

Clinical Trial Design Using Stata

Alex Asher

Senior Biostatistician and Software Developer
StataCorp LLC

2026 Stata Conference
Boston, MA

Outline

- Power and Sample Size Concepts
 - Null and alternative hypotheses
 - Critical values, alpha levels, power, effect size
- Power and Sample Size Using Stata
 - Built-in `power` methods
 - Power by simulation
 - Sample size by simulation
 - Leveraging AI tools
- Group Sequential Designs
 - Introduction to GSDs
 - Simulation-based GSDs
- Stepped Wedge Designs
 - Stepped Wedge vs DID
 - Example: education intervention

Research Question



image: Journal of Pharmacology & Clinical Research, licensed under CC BY 4.0

- Does our novel chemotherapy shrink tumors?
- Outcome: Tumor Shrinkage Rate (TSR)

$$\text{TSR} = (D_b - D_a) / (D_b \times t) \times 100\%$$

D_b is the longest diameter of the tumor before treatment

D_a is the longest diameter after treatment

t is the time elapsed in days

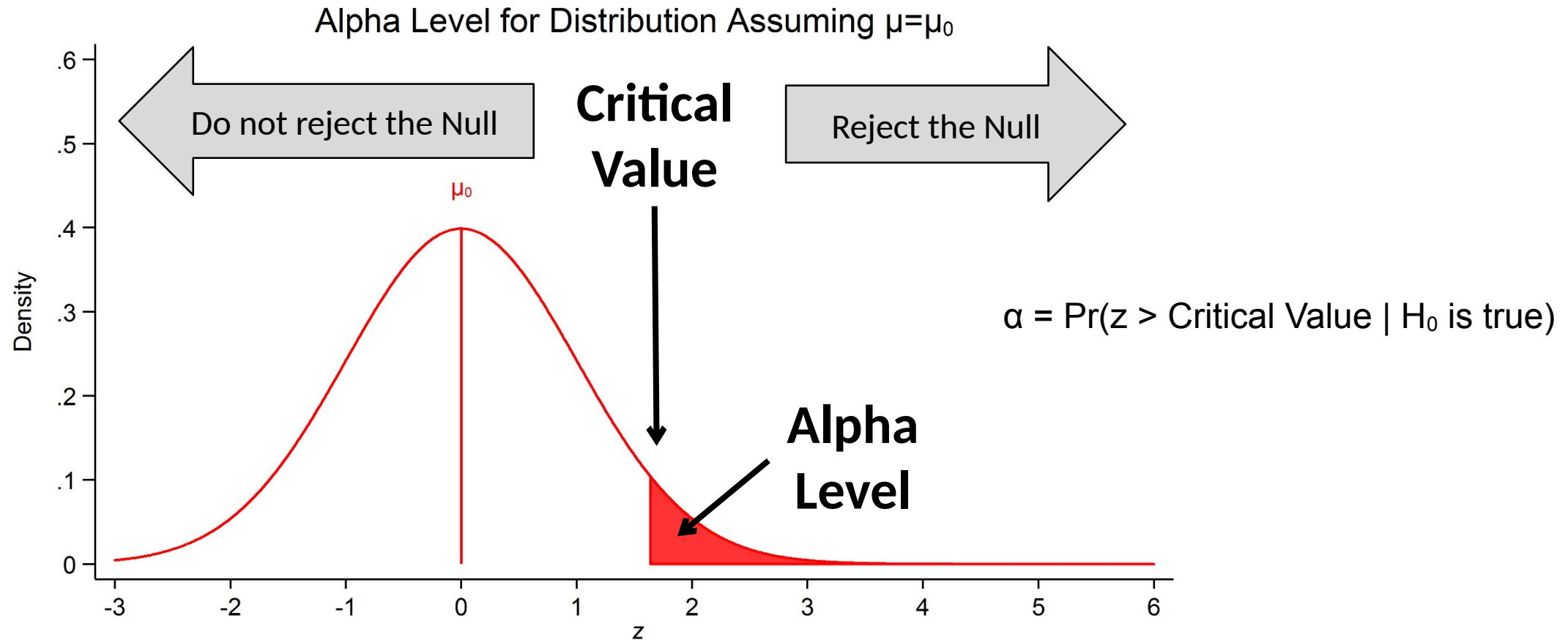
Null and alternative hypotheses

- $H_0: \text{TSR} = 0, \quad H_a: \text{TSR} > 0$
- Conduct a z-test with known SD

- Test statistic
$$Z = \frac{\bar{X} - \mu_0}{\sigma / \sqrt{n}}$$

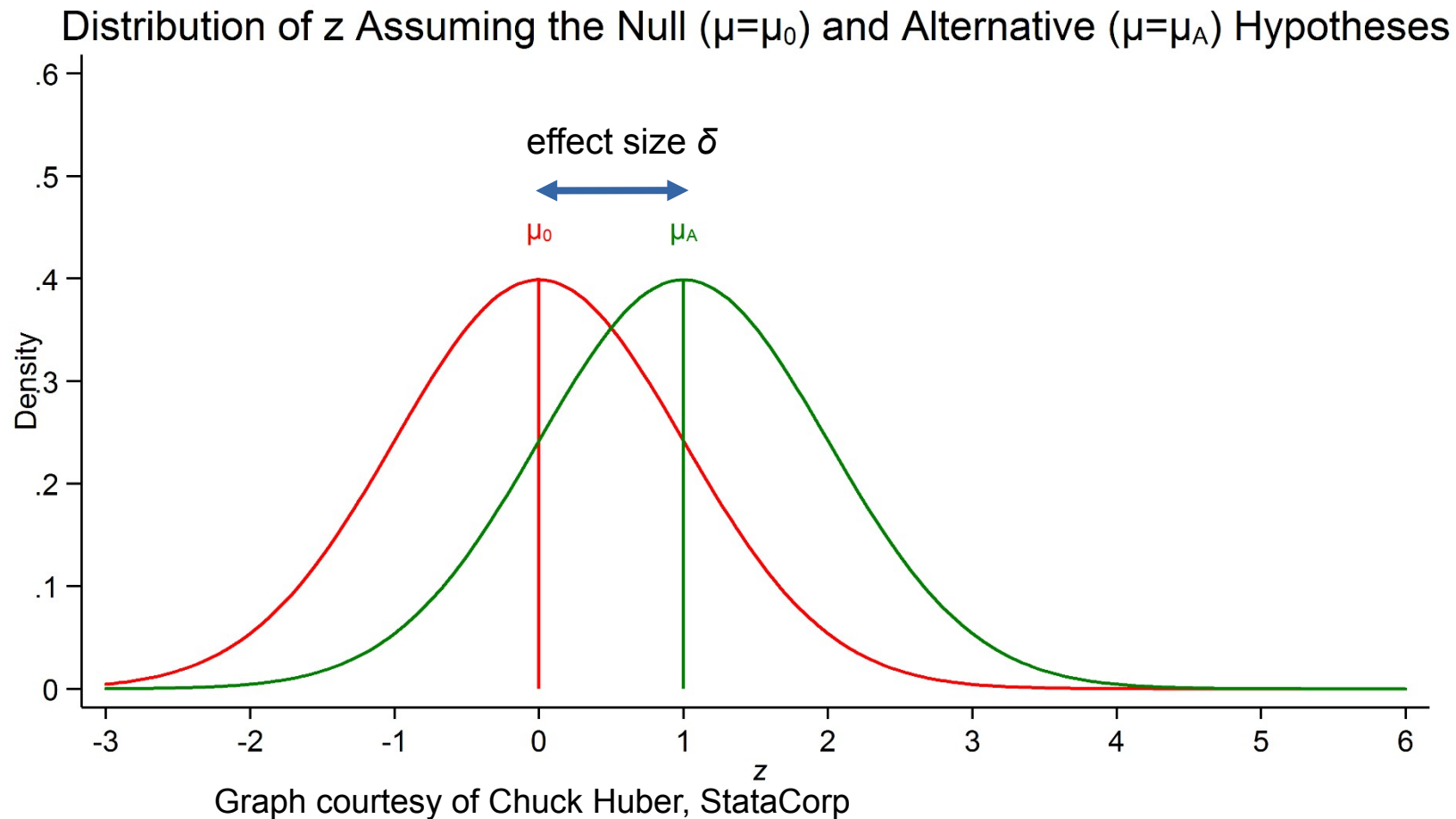
- x_i = observed TSR for participant i
 μ_0 = TSR under H_0
 σ = standard deviation (known)
 n = sample size

