteration options are available only with computations requiring iteration.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`notitle` does not appear in the dialog box.

Options

Main

`level(`*numlist*`)` specifies the confidence level, as a percentage, for CIs. The default is `level(95)` or as set by `set level`; see [R] **level**. If `alpha()` is specified, this value is set to be $100(1 - \texttt{alpha()})$. Only one of `level()` or `alpha()` may be specified.

`alpha(`*numlist*`)` sets the significance level. Only one of `level()` or `alpha()` may be specified.

`probwidth(`*numlist*`)` specifies the probability of obtaining a CI with the width no larger than a target CI width. The target CI width is either computed by the command or specified in option `width()`. This option is required to compute sample size and CI width.

`width(`*numlist*`)` specifies the target CI width, which represents the precision of the CI. This option is required to compute sample size and probability of CI width. For a two-sided CI, CI width is the distance between the upper and lower limits. For a one-sided CI, it is the distance from the limit to the estimate of the parameter of interest, such as a sample mean.

`n(`*numlist*`)` specifies the total number of subjects in the study to be used to compute the CI width and probability of CI width.

`n1(`*numlist*`)` specifies the number of subjects in the control group to be used to compute the CI width and probability of CI width.

`n2(`*numlist*`)` specifies the number of subjects in the experimental group to be used to compute the CI width and probability of CI width.

`nratio(`*numlist*`)` specifies the sample-size ratio of the experimental group relative to the control group, `N2/N1`, for two-sample CIs. The default is `nratio(1)`, meaning equal allocation between the two groups.

`compute(N1 | N2)` requests that the `ciwidth` command compute one of the group sample sizes given the other one, instead of the total sample size, for two-sample CIs. To compute the control-group sample size, you must specify `compute(N1)` and the experimental-group sample size in `n2()`. Alternatively, to compute the experimental-group sample size, you must specify `compute(N2)` and the control-group sample size in `n1()`.

`nfractional` specifies that fractional sample sizes be allowed. When this option is specified, fractional sample sizes are used in the intermediate computations and are also displayed in the output.

Also see the description and the use of options `n()`, `n1()`, `n2()`, `nratio()`, and `nfractional` for two-sample CIs in *Fractional sample sizes* of [PSS-4] **Unbalanced designs**.

`lower` specifies a lower one-sided CI and may not be combined with option `upper`. The default is a two-sided CI.

`upper` specifies an upper one-sided CI and may not be combined with option `lower`. The default is a two-sided CI.

`onesided` is a synonym for `upper`, which specifies an upper one-sided CI.

`parallel` requests that computations be performed in parallel over the lists of numbers specified for at least two study parameters as command arguments, starred options allowing *numlist*, or both. That is, when `parallel` is specified, the first computation uses the first value from each list of numbers, the second computation uses the second value, and so on. If the specified number lists are of different sizes, the last value in each of the shorter lists will be used in the remaining computations. By default, results are computed over all combinations of the number lists.

For example, let a_1 and a_2 be the list of values for one study parameter, and let b_1 and b_2 be the list of values for another study parameter. By default, `ciwidth` will compute results for all possible combinations of the two values in the two study parameters: (a_1, b_1), (a_1, b_2), (a_2, b_1), and (a_2, b_2). If `parallel` is specified, `ciwidth` will compute results for only two combinations: (a_1, b_1) and (a_2, b_2).

Table

`notable`, `table`, and `table()` control whether or not results are displayed in a tabular format. `table` is implied if any number list contains more than one element. `notable` is implied with graphical output—when either the `graph` or the `graph()` option is specified. `table()` is used to produce custom tables. See [PSS-3] **ciwidth, table** for details.

`saving(`*filename* [`, replace`]`)` creates a Stata data file (`.dta` file) containing the table values with variable names corresponding to the displayed *columns*. `replace` specifies that *filename* be overwritten if it exists. `saving()` is only appropriate with tabular output.

Graph

`graph` and `graph()` produce graphical output; see [PSS-3] **ciwidth, graph** for details.

The following options control an iteration procedure used by the `ciwidth` command for solving nonlinear equations.

Iteration

`init(#)` specifies an initial value for the sample size when iteration is used to compute the sample size. The default is to use a closed-form normal approximation to compute an initial sample size.

`iterate(#)` specifies the maximum number of iterations for the Newton method. The default is `iterate(500)`.

`tolerance(#)` specifies the tolerance used to determine whether successive parameter estimates have converged. The default is `tolerance(1e-12)`. See *Convergence criteria* in [M-5] **solvenl( )** for details.

`ftolerance(#)` specifies the tolerance used to determine whether the proposed solution of a nonlinear equation is sufficiently close to 0 based on the squared Euclidean distance. The default is `ftolerance(1e-12)`. See *Convergence criteria* in [M-5] **solvenl( )** for details.

`log` and `nolog` specify whether an iteration log is to be displayed. The iteration log is suppressed by default. Only one of `log`, `nolog`, `dots`, or `nodots` may be specified.

`dots` and `nodots` specify whether a dot is to be displayed for each iteration. The iteration dots are suppressed by default. Only one of `dots`, `nodots`, `log`, or `nolog` may be specified.

The following option is available with `ciwidth` but is not shown in the dialog box:

`notitle` prevents the command title from displaying.

Remarks and examples

Remarks are presented under the following headings:

This section describes how to perform PrSS analysis for CIs using the `ciwidth` command. For a software-free introduction to PrSS analysis, see [PSS-3] **Intro (ciwidth)**.

Using the ciwidth command

The `ciwidth` command computes sample size, CI width, and probability of CI width for various CIs. You can also add your own methods to the `ciwidth` command as described in [PSS-3] ***ciwidth usermethod***.

By default, all computations are performed for a two-sided CI, and the confidence level is set to 95%. You may change the confidence level by specifying the `level()` option. Alternatively, you can specify the significance level in the `alpha()` option. You can specify the `upper` and `lower` options to request upper and lower one-sided CIs.

To compute sample size, you must specify the CI width in the `width()` option and the probability of CI width in the `probwidth()` option. To compute CI width, you must specify the sample size in the `n()` option and the probability of CI width in the `probwidth()` option. You can also compute the probability of CI width given the sample size in `n()` and CI width in `width()`. For some CIs, you must also specify target values for parameters of interest as command arguments, such as a target variance for `ciwidth onevariance`.

The `probwidth()` option is analogous to the `power()` option in power and sample-size analysis. It accounts for the sampling variability of the CI width. For example, for CIs for means, the CI width depends on the variance, which is commonly estimated from the sample and may vary from one sample to another. To limit the impact of this sample-to-sample variability on the CI precision, you can use the `probwidth()` option to specify the probability that the width of a future CI will not exceed a target value. Without this adjustment, the results are generally conditional on the future data having the same variance as the one used for PrSS analysis and may underestimate the estimated sample size and CI width.

For CIs comparing two independent samples, you can compute one of the group sizes given the other one instead of the total sample size. In this case, you must specify the label of the group size you want to compute in the `compute()` option and the value of the other group size in the respective `n#()` option. For example, if we wanted to find the size of the second group given the size of the first group, we would specify the combination of options `compute(N2)` and `n1(#)`.

A balanced design is assumed by default for two-independent-samples CIs, but you can request an unbalanced design. For example, you can specify the allocation ratio n_2/n_1 between the two groups in the `nratio()` option or the individual group sizes in the `n1()` and `n2()` options. See [PSS-4] **Unbalanced designs** for more details about various ways of specifying an unbalanced design.

For sample-size determination, the reported integer sample sizes may not correspond exactly to the specified CI width and probability of CI width because of rounding. To obtain conservative results, the ciwidth command rounds up the sample size to the nearest integer so that the corresponding CI width is no larger than the requested one and the probability of CI width is no smaller than the requested one. You can specify the nfractional option to obtain the corresponding fractional sample size.

Some of ciwidth's computations require iteration. The defaults chosen for the iteration procedure should be sufficient for most situations. In a rare situation when you may want to modify the defaults, the ciwidth command provides options to control the iteration procedure. The most commonly used is the init() option for supplying an initial value of the estimated parameter. This option can be useful in situations where the computations are sensitive to the initial values. If you are performing computations for many combinations of various study parameters, you may consider reducing the default maximum number of iterations of 500 in the iterate() option so that the command is not spending time on calculations in difficult-to-compute regions of the parameter space. By default, ciwidth suppresses the iteration log. If desired, you can specify the log option to display the iteration log or the dots option to display iterations as dots to monitor the progress of the iteration procedure.

The ciwidth command can produce results for one study scenario or for multiple study scenarios when multiple values of the parameters are specified; see *Specifying multiple values of study parameters* below for details.

For a single result, ciwidth displays results as text. For multiple results or if the table option is specified, ciwidth displays results in a table. You can also display multiple results on a graph by specifying the graph option. Graphical output suppresses the table of the results; use the table option to also see the tabular output. You can customize the default tables and graphs by specifying suboptions within the respective options table() and graph(); see [PSS-3] **ciwidth, table** and [PSS-3] **ciwidth, graph** for details.

You can also save the tabular output to a Stata dataset by using the saving() option.

Specifying multiple values of study parameters

The ciwidth command can produce results for one study scenario or for multiple study scenarios when multiple values of the parameters are supplied to the supported options. The options that support multiple values specified as *numlist* are marked with a star in the syntax diagram.

For example, the n() option supports multiple values. You can specify multiple sample sizes as individual values, n(100 150 200), or as a range of values, n(100(50)200); see [U] **11.1.8 numlist** for other specifications.

In addition to options, you may specify multiple values of command arguments, values specified after the command name. For example, let $\#_1$ be the command argument in

```
. ciwidth onevariance #1, ...
```

If we want to specify multiple values for the command arguments, we must enclose these values in parentheses. For example,

```
. ciwidth onevariance (1 1.5), ...
```

or, more generally,

```
. ciwidth onevariance (numlist), ...
```

When multiple values are specified in multiple options or for multiple command arguments, the ciwidth command computes results for all possible combinations formed by the values from every option and command argument. In some cases, you may want to compute results in parallel for specific sets of values of the specified parameters. To request this, you can specify the parallel option. If the specified number lists are of varying sizes, *numlist* with the maximum size determines the number of final results produced by ciwidth. The last value from *numlist* of smaller sizes will be used in the subsequent computations.

For example,

```
. ciwidth onevariance (1 1.5), n(100 200) ...
```

is equivalent to

```
. ciwidth onevariance 1, n(100) ...
. ciwidth onevariance 1.5, n(100) ...
. ciwidth onevariance 1, n(200) ...
. ciwidth onevariance 1.5, n(200) ...
```

When the parallel option is specified,

```
. ciwidth onevariance (1 1.5), n(100 200) parallel ...
```

is equivalent to

```
. ciwidth onevariance 1, n(100) ...
. ciwidth onevariance 1.5, n(200) ...
```

When the parallel option is specified and *numlist* is of different sizes, the last value of the shorter *numlist* is used in the subsequent computations. For example,

```
. ciwidth onevariance (1 1.5 2), n(100 200) parallel ...
```

is equivalent to

```
. ciwidth onevariance 1, n(100) ...
. ciwidth onevariance 1.5, n(200) ...
. ciwidth onevariance 2, n(200) ...
```

PrSS analysis for CIs for one population parameter

The ciwidth command provides PrSS analysis for two types of one-sample CIs. ciwidth onemean performs PrSS analysis for a CI for one population mean, and ciwidth onevariance performs PrSS analysis for a CI for one population variance.

ciwidth onemean provides PrSS computations for a one-mean CI. You can perform computations assuming a known or unknown population standard deviation and adjust the results for a finite population. See [PSS-3] **ciwidth onemean**.

ciwidth onevariance provides PrSS computations for a one-variance CI. You can perform computations in the variance or standard deviation metric. See [PSS-3] **ciwidth onevariance**.

Below we show a quick example of PrSS analysis for a one-mean CI. See the individual entries for more examples.

▷ Example 1: PrSS analysis for a one-mean CI

A group of pediatricians would like to study the exposure of infants to television. They are interested in estimating a CI for the average number of hours of television watched by infants per day. Before conducting a study, the pediatricians would like to determine how many infants they need to enroll in the study so that the CI width is not too large. They hypothesize that the average number of hours that infants spend watching television has a standard deviation of 0.8 hours. The group of pediatricians would like to compute the sample size required to produce a two-sided 95% CI with a width of 0.5 hours given this study parameter.

We use `ciwidth onemean` to compute the required sample size. We specify the standard deviation of 0.8 in the `sd()` option and the CI width of 0.5 in the `width()` option. However, this CI width can vary from sample to sample because samples tend to have different variances. To adjust the results for the sampling variability, we can use the `probwidth()` option to specify the probability that the width of the CI does not exceed our target value.

To be 96% certain that the CI width in a future study will be no larger than 0.5 hours, we specify `probwidth(0.96)`. By default, the confidence level is 95%, so we do not need to specify the `level(95)` option.

```
. ciwidth onemean, sd(0.8) probwidth(0.96) width(0.5)

Performing iteration ...

Estimated sample size for a one-mean CI
Student's t two-sided CI

Study parameters:

        level =     95.00
     Pr_width =    0.9600
        width =    0.5000
           sd =    0.8000

Estimated sample size:

            N =        56
```

The pediatricians need to enroll 56 infants in the study to be 96% certain that the 95% CI will be no wider than 0.5 for the average number of hours of television that infants watch per day.

All `ciwidth` commands have a similar output format. Information about the type of CI is displayed first. The input and implied values of the study parameters are displayed next under `Study parameters`. The estimated parameters, in this case the sample size, are displayed last.

Now suppose that we come across a pilot study reporting that infants watch an average of 2.5 hours of television per day, with a standard deviation of 0.8 hours. Now that the value of our standard deviation is known, rather than hypothesized, we can specify the `knownsd` option. With this option specified, `ciwidth onemean` will perform computations for a normal two-sided CI.

```
. ciwidth onemean, sd(0.8) width(0.5) knownsd

Estimated sample size for a one-mean CI
Normal two-sided CI

Study parameters:

        level =     95.00
        width =    0.5000
           sd =    0.8000

Estimated sample size:

            N =        40
```

Now that our standard deviation is known, the required sample size is smaller.

Let's suppose that we anticipate on having 50 infants in the study, and we want to see how the probability of obtaining the target CI width of 0.5 changes with this sample size. We now specify the `n(50)` option in addition to the CI width and standard deviation.

```
. ciwidth onemean, sd(0.8) n(50) width(0.5)

Estimated probability of width for a one-mean CI
Student's t two-sided CI

Study parameters:

        level =     95.00
            N =        50
        width =    0.5000
           sd =    0.8000

Estimated probability of width:

     Pr_width =    0.8501
```

With a sample of 50 infants, we are 85% certain that the CI width in a future study will be no larger than 0.5 hours, given a standard deviation of 0.8 for the average number of hours of television that infants watch per day. As expected, a smaller sample size leads to less certainty about the CI width.

◁

PrSS analysis for CIs comparing two independent samples

The `ciwidth twomeans` command provides PrSS analysis for CIs comparing the means from two independent samples. You can perform computations assuming known or unknown and equal or unequal population standard deviations. See [PSS-3] **ciwidth twomeans**.

Below we show a quick example of PrSS analysis for the CI comparing two means. See the individual entry for more examples.

▷ Example 2: PrSS analysis for a two-means difference CI

A pharmaceutical company would like to conduct a study to compare a new weight-loss drug with an older drug. Investigators are planning to compare the average weight loss in the two drugs by constructing a CI for the difference between the means of the two groups. The average weight loss for people taking the old drug for 3 months has a standard deviation of 5.5 pounds. The new drug is expected to produce greater weight loss, with a smaller standard deviation of 5 pounds for the same period of time. Investigators want to find out how many subjects they need to recruit to obtain a two-sided 95% CI for a difference between the two means with a target width of 6 pounds.

We use `ciwidth twomeans` to perform the PrSS analysis. We specify the control- and experimental-group standard deviations in the respective `sd1()` and `sd2()` options, along with the `knownsds` option. We also specify the target CI width of 6 pounds in the `width()` option.

```
. ciwidth twomeans, sd1(5.5) sd2(5) width(6) knownsds

Estimated sample sizes for a two-means-difference CI
Normal two-sided CI

Study parameters:

        level =     95.00
        width =    6.0000
          sd1 =    5.5000
          sd2 =    5.0000

Estimated sample sizes:

            N =        48
  N per group =        24
```

We need a sample of 48 subjects, 24 per group. Notice that our results assume that a future study will have the same standard deviations, 5.5 and 5, in the two groups.

◁

PrSS analysis for CIs comparing paired samples

`ciwidth pairedmeans` provides PrSS analysis for a CI for a difference between the means from two paired samples. You can perform computations assuming a known or unknown population standard deviation of the paired differences and adjust for a finite population. See [PSS-3] **ciwidth pairedmeans**. Below we show a quick example of using `ciwidth pairedmeans`.

▷ Example 3: PrSS analysis for a CI comparing paired means

A forester would like to study the effects of a fertilizer treatment on heights of Virginia pine trees. The trees are planted in pairs with only one of them receiving the fertilizer treatment. The average height of untreated trees is 27.5 feet, with a standard deviation of 4.5 feet. The fertilizer treatment is expected to increase the average height to 30 feet, with the same standard deviation of 4.5 feet. The correlation between the paired-tree heights is expected to be 0.4. The forester would like to know how many pairs of pine trees need to be planted to produce a two-sided 95%-level CI for the difference between the two means with a target width of 2 feet. The forester also wants to be at least 90% certain that the produced CI will have the width no larger than 2 feet.

We use `ciwidth pairedmeans` for our PrSS analysis. We supply the common value of standard deviations in the `sd()` option and the correlation of 0.4 in the `corr()` option. We also specify the CI width of 2 in the `width()` option and the probability of CI width of 0.9 in the `probwidth()` option.

```
. ciwidth pairedmeans, sd(4.5) corr(0.4) probwidth(0.9) width(2)

Performing iteration ...

Estimated sample size for a paired-means-difference CI
Student's t two-sided CI assuming sd1 = sd2 = sd

Study parameters:

        level =   95.0000           sd =    4.5000
     Pr_width =    0.9000         corr =    0.4000
        width =    2.0000
         sd_d =    4.9295

Estimated sample size:

            N =       113
```

The forester needs 113 pairs of pine trees to be 90% certain that the obtained two-sided 95% CI will have the width as narrow as 2 feet.

Note that the estimates of the means, 27.5 and 30 feet, do not affect the computation of the required sample size. We mention the average height of the trees for context of their standard deviations, which are used for the computation.

◁

Tables of results

When ciwidth is used to perform computations for a single set of study parameters, the results can be displayed either as text or in a table. The default is to display the results as text:

```
. ciwidth onemean, width(0.5) probwidth(0.9)

Performing iteration ...

Estimated sample size for a one-mean CI
Student's t two-sided CI

Study parameters:

        level =     95.00
     Pr_width =    0.9000
        width =    0.5000
           sd =    1.0000

Estimated sample size:

            N =        77
```

You can specify the table option to display the results in a table:

```
. ciwidth onemean, width(0.5) probwidth(0.9) table

Performing iteration ...

Estimated sample size for a one-mean CI
Student's t two-sided CI
```

level	N	Pr_width	width	sd
95	77	.9	.5	1

For multiple sets of study parameters, when command arguments or options contain number lists, the results are automatically displayed in a table:

```
. ciwidth onemean, width(0.4 0.45 0.5) probwidth(0.9)

Performing iteration ...

Estimated sample size for a one-mean CI
Student's t two-sided CI
```

level	N	Pr_width	width	sd
95	116	.9	.4	1
95	93	.9	.45	1
95	77	.9	.5	1

In this example, we specified multiple sample sizes.

Default tables can be modified by specifying the table() option. For example, we can change the column labels:

```
. ciwidth onemean, width(0.4 0.45 0.5) probwidth(0.9) table(, labels(N "Sample
> size"))
Performing iteration ...
Estimated sample size for a one-mean CI
Student's t two-sided CI
```

level	Sample size	Pr_width	width	sd
95	116	.9	.4	1
95	93	.9	.45	1
95	77	.9	.5	1

Or we can select which columns we want to display:

```
. ciwidth onemean, width(0.4 0.45 0.5) probwidth(0.9) table(N width)
Performing iteration ...
Estimated sample size for a one-mean CI
Student's t two-sided CI
```

N	width
116	.4
93	.45
77	.5

The order of displayed columns follows the specified order. For more examples, see [PSS-3] **ciwidth, table**.

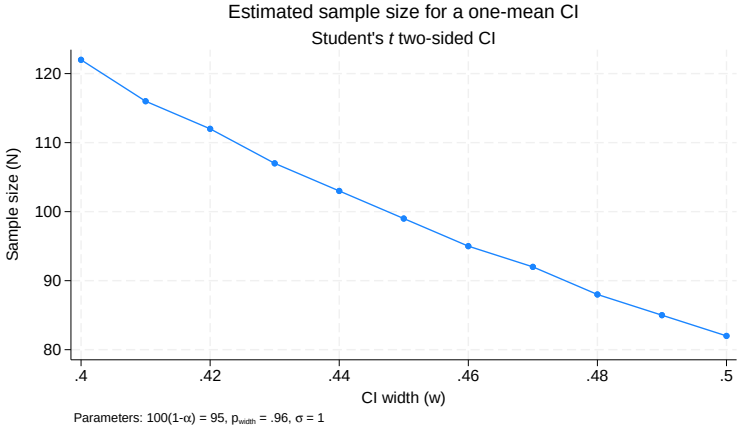
Sample-size and other curves

During the planning stage of a study, it is difficult to decide on the number of subjects to enroll on the basis of only one set of study parameters. It is common to investigate the effect of various study scenarios on CI precision or sample size. We can plot the estimated CI width, probability of CI width, or sample size versus one of the study parameters. The ciwidth command provides the graph and graph() options to produce such curves.

More precisely, when graph is specified, the estimated parameter such as CI width or sample size is plotted on the y axis, and the varying parameter is plotted on the x axis.

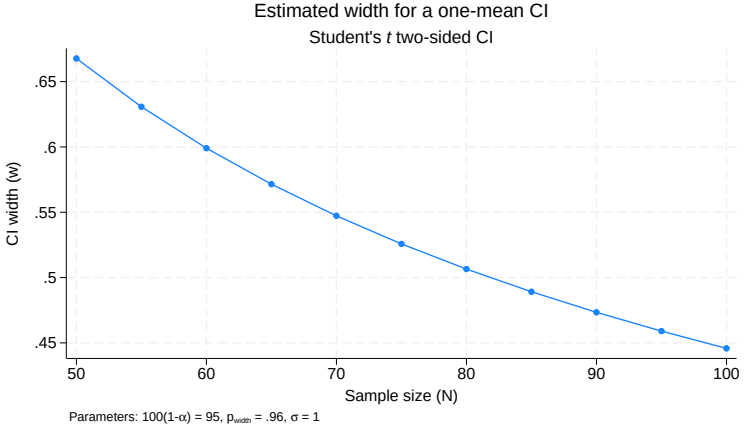
For example, we compute the sample size and plot it as a function of CI width for a range of CI width values between 0.4 and 0.5 with a step size of 0.01:

```
. ciwidth onemean, width(0.4(0.01)0.5) probwidth(0.96) graph
```



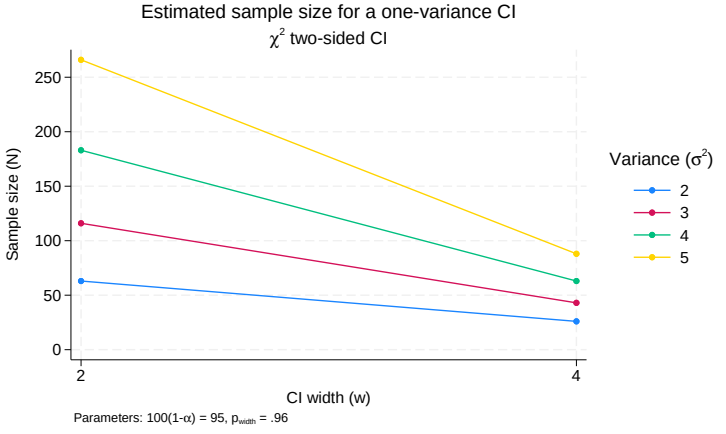
Or we can compute CI width and plot it as a function of sample size:

```
. ciwidth onemean, n(50(5)100) probwidth(0.96) graph
```



We can produce curves for multiple values of multiple parameters, such as the sample variances and the CI widths in this example:

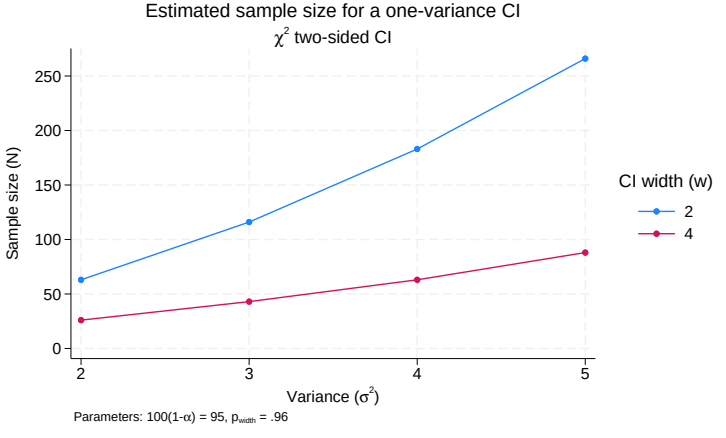
```
. ciwidth onevariance (2 3 4 5), width(2 4) probwidth(0.96) graph
```



The above graphs are the default graphs produced by `ciwidth, graph`. Similarly to tabular output, you can customize graphical output by specifying the `graph()` option.

For example, in the above, we could plot the sample size versus sample variances by using the `graph(xdimension(v))` option.

```
. ciwidth onevariance (2 3 4 5), width(2 4) probwidth(0.96)
> graph(xdimension(v))
```



By default, when a graph is produced, the tabular output is suppressed. You can specify the `table` option if you also want to see results in a table.

For more examples, see [PSS-3] **ciwidth, graph**.

Add your own methods to ciwidth

The ciwidth command provides several built-in methods, but sometimes, you may want to compute sample size or CI width yourself. For example, you may need to do this by simulation, or you may want to use a method that is not available in any software package. ciwidth makes it easy for you to add your own method. All you need to do is to write a program that computes sample size, probability of CI width, or CI width, and the ciwidth command will do the rest for you. It will deal with the support of multiple values in options and with automatic generation of graphs and tables of results.

See *A quick example* and *More examples: Compute probability of CI width for a one-proportion CI* in [PSS-3] ***ciwidth usermethod*** for examples.

Stored results

ciwidth stores the following in r():

Scalars

r(level)	confidence level
r(alpha)	significance level
r(N)	total sample size
r(N_a)	actual sample size
r(N1)	sample size of the control group
r(N2)	sample size of the experimental group
r(nratio)	ratio of sample sizes, N2/N1
r(nratio_a)	actual ratio of sample sizes
r(nfractional)	1 if nfractional is specified, 0 otherwise
r(onesided)	1 for a one-sided CI, 0 otherwise
r(Pr_width)	probability of CI width
r(Pr_width_a)	actual probability of CI width (for sample-size determination when probwidth() specified)
r(width)	CI width
r(width_a)	actual CI width (for sample-size determination of some methods)
r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise
r(init)	initial value for estimated parameter
r(maxiter)	maximum number of iterations
r(iter)	number of iterations performed
r(tolerance)	requested parameter tolerance
r(deltax)	final parameter tolerance achieved
r(ftolerance)	requested distance of the objective function from zero
r(function)	final distance of the objective function from zero
r(converged)	1 if iteration algorithm converged, 0 otherwise

Macros

r(type)	ci
r(method)	the name of the specified ciwidth method
r(onesidedci)	upper or lower (for a one-sided CI)
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results

Also see *Stored results* in the method-specific manual entries for the full list of stored results.

Methods and formulas

By default, the `ciwidth` command rounds sample sizes to integers and uses integer values in the computations. To ensure conservative results, the command rounds down the input sample sizes and rounds up the output sample sizes. See *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for details.

Some of `ciwidth`'s methods require iteration, such as a sample-size determination for a Student's t CI in `ciwidth onemean`. The methods use the Mata function `solvenl()`, and its Newton's method described in *Newton-type methods* in [M-5] **solvenl( )**, to solve nonlinear equations.

See *Methods and formulas* in the method-specific manual entries for details.

Also see

[PSS-3] **Intro (ciwidth)** — Introduction to precision and sample-size analysis for confidence intervals

[PSS-5] **Glossary**

[R] **ci** — Confidence intervals for means, proportions, and variances

ciwidth usermethod — Add your own methods to the ciwidth command

Description Syntax Remarks and examples References Also see

Description

The ciwidth command allows you to add your own methods to ciwidth and produce tables and graphs of results automatically.

Syntax

Compute sample size

```
ciwidth usermethod ..., width(numlist) [probwidth(numlist) ciwidthopts useropts]
```

Compute CI width

```
ciwidth usermethod ..., nspec [probwidth(numlist) ciwidthopts useropts]
```

Compute probability of CI width

```
ciwidth usermethod ..., nspec width(numlist) [ciwidthopts useropts]
```

usermethod is the name of the method you would like to add to the ciwidth command. When naming your ciwidth methods, you should follow the same convention as for naming the programs you add to Stata—do not pick "nice" names that may later be used by Stata's official methods. The length of *usermethod* may not exceed 14 characters.

useropts are the options supported by your method *usermethod*.

nspec contains n(*numlist*) for a one-sample CI or any of the sample-size options of *ciwidthopts* such as n1(*numlist*) and n2(*numlist*) for a two-sample CI.

collect is allowed; see **[U] 11.1.10 Prefix commands**.

Remarks and examples

Adding your own methods to ciwidth is easy. Suppose you want to add a method called mymethod to ciwidth. Simply

1. write an r-class program called ciwidth_cmd_mymethod that computes sample size, probability of CI width, or CI width and follows ciwidth's convention for naming common options and storing results; and
2. place the program where Stata can find it.

You are done. You can now use mymethod within ciwidth like any other official ciwidth method.

Remarks are presented under the following headings:

A quick example

Before we discuss the technical details in the following sections, let's try an example. Let's write a program to compute CI width for a one-mean normal-based CI given sample size, standard deviation, and confidence level. For simplicity, we assume a two-sided CI. We will call our new method `mymean`. (Note that this method is available in the official `ciwidth onemean` command when you specify the `knownsd` option.)

We create an ado-file called `ciwidth_cmd_mymean.ado` that contains the following Stata program:

```
// evaluator
program ciwidth_cmd_mymean, rclass
        version 19.5      // (or version 19 if you do not have StataNow)
        /* parse options */
        syntax, n(integer)          /// sample size
                [ Level(cilevel)    /// confidence level
                  Stddev(real 1) ]  /// standard deviation
        /* compute CI width */
        tempname width
        scalar `width' = 2*invnormal(1/2+`level'/200)*`stddev'/sqrt(`n')
        /* store results */
        return scalar level   = `level'
        return scalar N       = `n'
        return scalar width   = `width'
        return scalar stddev  = `stddev'
end
```

Our ado-program consists of three sections: the `syntax` command for parsing options, the CI width computation, and stored or returned results. The three sections work as follows:

The `ciwidth_cmd_mymean` program has two of `ciwidth`'s common options, `level()` for confidence level and `n()` for sample size, and it has its own option, `stddev()`, with the minimum abbreviation `s()` and default value of 1, to specify a standard deviation.

After the options are parsed, the CI width is computed and stored in a temporary scalar `` `width' ``.

Finally, the resulting CI width and other results are stored in return scalars. Following `ciwidth`'s convention for naming commonly returned results, the confidence level is stored in `r(level)`, the sample size in `r(N)`, and the computed CI width in `r(width)`. The program additionally stores the standard deviation in `r(stddev)`.

We can now use `mymean` within `ciwidth` as we would any other existing method of `ciwidth`:

```
. ciwidth mymean, level(95) n(10) stddev(0.25)
Estimated width
Two-sided CI
```

level	N	width
95	10	.3099

We can check our result using the official `ciwidth onemean`:

```
. ciwidth onemean, level(95) n(10) sd(0.25) knownsd
Estimated width for a one-mean CI
Normal two-sided CI
Study parameters:
        level =     95.00
            N =        10
           sd =    0.2500
Estimated width:
        width =    0.3099
```

We can compute results for multiple sample sizes by specifying multiple values in the `n()` option. Note that our `ciwidth_cmd_mymean` program accepts only one value at a time in `n()`. When a numlist is specified in the `ciwidth` command's `n()` option, `ciwidth` automatically handles that *numlist* for us.

```
. ciwidth mymean, level(95) n(10 20) stddev(0.25)
Estimated width
Two-sided CI
```

level	N	width
95	10	.3099
95	20	.2191

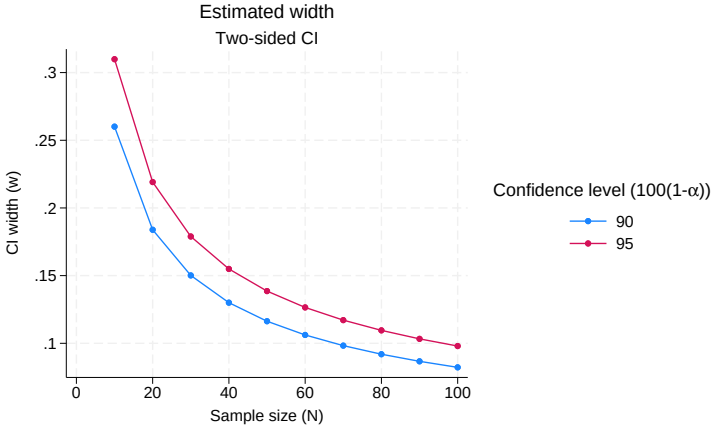
We can also compute results for multiple sample sizes and confidence levels without any additional effort on our part:

```
. ciwidth mymean, level(90 95) n(10 20) stddev(0.25)
Estimated width
Two-sided CI
```

level	N	width
90	10	.2601
90	20	.1839
95	10	.3099
95	20	.2191

We can even produce a graph by merely specifying the `graph` option:

```
. ciwidth mymean, level(90 95) n(10(10)100) stddev(0.25) graph
```



The above is just a simple example. Your program can be as complicated as you would like: you can even use simulations to compute your results; see *More examples: Compute probability of CI width for a one-proportion CI*. You can also customize your tables and graphs with a little extra effort.

