**ciwidth, table** — Produce table of results from the ciwidth command



## Description

`ciwidth, table` displays results in a tabular format. `table` is implied if any of the `ciwidth` command's arguments or options contain more than one element. The `table` option is useful if you are producing graphs and would like to see the table as well or if you are producing results one case at a time using a loop and wish to display results in a table. The `notable` option suppresses table results; it is implied with the graphical output of `ciwidth, graph`; see [PSS-3] **ciwidth, graph**.

## Quick start

Sample size required to achieve a target CI width of 1 with a 90% probability for a one-sample mean, in tabular format

```
ciwidth onemean, width(1) probwidth(0.9) table
```

Same as above, but change column labels of `N` and `sd` to `Sample size` and `Std. dev.`, respectively

```
ciwidth onemean, width(1) probwidth(0.9) ///
  table(, labels(N "Sample size" sd "Std. dev."))
```

## Menu

Statistics > Power, precision, and sample size

# Syntax

*Produce default table*

```
ciwidth ..., table ...
```

*Suppress table*

```
ciwidth ..., notable ...
```

*Produce custom table*

```
ciwidth ..., table([colspec] [, tableopts]) ...
```

where *colspec* is

```
column[ :label] [column[ :label] [...]]
```

*column* is one of the columns defined below, and *label* is a column label (may contain quotes and compound quotes).

| *tableopts* | Description |
|---|---|
| Table | |
| `add` | add *column*s to the default table |
| `labels(`*labspec*`)` | change default labels for specified columns; default labels are column names |
| `widths(`*widthspec*`)` | change default column widths; default is specific to each column |
| `formats(`*fmtspec*`)` | change default column formats; default is specific to each column |
| `noformat` | do not use default column formats |
| `separator(#)` | draw a horizontal separator line every # lines; default is `separator(0)`, meaning no separator lines |
| `divider` | draw divider lines between columns |
| `byrow` | display rows as computations are performed; seldom used |
| `noheader` | suppress table header; seldom used |
| `continue` | draw a continuation border in the table output; seldom used |

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

`noheader` and `continue` are not shown in the dialog box.

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

By default, the following columns are displayed:

`level`, `width`, and `N` are always displayed;

`N1` and `N2` are displayed for two-sample methods;

additional columns specific to each `ciwidth` method may be displayed.

## Suboptions

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

Table

`add` requests that the columns specified in *colspec* be added to the default table. The columns are added to the end of the table.

`labels(`*labspec*`)` specifies the labels to be used in the table for the specified columns. *labspec* is

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

`labels()` takes precedence over the specification of column labels in *colspec*.

`widths(`*widthspec*`)` specifies column widths. The default values are the widths of the default column formats plus one. If the `noformat` option is used, the default for each column is nine. The column widths are adjusted to accommodate longer column labels and larger format widths. *widthspec* is either a list of values including missing values (*numlist*) or

*column* # [ *column* # [ ... ] ]

For the value-list specification, the number of specified values may not exceed the number of columns in the table. A missing value (`.`) may be specified for any column to indicate the default width. If fewer widths are specified than the number of columns in the table, the last width specified is used for the remaining columns.

The alternative column-list specification provides a way to change widths of specific columns.

`formats(`*fmtspec*`)` specifies column formats. The default is `%7.0gc` for integer-valued columns and `%7.4g` for real-valued columns. *fmtspec* is either a string value-list of formats that may include empty strings or a column list:

*column* `"`*fmt*`"` [ *column* `"`*fmt*`"` [ ... ] ]

For the value-list specification, the number of specified values may not exceed the number of columns in the table. An empty string (`""`) may be specified for any column to indicate the default format. If fewer formats are specified than the number of columns in the table, the last format specified is used for the remaining columns.

The alternative column-list specification provides a way to change formats of specific columns.

`noformat` requests that the default formats not be applied to the column values. If this suboption is specified, the column values are based on the column width.

`separator(#)` specifies how often separator lines should be drawn between rows of the table. The default is `separator(0)`, meaning that no separator lines should be displayed.

`divider` specifies that divider lines be drawn between columns. The default is no dividers.

`byrow` specifies that table rows be displayed as computations are performed. By default, the table is displayed after all computations are performed. This suboption may be useful when the computation of each row of the table takes a long time.

The following suboptions are available but are not shown in the dialog box:

`noheader` prevents the table header from displaying. This suboption is useful when the command is issued repeatedly, such as within a loop.

`continue` draws a continuation border at the bottom of the table. This suboption is useful when the command is issued repeatedly, such as within a loop.

# Remarks and examples

Remarks are presented under the following headings:



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

## Using ciwidth, table

If you specify the `table` option or include more than one element in command arguments or in options allowing multiple values, the `ciwidth` command displays results in a tabular form. If desired, you can suppress the table by specifying the `notable` option. The `table` option is useful if you are producing graphical output or if you are producing results one case at a time, such as within a loop, and wish to display results in a table; see example 4 below.

