Title

stata.com

power, graph — Graph results from the power command

Description

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

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

Quick start

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

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

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

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

Graph of power estimates versus the specified list of sample sizes

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

Add labels for distinct values on the y axis

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

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

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

Plots of power for each significance level

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

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

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

Plots of power for combinations of alternative means and significance levels

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

Same as above

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

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

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

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

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

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

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

Menu

Statistics > Power, precision, and sample size

Syntax

Produce default graph

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

Graph power against sample size

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

Graph sample size against target parameter

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

Graph effect size against sample size

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

Produce other custom graphs

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

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

dimlist may contain any of the following columns:

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

colspec is

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

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

where *lablist* is defined as

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

elablist is defined as

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

and the #s are the levels of the dimension.

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

Suboptions

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

Main

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Labels

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

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

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

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

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

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

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

Multiple columns may be relabeled by typing, for example,

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

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

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

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

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

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

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

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

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

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

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

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

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

Plot

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

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

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

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

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

Add plots

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

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

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

twoway_options are any of the options documented in [G-3] ***twoway_options***. These include options for titling the graph (see [G-3] ***title_options***); for saving the graph to disk (see [G-3] ***saving_option***); for controlling the labeling and look of the axes (see [G-3] ***axis_options***); for controlling the look, contents, position, and organization of the legend (see [G-3] ***legend_options***); for adding lines (see [G-3] ***added_line_options***) and text (see [G-3] ***added_text_options***); and for controlling other aspects of the graph's appearance (see [G-3] ***twoway_options***).

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

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

Remarks and examples

stata.com

Remarks are presented under the following headings:

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

Using power, graph

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

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

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

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

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

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

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

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

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

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

Graph symbols

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

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

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

Default graphs

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

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

Example 1: Power curves

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

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

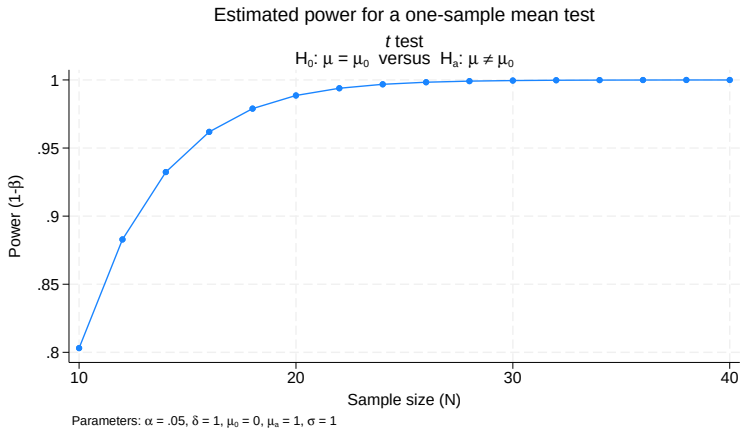


Figure 1.

As expected, power increases as sample size increases.

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

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

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

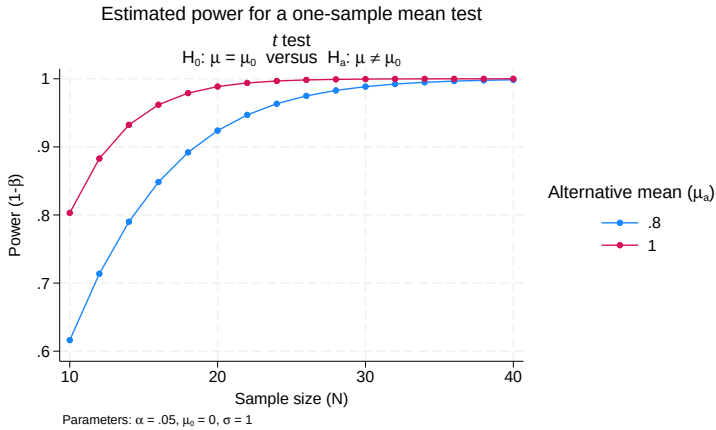


Figure 2.