Critical Value and Alpha Level

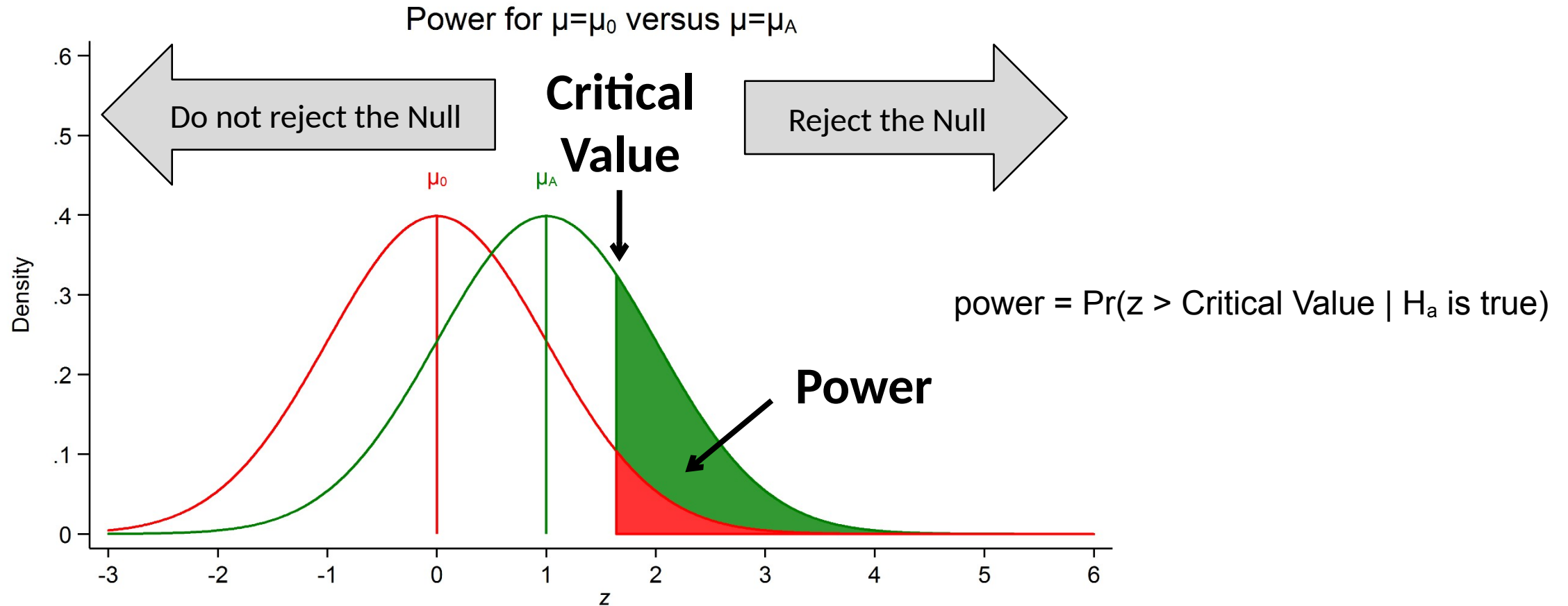


Graph courtesy of Chuck Huber, StataCorp

Distribution Under the Alternative



Critical Value and Power



Alpha Level and Power

- Alpha Level:
 - The probability of rejecting the null hypothesis assuming that the null hypothesis is true.
- Statistical Power:
 - The probability of rejecting the null hypothesis assuming that the alternative hypothesis is true.

Solve for Power

- User must specify:
 - Alpha
 - One-sided or two-sided
 - Often 0.05 or 0.01 for two-sided tests
 - We will conduct an upper one-sided test at the 0.025 level
 - Sample size
 - Often dictated by funding
 - We will use $n=50$
 - Effect size
 - Difference between hypothesized parameter values
 - Can be scaled or unscaled (specify SD if unscaled)
 - We will use $\delta=0.5$ and $\sigma=1$

Power calculation in Stata

The screenshot displays the StataNow/MP 18.5 software interface. On the left, the 'Statistics' menu is open, showing a list of statistical analysis categories. A large blue arrow points from the 'Power, precision, and sample size' option in this menu to a secondary window on the right. This window, titled 'Power, precision, and sample-size analysis', contains a list of methods organized by category. A second blue arrow points from the 'Population parameter' category to the 'Test comparing one mean to a reference value' method.

StataNow/MP 18.5

File Edit Data Graphics Statistics User Window Help

Summaries, tables, and tests

Linear models and related

Binary outcomes

Ordinal outcomes

Categorical outcomes

Count outcomes

Fractional outcomes

Generalized linear models

Choice models

Time series

Multivariate time series

Spatial autoregressive models

Longitudinal/panel data

Multilevel mixed-effects models

Survival analysis

Epidemiology and related

Endogenous covariates

Sample-selection models

Causal inference/treatment effects

SEM (structural equation modeling)

LCA (latent class analysis)

FMM (finite mixture models)

IRT (item response theory)

Multivariate analysis

Survey data analysis

Lasso

Meta-analysis

Multiple imputation

Nonparametric analysis

Exact statistics

Resampling

Power, precision, and sample size

Bayesian analysis

Bayesian model averaging

History

Filter commands here

Command

There are no items to

Power, precision, and sample-size analysis

Methods organized by:

Sort Filter methods here

All methods

Population parameter

- Correlations
- Hazard rates
- Means
- Odds ratio
- Proportions
- R-squared
- Regression slopes
- Standard deviations
- Survival rates
- Variances
- Outcome
- Hypothesis test
- Confidence interval
- Sample
- Group sequential design

- Fisher's z test comparing one correlation to a reference value
- Test comparing partial correlation to a reference value in a multiple linear regression
- One-way analysis of variance
- Repeated-measures analysis of variance
- Test comparing one mean to a reference value in a cluster randomized design
- Test comparing one mean to a reference value in a group sequential design
- Test comparing two independent means in a cluster randomized design
- Test comparing two independent means in a group sequential design
- Paired test comparing two correlated means, specify standard deviation of the differences
- CI for a paired-means difference, specify the standard deviation of the difference
- Fisher's z test comparing two independent correlations
- Exponential test comparing two independent hazard rates
- Two-way analysis of variance
- Test comparing one mean to a reference value
- CI for one mean
- Test comparing two independent means
- CI for a two-means difference
- Paired test comparing two correlated means, specify correlation between paired observations
- CI for a paired-means difference, specify correlation between paired observations
- Test of association in 1:M matched case-control studies

Power calculation in Stata

power onemean - Power analysis for a one-sample mean test

Main Table Graph Iteration

Compute: Power * Accepts numlist [\(Examples\)](#)