Steps for adding a new method to the ciwidth command

Suppose you want to add your own method, *usermethod*, to the `ciwidth` command. Here is an outline of the steps to follow:

1. Create the evaluator, an r-class program called `ciwidth_cmd_`*usermethod* and defined by the ado-file `ciwidth_cmd_`*usermethod*`.ado`, that performs precision and sample-size computations and follows `ciwidth`'s convention for naming options and storing results.
2. Optionally, create an initializer, an s-class program called `ciwidth_cmd_`*usermethod*`_init` and defined by the ado-file `ciwidth_cmd_`*usermethod*`_init.ado`, that specifies information about table columns, options that may allow a numlist, and so on.
3. Optionally, create a parser, a program called `ciwidth_cmd_`*usermethod*`_parse` and defined by the ado-file `ciwidth_cmd_`*usermethod*`_parse.ado`, that checks the syntax of user-specific options, *useropts*.
4. Place all of your programs where Stata can find them.

You can now use your *usermethod* with `ciwidth`:

```
. ciwidth usermethod ...
```

You may also use programs within `ciwidth` that are not defined by an ado-file (that is, they were defined in a do-file or by hand).

Convention for naming options and storing results

For the `ciwidth` command to automatically recognize its common options, you must ensure that you follow `ciwidth`'s naming convention for these options in your program. For example, `ciwidth` specifies the confidence level in the `level()` option with minimum abbreviation of `l()`. You need to ensure that you use the same option with the same abbreviation in your evaluator to specify the confidence level. The same applies to all of `ciwidth`'s common options described in [PSS-3] **ciwidth**.

You can specify additional method-specific options, but `ciwidth` will not know about them unless you make it aware of them; see *Allowing multiple values in method-specific options* for details.

Unlike `ciwidth`'s official methods, user-defined methods do not require specifying the `probwidth()` option by default because some computations, such as our earlier normal-based one-mean CI example, may not need the probability of CI width. The probability of CI width is often needed when the computation of the width depends on unknown parameters that are themselves estimated from the data. For instance, if the standard deviation is not known a priori, the computation of the CI width for a one-mean CI incorporates the uncertainty about the specified standard deviation because its estimate may vary from one sample to another. The specified probability of CI width is used to ensure that the estimated CI width is no larger than the desired width of a future CI with the prespecified probability. This is the default method of `ciwidth onemean`. Also see *More examples: Compute probability of CI width for a one-proportion CI* for an example of computing probability of CI width.

To produce tables and graphs of results, you must ensure that your evaluator follows `ciwidth`'s convention for storing results. `ciwidth`'s commonly stored results are described in *Stored results* of [PSS-3] **ciwidth**. For example, the value for a confidence level should be stored in the scalar `r(level)`, the value for a total sample size in the scalar `r(N)`, the value for CI width in the scalar `r(width)`, the value for probability of CI width, if available, in the scalar `r(Pr_width)`, and so on.

You can also store additional method-specific results, but `ciwidth` will not know about them unless you make it aware of them; see *Customizing default tables* for details.

Allowing multiple values in method-specific options

By default, the `ciwidth` command accepts multiple values only within its common options. If you want to allow multiple values in the method-specific options *useropts*, you need to let `ciwidth` know about them. This is done via the initializer.

To allow the specification of multiple values, or a numlist, in method-specific options, you need to list the names of the options with proper abbreviations in an s-class macro `s(prss_numopts)` within the `ciwidth_cmd_`*usermethod*`_init` program.

Recall our earlier example in which we added the `mymean` method, calculating the CI width of a two-sided normal CI for one-sample mean, to `ciwidth`. We computed CI widths for multiple values of confidence level and sample size. What if we would also like to specify multiple values of standard deviation in the `stddev()` option of `mymean`? If we do this now, we will receive an error message,

```
. ciwidth mymean, level(95) n(10) stddev(0.25 0.5)
option stddev() incorrectly specified
r(198);
```

because the `stddev()` option is not allowed to include *numlist* by the evaluator and is not one of `ciwidth`'s common options. To make `ciwidth` recognize this option as one allowing *numlist*, we need to specify this in the initializer. Following the guidelines, we create an ado-file called

`ciwidth_cmd_mymean_init.ado` and specify the name of the `stddev()` option (with the corresponding abbreviation) in the s-class macro `s(prss_numopts)` within the `ciwidth_cmd_mymean_init` program.

```
// initializer
program ciwidth_cmd_mymean_init, sclass
        version 19.5      // (or version 19 if you do not have StataNow)
        sreturn clear
        sreturn local prss_numopts "Stddev"
end
```

We now can specify multiple standard deviations:

```
. ciwidth mymean, level(95) n(10) stddev(0.25 0.5)
Estimated width
Two-sided CI
```

level	N	width
95	10	.3099
95	10	.6198

Customizing default tables

The `ciwidth` command with user-defined methods always displays results in a table. By default, it displays columns `level` or `alpha` (whichever is specified), `N`, and `width`, which contain the confidence level, the sample size, and the CI width, respectively. If option `probwidth()` or both options `n()` and `width()` are specified, the `Pr_width` column is also shown in the default table following the `N` column. See *Setting supported columns* and *Modifying the default table columns* for details on how to customize the default table columns.

The default column labels are the column names, and the default formats are `%7.4g` for `level`, `alpha`, `width`, and `Pr_width` and `%7.0gc` for `N`. These and other settings controlling the look of the default table can be changed as described in *Modifying the look of the default table*.

You can always use the `table()` option to customize your table. However, if you want to modify how the table looks by default, you need to follow the steps described in the following sections:

Setting supported columns

The `ciwidth` command automatically supports a number of columns, such as `level`, `alpha`, `width`, `Pr_width`, `N`, etc. The supported columns are the columns that can be accessed within `ciwidth`'s options `table()` and `graph()`.

Most of the time, you will have additional columns, *usercolnames*, which you will want `ciwidth` to support. To make `ciwidth` recognize the columns as supported columns, you must list the names of the additional columns, *usercolnames*, in an s-class macro `s(prss_colnames)` in the initializer. Columns *usercolnames* will then be added to `ciwidth`'s list of supported columns. Columns *usercolnames* will also be displayed in the default table unless `s(prss_tabcolnames)` or `s(prss_alltabcolnames)` is set.

If you want to reset `ciwidth`'s list of supported columns, that is, to specify all the supported columns manually, you should use the `s(prss_allcolnames)` macro. The supported columns will then include only the ones you listed in the macro. If you specify `s(prss_allcolnames)`, you must remember to include `ciwidth`'s main columns `N`, `width`, `level`, `Pr_width` (if applicable) in your list. Otherwise, you may not be able to use some of `ciwidth`'s functionality, such as default graphs. If `s(prss_colnames)` is specified together with `s(prss_allcolnames)`, the former will be ignored. The specified supported columns will be automatically displayed in the default table unless `s(prss_alltabcolnames)` is set.

The values corresponding to the specified columns must be stored by the evaluator in `r()` scalars with the same names as the column names. For example, the value for column `level` is stored in `r(level)`, the value for column `width` is stored in `r(width)`, and the value for column `N` is stored in `r(N)`.

Any column not listed in `s(prss_colnames)` or `s(prss_allcolnames)` will not be available within the `ciwidth` command. For example, any reference to such a column within `ciwidth`'s options `table()` and `graph()` will result in an error.

Modifying the default table columns

By default, `ciwidth` displays the specified supported columns. If you want to display only a subset of those columns, you can use either `s(prss_tabcolnames)` or `s(prss_alltabcolnames)` to specify the columns to be displayed. You specify additional columns to be displayed in `s(prss_tabcolnames)` and a complete list of the displayed columns in `s(prss_alltabcolnames)`. If you specify `s(prss_tabcolnames)`, the displayed columns will include `level` or `alpha` (whichever is specified with the command), `N`, `Pr_width` (if applicable), `width` and the additional columns you specified. If you specify `s(prss_alltabcolnames)`, only the columns listed in this macro will be displayed. One situation when you may want to do this is if you want to change the order in which the columns are displayed by default. If you specify both macros, `s(prss_tabcolnames)` will be ignored. You can specify only the names of supported columns in these macros.

Modifying the look of the default table

The default table column labels are the column names. You can change this by specifying your own column labels in the `s(prss_collabels)` macro. The labels must be properly quoted if they contain spaces or quotes. The labels must be specified for all columns listed in `s(prss_colnames)` or `s(prss_allcolnames)`.

The default column formats are `%7.0gc` for sample sizes and `%7.4g` for all other columns. You can change this by specifying your own column formats in the `s(prss_colformats)` macro. The formats must be quoted and specified for all columns listed in `s(prss_colnames)` or `s(prss_allcolnames)`.

The default column widths are the widths of the default formats plus one. You can specify your own column widths in the `s(prss_colwidths)` macro. The widths must be specified for all columns listed in `s(prss_colnames)` or `s(prss_allcolnames)`.

Example continued

Continuing our `mymean` example, we want to add a column containing the specified standard deviation to the list of supported columns. The specified standard deviation is stored in `r(stddev)` in the `mymean` evaluator, so the name of our column is `stddev`. We specify it in `s(prss_colnames)` in our initializer as follows:

```
// initializer
program drop ciwidth_cmd_mymean_init
program ciwidth_cmd_mymean_init, sclass
        version 19.5      // (or version 19 if you do not have StataNow)
        sreturn clear
        sreturn local prss_numopts "Stddev"
        sreturn local prss_colnames "stddev"  // <-- new line
end
```

We rerun our command now and see that the `stddev` column is added to the default table:

```
. ciwidth mymean, level(95) n(10) stddev(0.25)
Estimated width
Two-sided CI
```

level	N	width	stddev
95	10	.3099	.25

We can also change the default column label of the `stddev` column to "Std. dev.". Note that we can do this within `ciwidth`'s option `table()`, but if we want this label to show up automatically in the default table, we should specify it in the initializer. We specify the column label in the `s(prss_collabels)` macro.

```
// initializer
program drop ciwidth_cmd_mymean_init
program ciwidth_cmd_mymean_init, sclass
        version 19.5      // (or version 19 if you do not have StataNow)
        sreturn clear
        sreturn local prss_numopts "Stddev"
        sreturn local prss_colnames "stddev"
        sreturn local prss_collabels `""Std. dev.""'  // <-- new line
end
```

The column containing standard deviation now has the new label

```
. ciwidth mymean, level(95) n(10) stddev(0.25)
Estimated width
Two-sided CI
```

level	N	width	Std. dev.
95	10	.3099	.25

Customizing default graphs

By default, `ciwidth` plots the estimated CI width on the y axis and the specified sample size on the x axis or the estimated sample size on the y axis and the specified CI width on the x axis. See [PSS-3] **ciwidth, graph** for details about other default settings.

You can overwrite the default column labels displayed on the graph by specifying the `s(prss_colgrlabels)` macro. The specification of the graph labels is the same as the specification of table column labels.

You can also overwrite the default symbols that are used to label the results on the graph by specifying the new symbols in the macro `s(prss_colgrsymbols)`. The symbols must be specified for all columns listed in `s(prss_colnames)` or `s(prss_allcolnames)`.

Other settings

If your method supports command arguments, the arguments specified directly following the method name, you can specify their corresponding column names in the `s(prss_argnames)` macro. You can then refer to these arguments as `arg1`, `arg2`, and so on, when producing tables or graphs.

`ciwidth` *usermethod* uses the following generic titles: "Estimated sample size" for sample-size determination, "Estimated width" for CI width determination, and "Estimated probability of width" for probability of CI width determination. You can extend these titles to be more specific to your method by adding text in the `s(prss_title)` macro. For example, if `s(prss_title)` contains "for my CI", the resulting titles will be "Estimated sample size for my CI", "Estimated width for my CI", and "Estimated probability of width for my CI".

`ciwidth` *usermethod* uses the following generic subtitles: "Two-sided CI" for a two-sided CI, "One-sided upper CI" when the `upper` option is specified, and "One-sided lower CI" when the `lower` option is specified. You can change the default subtitle by specifying the `s(prss_subtitle)` macro.

The steps for adding your own two-sample methods are the same as those for adding one-sample methods, except you may need to set the `s(prss_samples)` macro to contain `twosample` in the initializer. If any of the two-sample options `n1()`, `n2()`, and `nratio()` are specified, `ciwidth` automatically recognizes the method as a two-sample method. If these options are not used and you need the method to be recognized as a two-sample method, you must set `s(prss_samples)` to `twosample`. If you do not want `ciwidth` to respect the two-sample options, you should set `s(prss_samples)` to `onesample`.

Handling parsing more efficiently

The `ciwidth` command checks its common options, but you are responsible for checking your method-specific options, *useropts*, or their interaction with `ciwidth`'s common options. You can certainly do this in your evaluator. However, the checks will then be performed each time your evaluator is called. You can instead perform all of your checks once within the parser.

Your parser may be an s-class command and may set any of the `s()` results typically specified in the initializer. This may be useful, for example, for building the columns displayed in the default table dynamically, depending on which options were specified. If all desired `s()` results are set in the parser, you do not need an initializer.

More examples: Compute probability of CI width for a one-proportion CI

For some CIs, the expressions for the required sample size or CI width may not be available or difficult to compute. In such cases, you can use simulation to obtain the results. And you can turn your simulation program into a user-defined `ciwidth` method. Huber (2019a) and Huber (2019b) describe how to compute power by simulation and integrate the simulation program in the `power` command. The same principles apply to the simulation of CI width or probability of CI width and its integration in the `ciwidth` command.

The `ciwidth` command does not provide precision and sample-size analysis for CIs for proportions. The width of CIs for proportions depends on the estimates of proportions. Its estimation thus needs to account for the uncertainty in the proportion estimates. There are no closed-form solutions to compute the required sample size or width for CIs for proportions that incorporate the probability of CI width. But we can compute the probability of CI width for a given sample size and target width using simulation. We can then vary the sample sizes to see which ones correspond to high values of the probability of CI width for the desired CI width. Let's do this for the binomial CI for one proportion.

We can estimate the probability of CI width as the proportion of times the width of the CI computed from simulated samples is less than or equal to the desired CI width for a given sample size and probability of success. Our steps are to 1) create a program that simulates the data and computes the width of the estimated CI; 2) run the program multiple times and compute the probability of CI width; and 3) add our computations to `ciwidth` as a new method. We can actually compute the probability of CI width exactly, without the simulation, for the binomial CI. So we compare our simulation results with the exact computation in step 4.

Step 1: Program to simulate the data and compute the CI width

We start with a simple program `myonepropsim` below.

```
program myonepropsim, rclass
        version 19.5      // (or version 19 if you do not have StataNow)
        args n p level
        clear
        set obs `n'
        generate byte y = rbinomial(1, `p')
        ci proportions y, level(`level')
        return scalar w = r(ub)-r(lb)
end
```

Our program requires three arguments: `n` for sample size, `p` for proportion estimate, and `level` for confidence level. It generates `` `n' `` observations for the binary outcome `y` from a Bernoulli distribution with a specified probability of success `` `p' ``. (`` `' `` refers to the specified values for the arguments.) It uses the `ci proportions` command ([R] **ci**) to estimate the proportion of successes (`y==1`) and its binomial CI. It then computes and stores in the return scalar `r(w)` the estimated CI width—the difference between the upper and lower CI bounds stored by `ci proportions` in return scalars `r(ub)` and `r(lb)`, respectively.

Let's run our program. Suppose that we want to simulate 50 Bernoulli observations with a low success probability of 0.1 and compute the width of the corresponding 95% two-sided binomial CI for the proportion of successes. Because we randomly generate the data, we use `set seed` prior to calling `myonepropsim` for reproducibility.

```
. set seed 1234
. myonepropsim 50 0.1 95
Number of observations (_N) was 0, now 50.
```

Variable	Obs	Proportion	Std. err.	Binomial exact [95% conf.	interval]
y	50	.18	.0543323	.0857621	.3143694

```
. return list
scalars:
                  r(w) =  .2286073331759869
```

From the stored results, the estimated CI width, `r(w)`, is 0.23.

Step 2: Estimating probability of CI width using simulation

Suppose that our target CI width is 0.2. To estimate the probability of CI width, we need to call our `myonepropsim` program multiple times and compute the proportion of times the estimated CI width was less than or equal to our target width of 0.2.

Stata has a handy command to run simulations—`simulate` ([R] **simulate**).

```
. set seed 1234
. simulate w=r(w), reps(100): myonepropsim 50 0.1 95
      Command: myonepropsim 50 0.1 95
            w: r(w)
Simulations (100): .........10.........20.........30.........40.........50.....
> ....60.........70.........80.........90.........100 done
. count if w <= 0.2
  75
. display r(N)/100
.75
```

`simulate` runs `myonepropsim` 100 times, as specified by `simulate`'s `reps()` option, and stores the computed CI widths in the `w` variable, as requested by the `w=r(w)` specification. We then count the number of observations of `w` that are less than or equal to 0.2 and estimate the probability of CI width to be 0.75.

Step 3: Adding probability of CI width computation to ciwidth

Our final step is to combine all of our computations in a single program and integrate it into the `ciwidth` command. Following `ciwidth`'s convention, we call our new program `ciwidth_cmd_myonepropsim`.

```
program ciwidth_cmd_myonepropsim, rclass
        version 19.5      // (or version 19 if you do not have StataNow)
        /* parse command arguments and options */
        syntax anything(name=p),          /// proportion estimate
                        n(integer)        /// sample size
                        Width(real)       /// target CI width
                        [ Level(cilevel) /// confidence level
                        reps(integer 100) qui ]
        /* compute probability of CI width using simulation */
        display as txt _n "Computing Pr(width) for n=`n' and width=`w' ..."
        `qui' simulate w=r(w), reps(`reps'): myonepropsim `n' `p' `level'
        quietly count if w <= `width'
        /* store results */
        return scalar Pr_width = r(N)/`reps'
        return scalar level = `level'
        return scalar N = `n'
        return scalar width = `width'
        return scalar p = `p'
end
```

As in *A quick example*, we use `syntax` to parse options. We have three new options. The `width()` option, with the minimum abbreviation `w()`, is one of `ciwidth`'s common options; it specifies the target CI width. The `reps()` option is specific to our method—it specifies the number of replications for the simulation, with the default of 100 replications. Finally, `qui` suppresses the output from the `simulate` command that we display by default.

In addition to options, our program requires that the proportion estimate be specified as a command argument. We could have specified it as an option, say, `proportion(real)`, but here we wanted to demonstrate how to handle arguments with user-defined `ciwidth` methods. Also, official `ciwidth` methods typically specify estimates of parameters of interest, such as proportion, as command arguments, which are specified following the command name.

The block after `syntax` includes our earlier `simulate` command to which we now pass the content of the specified command argument and options instead of the hard-coded values. We also compute the estimate of probability of CI width and store it in the `r(Pr_width)` scalar, following `ciwidth`'s naming convention for the common stored results; see *Stored results* of [PSS-3] **ciwidth**. We also store other results in the corresponding return scalars.

Let's recompute the probability of CI width from *Step 2: Estimating probability of CI width using simulation* but now using `ciwidth myonepropsim`.

```
. set seed 1234
. ciwidth myonepropsim 0.1, n(50) width(0.2)
Computing Pr(width) for n=50 and width=.2 ...
      Command: myonepropsim 50 .1 95
            w: r(w)
Simulations (100): .........10.........20.........30.........40.........50.....
> ....60.........70.........80.........90.........100 done
Estimated probability of width
Two-sided CI
```

level	N	Pr_width	width
95	50	.75	.2

We obtain the same estimate of 0.75. We used the default value of the `level()` option, which is `level(95)` or as set by `set level`; see [R] **level**.

Notice that our default table now contains the `Pr_width` column. Because we specified both `n()` and `width()`, `ciwidth` recognized this as the case for computing probability of CI width and automatically added its column to the default table. However, we are missing the proportion estimate in our default table. `ciwidth` is not aware of user-defined command arguments until we specify them in the initializer.

```
program ciwidth_cmd_myonepropsim_init, sclass
        version 19.5      // (or version 19 if you do not have StataNow)
        sreturn clear
        sreturn local prss_argnames = "p"
        sreturn local prss_colnames = "p"
        sreturn local prss_subtitle = "Two-sided binomial CI"
end
```

We list the name of the stored result containing the proportion estimate in macro `prss_argnames` to allow the specification of multiple values for the proportion and in macro `prss_colnames` to add the proportion column to the default table. We also specify a more descriptive subtitle to be used in the output.

If we rerun our previous command (with the `qui` option to suppress the output from the `simulate` command),

```
. set seed 1234
. ciwidth myonepropsim 0.1, n(50) width(0.2) qui
Computing Pr(width) for n=50 and width=.2 ...
Estimated probability of width
Two-sided binomial CI
```

level	N	Pr_width	width	p
95	50	.75	.2	.1

we will now see the proportion column in the table and the new subtitle.

The estimated probability of CI width of 0.75 is somewhat low. We can specify multiple sample sizes to find an acceptable value of probability of CI width.

```
. set seed 1234
. ciwidth myonepropsim 0.1, n(50 70 100) width(0.2) qui
Computing Pr(width) for n=50 and width=.2 ...
Computing Pr(width) for n=70 and width=.2 ...
Computing Pr(width) for n=100 and width=.2 ...
Estimated probability of width
Two-sided binomial CI
```

level	N	Pr_width	width	p
95	50	.75	.2	.1
95	70	1	.2	.1
95	100	1	.2	.1

We can further explore the sample sizes between 50 and 70:

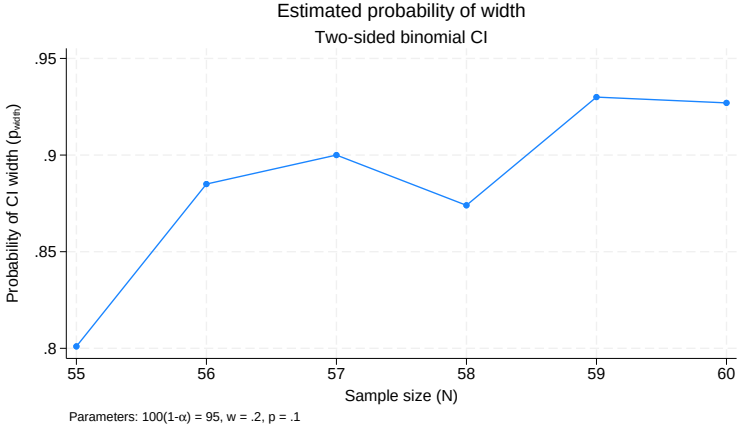
```
. set seed 1234
. ciwidth myonepropsim 0.1, n(50(5)70) width(0.2) qui
Computing Pr(width) for n=50 and width=.2 ...
Computing Pr(width) for n=55 and width=.2 ...
Computing Pr(width) for n=60 and width=.2 ...
Computing Pr(width) for n=65 and width=.2 ...
Computing Pr(width) for n=70 and width=.2 ...
Estimated probability of width
Two-sided binomial CI
```

level	N	Pr_width	width	p
95	50	.75	.2	.1
95	55	.78	.2	.1
95	60	.97	.2	.1
95	65	.98	.2	.1
95	70	1	.2	.1

Finally, we can compute probability of CI widths for sample sizes between 55 and 60 and plot the results in addition to the table. We also specify more replications to obtain more precise estimates of the probability of CI width.

```
. set seed 1234
. ciwidth myonepropsim 0.1, n(55(1)60) width(0.2) qui reps(1000) table graph
Computing Pr(width) for n=55 and width=.2 ...
Computing Pr(width) for n=56 and width=.2 ...
Computing Pr(width) for n=57 and width=.2 ...
Computing Pr(width) for n=58 and width=.2 ...
Computing Pr(width) for n=59 and width=.2 ...
Computing Pr(width) for n=60 and width=.2 ...
Estimated probability of width
Two-sided binomial CI
```

level	N	Pr_width	width	p
95	55	.801	.2	.1
95	56	.885	.2	.1
95	57	.9	.2	.1
95	58	.874	.2	.1
95	59	.93	.2	.1
95	60	.927	.2	.1



For example, the sample size of 57 corresponds to the probability of 0.9 that the width of a future 95% two-sided binomial CI for one proportion will not exceed 0.2 given the proportion estimate of 0.1.

The probability of CI width is typically a monotonically increasing function of the sample size. However, similarly to the power of the binomial test, the probability of CI width for the binomial CI may not be monotonic with respect to the sample size, as we see in this example, because of the discrete nature of the binomial distribution.

Of course, because we use simulation, if we rerun `ciwidth myonepropsim` with a different seed, we will get different results. The results, however, should be comparable provided the number of replications is sufficiently large.

Step 4: Computing exact probability of CI width

We can compute the probability of CI width for the binomial CI more easily by using the exact formula

$$\Pr(w) = \sum_{k=0}^{n} \texttt{binomialp}(n,k,p) \times I(w_{k,n} \leq w_{\text{target}})$$

where $\texttt{binomialp}(n,k,p)$ is the probability of observing k successes in n trials with the success probability p; w_{target} is the target CI width; $w_{k,n}$ is the width of the binomial CI computed given k observed successes in n trials; and the indicator function

$$I(w_{k,n} \leq w_{\text{target}}) = \begin{cases} 1, & \text{if } w_{k,n} \leq w_{\text{target}} \\ 0, & \text{otherwise} \end{cases}$$

The `ciwidth_cmd_myoneprop` program below uses the formula above to compute the probability of CI width.

```
program ciwidth_cmd_myoneprop, rclass
        version 19.5      // (or version 19 if you do not have StataNow)
        /* parse command arguments and options */
        syntax anything(name=p),            /// proportion estimate
                        n(integer)          /// sample size
                        Width(real)         /// target CI width
                        [ Level(cilevel) ] /// confidence level
        /* compute probability of CI width using exact formula */
        tempname Pr_width
        scalar `Pr_width' = 0
        forvalues k = 0/`n' {
                quietly cii proportions `n' `k', level(`level')
                if (r(ub)-r(lb) <= `width') {
                        scalar `Pr_width' = `Pr_width' + binomialp(`n',`k',`p')
                }
        }
        /* store results */
        return scalar Pr_width = `Pr_width'
        return scalar level = `level'
        return scalar N = `n'
        return scalar width = `width'
        return scalar p = `p'
end
```

This program is similar to our earlier `ciwidth_cmd_myonepropsim` program but without the `reps()` and `qui` options and using the exact formula instead of simulation to compute the probability of CI width. Also, instead of using `ci proportions`, which estimates the binomial CI from the data, we use its immediate version, `cii proportions`, which uses the numbers supplied in `` `n' `` and `` `k' `` to compute the CI; see **[U] 19 Immediate commands** for a general discussion of immediate commands.

We followed `ciwidth`'s naming convention for the `ciwidth_cmd_myoneprop` program, so we can use `myoneprop` with `ciwidth`. Let's compute the exact probability of CI width for sample sizes, n, between 55 and 60 given the target CI width, w_{target}, of 0.2 and probability of success, p, of 0.1, using the default 95% confidence level.

```
. ciwidth myoneprop 0.1, n(55(1)60) width(0.2)
Estimated probability of width
Two-sided CI
```

level	N	Pr_width	width
95	55	.8196	.2
95	56	.897	.2
95	57	.888	.2
95	58	.8785	.2
95	59	.9334	.2
95	60	.9269	.2

Our results are similar to the simulation results from the last table in *Step 3: Adding probability of CI width computation to ciwidth*. We can match the exact results even more closely if we use more replications, say, 10,000, during simulation.

Initializer's s() return settings

The following `s()` results may be set by the initializer or parser:

Macros

`s(prss_samples)`	`onesample` for a one-sample CI or `twosample` for a two-sample CI
`s(prss_colnames)`	columns to be added to the default supported columns
`s(prss_allcolnames)`	all supported columns
`s(prss_tabcolnames)`	columns to be added to the default table
`s(prss_alltabcolnames)`	all columns to be displayed in the default table
`s(prss_collabels)`	labels for the specified columns
`s(prss_colformats)`	formats for the specified columns
`s(prss_colwidths)`	widths for the specified columns
`s(prss_colgrlabels)`	labels to be used to label columns on the graph
`s(prss_colgrsymbols)`	symbols to be used to label columns on the graph
`s(prss_argnames)`	column names containing command arguments
`s(prss_title)`	method-specific title
`s(prss_subtitle)`	subtitle

References

Huber, C. 2019a. Calculating power using Monte Carlo simulations, part 1: The basics. *The Stata Blog: Not Elsewhere Classified*. https://blog.stata.com/2019/01/10/calculating-power-using-monte-carlo-simulations-part-1-the-basics/.

———. 2019b. Calculating power using Monte Carlo simulations, part 2: Running your simulation using power. *The Stata Blog: Not Elsewhere Classified*. https://blog.stata.com/2019/01/29/calculating-power-using-monte-carlo-simulations-part-2-running-your-simulation-using-power/.

Also see

[PSS-3] **ciwidth** — Precision and sample-size analysis for CIs

[PSS-3] **Intro (ciwidth)** — Introduction to precision and sample-size analysis for confidence intervals

[PSS-5] **Glossary**

ciwidth, graph — Graph results from the ciwidth command

Description

The `graph()` option of `ciwidth` specifies that results of the `ciwidth` command be graphed.

While there are many options for controlling the look of the graph, you will often merely need to specify the `graph` option with your `ciwidth` command.

Quick start

Graph of sample-size estimates versus the specified list of CI width values

```
ciwidth onemean, probwidth(0.9) width(0.5(0.5)2) graph
```

Graph of sample-size estimates versus the specified list of confidence levels

```
ciwidth onemean, level(80(5)90) probwidth(0.9) width(0.5) graph
```

Same as above, but use significance level as the x axis

```
ciwidth onemean, level(80(5)90) probwidth(0.9) width(0.5) ///
  graph(xdimension(alpha))
```

Graph of probability of CI width versus the specified list of CI width values

```
ciwidth onemean, n(10) width(0.5(0.5)2) graph
```

Add labels for distinct values on the y axis

```
ciwidth onemean, n(10) width(0.5(0.5)2) graph(yvalues)
```

Same as above, but display the probability of CI width with only three decimal points

```
ciwidth onemean, n(10) width(0.5(0.5)2) ///
  graph(yvalues ylabel(,format(%4.3f)))
```

Graph of CI width versus the specified list of sample sizes at each confidence level

```
ciwidth onemean, level(90 95) n(30(5)45) probwidth(0.9) graph
```

Same as above, but produce a separate subgraph for each confidence level

```
ciwidth onemean, level(90 95) n(30(5)45) probwidth(0.9) ///
  graph(bydimension(level))
```

Menu

Statistics > Power, precision, and sample size

Syntax

Produce default graph

```
ciwidth ..., graph ...
```

Graph sample size against CI width

```
ciwidth ..., graph(y(N) x(width)) ...
```

Graph sample size against probability of CI width

```
ciwidth ..., graph(y(N) x(Pr_width)) ...
```

Graph CI width against sample size

```
ciwidth ..., graph(y(width) x(N)) ...
```

Graph probability of CI width against sample size

```
ciwidth ..., graph(y(Pr_width) x(N)) ...
```

Produce other custom graphs

```
ciwidth ..., graph(graphopts) ...
```

graphopts	Description
Main	
`ydimension(`*dimlist* [`,` *dimopts*]`)`	use *dimlist* to define y axis
`xdimension(`*dimlist* [`,` *dimopts*]`)`	use *dimlist* to define x axis
`plotdimension(`*dimlist* [`,` *dimopts*]`)`	create plots for groups in *dimlist*
`bydimension(`*dimlist* [`,` *dimopts*]`)`	create subgraphs for groups in *dimlist*
`graphdimension(`*dimlist* [`,` *dimopts*]`)`	create graphs for groups in *dimlist*
`horizontal`	swap x and y axes
`schemegrid`	do not apply default x and y grid lines
`name(`*name* | *stub* [`, replace`]`)`	name of graph, or *stub* if multiple graphs
Labels	
`yregular`	place regularly spaced ticks and labels on the y axis
`xregular`	place regularly spaced ticks and labels on the x axis
`yvalues`	place ticks and labels on the y axis for each distinct value
`xvalues`	place ticks and labels on the x axis for each distinct value
`collabels(`*colspec*`)`	change default labels for columns
`nolabels`	label groups with their values, not their labels
`allsimplelabels`	forgo column label and equal signs in all labels
`nosimplelabels`	include column label and equal signs in all labels
`eqseparator(`*string*`)`	replace equal sign separator with *string*
`separator(`*string*`)`	separator for labels when multiple columns are specified in a dimension
`noseparator`	do not use a separator
`format(`*%fmt*`)`	format for converting numeric values to labels
Plot	
`plotopts(`*plot_options*`)`	affect rendition of all plots
`plot#opts(`*plot_options*`)`	affect rendition of #th plot
`recast(`*plottype*`)`	plot all plots using *plottype*
Add plots	
`addplot(`*plot*`)`	add other plots to the generated graph
Y axis, X axis, Titles, Legend, Overall, By	
twoway_options	any options documented in [G-3] ***twoway_options***
`byopts(`*byopts*`)`	how subgraphs are combined, labeled, etc.

dimlist may contain any of the following columns:

column	Description
`level`	confidence level
`alpha`	significance level
`N`	total number of subjects
`N1`	number of subjects in the control group
`N2`	number of subjects in the experimental group
`nratio`	ratio of sample sizes, experimental to control
`Pr_width`	probability of CI width
`width`	CI width
method_columns	columns specific to the method specified with `ciwidth`

colspec is

column "*label*" [*column* "*label*" [...]]

dimopts	Description
`labels(`*lablist*`)`	list of quoted strings to label each level of the dimension
`elabels(`*elablist*`)`	list of enumerated labels
`nolabels`	label groups with their values, not their labels
`allsimplelabels`	forgo column name and equal signs in all labels
`nosimplelabels`	include column name and equal signs in all labels
`eqseparator(`*string*`)`	replace equal sign separator with *string* in the dimension
`separator(`*string*`)`	separator for labels when multiple columns are specified in the dimension
`noseparator`	do not use a separator
`format(`*%fmt*`)`	format for converting numeric values to labels

where *lablist* is defined as

"*label*" ["*label*" [...]]

elablist is defined as

"*label*" [# "*label*" [...]]

and the #s are the levels of the dimension.

plot_options	Description
marker_options	change look of markers (color, size, etc.)
marker_label_options	add marker labels; change look or position
cline_options	change look of the line

Suboptions

The following are suboptions within the graph() option of the ciwidth command.

Main

ydimension(), xdimension(), plotdimension(), bydimension(), and graphdimension() specify the dimension to be used for the graph's y axis, x axis, plots, by() subgraphs, and graphs.

The default dimensions are based on your analysis. The y dimension is the parameter being estimated: sample size, CI width, or probability of CI width. If there is only one column containing multiple values, this column is plotted on the x dimension. Otherwise, the x dimension is CI width for sample-size determination and sample size when estimating CI width and probability of CI width. Other columns that contain multiple values are used as plot dimensions. See *Default graphs* below for details. You may override the defaults and explicitly control which columns are used on each dimension of the graph using these dimension suboptions.

