by_option — Option for repeating graph command

Syntax

by_option  Description
by(varlist[ , byopts ])  repeat for by-groups

by() is merged-implicit; see [G-4] concept: repeated options.

byopts  Description

total  add total group
missing  add missing groups
colfirst  display down columns
rows(#)  display in # rows or # columns
cols(#)  display in # rows or # columns
holes(numlist)  positions to leave blank
iscale([*]#)  size of text and markers

compact  synonym for style(compact)
style(bystyle)  overall style of presentation

[no] edgelabel  label x axes of edges
[no] rescale  separate y and x scales for each group
[no] yrescale  separate y scale for each group
[no] xrescale  separate x scale for each group
[no] iyaxes  show individual y axes
[no] ixaxes  show individual x axes
[no] iytick  show individual y-axes ticks
[no] ixtick  show individual x-axes ticks
[no] iylabel  show individual y-axes labels
[no] ixlabel  show individual x-axes labels
[no] iytitle  show individual y-axes titles
[no] ixtitle  show individual x-axes titles
imargin(marginstyle)  margin between graphs

legend_options  show legend and placement of legend
title_options  overall titles
region_options  overall outlining, shading, and aspect

The title_options and region_options on the command on which by() is appended
will become the titles and regions for the individual by-groups.
**by_option — Option for repeating graph command**

### Description

Option `by()` repeats the `graph` command for each value of `varlist` and arrays the resulting individual graphs into one graph. `varlist` may be a numeric or a string variable.

### Option

`by(varlist[, byopts])` specifies that the `graph` command be repeated for each unique set of values of `varlist` and that the resulting individual graphs be arrayed into one graph.

#### byopts

- **total** specifies that, in addition to the graphs for each by-group, a graph be added for all by-groups combined.
- **missing** specifies that, in addition to the graphs for each by-group, graphs be added for missing values of `varlist`. Missing is defined as `.`, `.a`, `.z` for numeric variables and `""` for string variables.
- **colfirst** specifies that the individual graphs be arrayed down the columns rather than across the rows. That is, if there were four groups, the graphs would be displayed

<table>
<thead>
<tr>
<th>default</th>
<th>colfirst</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2</td>
<td>1 3</td>
</tr>
<tr>
<td>3 4</td>
<td>2 4</td>
</tr>
</tbody>
</table>

- **rows(#) and cols(#)** are alternatives. They specify that the resulting graphs be arrayed as # rows and however many columns are necessary, or as # columns and however many rows are necessary. The default is `cols(c), c = ceil(sqrt(G))` where `G` is the total number of graphs to be presented and `ceil()` is the function that rounds nonintegers up to the next integer. For instance, if four graphs are to be displayed, the result will be presented in a $2 \times 2$ array. If five graphs are to be displayed, the result will be presented as a $2 \times 3$ array because `ceil(sqrt(5))==3`.

- **cols(#) may be specified as larger or smaller than c; r will be the number of rows implied by c. Similarly, rows(#) may be specified as larger or smaller than r.**

- **holes(numlist)** specifies which positions in the array are to be left unfilled. Consider drawing a graph with three groups and assume that the three graphs are being displayed in a $2 \times 2$ array. By default, the first group will appear in the graph at (1,1), the second in the graph at (1,2), and the third in the graph at (2,1). Nothing will be displayed in the (2,2) position. Specifying `holes(3)` would cause position (2,1) to be left blank, so the third group would appear in (2,2).

  The numbers that you specify in `holes()` correspond to the position number,

<table>
<thead>
<tr>
<th>1 2 3</th>
<th>1 2 3 4</th>
<th>1 2 3 4 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 4</td>
<td>4 5 6</td>
<td>5 6 7 8</td>
</tr>
<tr>
<td>7 8 9</td>
<td>9 10 11</td>
<td>11 12 13</td>
</tr>
</tbody>
</table>
  | 12 14 15 16 | 16 17 18 19 20 | 21 22 23 24 25 | etc.
The above is the numbering when \texttt{colfirst} is not specified. If \texttt{colfirst} is specified, the positions are transposed:

\begin{verbatim}
  1 3  1 4  1 5  9 13  1 6  11 16 21
  2 4  2 5  2 6 10 14  2 7 12 17 22
  3 6  3 7  3 8 11 15  3 9 14 19 24
  4 8  4 9  4 10 15 19 24
  5 10 15 20 25 etc.
\end{verbatim}

\texttt{iscale(#) and iscale(*#)} specify a size adjustment (multiplier) to be used to scale the text and markers.

