ciwidth, graph — Graph results from the ciwidth command

Description

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

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

Quick start

Graph of sample-size estimates versus the specified list of CI width values

```
ciwidth onemean, probwidth(0.9) width(0.5(0.5)2) graph
```

Graph of sample-size estimates versus the specified list of confidence levels

```
ciwidth onemean, level(80(5)90) probwidth(0.9) width(0.5) graph
```

Same as above, but use significance level as the x axis

```
ciwidth onemean, level(80(5)90) probwidth(0.9) width(0.5) ///
    graph(xdimension(alpha))
```

Graph of probability of CI width versus the specified list of CI width values

```
ciwidth onemean, n(10) width(0.5(0.5)2) graph
```

Add labels for distinct values on the y axis

```
ciwidth onemean, n(10) width(0.5(0.5)2) graph(yvalues)
```

Same as above, but display the probability of CI width with only three decimal points

```
ciwidth onemean, n(10) width(0.5(0.5)2) ///
    graph(yvalues ylabel(,format(%4.3f)))
```

Graph of CI width versus the specified list of sample sizes at each confidence level

```
ciwidth onemean, level(90 95) n(30(5)45) probwidth(0.9) graph
```

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

```
ciwidth onemean, level(90 95) n(30(5)45) probwidth(0.9) ///
    graph(bydimension(level))
```

Menu

Statistics > Power, precision, and sample size

Syntax

Produce default graph

```
ciwidth ..., graph ...
```

Graph sample size against CI width

```
ciwidth ..., graph(y(N) x(width)) ...
```

Graph sample size against probability of CI width

```
ciwidth ..., graph(y(N) x(Pr_width)) ...
```

Graph CI width against sample size

```
ciwidth ..., graph(y(width) x(N)) ...
```

Graph probability of CI width against sample size

```
ciwidth ..., graph(y(Pr_width) x(N)) ...
```

Produce other custom graphs

```
ciwidth ..., graph(graphopts) ...
```

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

dimlist may contain any of the following columns:

column	Description
`level`	confidence level
`alpha`	significance level
`N`	total number of subjects
`N1`	number of subjects in the control group
`N2`	number of subjects in the experimental group
`nratio`	ratio of sample sizes, experimental to control
`Pr_width`	probability of CI width
`width`	CI width
method_columns	columns specific to the method specified with `ciwidth`

colspec is

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

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

where *lablist* is defined as

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

elablist is defined as

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

and the #s are the levels of the dimension.

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

Suboptions

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

Main

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Labels

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

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

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

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

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

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

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

Multiple columns may be relabeled by typing, for example,

```
collabels(N "N" v "Variance")
```

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

```
collabels(alpha "{&alpha}" N1 "N{sub:1}")
```

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

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

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

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

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

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

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

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

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

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

Plot

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

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

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

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

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

Add plots

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

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

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

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

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

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

Remarks and examples

`ciwidth, graph` produces sample-size curves and other graphical output from the `ciwidth` command. These graphs are useful for visualizing the results of sensitivity analysis, which investigates the effect of varying study parameters on sample size and CI width. The true values of study parameters are usually unknown. PrSS uses best guesses for these values. It is important to evaluate the sensitivity of the computed sample size to the chosen values of study parameters. For example, to evaluate variability of sample-size values, you can compute sample sizes for various ranges of values for the parameters of interest and display the resulting sample sizes in a table (see [PSS-3] **ciwidth, table**) or plot them on a graph.

Remaining remarks are presented under the following headings:

Using ciwidth, graph

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

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

By default, `ciwidth, graph` plots the column corresponding to the estimated parameter on the y axis: `N`, when sample size is computed; `width`, when CI width is computed; and `Pr_width` when probability of CI width is computed. When there is only one varying column, the x axis uses this column by default. When there are multiple varying columns, the default x axis depends on what is being computed.

If sample size is computed (sample-size determination), the default x axis is CI width if CI width varies. If CI width does not vary, the sample size is plotted against one of the other varying parameters.

If CI width is computed (precision determination), the default x axis is the sample size if sample size varies. If the sample size does not vary, CI width is plotted against one of the other varying parameters.

If probability of CI width is computed, the default x axis is the sample size if sample size varies. If the sample size does not vary, probability of CI width is plotted against one of the other varying parameters.

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

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

Graph symbols

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

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

Symbol	Description
$100(1-\alpha)$	confidence level
α	significance level
N	total sample size
N_1	sample size of the control group
N_2	sample size of the experimental group
N_2/N_1	ratio of sample sizes, experimental to control
p_{width}	probability of CI width
w	CI width
method_symbols	symbols specific to the *method* specified with `ciwidth`

Default graphs

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

In what follows, we graph the results of PrSS analysis for a one-mean CI; see [PSS-3] **ciwidth onemean**.