Each method specified with the `ciwidth` command has its own default table. Among the columns that are always included in the default table are confidence level (`level`), CI width (`width`), and total sample size (`N`).

Depending on the method and study design, additional columns are also included by default. For example, `ciwidth onemean` has an additional column for standard deviation.

You can build your own table by specifying the columns and, optionally, their labels in the `table()` option. You can also add columns to the default table by specifying `add` within `ciwidth`'s `table()` option. The columns are displayed in the order they are specified. Each method provides its own list of supported columns; see the description of the `table()` option for each method. You can further customize the table by specifying various suboptions within `ciwidth`'s `table()` option.

The default column labels are the column names. You can provide your own column labels in *colspec* or by specifying `table()`'s suboption `labels()`. Labels containing spaces should be enclosed in quotes, and labels containing quotes should be enclosed in compound quotes. The `labels()` suboption is useful for changing the labels of existing columns; see example 2 below for details.

The default formats are `%7.4g` for real-valued columns and `%7.0gc` for integer-valued columns. If the `noformat` suboption is specified, the default column widths are nine characters. You can use `formats()` to change the default column formats and `widths()` to change the default column widths. The `formats()` and `widths()` suboptions provide two alternative specifications, a value-list specification or a column-list specification. The value-list specification accepts a list of values—strings for formats and numbers for widths—corresponding to each column of the displayed table. Empty strings (`""`) for formats and missing values (`.`) for widths are allowed and denote the default values. It is an error to specify more values than the number of displayed columns. If fewer values are specified, then the last value specified is used for the remaining columns. The column-list specification includes a list of pairs containing a column name followed by the corresponding value of the format or width. This specification is useful if you want to modify the formats or the widths of only selected columns. For column labels or formats exceeding the default column width, the widths of the respective columns are adjusted to accommodate the column labels and the specified formats.

If you specify the `noformat` suboption, the default formats are ignored, and the format of a column is determined by the column width: if the column width is #, the displayed format is `%`$(\#-2)$`.0g`. For example, if the column width is 9, the displayed format is `%7.0g`.

You may further customize the look of the table by using `separator(#)` to include separator lines after every # lines and by using the `divider` suboption to include divider lines between columns.

The `noheader` and `continue` suboptions are useful when you are building your own table within a loop; see example 4 in *Custom tables*.

In what follows, we demonstrate the default and custom tables of the results from PrSS analysis for a one-mean CI and a one-variance CI; see [PSS-3] **ciwidth onemean** and [PSS-3] **ciwidth onevariance**.

## Default tables

If there is only one set of results, the `ciwidth` command displays those results as text. When the `ciwidth` command has multiple sets of results, they are automatically displayed in a table. You can also specify the `table` option at any time to request that results be displayed in a table.

The displayed columns are specific to the chosen method of analysis and to the options specified with the command. The columns that always appear in the table include the confidence level (`level`), CI width (`width`), and total sample size (`N`).

### ▷ Example 1: Default tables from ciwidth onemean

Suppose we want to explore the required sample size to achieve a certain precision for a one-mean CI. Below we estimate the required sample size for obtaining the target CI widths no larger than 1, 2, and 3 with a probability of CI width of 0.9. See [PSS-3] **ciwidth onemean** for details.

```
. ciwidth onemean, width(1 2 3) probwidth(0.9) sd(2)
Performing iteration ...
Estimated sample size for a one-mean CI
Student's t two-sided CI
```

| level | N | Pr_width | width | sd |
|---|---|---|---|---|
| 95 | 77 | .9 | 1 | 2 |
| 95 | 24 | .9 | 2 | 2 |
| 95 | 14 | .9 | 3 | 2 |

As we mentioned earlier, the `level`, `width`, and `N` columns are displayed in the default table. Column `Pr_width` is also displayed in the default table whenever `probwidth()` is specified. The `ciwidth onemean` command additionally displays the standard deviation column.

◁

## Modifying default tables

We can modify labels, widths, and formats of the default columns by specifying the corresponding suboptions within the `table()` option. We can also add columns to the default table by using `table()`'s suboption `add`.

### ▷ Example 2: Modifying default tables from ciwidth onemean

We can change the default labels of all or selected columns by using the `labels()` suboption within `ciwidth`'s `table()` option. For example, we can change the labels of the sample-size columns and standard deviation columns of the first table in example 1 to "Sample size" and "Std. dev.", respectively.

```
. ciwidth onemean, width(1 2 3) probwidth(0.9) sd(2)
> table(, labels(N "Sample size" sd "Std. dev."))
Performing iteration ...
Estimated sample size for a one-mean CI
Student's t two-sided CI
```

| level | Sample size | Pr_width | width | Std. dev. |
|---|---|---|---|---|
| 95 | 77 | .9 | 1 | 2 |
| 95 | 24 | .9 | 2 | 2 |
| 95 | 14 | .9 | 3 | 2 |