By default, \texttt{iscale()} gets smaller and smaller the larger is $G$, the number of by-groups and hence the number of graphs presented. The default is parameterized as a multiplier $f(G)$—$0 < f(G) < 1, f'(G) < 0$—that is used to multiply msize(), \{ y | x \} label(), labsize()), and the like. The size of everything except the overall titles, subtitles, captions, and notes is affected by \texttt{iscale()}.

If you specify \texttt{iscale(#)}, the number you specify is substituted for $f(G)$. \texttt{iscale(1)} means text and markers should appear at the same size as they would were each graph drawn separately. \texttt{iscale(.5)} displays text and markers at half that size. We recommend you specify a number between 0 and 1, but you are free to specify numbers larger than 1.

If you specify \texttt{iscale(*#)}, the number you specify is multiplied by $f(G)$ and that product is used to scale text and markers. \texttt{iscale(*1)} is the default. \texttt{iscale(*1.2)} means text and markers should appear 20\% larger than \texttt{graph, by() would usually choose}. \texttt{iscale(*.8)} would make them 20\% smaller.

\texttt{compact} is a synonym for \texttt{style(compact)}. It makes no difference which you type. See the description of the \texttt{style()} option below, and see \textit{By-styles} under Remarks and examples.

\texttt{style(bystyle)} specifies the overall look of the by-graphs. The style determines whether individual graphs have their own axes and labels or if instead the axes and labels are shared across graphs arrayed in the same row or in the same column, how close the graphs are to be placed to each other, etc. The other options documented below will allow you to change the way the results are displayed, but the \texttt{bystyle} specifies the starting point.

You need not specify \texttt{style()} just because there is something you want to change. You specify \texttt{style()} when another style exists that is exactly what you desire or when another style would allow you to specify fewer changes to obtain what you want.

See \texttt{[G-4] bystyle} for a list of by-style choices. The \texttt{byopts} listed below modify the by-style:

\texttt{edgelabel} and \texttt{noedgelabel} specify whether the last graphs of a column that do not appear in the last row are to have their \texttt{x} axes labeled. See \textit{Labeling the edges} under Remarks and examples below.

\texttt{rescale}, \texttt{yrescale}, and \texttt{xrescale} (and \texttt{norescale}, \texttt{noyrescale}, and \texttt{noxrescale}) specify that the scales of each graph be allowed to differ (or forced to be the same). Whether \texttt{X} or \texttt{noX} is the default is determined by \texttt{style()}. Usually, \texttt{noX} is the default and \texttt{rescale}, \texttt{yrescale}, and \texttt{xrescale} are the options. By default, all the graphs will share the same scaling for their axes. Specifying \texttt{yrescale} will allow the \texttt{y} scales to differ across the graphs, specifying \texttt{xrescale} will allow the \texttt{x} scales to differ, and specifying \texttt{rescale} is equivalent to specifying \texttt{yrescale} and \texttt{xrescale}.