▷ Example 1: Sample-size curves

When we compute sample size given a range of CI widths, `ciwidth, graph` plots sample size on the y axis and CI width on the x axis.

```
. ciwidth onemean, width(0.5(0.25)2) probwidth(0.9) graph
```

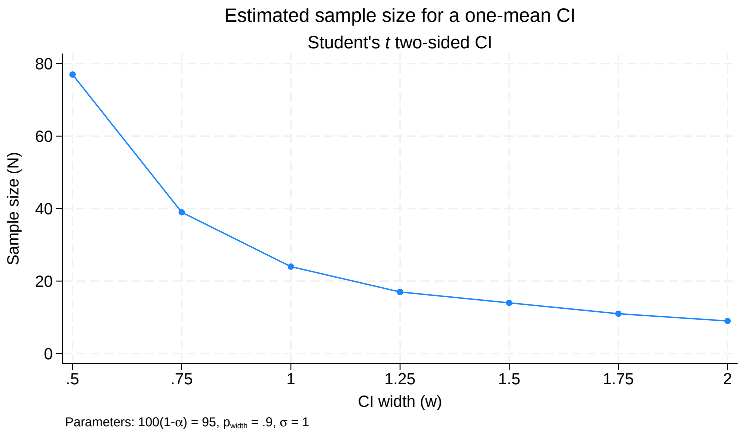


Figure 1.

As expected, sample size decreases as CI width increases.

The default axis labels include column labels and column symbols in parentheses. The labels can be changed, as we show in example 5. The values of constant parameters are displayed in the note titled "Parameters": confidence level $100(1-\alpha)$ is 95, probability of CI width p_{width} is 0.9, and standard deviation σ is 1.

In addition to varying CI width, we may compute sample sizes for different standard deviations.

```
. ciwidth onemean, width(0.5(0.25)2) probwidth(0.9) sd(1 2) graph
```

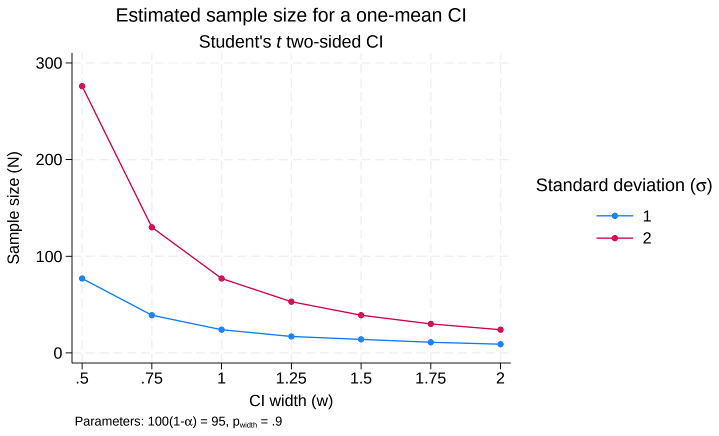


Figure 2.

For a given CI width, the larger the standard deviation, the larger the sample size.

`ciwidth, graph` displays two sample-size curves corresponding to the specified standard deviation values as shown on the legend. The first curve is displayed in navy, and the second curve is displayed in maroon. The default colors of the lines and, in general, the overall look of the graph are determined by the current graph scheme. The scheme used here is `stgcolor`; see [G-2] **set scheme** for details. We also show how to change the default look of the curves in example 6.

We can obtain sample-size curves for varying values of several parameters. `ciwidth, graph` plots a separate curve for each unique combination of the values of the parameters (except the parameter used as the x axis) on one plot. Alternatively, you can display curves on separate plots (`by` graphs) or even on separate graphs; see example 4.

If we specify only one CI width in the previous figure, the values of the standard deviation will be plotted on the x axis. You can try this yourself if you would like.