Error probabilities: 0.025 * Significance level

Sample size: 50 * Sample size

Effect size

Means: 0 * Null, 0.5 * Alternative

Standard deviation: 1 * Standard deviation, ☒ Assume a known standard deviation

* Finite population correction: None

Sides: One-sided test

☐ Treat number lists in starred(*) options as parallel

? ↺ 📄 Submit Cancel OK

```
. power onemean 0 0.5, n(50) alpha(0.025) knownsd onesided
```

Estimated power for a one-sample mean test
z test

$H_0: m = m_0$ versus $H_a: m > m_0$

Study parameters:

alpha = 0.0250
N = 50
delta = 0.5000
m0 = 0.0000
ma = 0.5000
sd = 1.0000

Estimated power:

power = 0.9424

Power calculation by simulation

```
**# program onmeansim: one sample z-test (upper one-sided test)
program onmeansim, rclass
    version 19
    syntax, n(integer)          /// Sample size
        [ alpha(real 0.025)    /// Alpha level
          m0(real 0)            /// Mean under the null
          ma(real 0.5)          /// Mean under the alternative
        ]
    // Generate a normal random variable with mean = `ma' and SD = 1
    quietly drawnorm y, n(`n') mean(`ma') sd(1) clear
    // Upper one-sided z-test
    ztest y == `m0'
    return scalar reject = (r(p_u) < `alpha')
end
```

Power calculation by simulation

```
. // Try it out  
. set seed 9  
  
. onmeansim, n(50)
```

One-sample z test

Variable	Obs	Mean	Std. err.	Std. dev.	[95% conf. interval]	
y	50	.6792907	.1414214	1	.4021099	.9564715

mean = mean(y) z = 4.8033

H0: mean = 0

Ha: mean < 0

Pr(Z < z) = 1.0000

Ha: mean != 0

Pr(|Z| > |z|) = 0.0000

Ha: mean > 0

Pr(Z > z) = 0.0000

```
. return list
```

scalars:

r(reject) = 1

Create a power *usermethod* command

```
*** program power_cmd_onmeansim: power by simulation
program power_cmd_onmeansim, rclass
    version 19
    syntax, n(integer)          /// Sample size
        [ alpha(real 0.025)    /// Alpha level
          m0(real 0)            /// Mean under the null
          ma(real 0.5)          /// Mean under the alternative
          reps(integer 100)     /// Number of replicates
          seed(numlist max=1)   /// Random seed (no default seed)
        ]

    // Set seed, if specified
    if ("`seed'" != "") set seed `seed'

    // Simulate data and test the null hypothesis
    quietly simulate reject=r(reject), reps(`reps'):    ///
        onmeansim, n(`n') m0(`m0') ma(`ma') alpha(`alpha')
    quietly summarize reject

    // Return results
    return scalar power = r(mean)
    return scalar N      = `n'
    return scalar alpha  = `alpha'
    return scalar m0     = `m0'
    return scalar ma     = `ma'
    return scalar delta  = `ma' - `m0'

end
```

Run `power onmeansim`

```
. // Start with N=50. Use 1000 reps for accuracy. Set seed for reproducibility.  
. power onmeansim, n(50) alpha(0.025) m0(0) ma(0.5) reps(1000) seed(9)
```

Estimated power

Two-sided test



alpha	power	N
.025	.938	50

MC Error

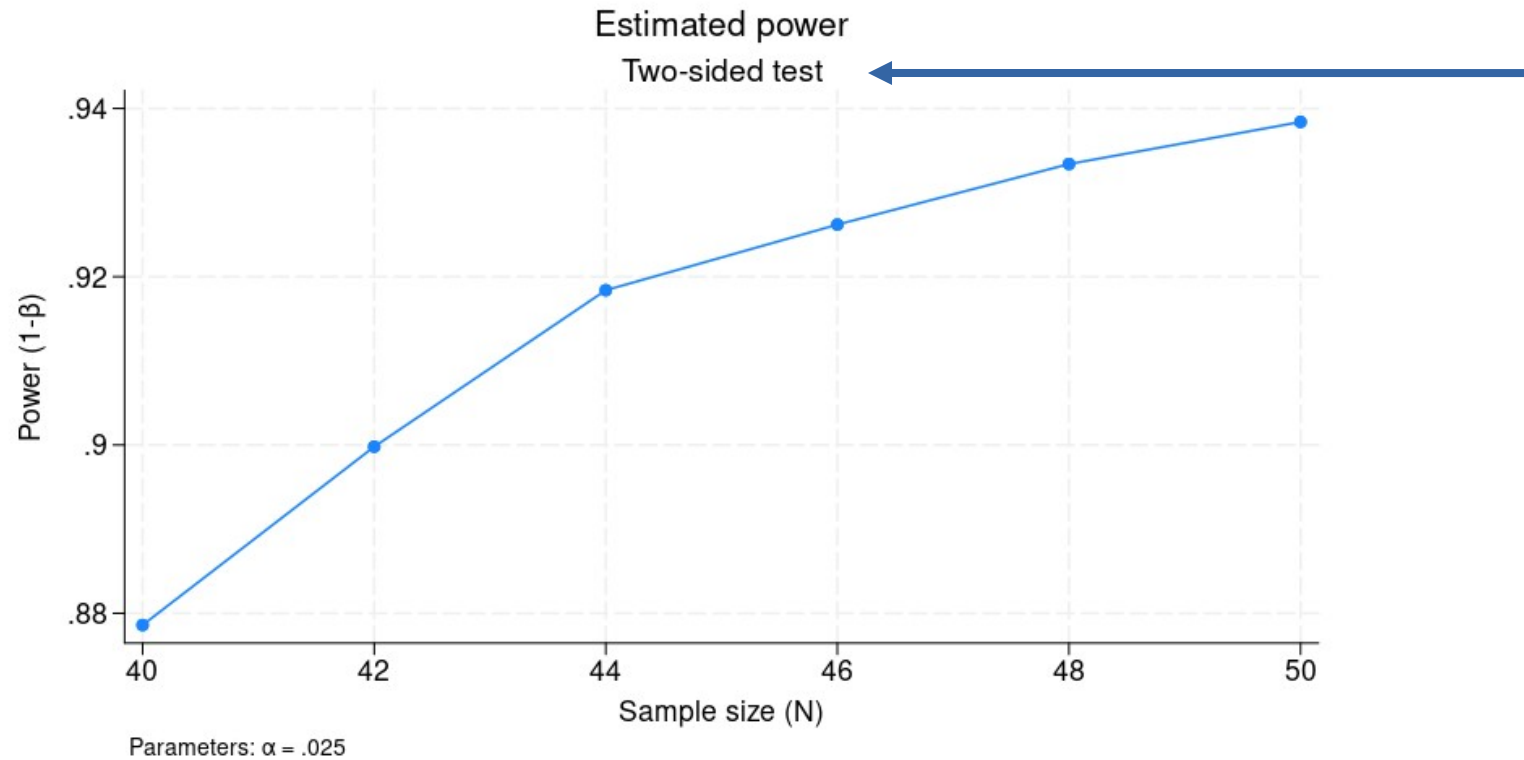
$$SE(\hat{power}) = \sqrt{\frac{power * (1 - power)}{reps}}$$

```
. // Calculate Monte Carlo error (standard error of power estimate)
. local SEpower = sqrt( r(power) * (1-r(power)) / 1000 )

. display "SE of power estimate: `SEpower'"
SE of power estimate: .007626008130077
```

Sample size calculation by simulation

```
. // Option 1: provide a numlist of sample sizes  
. set seed 9 // Bonus question: Why don't we use option seed()?   
  
. power onmeansim, n(40(2)50) alpha(0.025) m0(0) ma(0.5) reps(5000) graph
```