Each of these suboptions supports suboptions that control the labeling of the dimension—axis labels for ydimension() and xdimension(), plot labels for plotdimension(), subgraph titles for bydimension(), and graph titles for graphdimension().

For examples using the dimension suboptions, see *Changing default graph dimensions* below.

ydimension(*dimlist* [, *dimopts*]) specifies the columns for the y axis in *dimlist* and controls the content of those labels with *dimopts*.

xdimension(*dimlist* [, *dimopts*]) specifies the columns for the x axis in *dimlist* and controls the content of those labels with *dimopts*.

plotdimension(*dimlist* [, *dimopts*]) specifies in *dimlist* the columns whose levels determine the plots and optionally specifies in *dimopts* the content of the plots' labels.

bydimension(*dimlist* [, *dimopts*]) specifies in *dimlist* the columns whose levels determine the by() subgraphs and optionally specifies in *dimopts* the content of the subgraphs' titles.

graphdimension(*dimlist* [, *dimopts*]) specifies in *dimlist* the columns whose levels determine the graphs and optionally specifies in *dimopts* the content of the graphs' titles.

See the definition of *columns in graph* in [PSS-5] **Glossary**.

horizontal reverses the default x and y axes. By default, the values computed by ciwidth are plotted on the y axis, and the x axis represents one of the other columns. Specifying horizontal swaps the axes.

One common use is to put sample size on the x axis even when it is the value computed by ciwidth. This suboption can also be useful with the long labels produced when the parallel option is specified with ciwidth.

See *Parallel plots* below for an example of the horizontal suboption.

schemegrid specifies that x and y grid lines not always be drawn on the ciwidth graph. Instead, whether grid lines are drawn will be determined by the current scheme.

name(*name* | *stub* [, replace]) specifies the name of the graph or graphs. If the graphdimension() suboption is specified, then the argument of name() is taken to be *stub*, and graphs named *stub*1, *stub*2, ... are created.

replace specifies that existing graphs of the same name may be replaced.

If `name()` is not specified, default names are used, and the graphs may be replaced by subsequent `ciwidth` graphs or other graphing commands.

Labels

All the suboptions listed under the **Labels** tab may be specified directly within the `graph()` option. All of them except `yregular`, `xregular`, `yvalues`, and `xvalues` may be specified as *dimopts* within `ydimension()`, `xdimension()`, `plotdimension()`, `bydimension()`, and `graphdimension()`. When suboptions are specified in one of the dimension options, only the labels for that dimension are affected. When suboptions are specified outside the dimension options, all labels on all dimensions are affected. Specifications within the dimension options take precedence.

`yregular` and `yvalues` specify how tick marks and labels are to be placed on the y axis.

`yregular` specifies that regularly spaced ticks and labels be placed on the y axis.

`yvalues` specifies that a tick and label be placed for each distinct value.

If neither is specified, an attempt is made to choose the most reasonable option based on your results. Labeling may also be specified using the standard `graph twoway` axis labeling rules and options.

`xregular` and `xvalues` do the same for tick marks and labels to be placed on the x axis.

`collabels(`*colspec*`)` specifies labels to be used on the graph for the specified columns. For example, `collabels(N "N")` specifies that wherever the column `N` is used on a graph—axis label, plot label, graph title, legend title, etc.—"N" be shown rather than the default label "Sample size".

Multiple columns may be relabeled by typing, for example,

```
collabels(N "N" v "Variance")
```

and SMCL tags for Greek characters and other typesetting can be used by typing, for example,

```
collabels(alpha "{&alpha}" N1 "N{sub:1}")
```

See the definition of *columns in graph* in [PSS-5] **Glossary**.

`nolabels` specifies that value labels not be used to construct graph labels and titles for the levels in the dimension. By default, if a column in a dimension has value labels, those labels are used to construct labels and titles for axis ticks, plots, subgraphs, and graphs.

`allsimplelabels` and `nosimplelabels` control whether a graph's labels and titles include just the values of the columns or also include column labels and equal signs. The default depends on whether the dimension is an axis dimension or one of the plot, by, and graph dimensions. It also depends on whether the values for the level of the dimension are labeled. An example of a simple label is "alpha" or ".05" and of a nonsimple label is "alpha=.05".

In `ciwidth, graph` simple labels are almost universally best for x and y axes and also best for most plot labels. Labels with an equal sign are typically preferred for subgraph and graph titles. These are the defaults used by `ciwidth, graph`. The `allsimplelabels` and `nosimplelabels` suboptions let you override the default labeling.

`allsimplelabels` specifies that all titles and labels use just the value or value label of the column.

`nosimplelabels` specifies that all titles and labels include *dimname*`=` before the value or value label.

`eqseparator(`*string*`)` specifies a custom separator between column labels and values in labels. Use *string* in place of the default equal sign. This option is for use with `nosimplelabels`.

`separator(`*string*`)` and `noseparator` control the separator between label sections when more than one column is used to specify a dimension. The default separator is a comma followed by a space, but no separator may be requested with `noseparator`, or the default may be changed to any string with `separator()`.

For example, if `bydimension(a b)` is specified, the subgraph labels in our graph legend might be "a=1, b=1", "a=1, b=2", ... Specifying `separator(:)` would create labels "a=1:b=1", "a=1:b=2", ...

`format(`*%fmt*`)` specifies how numeric values are to be formatted for display as axis labels, labels on plots, and titles on subgraphs and graphs.

Plot

`plotopts(`*plot_options*`)` affects the rendition of all plots. The *plot_options* can affect the size and color of markers, whether and how the markers are labeled, and whether and how the points are connected; see [G-3] ***marker_options***, [G-3] ***marker_label_options***, and [G-3] ***cline_options***.

These settings may be overridden for specific plots by using the `plot#opts()` suboption.

`plot#opts(`*plot_options*`)` affects the rendition of the #th plot. The *plot_options* can affect the size and color of markers, whether and how the markers are labeled, and whether and how the points are connected; see [G-3] ***marker_options***, [G-3] ***marker_label_options***, and [G-3] ***cline_options***.

`recast(`*plottype*`)` specifies that results be plotted using *plottype*. *plottype* may be `scatter`, `line`, `connected`, `area`, `bar`, `spike`, `dropline`, or `dot`; see [G-2] **graph twoway**. When `recast()` is specified, the plot-rendition options appropriate to the specified *plottype* may be used in lieu of *plot_options*. For details on those suboptions, follow the appropriate link from [G-2] **graph twoway**.

You may specify `recast()` within a `plotopts()` or `plot#opts()` suboption. It is better, however, to specify it as documented here, outside those suboptions. When it is specified outside those suboptions, you have greater access to the plot-specific rendition suboptions of your specified *plottype*.

Add plots

`addplot(`*plot*`)` provides a way to add other plots to the generated graph; see [G-3] ***addplot_option***.

If multiple graphs are drawn by a single `ciwidth` command or if *plot* specifies plots with multiple y variables, for example, `scatter y1 y2 x`, then the graph's legend will not clearly identify all the plots and will require customization using the `legend()` suboption; see [G-3] ***legend_options***.

Y axis, X axis, Titles, Legend, Overall, By

twoway_options are any of the options documented in [G-3] ***twoway_options***. These include options for titling the graph (see [G-3] ***title_options***); for saving the graph to disk (see [G-3] ***saving_option***); for controlling the labeling and look of the axes (see [G-3] ***axis_options***); for controlling the look, contents, position, and organization of the legend (see [G-3] ***legend_options***); for adding lines (see [G-3] ***added_line_options***) and text (see [G-3] ***added_text_options***); and for controlling other aspects of the graph's appearance (see [G-3] ***twoway_options***).

The `label()` suboption of the `legend()` option has no effect on `ciwidth, graph`. Use the `order()` suboption instead.

`byopts(`*byopts*`)` affects the appearance of the combined graph when `bydimension()` is specified or when the default graph has subgraphs, including the overall graph title, the position of the legend, and the organization of subgraphs. See [G-3] ***by_option***.

Remarks and examples

ciwidth, graph produces sample-size curves and other graphical output from the ciwidth command. These graphs are useful for visualizing the results of sensitivity analysis, which investigates the effect of varying study parameters on sample size and CI width. The true values of study parameters are usually unknown. PrSS uses best guesses for these values. It is important to evaluate the sensitivity of the computed sample size to the chosen values of study parameters. For example, to evaluate variability of sample-size values, you can compute sample sizes for various ranges of values for the parameters of interest and display the resulting sample sizes in a table (see [PSS-3] **ciwidth, table**) or plot them on a graph.

Remaining remarks are presented under the following headings:

Using ciwidth, graph

In most cases, you will probably be satisfied with the graphs that ciwidth produces by default when you specify the graph option. For other cases, ciwidth, graph() offers many options for you to produce the graph you desire.

Think of ciwidth, graph() as graphing the columns of ciwidth, table. One of the columns will be placed on the x axis, another will be placed on the y axis, and, if you have more columns with varying values, separate plots will be created for each. Similarly, we use the terms "column symbol", "column name", and "column label" to refer to symbols, names, and labels that appear in tables when tabular output is requested.

By default, ciwidth, graph plots the column corresponding to the estimated parameter on the y axis: N, when sample size is computed; width, when CI width is computed; and Pr_width when probability of CI width is computed. When there is only one varying column, the x axis uses this column by default. When there are multiple varying columns, the default x axis depends on what is being computed.

If sample size is computed (sample-size determination), the default x axis is CI width if CI width varies. If CI width does not vary, the sample size is plotted against one of the other varying parameters.

If CI width is computed (precision determination), the default x axis is the sample size if sample size varies. If the sample size does not vary, CI width is plotted against one of the other varying parameters.

If probability of CI width is computed, the default x axis is the sample size if sample size varies. If the sample size does not vary, probability of CI width is plotted against one of the other varying parameters.

ciwidth, graph() provides great flexibility for customizing graphical output. You can make minor changes such as modifying the graph or axes titles or modifying the line color and style, or you can completely redesign the graph by changing the axes and style of the graph. The Graph Editor can also be used for additional customization; see [G-1] **Graph Editor**.

When you produce a graph, the table of results is suppressed. You can request that the table be displayed in addition to the graph by specifying the table option with graph().

Graph symbols

Whenever space allows, such as on y and x axes, graphical output displays *extended column labels*, which include column labels and column symbols in parentheses. In other cases, such as in legend labels or by graph titles, graphical output includes only column (parameter) symbols for readability.

The following common symbols are used. See the documentation entry of the specified `ciwidth` method for additional symbols specific to that method.

Symbol	Description
$100(1-\alpha)$	confidence level
α	significance level
N	total sample size
N_1	sample size of the control group
N_2	sample size of the experimental group
N_2/N_1	ratio of sample sizes, experimental to control
p_{width}	probability of CI width
w	CI width
method_symbols	symbols specific to the *method* specified with `ciwidth`

Default graphs

We start with a demonstration of several default graphs and then show how you can produce custom graphs in the subsequent sections.

In what follows, we graph the results of PrSS analysis for a one-mean CI; see [PSS-3] **ciwidth onemean**.

▷ Example 1: Sample-size curves

When we compute sample size given a range of CI widths, `ciwidth, graph` plots sample size on the y axis and CI width on the x axis.

```
. ciwidth onemean, width(0.5(0.25)2) probwidth(0.9) graph
```

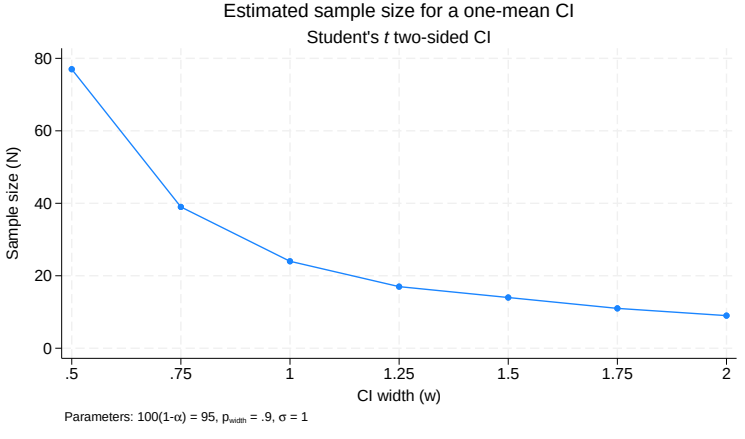


Figure 1.

As expected, sample size decreases as CI width increases.

The default axis labels include column labels and column symbols in parentheses. The labels can be changed, as we show in example 5. The values of constant parameters are displayed in the note titled "Parameters": confidence level $100(1-\alpha)$ is 95, probability of CI width p_{width} is 0.9, and standard deviation σ is 1.

In addition to varying CI width, we may compute sample sizes for different standard deviations.

```
. ciwidth onemean, width(0.5(0.25)2) probwidth(0.9) sd(1 2) graph
```

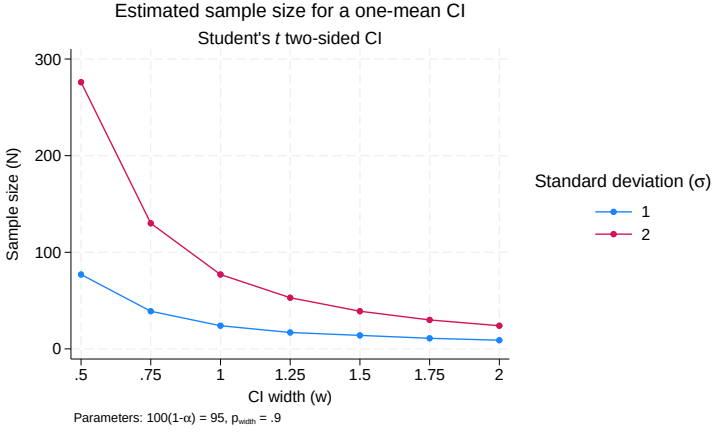


Figure 2.

For a given CI width, the larger the standard deviation, the larger the sample size.

`ciwidth, graph` displays two sample-size curves corresponding to the specified standard deviation values as shown on the legend. The first curve is displayed in navy, and the second curve is displayed in maroon. The default colors of the lines and, in general, the overall look of the graph are determined by the current graph scheme. The scheme used here is `stgcolor`; see [G-2] **set scheme** for details. We also show how to change the default look of the curves in example 6.

We can obtain sample-size curves for varying values of several parameters. `ciwidth, graph` plots a separate curve for each unique combination of the values of the parameters (except the parameter used as the x axis) on one plot. Alternatively, you can display curves on separate plots (`by` graphs) or even on separate graphs; see example 4.

If we specify only one CI width in the previous figure, the values of the standard deviation will be plotted on the x axis. You can try this yourself if you would like.

◁

▷ Example 2: CI precision curves

Instead of sample-size curves, we can plot estimated CI widths for a range of sample-size values to get an idea of how the sample size affects the CI precision.

```
. ciwidth onemean, n(10(2)40) probwidth(0.9) graph
```

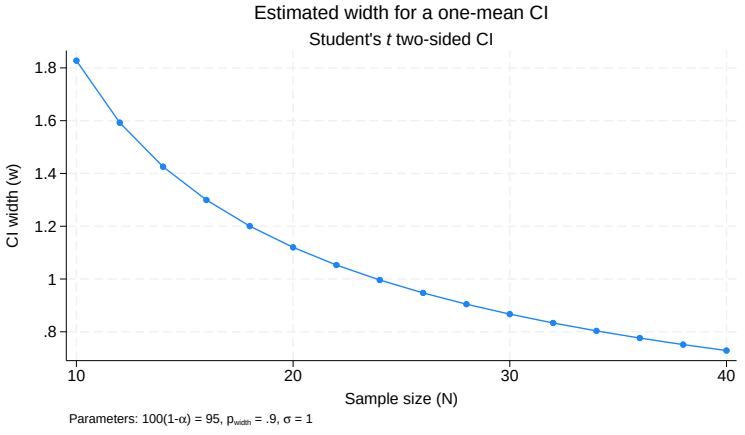


Figure 3.

The CI width decreases as the sample size increases. The larger the sample size, the more precise the CI.

This graph has the same overall look as figure 1, except CI width is plotted on the y axis and sample size is plotted on the x axis.

We may want to investigate how other study parameters, such as the standard deviation, affect the CI precision.

```
. ciwidth onemean, n(10(2)40) probwidth(0.9) sd(1 2) graph
```

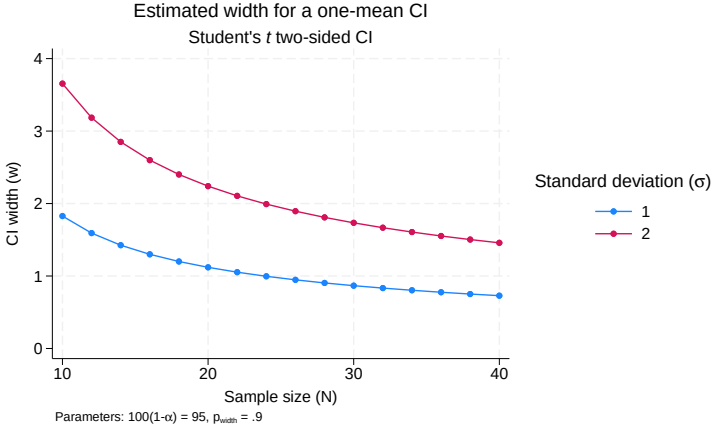


Figure 4.

The larger the standard deviation, the larger the CI width and thus the lower the CI precision.

When multiple study parameters each contain multiple values, as in the above figure, the default x axis for sample-size curves is the CI width, provided that the CI width parameter varies, and for precision curves is the sample size, provided it varies. You can plot a different parameter on the x axis, for instance the standard deviation; in example 3, we demonstrate how to change the default axes.

Changing default graph dimensions

So far, we have demonstrated the graphs that `ciwidth, graph` produces by default. In this section, we demonstrate how you can modify the default graphs with `ciwidth, graph()` and its suboptions.

▷ Example 3: Changing default graph axes

The default y axis corresponds to the computed study parameter—sample size for sample-size determination and CI width for precision determination. You would rarely need to change the dimensions when computing these parameters. On the other hand, probabilities are also plotted by default on the y axis when computing probability of CI width, but because of the likely limited range for this parameter, you may want to change the dimensions for these computations.

In figure 4, by default, the CI width is plotted against varying values of the sample size, and a separate curve is plotted for each of the varying values of the standard deviation. We can change the default x axis by specifying the `xdimension()` suboption (abbreviated to `x()`) within `ciwidth`'s `graph()` option. In this example, we specified fewer sample sizes and more standard deviations to obtain a more readable graph.

```
. ciwidth onemean, n(10 25 40) probwidth(0.9) sd(1(0.2)2) graph(x(sd))
```

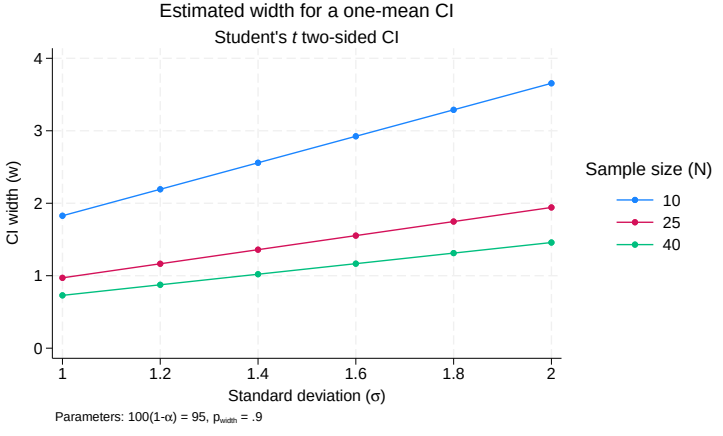


Figure 5.

The x axis now contains the values of the standard deviation, and a separate curve is now plotted for each sample size.

When the `xdimension()` suboption is specified, the x axis is replaced with the specified column, and the column corresponding to the default x axis is used as a plot dimension.

◁

Example 4: By graphs and multiple graphs

In figure 4, we plotted multiple CI precision curves corresponding to different standard deviation values on one graph. Alternatively, we can produce a separate plot for each of the standard deviation values by specifying the column `sd` in the `bydimension()` suboption (abbreviated to `by()`) within `ciwidth`'s `graph()` option.

```
. ciwidth onemean, n(10(2)40) probwidth(0.9) sd(1 2) graph(by(sd))
```

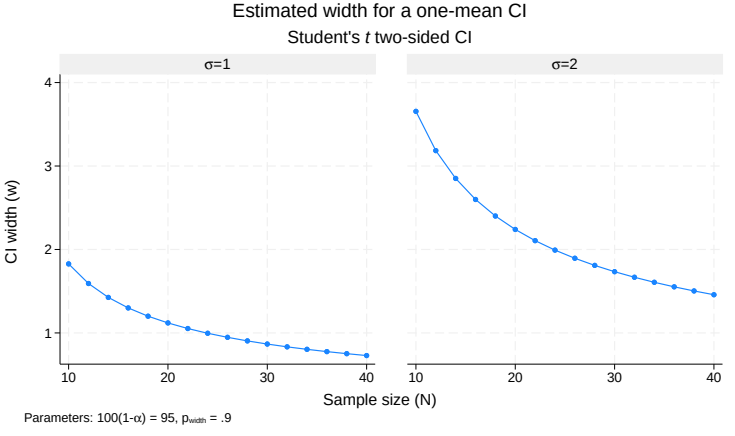


Figure 6.

For examples of how to modify the look of a by graph, see example 7.

In the presence of many varying parameters, even by graphs may look crowded. In this case, you may consider producing multiple by graphs. In the example above, suppose that we also want to vary the confidence level. We add the `level(90 95)` option to the previous command.

```
. ciwidth onemean, n(10(2)40) probwidth(0.9) sd(1 2) level(90 95) graph(by(sd))
  (output omitted)
```

The above command produces a graph containing two by graphs. Each by graph contains two plots, with each plot corresponding to unique values of the confidence level. We leave this for you to verify.

Instead, we can request that a separate graph be produced for each of the confidence levels by specifying the `graphdimension(level)` suboption (abbreviated to `graph()`) within `ciwidth`'s `graph()` option:

```
. ciwidth onemean, n(10(2)40) probwidth(0.9) sd(1 2) level(90 95)
> graph(by(sd) graph(level))
```

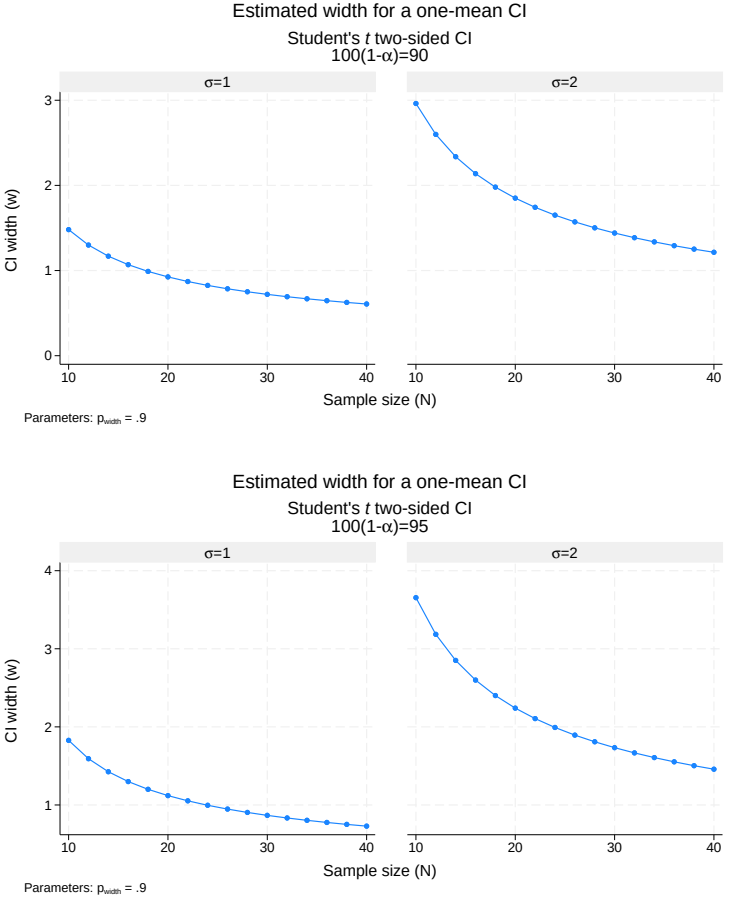


Figure 7.

Changing the look of graphs

Reasonable defaults for axis labels are chosen based on your results. You can modify the defaults by using any of `ciwidth, graph()`'s labeling suboptions or `graph twoway`'s *axis_label_options*; see [G-3] ***axis_label_options***.

▷ Example 5: Modifying axis labels

Rather than placing ticks and labels at equally spaced values, as in figure 3, we can request that ticks and labels be placed on the y and x axes for each distinct value.

```
. ciwidth onemean, n(10(2)20) probwidth(0.9)
> graph(yvalues xvalues ylabel(, format(%4.3f)))
```

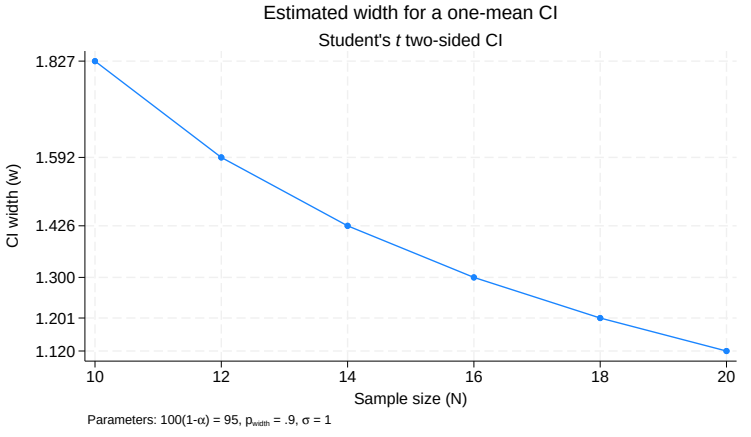


Figure 8.

In this example, to improve readability, we changed the default format of the values on the y axis to show only three decimal points by using `ylabel(, format(%4.3f))`.

We can use `ylabel()` and `xlabel()` to add text labels for some of the axis values. For example, suppose that our budget is 30 subjects. We can use `xlabel()` to label the sample-size value of 30 as "Budgeted".

```
. ciwidth onemean, n(10(2)40) probwidth(0.9) graph(xlabel(30 "Budgeted", add))
```

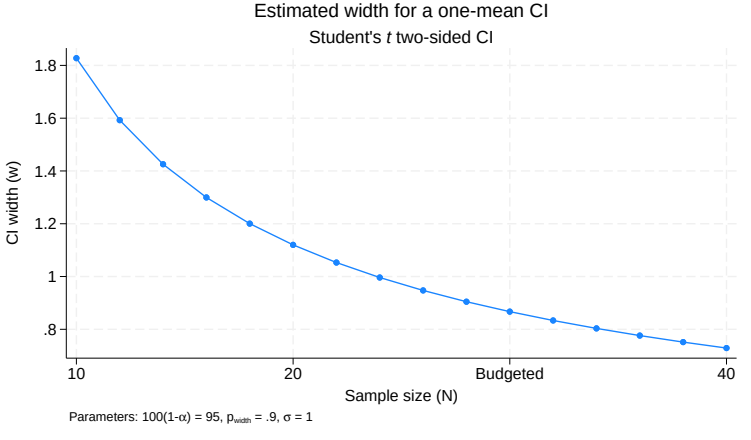


Figure 9.

We can use `ytitle()` and `xtitle()` to change the axis titles.

```
. ciwidth onemean, n(10(2)20) probwidth(0.9) graph(ytitle("CI width")
> xtitle("Sample size") title("Estimated width") subtitle("") note(""))
```

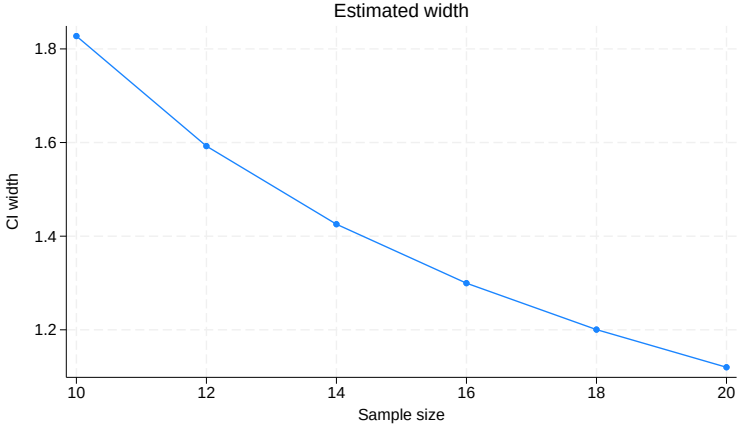


Figure 10.

In addition to modifying the axis titles, we also shortened the default title and suppressed the default subtitle and note.

You may find the `collabels()` suboption useful to override the default column labels. The specified column labels will be used wherever the corresponding column is used on the graph.

For example, change the default labels of the CI width and sample-size columns from figure 4 to "CI width" and "N", respectively, as follows:

```
. ciwidth onemean, n(10(2)40) sd(1 2) probwidth(0.9)
> graph(collabels(N "N" width "CI width"))
```

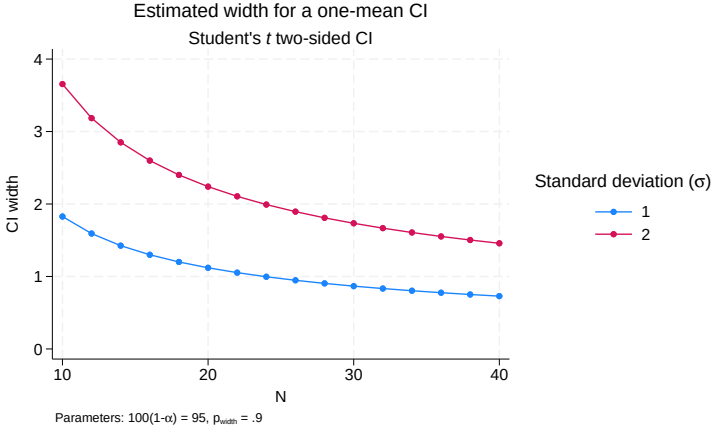


Figure 11.

When plotting multiple curves, by default, only the numeric values are used to label each plot in the legend. This is the simplest form of a label, but we can be more explicit and label each plot with the parameter symbol, an equal sign, and the value it takes on for each plot. We do this with the `nosimplelabels` suboption (abbreviated to `nosimple`) below. And to save space, we can suppress the legend title, which is included by default, by specifying an empty string.

```
. ciwidth onemean, n(10(2)40) sd(1 2) probwidth(0.9)
> graph(nosimple legend(title("")))
```

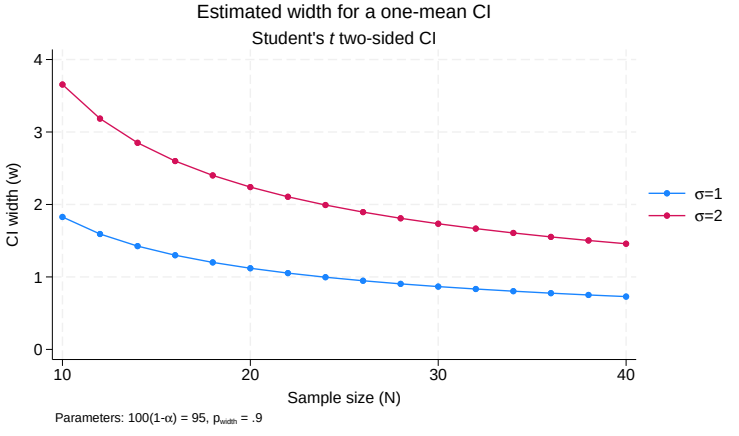


Figure 12.

The `nosimple` option will use an equal sign as the separator, but if we wanted to use another symbol (say, a colon), we could modify the command to something like the following:

```
ciwidth onemean, n(10(2)40) sd(1 2) probwidth(0.9) ///
 graph(nosimple legend(title("")) eqsep(" : "))
```

◁

▷ Example 6: Plot options

We can use the `plotopts()` and `plot#opts()` suboptions within `graph()` to modify the default look of the plotted lines. If there are multiple curves, the `plotopts()` suboption will apply changes to all curves. Use the corresponding `plot#opts()` suboption to change the look of only the #th curve. In the example below, we demonstrate how to use these suboptions.

We can label each data point on the graph with its corresponding sample-size value by specifying `mlabel()` within the `plotopts()` suboption, as shown below. We use `mlabpos()` to place the marker labels at the one o'clock position. See [G-3] ***marker_label_options*** for more details about these options.

```
. ciwidth onemean, width(0.5(0.5)2) probwidth(0.9)
> graph(plotopts(mlabel(N) mlabpos(1)))
```

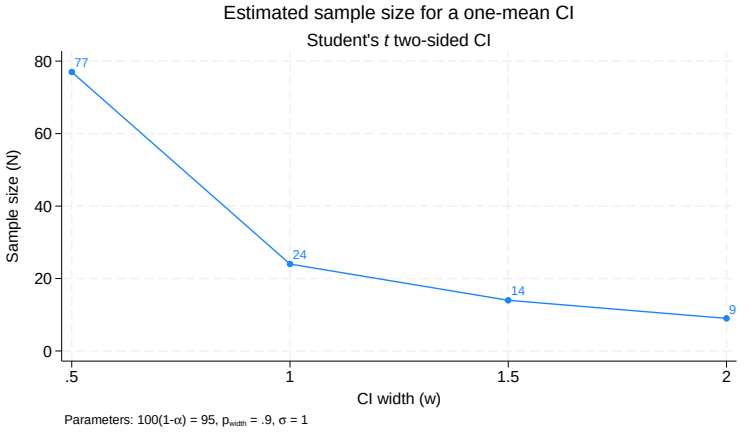


Figure 13.