◁

▷ Example 2: CI precision curves

Instead of sample-size curves, we can plot estimated CI widths for a range of sample-size values to get an idea of how the sample size affects the CI precision.

```
. ciwidth onemean, n(10(2)40) probwidth(0.9) graph
```

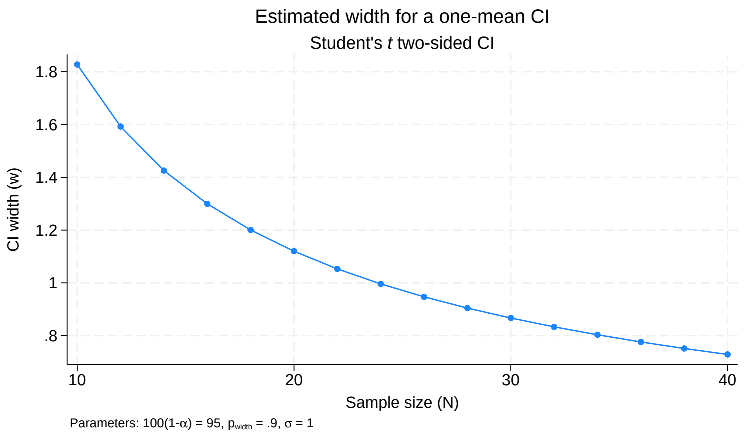


Figure 3.

The CI width decreases as the sample size increases. The larger the sample size, the more precise the CI.

This graph has the same overall look as figure 1, except CI width is plotted on the y axis and sample size is plotted on the x axis.

We may want to investigate how other study parameters, such as the standard deviation, affect the CI precision.

```
. ciwidth onemean, n(10(2)40) probwidth(0.9) sd(1 2) graph
```

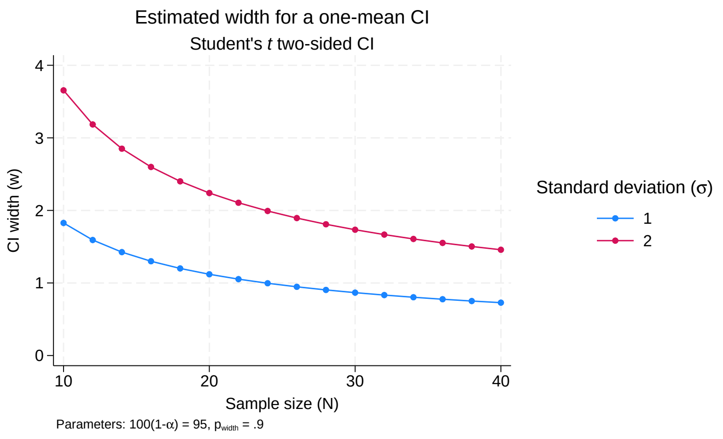


Figure 4.

The larger the standard deviation, the larger the CI width and thus the lower the CI precision.

When multiple study parameters each contain multiple values, as in the above figure, the default x axis for sample-size curves is the CI width, provided that the CI width parameter varies, and for precision curves is the sample size, provided it varies. You can plot a different parameter on the x axis, for instance the standard deviation; in example 3, we demonstrate how to change the default axes.

◁

Changing default graph dimensions

So far, we have demonstrated the graphs that `ciwidth, graph` produces by default. In this section, we demonstrate how you can modify the default graphs with `ciwidth, graph()` and its suboptions.

▷ Example 3: Changing default graph axes

The default y axis corresponds to the computed study parameter—sample size for sample-size determination and CI width for precision determination. You would rarely need to change the dimensions when computing these parameters. On the other hand, probabilities are also plotted by default on the y axis when computing probability of CI width, but because of the likely limited range for this parameter, you may want to change the dimensions for these computations.

In figure 4, by default, the CI width is plotted against varying values of the sample size, and a separate curve is plotted for each of the varying values of the standard deviation. We can change the default x axis by specifying the `xdimension()` suboption (abbreviated to `x()`) within `ciwidth`'s `graph()` option. In this example, we specified fewer sample sizes and more standard deviations to obtain a more readable graph.

```
. ciwidth onemean, n(10 25 40) probwidth(0.9) sd(1(0.2)2) graph(x(sd))
```

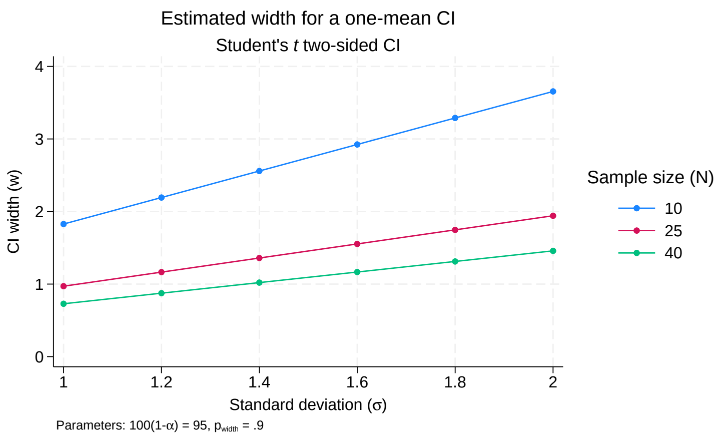


Figure 5.