\texttt{iyaxes} and \texttt{ixaxes} (and \texttt{noiyaxes} and \texttt{noixaxes}) specify whether the \texttt{y} axes and \texttt{x} axes are to be displayed with each graph. The default with most styles and schemes is to place \texttt{y} axes on the leftmost graph of each row and to place \texttt{x} axes on the bottommost graph of each column. The \texttt{y} and \texttt{x} axes include the default ticks and labels but exclude the axes titles.
by_option — Option for repeating graph command

iytick and ixtick (and noiytick and noixtick) are seldom specified. If you specified iyaxis and then wanted to suppress the ticks, you could also specify noiytick. In the rare event where specifying iyaxis did not result in the ticks being displayed (because of how the style or scheme works), specifying iytick would cause the ticks to be displayed.

iylabel and ixlabel (and noiylabel and noixlabel) are seldom specified. If you specified iyaxis and then wanted to suppress the axes labels, you could also specify noiylabel. In the rare event where specifying iyaxis did not result in the labels being displayed (because of how the style or scheme works), specifying iylabel would cause the labels to be displayed.

iytitle and ixtitle (and noiytitle and noixtitle) are seldom specified. If you specified iyaxis and then wanted to add the y-axes titles (which would make the graph appear busy), you could also specify iytitle. In the rare event where specifying iyaxis resulted in the titles being displayed (because of how the style or scheme works), specifying noiytitle would suppress displaying the title.

imargin(marginstyle) specifies the margins between the individual graphs.

legend_options used within by() sets whether the legend is drawn and the legend’s placement; see Use of legends with by() below. The legend() option is normally merged-implicit, but when used inside by(), it is unique; see [G-4] concept: repeated options

Remarks and examples

Remarks are presented under the following headings:

Typical use
Placement of graphs
Treatment of titles
by() uses subtitle() with graph
Placement of the subtitle()
by() uses the overall note()
Use of legends with by()
By-styles
Labeling the edges
Specifying separate scales for the separate plots
History

Typical use

One often has data that divide into different groups—person data where the persons are male or female or in different age categories (or both), country data where the countries can be categorized into different regions of the world, or, as below, automobile data where the cars are foreign or domestic. If you type
by_option — Option for repeating graph command

you obtain a scatterplot of mpg versus weight. If you add by(foreign) as an option, you obtain two graphs, one for each value of foreign:

```
. scatter mpg weight, by(foreign)
```
If you add `total`, another graph will be added representing the overall total:

```
.scatter mpg weight, by(foreign, total)
```

Here there were three graphs to be presented and `by()` chose to display them in a $2 \times 2$ array, leaving the last position empty.
Placement of graphs

By default, `by()` places the graphs in a rectangular $R \times C$ array and leaves empty the positions at the end:

<table>
<thead>
<tr>
<th>Number of graphs</th>
<th>Array dimension</th>
<th>Positions left empty</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$1 \times 1$</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>$1 \times 2$</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>$2 \times 2$</td>
<td>$4=(2,2)$</td>
</tr>
<tr>
<td>4</td>
<td>$2 \times 2$</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>$2 \times 3$</td>
<td>$6=(3,3)$</td>
</tr>
<tr>
<td>6</td>
<td>$2 \times 3$</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>$3 \times 3$</td>
<td>$8=(3,2)$ $9=(3,3)$</td>
</tr>
<tr>
<td>8</td>
<td>$3 \times 3$</td>
<td>$9=(3,3)$</td>
</tr>
<tr>
<td>9</td>
<td>$3 \times 3$</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>$3 \times 4$</td>
<td>$11=(3,3)$ $12=(3,4)$</td>
</tr>
<tr>
<td>11</td>
<td>$3 \times 4$</td>
<td>$12=(3,4)$</td>
</tr>
<tr>
<td>12</td>
<td>$3 \times 4$</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>$4 \times 4$</td>
<td>$14=(4,2)$ $15=(4,3)$ $16=(4,4)$</td>
</tr>
<tr>
<td>14</td>
<td>$4 \times 4$</td>
<td>$15=(4,3)$ $16=(4,4)$</td>
</tr>
<tr>
<td>15</td>
<td>$4 \times 4$</td>
<td>$16=(4,4)$</td>
</tr>
<tr>
<td>16</td>
<td>$4 \times 4$</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>$4 \times 5$</td>
<td>$18=(4,3)$ $19=(4,4)$ $20=(4,5)$</td>
</tr>
<tr>
<td>18</td>
<td>$4 \times 5$</td>
<td>$19=(4,4)$ $20=(4,5)$</td>
</tr>
<tr>
<td>19</td>
<td>$4 \times 5$</td>
<td>$20=(4,5)$</td>
</tr>
<tr>
<td>20</td>
<td>$4 \times 5$</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>$5 \times 5$</td>
<td>$22=(5,2)$ $23=(5,3)$ $24=(5,4)$ $25=(5,5)$</td>
</tr>
<tr>
<td>22</td>
<td>$5 \times 5$</td>
<td>$23=(5,3)$ $24=(5,4)$ $25=(5,5)$</td>
</tr>
<tr>
<td>23</td>
<td>$5 \times 5$</td>
<td>$24=(5,4)$ $25=(5,5)$</td>
</tr>
<tr>
<td>24</td>
<td>$5 \times 5$</td>
<td>$25=(5,5)$</td>
</tr>
<tr>
<td>25</td>
<td>$5 \times 5$</td>
<td></td>
</tr>
<tr>
<td>etc.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Options `rows()`, `cols()`, and `holes()` allow you to control this behavior.