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

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

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

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

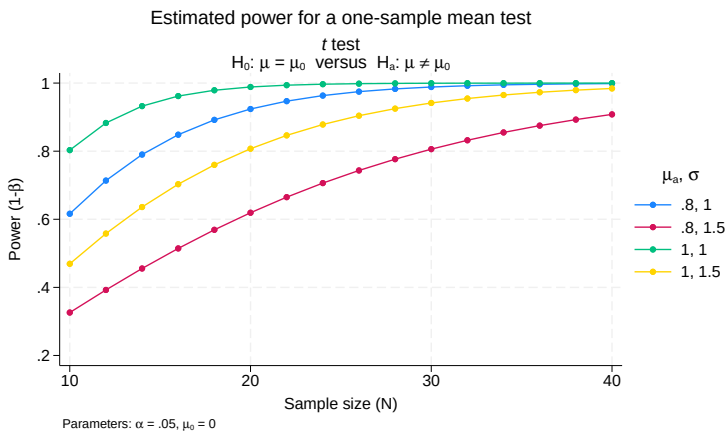


Figure 3.

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

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

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

◁

▷ Example 2: Sample-size curves

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

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

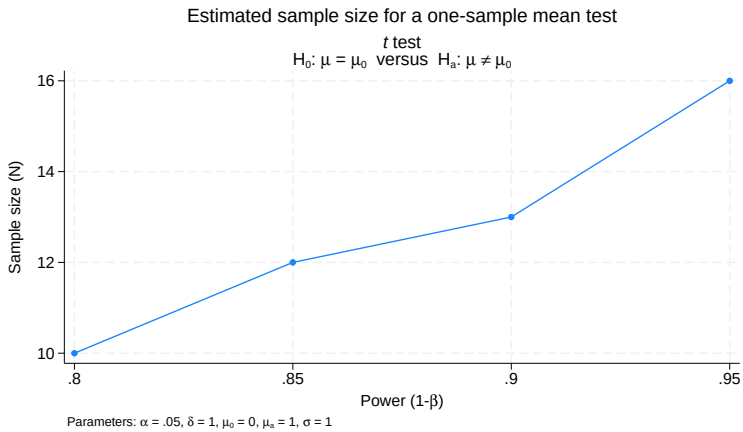


Figure 4.

The sample size increases as the power increases.

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

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

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

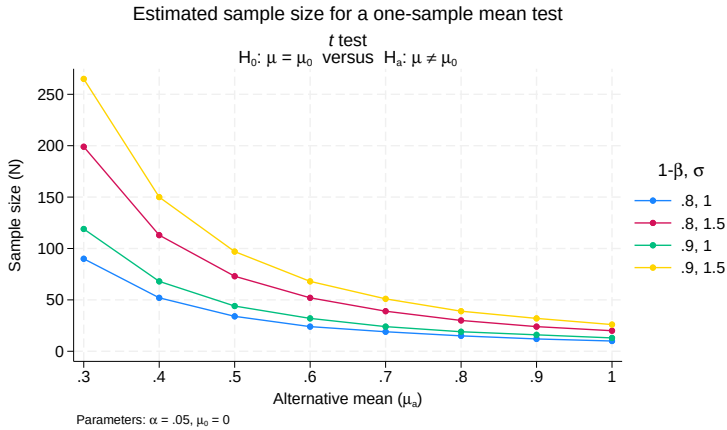


Figure 5.

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

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

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

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

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

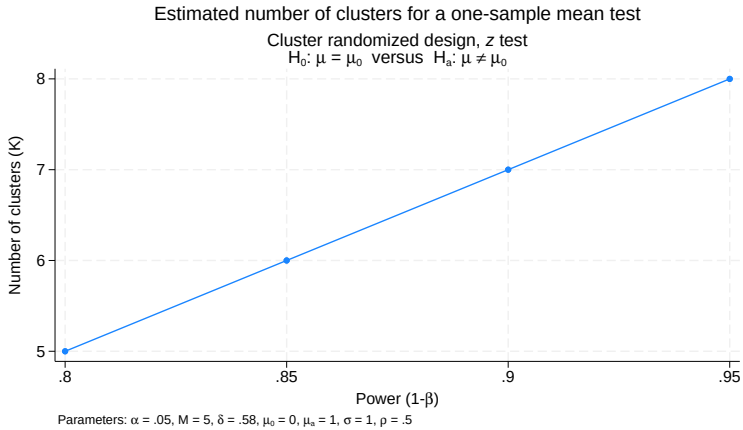


Figure 6.

The number of clusters increases as the power increases.