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

When the `xdimension()` suboption is specified, the x axis is replaced with the specified column, and the column corresponding to the default x axis is used as a plot dimension.

◁

▷ Example 4: By graphs and multiple graphs

In figure 4, we plotted multiple CI precision curves corresponding to different standard deviation values on one graph. Alternatively, we can produce a separate plot for each of the standard deviation values by specifying the column `sd` in the `bydimension()` suboption (abbreviated to `by()`) within `ciwidth`'s `graph()` option.

```
. ciwidth onemean, n(10(2)40) probwidth(0.9) sd(1 2) graph(by(sd))
```

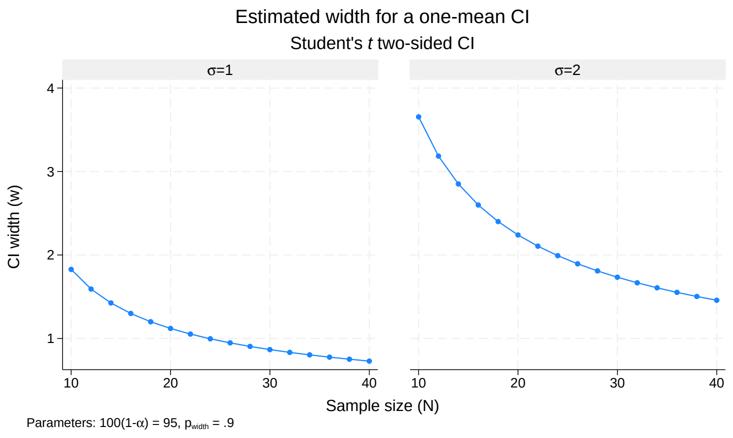


Figure 6.

For examples of how to modify the look of a `by` graph, see example 7.

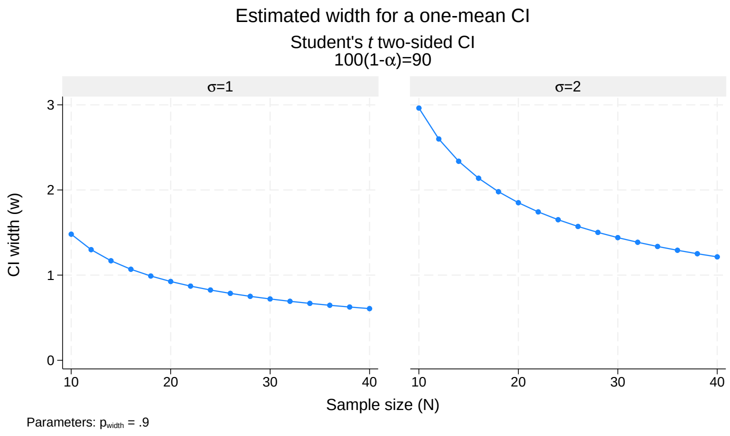
In the presence of many varying parameters, even `by` graphs may look crowded. In this case, you may consider producing multiple `by` graphs. In the example above, suppose that we also want to vary the confidence level. We add the `level(90 95)` option to the previous command.

```
. ciwidth onemean, n(10(2)40) probwidth(0.9) sd(1 2) level(90 95) graph(by(sd))
  (output omitted )
```

The above command produces a graph containing two `by` graphs. Each `by` graph contains two plots, with each plot corresponding to unique values of the confidence level. We leave this for you to verify.

Instead, we can request that a separate graph be produced for each of the confidence levels by specifying the `graphdimension(level)` suboption (abbreviated to `graph()`) within `ciwidth`'s `graph()` option:

```
. ciwidth onemean, n(10(2)40) probwidth(0.9) sd(1 2) level(90 95)
> graph(by(sd) graph(level))
```



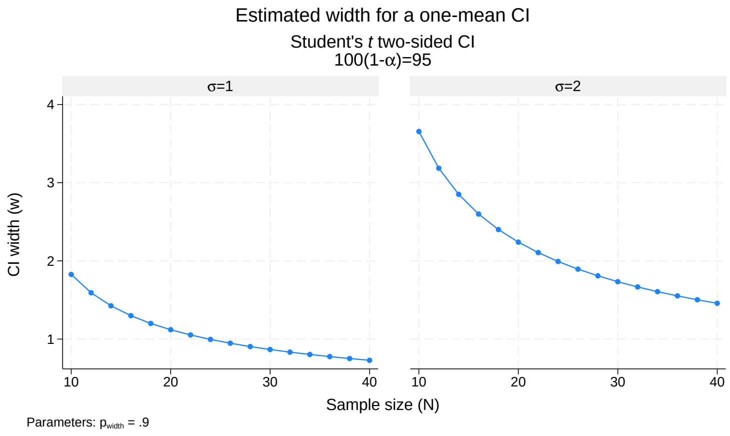


Figure 7.