Customize output with an initializer

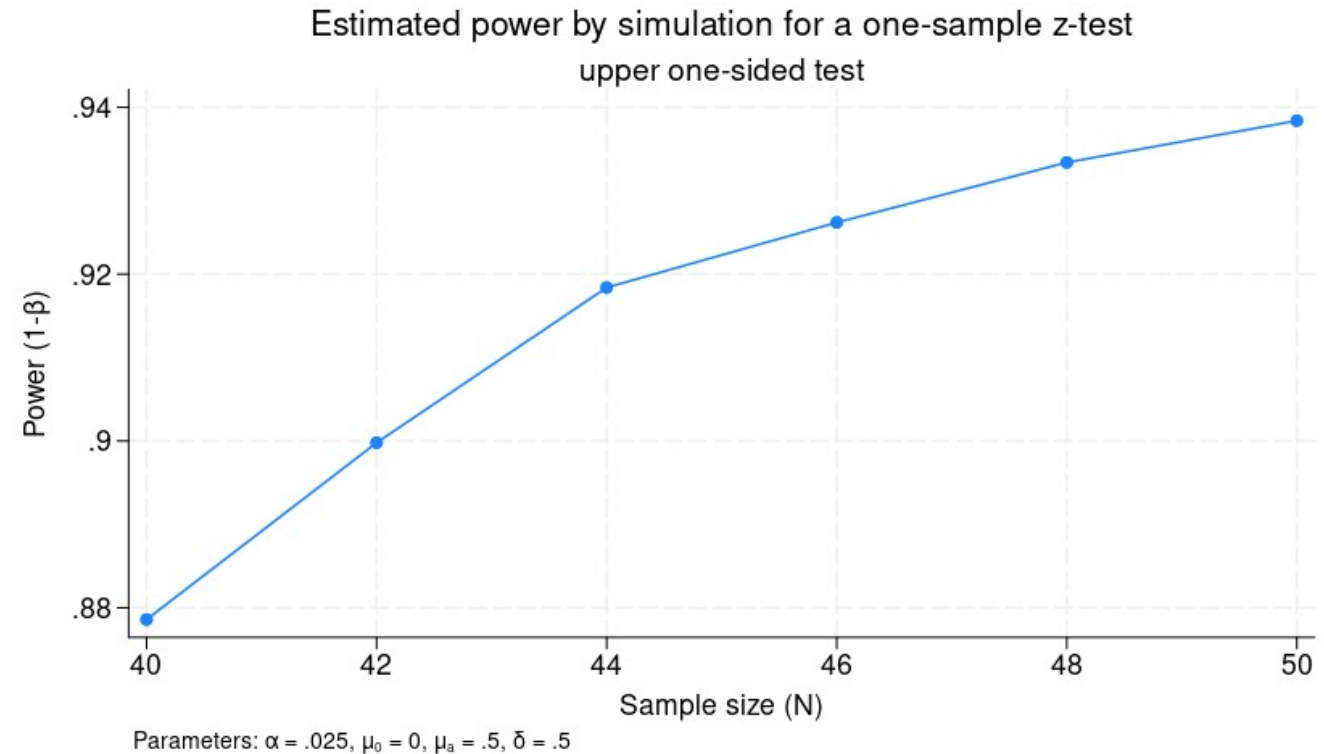
```
**# program power_cmd_onemeansim_init: use initializer to customize output
program power_cmd_onemeansim_init, sclass
    version 19
    sreturn clear
    sreturn local pss_numopts "m0 ma" // Options allowing numlists
    sreturn local pss_colnames "m0 ma delta" // Columns to add to default table
    sreturn local pss_colgrsymbols " $\mu_{\text{sub:0}}$   $\mu_{\text{sub:a}}$   $\delta$ " // Graph symbols
    sreturn local pss_title "by simulation for a one-sample z-test"
    sreturn local pss_subtitle "upper one-sided test"
end
```

Customize output with an initializer

```
. // Re-run with initializer  
. set seed 9  
  
. power onmeansim, n(40(2)50) alpha(0.025) m0(0) ma(0.5) reps(5000) graph table
```

Estimated power by simulation for a one-sample z-test
upper one-sided test

alpha	power	N	m0	ma	delta
.025	.8786	40	0	.5	.5
.025	.8998	42	0	.5	.5
.025	.9184	44	0	.5	.5
.025	.9262	46	0	.5	.5
.025	.9334	48	0	.5	.5
.025	.9384	50	0	.5	.5



Use `power` to calculate sample size by simulation

```
## program power_cmd_onemeansampsize: simulation-based sample-size evaluator
program power_cmd_onemeansampsize, rclass
    version 19
    syntax, power(numlist >0 <1 max=1) /// Requested power
    [ alpha(real 0.05)          /// Alpha level
      m0(real 0)                /// Mean under the null
      ma(real 0.5)              /// Mean under the alternative
      reps(integer 100)         /// Number of repetitions for power calculation
      seed(numlist max=1)       /// Random seed (no default seed)
      startval(integer 100)     /// Starting value for sample size
      maxval(integer 1000)     /// Maximum value for sample size
      minval(integer 1)        /// Minimum value for sample size
      powtol(real 0.01)        /// Tolerance for power search (default 1%)
      biter(integer 100)       /// Max iterations for bisection algorithm
      nfractional              /// Unused, but required by -gsdesign-
      onesided                /// Unused, but required by -gsdesign-
    ]

    // Set seed, if specified
    if ("`seed'" != "") set seed `seed'

    // Test powtol() to ensure it's not set too low
    assert (`powtol' > 0)

    // Test maxval() to ensure it's not set too low
    power_cmd_onemeansim, n(`maxval') alpha(`alpha') m0(`m0') ma(`ma') reps(`reps')
    if (r(power) < `power') {
        di as error "error: {bf:maxval()} too small"
        exit 480
    }

    // Test -minval()- to ensure it's not set too high
    power_cmd_onemeansim, n(`minval') alpha(`alpha') m0(`m0') ma(`ma') reps(`reps')
    if (r(power) > `power') {
        di as error "error: {bf:minval()} too large"
        exit 480
    }

    // Compare requested power to power attained via simulation at the starting value
    local thisN = `startval'
    local lastN = .
    power_cmd_onemeansim, n(`thisN') alpha(`alpha') m0(`m0') ma(`ma') reps(`reps')
    local pdiff = r(power) - `power'
```

```
    // Repeat guess-n-check process using bisection algorithm
    local i = 1
    while (abs(`pdiff') > `powtol') {
        local ++i
        if (`i' > `biter') {
            di as error "convergence not achieved"
            exit 430
        }

        // Bisection algorithm
        local lastN = `thisN'
        if (`pdiff' > 0) { // Overpowered: decrease N
            local maxval = `thisN'
        }
        else { // Underpowered: increase N
            local minval = `thisN'
        }
        local thisN = ceil((`maxval' + `minval') / 2)
        if (`thisN' == `lastN') { // Already converged
            continue, break
        }
        power_cmd_onemeansim, n(`thisN') alpha(`alpha') m0(`m0') ma(`ma') reps(`reps')
        local pdiff = r(power) - `power'
    }

    // Return results
    return scalar power = r(power)
    return scalar N = `thisN'
    return scalar alpha = `alpha'
    return scalar m0 = `m0'
    return scalar ma = `ma'
    return scalar delta = `ma' - `m0'
    return local direction = "upper"
end

## program power_cmd_onemeansampsize_init: use initializer to customize output
program power_cmd_onemeansampsize_init, sclass
    version 19
    sreturn clear
    return local pss_numopts "m0 ma" // Options allowing numlists
    return local pss_colnames "m0 ma delta" // Columns to add to default table
    return local pss_colgrsymbols "μ{sub:0} μ{sub:a} δ" // Graph symbols
    return local pss_title "by simulation for a one-sample z-test"
    return local pss_subtitle "upper one-sided test"
end
```