◁

▷ Example 3: “Effect-size” curves

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

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

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

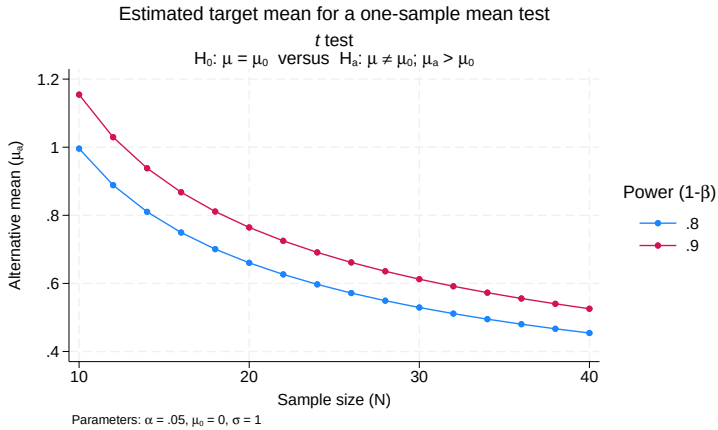


Figure 7.

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

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

Changing default graph dimensions

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

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

Example 4: Changing default graph axes

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

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

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

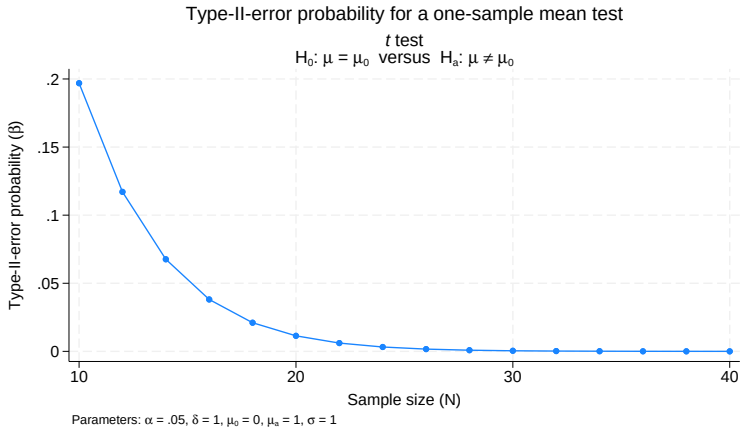


Figure 8.

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

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

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

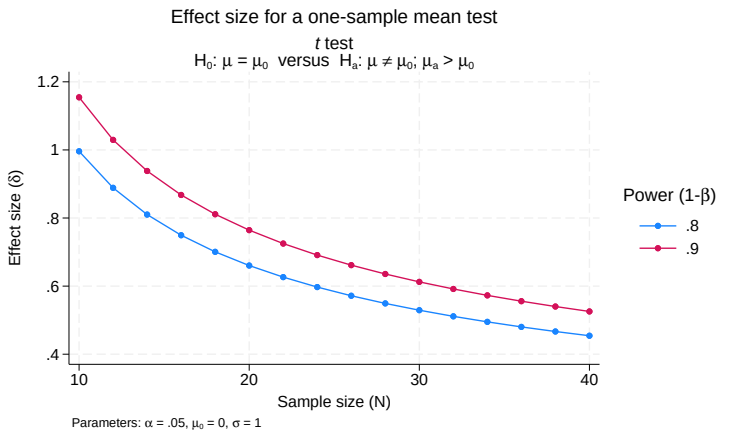


Figure 9.

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

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

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

```
. power onemean 0 (0.3(0.1)1), n(10 20 40) graph(x(ma))
```

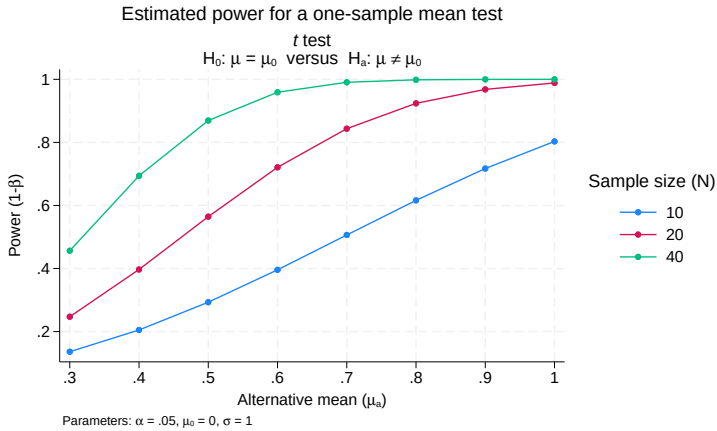


Figure 10.

The x axis now contains the values of the alternative mean, and a separate curve is now plotted for each sample size.

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

◁

▷ Example 5: By graphs and multiple graphs

Let's return to figure 3 demonstrating multiple power curves on one graph. Suppose we want to more easily see the impact of standard deviation on power given an alternative mean value. We can produce a separate plot for each of the mean values by specifying the column `ma` in the `bydimension()` suboption (abbreviated to `by()`) within `power`'s `graph()` option.