For plots containing multiple curves, such as figure 4, the `plotopts()` suboption controls the look of all curves. For example, we can change the marker symbol from the default solid circle to a solid triangle.

```
. ciwidth onemean, n(10(2)40) probwidth(0.9) sd(1 2)
> graph(plotopts(msymbol(T)))
```

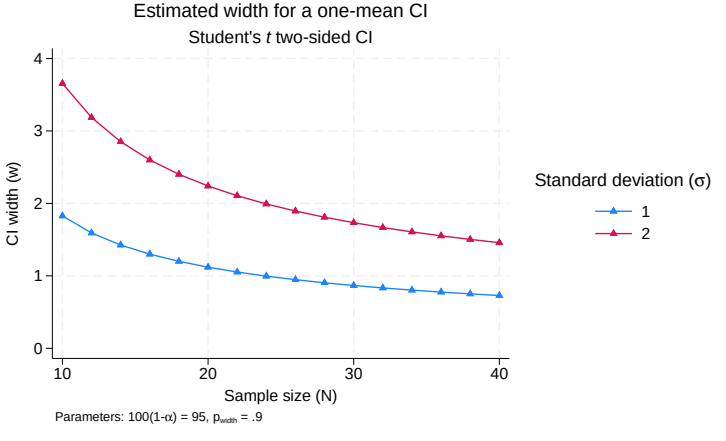


Figure 14.

To control the look of each curve, we can use multiple `plot#opts()` suboptions. For example, we can request that the curves corresponding to the same standard deviation be plotted using the same color:

```
. ciwidth onemean, n(10(2)40) probwidth(0.9) sd(1 2) level(90 95)
> graph(plot3opts(color(navy)) plot4opts(color(maroon)))
```

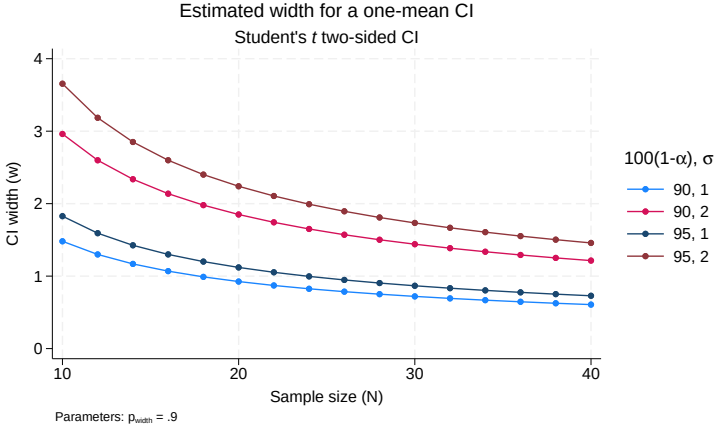


Figure 15.

Example 7: Modifying the look of by-graphs

The look of by graphs is controlled by the `byopts()` suboption specified within `ciwidth`'s `graph()` option.

In figure 6, we plotted the CI width for two standard deviation values using the same y axis. To allow the scales of the two by graphs to differ, we specify `yrescale` within the `byopts()` suboption.

```
. ciwidth onemean, n(10(2)40) probwidth(0.9) sd(1 2)
> graph(by(sd) byopts(yrescale))
```

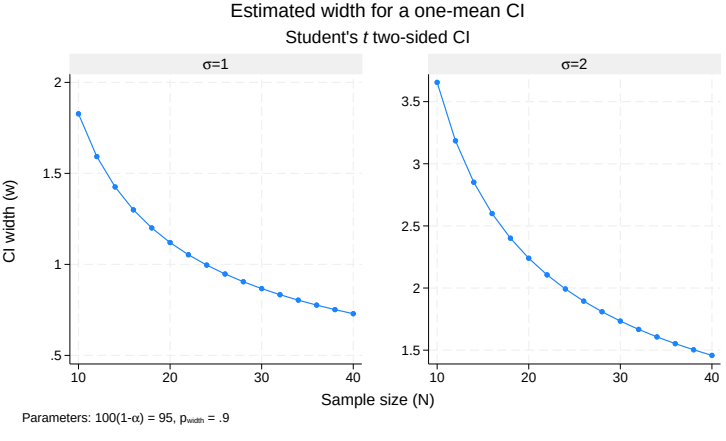


Figure 16.

We can also specify suboptions within `byopts()` to change the overall graph title and subtitle.

```
. ciwidth onemean, n(10(2)40) probwidth(0.9) sd(1 2)
> graph(by(sd) byopts(yrescale title("Width vs sample size") subtitle("")))
```

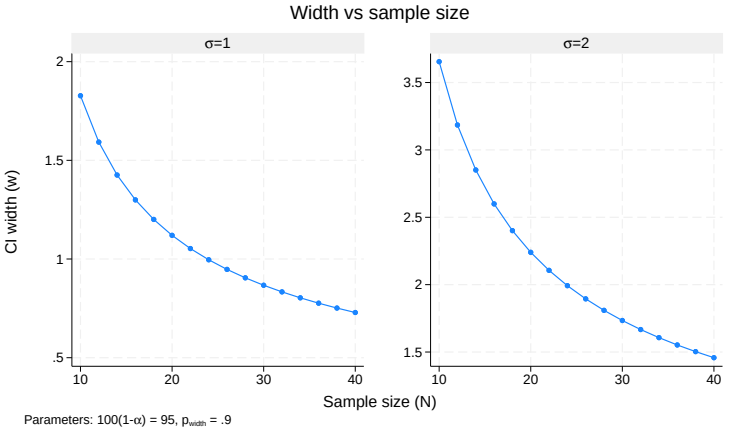


Figure 17.

Note that if you use `title()` and `subtitle()` outside `byopts()`, you will change the title and subtitle of the individual by graphs and not the overall graph.

We could vary the probability of CI width by typing

```
ciwidth onemean, n(10(2)40) probwidth(0.7 0.9) sd(1 2)  ///
  graph(by(sd) byopts(yrescale title("Width vs sample size") subtitle("")))
```

so that a legend indicating which line corresponds to each probability of CI width would appear on the right side of the graph. In by-graphs, this legend can make each graph appear tall and narrow. We can then include `scheme(stcolor_alt)` within the `graph()` option to create this graph using a scheme that puts the legend at the bottom, allowing more width for each graph.

```
. ciwidth onemean, n(10(2)40) probwidth(0.7 0.9) sd(1 2)
> graph(by(sd) scheme(stcolor_alt)
> byopts(yrescale title("Width vs sample size") subtitle("")))
```

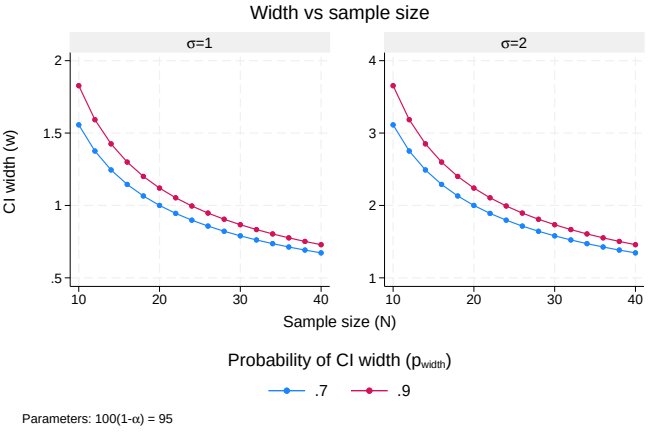


Figure 18.

Parallel plots

Sometimes, you may be interested in comparing sample sizes for parallel sets of parameters, that is, parameters that vary in parallel instead of being nested. In this situation, the results represent a collection of data points rather than a curve, and they are displayed on the graph as a scatterplot without connecting points.

For such parallel plots, the default display of the results on the x axis may be cumbersome. A more appealing look may be a graph that swaps the y and x axes, the horizontal graph. Such a look may be achieved by specifying the `horizontal` suboption within `graph()`.

```
. ciwidth onemean, width(0.5(0.1)2) probwidth(0.9) sd(1(0.1)1.9)
> parallel graph(x(width sd) horizontal nosimplelabels ytitle(""))
```

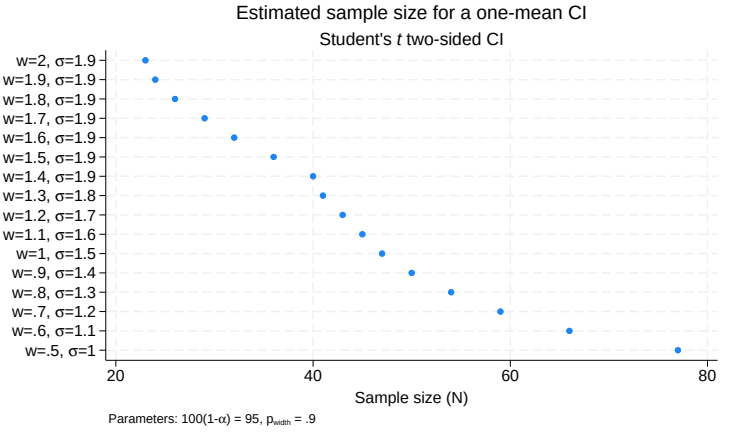


Figure 19.

To improve the look of the horizontal graph, we specified the `nosimplelabels` suboption to request that the labels on the y axis include the parameter symbol; we also suppressed the y-axis title.

Also see

[PSS-3] **ciwidth** — Precision and sample-size analysis for CIs

[PSS-3] **ciwidth, table** — Produce table of results from the ciwidth command

ciwidth, table — Produce table of results from the ciwidth command

Description

`ciwidth, table` displays results in a tabular format. `table` is implied if any of the `ciwidth` command's arguments or options contain more than one element. The `table` option is useful if you are producing graphs and would like to see the table as well or if you are producing results one case at a time using a loop and wish to display results in a table. The `notable` option suppresses table results; it is implied with the graphical output of `ciwidth, graph`; see [PSS-3] **ciwidth, graph**.

Quick start

Sample size required to achieve a target CI width of 1 with a 90% probability for a one-sample mean, in tabular format

```
ciwidth onemean, width(1) probwidth(0.9) table
```

Same as above, but change column labels of `N` and `sd` to `Sample size` and `Std. dev.`, respectively

```
ciwidth onemean, width(1) probwidth(0.9) ///
  table(, labels(N "Sample size" sd "Std. dev."))
```

Menu

Statistics > Power, precision, and sample size

Syntax

Produce default table

```
ciwidth ..., table ...
```

Suppress table

```
ciwidth ..., notable ...
```

Produce custom table

```
ciwidth ..., table([colspec] [, tableopts]) ...
```

where *colspec* is

```
column[ :label] [column[ :label] [...]]
```

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

tableopts	Description
Table	
`add`	add *columns* to the default table
`labels(`*labspec*`)`	change default labels for specified columns; default labels are column names
`widths(`*widthspec*`)`	change default column widths; default is specific to each column
`formats(`*fmtspec*`)`	change default column formats; default is specific to each column
`noformat`	do not use default column formats
`separator(#)`	draw a horizontal separator line every # lines; default is `separator(0)`, meaning no separator lines
`divider`	draw divider lines between columns
`byrow`	display rows as computations are performed; seldom used
`noheader`	suppress table header; seldom used
`continue`	draw a continuation border in the table output; seldom used

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`noheader` and `continue` are not shown in the dialog box.

column	Description
`level`	confidence level
`alpha`	significance level
`N`	total number of subjects
`N1`	number of subjects in the control group
`N2`	number of subjects in the experimental group
`nratio`	ratio of sample sizes, experimental to control
`Pr_width`	probability of CI width
`width`	CI width
`_all`	display all supported columns
method_columns	columns specific to the method specified with `ciwidth`

By default, the following columns are displayed:

`level`, `width`, and `N` are always displayed;

`N1` and `N2` are displayed for two-sample methods;

additional columns specific to each `ciwidth` method may be displayed.

Suboptions

The following are suboptions within the `table()` option of the `ciwidth` command.

Table

`add` requests that the columns specified in *colspec* be added to the default table. The columns are added to the end of the table.

`labels(`*labspec*`)` specifies the labels to be used in the table for the specified columns. *labspec* is

column "*label*" [*column* "*label*" [...]]

`labels()` takes precedence over the specification of column labels in *colspec*.

`widths(`*widthspec*`)` specifies column widths. The default values are the widths of the default column formats plus one. If the `noformat` option is used, the default for each column is nine. The column widths are adjusted to accommodate longer column labels and larger format widths. *widthspec* is either a list of values including missing values (*numlist*) or

column # [*column* # [...]]

For the value-list specification, the number of specified values may not exceed the number of columns in the table. A missing value (`.`) may be specified for any column to indicate the default width. If fewer widths are specified than the number of columns in the table, the last width specified is used for the remaining columns.

The alternative column-list specification provides a way to change widths of specific columns.

`formats(`*fmtspec*`)` specifies column formats. The default is `%7.0gc` for integer-valued columns and `%7.4g` for real-valued columns. *fmtspec* is either a string value-list of formats that may include empty strings or a column list:

column "*fmt*" [*column* "*fmt*" [...]]

For the value-list specification, the number of specified values may not exceed the number of columns in the table. An empty string (`""`) may be specified for any column to indicate the default format. If fewer formats are specified than the number of columns in the table, the last format specified is used for the remaining columns.

The alternative column-list specification provides a way to change formats of specific columns.

`noformat` requests that the default formats not be applied to the column values. If this suboption is specified, the column values are based on the column width.

`separator(#)` specifies how often separator lines should be drawn between rows of the table. The default is `separator(0)`, meaning that no separator lines should be displayed.

`divider` specifies that divider lines be drawn between columns. The default is no dividers.

`byrow` specifies that table rows be displayed as computations are performed. By default, the table is displayed after all computations are performed. This suboption may be useful when the computation of each row of the table takes a long time.

The following suboptions are available but are not shown in the dialog box:

`noheader` prevents the table header from displaying. This suboption is useful when the command is issued repeatedly, such as within a loop.

`continue` draws a continuation border at the bottom of the table. This suboption is useful when the command is issued repeatedly, such as within a loop.

Remarks and examples

Remarks are presented under the following headings:

`ciwidth, table` displays results from the `ciwidth` command in a table. This is useful for sensitivity analysis, which investigates the effect of varying study parameters on CI precision, sample size, or other components of the study. The true values of study parameters are usually unknown. PrSS analysis uses best guesses for these values. It is important to evaluate the sensitivity of the computed CI precision or sample size to the chosen values of study parameters. For example, to evaluate variability of CI width, you can compute CI widths for various ranges of values for the parameters of interest and display the resulting widths in a table or plot them on a graph (see [PSS-3] **ciwidth, graph**).

Using ciwidth, table

If you specify the `table` option or include more than one element in command arguments or in options allowing multiple values, the `ciwidth` command displays results in a tabular form. If desired, you can suppress the table by specifying the `notable` option. The `table` option is useful if you are producing graphical output or if you are producing results one case at a time, such as within a loop, and wish to display results in a table; see example 4 below.

Each method specified with the `ciwidth` command has its own default table. Among the columns that are always included in the default table are confidence level (`level`), CI width (`width`), and total sample size (`N`).

Depending on the method and study design, additional columns are also included by default. For example, `ciwidth onemean` has an additional column for standard deviation.

You can build your own table by specifying the columns and, optionally, their labels in the `table()` option. You can also add columns to the default table by specifying `add` within `ciwidth`'s `table()` option. The columns are displayed in the order they are specified. Each method provides its own list of supported columns; see the description of the `table()` option for each method. You can further customize the table by specifying various suboptions within `ciwidth`'s `table()` option.

The default column labels are the column names. You can provide your own column labels in *colspec* or by specifying `table()`'s suboption `labels()`. Labels containing spaces should be enclosed in quotes, and labels containing quotes should be enclosed in compound quotes. The `labels()` suboption is useful for changing the labels of existing columns; see example 2 below for details.

The default formats are `%7.4g` for real-valued columns and `%7.0gc` for integer-valued columns. If the `noformat` suboption is specified, the default column widths are nine characters. You can use `formats()` to change the default column formats and `widths()` to change the default column widths. The `formats()` and `widths()` suboptions provide two alternative specifications, a value-list specification or a column-list specification. The value-list specification accepts a list of values—strings for formats and numbers for widths—corresponding to each column of the displayed table. Empty strings (`""`) for formats and missing values (`.`) for widths are allowed and denote the default values. It is an error to specify more values than the number of displayed columns. If fewer values are specified, then the last value specified is used for the remaining columns. The column-list specification includes a list of pairs containing a column name followed by the corresponding value of the format or width. This specification is useful if you want to modify the formats or the widths of only selected columns. For column labels or formats exceeding the default column width, the widths of the respective columns are adjusted to accommodate the column labels and the specified formats.

If you specify the `noformat` suboption, the default formats are ignored, and the format of a column is determined by the column width: if the column width is #, the displayed format is `%(#-2).0g`. For example, if the column width is 9, the displayed format is `%7.0g`.

You may further customize the look of the table by using `separator(#)` to include separator lines after every # lines and by using the `divider` suboption to include divider lines between columns.

The `noheader` and `continue` suboptions are useful when you are building your own table within a loop; see example 4 in *Custom tables*.

In what follows, we demonstrate the default and custom tables of the results from PrSS analysis for a one-mean CI and a one-variance CI; see [PSS-3] **ciwidth onemean** and [PSS-3] **ciwidth onevariance**.

Default tables

If there is only one set of results, the `ciwidth` command displays those results as text. When the `ciwidth` command has multiple sets of results, they are automatically displayed in a table. You can also specify the `table` option at any time to request that results be displayed in a table.

The displayed columns are specific to the chosen method of analysis and to the options specified with the command. The columns that always appear in the table include the confidence level (`level`), CI width (`width`), and total sample size (`N`).

▷ Example 1: Default tables from ciwidth onemean

Suppose we want to explore the required sample size to achieve a certain precision for a one-mean CI. Below we estimate the required sample size for obtaining the target CI widths no larger than 1, 2, and 3 with a probability of CI width of 0.9. See [PSS-3] **ciwidth onemean** for details.

```
. ciwidth onemean, width(1 2 3) probwidth(0.9) sd(2)
Performing iteration ...
Estimated sample size for a one-mean CI
Student's t two-sided CI
```

level	N	Pr_width	width	sd
95	77	.9	1	2
95	24	.9	2	2
95	14	.9	3	2

As we mentioned earlier, the `level`, `width`, and `N` columns are displayed in the default table. Column `Pr_width` is also displayed in the default table whenever `probwidth()` is specified. The `ciwidth onemean` command additionally displays the standard deviation column.

◁

Modifying default tables

We can modify labels, widths, and formats of the default columns by specifying the corresponding suboptions within the `table()` option. We can also add columns to the default table by using `table()`'s suboption `add`.

▷ Example 2: Modifying default tables from ciwidth onemean

We can change the default labels of all or selected columns by using the `labels()` suboption within `ciwidth`'s `table()` option. For example, we can change the labels of the sample-size columns and standard deviation columns of the first table in example 1 to "Sample size" and "Std. dev.", respectively.

```
. ciwidth onemean, width(1 2 3) probwidth(0.9) sd(2)
> table(, labels(N "Sample size" sd "Std. dev."))
Performing iteration ...
Estimated sample size for a one-mean CI
Student's t two-sided CI
```

level	Sample size	Pr_width	width	Std. dev.
95	77	.9	1	2
95	24	.9	2	2
95	14	.9	3	2

We can also change default column formats and widths by using the `formats()` and `widths()` suboptions.

```
. ciwidth onemean, width(1 2 3) probwidth(0.9) sd(2)
> table(, labels(N "Sample size" sd "Std. dev.") widths(N 14 sd 14)
> formats(width "%7.5f"))
Performing iteration ...
Estimated sample size for a one-mean CI
Student's t two-sided CI
```

level	Sample size	Pr_width	width	Std. dev.
95	77	.9	1.00000	2
95	24	.9	2.00000	2
95	14	.9	3.00000	2

For this table, we changed the default column widths of the sample-size and standard deviation columns to 14. We also changed the format of the width column from the default, `%7.4g`, to `%7.5f`.

◁

▷ Example 3: Modifying default tables from ciwidth onevariance

We can also add columns to the default table by using `table()`'s suboption `add`. In the `ciwidth onevariance` example below, the default columns are `level`, `N`, `Pr_width`, `width`, and `v`.

```
. ciwidth onevariance 1, width(1 2) probwidth(0.9)
Performing iteration ...
Estimated sample size for a one-variance CI
Chi-squared two-sided CI
```

level	N	Pr_width	width	v
95	57	.9	1	1
95	23	.9	2	1

We can also add the standard deviation column, `s`, to the table:

```
. ciwidth onevariance 1, width(1 2) probwidth(0.9) table(s, add)
Performing iteration ...
Estimated sample size for a one-variance CI
Chi-squared two-sided CI
```

level	N	Pr_width	width	v	s
95	57	.9	1	1	1
95	23	.9	2	1	1

◁

Custom tables

You can use the `table()` option to build custom tables, with the columns you want in the order you want. You can also build a table within a `foreach` or `forvalues` loop.

▷ Example 4: Building table using a loop

Some options of `ciwidth` commands may not allow the *numlist* specification. In this case, you can build a table manually by using a loop with either `foreach` (see [P] **foreach**) or `forvalues` (see [P] **forvalues**). One way to do this is to write a program that loops over parameters of interest. We demonstrate a program that loops over varying values of the variance of `ciwidth onevariance`. You can easily adapt this program to meet your needs.

```
program dotable
        args var
        numlist "`var'"                  // expand the numeric list in macro var
        local var "`r(numlist)'"
        local nvals : list sizeof var
        local i 1
        foreach val of local var {       // loop over numeric values in var
                if (`i'==1) {
                        ciwidth onevariance `val', width(2) probwidth(0.9) ///
        table(, continue)
                }
                else if (`i'<`nvals') {
                        ciwidth onevariance `val', width(2) probwidth(0.9) ///
      table(, noheader continue) notitle
                }
                else {
                        ciwidth onevariance `val', width(2) probwidth(0.9) ///
      table(, noheader) notitle
                }
                local ++i
        }
end
```

The `dotable` program accepts one argument, `var`, which may contain one or more numeric values of the variance specified as *numlist*. The program uses combinations of `continue`, `noheader`, and `notitle` to display a table. The first call to `ciwidth onevariance` requests that the table be displayed without the bottom line by specifying the `continue` suboption within `table()`. The subsequent calls (except the last) specify the `continue` suboption, the `notitle` option with `ciwidth onevariance`, and `noheader` within the `table()` option to request that neither the output before the table nor the table header be displayed. The last call omits the `continue` suboption so that the bottom line is displayed.

As a result, we obtain the following table:

```
. dotable "1(0.2)2"
Performing iteration ...
Estimated sample size for a one-variance CI
Chi-squared two-sided CI
```

level	N	Pr_width	width	v
95	23	.9	2	1
95	29	.9	2	1.2
95	35	.9	2	1.4
95	41	.9	2	1.6
95	49	.9	2	1.8
95	57	.9	2	2

Stored results

`ciwidth, table` stores the following in `r()` in addition to other results stored by `ciwidth`:

Scalars

`r(separator)`	number of lines between separator lines in the table
`r(divider)`	`1` if `divider` is requested in the table, `0` otherwise

Macros

`r(columns)`	displayed table columns
`r(labels)`	table column labels
`r(widths)`	table column widths
`r(formats)`	table column formats

Matrices

`r(pss_table)`	table of results

Also see

[PSS-3] **ciwidth** — Precision and sample-size analysis for CIs

[PSS-3] **ciwidth, graph** — Graph results from the ciwidth command

ciwidth onemean — Precision analysis for a one-mean CI

Description

`ciwidth onemean` computes sample size, CI width, and probability of CI width for a CI for a population mean. It can compute sample size for a given CI width and probability of CI width. Alternatively, it can compute CI width for a given sample size and probability of CI width. It can also compute probability of CI width for a given sample size and CI width. Also see [PSS-3] **ciwidth** for PrSS analysis for other CI methods.

For power and sample-size analysis for a one-sample mean test, see [PSS-2] **power onemean**.

Quick start

Sample size for a two-sided 95% CI for a population mean given a CI width of 5 and a standard deviation of 12, with the probability of CI width of 0.9

```
ciwidth onemean, width(5) probwidth(0.9) sd(12)
```

Same as above, but for an upper one-sided CI with a 90% confidence level

```
ciwidth onemean, width(5) probwidth(0.9) sd(12) level(90) upper
```

Sample size for a two-sided 95% CI for a population mean given a CI width of 5, assuming a known population standard deviation of 12

```
ciwidth onemean, width(5) sd(12) knownsd
```

CI width for sample sizes of 50, 60, 70, and 80, given a probability of CI width of 0.9

```
ciwidth onemean, n(50(10)80) probwidth(0.9) sd(12)
```

Same as above, but display results in a graph of CI width versus sample size

```
ciwidth onemean, n(50(10)80) probwidth(0.9) sd(12) graph
```

Probability that the width of a two-sided 95% CI is no larger than 5 for a sample size of 100

```
ciwidth onemean, n(100) width(5) sd(12)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
ciwidth onemean, width(numlist) probwidth(numlist) [ options ]
```

Compute CI width

```
ciwidth onemean, probwidth(numlist) n(numlist) [ options ]
```

Compute probability of CI width

```
ciwidth onemean, width(numlist) n(numlist) [ options ]
```

options	Description
Main	
* level(*numlist*)	confidence level; default is level(95)
* alpha(*numlist*)	significance level; default is alpha(0.05)
* probwidth(*numlist*)	probability of CI width; required to compute sample size and CI width
* width(*numlist*)	CI width; required to compute sample size and probability of CI width
* n(*numlist*)	sample size; required to compute CI width and probability of CI width
nfractional	allow fractional sample sizes
* sd(*numlist*)	standard deviation; default is sd(1)
knownsd	request computation assuming a known standard deviation; default is to assume an unknown standard deviation
* fpc(*numlist*)	finite population correction (FPC) as a sampling rate or as a population size
lower	lower one-sided CI; default is two-sided CI
upper	upper one-sided CI; default is two-sided CI
onesided	synonym for option upper
parallel	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)

Table

[no]table[(*tablespec*)]	suppress table or display results as a table; see [PSS-3] **ciwidth, table**
saving(*filename* [, replace])	save the table data to *filename*; use replace to overwrite existing *filename*

Graph

graph[(*graphopts*)]	graph results; see [PSS-3] **ciwidth, graph**

Iteration

init(#)	initial value for sample size; default is to use a closed-form normal approximation
iterate(#)	maximum number of iterations; default is iterate(500)
tolerance(#)	parameter tolerance; default is tolerance(1e-12)
ftolerance(#)	function tolerance; default is ftolerance(1e-12)
[no]log	suppress or display iteration log
[no]dots	suppress or display iterations as dots
notitle	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the parallel option.

collect is allowed; see **[U] 11.1.10 Prefix commands**.

notitle does not appear in the dialog box.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
level	confidence level	$100(1-\alpha)$
alpha	significance level	α
N	number of subjects	N
Pr_width	probability of CI width	p_{width}
width	CI width	w
sd	standard deviation	σ
fpc	FPC as population size	N_{pop}
	FPC as sampling rate	γ
_all	display all supported columns	

Column alpha is shown in the default table in place of column level if alpha() is specified.

Column fpc is shown in the default table if fpc() is specified.

Options

Main

`level()`, `alpha()`, `probwidth()`, `width()`, `n()`, `nfractional`; see [PSS-3] **ciwidth**. `probwidth()` may not be combined with `knownsd`. The `nfractional` option is allowed only for sample-size determination.

`sd(`*numlist*`)` specifies the population standard deviation or its estimate. The default is `sd(1)`. By default, `sd()` specifies an estimate for the unknown population standard deviation. If `knownsd` is specified, `sd()` specifies the known value for the population standard deviation.

`knownsd` requests that the standard deviation be treated as known in the computation. By default, the standard deviation is treated as unknown, and the computation is performed for a Student's t-based CI. If `knownsd` is specified, the computation is performed for a normal-based CI. `knownsd` may not be combined with `probwidth()` and is not allowed when computing the probability of CI width.

`fpc(`*numlist*`)` requests that a finite population correction be used in the computation. If `fpc()` has values between 0 and 1, it is interpreted as a sampling rate, n/N, where N is the total number of units in the population. When sample size n is specified, if `fpc()` has values greater than n, it is interpreted as a population size, but it is an error to have values between 1 and n. For sample-size determination, `fpc()` with a value greater than 1 is interpreted as a population size. It is an error for `fpc()` to have a mixture of sampling rates and population sizes.

`lower`, `upper`, `onesided`, `parallel`; see [PSS-3] **ciwidth**.

Table

`table`, `table()`, `notable`; see [PSS-3] **ciwidth, table**.

`saving()`; see [PSS-3] **ciwidth**.

Graph

`graph`, `graph()`; see [PSS-3] **ciwidth, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies an initial value for the sample size when iteration is used to compute the sample size. The default is to use a closed-form normal approximation to compute an initial sample size.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-3] **ciwidth**.

The following option is available with `ciwidth onemean` but is not shown in the dialog box:

`notitle`; see [PSS-3] **ciwidth**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `ciwidth onemean` command and the methodology for PrSS analysis for a CI for a population mean. See [PSS-3] **Intro (ciwidth)** for a general introduction to PrSS analysis, and see [PSS-3] **ciwidth** for a general introduction to the `ciwidth` command. For PSS analysis for hypothesis tests, see [PSS-2] **power**.

Introduction

Mean estimation is one of the most popular statistical analyses. Schools may measure the mean test score for their students. Factories may measure the mean strength of new alloys or the mean lifetime of bulbs. CIs are often used for inference about the population mean μ. They provide the ranges for plausible values for the mean based on a random sample from a population of interest. The wider the ranges, the less precise the CI.

The precision of a CI is commonly measured by its width w or, for a symmetric CI such as the CI for a population mean, by its half-width d, also known as the margin of error. For example, a two-sided one-mean CI is formed as $[\hat{\mu}-d, \hat{\mu}+d]$, where $\hat{\mu}$ is the mean point estimate. The CI width, the distance between the upper and lower limits, is $w = 2d$; it does not depend on the mean estimate. The smaller the d or w the more precise the CI.

In PrSS analysis, it is usually of interest to determine the sample size that would be sufficient for a CI to have a prespecified width in a future study. Generally, larger sample sizes lead to more precise CIs. To compute the required sample size, we need to know the expression for w. The expression for w depends on various assumptions.

Suppose that we have a random sample of n i.i.d. observations from a normal distribution with mean μ and standard deviation σ. If we know a population standard deviation σ, $w = 2z_{1-\alpha/2}\sigma/\sqrt{n}$, where $z_{1-\alpha/2}$ is the $(1-\alpha/2)$th quantile of the standard normal distribution. If we do not know a σ and estimate it from the sample using the sample standard deviation s, $w = 2t_{n-1,1-\alpha/2}\, s/\sqrt{n}$, where $t_{n-1,1-\alpha/2}$ is the $(1-\alpha/2)$th quantile of the Student's t distribution with $n-1$ degrees of freedom. $100(1-\alpha)\%$ is the confidence level of a CI. Using the relationship between CIs and hypothesis tests, a $100(1-\alpha)\%$ one-mean CI can be viewed as a set of hypothetical values of a mean that cannot be rejected by the corresponding one-sample mean test at the significance level α.

In the case of an unknown standard deviation, the CI width w depends on the sample estimate s of the standard deviation and thus will vary from one sample to another. To ensure that, in a future study, a CI has the desired width, this sampling variability of w must be accounted for when computing the required sample size. Kupper and Hafner (1989) introduce what we call the probability of CI width that specifies the probability of a future CI to have the width of no larger than some prespecified CI width for a given sample size. This probability is defined based on the assumption of a χ^2 distribution for the sample variance s^2; see *Methods and formulas* for details.

The random sample is typically drawn from an infinite population. When the sample is drawn from a population of a fixed size, sampling variability must be adjusted for a finite population size.

You can use `ciwidth onemean` to perform PrSS analysis for a CI for a population mean. We discuss the command details in the next section.

Using ciwidth onemean

`ciwidth onemean` computes sample size, CI width, or probability of CI width for a one-mean CI. By default, a two-sided CI is assumed, and the confidence level is set to 95%. You may change the confidence level by specifying the `level()` option. Alternatively, you can specify the significance level in the `alpha()` option. You can specify the `upper` and `lower` options to request upper and lower one-sided CIs.

To compute sample size, you must specify the CI width in the `width()` option and the probability of CI width in the `probwidth()` option. To compute CI width, you must specify the sample size in the `n()` option and the probability of CI width in the `probwidth()` option. You can also compute the probability of CI width given the sample size in `n()` and CI width in `width()`.

For CIs for means, the CI width does not depend on the mean point estimate, the sample mean, so it is not needed in the computations.

By default, all computations assume an unknown standard deviation and use a default value of 1 as the estimate of the standard deviation. You may specify other values for the standard deviation in the `sd()` option. For a known standard deviation, you can specify the `knownsd` option to request a normal-based CI instead of the default Student's t-based CI. In this case, the `sd()` option specifies the actual population standard deviation.

By default, the computed sample size is rounded up. You can specify the `nfractional` option to see the corresponding fractional sample size; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. The `nfractional` option is allowed only for sample-size determination.

Some of `ciwidth onemean`'s computations require iteration. For example, when the standard deviation is unknown, the sample-size computation requires iteration. The default initial value of the estimated sample size is obtained by using a closed-form normal approximation. It may be changed by specifying the `init()` option. See [PSS-3] **ciwidth** for the descriptions of other options that control the iteration procedure.

All computations assume an infinite population. For a finite population, use the `fpc()` option to specify a sampling rate or a population size.

In the following sections, we describe the use of `ciwidth onemean` accompanied by examples for computing sample size, CI width, and probability of CI width.

Computing sample size

To compute the sample size required for a one-mean CI to have the width no larger than a target width, you must specify the target CI width in the `width()` option and the desired probability of achieving the target CI width in the `probwidth()` option.

▷ Example 1: Sample size for a one-mean CI

Consider an example from Meeker, Hahn, and Escobar (2017, 152) of an experiment that measures the mean tensile strength of a new alloy. The number of specimens needed to be tested is of interest. We assume that the strength follows a normal distribution and the true population standard deviation is unknown. We use a conservative estimate of 2,500 kg for it in this example. We want to compute the sample size required for the two-sided 95% CI for the mean tensile strength to have the width of no larger than 3,000 kg. In addition to the sample size, the CI width depends on the standard deviation. Because we assume an unknown standard deviation, the CI width may vary from one study to another. We must

account for this sampling variability when estimating the sample size. So, to ensure that the CI will have the width of at most 3,000 kg, we must also specify the probability of achieving the target CI width. If we do not, our results will be based on the assumption that a future sample will have the standard deviation of 2,500 kg, which may not be a realistic assumption given an unknown standard deviation. In our example, we will use the probability of CI width of 0.96. To compute the sample size, we type

```
. ciwidth onemean, sd(2500) probwidth(0.96) width(3000)
Performing iteration ...
Estimated sample size for a one-mean CI
Student's t two-sided CI
Study parameters:
        level =     95.00
     Pr_width =    0.9600
        width = 3000.0000
           sd = 2500.0000
Estimated sample size:
            N =        20
```

We find that a sample of 20 specimens is required to obtain a two-sided 95% CI for the mean that, with a probability of 0.96, will have the width of no more than 3,000 kg.