Use AI to code sample size by simulation

- Use the latest model you can get your hands on
 - Current frontier models did great, even with vague prompts and no MCP
write a power usermethod wrapper to calculate sample size for power_cmd_onemmeansim using bisection
 - Passed 1st try on low effort: Claude Sonnet 5.5, Opus 5.5, Fable 5.1, Google Gemini 3.8 Flash
 - ~50% pass rate, even with better prompts: Claude Sonnet 4.5, Opus 4.5 (last year's models)
 - Simple tools for simple tasks
 - Detailed, specific prompts become more important with more complex tasks
 - Even more complex tasks warrant a separate planning step at high effort (Buchanan rule)
- Trust no LLM
 - Good vibecoding-candidates would take you a long time to code but a short time to check
 - Conduct your own tests, and include edge cases
 - AI adversarial-reviewers are a good start, but don't trust a fox to guard the henhouse

Calculate sample size by simulation

```
. // Try it as a -power- command  
. power onemeansampsize, power(0.9) alpha(0.025) m0(0) ma(0.5) startval(100) reps(500) seed(9)
```

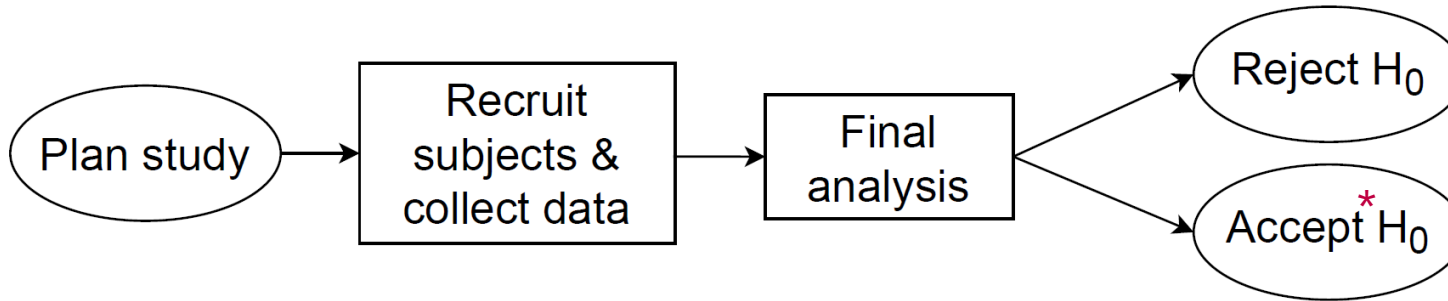
Estimated sample size by simulation for a one-sample z-test
upper one-sided test

alpha	power	N	m0	ma	delta
.025	.896	39	0	.5	.5

Fixed Sample Design vs Group Sequential Design

or: how I learned to stop worrying and accept the null

Fixed design

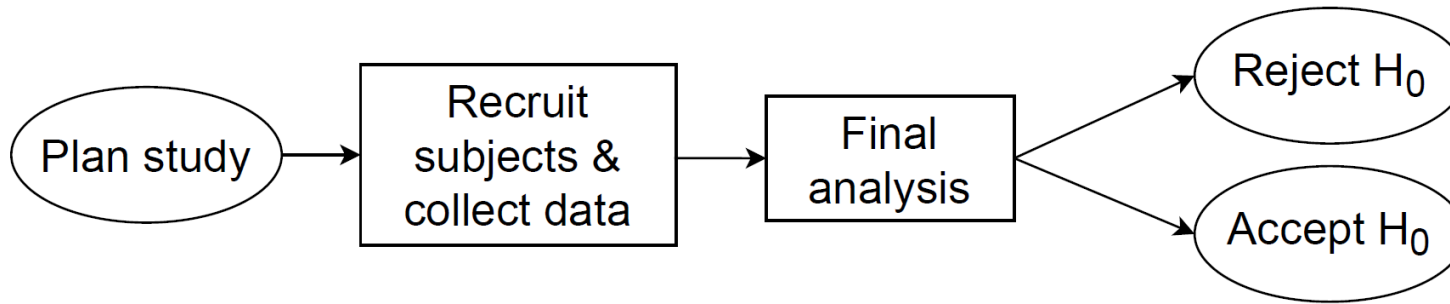


* the fact that we are unable to reject

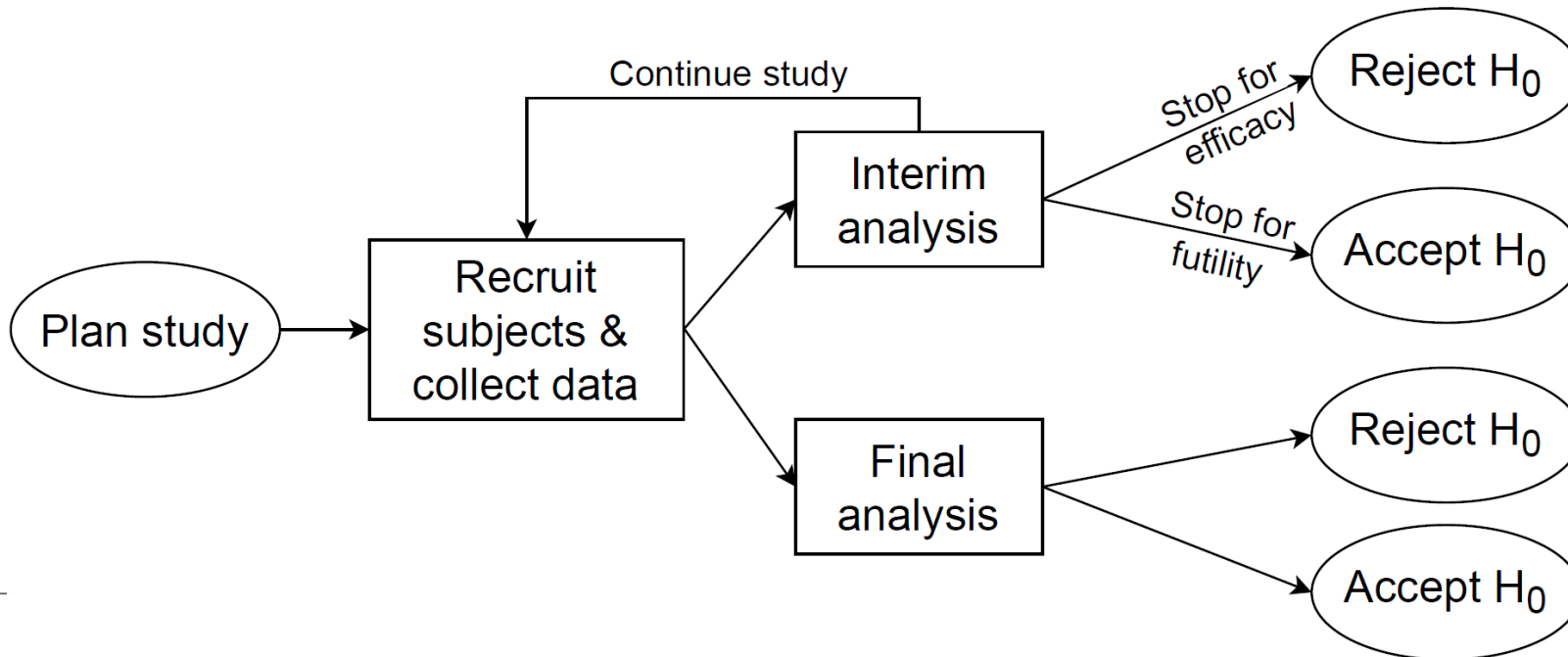
Fixed Sample Design vs Group Sequential Design