◁

Changing the look of graphs

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

▷ Example 5: Modifying axis labels

Rather than placing ticks and labels at equally spaced values, as in figure 3, we can request that ticks and labels be placed on the y and x axes for each distinct value.

```
. ciwidth onemean, n(10(2)20) probwidth(0.9)
> graph(yvalues xvalues ylabel(, format(%4.3f)))
```

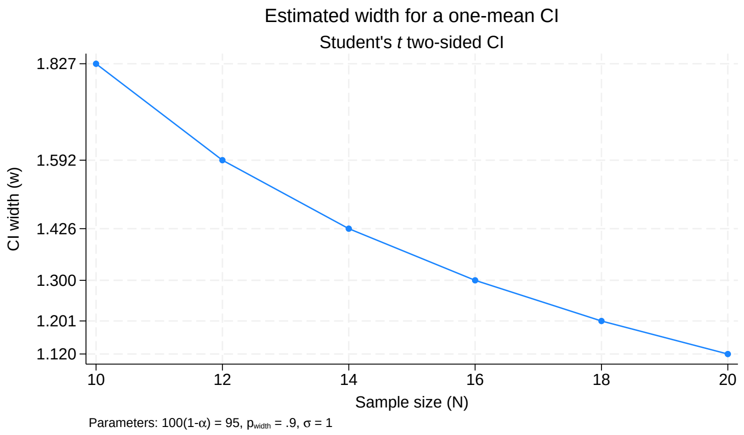


Figure 8.

In this example, to improve readability, we changed the default format of the values on the y axis to show only three decimal points by using `ylabel(, format(%4.3f))`.

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

```
. ciwidth onemean, n(10(2)40) probwidth(0.9) graph(xlabel(30 "Budgeted", add))
```

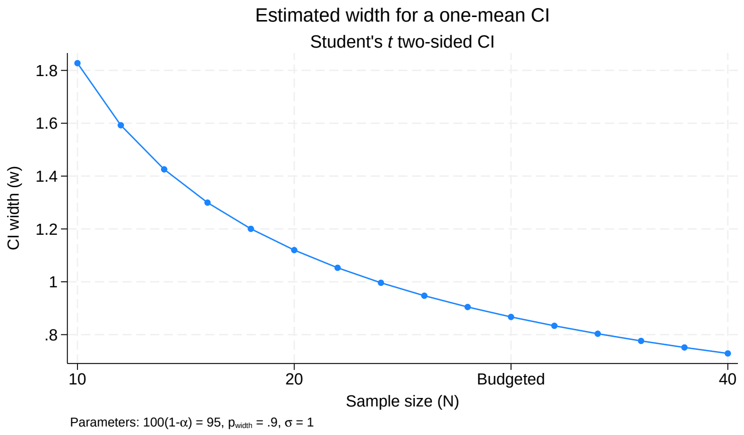


Figure 9.

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

```
. ciwidth onemean, n(10(2)20) probwidth(0.9) graph(ytitle("CI width")
> xtitle("Sample size") title("Estimated width") subtitle("") note(""))
```

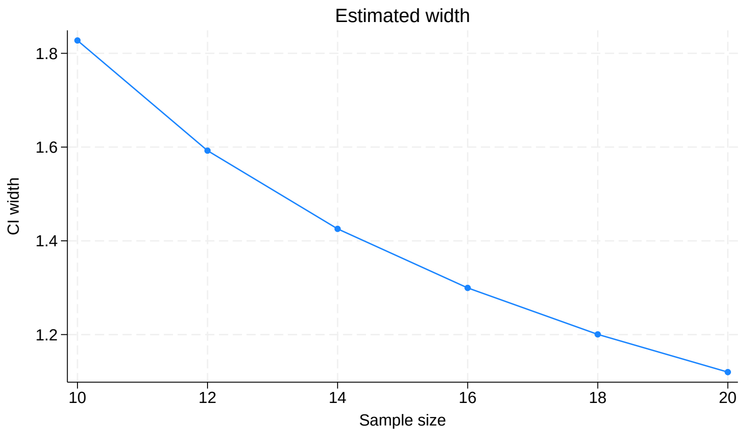


Figure 10.