As we mentioned in *Using ciwidth onemean* and as is also indicated in the output, sample-size computation requires iteration when the standard deviation is unknown. The iteration log is suppressed by default, but you can display it by specifying the `log` option.

◁

▷ Example 2: Known variance

If we know the population standard deviation, we can use the `knownsd` option to compute results for a normal z-based CI. Because we assume a known standard deviation, the `probwidth()` option is not needed. In fact, it is not allowed in combination with `knownsd`.

```
. ciwidth onemean, sd(2500) width(3000) knownsd
Estimated sample size for a one-mean CI
Normal two-sided CI
Study parameters:
        level =     95.00
        width = 3000.0000
           sd = 2500.0000
Estimated sample size:
            N =        11
```

The output now indicates that the computation is based on a normal CI instead of the default Student's t CI. We find that a smaller sample of 11 subjects is required for the mean CI to achieve the same CI width as in example 1 when the standard deviation is known.

◁

Computing CI width

To compute the CI width, you must specify the sample size in the `n()` option and the desired probability of achieving the target CI width in the `probwidth()` option.

▷ Example 3: Precision of a one-mean CI

Continuing with example 1, we can compute the CI width for a given sample size. Suppose we want to compute the CI width corresponding to the sample size of 20 from example 1. We replace the `width(3000)` option with the `n(20)` option:

```
. ciwidth onemean, sd(2500) probwidth(0.96) n(20)

Estimated width for a one-mean CI
Student's t two-sided CI

Study parameters:

        level =     95.00
            N =        20
     Pr_width =    0.9600
           sd = 2500.0000

Estimated width:

        width = 2990.8196
```

The estimated CI width is about 2,991 kg and is smaller than the width of 3,000 kg used in example 1. The actual sample size corresponding to the CI width of 3,000 is slightly smaller than 20. You can see this by specifying the `nfractional` option in example 1 to report a fractional sample size. By default, `ciwidth` methods round the actual sample size up to the nearest integer. The CI width corresponding to a larger sample size will be smaller.

◁

▷ Example 4: One-sided CI

You can specify the `upper` or `lower` option to request an upper or lower one-sided CI. Continuing with example 3, we estimate the CI width for an upper 95% CI.

```
. ciwidth onemean, sd(2500) probwidth(0.96) n(20) upper

Estimated width for a one-mean CI
Student's t upper CI

Study parameters:

        level =     95.00
            N =        20
     Pr_width =    0.9600
           sd = 2500.0000

Estimated width:

        width = 1235.4192
```

The upper bound for the mean tensile strength of a new alloy is about 1,235 kg from its point estimate.

◁

▷ Example 5: Multiple values of study parameters

To investigate the effect of the sample size on the CI width, we can specify a list of sample sizes in the `n()` option:

```
. ciwidth onemean, sd(2500) probwidth(0.96) n(10(10)50)
Estimated width for a one-mean CI
Student's t two-sided CI
```

level	N	Pr_width	width	sd
95	10	.96	5003	2500
95	20	.96	2991	2500
95	30	.96	2289	2500
95	40	.96	1912	2500
95	50	.96	1669	2500

As expected, when the sample size increases, the CI width decreases. The decrease is particularly prominent when the sample size increases from 10 to 20.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-3] **ciwidth, table**. If you wish to produce sample-size and other curves, see [PSS-3] **ciwidth, graph**.

◁

Computing probability of CI width

To compute the probability that the width of a future CI will be no larger than the specified width, you must specify the sample size in the `n()` option and the target CI width in the `width()` option.

▷ Example 6: Computing probability of CI width for a one-mean CI

Continuing with example 1, we may also want to know the probability that the CI width in a future study will be no larger than a prespecified value for a given sample size. To compute the probability of CI width, we specify the sample size of 20 in `n()` and the CI width of 3,000 in `width()`. We use the same estimate of 2,500 for the standard deviation.

```
. ciwidth onemean, sd(2500) n(20) width(3000)
Estimated probability of width for a one-mean CI
Student's t two-sided CI
Study parameters:
        level =     95.00
            N =        20
        width = 3000.0000
           sd = 2500.0000
Estimated probability of width:
     Pr_width =    0.9619
```

The estimated probability is about 96%, which is consistent with what we used in example 1.

◁

Stored results

ciwidth onemean stores the following in r():

Scalars

r(level)	confidence level
r(alpha)	significance level
r(N)	sample size
r(nfractional)	1 if nfractional is specified, 0 otherwise
r(onesided)	1 for a one-sided CI, 0 otherwise
r(Pr_width)	probability of CI width
r(Pr_width_a)	actual probability of CI width (for sample-size determination when probwidth() specified)
r(width)	CI width
r(width_a)	actual CI width (for sample-size determination when knownsd specified)
r(sd)	standard deviation
r(knownsd)	1 if option knownsd is specified, 0 otherwise
r(fpc)	finite population correction (if specified)
r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise
r(init)	initial value for sample size
r(maxiter)	maximum number of iterations
r(iter)	number of iterations performed
r(tolerance)	requested parameter tolerance
r(deltax)	final parameter tolerance achieved
r(ftolerance)	requested distance of the objective function from zero
r(function)	final distance of the objective function from zero
r(converged)	1 if iteration algorithm converged, 0 otherwise

Macros

r(type)	ci
r(method)	onemean
r(onesidedci)	upper or lower (for a one-sided CI)
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results

Methods and formulas

Let $\mathbf{x} = (x_1, \ldots, x_n)$ be a random sample of i.i.d. observations from a normal population with mean μ and variance σ^2. A general two-sided CI is defined as $[\ll(\mathbf{x}), ul(\mathbf{x})]$, a lower one-sided CI as $[\ll(\mathbf{x}), \infty)$, and an upper one-sided CI as $(-\infty, ul(\mathbf{x})]$, where $\ll(\mathbf{x}) = \ll$ and $ul(\mathbf{x}) = ul$ are the respective lower and upper confidence limits. Let w be the CI width.

Let

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \text{and} \qquad s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2$$

be the sample mean and the sample variance, respectively.

A two-sided CI for the population mean μ is constructed as

$$[\overline{x} - w/2, \overline{x} + w/2]$$

where $w/2$ is the half-width or margin of error.

Lower and upper one-sided CIs are constructed as

$$[\overline{x} - w, \infty)$$

$$(-\infty, \overline{x} + w]$$

We use the CI width w as our measure of CI precision. Let $100(1-\alpha)\%$ denote the confidence level, where $0 \leq \alpha \leq 1$ is the corresponding significance level.

The formulas below are based on Kupper and Hafner (1989), Ryan (2013), Dixon and Massey (1983), Zar (2010), and Chow et al. (2018).

Methods and formulas are presented under the following headings:

Known standard deviation
Unknown standard deviation
Finite population size

Known standard deviation

In the case of a known standard deviation, the sampling distribution of the statistic $z = \sqrt{n}(\overline{x} - \mu)/\sigma$ follows the standard normal distribution. Let $z_{1-\alpha}$ be the $(1-\alpha)$th quantile of the standard normal distribution.

Based on the normal distribution of z, the corresponding two-sided, lower, and upper CIs are

$$\left[\overline{x} - z_{1-\alpha/2}\frac{\sigma}{\sqrt{n}}, \overline{x} + z_{1-\alpha/2}\frac{\sigma}{\sqrt{n}}\right]$$

$$\left[\overline{x} - z_{1-\alpha}\frac{\sigma}{\sqrt{n}}, \infty\right)$$

$$\left(-\infty, \overline{x} + z_{1-\alpha}\frac{\sigma}{\sqrt{n}}\right]$$

The corresponding CI width is

$$w = \begin{cases} \frac{2z_{1-\alpha/2}\sigma}{\sqrt{n}} & \text{for a two-sided CI} \\ \frac{z_{1-\alpha}\sigma}{\sqrt{n}} & \text{for lower and upper one-sided CIs} \end{cases} \tag{1}$$

The sample size n is computed by inverting (1):

$$n = \begin{cases} \left(\frac{2z_{1-\alpha/2}\sigma}{w}\right)^2 & \text{for a two-sided CI} \\ \left(\frac{z_{1-\alpha}\sigma}{w}\right)^2 & \text{for lower and upper one-sided CIs} \end{cases} \tag{2}$$

Unknown standard deviation

In the case of an unknown standard deviation, an unbiased estimator s is used in place of σ in the definition of a z statistic from *Known standard deviation*. The sampling distribution of the corresponding statistic $t = \sqrt{n}(\overline{x}-\mu)/s$ follows a Student's t distribution with $n-1$ degrees of freedom. Let $t_{n-1,1-\alpha}$ be the $(1-\alpha)$th quantile of the Student's t distribution with $n-1$ degrees of freedom.

The corresponding two-sided, lower, and upper CIs are

$$\left[\overline{x} - t_{n-1,1-\alpha/2}\frac{s}{\sqrt{n}}, \overline{x} + t_{n-1,1-\alpha/2}\frac{s}{\sqrt{n}}\right]$$

$$\left[\overline{x} - t_{n-1,1-\alpha}\frac{s}{\sqrt{n}}, \infty\right)$$

$$\left(-\infty, \overline{x} + t_{n-1,1-\alpha}\frac{s}{\sqrt{n}}\right]$$

The CI width depends on the sample standard deviation s, which will vary between random samples. To account for the sampling variability of s and consequently of the CI width, Kupper and Hafner (1989) consider the probability of CI width, $\Pr(w)$, such that

$$\Pr\left(2t_{n-1,1-\alpha/2}\frac{s}{\sqrt{n}} \leq w\right) \geq \Pr(w) \tag{3}$$

The above is for a two-sided CI, but a similar expression can be constructed for a one-sided CI. $\Pr(w)$ is the probability that the width of a future CI will not exceed some prespecified width w.

Formula (3) can be rewritten as

$$\Pr\left\{\frac{(n-1)s^2}{\sigma^2} \geq \frac{n(n-1)w^2}{4\sigma^2 t^2_{n-1,1-\alpha/2}}\right\} \geq \Pr(w) \tag{4}$$

When data **x** are normally distributed, the sampling distribution of the statistic $(n-1)s^2/\sigma^2$ is a χ^2 distribution with $n-1$ degrees of freedom. Then, using (4), we can compute the probability of CI width as follows:

$$\Pr(w) = \begin{cases} \chi^2_{n-1}\left\{\frac{n(n-1)w^2}{4\sigma^2 t^2_{n-1,1-\alpha/2}}\right\} & \text{for a two-sided CI} \\ \chi^2_{n-1}\left\{\frac{n(n-1)w^2}{\sigma^2 t^2_{n-1,1-\alpha}}\right\} & \text{for lower and upper one-sided CI}s \end{cases} \tag{5}$$

where $\chi^2_{n-1}(\cdot)$ is the c.d.f. of a χ^2 distribution with $n-1$ degrees of freedom.

We can compute the CI width from (5) as follows:

$$w = \begin{cases} 2t_{n-1,1-\alpha/2}\sigma\sqrt{\frac{\chi^2_{n-1,\Pr(w)}}{n(n-1)}} & \text{for a two-sided CI} \\ t_{n-1,1-\alpha}\sigma\sqrt{\frac{\chi^2_{n-1,\Pr(w)}}{n(n-1)}} & \text{for lower and upper one-sided CI}s \end{cases} \tag{6}$$

where $\chi^2_{n-1,p}$ is the pth quantile of a χ^2 distribution with $n-1$ degrees of freedom.

We solve for the sample size iteratively using (6) with initial values obtained from (2).

Finite population size

The above formulas assume that the random sample is drawn from an infinite population. In cases when the size of the population is known, we need to make the following adjustment to the standard deviation,

$$\sigma_{\text{fpc}} = \sigma\sqrt{\left(1 - \frac{n}{N}\right)}$$

where σ_{fpc} is the population standard deviation adjusted for finite population size.

If the `nfractional` option is not specified, the computed sample size is rounded up.

References

Chow, S.-C., J. Shao, H. Wang, and Y. Lokhnygina. 2018. *Sample Size Calculations in Clinical Research*. 3rd ed. Boca Raton, FL: CRC Press.

Dixon, W. J., and F. J. Massey, Jr. 1983. *Introduction to Statistical Analysis*. 4th ed. New York: McGraw–Hill.

Kupper, L. L., and K. B. Hafner. 1989. How appropriate are popular sample size formulas? *American Statistician* 43: 101–105. https://doi.org/10.2307/2684511.

Meeker, W. Q., G. J. Hahn, and L. A. Escobar. 2017. *Statistical Intervals: A Guide for Practitioners and Researchers*. 2nd ed. Hoboken, NJ: Wiley.

Ryan, T. P. 2013. *Sample Size Determination and Power*. Hoboken, NJ: Wiley. https://doi.org/10.1002/9781118439241.

Zar, J. H. 2010. *Biostatistical Analysis*. 5th ed. Upper Saddle River, NJ: Pearson.

Also see

[PSS-3] **ciwidth** — Precision and sample-size analysis for CIs

[PSS-3] **ciwidth, graph** — Graph results from the ciwidth command

[PSS-3] **ciwidth, table** — Produce table of results from the ciwidth command

[PSS-2] **power onemean** — Power analysis for a one-sample mean test

[PSS-5] **Glossary**

[R] **ci** — Confidence intervals for means, proportions, and variances

ciwidth twomeans — Precision analysis for a two-means-difference CI

Description

`ciwidth twomeans` computes sample size, CI width, and probability of CI width for a CI for a difference between two means from independent samples. It can compute sample size for a given CI width and probability of CI width. Alternatively, it can compute CI width for a given sample size and probability of CI width. It can also compute probability of CI width for a given sample size and CI width. Also see [PSS-3] **ciwidth** for PrSS analysis for other CI methods.

For power and sample-size analysis for a two-sample mean test, see [PSS-2] **power twomeans**.

Quick start

Sample size required for a two-sided 95% CI for the difference between two means to have a width no larger than 12 with a probability of 90%, assuming a common standard deviation of 9

```
ciwidth twomeans, width(12) probwidth(0.9) sd(9)
```

Same as above, but for an upper one-sided CI

```
ciwidth twomeans, width(12) probwidth(0.9) sd(9) upper
```

Sample size required for a two-sided 95% CI for the difference between two means to have a width no larger than 12, with known control- and experimental-group standard deviations of 7 and 10, respectively

```
ciwidth twomeans, width(12) sd1(7) sd2(10) knownsds
```

CI width for a total sample size of 74 with balanced group sizes, given a 90% probability that the CI width will be no larger than the estimated value

```
ciwidth twomeans, n(74) probwidth(0.9) sd(9)
```

Same as above, but for sample sizes of 45 and 30 in groups 1 and 2, respectively

```
ciwidth twomeans, n1(45) n2(30) probwidth(0.9) sd(9)
```

Probability that the CI width is no larger than 12 for a sample size of 50

```
ciwidth twomeans, width(12) n(50) sd(9)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
ciwidth twomeans, width(numlist) probwidth(numlist) [options]
```

Compute CI width

```
ciwidth twomeans, probwidth(numlist) n(numlist) [options]
```

Compute probability of CI width

```
ciwidth twomeans, width(numlist) n(numlist) [options]
```

options	Description
Main	
* level(*numlist*)	confidence level; default is level(95)
* alpha(*numlist*)	significance level; default is alpha(0.05)
* probwidth(*numlist*)	probability of CI width; required to compute sample size and CI width
* width(*numlist*)	CI width; required to compute sample size and probability of CI width
* n(*numlist*)	total sample size; required to compute CI width and probability of CI width
* n1(*numlist*)	sample size of the control group
* n2(*numlist*)	sample size of the experimental group
* nratio(*numlist*)	ratio of sample sizes, N2/N1; default is nratio(1), meaning equal group sizes
compute(N1 | N2)	solve for N1 given N2 or for N2 given N1
nfractional	allow fractional sample sizes
* sd(*numlist*)	common standard deviation of the control and the experimental groups assuming equal standard deviations in both groups; default is sd(1)
* sd1(*numlist*)	standard deviation of the control group; requires sd2() and knownsds
* sd2(*numlist*)	standard deviation of the experimental group; requires sd1() and knownsds
knownsds	request computation assuming known standard deviations for both groups; default is to assume unknown standard deviations
lower	lower one-sided CI; default is two-sided CI
upper	upper one-sided CI; default is two-sided CI
onesided	synonym for option upper
parallel	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)

Table

[no]table[(*tablespec*)]	suppress table or display results as a table; see [PSS-3] **ciwidth, table**
saving(*filename* [, replace])	save the table data to *filename*; use replace to overwrite existing *filename*

Graph

graph[(*graphopts*)]	graph results; see [PSS-3] **ciwidth, graph**

Iteration

init(#)	initial value for sample size; default is to use a closed-form normal approximation
iterate(#)	maximum number of iterations; default is iterate(500)
tolerance(#)	parameter tolerance; default is tolerance(1e-12)
ftolerance(#)	function tolerance; default is ftolerance(1e-12)
[no]log	suppress or display iteration log
[no]dots	suppress or display iterations as dots
notitle	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the parallel option.

collect is allowed; see **[U] 11.1.10 Prefix commands**.

notitle does not appear in the dialog box.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
level	confidence level	$100(1-\alpha)$
alpha	significance level	α
N	total number of subjects	N
N1	number of subjects in the control group	N_1
N2	number of subjects in the experimental group	N_2
nratio	ratio of sample sizes, experimental to control	N_2/N_1
Pr_width	probability of CI width	p_{width}
width	CI width	w
sd	common standard deviation	σ
sd1	control-group standard deviation	σ_1
sd2	experimental-group standard deviation	σ_2
_all	display all supported columns	

Column alpha is shown in the default table in place of column level if alpha() is specified.

Columns nratio, sd, sd1, and sd2 are shown in the default table if the corresponding options are specified.

Options

Main

`level()`, `alpha()`, `probwidth()`, `width()`, `n()`, `n1()`, `n2()`, `nratio()`, `compute()`, `nfractional`; see [PSS-3] **ciwidth**. `probwidth()` may not be combined with `sd1()`, `sd2()`, and `knownsds`.

`sd(`*numlist*`)` specifies the common standard deviation of the control and the experimental groups assuming equal standard deviations in both groups. The default is `sd(1)`.

`sd1(`*numlist*`)` specifies the standard deviation of the control group. If you specify `sd1()`, you must also specify `sd2()` and `knownsds`. `sd1()` may not be combined with `probwidth()`.

`sd2(`*numlist*`)` specifies the standard deviation of the experimental group. If you specify `sd2()`, you must also specify `sd1()` and `knownsds`. `sd2()` may not be combined with `probwidth()`.

`knownsds` requests that standard deviations of each group be treated as known in the computation. By default, standard deviations are treated as unknown, and the computation is performed for a Student's t-based CI. If `knownsds` is specified, the computation is performed for a normal-based CI. `knownsds` may not be combined with `probwidth()` and is not allowed when computing the probability of CI width.

`lower`, `upper`, `onesided`, `parallel`; see [PSS-3] **ciwidth**.

Table

`table`, `table()`, `notable`; see [PSS-3] **ciwidth, table**.

`saving()`; see [PSS-3] **ciwidth**.

Graph

`graph`, `graph()`; see [PSS-3] **ciwidth, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies the initial value for the estimated sample size for sample-size determination. The estimated sample size is either the control-group size n_1 or, if `compute(N2)` is specified, the experimental-group size n_2. The default is to use a closed-form normal approximation to compute an initial sample size.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-3] **ciwidth**.

The following option is available with `ciwidth twomeans` but is not shown in the dialog box:

`notitle`; see [PSS-3] **ciwidth**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `ciwidth twomeans` command and the methodology for PrSS analysis for a CI for a difference between two means from independent samples. See [PSS-3] **Intro (ciwidth)** for a general introduction to PrSS analysis, and see [PSS-3] **ciwidth** for a general introduction to the `ciwidth` command. For PSS analysis for hypothesis tests, see [PSS-2] **power**.

Introduction

The analysis of means is one of the most commonly used approaches in many statistical studies. Many applications lead to the study of two independent means, such as studies comparing the average mileage of foreign and domestic cars, the average SAT scores obtained from two different coaching classes, the average yields of a crop using two different fertilizers, and so on. The two populations of interest are assumed to be independent. We are interested in a CI for the difference $\mu_D = \mu_2 - \mu_1$ between the two means μ_2 and μ_1. The wider the ranges of the CI are, the less precise it is.

The precision of a CI is commonly measured by its width w or, for a symmetric CI such as the CI for the mean difference, by its half-width d, also known as the margin of error. For example, a two-sided two-means-difference CI is formed as $[\widehat{\mu}_D - d, \widehat{\mu}_D + d]$, where $\widehat{\mu}_D$ is the point estimate of the mean difference. The CI width, the distance between the upper and lower limits, is $w = 2d$; it does not depend on the means difference estimate. The smaller the d or w the more precise the CI.

In PrSS analysis, it is usually of interest to determine the sample size that would be sufficient for a CI to have a prespecified width in a future study. Generally, larger sample sizes lead to more precise CIs. To compute the required sample size, we need to know the expression for w. The expression for w depends on various assumptions.

Similarly to the width of a one-mean CI, the CI width w of a two-means-difference CI depends on sample estimates of standard deviations, in the case of unknown standard deviations. Therefore, w will vary from one sample to another. To ensure that a CI has the desired width in a future study, this sampling variability of w must be accounted for when computing the required sample size. Kupper and Hafner (1989) introduce what we call the probability of CI width, which specifies the probability of a future CI to have the width of no larger than some prespecified CI width for a given sample size. See *Methods and formulas* for details.

You can use `ciwidth twomeans` to perform PrSS analysis for a CI for the difference between two independent means. We discuss the command details in the next section.

Using ciwidth twomeans

`ciwidth twomeans` computes sample size, CI width, or probability of CI width for a two-means-difference CI. By default, a two-sided CI is assumed, and the confidence level is set to 95%. You may change the confidence level by specifying the `level()` option. Alternatively, you can specify the significance level in the `alpha()` option. You can specify the `upper` and `lower` options to request upper and lower one-sided CIs. By default, all computations assume a balanced- or equal-allocation design; see [PSS-4] **Unbalanced designs** for a description of how to specify an unbalanced design.

To compute total sample size, you must specify the CI width in the `width()` option and the probability of CI width in the `probwidth()` option. To compute CI width, you must specify the sample size in the `n()` option and the probability of CI width in the `probwidth()` option. You can also compute the probability of CI width given the sample size in `n()` and CI width in `width()`.

Instead of the total sample size, you can compute one of the group sizes given the other one. To compute the control-group sample size, you must specify the `compute(N1)` option and the sample size of the experimental group in the `n2()` option. Likewise, to compute the experimental-group sample size, you must specify the `compute(N2)` option and the sample size of the control group in the `n1()` option.

For CIs for means, the CI width does not depend on the mean point estimates, the sample means, so they are not needed in the computations.

By default, all computations are performed for a two-sample CI that assumes equal and unknown standard deviations. By default, the common standard deviation is set to one but may be changed by specifying the `sd()` option. To specify a known common standard deviation, use the `knownsds` option. To specify different standard deviations, use the respective `sd1()` and `sd2()` options. These options must be specified together and in combination with `knownsds`; they may not be used in combination with `sd()`. When `sd1()` and `sd2()` are specified, the computations are based on a normal z-based CI. The `sd1()`, `sd2()`, and `knownsds` options may not be combined with `probwidth()`.

Instead of the total sample size `n()`, you can specify individual group sizes in `n1()` and `n2()`, or specify one of the group sizes and `nratio()` when computing CI width or effect size. Also see *Two samples* in [PSS-4] **Unbalanced designs** for more details.

By default, the computed sample size is rounded up. You can specify the `nfractional` option to see the corresponding fractional sample size; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. The `nfractional` option is allowed only for sample-size determination.

Some of `ciwidth twomeans`'s computations require iteration. For example, when standard deviations are equal but unknown, the sample-size computation requires iteration. The default initial value of the estimated sample size is obtained by using a closed-form normal approximation. It may be changed by specifying the `init()` option. See [PSS-3] **ciwidth** for the descriptions of other options that control the iteration procedure.

In the following sections, we describe the use of `ciwidth twomeans` accompanied by examples for computing sample size, CI width, and probability of CI width.

Computing sample size

To compute the sample size required for a two-means-difference CI to have the width no larger than a target width, you must specify the target CI width in the `width()` option and the desired probability of achieving the target CI width in the `probwidth()` option.

▷ Example 1: Sample size for a two-means-difference CI

Similarly to the study in [PSS-2] **power twomeans**, we want to investigate the effects of smoking on lung function of males. In that entry, we tested the means of the forced expiratory volume (FEV), measured in liters (L), across smokers and nonsmokers, where better lung function implied higher values of FEV. Here, we wish to estimate the CI for the difference between the mean FEV of the two groups.

We are designing a new study for this objective, and we wish to find out how many subjects we need to enroll so that the width of a two-sided 95% CI for the mean FEV difference is no larger than 0.5 L with a probability of 0.96.

Assuming equal numbers of subjects in each group and a common standard deviation of 1, we compute the required sample size:

```
. ciwidth twomeans, probwidth(0.96) width(0.5)
Performing iteration ...
Estimated sample sizes for a two-means-difference CI
Student's t two-sided CI assuming sd1 = sd2 = sd
Study parameters:
        level =     95.00
     Pr_width =    0.9600
        width =    0.5000
           sd =    1.0000
Estimated sample sizes:
            N =       286
  N per group =       143
```

We need a total sample of 286 subjects, 143 per group.

The default computation is for the case of equal and unknown standard deviations, as indicated by the output. You can specify the `knownsds` option to request the computation assuming known standard deviations, but note that `knownsds` cannot be used in conjunction with `probwidth()`.

◁

▷ Example 2: Computing one of the group sizes

Suppose we anticipate a sample of 120 nonsmoking subjects. We wish to compute the required number of subjects in the smoking group, leaving all other study parameters unchanged from example 1. We specify the number of subjects in the nonsmoking group in the `n1()` option and specify the `compute(N2)` option.

```
. ciwidth twomeans, probwidth(0.96) width(0.5) n1(120) compute(N2)
Performing iteration ...
Estimated sample sizes for a two-means-difference CI
Student's t two-sided CI assuming sd1 = sd2 = sd
Study parameters:
        level =     95.00
     Pr_width =    0.9600
        width =    0.5000
           sd =    1.0000
           N1 =       120
Estimated sample sizes:
            N =       296
           N2 =       176
```

We need a sample of 176 smoking subjects given a sample of 120 nonsmoking subjects.

◁

▷ Example 3: Unbalanced design

By default, `ciwidth twomeans` computes sample size for a balanced- or equal-allocation design. If we know the allocation ratio of subjects between the groups, we can compute the required sample size for an unbalanced design by specifying the `nratio()` option.

Continuing with example 1, suppose that we anticipate on recruiting twice as many smokers as nonsmokers; that is, $n_2/n_1 = 2$. We specify the `nratio(2)` option to compute the required sample size for the specified unbalanced design.

```
. ciwidth twomeans, probwidth(0.96) width(0.5) nratio(2)
Performing iteration ...
Estimated sample sizes for a two-means-difference CI
Student's t two-sided CI assuming sd1 = sd2 = sd
Study parameters:
        level =     95.00
     Pr_width =    0.9600
        width =    0.5000
           sd =    1.0000
        N2/N1 =    2.0000
Estimated sample sizes:
            N =       321
           N1 =       107
           N2 =       214
```

We need a total sample size of 321 subjects, which is larger than the required total sample size for the corresponding balanced design from example 1.

Also see *Two-samples* in [PSS-4] **Unbalanced designs** for more examples of unbalanced designs.

◁

Computing CI width

To compute the CI width, you must specify the sample size in the `n()` option and the desired probability of achieving the target CI width in the `probwidth()` option.

▷ Example 4: Computing CI width for a two-means-difference CI

Suppose that we have enough resources to enroll 250 subjects in our study on FEV across smokers and nonsmokers. Further suppose that we would like to be 96% certain that the width of a future CI for this sample size will be no larger than the value we estimate. Given these parameters, we compute the CI width as follows:

```
. ciwidth twomeans, probwidth(0.96) n(250)
Estimated width for a two-means-difference CI
Student's t two-sided CI assuming sd1 = sd2 = sd
Study parameters:
        level =     95.00
            N =       250
  N per group =       125
     Pr_width =    0.9600
           sd =    1.0000
Estimated width:
        width =    0.5373
```

The estimated width for a future two-sided 95% CI for the mean difference between the smoking and nonsmoking groups is 0.54, given other parameters.

◁

Computing probability of CI width

To compute the probability that the width of a future CI will be no larger than the specified width, you must specify the sample size in the `n()` option and the target CI width in the `width()` option.

▷ Example 5: Computing probability of CI width for a two-means-difference CI

Continuing with example 1, suppose that we have only enough resources to enroll a total of 250 subjects, instead of the 286 we computed before. Assuming equal-sized groups, and given this smaller sample size, we would like to know the probability of obtaining the target CI width of 0.5 for a two-sided 95% CI:

```
. ciwidth twomeans, width(0.5) n(250)
Estimated probability of width for a two-means-difference CI
Student's t two-sided CI assuming sd1 = sd2 = sd
Study parameters:
        level =     95.00
            N =       250
  N per group =       125
        width =    0.5000
           sd =    1.0000
Estimated probability of width:
     Pr_width =    0.5427
```

The estimated probability, given the total sample size of 250, is 54% and is rather low.

◁

▷ Example 6: Multiple values of study parameters

As a variation of example 5, we would like to see how increasing the sample size, from the 250 we specified above, affects the probability of achieving a target CI width of 0.5. We compute the probability of CI width for a range of sample sizes between 250 and 300, with the step size of 10, by specifying the corresponding numlist in the `n()` option.

```
. ciwidth twomeans, width(0.5) n(250(10)300)
Estimated probability of width for a two-means-difference CI
Student's t two-sided CI assuming sd1 = sd2 = sd
```

level	N	N1	N2	Pr_width	width	sd
95	250	125	125	.5427	.5	1
95	260	130	130	.7129	.5	1
95	270	135	135	.8467	.5	1
95	280	140	140	.9316	.5	1
95	290	145	145	.9749	.5	1
95	300	150	150	.9925	.5	1

Assuming a balanced design, the probability of CI width increases from 54% to 99% as the sample size increases from 250 to 300, given a target CI width of 0.5.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-3] **ciwidth, table**. If you wish to produce sample-size and other curves, see [PSS-3] **ciwidth, graph**.

◁

▷ Example 7: One-sided CI

By default, `ciwidth twomeans` performs computations based on a two-sided CI. You can specify the `upper` or `lower` option to request either an upper or lower one-sided CI.

Suppose we want to know the probability of achieving a smaller target CI width of 0.25 for an upper one-sided 95% CI for the difference between FEV means, given a smaller sample size of 200.

```
. ciwidth twomeans, width(0.25) n(200) upper

Estimated probability of width for a two-means-difference CI
Student's t upper CI assuming sd1 = sd2 = sd

Study parameters:

        level =     95.00
            N =       200
  N per group =       100
        width =    0.2500
           sd =    1.0000

Estimated probability of width:

     Pr_width =    0.9199
```

Given a total sample size of 200, we are 92% likely to obtain an upper one-sided 95% CI with the width no larger than 0.25 in a future study.

◁

Stored results

`ciwidth twomeans` stores the following in `r()`:

Scalars

`r(level)`	confidence level
`r(alpha)`	significance level
`r(N)`	total sample size
`r(N_a)`	actual sample size
`r(N1)`	sample size of the control group
`r(N2)`	sample size of the experimental group
`r(nratio)`	ratio of sample sizes, `N2/N1`
`r(nratio_a)`	actual ratio of sample sizes
`r(nfractional)`	1 if `nfractional` is specified, 0 otherwise
`r(onesided)`	1 for a one-sided CI, 0 otherwise
`r(Pr_width)`	probability of CI width
`r(Pr_width_a)`	actual probability of CI width (for sample-size determination when `probwidth()` specified)
`r(width)`	CI width
`r(width_a)`	actual CI width (for sample-size determination when `knownsds` specified)
`r(sd)`	common standard deviation of the control and experimental groups
`r(sd1)`	standard deviation of the control group
`r(sd2)`	standard deviation of the experimental group
`r(knownsds)`	1 if option `knownsds` is specified, 0 otherwise
`r(separator)`	number of lines between separator lines in the table
`r(divider)`	1 if `divider` is requested in the table, 0 otherwise
`r(init)`	initial value for sample size
`r(maxiter)`	maximum number of iterations
`r(iter)`	number of iterations performed
`r(tolerance)`	requested parameter tolerance
`r(deltax)`	final parameter tolerance achieved
`r(ftolerance)`	requested distance of the objective function from zero
`r(function)`	final distance of the objective function from zero
`r(converged)`	1 if iteration algorithm converged, 0 otherwise

Macros

r(type)	ci
r(method)	twomeans
r(onesidedci)	upper or lower (for a one-sided CI)
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results

Methods and formulas

Consider two independent samples with n_1 subjects in the control group and n_2 subjects in the experimental group. Let $\mathbf{x}_1 = (x_{11}, \ldots, x_{1n_1})$ be a random sample of size n_1 from a normal population with mean μ_1 and variance σ_1^2. Let $\mathbf{x}_2 = (x_{21}, \ldots, x_{2n_2})$ be a random sample of size n_2 from a normal population with mean μ_2 and variance σ_2^2. We are interested in a CI for the mean difference $\mu_2 - \mu_1$ estimated using samples $\mathbf{x}_1$ and $\mathbf{x}_2$. Let $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2)$. A general two-sided CI is defined as $[\ll(\mathbf{x}), ul(\mathbf{x})]$, a lower one-sided CI as $[\ll(\mathbf{x}), \infty)$, and an upper one-sided CI as $(-\infty, ul(\mathbf{x})]$, where $\ll(\mathbf{x}) = \ll$ and $ul(\mathbf{x}) = ul$ are the respective lower and upper confidence limits. Let w be the CI width.

The sample means and variances for the two independent samples are

$$\overline{x}_1 = \frac{1}{n_1}\sum_{i=1}^{n_1} x_{1i} \qquad \text{and} \qquad s_1^2 = \frac{1}{n_1 - 1}\sum_{i=1}^{n_1}(x_{1i} - \overline{x}_1)^2$$

$$\overline{x}_2 = \frac{1}{n_2}\sum_{i=1}^{n_2} x_{2i} \qquad \text{and} \qquad s_2^2 = \frac{1}{n_2 - 1}\sum_{i=1}^{n_2}(x_{2i} - \overline{x}_2)^2$$

Let $\overline{x}_D$ be the sample mean difference $\overline{x}_2 - \overline{x}_1$.

