**Description**

twoway is a family of plots, all of which fit on numeric y and x scales.

**Menu**

Graphics > Twoway graph (scatter, line, etc.)

**Syntax**

```
[graph] twoway plot [if] [in] [, twoway_options]
```

where the syntax of `plot` is

```
[ ( plottype varlist . . . , options [ ] ) [1] ]
```
<table>
<thead>
<tr>
<th><code>plottype</code></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>scatter</code></td>
<td>scatterplot</td>
</tr>
<tr>
<td><code>line</code></td>
<td>line plot</td>
</tr>
<tr>
<td><code>connected</code></td>
<td>connected-line plot</td>
</tr>
<tr>
<td><code>scatteri</code></td>
<td><code>scatter</code> with immediate arguments</td>
</tr>
<tr>
<td><code>area</code></td>
<td>line plot with shading</td>
</tr>
<tr>
<td><code>bar</code></td>
<td>bar plot</td>
</tr>
<tr>
<td><code>spike</code></td>
<td>spike plot</td>
</tr>
<tr>
<td><code>dropline</code></td>
<td>dropline plot</td>
</tr>
<tr>
<td><code>dot</code></td>
<td>dot plot</td>
</tr>
<tr>
<td><code>rarea</code></td>
<td>range plot with area shading</td>
</tr>
<tr>
<td><code>rbar</code></td>
<td>range plot with bars</td>
</tr>
<tr>
<td><code>rspike</code></td>
<td>range plot with spikes</td>
</tr>
<tr>
<td><code>rcap</code></td>
<td>range plot with capped spikes</td>
</tr>
<tr>
<td><code>rcapsym</code></td>
<td>range plot with spikes capped with symbols</td>
</tr>
<tr>
<td><code>rscatter</code></td>
<td>range plot with markers</td>
</tr>
<tr>
<td><code>rline</code></td>
<td>range plot with lines</td>
</tr>
<tr>
<td><code>rconnected</code></td>
<td>range plot with lines and markers</td>
</tr>
<tr>
<td><code>pcspike</code></td>
<td>paired-coordinate plot with spikes</td>
</tr>
<tr>
<td><code>pccapsym</code></td>
<td>paired-coordinate plot with spikes capped with symbols</td>
</tr>
<tr>
<td><code>pcarrow</code></td>
<td>paired-coordinate plot with arrows</td>
</tr>
<tr>
<td><code>pcbarrow</code></td>
<td>paired-coordinate plot with arrows having two heads</td>
</tr>
<tr>
<td><code>pcscatter</code></td>
<td>paired-coordinate plot with markers</td>
</tr>
<tr>
<td><code>pci</code></td>
<td><code>pcspike</code> with immediate arguments</td>
</tr>
<tr>
<td><code>pcarrowi</code></td>
<td><code>pcarrow</code> with immediate arguments</td>
</tr>
<tr>
<td><code>tsline</code></td>
<td>time-series plot</td>
</tr>
<tr>
<td><code>tsrline</code></td>
<td>time-series range plot</td>
</tr>
<tr>
<td><code>contour</code></td>
<td>contour plot with filled areas</td>
</tr>
<tr>
<td><code>contourline</code></td>
<td>contour lines plot</td>
</tr>
<tr>
<td><code>mband</code></td>
<td>median-band line plot</td>
</tr>
<tr>
<td><code>mspline</code></td>
<td>spline line plot</td>
</tr>
<tr>
<td><code>lowess</code></td>
<td>LOWESS line plot</td>
</tr>
<tr>
<td><code>lfit</code></td>
<td>linear prediction plot</td>
</tr>
<tr>
<td><code>qfit</code></td>
<td>quadratic prediction plot</td>
</tr>
<tr>
<td><code>fpfit</code></td>
<td>fractional polynomial plot</td>
</tr>
<tr>
<td><code>lfitci</code></td>
<td>linear prediction plot with CIs</td>
</tr>
<tr>
<td><code>qfitci</code></td>
<td>quadratic prediction plot with CIs</td>
</tr>
<tr>
<td><code>fpfitci</code></td>
<td>fractional polynomial plot with CIs</td>
</tr>
<tr>
<td><code>function</code></td>
<td>line plot of function</td>
</tr>
<tr>
<td><code>histogram</code></td>
<td>histogram plot</td>
</tr>
<tr>
<td><code>kdensity</code></td>
<td>kernel density plot</td>
</tr>
<tr>
<td><code>lpoly</code></td>
<td>local polynomial smooth plot</td>
</tr>
<tr>
<td><code>lpolyci</code></td>
<td>local polynomial smooth plot with CIs</td>
</tr>
</tbody>
</table>
The leading graph is optional. If the first (or only) plot is scatter, you may omit twoway as well, and then the syntax is

```
scatter ... [, scatter_options] [ || plot [plot [... ]]]
```
and the same applies to line. The other plottypes must be preceded by twoway.