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

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

For example, change the default labels of the CI width and sample-size columns from figure 4 to “CI width” and “N”, respectively, as follows:

```
. ciwidth onemean, n(10(2)40) sd(1 2) probwidth(0.9)
> graph(collabels(N "N" width "CI width"))
```

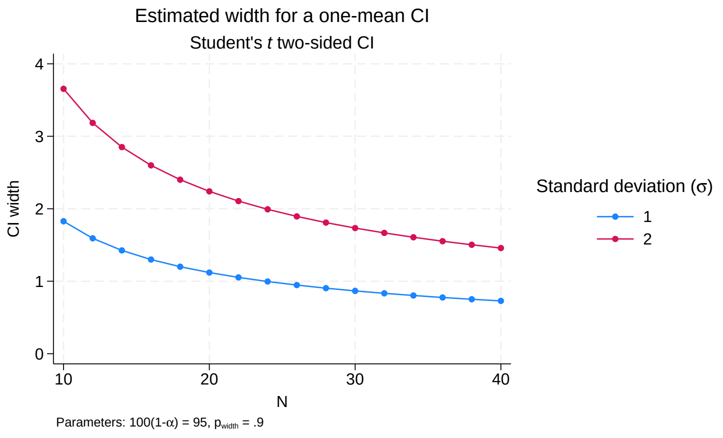


Figure 11.

When plotting multiple curves, by default, only the numeric values are used to label each plot in the legend. This is the simplest form of a label, but we can be more explicit and label each plot with the parameter symbol, an equal sign, and the value it takes on for each plot. We do this with the nosimplelabels suboption (abbreviated to nosimple) below. And to save space, we can suppress the legend title, which is included by default, by specifying an empty string.

```
. ciwidth onemean, n(10(2)40) sd(1 2) probwidth(0.9)
> graph(nosimple legend(title("")))
```

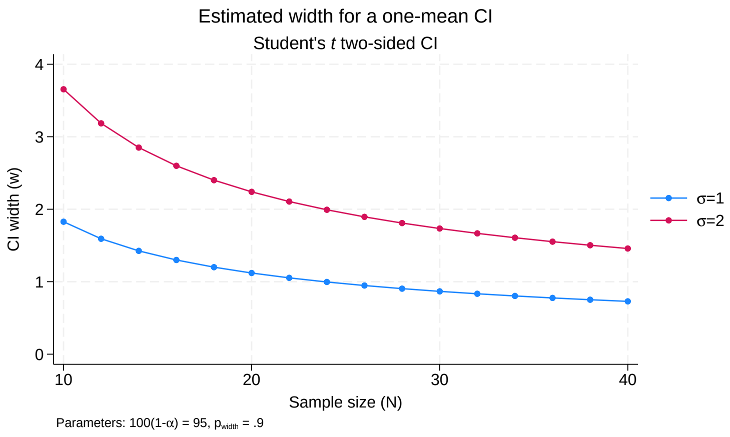


Figure 12.

The nosimple option will use an equal sign as the separator, but if we wanted to use another symbol (say, a colon), we could modify the command to something like the following:

```
ciwidth onemean, n(10(2)40) sd(1 2) probwidth(0.9) ///
 graph(nosimple legend(title("")) eqsep(" : "))
```

◁

▷ Example 6: Plot options

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

We can label each data point on the graph with its corresponding sample-size value by specifying mlabel() within the plotopts() suboption, as shown below. We use mlabpos() to place the marker labels at the one o'clock position. See [G-3] ***marker_label_options*** for more details about these options.

```
. ciwidth onemean, width(0.5(0.5)2) probwidth(0.9)
> graph(plotopts(mlabel(N) mlabpos(1)))
```

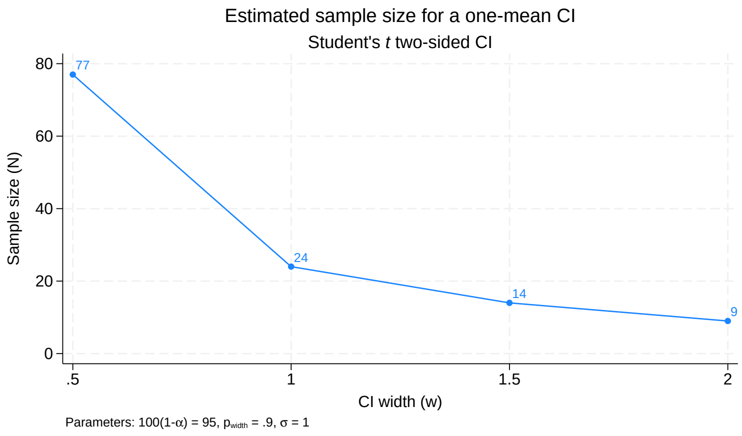


Figure 13.