You may specify either `rows()` or `cols()`, but not both. In the previous section, we drew

```
.scatter mpg weight, by(foreign, total)
```

and had three graphs displayed in a $2 \times 2$ array with a hole at 4. We could draw the graph in a $1 \times 3$ array by specifying either `rows(1)` or `cols(3)`,

```
.scatter mpg weight, by(foreign total)
```
or we could stay with the $2 \times 2$ array and move the hole to 3,

```
.scatter mpg weight, by(foreign, total holes(3))
```

## Treatment of titles

Were you to type

```
.scatter yvar xvar, title("My title") by(catvar)
```

“My title” will be repeated above each graph. `by()` repeats the entire `graph` command and then arrays the results.

To specify titles for the entire graph, specify the `title_options`—see [G-3] `title_options`—inside the `by()` option:

```
.scatter yvar xvar, by(catvar, title("My title"))
```
by() uses subtitle() with graph

by() labels each graph by using the subtitle() title_option. For instance, in

```
 . scatter mpg weight, by(foreign, total)
```

by() labeled the graphs “Domestic”, “Foreign”, and “Total”. The subtitle “Total” is what by() uses when the total option is specified. The other two subtitles by() obtained from the by-variable foreign.

by() may be used with numeric or string variables. Here foreign is numeric but happens to have a value label associated with it. by() obtained the subtitles “Domestic” and “Foreign” from the value label. If foreign had no value label, the first two graphs would have been subtitled “0” and “1”, the numeric values of variable foreign. If foreign had been a string variable, the subtitles would have been the string contents of foreign.

If you wish to suppress the subtitle, type

```
 . scatter mpg weight, subtitle("") by(foreign, total)
```

If you wish to add “Extra info” to the subtitle, type

```
 . scatter mpg weight, subtitle("Extra info", suffix) by(foreign, total)
```

Be aware, however, that “Extra info” will appear above each graph.

Placement of the subtitle()

You can use subtitle()’s suboptions to control the placement of the identifying label. For instance,

```
 . scatter mpg weight, 
     subtitle(, ring(0) pos(1) nobexpand) by(foreign, total)
```

The result will be to move the identifying label inside the individual graphs, displaying it in the northeast corner of each. Type

```
 . scatter mpg weight, 
     subtitle(, ring(0) pos(11) nobexpand) by(foreign, total)
```

and the identifying label will be moved to the northwest corner.
ring(0) moves the subtitle inside the graph’s plot region and position() defines the location, indicated as clock positions. nobexpand is rather strange, but just remember to specify it. By default, by() sets subtitles to expand to the size of the box that contains them, which is unusual but makes the default-style subtitles look good with shading.

See [G-3] `title_options`.