Regardless of how the command is specified, twoway_options may be specified among the scatter_options, line_options, etc., and they will be treated just as if they were specified among the twoway_options of the graph twoway command.

**Remarks and examples**

Remarks are presented under the following headings:

- **Definition**
- **Syntax**
- **Multiple if and in restrictions**
- **twoway and plot options**

**Definition**

Twoway graphs show the relationship between numeric data. Say that we have data on life expectancy in the United States between 1900 and 1940:

```
use https://www.stata-press.com/data/r16/uslifeexp2
(U.S. life expectancy, 1900-1940)
list in 1/8
```

```
<table>
<thead>
<tr>
<th>year</th>
<th>le</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1900</td>
</tr>
<tr>
<td>2</td>
<td>1901</td>
</tr>
<tr>
<td>3</td>
<td>1902</td>
</tr>
<tr>
<td>4</td>
<td>1903</td>
</tr>
<tr>
<td>5</td>
<td>1904</td>
</tr>
<tr>
<td>6</td>
<td>1905</td>
</tr>
<tr>
<td>7</td>
<td>1906</td>
</tr>
<tr>
<td>8</td>
<td>1907</td>
</tr>
</tbody>
</table>
```
We could graph these data as a twoway scatterplot,

```stata
. twoway scatter le year
```

or we could graph these data as a twoway line plot,

```stata
. twoway line le year
```
or we could graph these data as a twoway connected plot, marking both the points and connecting them with straight lines,

```
. twoway connected le year
```

or we could graph these data as a scatterplot and put on top of that the prediction from a linear regression of \( le \) on \( year \),

```
. twoway (scatter le year) (lfit le year)
```

or we could graph these data in many other ways.

These all are examples of twoway graphs. What distinguishes a twoway graph is that it fits onto numeric \( y \) and \( x \) axes.

Each of what we produced above is called a \textit{graph}. What appeared in the graphs are called \textit{plots}. In the first graph, the plottype was a scatter; in the second, the plottype was a line; in the third, the plottype was connected; and in the fourth, there were two plots: a scatter combined with a line plot of a linear fit.
twoway provides many different plottypes. Some, such as scatter and line, simply render the data in different ways. Others, such as lfit, transform the data and render that. And still others, such as function, actually make up data to be rendered. This last class makes it easy to overlay \( y = x \) lines or \( y = f(x) \) functions on your graphs.

By the way, in case you are wondering, there are no errors in the above data. In 1918, there was an outbreak of influenza known as the 1918 Influenza Pandemic, which in the United States, was the worst epidemic ever known and which killed more citizens than all combat deaths of the 20th century.

Syntax

If we want to graph \( y_1 \) versus \( x \) and \( y_2 \) versus \( x \), the formal way to type this is

```
.graph twoway (scatter y1 x) (scatter y2 x)
```

If we wanted \( y_1 \) versus \( x \) plotted with solid circles and \( y_2 \) versus \( x \) plotted with hollow circles, formally we would type

```
.graph twoway (scatter y1 x, ms(O)) (scatter y2 x, ms(Oh))
```

If we wanted \( y_1 \) versus \( x \) plotted with solid circles and wanted a line graph for \( y_2 \) versus \( x \), formally we would type

```
.graph twoway (scatter y1 x, ms(O)) (line y2 x, sort)
```

The sort option is included under the assumption that the data are not already sorted by \( x \).

We have shown the formal way to type each of our requests, but few people would type that. First, most users omit the graph:

```
.twoway (scatter y1 x) (scatter y2 x)
.twoway (scatter y1 x, ms(O)) (scatter y2 x, ms(Oh))
.twoway (scatter y1 x, ms(O)) (line y2 x, sort)
```

Second, most people use the ||-separator notation rather than the ()-binding notation:

```
.twoway scatter y1 x || scatter y2 x
.twoway scatter y1 x, ms(0) || scatter y2 x, ms(Oh)
.twoway scatter y1 x, ms(0) || line y2 x, sort
```

Third, most people now omit the twoway:

```
.scatter y1 x || scatter y2 x
.scatter y1 x, ms(0) || scatter y2 x, ms(Oh)
.scatter y1 x, ms(0) || line y2 x, sort
```