For plots containing multiple curves, such as figure 4, the `plotopts()` suboption controls the look of all curves. For example, we can change the marker symbol from the default solid circle to a solid triangle.

```
. ciwidth onemean, n(10(2)40) probwidth(0.9) sd(1 2)
> graph(plotopts(msymbol(T)))
```

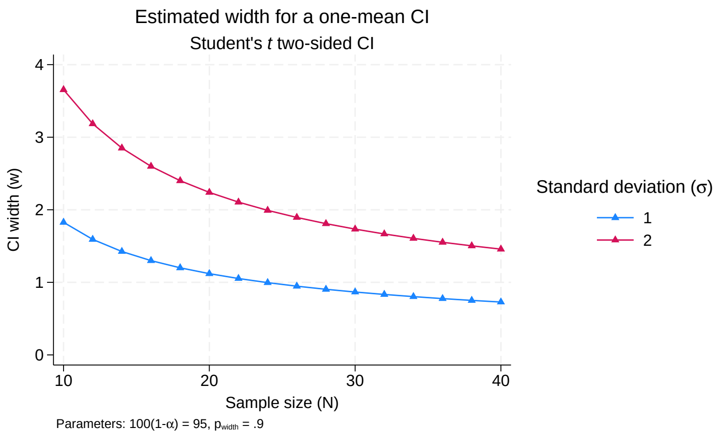


Figure 14.

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

```
. ciwidth onemean, n(10(2)40) probwidth(0.9) sd(1 2) level(90 95)
> graph(plot3opts(color(navy)) plot4opts(color(maroon)))
```

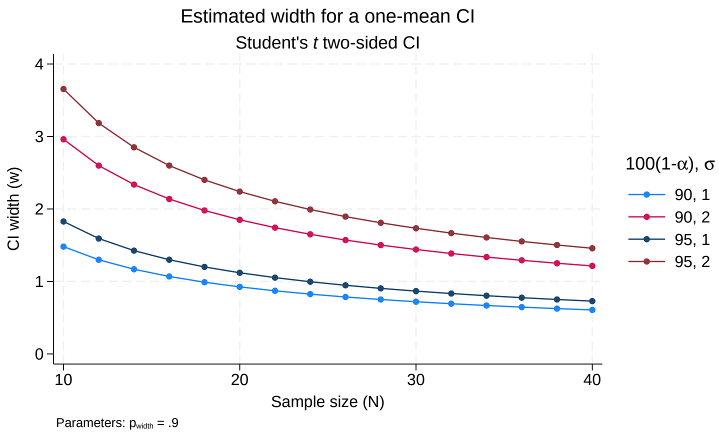


Figure 15.

◁

▷ Example 7: Modifying the look of by-graphs

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

In figure 6, we plotted the CI width for two standard deviation values using the same y axis. To allow the scales of the two `by` graphs to differ, we specify `yrescale` within the `byopts()` suboption.

```
. ciwidth onemean, n(10(2)40) probwidth(0.9) sd(1 2)
> graph(by(sd) byopts(yrescale))
```

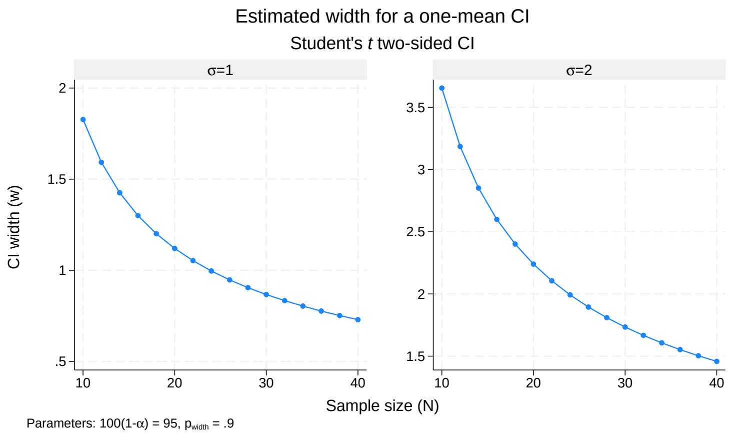


Figure 16.

