**added_text_options** — Options for adding text to twoway graphs

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
<th>Options</th>
<th>Remarks and examples</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>text</strong>(text_arg)</td>
<td>add text at specified y x</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>ttext</strong>(text_arg)</td>
<td>add text at specified y t</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The above options are *merged-implicit*; see [G-4] **concept: repeated options**.

where `text_arg` is

```
loc_and_text [ loc_and_text ... ] [ , textbox_options ]
```

and where `loc_and_text` is

```
#y #x "text" [ "text" ... ]
```

**Description**

`text()` adds the specified text to the specified location in the plot region.

`ttext()` is an extension to `text()`, accepting a date in place of `#x` when the time axis has a time format; see [U] 11.1.9 datelist.

**Options**

`text(text_arg)` and `ttext(text_arg)` specify the location and text to be displayed.
added_text_options — Options for adding text to twoway graphs

Suboptions

`yaxis(#)` and `xaxis(#)` specify how `#y` and `#x` are to be interpreted when there are multiple `y`, `x`, or `t` axis scales; see [G-3] axis_choice_options.

In the usual case, there is one `y` axis and one `x` axis, so options `yaxis()` and `xaxis()` are not specified. `#y` is specified in units of the `y` scale and `#x` in units of the `x` scale.

In the multiple-axis case, specify `yaxis(#)` and/or `xaxis(#)` to specify which units you wish to use. `yaxis(1)` and `xaxis(1)` are the defaults.

`placement(compassdirstyle)` specifies where the textbox is to be displayed relative to `#y` and `#x`. The default is usually `placement(center)`. The default is controlled both by the scheme and by the `textbox_option tstyle(textboxstyle)`; see [G-4] schemes intro and [G-3] textbox_options. The available choices are

<table>
<thead>
<tr>
<th>compassdirstyle</th>
<th>Location of text</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>centered on the point, vertically and horizontally</td>
</tr>
<tr>
<td>n</td>
<td>above the point, centered</td>
</tr>
<tr>
<td>ne</td>
<td>above and to the right of the point</td>
</tr>
<tr>
<td>e</td>
<td>right of the point, vertically centered</td>
</tr>
<tr>
<td>se</td>
<td>below and to the right of the point</td>
</tr>
<tr>
<td>s</td>
<td>below point, centered</td>
</tr>
<tr>
<td>sw</td>
<td>below and to the left of the point</td>
</tr>
<tr>
<td>w</td>
<td>left of the point, vertically centered</td>
</tr>
<tr>
<td>nw</td>
<td>above and to the left of the point</td>
</tr>
</tbody>
</table>

You can see [G-4] compassdirstyle, but that will just give you synonyms for `c`, `n`, `ne`, ..., `nw`.

textbox_options specifies the look of the text; see [G-3] textbox_options.

Remarks and examples

Remarks are presented under the following headings:

Typical use

Use of the textbox option `width()`

Typical use

text() is used for placing annotations on graphs. One example is the labeling of outliers. For instance, type

```
. use http://www.stata-press.com/data/r13/auto
(1978 Automobile Data)
. twoway qfitci mpg weight, stdf || scatter mpg weight
(graph omitted)
```
There are four outliers. First, we find the outliers by typing

```stata
. quietly regress mpg weight
. predict hat
(option xb assumed; fitted values)
. predict s, stdf
. generate upper = hat + 1.96*s
. list make mpg weight if mpg>upper
```

<table>
<thead>
<tr>
<th>make</th>
<th>mpg</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cad. Seville</td>
<td>21</td>
<td>4,290</td>
</tr>
<tr>
<td>Plym. Arrow</td>
<td>28</td>
<td>3,260</td>
</tr>
<tr>
<td>Datsun 210</td>
<td>35</td>
<td>2,020</td>
</tr>
<tr>
<td>Subaru</td>
<td>35</td>
<td>2,050</td>
</tr>
<tr>
<td>VW Diesel</td>
<td>41</td>
<td>2,040</td>
</tr>
</tbody>
</table>

Now we can remake the graph and label the outliers:

```stata
. twoway qfitci mpg weight, stdf ||
   scatter mpg weight, ms(O)
   text(41 2040 "VW Diesel", place(e))
   text(28 3260 "Plymouth Arrow", place(e))
   text(35 2050 "Datsun 210 and Subaru", place(e))
```
Another common use of `text` is to add an explanatory box of text inside the graph:

```
use http://www.stata-press.com/data/r13/uslifeexp, clear
(U.S. life expectancy, 1900–1999)
    twoway line le year ||
          fpfit le year ||
          , ytitle("Life Expectancy, years")
          xlabel(1900 1918 1940(20)2000)
          title("Life Expectancy at Birth")
          subtitle("U.S., 1900–1999")
          note("Source: National Vital Statistics Report, Vol. 50 No. 6")
          legend(off)
          text( 48.5 1923
                 "The 1918 Influenza Pandemic was the worst epidemic"
                 "known in the U.S."
                 "More citizens died than in all combat deaths of the"
                 "20th century."
                 , place(se) box just(left) margin(l+4 t+1 b+1) width(85) )
```

The only thing to note in the above command is the `text()` option:

```
    text( 48.5 1923
           "The 1918 Influenza Pandemic was the worst epidemic"
           "known in the U.S."
           "More citizens died than in all combat deaths of the"
           "20th century."
           , place(se) box just(left) margin(l+4 t+1 b+1) width(85) )
```

and, in particular, we want to draw your eye to the location of the text and the suboptions:

```
    text( 48.5 1923
           ... , place(se) box just(left) margin(l+4 t+1 b+1) width(85) )
```

We placed the text at $y = 48.5$, $x = 1923$, `place(se)`, meaning the box will be placed below and to the right of $y = 48.5$, $x = 1923$.

The other suboptions, `box just(left) margin(l+4 t+1 b+1) width(85)` are `textbox_options`. We specified `box` to draw a border around the textbox, and we specified `just(left)`, an abbreviation for `justification(left)`, so that the text was left-justified inside the box. `margin(l+4 t+1 b+1)`
made the text in the box look better. On the left we added 4%, and on the top and bottom we added 1%; see [G-3] textbox_options and [G-4] relativesize. width(85) was specified to solve the problem described below.

### Use of the textbox option width()

Let us look at the results of the above command, omitting the width() suboption. What you would see on your screen—or in a printout—might look virtually identical to the version we just drew, or it might look like this

![Graph of Life Expectancy at Birth](image)

That is, Stata might make the textbox too narrow or too wide. In the above illustrations, we have exaggerated the extent of the problem, but it is common for the box to run a little narrow or a little wide. Moreover, with respect to this one problem, how the graph appears on your screen is no guarantee of how it will appear when printed.
This problem arises because Stata uses an approximation formula to determine the width of the text. This approximation is good for some fonts and poorer for others.

When the problem arises, use the `textbox_option width(relativesize)` to work around it. `width()` overrides Stata’s calculation. In fact, we drew the two examples above by purposely misstating the `width()`. In the first case, we specified `width(40)`, and in the second, `width(95)`.

Getting the `width()` right is a matter of trial and error. The correct width will nearly always be between 0 and 100.

Corresponding to `width(relativesize)`, there is also the `textbox_option height(relativesize)`, but Stata never gets the height incorrect.

Reference


Also see

[G-3] `textbox_options` — Options for textboxes and concept definition