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

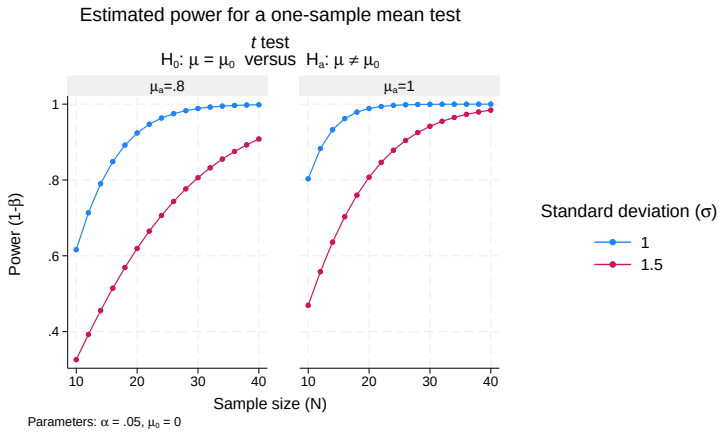


Figure 11.

For examples of how to modify the look of a `by` graph, see example 8.

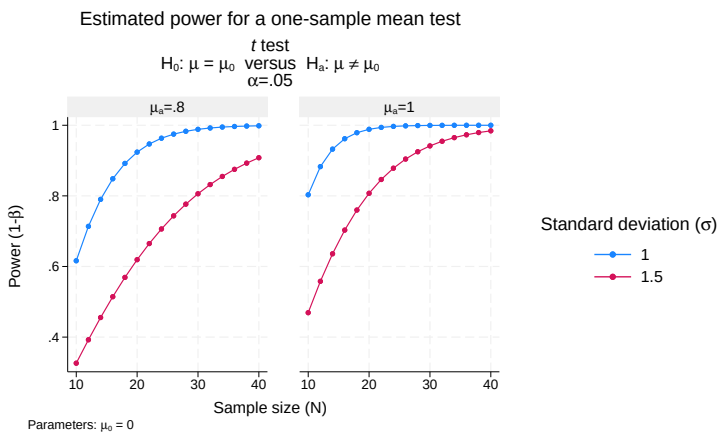
In the presence of many varying parameters, even `by` graphs may look crowded. In this case, you may consider producing multiple `by` graphs. In the example above, suppose that we also want to vary the significance level α. We add the `alpha(0.05 0.1)` option to the previous command.

```
. power onemean 0 (0.8 1), n(10(2)40) sd(1 1.5) alpha(0.05 0.1) graph(by(ma))
(output omitted)
```

The above command produces a graph containing two `by` graphs. Each `by` graph contains four curves in which each corresponds to a unique combination of values of the standard deviation and significance level. We leave this for you to verify.

This is a lot of information for a single graph to convey, so instead, we request that a separate graph be produced for each of the significance levels by specifying the `graphdimension(alpha)` suboption (abbreviated to `graph()`) within `power`'s `graph()` option.

```
. power onemean 0 (0.8 1), n(10(2)40) sd(1 1.5) alpha(0.05 0.1)
> graph(by(ma) graph(alpha))
```



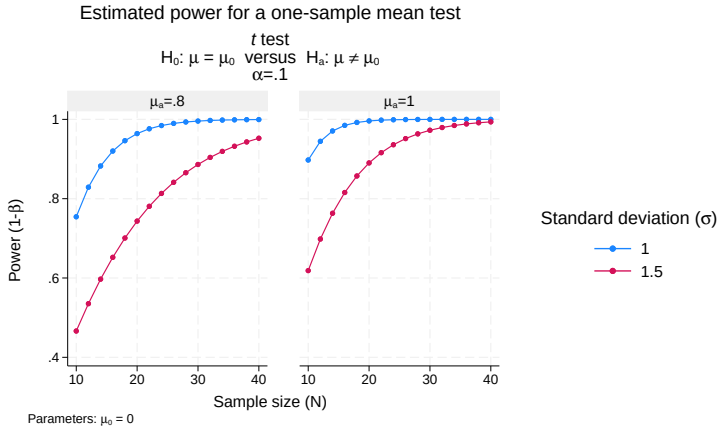


Figure 12.