We can also specify suboptions within `byopts()` to change the overall graph title and subtitle.

```
. ciwidth onemean, n(10(2)40) probwidth(0.9) sd(1 2)
> graph(by(sd) byopts(yrescale title("Width vs sample size") subtitle("")))
```

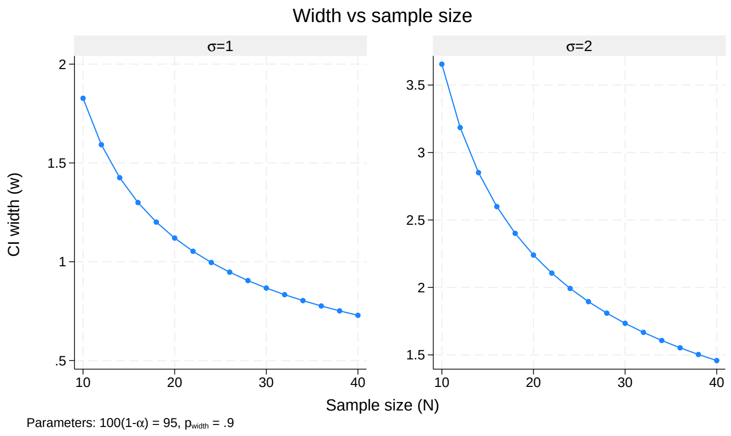


Figure 17.

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

We could vary the probability of CI width by typing

```
ciwidth onemean, n(10(2)40) probwidth(0.7 0.9) sd(1 2)  ///
  graph(by(sd) byopts(yrescale title("Width vs sample size") subtitle("")))
```

so that a legend indicating which line corresponds to each probability of CI width would appear on the right side of the graph. In by-graphs, this legend can make each graph appear tall and narrow. We can then include `scheme(stcolor_alt)` within the `graph()` option to create this graph using a scheme that puts the legend at the bottom, allowing more width for each graph.

```
. ciwidth onemean, n(10(2)40) probwidth(0.7 0.9) sd(1 2)
> graph(by(sd) scheme(stcolor_alt)
> byopts(yrescale title("Width vs sample size") subtitle("")))
```

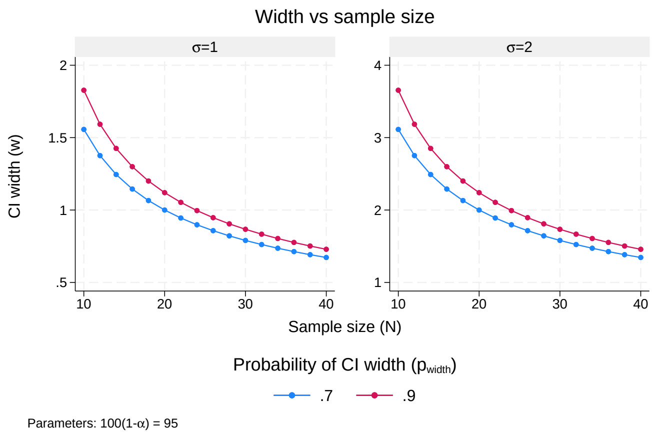


Figure 18.

◁

Parallel plots

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

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

```
. ciwidth onemean, width(0.5(0.1)2) probwidth(0.9) sd(1(0.1)1.9)
> parallel graph(x(width sd) horizontal nosimplelabels ytitle(""))
```

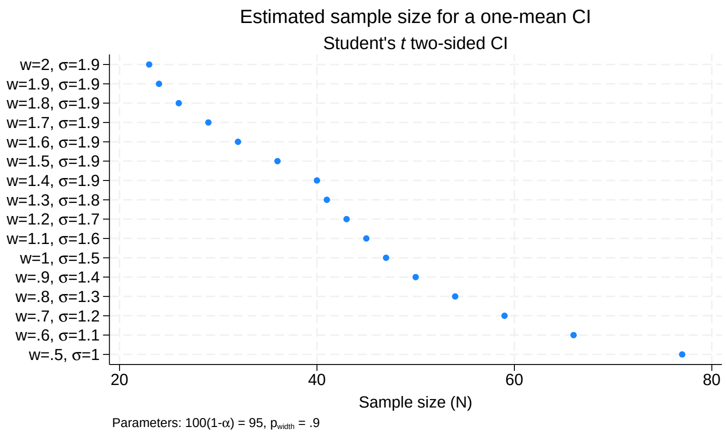


Figure 19.

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

Also see

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

[PSS-3] **ciwidth, table** — Produce table of results from the ciwidth command