Title

stata.com

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
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

stata.com

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 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 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.

◁

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 7: 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 8: 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 9: 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 10: 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 11: 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 12: 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

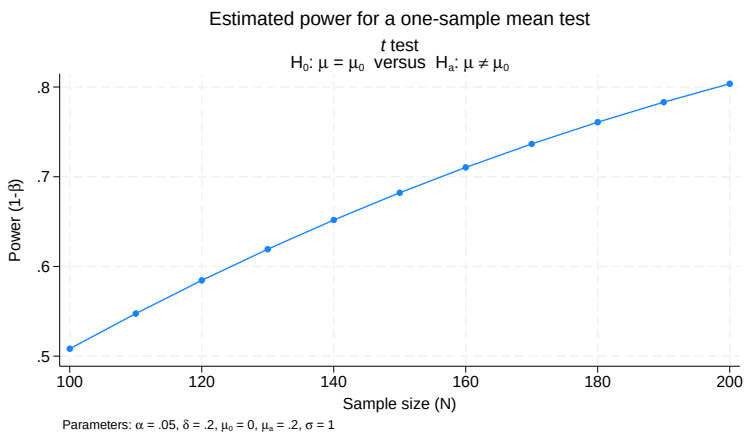
During the planning stage of a study, it is difficult to decide on a number of subjects to be enrolled in a study on the basis of only one set of study parameters. It is common to investigate the effect of various study scenarios on power. Power curves, or plots of estimated power versus one of the study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

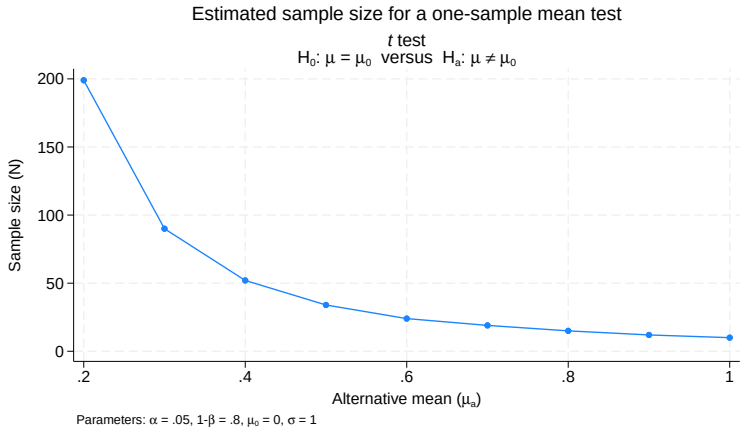
For example, we compute power and plot it as a function of sample size for a range of sample-size values between 100 and 200 with a step size of 10:

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



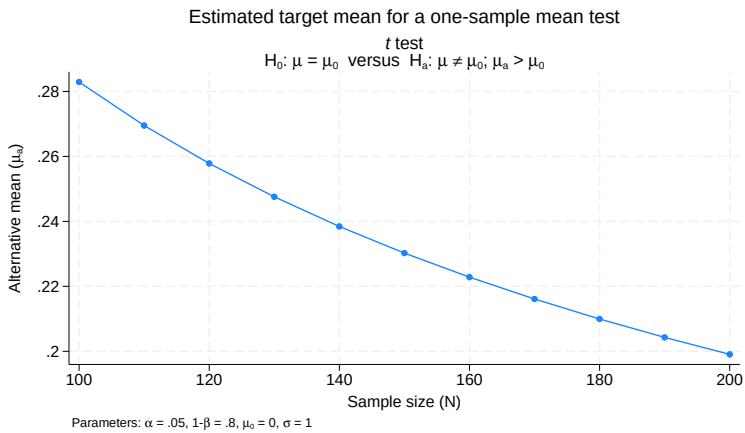
Or we can compute sample size and plot it as a function of the alternative mean when the mean ranges between 0.2 and 1 with a step size of 0.1:

```
. 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:

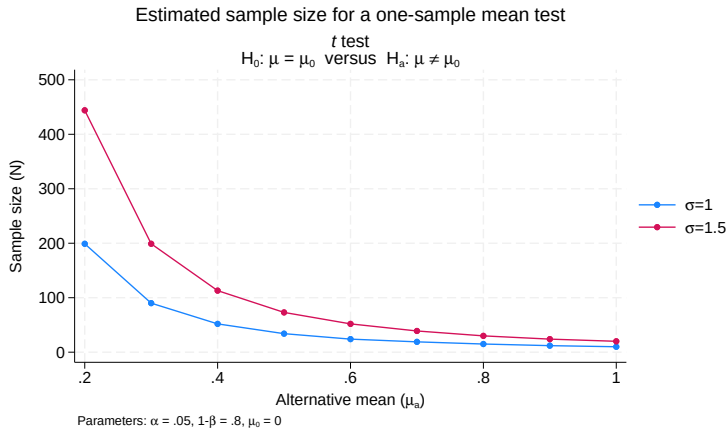
```
. power onemean 0, n(100(10)200) power(0.8) graph
```



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 18.0             // (or version 18.5 for 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
```

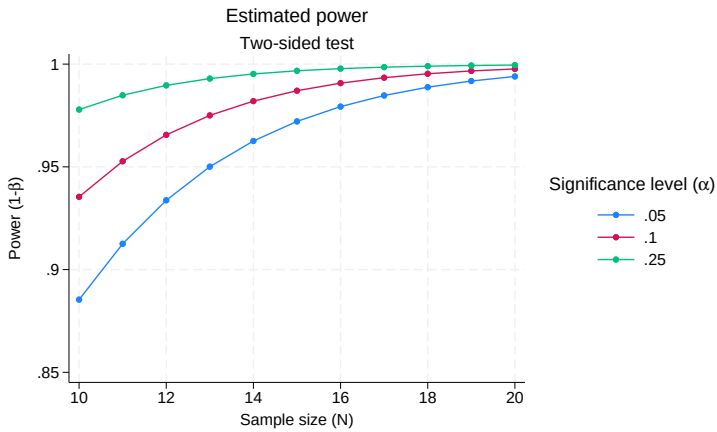
alpha	power	N
.05	.8854	10
.05	.9721	15
.05	.994	20
.05	.9988	25

`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.

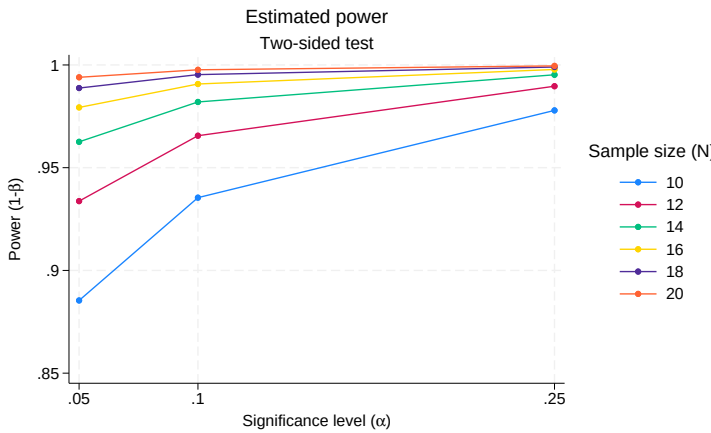
We can supply both sample sizes and significance levels and request a graph instead of a table:

```
. 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.

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