Changing the look of graphs

▷ Example 6: Modifying axis labels

Reasonable defaults for axis labels are chosen based on your results. You can modify the defaults by using any of `power, graph()`'s labeling suboptions or `graph twoway`'s *axis_label_options*; see [G-3] ***axis_label_options***.

For example, we can request that ticks and labels of the y and x axes be placed for each distinct value instead of using equally spaced values as in figure 1.

```
. power onemean 0 1, n(10(2)20) graph(yvalues xvalues ylabel(, format(%4.3f)))
```

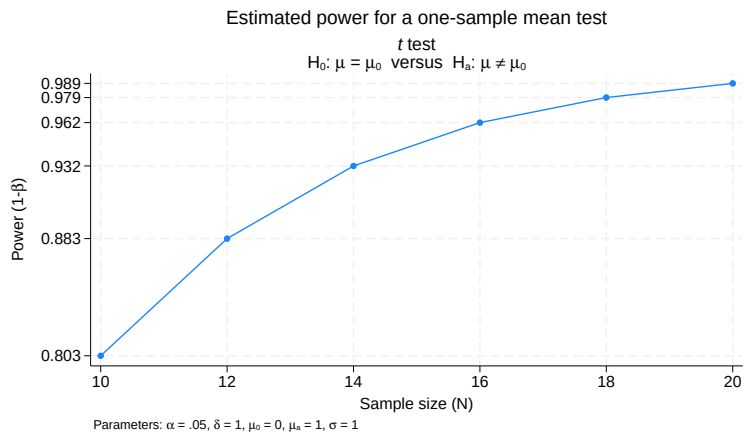


Figure 13.

In this example, we specified fewer sample sizes to obtain a more readable graph. To further improve readability, we also changed the default format of the values on the y axis to show only three decimal points by using `ylabel(, format(%4.3f))`.

We can use `ylabel()` and `xlabel()` to add text labels for some of the axis values. For example, suppose that our budget is 30 subjects. We can use `xlabel()` to label the sample-size value of 30 as "Budgeted".

```
. power onemean 0 1, n(10(2)40) graph(xlabel(30 "Budgeted", add))
```

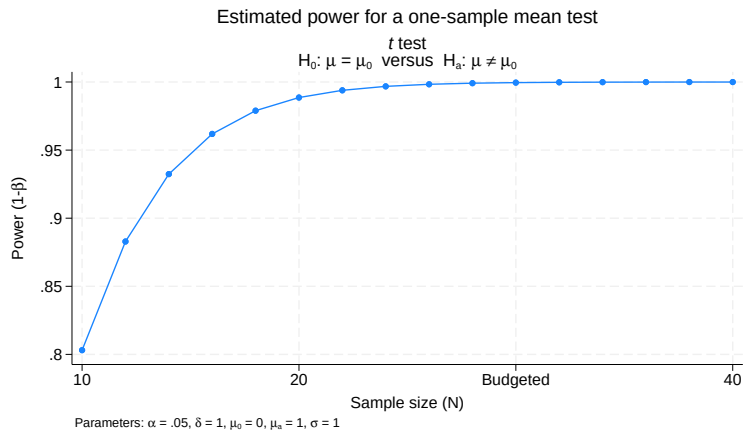


Figure 14.

We can use `ytitle()` and `xtitle()` to change the axis titles.

```
. power onemean 0 1, n(10(2)20) graph(ytitle("Power") xtitle("Sample size")
> title("Estimated power") subtitle("") note(""))
```

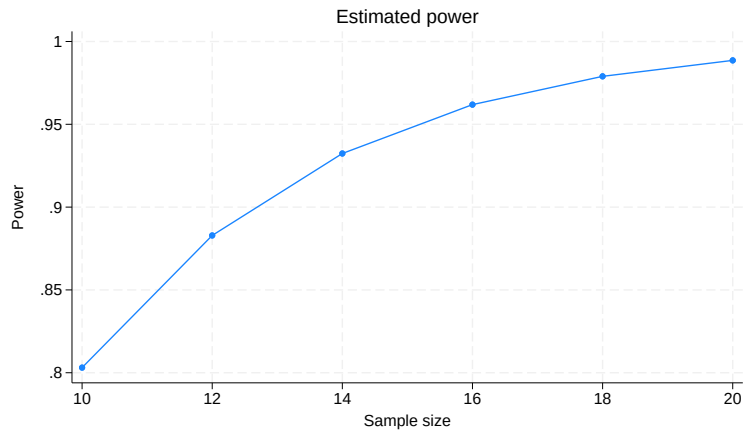


Figure 15.

In addition to modifying the axis titles, we also shortened the default title and suppressed the default subtitle and note.