And finally, most people quickly realize that scatter allows us to plot more than one \( y \) variable against the same \( x \) variable:

```
.scatter y1 y2 x
.scatter y1 y2 x, ms(0 Oh)
.scatter y1 x, ms(0) || line y2 x, sort
```

The third example did not change: in that example, we are combining a scatterplot and a line plot. Actually, in this particular case, there is a way we can combine that, too:

```
.scatter y1 y2 x, ms(0 i) connect(., 1)
```

That we can combine scatter and line just happens to be an oddity of the examples we picked. It is important to understand that there is nothing wrong with any of the above ways of typing our request, and sometimes the wordier syntaxes are the only way to obtain what we want. If we wanted to graph \( y_1 \) versus \( x_1 \) and \( y_2 \) versus \( x_2 \), the only way to type that is
or to type the equivalent in one of the wordier syntaxes above it. We have to do this because `scatter`
(see [G-2] graph twoway scatter) draws a scatterplot against one \textit{x} variable. Therefore, if we want
two different \textit{x} variables, we need two different scatters.

In any case, we will often refer to the `graph twoway` command, even though, when we give the
command, we will seldom type the graph, and mostly, we will not type the `twoway` either.

**Multiple if and in restrictions**

Each plot may have its own if \textit{exp} and in range restrictions:

```
   . twoway (scatter mpg weight if foreign, msymbol(O))
         (scatter mpg weight if !foreign, msymbol(Oh))
```

Multiple plots in one `graph twoway` command draw one graph with multiple things plotted in
it. The above will produce a scatter of mpg versus weight for foreign cars (making the points with
solid circles) and a scatter of mpg versus weight for domestic cars (using hollow circles).

Also, the `graph twoway` command itself can have if \textit{exp} and in range restrictions:

```
   . twoway (scatter mpg weight if foreign, msymbol(O))
         (scatter mpg weight if !foreign, msymbol(Oh)) if mpg>20
```

The if mpg>20 restriction will apply to both scatters.

We have chosen to show these two examples with the ()-binding notation because it makes the
scope of each if \textit{exp} so clear. In ||-separator notation, the commands would read

```
   . twoway scatter mpg weight if foreign, msymbol(O) ||
         scatter mpg weight if !foreign, msymbol(Oh)
```

and

```
   . twoway scatter mpg weight if foreign, msymbol(O) ||
         scatter mpg weight if !foreign, msymbol(Oh) || if mpg>20
```

or even

```
   . scatter mpg weight if foreign, msymbol(O) ||
         scatter mpg weight if !foreign, msymbol(Oh)
```

and

```
   . scatter mpg weight if foreign, msymbol(O) ||
         scatter mpg weight if !foreign, msymbol(Oh) || if mpg>20
```

We may specify `graph twoway` restrictions only, of course:

```
   . twoway (scatter mpg weight) (lfit mpg weight) if !foreign
   . scatter mpg weight || lfit mpg weight || if !foreign
```

**twoway and plot options**

`graph twoway` allows options, and the individual plots allow options. For instance, `graph twoway`
allows the `saving()` option, and `scatter` (see [G-2] graph twoway scatter) allows the `msymbol()`
option, which specifies the marker symbol to be used. Nevertheless, we do not have to keep track of
which option belongs to which. If we type

```
   . scatter mpg weight, saving(mygraph) msymbol(Oh)
```
the results will be the same as if we more formally typed

```
.twoway (scatter mpg weight, msymbol(Oh)), saving(mygraph)
```

Similarly, we could type

```
.scatter mpg weight, msymbol(Oh) || lfit mpg weight, saving(mygraph)
```

or

```
.scatter mpg weight, msymbol(Oh) saving(mygraph) || lfit mpg weight
```

and, either way, the results would be the same as if we typed

```
.twoway (scatter mpg weight, msymbol(Oh))
.lfit mpg weight), saving(mygraph)
```

We may specify a `graph twoway` option “too deeply”, but we cannot go the other way. The following is an error:

```
.scatter mpg weight || lfit mpg weight ||, msymbol(Oh) saving(mygraph)
```

It is an error because we specified a `scatter` option where only a `graph twoway` option may be specified, and given what we typed, there is insufficient information for `graph twoway` to determine for which `plot` we meant the `msymbol()` option. Even when there is sufficient information (say that option `msymbol()` were not allowed by `lfit`), it would still be an error. `graph twoway` can reach in and pull out its options, but it cannot take from its options and distribute them back to the individual plots.

References