or: how I learned to stop worrying and accept the null

Fixed design

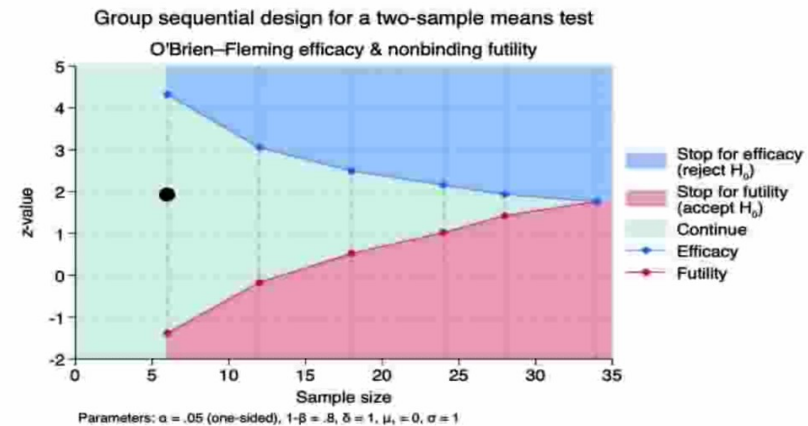


Group sequential design



GSD in action

Group sequential designs



STATA 18

From <https://youtu.be/hO2qW1NLrMk?si=gXsB8zDkAmq9K2IE> by Meghan Cain, StataCorp

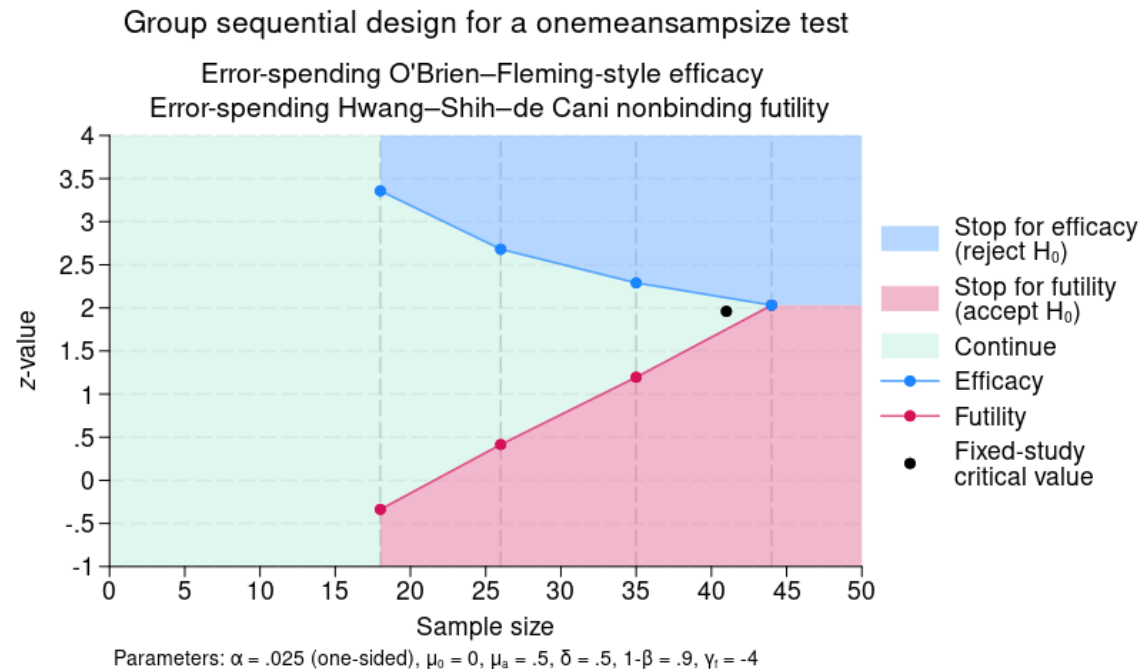
GSD with efficacy & futility bounds

```
. gsdesign onmeansampsize, power(0.9) alpha(0.025) m0(0) ma(0.5) startval(50) reps(500) seed(9) ///  
>                                onesided information(40 60 80 100) efficacy(errobfleming)          ///  
>                                futility(hsdecani(-4)) graphbounds
```

Group sequential design by simulation for a one-sample z-test
upper one-sided test

Efficacy: Error-spending O'Brien-Fleming style

Futility: Error-spending Hwang-Shih-de Cani, nonbinding, $\gamma = -4.0000$

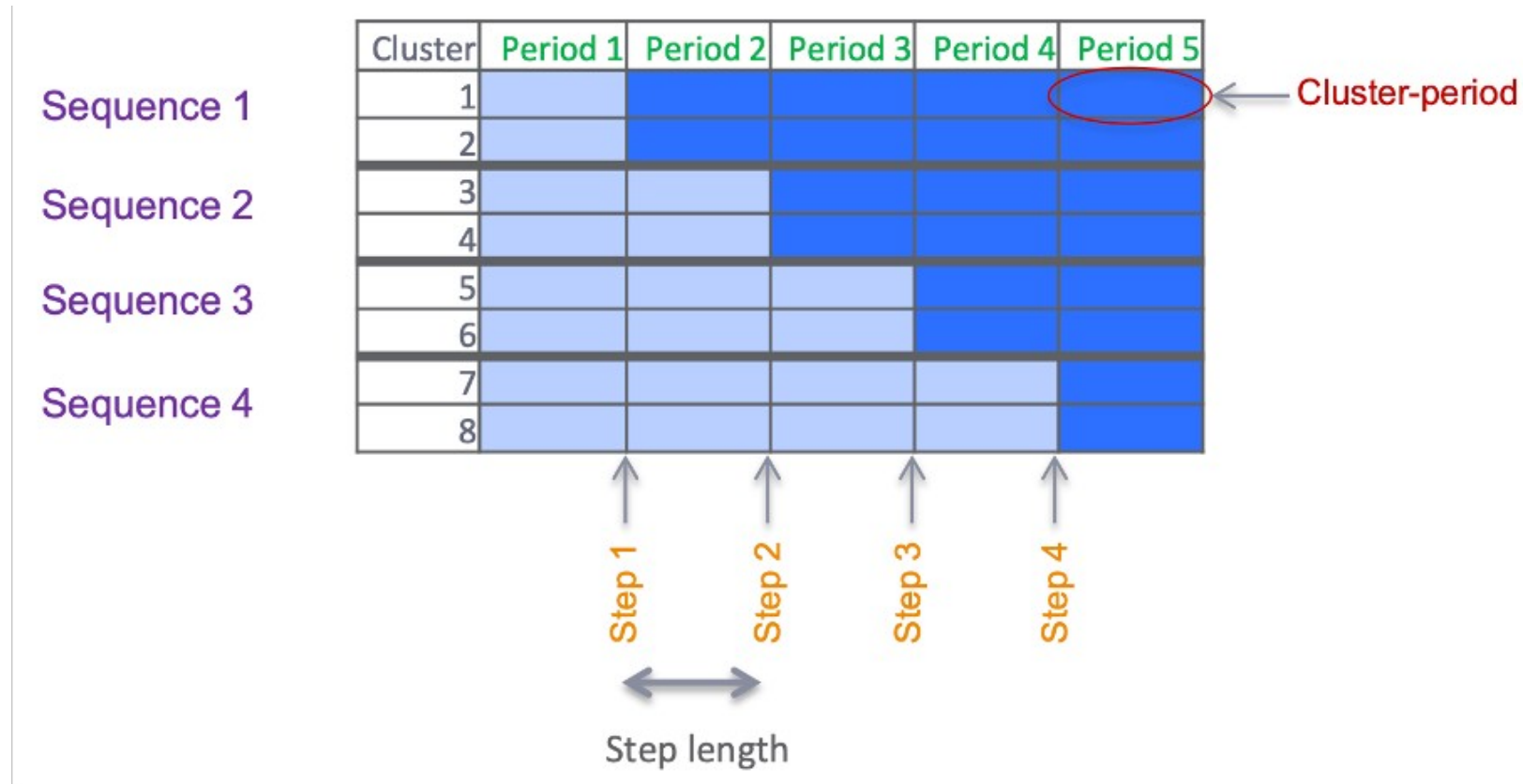


Stepped-Wedge Designs

Project of Biostatistics Intern Hongyin Lai

Stepped-wedge cluster randomized trial (SW-CRT)