You may find the `collabels()` suboption useful to override the default column labels. The specified column labels will be used wherever the corresponding column is used on the graph.

For example, change the default labels of the power, sample-size, and alternative-mean columns to, respectively, "Power", "N", and "Alternative mean" in figure 2 as follows:

```
. power onemean 0 (0.8 1), n(10(2)40)
> graph(collabels(N "N" power "Power" ma "Alternative mean"))
```

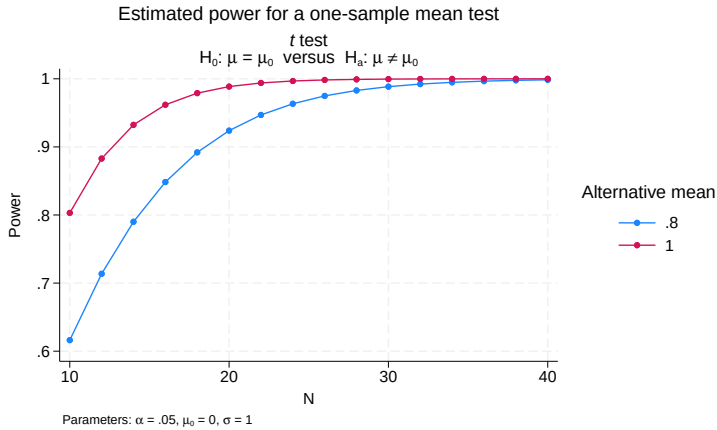


Figure 16.

For overlaid plots, we may consider alternative labeling of the plotted curves in the legend by using the `nosimplelabels` suboption (abbreviated to `nosimple`). We also suppress the legend title and request that an equality sign with a space on each side be used as a separator.

```
. power onemean 0 (0.8 1), n(10(2)40)
> graph(nosimple legend(title("")) eqsep(" = "))
```

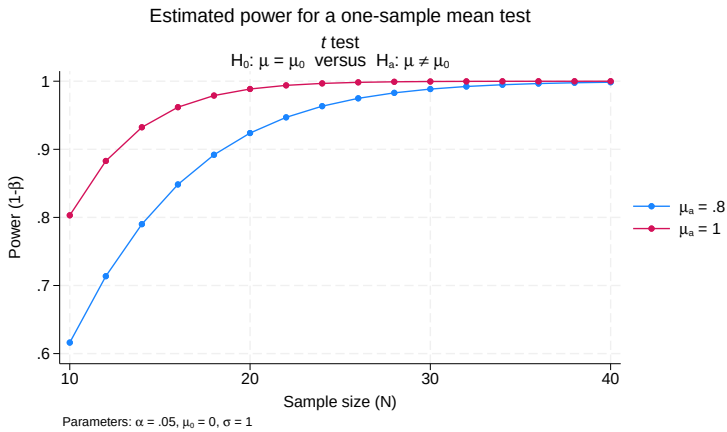


Figure 17.

Example 7: Plot options

We can use the `plotopts()` and `plot#opts()` suboptions within `graph()` to modify the default look of the plotted lines. If there are multiple curves, the `plotopts()` suboption will apply changes to all curves. Use the corresponding `plot#opts()` suboption to change the look of only the specific #th curve.

Here are a few examples of using these suboptions.

```
. power onemean 0 (0.1(0.1)1), graph(plotopts(mlabel(N) mlabpos(1)))
```

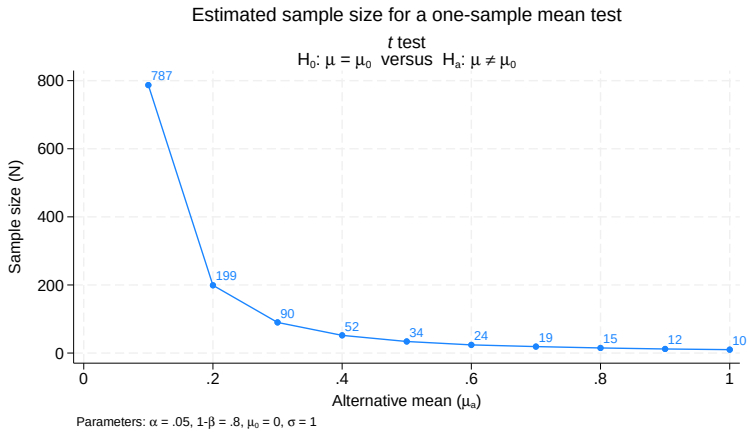


Figure 18.