A two-sided CI for the means difference is constructed as

$$[\overline{x}_D - w/2, \overline{x}_D + w/2]$$

where $w/2$ is the half-width or margin of error.

Lower and upper one-sided CIs are

$$[\overline{x}_D - w, \infty)$$
$$(-\infty, \overline{x}_D + w]$$

We use the CI width w as our measure of CI precision. Let $100(1-\alpha)\%$ denote the confidence level, where $0 \le \alpha \le 1$ is the corresponding significance level.

Let $R = n_2/n_1$ denote the allocation ratio. Then $n_2 = R \times n_1$ and CI width can be viewed as a function of n_1. Therefore, for sample-size determination, the control-group sample size n_1 is computed first. The experimental-group size n_2 is then computed as $R \times n_1$, and the total sample size is computed as $n = n_1 + n_2$. By default, sample sizes are rounded to integer values; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for details.

PrSS analysis using `ciwidth twomeans` can be performed under three different assumptions: 1) population standard deviations are known and equal; 2) population standard deviations are known and unequal; and 3) population standard deviations are unknown but equal. We describe each case below.

The following formulas are based on Kupper and Hafner (1989), Ryan (2013), Dixon and Massey (1983), Zar (2010), and Chow et al. (2018).

Methods and formulas are presented under the following headings:

Known equal and unequal standard deviations

Below we present formulas for the computations that assume unequal standard deviations. When standard deviations are equal, the corresponding formulas are special cases of the formulas below with $\sigma_1 = \sigma_2 = \sigma$.

Let σ_D denote the standard deviation of the difference between the two sample means. With known standard deviation, $\sigma_D = \sqrt{\sigma_1^2/n_1 + \sigma_2^2/n_2}$. The statistic

$$\text{TS} = \frac{\overline{x}_D - (\mu_2 - \mu_1)}{\sigma_D}$$

follows a normal distribution.

Let $z_{1-\alpha}$ be the $(1-\alpha)$th quantile of a standard normal distribution. Based on the normal distribution of z, the CIs are

$$\begin{cases} \left[\overline{x}_D - z_{1-\alpha/2}\sigma_D, \overline{x}_D + z_{1-\alpha/2}\sigma_D\right] & \text{for a two-sided CI} \\ [\overline{x}_D - z_{1-\alpha}\sigma_D, \infty) & \text{for a lower CI} \\ (-\infty, \overline{x}_D + z_{1-\alpha}\sigma_D] & \text{for an upper CI} \end{cases}$$

After expanding σ_D, the corresponding width w is

$$w = \begin{cases} 2z_{1-\alpha/2}\sqrt{\sigma_1^2/n_1 + \sigma_2^2/n_2} & \text{for a two-sided CI} \\ z_{1-\alpha}\sqrt{\sigma_1^2/n_1 + \sigma_2^2/n_2} & \text{for lower and upper one-sided CIs} \end{cases}$$

The control-group sample size n_1 is computed as follows:

$$n_1 = \begin{cases} 4\left(\frac{z_{1-\alpha/2}}{w}\right)^2\left(\sigma_1^2 + \frac{\sigma_2^2}{R}\right) & \text{for a two-sided CI} \\ \left(\frac{z_{1-\alpha}}{w}\right)^2\left(\sigma_1^2 + \frac{\sigma_2^2}{R}\right) & \text{for lower and upper one-sided CIs} \end{cases} \tag{1}$$

If one of the group sizes is known, the other one is computed using the following formula. For example, for a two-sided CI, to compute n_1 given n_2, we use the following formula:

$$n_1 = \frac{\sigma_1^2}{\left(\frac{w}{2z_{1-\alpha/2}}\right)^2 - \frac{\sigma_2^2}{n_2}} \tag{2}$$

Unknown and equal standard deviations

When the standard deviations of the control group and the experimental group are unknown and equal, the statistic

$$t = \frac{\overline{x}_D - (\mu_2 - \mu_1)}{s_D}$$

follows a Student's t distribution with ν degrees of freedom. The estimated standard deviation of the sample mean difference s_D is

$$s_D = s_p\sqrt{1/n_1 + 1/n_2}$$

where $s_p = \left\{\sum_{i=1}^{n_1}(x_{1i} - \overline{x}_1)^2 + \sum_{i=1}^{n_2}(x_{2i} - \overline{x}_2)^2\right\}/(n_1 + n_2 - 2)$ is the pooled-sample standard deviation.

The degrees of freedom ν is

$$\nu = n_1 + n_2 - 2$$

Let $t_{\nu,\alpha}$ denote the αth quantile of a Student's t distribution with ν degrees of freedom. The CIs are

$$\begin{cases} \left[\overline{x}_D - t_{\nu,1-\alpha/2}s_D, \overline{x}_D + t_{\nu,1-\alpha/2}s_D\right] & \text{for a two-sided CI} \\ \left[\overline{x}_D - t_{\nu,1-\alpha}s_D, \infty\right) & \text{for a lower CI} \\ \left(-\infty, \overline{x}_D + t_{\nu,1-\alpha}s_D\right] & \text{for an upper CI} \end{cases}$$

Similarly to the case of an unknown standard deviation for a one-mean CI, the CI width depends on the sample standard deviations. Using the fact that statistic $\nu s_D^2/\sigma_D^2$ follows a χ^2 distribution with ν degrees of freedom, we can compute the probability that the CI width is no larger than a specified value w.

The probability of CI width is

$$\Pr(w) = \begin{cases} \chi^2_\nu\left\{\frac{\nu w^2}{4t^2_{\nu,1-\alpha/2}\sigma^2\left(\frac{1}{n_1}+\frac{1}{n_2}\right)}\right\} & \text{for a two-sided CI} \\ \chi^2_\nu\left\{\frac{\nu w^2}{t^2_{\nu,1-\alpha}\sigma^2\left(\frac{1}{n_1}+\frac{1}{n_2}\right)}\right\} & \text{for lower and upper one-sided CIs} \end{cases} \tag{4}$$

where $\chi^2_\nu(\cdot)$ is the c.d.f. of a χ^2 distribution with ν degrees of freedom.

We can compute the desired CI width from (4):

$$w = \begin{cases} 2t_{\nu,1-\alpha/2}\sigma\sqrt{\frac{\chi^2_{\nu,\Pr(w)}}{\nu}\left(\frac{1}{n_1}+\frac{1}{n_2}\right)} & \text{for a two-sided CI} \\ t_{\nu,1-\alpha}\sigma\sqrt{\frac{\chi^2_{\nu,\Pr(w)}}{\nu}\left(\frac{1}{n_1}+\frac{1}{n_2}\right)} & \text{for lower and upper one-sided CIs} \end{cases} \tag{5}$$

where $\chi^2_{\nu,p}$ is the pth quantile of a χ^2 distribution with ν degrees of freedom.

We can solve for the sample sizes iteratively from (5) using initial values obtained from (1) and (2).

References

Chow, S.-C., J. Shao, H. Wang, and Y. Lokhnygina. 2018. *Sample Size Calculations in Clinical Research*. 3rd ed. Boca Raton, FL: CRC Press.

Dixon, W. J., and F. J. Massey, Jr. 1983. *Introduction to Statistical Analysis*. 4th ed. New York: McGraw–Hill.

Juul, S., and M. Frydenberg. 2021. *An Introduction to Stata for Health Researchers*. 5th ed. College Station, TX: Stata Press.

Kupper, L. L., and K. B. Hafner. 1989. How appropriate are popular sample size formulas? *American Statistician* 43: 101–105. https://doi.org/10.2307/2684511.

Ryan, T. P. 2013. *Sample Size Determination and Power*. Hoboken, NJ: Wiley. https://doi.org/10.1002/9781118439241.

Zar, J. H. 2010. *Biostatistical Analysis*. 5th ed. Upper Saddle River, NJ: Pearson.

Also see

[PSS-3] **ciwidth** — Precision and sample-size analysis for CIs

[PSS-3] **ciwidth, graph** — Graph results from the ciwidth command

[PSS-3] **ciwidth, table** — Produce table of results from the ciwidth command

[PSS-2] **power twomeans** — Power analysis for a two-sample means test

[PSS-5] **Glossary**

[R] **ttest** — t tests (mean-comparison tests)

[R] **ztest** — z tests (mean-comparison tests, known variance)

ciwidth pairedmeans — Precision analysis for a paired-means-difference CI

Description

`ciwidth pairedmeans` computes sample size, CI width, and probability of CI width for a CI for the difference between two means from paired samples. It can compute sample size for a given CI width and probability of CI width. Alternatively, it can compute CI width for a given sample size and probability of CI width. It can also compute probability of CI width for a given sample size and CI width. Also see [PSS-3] **ciwidth** for PrSS analysis for other CI methods.

For power and sample-size analysis for a two-sample paired-means test, see [PSS-2] **power pairedmeans**.

Quick start

Sample size required for a two-sided 95% CI for the difference between paired means to have a width no larger than 12 with a probability of 90%, assuming the standard deviation for the differences of 36

```
ciwidth pairedmeans, width(12) probwidth(0.9) sddiff(36)
```

Same as above, but instead of standard deviation of the differences, specify correlation between paired observations of 0.5 with pretreatment standard deviation of 29 and posttreatment standard deviation of 40

```
ciwidth pairedmeans, width(12) probwidth(0.9) corr(.5) sd1(29) sd2(40)
```

CI width for sample sizes of 20, 40, 60, and 80, given a 90% probability that the CI width will be no larger than the estimated value

```
ciwidth pairedmeans, n(20(20)80) probwidth(0.9) sddiff(36)
```

Same as above, but display results as a graph of CI width versus sample size

```
ciwidth pairedmeans, n(20(20)80) probwidth(0.9) sddiff(36) graph
```

Probability that the CI width is no larger than 12 for a sample size of 50

```
ciwidth pairedmeans, width(12) n(50) sddiff(36)
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

```
ciwidth pairedmeans, corrspec width(numlist) probwidth(numlist)
  [options]
```

Compute CI width

```
ciwidth pairedmeans, corrspec probwidth(numlist) n(numlist) [options]
```

Compute probability of CI width

```
ciwidth pairedmeans, corrspec width(numlist) n(numlist) [options]
```

where *corrspec* is one of

```
sddiff()
corr() [sd()]
corr() [sd1() sd2()]
```

options	Description
Main	
* `level(`*numlist*`)`	confidence level; default is `level(95)`
* `alpha(`*numlist*`)`	significance level; default is `alpha(0.05)`
* `probwidth(`*numlist*`)`	probability of CI width; required to compute sample size and CI width
* `width(`*numlist*`)`	CI width; required to compute sample size and probability of CI width
* `n(`*numlist*`)`	sample size; required to compute CI width and probability of CI width
`nfractional`	allow fractional sample sizes
* `sddiff(`*numlist*`)`	standard deviation σ_d of the differences; may not be combined with `corr()`
* `corr(`*numlist*`)`	correlation between paired observations; required unless `sddiff()` is specified
* `sd(`*numlist*`)`	common standard deviation; default is `sd(1)` and requires `corr()`
* `sd1(`*numlist*`)`	standard deviation of the pretreatment group; requires `corr()`
* `sd2(`*numlist*`)`	standard deviation of the posttreatment group; requires `corr()`
`knownsd`	request computation assuming a known standard deviation σ_d; default is to assume an unknown standard deviation
* `fpc(`*numlist*`)`	finite population correction (FPC) as a sampling rate or as a population size
`lower`	lower one-sided CI; default is two-sided CI
`upper`	upper one-sided CI; default is two-sided CI
`onesided`	synonym for option `upper`
`parallel`	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[`no`]`table`[(*tablespec*)]	suppress table or display results as a table; see [PSS-3] **ciwidth, table**
`saving(`*filename* [`, replace`]`)`	save the table data to *filename*; use `replace` to overwrite existing *filename*
Graph	
`graph`[(*graphopts*)]	graph results; see [PSS-3] **ciwidth, graph**

Iteration	
`init(#)`	initial value for sample size; default is to use a closed-form normal approximation
`iterate(#)`	maximum number of iterations; default is `iterate(500)`
`tolerance(#)`	parameter tolerance; default is `tolerance(1e-12)`
`ftolerance(#)`	function tolerance; default is `ftolerance(1e-12)`
`[no]log`	suppress or display iteration log
`[no]dots`	suppress or display iterations as dots
`notitle`	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the `parallel` option.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`notitle` does not appear in the dialog box.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`level`	confidence level	$100(1-\alpha)$
`alpha`	significance level	α
`N`	number of subjects	N
`Pr_width`	probability of CI width	p_{width}
`width`	CI width	w
`sd_d`	standard deviation of the differences	σ_d
`sd`	common standard deviation	σ
`sd1`	standard deviation of the pretreatment group	σ_1
`sd2`	standard deviation of the posttreatment group	σ_2
`corr`	correlation between paired observations	ρ
`fpc`	FPC as a population size	N_{pop}
	FPC as a sampling rate	γ
`_all`	display all supported columns	

Column `alpha` is shown in the default table in place of column `level` if `alpha()` is specified.

Columns `sd`, `sd1`, `sd2`, `corr`, and `fpc` are shown in the default table if the corresponding options are specified.

Options

Main

`level()`, `alpha()`, `probwidth()`, `width()`, `n()`, `nfractional`; see [PSS-3] **ciwidth**. `probwidth()` may not be combined with `knownsd`. The `nfractional` option is allowed only for sample-size determination.

`sddiff(`*numlist*`)` specifies the standard deviation σ_d of the differences. Either `sddiff()` or `corr()` must be specified.

`corr(`*numlist*`)` specifies the correlation between paired, pretreatment and posttreatment, observations. This option along with `sd1()` and `sd2()` or `sd()` is used to compute the standard deviation of the differences unless that standard deviation is supplied directly in the `sddiff()` option. Either `corr()` or `sddiff()` must be specified.

`sd(`*numlist*`)` specifies the common standard deviation of the pretreatment and posttreatment groups. Specifying `sd(#)` implies that both `sd1()` and `sd2()` are equal to #. Options `corr()` and `sd()` are used to compute the standard deviation of the differences unless that standard deviation is supplied directly with the `sddiff()` option. The default is `sd(1)`.

`sd1(`*numlist*`)` specifies the standard deviation of the pretreatment group. Options `corr()`, `sd1()`, and `sd2()` are used to compute the standard deviation of the differences unless that standard deviation is supplied directly with the `sddiff()` option.

`sd2(`*numlist*`)` specifies the standard deviation of the posttreatment group. Options `corr()`, `sd1()`, and `sd2()` are used to compute the standard deviation of the differences unless that standard deviation is supplied directly with the `sddiff()` option.

`knownsd` requests that the standard deviation of the differences σ_d be treated as known in the computation. By default, the standard deviation is treated as unknown, and the computation is performed for a Student's t-based CI. If `knownsd` is specified, the computation is performed for a normal-based CI. `knownsd` may not be combined with `probwidth()` and is not allowed when computing the probability of CI width.

`fpc(`*numlist*`)` requests that a finite population correction be used in the computation. If `fpc()` has values between 0 and 1, it is interpreted as a sampling rate, n/N, where N is the total number of units in the population. When sample size n is specified, if `fpc()` has values greater than n, it is interpreted as a population size, but it is an error to have values between 1 and n. For sample-size determination, `fpc()` with a value greater than 1 is interpreted as a population size. It is an error for `fpc()` to have a mixture of sampling rates and population sizes.

`lower`, `upper`, `onesided`, `parallel`; see [PSS-3] **ciwidth**.

Table

`table`, `table()`, `notable`; see [PSS-3] **ciwidth, table**.

`saving()`; see [PSS-3] **ciwidth**.

Graph

`graph`, `graph()`; see [PSS-3] **ciwidth, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies an initial value for the sample size when iteration is used to compute the sample size. The default is to use a closed-form normal approximation to compute an initial sample size.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-3] **ciwidth**.

The following option is available with `ciwidth pairedmeans` but is not shown in the dialog box:

`notitle`; see [PSS-3] **ciwidth**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `ciwidth pairedmeans` command and the methodology for PrSS analysis for a CI for the difference between two means from paired samples. See [PSS-3] **Intro (ciwidth)** for a general introduction to PrSS analysis, and see [PSS-3] **ciwidth** for a general introduction to the `ciwidth` command. For PSS analysis for hypothesis tests, see [PSS-2] **power**.

Introduction

The analysis of paired means is commonly used in settings such as repeated-measures designs with before and after measurements on the same individual or cross-sectional studies of paired measurements from twins. For example, a company might initiate a voluntary exercise program and measure the average weight loss of participants from the first to sixth month. Or a school district might design an intensive remedial program for students with low math scores, and then analyze how much the students' math scores improve from the pretest to the posttest. For paired data, the inference is made on the mean difference accounting for the dependence between the two groups.

To compare two paired means, we assume that the two correlated samples are drawn from two normal populations with means μ_1 and μ_2 and standard deviations σ_1 and σ_2. The construction of a CI for the paired-means difference and its PrSS analysis are analogous to those of a CI for one population mean from *Introduction* in [PSS-3] **ciwidth onemean**, with mean μ replaced by the mean difference $\mu_d = \mu_2 - \mu_1$ and standard deviation σ by the standard deviation of the paired differences σ_d.

The `ciwidth pairedmeans` command provides PrSS analysis for a CI for the difference between two correlated means. We discuss the command details in the next section.

Using ciwidth pairedmeans

`ciwidth pairedmeans` computes sample size, CI width, or probability of CI width for a paired-means-difference CI. By default, a two-sided CI is assumed, and the confidence level is set to 95%. You may change the confidence level by specifying the `level()` option. Alternatively, you can specify the significance level in the `alpha()` option. You can specify the `upper` and `lower` options to request upper and lower one-sided CIs.

To compute sample size, you must specify the CI width in the `width()` option and the probability of CI width in the `probwidth()` option. To compute CI width, you must specify the sample size in the `n()` option and the probability of CI width in the `probwidth()` option. You can also compute the probability of CI width given the sample size in `n()` and CI width in `width()`.

For CIs for means, the CI width does not depend on the mean point estimates, the sample means, so they are not needed in the computations.

For all computations, you must specify either the standard deviation of the differences in the `sddiff()` option or the correlation between the paired observations in the `corr()` option. If you specify the `corr()` option, then individual standard deviations of the pretreatment and posttreatment groups may also be specified in the respective `sd1()` and `sd2()` options. By default, their values are set to 1. When the two standard deviations are equal, you may specify the common standard deviation in the `sd()` option instead of specifying them individually. By default, all computations assume an unknown standard deviation of the differences. When the standard deviation of the differences is known, you can specify the `knownsd` option to request a normal-based CI instead of the default Student's t-based CI.

By default, the computed sample size is rounded up. You can specify the `nfractional` option to see the corresponding fractional sample size; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. The `nfractional` option is allowed only for sample-size determination.

Some of `ciwidth pairedmeans`'s computations require iteration. For example, when the standard deviation of the differences is unknown, the sample-size computation requires iteration. The default initial value of the estimated sample size is obtained by using a closed-form normal approximation. It may be changed by specifying the `init()` option. See [PSS-3] **ciwidth** for the descriptions of other options that control the iteration procedure.

All computations assume an infinite population. For a finite population, use the `fpc()` option to specify a sampling rate or a population size.

In the following sections, we describe the use of `ciwidth pairedmeans` accompanied by examples for computing sample size, CI width, and probability of CI width.

Computing sample size

To compute the sample size required for a paired-means-difference CI to have the width no larger than a target width, you must specify the target CI width in the `width()` option and the desired probability of achieving the target CI width in the `probwidth()` option.

▷ Example 1: Sample size for a paired-means-difference CI

Consider an example similar to example 1 in [PSS-2] **power pairedmeans**. We study the low birthweight (LBW) infants as in Howell (2002, 186). The variable of interest is the decline of the Bayley mental development index (MDI) of 24-months-old infants compared with 6-months-old infants. We use an estimate of the standard deviation of the differences of 16.04.

We want to obtain the minimum sample size that is required to obtain a two-sided 95% CI with the width of at most 6, or the margin of error of 3, for the decline of the mean MDI score in a 24-month group. Because we use an estimate of the variance in our computations, we need to account for its sampling variability to ensure that the width of the actual CI in our new study does not exceed the target width. We use the probability of 0.98 that the width of a future CI does not exceed 6. We specify it in the `probwidth()` option, and the CI width of 6 in the `width()` option, and the estimate of the standard deviation of the differences 16.04 in the `sddiff()` option.

```
. ciwidth pairedmeans, width(6) probwidth(0.98) sddiff(16.04)
Performing iteration ...
Estimated sample size for a paired-means-difference CI
Student's t two-sided CI
Study parameters:
        level =   95.0000
     Pr_width =    0.9800
        width =    6.0000
         sd_d =   16.0400
Estimated sample size:
            N =       141
```

A sample of 141 infants is required for us to be 98% certain that the CI width will be no larger than 6 for the decline of the mean MDI score. The output reminds us that this was computed assuming a two-sided CI with the 95% confidence level.

As we mentioned in the previous section, sample-size determination requires iteration in the case of an unknown standard deviation. By default, `ciwidth pairedmeans` suppresses the iteration log, but we can choose to display it by specifying the `log` option.

◁

▷ Example 2: Specifying individual standard deviations

For instances in which you do not have the standard deviation of the differences, you can also compute the required sample size by specifying the individual standard deviations as well as the correlation among the paired observations.

Howell (2002) reported the estimates of group-specific standard deviations: 13.85 in the 6-month group and 12.95 in the 24-month group. Using the values of individual standard deviations and the standard deviation of the differences from the previous example, we obtain the correlation between the observations in the 6-month group and the 24-month group to be $(13.85^2 + 12.95^2 - 16.04^2)/(2 \times 13.85 \times 12.95) = 0.285$. To compute the sample size, we specify the group-specific standard deviations in `sd1()` and `sd2()` and the correlation in `corr()`.

```
. ciwidth pairedmeans, width(6) probwidth(0.98) corr(0.285) sd1(13.85)
> sd2(12.95)
Performing iteration ...
Estimated sample size for a paired-means-difference CI
Student's t two-sided CI
Study parameters:
        level =   95.0000          sd1 =   13.8500
     Pr_width =    0.9800          sd2 =   12.9500
        width =    6.0000         corr =    0.2850
         sd_d =   16.0403
Estimated sample size:
            N =       141
```

We obtain the same sample size as in example 1.

Study parameters are divided into two columns. The parameters that are always displayed are listed in the first column, and the parameters that are displayed only if they are specified are listed in the second column. The correlation and standard deviations are reported in the second column.

◁

Computing CI width

To compute the CI width, you must specify the sample size in the `n()` option and the desired probability of achieving the target CI width in the `probwidth()` option.

▷ Example 3: Precision of a paired-means-difference CI

Continuing with example 1, suppose that we have a sample of 120 subjects. To compute the CI width for this sample size, we specify 120 in the `n()` option and replace the `width()` option with it. We leave the other parameters unchanged:

```
. ciwidth pairedmeans, n(120) probwidth(0.98) sddiff(16.04)

Estimated width for a paired-means-difference CI
Student's t two-sided CI

Study parameters:

        level =   95.0000
            N =      120
     Pr_width =   0.9800
         sd_d =  16.0400

Estimated width:

        width =   6.5705
```

Compared with the CI width of 6 in example 1, the CI width of 6.6 in this example is larger, as would be expected with a smaller sample size of 120.

◁

▷ Example 4: One-sided CI

Rather than performing computations for the default two-sided CIs, we can request `upper` and `lower` one-sided CIs. We modify the command from example 3 by instead requesting a lower one-sided 95% CI for the difference of the MDI scores:

```
. ciwidth pairedmeans, n(120) probwidth(0.98) sddiff(16.04) lower

Estimated width for a paired-means-difference CI
Student's t lower CI

Study parameters:

        level =   95.0000
            N =      120
     Pr_width =   0.9800
         sd_d =  16.0400

Estimated width:

        width =   2.7504
```

Given the sample size of 120, the estimated width for a 95% lower one-sided CI for the differences of the MDI scores is about 2.75.

◁

▷ Example 5: Multiple values of study parameters

To investigate the effect of sample size on CI width, we can specify a list of sample sizes in the `n()` option:

```
. ciwidth pairedmeans, n(100(10)150) probwidth(0.98) sddiff(16.04)
Estimated width for a paired-means-difference CI
Student's t two-sided CI
```

level	N	Pr_width	width	sd_d
95	100	.98	7.294	16.04
95	110	.98	6.905	16.04
95	120	.98	6.57	16.04
95	130	.98	6.278	16.04
95	140	.98	6.021	16.04
95	150	.98	5.792	16.04

As expected, when the sample size increases, the CI width decreases.

For multiple values of parameters, the results are automatically displayed in a table. For more examples of tables, see [PSS-3] **ciwidth, table**. If you wish to produce sample-size and other curves, see [PSS-3] **ciwidth, graph**.

◁

Computing probability of CI width

To compute the probability that the width of a future CI will be no larger than the specified width, you must specify the sample size in the `n()` option and the target CI width in the `width()` option.

▷ Example 6: Probability of CI width for a paired-means-difference CI

Continuing with example 1, let's determine how certain we can be that the CI width in a future study will be no larger than a prespecified value for a given sample size. We use the same CI width and standard deviation as in example 1, and we use a sample size of 120 in `n()`.

```
. ciwidth pairedmeans, n(120) width(6) sddiff(16.04)
Estimated probability of width for a paired-means-difference CI
Student's t two-sided CI
Study parameters:
        level =   95.0000
            N =      120
        width =    6.0000
         sd_d =   16.0400
Estimated probability of width:
     Pr_width =    0.7175
```

The estimated probability is 72%. Compared with example 1, with a smaller sample size of 120, we are about 26% less certain that the CI width in a future study will be no larger than 6.

◁

Stored results

ciwidth pairedmeans stores the following in r():

Scalars

r(level)	confidence level
r(alpha)	significance level
r(N)	sample size
r(nfractional)	1 if nfractional is specified, 0 otherwise
r(onesided)	1 for a one-sided CI, 0 otherwise
r(Pr_width)	probability of CI width
r(Pr_width_a)	actual probability of CI width (for sample-size determination when probwidth() specified)
r(width)	CI width
r(width_a)	actual CI width (for sample-size determination when knownsd specified)
r(corr)	correlation between paired observations
r(sd_d)	standard deviation of the differences
r(sd1)	standard deviation of the pretreatment group
r(sd2)	standard deviation of the posttreatment group
r(sd)	common standard deviation
r(knownsd)	1 if option knownsd is specified, 0 otherwise
r(fpc)	finite population correction (if specified)
r(separator)	number of lines between separator lines in the table
r(divider)	1 if divider is requested in the table, 0 otherwise
r(init)	initial value for sample size
r(maxiter)	maximum number of iterations
r(iter)	number of iterations performed
r(tolerance)	requested parameter tolerance
r(deltax)	final parameter tolerance achieved
r(ftolerance)	requested distance of the objective function from zero
r(function)	final distance of the objective function from zero
r(converged)	1 if iteration algorithm converged, 0 otherwise

Macros

r(type)	ci
r(method)	pairedmeans
r(onesidedci)	upper or lower (for a one-sided CI)
r(columns)	displayed table columns
r(labels)	table column labels
r(widths)	table column widths
r(formats)	table column formats

Matrices

r(pss_table)	table of results

Methods and formulas

Consider a sequence of n paired observations denoted by x_{ij} for $i = 1, \ldots, n$ and groups $j = 1, 2$. Individual observations correspond to the pair (x_{i1}, x_{i2}), and inference is made on the differences within the pairs. Let $\mu_d = \mu_2 - \mu_1$ denote the mean difference, where μ_j is the population mean of group j, and $D_i = x_{i2} - x_{i1}$ denote the difference between paired individual observations.

PrSS analysis for a paired-means-difference CI is analogous to a one-mean CI where the sample of differences D_i's is treated as a single sample (Dixon and Massey 1983). The standard deviation of the differences, σ_d, is used in place of the one-sample standard deviation. For more information, see *Methods and formulas* in [PSS-3] **ciwidth onemean**.

References

Dixon, W. J., and F. J. Massey, Jr. 1983. *Introduction to Statistical Analysis*. 4th ed. New York: McGraw–Hill.

Howell, D. C. 2002. *Statistical Methods for Psychology*. 5th ed. Belmont, CA: Wadsworth.

Also see

[PSS-3] **ciwidth** — Precision and sample-size analysis for CIs

[PSS-3] **ciwidth, graph** — Graph results from the ciwidth command

[PSS-3] **ciwidth, table** — Produce table of results from the ciwidth command

[PSS-2] **power pairedmeans** — Power analysis for a two-sample paired-means test

[PSS-5] **Glossary**

[R] **ttest** — t tests (mean-comparison tests)

[R] **ztest** — z tests (mean-comparison tests, known variance)

ciwidth onevariance — Precision analysis for a one-variance CI

Description

`ciwidth onevariance` computes sample size, CI width, and probability of CI width for a CI for a population variance. It can compute sample size for a given CI width and probability of CI width. Alternatively, it can compute CI width for a given sample size and probability of CI width. It can also compute probability of CI width for a given sample size and CI width. The computation is available for the variance or the standard deviation. Also see [PSS-3] **ciwidth** for PrSS analysis for other CI methods.

For power and sample-size analysis for a one-sample variance test, see [PSS-2] **power onevariance**.

Quick start

Sample size required for a two-sided 95% CI for a population variance to have a width no larger than 2 with a probability of 90%, using population-variance estimate $v = 4$,

```
ciwidth onevariance 4, width(2) probwidth(0.9)
```

Same as above, but specify multiple widths and graph the result

```
ciwidth onevariance 4, width(2 3 4) probwidth(0.9) graph
```

CI width for a sample size of 30, with a 90% probability that the CI width will be no larger than the estimated value

```
ciwidth onevariance 4, n(30) probwidth(0.9)
```

Same as above, but specify standard deviations rather than variances

```
ciwidth onevariance 4, sd n(30) probwidth(0.9)
```

Same as above, but specify an upper one-sided CI

```
ciwidth onevariance 4, sd n(30) probwidth(0.9) upper
```

Menu

Statistics > Power, precision, and sample size

Syntax

Compute sample size

Variance scale

```
ciwidth onevariance v, width(numlist) probwidth(numlist) [ options ]
```

Standard deviation scale

```
ciwidth onevariance s, sd width(numlist) probwidth(numlist) [ options ]
```

Compute CI width

Variance scale

```
ciwidth onevariance v, probwidth(numlist) n(numlist) [ options ]
```

Standard deviation scale

```
ciwidth onevariance s, sd probwidth(numlist) n(numlist) [ options ]
```

Compute probability of CI width

Variance scale

```
ciwidth onevariance v, width(numlist) n(numlist) [ options ]
```

Standard deviation scale