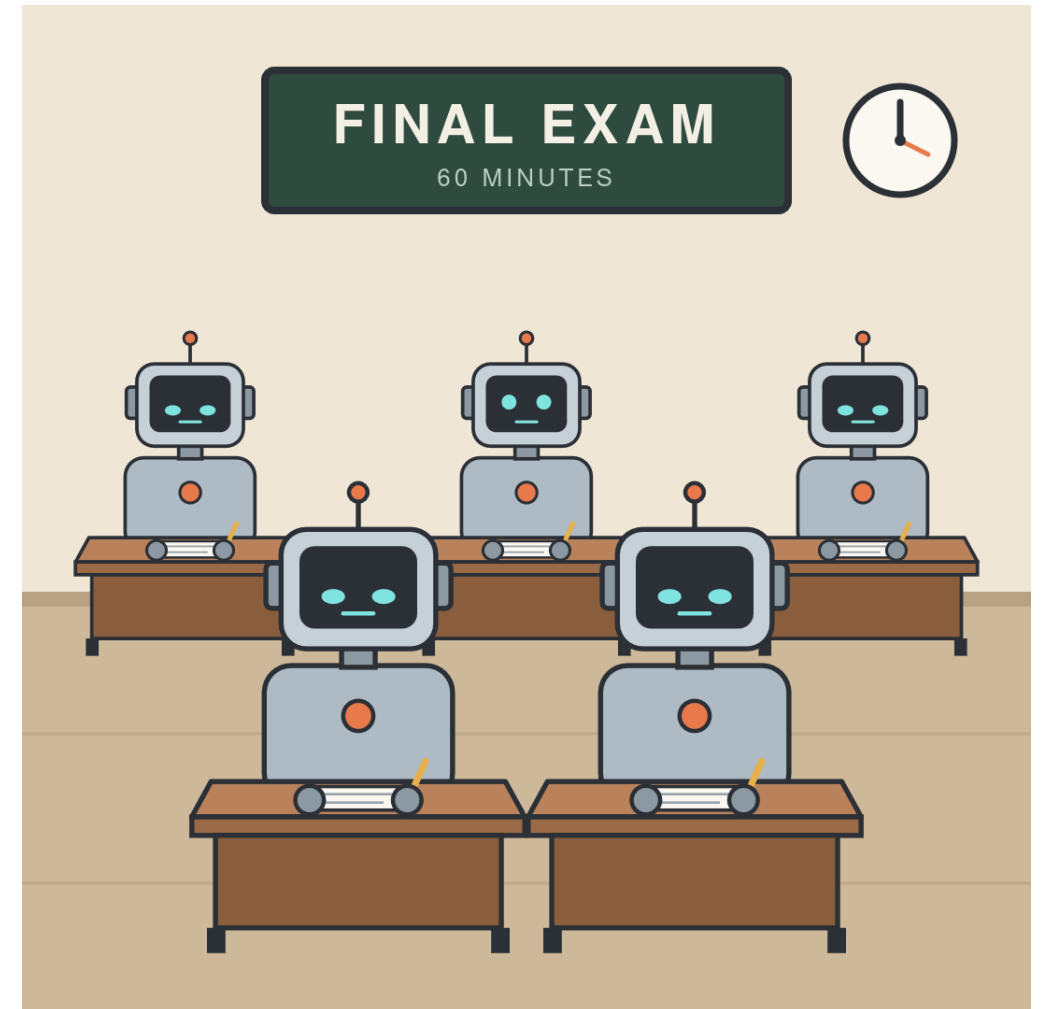
Clusters start the trial in the control condition and gradually switch (unidirectional) to the intervention over the study period
Popularly used in evaluation of public health interventions and health science research



Credits Slide: Hongyin Lai
Picture references: Taljaard, M. (2019, December 19). *Stepped wedge cluster trials: What, how, and when?* [Slides and webinar]. NIA IMPACT Collaboratory Grand Rounds.

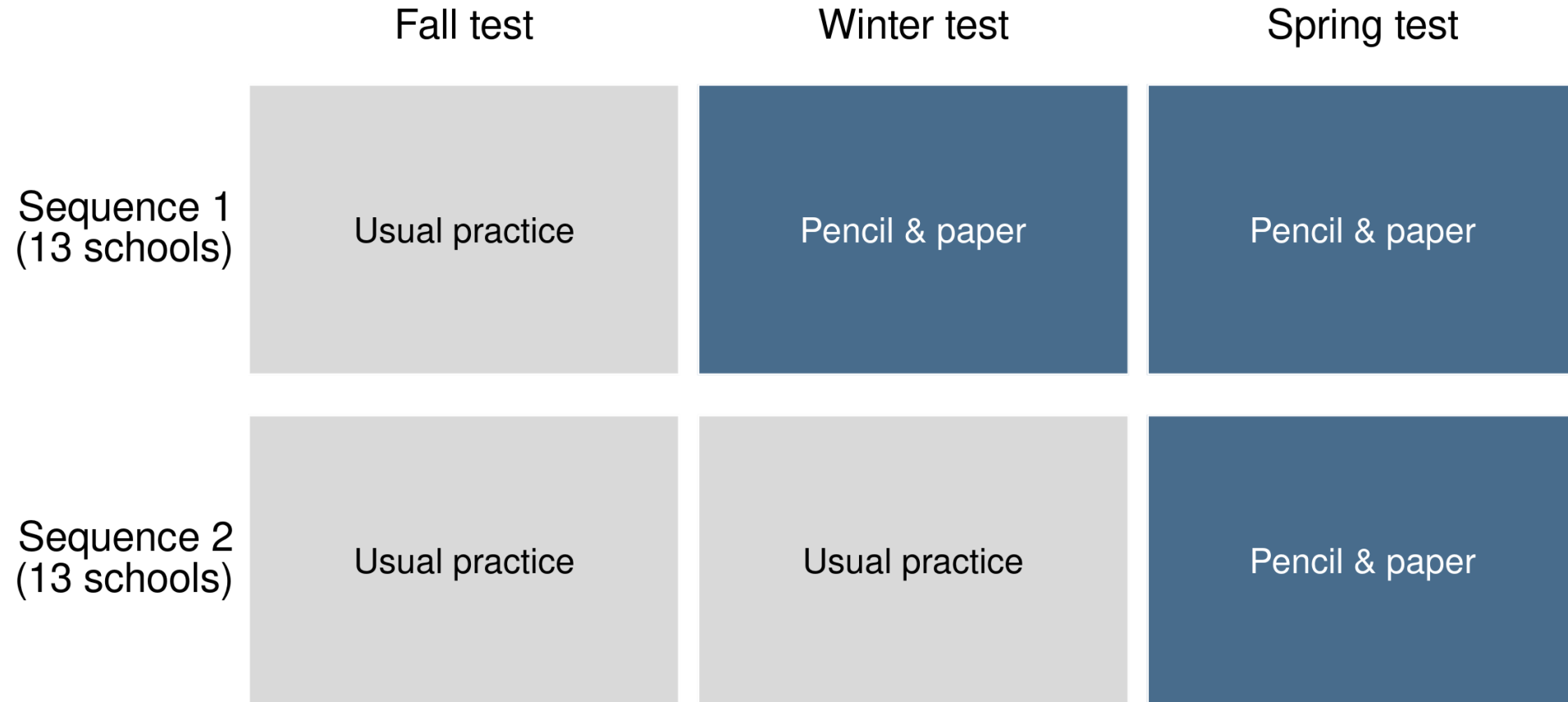
Example: Mitigating the effects of AI in the classroom

- **Problem:** AI writes homework and lab reports, so students learn less
- **Intervention:** Biology lab reports written by hand, in class
- **Design:** stepped wedge across Houston high schools; which schools go first is randomized
- **Outcome:** Texas's new state tests: fall, winter, spring (continuous response)
- **Cohort:** Same students take all three tests (closed cohort)
- **Goal:** detect a 0.2 SD gain in standardized scores



Credits AI-generated image of AI-enhanced test-takers: Claude Design

Stepped-Wedge Educational Study



Which schools go first is randomized. Same Biology students take all three tests.