**by() uses the overall note()**

By default, by() adds an overall note saying “Graphs by . . .”. When you type

```
. scatter yvar xvar, by(catvar)
```

results are the same as if you typed

```
. scatter yvar xvar, by(catvar, note("Graphs by . . ."))
```

If you want to suppress the note, type

```
. scatter yvar xvar, by(catvar, note(""))
```

If you want to change the overall note to read “My note”, type

```
. scatter yvar xvar, by(catvar, note("My note"))
```

If you want to add your note after the default note, type

```
. scatter yvar xvar, by(catvar, note("My note", suffix))
```

**Use of legends with by()**

If you wish to modify or suppress the default legend, you must do that differently when by() is specified. For instance, `legend(off)`—see [G-3] `legend_options`—will suppress the legend, yet typing

```
. line y1 y2 x, by(group) legend(off)
```

will not have the intended effect. The `legend(off)` will seemingly be ignored. You must instead type

```
. line y1 y2 x, by(group, legend(off))
```

We moved `legend(off)` inside the by().

Remember that by() repeats the graph command. If you think carefully, you will realize that the legend never was displayed at the bottom of the individual plots. It is instructive to type

```
. line y1 y2 x, legend(on) by(group)
```

This graph will have many legends: one underneath each of the plots in addition to the overall legend at the bottom of the graph! by() works exactly as advertised: it repeats the entire graph command for each value of group.

In any case, it is the overall `legend()` that we want to suppress, and that is why we must specify `legend(off)` inside the by() option; this is the same issue as the one discussed under Treatment of titles above.

The issue becomes a little more complicated when, rather than suppressing the legend, we wish to modify the legend’s contents or position. Then the `legend()` option to modify the contents is specified outside the by() and the `legend()` option to modify the location is specified inside. See Use of legends with by() in [G-3] `legend_options`.
By-styles

Option `style(bystyle)` specifies the overall look of by-graphs; see [G-4] `bystyle` for a list of `bystyle` choices. One `bystyle` worth noting is `compact`. Specifying `style(compact)` causes the graph to be displayed in a more compact format. Compare

```
. use http://www.stata-press.com/data/r13/lifeexp, clear
(Life expectancy, 1998)
. scatter lexp gnppc, by(region, total)
```

with

```
. scatter lexp gnppc, by(region, total style(compact))
```

`style(compact)` pushes the graphs together horizontally and vertically, leaving more room for the individual graphs. The disadvantage is that, pushed together, the values on the axes labels sometimes run into each other, as occurred above with the 40,000 of the S.A. graph running into the 0 of the Total graph. That problem could be solved by dividing `gnppc` by 1,000.

Rather than typing out `style(compact)`, you may specify `compact`, and you may further abbreviate that as `com`. 
Labeling the edges

Consider the graph

```
.suuse auto
(1978 Automobile Data)
.scatter mpg weight, by(foreign, total)
```

The \( x \) axis is labeled in the graph in the (1,2) position. When the last graph of a column does not appear in the last row, its \( x \) axis is referred to as an edge. In `style(default)`, the default is to label the edges, but you could type

```
.scatter mpg weight, by(foreign, total noedgelabel)
```

to suppress that. This results in the rows of graphs being closer to each other.

Were you to type

```
.scatter mpg weight, by(foreign, total style(compact))
```

you would discover that the \( x \) axis of the (1,2) graph is not labeled. With `style(compact)`, the default is `noedgelabel`, but you could specify `edgelabel` to override that.
Specifying separate scales for the separate plots

If you type

```
.scatter yvar xvar, by(catvar, yrescale)
```

each graph will be given a separately scaled y axis; if you type

```
.scatter yvar xvar, by(catvar, xrescale)
```

each graph will be given a separately scaled x axis; and if you type

```
.scatter yvar xvar, by(catvar, yrescale xrescale)
```

both scales will be separately set.

History

The twoway scatterplots produced by the `by()` option are similar to what are known as casement displays (see Chambers et al. [1983, 141–145]). A traditional casement display, however, aligns all the graphs either vertically or horizontally.

References


Also see

[G-3] `region_options` — Options for shading and outlining regions and controlling graph size

[G-3] `title_options` — Options for specifying titles