We can also change default column formats and widths by using the `formats()` and `widths()` suboptions.

```
. ciwidth onemean, width(1 2 3) probwidth(0.9) sd(2)
> table(, labels(N "Sample size" sd "Std. dev.") widths(N 14 sd 14)
> formats(width "%7.5f"))
Performing iteration ...
Estimated sample size for a one-mean CI
Student's t two-sided CI
```

| level | Sample size | Pr_width | width | Std. dev. |
|---|---|---|---|---|
| 95 | 77 | .9 | 1.00000 | 2 |
| 95 | 24 | .9 | 2.00000 | 2 |
| 95 | 14 | .9 | 3.00000 | 2 |

For this table, we changed the default column widths of the sample-size and standard deviation columns to 14. We also changed the format of the width column from the default, `%7.4g`, to `%7.5f`.

◁

### ▷ Example 3: Modifying default tables from ciwidth onevariance

We can also add columns to the default table by using `table()`'s suboption `add`. In the `ciwidth onevariance` example below, the default columns are `level`, `N`, `Pr_width`, `width`, and `v`.

```
. ciwidth onevariance 1, width(1 2) probwidth(0.9)
Performing iteration ...
Estimated sample size for a one-variance CI
Chi-squared two-sided CI
```

| level | N | Pr_width | width | v |
|---|---|---|---|---|
| 95 | 57 | .9 | 1 | 1 |
| 95 | 23 | .9 | 2 | 1 |

We can also add the standard deviation column, `s`, to the table:

```
. ciwidth onevariance 1, width(1 2) probwidth(0.9) table(s, add)
Performing iteration ...
Estimated sample size for a one-variance CI
Chi-squared two-sided CI
```

| level | N | Pr_width | width | v | s |
|---|---|---|---|---|---|
| 95 | 57 | .9 | 1 | 1 | 1 |
| 95 | 23 | .9 | 2 | 1 | 1 |

◁

## Custom tables

You can use the `table()` option to build custom tables, with the columns you want in the order you want. You can also build a table within a `foreach` or `forvalues` loop.

### ▷ Example 4: Building table using a loop

Some options of `ciwidth` commands may not allow the *numlist* specification. In this case, you can build a table manually by using a loop with either `foreach` (see [P] **foreach**) or `forvalues` (see [P] **forvalues**). One way to do this is to write a program that loops over parameters of interest. We demonstrate a program that loops over varying values of the variance of `ciwidth onevariance`. You can easily adapt this program to meet your needs.

```
program dotable
        args var
        numlist "`var'"                  // expand the numeric list in macro var
        local var "`r(numlist)'"
        local nvals : list sizeof var
        local i 1
        foreach val of local var {       // loop over numeric values in var
                if (`i'==1) {
                        ciwidth onevariance `val', width(2) probwidth(0.9) ///
        table(, continue)
                }
                else if (`i'<`nvals') {
                        ciwidth onevariance `val', width(2) probwidth(0.9) ///
      table(, noheader continue) notitle
                }
                else {
                        ciwidth onevariance `val', width(2) probwidth(0.9) ///
      table(, noheader) notitle
                }
                local ++i
      }
end
```

The `dotable` program accepts one argument, `var`, which may contain one or more numeric values of the variance specified as *numlist*. The program uses combinations of `continue`, `noheader`, and `notitle` to display a table. The first call to `ciwidth onevariance` requests that the table be displayed without the bottom line by specifying the `continue` suboption within `table()`. The subsequent calls (except the last) specify the `continue` suboption, the `notitle` option with `ciwidth onevariance`, and `noheader` within the `table()` option to request that neither the output before the table nor the table header be displayed. The last call omits the `continue` suboption so that the bottom line is displayed.

As a result, we obtain the following table:

```
. dotable "1(0.2)2"
Performing iteration ...
Estimated sample size for a one-variance CI
Chi-squared two-sided CI
```

| level | N | Pr_width | width | v |
|---|---|---|---|---|
| 95 | 23 | .9 | 2 | 1 |
| 95 | 29 | .9 | 2 | 1.2 |
| 95 | 35 | .9 | 2 | 1.4 |
| 95 | 41 | .9 | 2 | 1.6 |
| 95 | 49 | .9 | 2 | 1.8 |
| 95 | 57 | .9 | 2 | 2 |

## Stored results

`ciwidth, table` stores the following in `r()` in addition to other results stored by `ciwidth`:

Scalars

| | |
|---|---|
| `r(separator)` | number of lines between separator lines in the table |
| `r(divider)` | `1` if `divider` is requested in the table, `0` otherwise |

Macros

| | |
|---|---|
| `r(columns)` | displayed table columns |
| `r(labels)` | table column labels |
| `r(widths)` | table column widths |
| `r(formats)` | table column formats |

Matrices

| | |
|---|---|
| `r(pss_table)` | table of results |

## Also see

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

[PSS-3] **ciwidth, graph** — Graph results from the ciwidth command