Stepped-Wedge vs DID

Stepped wedge trial	Difference-in-differences	Our example
Cluster	Unit (i)	A high school
Sequence	Treatment cohort, timing group (g)	The 13 schools that switch together
Step	Adoption date, first treated period	After the fall test, or after the winter test
Period	Time period (t)	Fall, winter, spring test
Cluster-period	Unit-by-time cell (i, t)	One school's scores on one test
Exposure time	Event time, relative time ($t - g$)	Tests taken since the school switched
Clusters not yet switched	Not-yet-treated controls	Sequence 2 at the winter test

A stepped wedge trial is a staggered-adoption DID whose adoption schedule is randomized:
no parallel-trends assumption needed

Example: Mitigating the effects of AI in the classroom

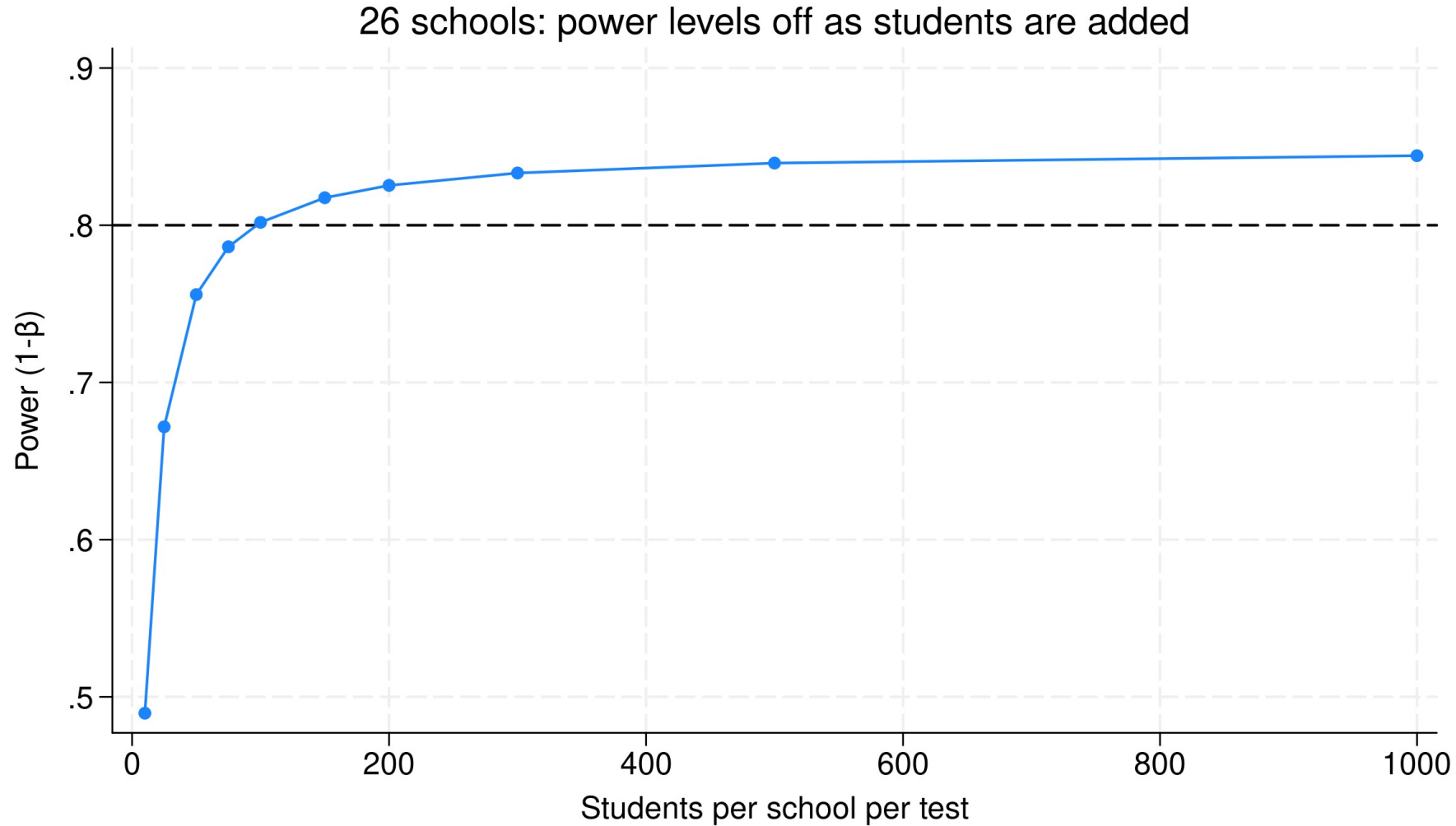
```
. * 26 schools (13 per sequence), effect 0.2 SD, two-sided alpha 0.05
. * rhow: correlation between students in the same school on the same test
. * cac: how much a school's mean carries over from test to test (cluster autocorrelation)
. * iac: correlation of a student with themselves across tests (individual autocorrelation)
. power swme, delta(0.2) seqclusters(13 13) nperiods(3) n(10 25 50 100 1000)      ///
>      design(cc) constr(be) sigma2(1) rhow(0.18) cac(0.9) iac(0.7) table
```

Estimated power for stepped-wedge mixed-effects model
Two-sided test | Design: cc | Correlation: be | Family: Gaussian | Link: Identity

alpha	power	Total_Clust	N_Period	N	N_Total	Effect_Size	Sigma2	df	Rhow	CAC	IAC
.05	.4896	26	3	10	260	.2	1	22	.18	.9	.7
.05	.6718	26	3	25	650	.2	1	22	.18	.9	.7
.05	.7559	26	3	50	1300	.2	1	22	.18	.9	.7
.05	.8018	26	3	100	2600	.2	1	22	.18	.9	.7
.05	.8443	26	3	1,000	26000	.2	1	22	.18	.9	.7



Example: Mitigating the effects of AI in the classroom



Available Community-Contributed Packages

- **steppedwedge**
 - Hemming & Girling, Stata Journal (2014)
 - `findit steppedwedge`
 - Power and detectable difference, cross-sectional designs, Hussey–Hughes mixed model
 - Has dialog box
- **power swgee**
 - Gallis, Li, Turner et al, Stata Journal (2022)
 - `ssc install power_swgee`
 - Power for cross-sectional and closed-cohort designs, analyzed by GEE
 - Works as a power command
- **swpermute**
 - Thompson et al, Stata Journal (2019)
 - `ssc install swpermute`
 - Permutation tests that reshuffle clusters across sequences
 - Analysis, not design



Code from all slides available upon request
aasher@stata.com