```
ciwidth onevariance s, sd width(numlist) n(numlist) [ options ]
```

where v and s are variance and standard deviation, respectively. Each argument may be specified either as one number or as a list of values in parentheses (see **[U] 11.1.8 numlist**).

options	Description
`sd`	request computation using the standard deviation scale; default is the variance scale
Main	
* `level(`*numlist*`)`	confidence level; default is `level(95)`
* `alpha(`*numlist*`)`	significance level; default is `alpha(0.05)`
* `probwidth(`*numlist*`)`	probability of CI width; required to compute sample size and CI width
* `width(`*numlist*`)`	CI width; required to compute sample size and probability of CI width
* `n(`*numlist*`)`	sample size; required to compute CI width and probability of CI width
`nfractional`	allow fractional sample sizes
`lower`	lower one-sided CI; default is two-sided CI
`upper`	upper one-sided CI; default is two-sided CI
`onesided`	synonym for option `upper`
`parallel`	treat number lists in starred options or in command arguments as parallel when multiple values per option or argument are specified (do not enumerate all possible combinations of values)
Table	
[`no`]`table`[(*tablespec*)]	suppress table or display results as a table; see [PSS-3] **ciwidth, table**
`saving(`*filename* [`, replace`]`)`	save the table data to *filename*; use `replace` to overwrite existing *filename*
Graph	
`graph`[(*graphopts*)]	graph results; see [PSS-3] **ciwidth, graph**
Iteration	
`init(`#`)`	initial value for sample size; default is to use a closed-form normal approximation
`iterate(`#`)`	maximum number of iterations; default is `iterate(500)`
`tolerance(`#`)`	parameter tolerance; default is `tolerance(1e-12)`
`ftolerance(`#`)`	function tolerance; default is `ftolerance(1e-12)`
[`no`]`log`	suppress or display iteration log
[`no`]`dots`	suppress or display iterations as dots
`notitle`	suppress the title

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see **[U] 11.1.8 numlist**. Also see the `parallel` option.

`collect` is allowed; see **[U] 11.1.10 Prefix commands**.

`sd` does not appear in the dialog box; specification of `sd` is done automatically by the dialog box selected.

`notitle` does not appear in the dialog box.

where *tablespec* is

column[:*label*] [*column*[:*label*] [...]] [, *tableopts*]

column is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

column	Description	Symbol
`level`	confidence level	$100(1-\alpha)$
`alpha`	significance level	α
`N`	number of subjects	N
`Pr_width`	probability of CI width	p_{width}
`width`	CI width	w
`v`	variance	σ^2
`s`	standard deviation	σ
`_all`	display all supported columns	

Column `alpha` is shown in the default table in place of column `level` if `alpha()` is specified.

Column `s` is shown in the default table in place of column `v` if option `sd` is specified.

Options

`sd` specifies that the computation be performed using the standard deviation scale. The default is to use the variance scale. Specification of the `sd` option is done automatically when the dialog box for standard deviation is selected.

Main

`level()`, `alpha()`, `probwidth()`, `width()`, `n()`, `nfractional`; see [PSS-3] **ciwidth**. The `nfractional` option is allowed only for sample-size determination.

`lower`, `upper`, `onesided`, `parallel`; see [PSS-3] **ciwidth**.

Table

`table`, `table()`, `notable`; see [PSS-3] **ciwidth, table**.

`saving()`; see [PSS-3] **ciwidth**.

Graph

`graph`, `graph()`; see [PSS-3] **ciwidth, graph**. Also see the *column* table for a list of symbols used by the graphs.

Iteration

`init(#)` specifies an initial value for the sample size when iteration is used to compute the sample size. The default is to use a closed-form normal approximation to compute an initial sample size.

`iterate()`, `tolerance()`, `ftolerance()`, `log`, `nolog`, `dots`, `nodots`; see [PSS-3] **ciwidth**.

The following option is available with `ciwidth onevariance` but is not shown in the dialog box:

`notitle`; see [PSS-3] **ciwidth**.

Remarks and examples

Remarks are presented under the following headings:

This entry describes the `ciwidth onevariance` command and the methodology for PrSS analysis for a CI for a population variance. See [PSS-3] **Intro (ciwidth)** for a general introduction to PrSS analysis, and see [PSS-3] **ciwidth** for a general introduction to the `ciwidth` command. For PSS analysis for hypothesis tests, see [PSS-2] **power**.

Introduction

The study of variance arises in cases where investigators are interested in measuring the variability of a process. For example, the accuracy of a thermometer in taking measurements, the variation in the weights of potato chips from one bag to another, and the variation in mileage across automobiles of the same model. Before undertaking the actual study, we may want to find the optimal sample size to measure the variations with a certain precision.

We are interested in a CI for the population variance σ^2. The precision of a CI is commonly measured by its width w. For example, a two-sided one-variance CI is formed as $[\hat{\sigma}^2 - w_{\text{lower}}, \hat{\sigma}^2 + w_{\text{upper}}]$, where $\hat{\sigma}^2$ is the variance point estimate. The CI width, the distance between the upper and lower limits, is $w = w_{\text{lower}} + w_{\text{upper}}$. The smaller the w the more precise the CI.

In PrSS analysis, it is usually of interest to determine the sample size that would be sufficient for a CI to have a prespecified width in a future study. Generally, larger sample sizes lead to more precise CIs. To compute the required sample size, we need to know the expression for w.

Just like with a one-mean CI, the CI width w for a one-variance CI depends on the sample estimate s^2 of the variance and thus will vary from one sample to another. To ensure that, in a future study, a CI has the desired width, this sampling variability of w must be accounted for when computing the required sample size. Kupper and Hafner (1989) introduce what we call the probability of CI width, which specifies the probability of a future CI to have the width of no larger than some prespecified CI width for a given sample size. This probability is defined based on the assumption of a χ^2 distribution for the sample variance s^2; see *Methods and formulas* for details.

You can use `ciwidth onevariance` to perform PrSS analysis for a CI for a population variance or standard deviation. We discuss the command details in the next section.

Using ciwidth onevariance

`ciwidth onevariance` computes sample size, CI width, or probability of CI width for a one-variance CI. By default, a two-sided CI is assumed, and the confidence level is set to 95%. You may change the confidence level by specifying the `level()` option. Alternatively, you can specify the significance level in the `alpha()` option. You can specify the `upper` and `lower` options to request upper and lower one-sided CIs.

To compute sample size, you must specify the CI width in the `width()` option and the probability of CI width in the `probwidth()` option. To compute CI width, you must specify the sample size in the `n()` option and the probability of CI width in the `probwidth()` option. You can also compute the probability of CI width given the sample size in `n()` and CI width in `width()`. In each case, you must also specify the variance v or standard deviation s as the command argument.

By default, the computation is performed for the variance parameter. You can use the `sd` option to specify the computation for the standard deviation.

By default, the computed sample size is rounded up. You can specify the `nfractional` option to see the corresponding fractional sample size; see *Fractional sample sizes* in [PSS-4] **Unbalanced designs** for an example. The `nfractional` option is allowed only for sample-size determination.

Some of `ciwidth onevariance`'s computations require iteration. For example, the sample-size computation requires iteration. The default initial value of the estimated sample size is obtained by using a closed-form normal approximation. It may be changed by specifying the `init()` option. See [PSS-3] **ciwidth** for the descriptions of other options that control the iteration procedure.

In the following sections, we describe the use of `ciwidth onevariance` accompanied by examples for computing sample size, CI width, and probability of CI width.

Computing sample size

To compute the sample size required for a one-variance CI to have the width no larger than a target width, you must specify the target CI width in the `width()` option and the desired probability of achieving the target CI width in the `probwidth()` option. You must also specify the variance v or standard deviation s as the command argument.

▷ Example 1: Sample size for a one-variance CI

Consider a study where interest lies in measuring the variability in mileage (measured in miles per gallon) of automobiles of a certain car manufacturer. Industry-wide standards maintain that a variation of at most two miles per gallon (mpg) from an average value is acceptable for commercial production.

We want to compute the required sample size such that the width of a two-sided 95% CI for the variance will not exceed 2 mpg with a 96% certainty. Suppose the variance is 4. We specify the variance $v = 4$ after the command name, the CI width of 2 in the `width()` option, and the probability of obtaining the target CI width in the `probwidth()` option:

```
. ciwidth onevariance 4, probwidth(0.96) width(2)
Performing iteration ...
Estimated sample size for a one-variance CI
Chi-squared two-sided CI
Study parameters:
        level =     95.00
     Pr_width =    0.9600
        width =    2.0000
            v =    4.0000
Estimated sample size:
            N =       183
```

We find that a sample of 183 cars is required for this study.

As we mentioned in the previous section, sample-size computation requires iteration. By default, `ciwidth onevariance` suppresses the iteration log, but it can be displayed by specifying the `log` option.

◁

Computing CI width

To compute the CI width, you must specify the sample size in the `n()` option and the desired probability of achieving the target CI width in the `probwidth()` option. You must also specify the variance v or standard deviation s as the command argument.

▷ Example 2: CI width for a one-variance CI

Continuing with example 1, suppose that we anticipate obtaining a sample of 150 cars and want to compute the CI width corresponding to this sample size. To compute the CI width, we use the study parameters from example 1, but we now specify the sample size of 150 in the `n()` option instead of the `width()` option:

```
. ciwidth onevariance 4, probwidth(0.96) n(150)

Estimated width for a one-variance CI
Chi-squared two-sided CI

Study parameters:

        level =     95.00
            N =       150
     Pr_width =    0.9600
            v =    4.0000

Estimated width:

        width =    2.2571
```

With a sample size smaller than the one we estimated in example 1, the width of the variance CI increases to about 2.3.

◁

▷ Example 3: Standard deviation scale

Continuing with example 2, suppose that we would like to compute the CI width using the standard deviation scale instead. Above we used a variance of 4; taking its square root, we specify a standard deviation of 2 as the command argument and the `sd` option. The other parameters remain unchanged from the example above.

```
. ciwidth onevariance 2, probwidth(0.96) n(150) sd

Estimated width for a one-standard-deviation CI
Chi-squared two-sided CI

Study parameters:

        level =     95.00
            N =       150
     Pr_width =    0.9600
            s =    2.0000

Estimated width:

        width =    0.5060
```

For a sample size of 150, probability of CI width of 0.96, and standard deviation of 2, the estimated largest width for a standard deviation CI is 0.51 mpg.

◁

Computing probability of CI width

To compute the probability that the width of a future CI will be no larger than the specified width, you must specify the sample size in the `n()` option and the target CI width in the `width()` option.

▷ Example 4: Computing probability of CI width for a one-variance CI

Because CI width may vary across samples, we may want to estimate the probability that the width of a future CI will not exceed a target value. Continuing with example 1, suppose that we have only enough resources to test the mileage of 150 automobiles. We can estimate the probability that the CI width will not exceed a target width of 2, given this sample size and a variance of 4:

```
. ciwidth onevariance 4, width(2) n(150)
Estimated probability of width for a one-variance CI
Chi-squared two-sided CI
Study parameters:
        level =     95.00
            N =       150
        width =    2.0000
            v =    4.0000
Estimated probability of width:
     Pr_width =    0.7453
```

For this sample size, we can be 75% certain that the CI width will be no more than 2 mpg for a 95% CI for the variance.

◁

▷ Example 5: Multiple values of study parameters

As a variation of example 4, we would like to see the effect of an increasing variance on the estimated probability of achieving a target CI width of 2. We compute the probability of CI width for a range of variances between 3 and 5, with the step size of 0.5, by specifying the corresponding numlist as the argument for `ciwidth onevariance`.

```
. ciwidth onevariance (3(0.5)5), width(2) n(150)
Estimated probability of width for a one-variance CI
Chi-squared two-sided CI
```

level	N	Pr_width	width	v
95	150	.9996	2	3
95	150	.969	2	3.5
95	150	.7453	2	4
95	150	.3591	2	4.5
95	150	.1074	2	5

The output shows that the probability of achieving the target CI width decreases rapidly as we increase the variance.

For multiple values of parameters, the results are automatically displayed in a table, as we see above. For more examples of tables, see [PSS-3] **ciwidth, table**. If you wish to produce sample-size and other curves, see [PSS-3] **ciwidth, graph**.

◁

Stored results

`ciwidth onevariance` stores the following in `r()`:

Scalars

`r(level)`	confidence level
`r(alpha)`	significance level
`r(N)`	sample size
`r(nfractional)`	1 if `nfractional` is specified, 0 otherwise
`r(onesided)`	1 for a one-sided CI, 0 otherwise
`r(Pr_width)`	probability of CI width
`r(Pr_width_a)`	actual probability of CI width (for sample-size determination when `probwidth()` specified)
`r(width)`	CI width
`r(v)`	variance
`r(s)`	standard deviation
`r(separator)`	number of lines between separator lines in the table
`r(divider)`	1 if `divider` is requested in the table, 0 otherwise
`r(init)`	initial value for sample size
`r(maxiter)`	maximum number of iterations
`r(iter)`	number of iterations performed
`r(tolerance)`	requested parameter tolerance
`r(deltax)`	final parameter tolerance achieved
`r(ftolerance)`	requested distance of the objective function from zero
`r(function)`	final distance of the objective function from zero
`r(converged)`	1 if iteration algorithm converged, 0 otherwise

Macros

`r(type)`	`ci`
`r(method)`	`onevariance`
`r(scale)`	`variance` or `standard deviation`
`r(onesidedci)`	`upper` or `lower` (for a one-sided CI)
`r(columns)`	displayed table columns
`r(labels)`	table column labels
`r(widths)`	table column widths
`r(formats)`	table column formats

Matrices

`r(pss_table)`	table of results

Methods and formulas

See *Methods and formulas* in [R] **ci** for a general description of CIs for variances.

Consider a random sample $\mathbf{x} = (x_1, \ldots, x_n)$ of size n from a normal population with mean μ and variance σ^2. We are interested in a CI for the population variance σ^2.

A general two-sided CI is defined as $[\ll(\mathbf{x}), ul(\mathbf{x})]$, a lower one-sided CI as $[\ll(\mathbf{x}), \infty)$, and an upper one-sided CI as $(0, ul(\mathbf{x})]$, where $\ll(\mathbf{x}) = \ll$ and $ul(\mathbf{x}) = ul$ are the respective lower and upper confidence limits. Let w be the CI width.

Let

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \text{and} \qquad s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2$$

be the sample mean and the sample variance, respectively.

A two-sided CI for the variance σ^2 is constructed as

$$[s^2 - w_{\text{lower}}, s^2 + w_{\text{upper}}]$$

where w_{lower} and w_{upper} are such that $w_{\text{upper}} + w_{\text{lower}} = w$.

Lower and upper one-sided CIs are constructed as

$$[s^2 - w_{\text{lower}}, \infty)$$
$$(0, s^2 + w_{\text{upper}}]$$

We define $w = w_{\text{lower}}$ for lower one-sided CIs and $w = w_{\text{upper}}$ for upper one-sided CIs.

We use the CI width w as our measure of CI precision. Let $100(1-\alpha)\%$ denote the confidence level, where $0 \le \alpha \le 1$ is the corresponding significance level.

The following formulas are based on Dixon and Massey (1983). The sampling distribution of the statistic $\chi^2 = (n-1)s^2/\sigma^2$ follows a χ^2 distribution with $n-1$ degrees of freedom. Let $\chi^2_{n-1,p}$ be the pth quantile of the χ^2 distribution with $n-1$ degrees of freedom.

Based on the χ^2 distribution, the constructed CIs are:

$$\begin{cases} \left[\frac{(n-1)s^2}{\chi^2_{n-1,1-\alpha/2}}, \frac{(n-1)s^2}{\chi^2_{n-1,\alpha/2}}\right] & \text{for a two-sided CI} \\ \left[\frac{(n-1)s^2}{\chi^2_{n-1,1-\alpha}}, \infty\right) & \text{for a lower CI} \\ \left(0, \frac{(n-1)s^2}{\chi^2_{n-1,\alpha}}\right] & \text{for an upper CI} \end{cases}$$

Similarly to the case of an unknown standard deviation for a one-mean CI, the CI width depends on the sample standard deviation. Again, using the fact that $(n-1)s^2/\sigma^2$ follows a χ^2 distribution with $n-1$ degrees of freedom, we can compute the probability that the CI width is no larger than a specified value w.

Let $\chi^2_{n-1}(\cdot)$ be the c.d.f. of the χ^2 distribution with $n-1$ degrees of freedom. The probability of CI width, $\Pr(w)$, is

$$\Pr(w) = \begin{cases} \chi^2_{n-1}\left(\frac{w^2}{\sigma^2(1/\chi^2_{n-1,1-\alpha/2} - 1/\chi^2_{n-1,\alpha/2})}\right) & \text{for a two-sided CI} \\ \chi^2_{n-1}\left(\frac{w^2}{\sigma^2\{1/(n-1) - 1/\chi^2_{n-1,\alpha}\}}\right) & \text{for a lower one-sided CI} \\ \chi^2_{n-1}\left(\frac{w^2}{\sigma^2\{1/\chi^2_{n-1,1-\alpha} - 1/(n-1)\}}\right) & \text{for an upper one-sided CI} \end{cases} \tag{1}$$

We can compute the desired CI width from (1).

$$w = \begin{cases} \sigma^2\chi^2_{n-1,\Pr(w)}\left(1/\chi^2_{n-1,1-\alpha/2} - 1/\chi^2_{n-1,\alpha/2}\right) & \text{for a two-sided CI} \\ \sigma^2\chi^2_{n-1,\Pr(w)}\{1/(n-1) - 1/\chi^2_{n-1,\alpha}\} & \text{for a lower one-sided CI} \\ \sigma^2\chi^2_{n-1,\Pr(w)}\{1/\chi^2_{n-1,1-\alpha} - 1/(n-1)\} & \text{for an upper one-sided CI} \end{cases} \tag{2}$$

where $\chi^2_{n-1,p}$ is the pth quantile of a χ^2 distribution with $n-1$ degrees of freedom.

We can solve for the sample size iteratively using (2). The default initial value for the sample size is computed using the closed-form formula based on a large-sample normal approximation. Specifically, for a large n, the log-transformed sample variance is approximately normally distributed with mean $2\ln(\sigma)$ and standard deviation $\sqrt{2/n}$.

If the `nfractional` option is not specified, the computed sample size is rounded up.

References

Dixon, W. J., and F. J. Massey, Jr. 1983. *Introduction to Statistical Analysis*. 4th ed. New York: McGraw–Hill.

Kupper, L. L., and K. B. Hafner. 1989. How appropriate are popular sample size formulas? *American Statistician* 43: 101–105. https://doi.org/10.2307/2684511.

Also see

[PSS-3] **ciwidth** — Precision and sample-size analysis for CIs

[PSS-3] **ciwidth, graph** — Graph results from the ciwidth command

[PSS-3] **ciwidth, table** — Produce table of results from the ciwidth command

[PSS-2] **power onevariance** — Power analysis for a one-sample variance test

[PSS-5] **Glossary**

[R] **ci** — Confidence intervals for means, proportions, and variances

[PSS-4] Design specification

Description Syntax Options Remarks and examples Also see

Description

This entry describes the specifications of unbalanced designs for two-sample studies, including power and sample-size analysis for two-sample hypothesis tests and precision and sample-size analysis of two-sample CIs. See [PSS-2] **power** for a general introduction to the `power` command for power analysis and [PSS-3] **ciwidth** for a general introduction to the `ciwidth` command for precision analysis.

Syntax

Two samples, compute sample size for unbalanced designs

Compute total sample size

cmdname ..., `nratio(`*numlist*`)` [`nfractional`] ...

Compute one group size given the other

cmdname ..., `n#(`*numlist*`) compute(N1|N2)` [`nfractional`] ...

Two samples, specify sample size for unbalanced designs

Specify total sample size and allocation ratio

cmdname ..., `n(`*numlist*`) nratio(`*numlist*`)` [`nfractional`] ...

Specify one of the group sizes and allocation ratio

cmdname ..., `n#(`*numlist*`) nratio(`*numlist*`)` [`nfractional`] ...

Specify total sample size and one of the group sizes

cmdname ..., `n(`*numlist*`) n#(`*numlist*`)` ...

Specify group sizes

cmdname ..., `n1(`*numlist*`) n2(`*numlist*`)` ...

cmdname can be either `power` for power analysis or `ciwidth` for precision analysis.

twosampleopts	Description
* `n(`*numlist*`)`	total sample size
* `n1(`*numlist*`)`	sample size of the control group
* `n2(`*numlist*`)`	sample size of the experimental group
* `nratio(`*numlist*`)`	ratio of sample sizes, `N2/N1`; default is `nratio(1)`, meaning equal group sizes
`compute(N1 | N2)`	solve for `N1` given `N2` or for `N2` given `N1`
`nfractional`	allow fractional sample sizes

*Specifying a list of values in at least two starred options, or at least two command arguments, or at least one starred option and one argument results in computations for all possible combinations of the values; see [U] **11.1.8 numlist**. Also see the `parallel` option.

Options

Main

`n(`*numlist*`)` specifies the total number of subjects in the study.

When used with `power`, this sample size is used for power or effect-size determination. If `n()` is specified, the power is computed. If `n()` and `power()` or `beta()` are specified, the minimum effect size that is likely to be detected in a study is computed.

When used with `ciwidth`, this sample size is used to compute the CI width and probability of CI width.

`n1(`*numlist*`)` specifies the number of subjects in the control group.

When used with `power`, this sample size is used for power or effect-size determination.

When used with `ciwidth`, this sample size is used to compute the CI width and probability of CI width.

`n2(`*numlist*`)` specifies the number of subjects in the experimental group. It is used for the same computations as `n1(`*numlist*`)`, as mentioned above.

`nratio(`*numlist*`)` specifies the sample-size ratio of the experimental group relative to the control group, `N2/N1`. The default is `nratio(1)`, meaning equal allocation between the two groups.

When used with `power`, this ratio is used for power and effect-size determination for two-sample tests.

When used with `ciwidth`, this ratio is used for computing CI width and probability of CI width for two-sample CIs.

`compute(N1 | N2)` requests that one of the group sample sizes be computed given the other one, instead of the total sample size. To compute the control-group sample size, you must specify `compute(N1)` and the experimental-group sample size in `n2()`. Alternatively, to compute the experimental-group sample size, you must specify `compute(N2)` and the control-group sample size in `n1()`.

`nfractional` specifies that fractional sample sizes be allowed. When this option is specified, fractional sample sizes are used in the intermediate computations and are also displayed in the output.

Remarks and examples

Remarks are presented under the following headings:

By default, for two-sample tests and CIs, the `power` and `ciwidth` commands assume a balanced design, but you may request an unbalanced design. A common way of specifying an unbalanced design is by specifying the `nratio()` option. You can also specify group sample sizes directly in the `n1()` and `n2()` options.

All considered options that accept arguments allow you to specify either one value # or *numlist*, a list of values as described in **[U] 11.1.8 numlist**. For simplicity, we demonstrate these options using only one value.

Below we describe in detail the specifications of unbalanced designs for two-sample methods and the handling of fractional sample sizes. As can be seen in *Syntax*, the specification of sample sizes, either total or group sample sizes, is the same across the `power` and `ciwidth` commands. In *Two samples*, we primarily use `power` in our examples, but the `n()`, `n1()`, `n2()`, `nratio()`, and `compute()` options shown there are used in the same fashion with `ciwidth`. Similarly, in *Fractional sample sizes*, we primarily use `ciwidth` in our examples, but the `nfractional` option shown there would be used in the same fashion with `power`.

Two samples

All two-sample methods, such as `power twomeans` and `ciwidth twomeans`, support the following options for specifying sample sizes: the total sample size `n()`, individual sample sizes `n1()` and `n2()`, and allocation ratio `nratio()`. The `compute()` option is useful if you want to compute one of the group sizes given the other one, instead of the total sample size.

We first describe the specifications and then demonstrate their use in real examples. The example below uses the `power` command, but the same principle applies to the `ciwidth` command.

We start with the sample-size determination—the default computation performed by the `power` command. The "switch" option for sample-size determination is the `power()` option. If you do not specify this option, it is implied with the default value of 0.8 corresponding to 80% power.

By default, group sizes are assumed to be equal; that is, the `nratio(1)` option is implied.

```
. power ..., [nratio(1)] ...
```

You can supply a different allocation ratio, n_2/n_1, to `nratio()` to request an unbalanced design.

```
. power ..., nratio(#) ...
```

To compute power or effect size, you must supply information about group sample sizes to `power`. Similarly, to compute CI width or probability of CI width, you must supply information about group sample sizes to `ciwidth`. There are several ways for you to do this. The simplest one, perhaps, is to specify the total sample size in the `n()` option.

```
. power ..., n(#) ...
```

The specification above assumes a balanced design in which the two group sizes are the same.

To request an unbalanced design, you can specify the desired allocation ratio between the two groups in the `nratio()` option:

```
. power ..., n(#) nratio(#) ...
```

The `nratio()` options assumes that the supplied values are the ratios of the second (experimental or comparison) group to the first (control or reference) group.

Alternatively, you can specify the two group sizes directly,

```
. power ..., n1(#) n2(#) ...
```

or you can specify one of the group sizes and the allocation ratio:

```
. power ..., n1(#) nratio(#) ...
. power ..., n2(#) nratio(#) ...
```

Also supported, but perhaps more rarely used, is a combination of the total sample size and one of the group sizes:

```
. power ..., n(#) n1(#) ...
. power ..., n(#) n2(#) ...
```

Below we demonstrate the described specifications using the `power twomeans` command, which provides power and sample-size analysis for tests of two independent means; see [PSS-2] **power twomeans** for details. In all examples, we use a value of 0 for the control-group mean, a value of 1 for the experimental-group mean, and the default values of the other study parameters.

▷ Example 1: Sample-size determination for a balanced design

By default, `power twomeans` computes sample size for a balanced design.

```
. power twomeans 0 1

Performing iteration ...

Estimated sample sizes for a two-sample means test
t test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1

Study parameters:

        alpha =    0.0500
        power =    0.8000
        delta =    1.0000
           m1 =    0.0000
           m2 =    1.0000
           sd =    1.0000

Estimated sample sizes:

            N =        34
  N per group =        17
```

The required total sample size is 34, with 17 subjects in each group.

The above is equivalent to specifying the `nratio(1)` option:

```
. power twomeans 0 1, nratio(1)
Performing iteration ...
Estimated sample sizes for a two-sample means test
t test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    1.0000
           m1 =    0.0000
           m2 =    1.0000
           sd =    1.0000
Estimated sample sizes:
            N =        34
  N per group =        17
```

Example 2: Sample-size determination for an unbalanced design

To compute sample size for an unbalanced design, we specify the ratio of the experimental-group size to the control-group size in the `nratio()` option. For example, if we anticipate twice as many subjects in the experimental group as in the control group, we compute the corresponding sample size by specifying `nratio(2)`:

```
. power twomeans 0 1, nratio(2)
Performing iteration ...
Estimated sample sizes for a two-sample means test
t test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1
Study parameters:
        alpha =    0.0500
        power =    0.8000
        delta =    1.0000
           m1 =    0.0000
           m2 =    1.0000
           sd =    1.0000
        N2/N1 =    2.0000
Estimated sample sizes:
            N =        39
           N1 =        13
           N2 =        26
```

The required total sample size is 39, with 13 subjects in the control group and 26 subjects in the experimental group. Generally, unbalanced designs require more subjects than the corresponding balanced designs. This is the case for precision and sample size as well.

▷ Example 3: Power determination for a balanced design

To compute power for a balanced design, we specify the total sample size in the `n()` option:

```
. power twomeans 0 1, n(30)

Estimated power for a two-sample means test
t test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1

Study parameters:

        alpha =    0.0500
            N =        30
  N per group =        15
        delta =    1.0000
           m1 =    0.0000
           m2 =    1.0000
           sd =    1.0000

Estimated power:

        power =    0.7529
```

Equivalently, we specify one of the group sizes in the `n1()` or `n2()` option:

```
. power twomeans 0 1, n1(15)

Estimated power for a two-sample means test
t test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1

Study parameters:

        alpha =    0.0500
            N =        30
           N1 =        15
           N2 =        15
        delta =    1.0000
           m1 =    0.0000
           m2 =    1.0000
           sd =    1.0000

Estimated power:

        power =    0.7529
```

Both specifications imply the `nratio(1)` option.

◁

▷ Example 4: Power determination for an unbalanced design

As we described in *Two samples*, there are a number of ways for you to request an unbalanced design for power and precision determination. Below we provide an example for each specification.

Specifying total sample size and allocation ratio

Similarly to example 2 but for power determination, we request an unbalanced design with twice as many subjects in the experimental group as in the control group by specifying the `nratio(2)` option:

```
. power twomeans 0 1, n(30) nratio(2)

Estimated power for a two-sample means test
t test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1

Study parameters:

        alpha =    0.0500
            N =        30
           N1 =        10
           N2 =        20
        N2/N1 =    2.0000
        delta =    1.0000
           m1 =    0.0000
           m2 =    1.0000
           sd =    1.0000

Estimated power:

        power =    0.7029
```

The computed power of 0.7029 is lower than the power of 0.7529 of the corresponding balanced design from example 3.

Specifying group sample sizes

Instead of the total sample size and the allocation ratio, we can specify the group sample sizes directly in the `n1()` and `n2()` options:

```
. power twomeans 0 1, n1(10) n2(20)

Estimated power for a two-sample means test
t test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1

Study parameters:

        alpha =    0.0500
            N =        30
           N1 =        10
           N2 =        20
        N2/N1 =    2.0000
        delta =    1.0000
           m1 =    0.0000
           m2 =    1.0000
           sd =    1.0000

Estimated power:

        power =    0.7029
```

Specifying one of the group sample sizes and allocation ratio

Alternatively, we can specify one of the group sizes and the allocation ratio. Here we specify the control-group size.

```
. power twomeans 0 1, n1(10) nratio(2)

Estimated power for a two-sample means test
t test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1

Study parameters:

        alpha =    0.0500
            N =        30
           N1 =        10
           N2 =        20
        N2/N1 =    2.0000
        delta =    1.0000
           m1 =    0.0000
           m2 =    1.0000
           sd =    1.0000

Estimated power:

        power =    0.7029
```

We could have specified the experimental-group size instead:

```
. power twomeans 0 1, n2(20) nratio(2)
  (output omitted)
```

Specifying total sample size and one of the group sample sizes

Finally, we can specify a combination of the total sample size and one of the group sizes—the control group:

```
. power twomeans 0 1, n1(10) n(30)

Estimated power for a two-sample means test
t test assuming sd1 = sd2 = sd
H0: m2 = m1  versus  Ha: m2 != m1

Study parameters:

        alpha =    0.0500
            N =        30
           N1 =        10
           N2 =        20
        N2/N1 =    2.0000
        delta =    1.0000
           m1 =    0.0000
           m2 =    1.0000
           sd =    1.0000

Estimated power:

        power =    0.7029
```

or the experimental group:

```
. power twomeans 0 1, n2(20) n(30)
  (output omitted)
```

◁

Options `n()`, `n1()`, and `n2()` require integer numbers. When you specify the `n1()` and `n2()` options, your sample sizes are guaranteed to be integers. This is not necessarily true for other specifications for which the resulting sample sizes may be fractional. See *Fractional sample sizes* for details about how the `power` and `ciwidth` commands handle fractional sample sizes.

We show examples using the `ciwidth` command, but the same principles apply to the `power` command.

Fractional sample sizes

Certain sample-size specifications may lead to fractional sample sizes. For example, if you specify an odd value for the total sample size of a two-sample study, the two group sample sizes would have to be fractional to accommodate the specified total sample size. Also, if you specify the `nratio()` option with a two-sample method, the resulting sample sizes may be fractional.

By default, the `power` and `ciwidth` commands round sample sizes to integers and use integer values in the computations. To ensure conservative results, the commands round down the input sample sizes and round up the output sample sizes.

▷ Example 5: Output sample sizes

For example, when we compute sample size, the sample size is rounded up to the nearest integer by default:

```
. ciwidth onemean, width(1) probwidth(0.9)

Performing iteration ...

Estimated sample size for a one-mean CI
Student's t two-sided CI

Study parameters:

        level =     95.00
     Pr_width =    0.9000
        width =    1.0000
           sd =    1.0000

Estimated sample size:

            N =        24
```

We computed sample size for a one-sample mean CI; see [PSS-3] **ciwidth onemean** for details.

We can specify the `nfractional` option to see the corresponding fractional sample size:

```
. ciwidth onemean, width(1) probwidth(0.9) nfractional

Performing iteration ...

Estimated sample size for a one-mean CI
Student's t two-sided CI

Study parameters:

        level =     95.00
     Pr_width =    0.9000
        width =    1.0000
           sd =    1.0000

Estimated sample size:

            N =   23.8582
```

The sample size of 24 reported above is the ceiling for the fractional sample size 23.86.

We can also compute the actual CI width corresponding to the rounded sample size:

```
. ciwidth onemean, n(24) probwidth(0.9)
Estimated width for a one-mean CI
Student's t two-sided CI
Study parameters:
        level =     95.00
            N =        24
     Pr_width =    0.9000
           sd =    1.0000
Estimated width:
        width =    0.9963
```

The actual CI width corresponding to the sample size of 24 is smaller than the specified CI width of 1 from the two previous examples because the sample size was rounded up.

◁

On the other hand, the `power` and `ciwidth` commands round down the input sample sizes.

▷ Example 6: Input sample sizes

For example, let's use `ciwidth twomeans` to compute the CI width for a two-means-difference CI, given a total sample size of 51. To be 95% certain that the width of a future CI for this sample size will be no larger than the value we estimate, we specify `probwidth(0.95)`. We use the default 95% confidence level; see [PSS-3] **ciwidth twomeans** for details.

```
. ciwidth twomeans, probwidth(0.95) n(51)
Estimated width for a two-means-difference CI
Student's t two-sided CI assuming sd1 = sd2 = sd
Study parameters:
        level =     95.00
            N =        51
     Pr_width =    0.9500
           sd =    1.0000
Actual sample sizes:
            N =        50
  N per group =        25
Estimated width:
        width =    1.3253
```

By default, `ciwidth twomeans` assumes a balanced design. To accommodate a balanced design, the command rounds down the group sample sizes from 25.5 to 25 for an actual total sample size of 50.

When the specified sample sizes differ from the resulting rounded sample sizes, the actual sample sizes used in the computations are reported. In our example, we requested a total sample size of 51, but the actual sample size used to compute the CI width was 50.

We can specify the `nfractional` option to request that fractional sample sizes be used in the computations.