We specified `mlabel()` within the `plotopts()` suboption to label each data point on the graph with its corresponding sample-size value. `mlabpos()` was used to place the marker labels at the one o'clock position.

For plots containing multiple curves such as in figure 3, the `plotopts()` suboption controls the look of all curves. For example, we can change the marker symbol from the default solid circle to a solid triangle.

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

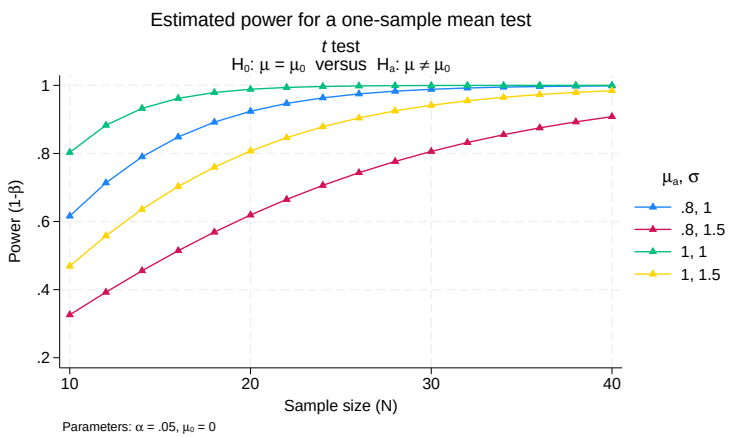


Figure 19.

To control the look of each curve, we can use multiple `plot#opts()` suboptions. For example, we can request that the curves corresponding to the same standard deviation be plotted using the same color:

```
. power onemean 0 (0.8 1), n(10(2)40) sd(1 1.5)
> graph(plot3opts(color(stblue)) plot4opts(color(stred)))
```

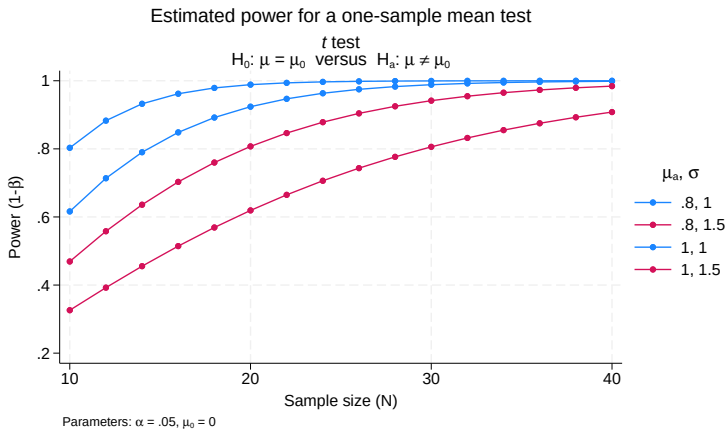


Figure 20.

Example 8: Modifying the look of by-graphs

In figure 11, we created a by-graph, where each graph appeared tall and narrow because of the legend on the right side. To allow extra width for the graphs, we could move the legend to the bottom and place the labels in two columns by using the `legend()` option. Instead, we will specify `scheme(stcolor_alt)` within `power`'s `graph()` option to use a scheme that automates the placement of the legend at the bottom.

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

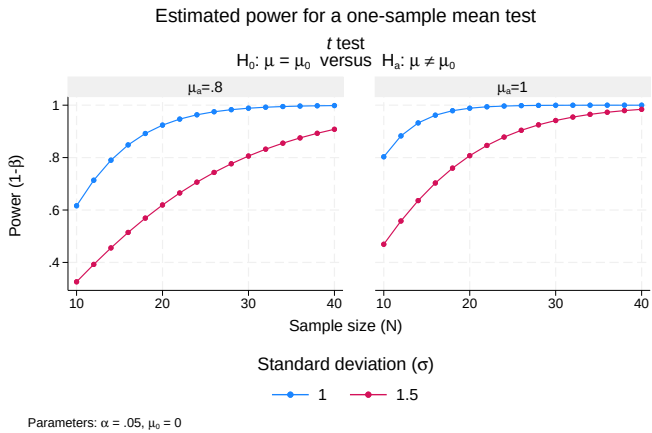


Figure 21.

The look of by graphs is further controlled by the byopts() suboption specified within power's graph() option.