```
. ciwidth twomeans, probwidth(0.95) n(51) nfractional
Estimated width for a two-means-difference CI
Student's t two-sided CI assuming sd1 = sd2 = sd
Study parameters:
        level =     95.00
            N =   51.0000
  N per group =   25.5000
     Pr_width =    0.9500
           sd =    1.0000
Estimated width:
        width =    1.3097
```

The fractional group sample sizes of 25.5 are now used in the computations.

If we want to preserve the total sample size of 51 and ensure that group sample sizes are integers, we can specify the group sizes directly:

```
. ciwidth twomeans, probwidth(0.95) n1(25) n2(26)
Estimated width for a two-means-difference CI
Student's t two-sided CI assuming sd1 = sd2 = sd
Study parameters:
        level =     95.00
            N =        51
           N1 =        25
           N2 =        26
        N2/N1 =    1.0400
     Pr_width =    0.9500
           sd =    1.0000
Estimated width:
        width =    1.3099
```

Alternatively, we can specify one of the group sizes (or the total sample size) and the corresponding allocation ratio $n_2/n_1 = 26/25 = 1.04$:

```
. ciwidth twomeans, probwidth(0.95) n1(25) nratio(1.04)
Estimated width for a two-means-difference CI
Student's t two-sided CI assuming sd1 = sd2 = sd
Study parameters:
        level =     95.00
            N =        51
           N1 =        25
           N2 =        26
        N2/N1 =    1.0400
     Pr_width =    0.9500
           sd =    1.0000
Estimated width:
        width =    1.3099
```

We obtain the same CI width of 1.31.

In the command above, the sample-size ratio we specified resulted in integer sample sizes. This may not always be the case. For example, if we specify the sample-size ratio of 1.3,

```
. ciwidth twomeans, probwidth(0.95) n1(25) nratio(1.3)
Estimated width for a two-means-difference CI
Student's t two-sided CI assuming sd1 = sd2 = sd
Study parameters:
        level =     95.00
           N1 =        25
        N2/N1 =    1.3000
     Pr_width =    0.9500
           sd =    1.0000
Actual sample sizes:
            N =        57
           N1 =        25
           N2 =        32
        N2/N1 =    1.2800
Estimated width:
        width =    1.2352
```

the experimental-group size of 32.5 is rounded down to 32. The total sample size used in the computation is 57, and the actual sample-size ratio is 1.28.

As before, we can specify the `nfractional` option to use the fractional experimental-group size of 32.5 in the computations:

```
. ciwidth twomeans, probwidth(0.95) n1(25) nratio(1.3) nfractional
Estimated width for a two-means-difference CI
Student's t two-sided CI assuming sd1 = sd2 = sd
Study parameters:
        level =     95.00
            N =   57.5000
           N1 =   25.0000
           N2 =   32.5000
        N2/N1 =    1.3000
     Pr_width =    0.9500
           sd =    1.0000
Estimated width:
        width =    1.2300
```

◁

Also see

[PSS-2] **power** — Power and sample-size analysis for hypothesis tests

[PSS-3] **ciwidth** — Precision and sample-size analysis for CIs

[PSS-5] **Glossary**

[PSS-5] Glossary of common terms

Glossary

1 : M matched case–control study. See *matched study*.

2 × 2 contingency table. A 2 × 2 contingency table is used to describe the association between a binary independent variable and a binary response variable of interest.

2 × 2 × K contingency table. See *stratified 2 × 2 tables*.

acceptance region. In hypothesis testing, an acceptance region is a set of sample values for which the null hypothesis cannot be rejected or can be accepted. It is the complement of the rejection region.

accrual period or **recruitment period** or **accrual**. The accrual period (or recruitment period) is the period during which subjects are being enrolled (recruited) into a study. Also see *follow-up period*.

actual alpha, **actual significance level**. This is an attained or observed significance level.

actual confidence-interval width. This is the CI width that is computed using the rounded-up sample size when the population standard deviation is known.

actual power. This is power corresponding to the actual sample size.

actual probability of confidence-interval width. `ciwidth` will calculate the required sample size for a specified probability of CI width, and if it is fractional, will round it up to report an integer. The actual probability of CI width is calculated using the rounded sample-size estimates.

actual sample size. For a two-sample study, when specifying one of the sample sizes and a sample-size ratio that result in noninteger sample sizes, `power` and `ciwidth` will round down the noninteger sample sizes to the nearest integers and use these integers for computations. The actual sample size is the rounded-down sample size.

actual sample-size ratio. When specifying a sample-size ratio that results in noninteger sample sizes, `power` and `ciwidth` will round down the input sample sizes and round up the computed sample sizes to the nearest integers. The actual sample-size ratio is computed using the rounded sample sizes.

administrative censoring. Administrative censoring is the right-censoring that occurs when the study observation period ends. All subjects complete the course of the study and are known to have experienced one of two outcomes at the end of the study: survival or failure. This type of censoring should not be confused with withdrawal and loss to follow-up. Also see *censored, uncensored, left-censored, and right-censored*.

allocation ratio. This ratio n_2/n_1 represents the number of subjects in the comparison, experimental group relative to the number of subjects in the reference, control group. Also see [PSS-4] **Unbalanced designs**.

alpha. Alpha, α, denotes the significance level.

alternative hypothesis. In hypothesis testing, the alternative hypothesis represents the counterpoint to which the null hypothesis is compared. When the parameter being tested is a scalar, the alternative hypothesis can be either one sided or two sided.

alternative value, **alternative parameter**. This value of the parameter of interest under the alternative hypothesis is fixed by the investigator in a power and sample-size analysis. For example, alternative mean value and alternative mean refer to a value of the mean parameter under the alternative hypothesis.

analysis of variance, **ANOVA**. This is a class of statistical models that studies differences between means from multiple populations by partitioning the variance of the continuous outcome into independent sources of variation due to effects of interest and random variation. The test statistic is then formed as a ratio of the expected variation due to the effects of interest to the expected random variation. Also see *one-way ANOVA*, *two-way ANOVA*, *one-way repeated-measures ANOVA*, and *two-way repeated-measures ANOVA*.

balanced design. A balanced design represents an experiment in which the numbers of treated and untreated subjects are equal. For many types of two-sample hypothesis tests, the power of the test is maximized with balanced designs. For both PrSS and PSS analyses, balanced designs tend to require fewer subjects than their corresponding unbalanced designs.

beta. Beta, β, denotes the probability of committing a type II error, namely, failing to reject the null hypothesis even though it is false.

between-subjects design. This is an experiment that has only between-subjects factors. See [PSS-2] **power oneway** and [PSS-2] **power twoway**.

between-subjects factor. This is a factor for which each subject receives only one of the levels.

binomial test. A binomial test is a test for which the exact sampling distribution of the test statistic is binomial; see [R] **bitest**. Also see [PSS-2] **power oneproportion**.

bisection method. This method finds a root x of a function $f(x)$ such that $f(x) = 0$ by repeatedly subdividing an interval on which $f(x)$ is defined until the change in successive root estimates is within the requested tolerance and function $f(\cdot)$ evaluated at the current estimate is sufficiently close to zero.

case–control study. An observational study that retrospectively compares characteristics of subjects with a certain problem (cases) with characteristics of subjects without the problem (controls). For example, to study association between smoking and lung cancer, investigators will sample subjects with and without lung cancer and record their smoking status. Case–control studies are often used to study rare diseases.

CCT. See *controlled clinical trial study*.

cell means. These are means of the outcome of interest within cells formed by the cross-classification of the two factors. See [PSS-2] **power twoway** and [PSS-2] **power repeated**.

cell-means model. A cell-means model is an ANOVA model formulated in terms of cell means.

χ^2 test. This test for which either an asymptotic sampling distribution or a sampling distribution of a test statistic is χ^2. See [PSS-2] **power onevariance** and [PSS-2] **power twoproportions**.

CI. See *confidence interval*.

CI precision. See *confidence-interval precision*.

CI precision graph. See *confidence-interval precision curve*.

CI width. See *confidence-interval width*.

clinical trial. A clinical trials is an experiment testing a medical treatment or procedure on human subjects.

clinically meaningful difference, **clinically meaningful effect**, **clinically significant difference**. Clinically meaningful difference represents the magnitude of an effect of interest that is of clinical importance. What is meant by "clinically meaningful" may vary from study to study. In clinical trials,

for example, if no prior knowledge is available about the performance of the considered clinical procedure, a standardized effect size (adjusted for standard deviation) between 0.25 and 0.5 may be considered of clinical importance.

cluster randomized design, **CRD**, **cluster randomized trial**, **CRT**, **group randomized trial**, **GRT**. Cluster randomized design is a type of randomized design in which groups of subjects or clusters are sampled instead of individual subjects. A cluster is the randomization unit, and an individual within a cluster is the analysis unit. Observations within a cluster tend to be correlated. The strength of the correlation is measured by the intraclass correlation. Also see *individual-level design*.

cluster size. The number of subjects in a group or cluster in a cluster randomized design. If cluster sizes vary between clusters, the coefficient of variation for cluster sizes is used for power and sample-size determination.

Cochran–Armitage test. The Cochran–Armitage test is a test for a linear trend in a probability of response in a $J \times 2$ contingency table. The test statistic has an asymptotic χ^2 distribution under the null hypothesis. See [PSS-2] **power trend**.

Cochran–Mantel–Haenszel test. See *Mantel–Haenszel test*.

coefficient of variation, **CV**. Coefficient of variation measures the spread or the variability of the observations relative to the mean.

cohort study. Typically an observational study, a cohort study may also be an experimental study in which a cohort, a group of subjects who have similar characteristics, is followed over time and evaluated at the end of the study. For example, cohorts of vaccinated and unvaccinated subjects are followed over time to study the effectiveness of influenza vaccines.

columns in graph. Think of `power, graph()` and `ciwidth, graph()` as graphing the columns of `power, table` and `ciwidth, table`, respectively. One of the columns will be placed on the x axis, another will be placed on the y axis, and, if you have more columns with varying values, separate plots will be created for each. Similarly, we use the terms "column symbol", "column name", and "column label" to refer to symbols, names, and labels that appear in tables when tabular output is requested.

common odds ratio. A measure of association in stratified 2×2 tables. It can be viewed as a weighted aggregate of stratum-specific odds ratios.

comparison value. See *alternative value*.

compound symmetry. A covariance matrix has a compound-symmetry structure if all the variances are equal and all the covariances are equal. This is a special case of the sphericity assumption.

concordant pairs. In a 2×2 contingency table, a concordant pair is a pair of observations that are both either successes or failures. Also see *discordant pairs* and *Introduction* under *Remarks and examples* in [PSS-2] **power pairedproportions**.

confidence bounds. See *confidence limits*.

confidence coefficient. See *confidence level*.

confidence interval. A confidence interval provides an interval estimate for a parameter of interest. It is constructed such that, in a repeated independent sampling, the proportion of confidence intervals containing the true parameter value will be larger than or equal to the specified confidence level, $1 - \alpha$. A confidence interval can also be viewed as a range of plausible values that cannot be rejected by the corresponding hypothesis test at a given significance level α. See *Confidence intervals* in [PSS-3] **Intro (ciwidth)**. Also see *one-sided confidence interval* and *two-sided confidence interval*.

confidence level. The confidence level sets the degree of certainty with which the CIs, constructed from repeated independent sampling, will be guaranteed to contain the true parameter value. For example, when specifying a confidence level of 95, the CI is guaranteed to contain the true parameter value 95% of the time. The confidence level is equal to $1 - \alpha$, where α is the significance level.

confidence limits. The confidence limits are the upper and lower limits of the confidence interval. For two-sided CIs, both confidence limits are finite. For one-sided CIs, one confidence limit is finite and the other is infinite. An upper one-sided CI has a lower confidence limit equal to negative infinity, whereas a lower one-sided CI has an upper confidence limit equal to infinity. See *Confidence intervals* in [PSS-3] **Intro (ciwidth)**.

confidence-interval half-width. The half-width of a confidence interval is equal to one half of the confidence-interval width, $w/2$, and is also known as the margin of error. The CI half-width is used as a measure of precision for a symmetric CI.

confidence-interval precision. The precision of a confidence interval is typically measured by its width. A larger width means a lower degree of precision and leads to a wider CI. A smaller width means a higher degree of precision and leads to a narrower CI.

confidence-interval precision curve. A confidence-interval precision curve is a graph of the estimated CI width as a function of some other study parameter, such as sample size or probability of CI width. The CI width is plotted on the y axis, and the sample size or other parameter is plotted on the x axis.

confidence-interval precision determination. This pertains to the computation of confidence-interval width given sample size, probability of CI width, and other study parameters.

confidence-interval width. For two-sided CIs, the width is defined as the difference between the upper and lower limits. For an upper one-sided CI, the width is the difference between the upper confidence limit and the point estimate. For a lower one-sided CI, the width is the difference between the lower confidence limit and the point estimate.

contrasts. Contrasts refers to a linear combination of cell means such that the sum of contrast coefficients is zero.

control covariates. See *reduced model*.

control group. A control group comprises subjects that are randomly assigned to a group where they receive no treatment or receives a standard treatment. In hypothesis testing, this is usually a reference group. Also see *experimental group*.

controlled clinical trial study. This is an experimental study in which treatments are assigned to two or more groups of subjects without the randomization.

CRD. See *cluster randomized design*.

critical region. See *rejection region*.

critical value. In hypothesis testing, a critical value is a boundary of the rejection region.

cross-sectional study. This type of observational study measures various population characteristics at one point in time or over a short period of time. For example, a study of the prevalence of breast cancer in the population is a cross-sectional study.

CRT. See *cluster randomized design*.

CV. See *coefficient of variation*.

delta. Delta, δ, in the context of power and sample-size calculations, denotes the effect size.

directional test. See *one-sided test*.

discordant pairs. In a 2×2 contingency table, discordant pairs are the success–failure or failure–success pairs of observations. Also see *concordant pairs* and *Introduction* under *Remarks and examples* in [PSS-2] **power pairedproportions**.

discordant proportion. This is a proportion of discordant pairs or discordant sets. Also see *Introduction* under *Remarks and examples* in [PSS-2] **power pairedproportions** as well as *Introduction* under *Remarks and examples* in [PSS-2] **power mcc**.

discordant sets. In a matched study with multiple controls matched to a case, discordant sets are the sets in which there is any success–failure or failure–success match between the case and any matched control. Also see *Introduction* under *Remarks and examples* in [PSS-2] **power mcc**.

dropout. Dropout is the withdrawal of subjects before the end of a study and leads to incomplete or missing data.

effect size. The effect size is the size of the clinically significant difference between the treatments being compared, typically expressed as a quantity that is independent of the unit of measure. For example, in a one-sample mean test, the effect size is a standardized difference between the mean and its reference value. In other cases, the effect size may be measured as an odds ratio or a risk ratio. See [PSS-2] **Intro (power)** to learn more about the relationship between effect size and the power of a test.

effect-size curve. The effect-size curve is a graph of the estimated effect size or target parameter as a function of some other study parameter such as the sample size. The effect size or target parameter is plotted on the y axis, and the sample size or other parameter is plotted on the x axis.

effect-size determination. This pertains to the computation of an effect size or a target parameter given power, sample size, and other study parameters.

equal-allocation design. See *balanced design*.

exact test. An exact test is one for which the probability of observing the data under the null hypothesis is calculated directly, often by enumeration. Exact tests do not rely on any asymptotic approximations and are therefore widely used with small datasets. See [PSS-2] **power oneproportion** and [PSS-2] **power twoproportions**.

experimental group. An experimental group is a group of subjects that receives a treatment or procedure of interest defined in a controlled experiment. In hypothesis testing, this is usually a comparison group. Also see *control group*.

experimental study. In an experimental study, as opposed to an observational study, the assignment of subjects to treatments is controlled by investigators. For example, a study that compares a new treatment with a standard treatment by assigning each treatment to a group of subjects is an experimental study.

exponential test. The exponential test is the parametric test comparing the hazard rates, λ_1 and λ_2, (or log hazards) from two independent exponential (constant only) regression models with the null hypothesis H_0: $\lambda_2 - \lambda_1 = 0$ [or H_0: $\ln(\lambda_2) - \ln(\lambda_1) = \ln(\lambda_2/\lambda_1) = 0$].

exposure odds ratio. An odds ratio of exposure in cases relative to controls in a case–control study.

F test. An F test is a test for which a sampling distribution of a test statistic is an F distribution. See [PSS-2] **power twovariances**.

factor, **factor variables**. This is a categorical explanatory variable with any number of levels.

finite population correction. When sampling is performed without replacement from a finite population, a finite population correction is applied to the standard error of the estimator to reduce sampling variance.

Fisher–Irwin's exact test. See *Fisher's exact test*.

Fisher's exact test. Fisher's exact test is an exact small-sample test of independence between rows and columns in a 2×2 contingency table. Conditional on the marginal totals, the test statistic has a hypergeometric distribution under the null hypothesis. See [PSS-2] **power twoproportions** and [R] **tabulate twoway**.

Fisher's z test. This is a z test comparing one or two correlations. See [PSS-2] **power onecorrelation** and [PSS-2] **power twocorrelations**. Also see *Fisher's z transformation*.

Fisher's z transformation. Fisher's z transformation applies an inverse hyperbolic tangent transformation to the sample correlation coefficient. This transformation is useful for testing hypothesis concerning Pearson's correlation coefficient. The exact sampling distribution of the correlation coefficient is complicated, while the transformed statistic is approximately standard normal.

fixed effects. Fixed effects represent all levels of the factor that are of interest.

follow-up period or **follow-up**. The (minimum) follow-up period is the period after the last subject entered the study until the end of the study. The follow-up defines the phase of a study during which subjects are under observation and no new subjects enter the study. If T is the total duration of a study, and r is the accrual period of the study, then follow-up period f is equal to $T - r$. Also see *accrual period*.

follow-up study. See *cohort study*.

fractional sample size. Fractional (noninteger) sample sizes occur when specifying an odd number for the total sample size in studies with an equal-allocation design. They may also occur when specifying noninteger sample-size ratios.

full model. In the regression context, a full model is a regression model that includes all covariates of interest. Also see *reduced model*.

Greenhouse–Geisser correction. See *nonsphericity correction*.

group randomized trial, **GRT**. See *cluster randomized design*.

hypothesis. A hypothesis is a statement about a population parameter of interest.

hypothesis testing, **hypothesis test**. This method of inference evaluates the validity of a hypothesis based on a sample from the population. See *Hypothesis testing* under *Remarks and examples* in [PSS-2] **Intro (power)**.

hypothesized value. See *null value*.

individual-level design. Individual-level design is a classical randomized design in which individual subjects or observations are sampled; thus they represent both units of randomization and units of analysis. In contrast, see *cluster randomized design*.

intercept. The intercept of a regression model is the constant term, which is reported as `_cons` by estimation commands that fit a regression model.

interaction effects. Interaction effects measure the dependence of the effects of one factor on the levels of the other factor. Mathematically, they can be defined as the differences among treatment means that are left after main effects are removed from these differences.

intraclass correlation. Intraclass correlation measures the dependence of observations in the same group or cluster.

J $\times$ 2 contingency table. A $J \times 2$ contingency table is used to describe the association between an ordinal independent variable with J levels and a binary response variable of interest.

Lagrange multiplier test. See *score test*.

likelihood-ratio test. The likelihood-ratio (LR) test is one of the three classical testing procedures used to compare the fit of two models, one of which, the constrained model, is nested within the full (unconstrained) model. Under the null hypothesis, the constrained model fits the data as well as the full model. The LR test requires one to determine the maximal value of the log-likelihood function for both the constrained and the full models. See [PSS-2] **power twoproportions** and [R] **lrtest**.

loss to follow-up. Subjects are lost to follow-up if they do not complete the course of the study for reasons unrelated to the event of interest. For example, loss to follow-up occurs if subjects move to a different area or decide to no longer participate in a study. Loss to follow-up should not be confused with administrative censoring. If subjects are lost to follow-up, the information about the outcome these subjects would have experienced at the end of the study, had they completed the study, is unavailable. Also see *withdrawal*, *administrative censoring*, and *follow-up period or follow-up*.

lower one-sided confidence interval. A lower one-sided confidence interval contains a range of values that are greater than or equal to the lower confidence limit ll. The confidence interval is defined by a finite lower confidence limit and an upper confidence limit of infinity: $[ll, \infty)$.

lower one-sided test, **lower one-tailed test**. A lower one-sided test is a one-sided test of a scalar parameter in which the alternative hypothesis is lower one sided, meaning that the alternative hypothesis states that the parameter is less than the value conjectured under the null hypothesis. Also see *One-sided test versus two-sided test* under *Remarks and examples* in [PSS-2] **Intro (power)**.

main effects. These are average, additive effects that are associated with each level of each factor. For example, the main effect of level j of a factor is the difference between the mean of all observations on the outcome of interest at level j and the grand mean.

Mantel–Haenszel test. The Mantel–Haenszel test evaluates whether the overall degree of association in stratified 2×2 tables is significant assuming that the exposure effect is the same across strata. See [PSS-2] **power cmh**.

margin of error. See *confidence-interval half-width*.

marginal homogeneity. Marginal homogeneity refers to the equality of one or more row marginal proportions with the corresponding column proportions. Also see *Introduction* under *Remarks and examples* in [PSS-2] **power pairedproportions**.

marginal proportion. This represents a ratio of the number of observations in a row or column of a contingency table relative to the total number of observations. Also see *Introduction* under *Remarks and examples* in [PSS-2] **power pairedproportions**.

matched study. In a matched study, an observation from one group is matched to one or more observations from another group with respect to one or more characteristics of interest. When multiple matches occur, the study design is $1 : M$, where M is the number of matches. Also see *paired data*, also known as $1 : 1$ matched data.

McNemar's test. McNemar's test is a test used to compare two dependent binary populations. The null hypothesis is formulated in the context of a 2×2 contingency table as a hypothesis of marginal homogeneity. See [PSS-2] **power pairedproportions** and the `mcc` command in [R] **Epitab**.

MDES. See *minimum detectable effect size*.

mean contrasts. See *contrasts*.

minimum detectable effect size. The minimum detectable effect size is the smallest effect size that can be detected by hypothesis testing for a given power and sample size.

minimum detectable value. The minimum detectable value represents the smallest amount or concentration of a substance that can be reliably measured.

mixed design. A mixed design is an experiment that has at least one between-subjects factor and one within-subject factor. See [PSS-2] **power repeated**.

multiple partial correlation. In the regression context, multiple partial correlation is the measure of association between the dependent variable and one or more independent variables of interest, while controlling for the effect of other variables in the model.

negative effect size. In power and sample-size analysis, we obtain a negative effect size when the postulated value of the parameter under the alternative hypothesis is less than the hypothesized value of the parameter under the null hypothesis. Also see *positive effect size*.

nominal alpha, **nominal significance level**. This is a desired or requested significance level.

noncentrality parameter. In power and sample-size analysis, a noncentrality parameter is the expected value of the test statistic under the alternative hypothesis.

nondirectional test. See *two-sided test*.

nonsphericity correction. This is a correction used for the degrees of freedom of a regular F test in a repeated-measures ANOVA to compensate for the lack of sphericity of the repeated-measures covariance matrix.

nuisance covariate, **nuisance variable**. A nuisance covariate is a variable that is important to include in a statistical model but is of little interest to the researcher. Model parameters describing the effects of nuisance covariates are considered nuisance parameters.

nuisance parameter. A nuisance parameter is a quantity that is estimated when fitting a statistical model but is of little interest to the researcher. For example, if you are intersted in testing the coefficient of one predictor in a linear regression model that has two predictor variables, you will consider the intercept and the coefficient of the nuisance covariate to be nuisance parameters.

null hypothesis. In hypothesis testing, the null hypothesis typically represents the conjecture that one is attempting to disprove. Often the null hypothesis is that a treatment has no effect or that a statistic is equal across populations.

null value, **null parameter**. This value of the parameter of interest under the null hypothesis is fixed by the investigator in a power and sample-size analysis. For example, null mean value and null mean refer to the value of the mean parameter under the null hypothesis.

number of clusters. The number of independent sampling units, groups or clusters, in a cluster randomized design.

observational study. In an observational study, as opposed to an experimental study, the assignment of subjects to treatments happens naturally and is thus beyond the control of investigators. Investigators can only observe subjects and measure their characteristics. For example, a study that evaluates the effect of exposure of children to household pesticides is an observational study.

observed level of significance. See *p-value*.

odds and odds ratio. The odds in favor of an event are $\text{Odds} = p/(1-p)$, where p is the probability of the event. Thus if $p = 0.2$, the odds are 0.25, and if $p = 0.8$, the odds are 4.

The log of the odds is $\ln(\text{Odds}) = \text{logit}(p) = \ln\{p/(1-p)\}$, and logistic regression models, for instance, fit $\ln(\text{Odds})$ as a linear function of the covariates.

The odds ratio is a ratio of two odds: $\text{Odds}_2/\text{Odds}_1$. The individual odds that appear in the ratio are usually for an experimental group and a control group or for two different demographic groups.

one-sample test. A one-sample test compares a parameter of interest from one sample with a reference value. For example, a one-sample mean test compares a mean of the sample with a reference value.

one-sided confidence interval. See *upper one-sided confidence interval* and *lower one-sided confidence interval*.

one-sided test, **one-tailed test**. A one-sided test is a hypothesis test of a scalar parameter in which the alternative hypothesis is one sided, meaning that the alternative hypothesis states that the parameter is either less than or greater than the value conjectured under the null hypothesis but not both. Also see *One-sided test versus two-sided test* under *Remarks and examples* in [PSS-2] **Intro (power)**.

one-way ANOVA, **one-way analysis of variance**. A one-way ANOVA model has a single factor. Also see [PSS-2] **power oneway**.

one-way repeated-measures ANOVA. A one-way repeated-measures ANOVA model has a single within-subject factor. Also see [PSS-2] **power repeated**.

paired data. Paired data consist of pairs of observations that share some characteristics of interest. For example, measurements on twins, pretest and posttest measurements, before and after measurements, repeated measurements on the same individual. Paired data are correlated and thus must be analyzed by using a paired test. See [PSS-3] **ciwidth pairedmeans** for PrSS analysis for a paired-means-difference CI.

paired observations. See *paired data*.

paired test. A paired test is used to test whether the parameters of interest of two paired populations are equal. The test takes into account the dependence between measurements. For this reason, paired tests are usually more powerful than their two-sample counterparts. For example, a paired-means or paired-difference test is used to test whether the means of two paired (correlated) populations are equal.

partial correlation. Partial correlation is the measure of association between two continuous variables, while controlling for the effect of other variables.

Pearson's correlation. Pearson's correlation ρ, also known as the product-moment correlation, measures the degree of association between two variables. Pearson's correlation equals the variables' covariance divided by their respective standard deviations, and ranges between -1 and 1. Zero indicates no correlation between the two variables.

population parameter. See *target parameter*.

positive effect size. In power and sample-size analysis, we obtain a positive effect size when the postulated value of the parameter under the alternative hypothesis is greater than the hypothesized value of the parameter under the null hypothesis. Also see *negative effect size*.

postulated value. See *alternative value*.

power. The power of a test is the probability of correctly rejecting the null hypothesis when it is false. It is often denoted as $1-\beta$ in the statistical literature, where β is the type-II-error probability. Commonly used values for power are 80% and 90%. See [PSS-2] **Intro (power)** for more details about power.

power and sample-size analysis. Power and sample-size analysis investigates the optimal allocation of study resources to increase the likelihood of the successful achievement of a study objective. The focus of power and sample-size analysis is on studies that use hypothesis testing for inference. Power and sample-size analysis provides an estimate of the sample size required to achieve the desired power of a test in a future study. See [PSS-2] **Intro (power)**. Also see *precision and sample-size analysis*.

power curve. A power curve is a graph of the estimated power as a function of some other study parameter such as the sample size. The power is plotted on the y axis, and the sample size or other parameter is plotted on the x axis. See [PSS-2] **power, graph**.

power determination. This pertains to the computation of a power given sample size, effect size, and other study parameters.

power function. The power functions is a function of the population parameter θ, defined as the probability that the observed sample belongs to the rejection region of a test for given θ. See *Hypothesis testing* under *Remarks and examples* in [PSS-2] **Intro (power)**.

power graph. See *power curve*.

precision and sample-size analysis. Just like power and sample-size analysis, precision and sample-size analysis investigates the optimal allocation of study resources to increase the likelihood of the successful achievement of a study objective. The focus of precision and sample-size analysis is on studies that use confidence intervals for inference. Precision and sample-size analysis provides an estimate of the sample size required to achieve the desired precision of a confidence interval in a future study. See [PSS-3] **Intro (ciwidth)**.

precision of a confidence interval. See *confidence-interval precision*.

probability of a type I error. This is the probability of committing a type I error of incorrectly rejecting the null hypothesis. Also see *significance level*.

probability of a type II error. This is the probability of committing a type II error of incorrectly accepting the null hypothesis. Common values for the probability of a type II error are 0.1 and 0.2 or, equivalently, 10% and 20%. Also see *beta* and *power*.

probability of confidence-interval width. The probability of CI width is the probability that the width of a CI in a future study will be no greater than a prespecified value.

probability of confidence-interval width determination. This pertains to the computation of the probability of CI width given CI width, sample size, and other study parameters.

prospective study. In a prospective study, the population or cohort is classified according to specific risk factors, such that the outcome of interest, typically various manifestations of a disease, can be observed over time and tied in to the initial classification. Also see *retrospective study*.

PrSS analysis. See *precision and sample-size analysis*.

PSS analysis. See *power and sample-size analysis*.

PSS Control Panel. The PSS Control Panel is a point-and-click graphical user interface for power and sample-size analysis. See [PSS-2] **GUI (power)**.

p-value. P-value is a probability of obtaining a test statistic as extreme or more extreme as the one observed in a sample assuming the null hypothesis is true.

R^2. See *coefficient of determination*.

random effects. Random effects represent a random sample of levels from all possible levels, and the interest lies in all possible levels.

randomized controlled trial. In this experimental study, treatments are randomly assigned to two or more groups of subjects.

RCT. See *randomized controlled trial*.

recruitment period. See *accrual period*.

reduced model. In the regression context, a reduced model is a regression model that contains only a subset of covariates from the corresponding full model. These covariates are referred to as "control covariates". The covariates that are not in the reduced model are referred to as "tested covariates".

reference value. See *null value*.

rejection region. In hypothesis testing, a rejection region is a set of sample values for which the null hypothesis can be rejected.

relative risk. See *risk ratio*.

retrospective study. In a retrospective study, a group with a disease of interest is compared with a group without the disease, and information is gathered in a retrospective way about the exposure in each group to various risk factors that might be associated with the disease. Also see *prospective study*.

risk difference. A risk difference is defined as the probability of an event occurring when a risk factor is increased by one unit minus the probability of the event occurring without the increase in the risk factor.

When the risk factor is binary, the risk difference is the probability of the outcome when the risk factor is present minus the probability when the risk factor is not present.

When one compares two populations, a risk difference is defined as a difference between the probabilities of an event in the two groups. It is typically a difference between the probability in the comparison group or experimental group and the probability in the reference group or control group.

risk factor. A risk factor is a variable that is associated with an increased or decreased probability of an outcome.

risk ratio. A risk ratio, also called a relative risk, measures the increase in the likelihood of an event occurring when a risk factor is increased by one unit. It is the ratio of the probability of the event when the risk factor is increased by one unit over the probability without that increase.

When the risk factor is binary, the risk ratio is the ratio of the probability of the event when the risk factor occurs over the probability when the risk factor does not occur.

When one compares two populations, a risk ratio is defined as a ratio of the probabilities of an event in the two groups. It is typically a ratio of the probability in the comparison group or experimental group to the probability in the reference group or control group.

sample size. This is the number of subjects in a sample. See [PSS-2] **Intro (power)** to learn more about the relationship between sample size and the power of a test.

sample-size curve. A sample-size curve is a graph of the estimated sample size as a function of some other study parameter such as power or CI width. The sample size is plotted on the y axis, and the power or other parameter is plotted on the x axis.

sample-size determination. This pertains to the computation of a sample size given either power and effect size or CI width and probability of CI width and any other study parameters. In a cluster randomized design, sample-size determination consists of determining the number of clusters given the cluster size or the cluster size given the number of clusters.

sample-size ratio. The ratio of the experimental-group sample size relative to the control-group sample size, n_2/n_1.

Satterthwaite's t test. Satterthwaite's t test is a modification of the two-sample t test to account for unequal variances in the two populations. See *Methods and formulas* in [PSS-2] **power twomeans** for details.

score test. A score test, also known as a Lagrange multiplier test, is one of the three classical testing procedures used to compare the fit of two models, one of which, the constrained model, is nested within the full (unconstrained) model. The null hypothesis is that the constrained model fits the data as well as the full model. The score test only requires one to fit the constrained model. See [PSS-2] **power oneproportion** and [R] **prtest**.

sensitivity analysis. Sensitivity analysis investigates the effect of varying study parameters on power, CI precision, probability of CI width, sample size, and other components of a study. The true values of study parameters are usually unknown, and analyses of power, precision, and sample size use best guesses for these values. It is therefore important to evaluate the sensitivity of the computed power, CI precision, or sample size in response to changes in study parameters. See [PSS-2] **power, table**, [PSS-2] **power, graph**, [PSS-3] **ciwidth, table**, and [PSS-3] **ciwidth, graph** for details.

sign test. A sign test is used to test the null hypothesis that the median of a distribution is equal to some reference value. A sign test is carried out as a test of binomial proportion with a reference value of 0.5. See [PSS-2] **power oneproportion** and [R] **bitest**.

significance level. In hypothesis testing, the significance level α is an upper bound for a probability of a type I error. See [PSS-2] **Intro (power)** to learn more about the relationship between significance level and the power of a test.

size of test. See *significance level*.

sphericity assumption. All differences between levels of the within-subject factor within-subject factor have the same variances.

stratified 2 × 2 tables. Stratified 2×2 tables describe the association between a binary independent variable and a binary response variable of interest. The analysis is stratified by a nominal (categorical) variable with K levels.

symmetry. In a 2×2 contingency table, symmetry refers to the equality of the off-diagonal elements. For a 2×2 table, a test of marginal homogeneity reduces to a test of symmetry.

t test. A t test is a test for which the sampling distribution of the test statistic is a Student's t distribution.

A one-sample t test is used to test whether the mean of a population is equal to a specified value when the variance must also be estimated. The test statistic follows Student's t distribution with $N - 1$ degrees of freedom, where N is the sample size.

A two-sample t test is used to test whether the means of two populations are equal when the variances of the populations must also be estimated. When the two populations' variances are unequal, a modification to the standard two-sample t test is used; see *Satterthwaite's t test*.

target parameter. In power and sample-size analysis, the target parameter is the parameter of interest or the parameter in the study about which hypothesis tests are conducted.

test statistic. In hypothesis testing, a test statistic is a function of the sample that does not depend on any unknown parameters.

tested covariates. See *reduced model*.

two-independent-samples test. See *two-sample test*.

two-sample paired test. See *paired test*.

two-sample test. A two-sample test is used to test whether the parameters of interest of the two independent populations are equal. For example, two-sample means test, two-sample variances, two-sample proportions test, two-sample correlations test.

two-sided confidence interval. A two-sided CI contains a plausible finite range of values for a parameter of interest. Two-sided CIs contain a finite upper limit for plausible values greater than the point estimate and a finite lower limit for plausible values less than the point estimate. See *Confidence intervals* in [PSS-3] **Intro (ciwidth)**.

two-sided test, **two-tailed test**. A two-sided test is a hypothesis test of a parameter in which the alternative hypothesis is the complement of the null hypothesis. In the context of a test of a scalar parameter, the alternative hypothesis states that the parameter is less than or greater than the value conjectured under the null hypothesis.

two-way ANOVA, **two-way analysis of variance**. A two-way ANOVA model contains two factors. Also see [PSS-2] **power twoway**.

two-way repeated-measures ANOVA, **two-factor** ANOVA. This is a repeated-measures ANOVA model with one within-subject factor and one between-subjects factor. The model can be additive (contain only main effects of the factors) or can contain main effects and an interaction between the two factors. Also see [PSS-2] **power repeated**.

type I error. The type I error of a test is the error of rejecting the null hypothesis when it is true; see [PSS-2] **Intro (power)** for more details.

type-I-error probability. See *probability of a type I error*.

type I study. A type I study is a study in which all subjects fail (or experience an event) by the end of the study; that is, no censoring of subjects occurs.

type II error. The type II error of a test is the error of not rejecting the null hypothesis when it is false; see [PSS-2] **Intro (power)** for more details.

type-II-error probability. See *probability of a type II error*.

type II study. A type II study is a study in which there are subjects who do not fail (or do not experience an event) by the end of the study. These subjects are known to be censored.

unbalanced design. An unbalanced design indicates an experiment in which the numbers of treated and untreated subjects differ. Also see [PSS-4] **Unbalanced designs**.

unequal-allocation design. See *unbalanced design*.

upper one-sided confidence interval. An upper one-sided confidence interval contains a range of values that are less than or equal to the upper confidence limit ul. The confidence interval is defined by a finite upper confidence limit and a lower confidence limit of negative infinity: $(-\infty, ul]$.

upper one-sided test, **upper one-tailed test**. An upper one-sided test is a one-sided test of a scalar parameter in which the alternative hypothesis is upper one sided, meaning that the alternative hypothesis states that the parameter is greater than the value conjectured under the null hypothesis. Also see *One-sided test versus two-sided test* under *Remarks and examples* in [PSS-2] **Intro (power)**.

Wald test. A Wald test is one of the three classical testing procedures used to compare the fit of two models, one of which, the constrained model, is nested within the full (unconstrained) model. Under the null hypothesis, the constrained model fits the data as well as the full model. The Wald test requires one to fit the full model but does not require one to fit the constrained model. Also see [PSS-2] **power oneproportion** and [R] **test**.

withdrawal. Withdrawal is the process under which subjects withdraw from a study for reasons unrelated to the event of interest. For example, withdrawal occurs if subjects move to a different area or decide to no longer participate in a study. Withdrawal should not be confused with administrative censoring. If subjects withdraw from the study, the information about the outcome those subjects would have experienced at the end of the study, had they completed the study, is unavailable. Also see *loss to follow-up* and *administrative censoring*.

within-subject design. This is an experiment that has at least one within-subject factor. See [PSS-2] **power repeated**.

within-subject factor. This is a factor for which each subject receives several of or all the levels.

z test. A z test is a test for which a potentially asymptotic sampling distribution of the test statistic is a normal distribution. For example, a one-sample z test of means is used to test whether the mean of a population is equal to a specified value when the variance is assumed to be known. The distribution of its test statistic is normal. See [PSS-2] **power onemean**, [PSS-2] **power twomeans**, and [PSS-2] **power pairedmeans**.

Subject and author index

See the combined subject index and the combined author index in the *Stata Index*.