For example, in figure 11, we can specify yrescale within the byopts() suboption to allow the scales of the two by graphs to differ. We use the alternative means of 0.5 and 1 instead of 0.8 and 1 to demonstrate differences between scales.

```
. power onemean 0 (0.5 1), n(10(2)40) sd(1 1.5) graph(by(ma)
> scheme(stcolor_alt) byopts(yrescale))
```

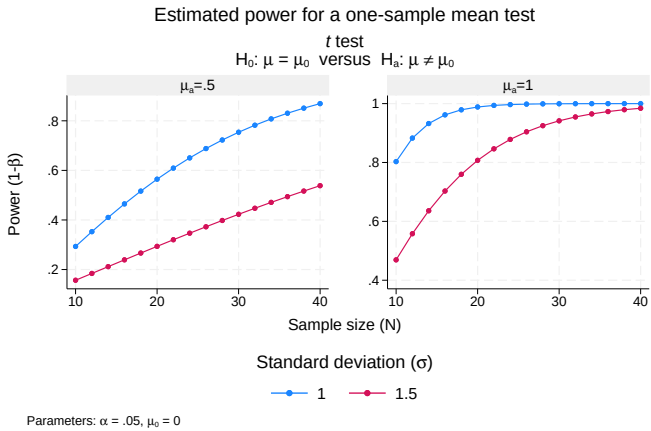


Figure 22.

We can use byopts() to change the overall graph title and subtitle.

```
. power onemean 0 (0.5 1), n(10(2)40) sd(1 1.5) graph(by(ma)
> scheme(stcolor_alt) byopts(yrescale title("Power vs sample size")
> subtitle("")))
```

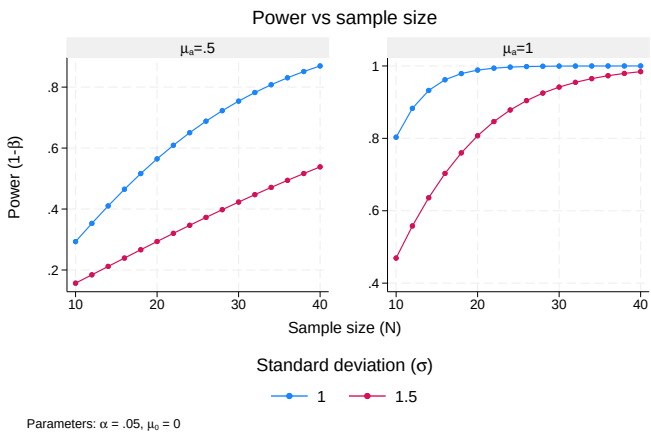


Figure 23.

Note that if you use title() and subtitle() outside byopts(), you will change the title and subtitle of the individual by graphs and not the overall graph.

◁

Parallel plots

Sometimes, you may be interested in comparing powers of parallel sets of parameters, that is, parameters that vary in parallel instead of being nested. In this situation, the results represent a collection of data points rather than a curve, and they are displayed on the graph as a scatterplot without connecting points.

For such parallel plots, the default display of the results on the y axis may be cumbersome. A more appealing look may be a graph that swaps the y and x axes, the horizontal graph. Such a look may be achieved by specifying the `horizontal` suboption within `graph()`.

```
. power onemean 0 (0.1(0.1)0.9), sd(1(0.1)1.9) parallel
> graph(x(ma sd) horizontal nosimple ytitle(""))
```

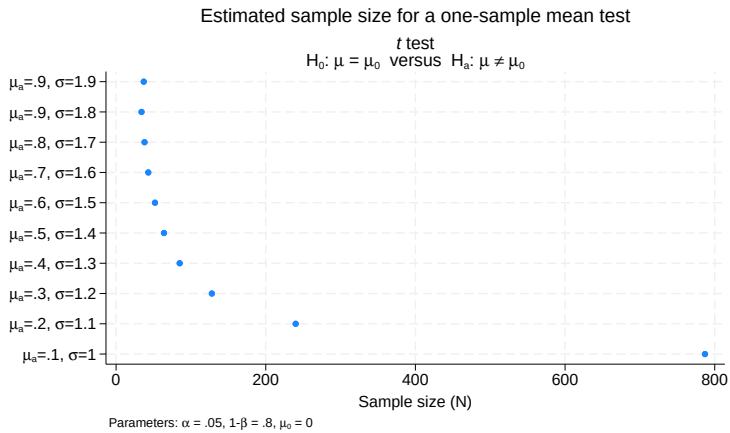


Figure 24.

To improve the look of the horizontal graph, we specified the `nosimplelabels` suboption to request that the labels on the y axis include the parameter symbol; we also suppressed the y-axis title.

Also see

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

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