**Description**

twoway contourline displays $z$ as contour lines in ($y$, $x$).

**Quick start**

Contour-line plot displaying the value of $z$ for each ($x$, $y$) pair

twoway contourline $z$ $y$ $x$

As above, but draw contour lines for $z = -4$ to 1.5 in increments of 0.25

twoway contourline $z$ $y$ $x$, ccuts(-4(.25)1.5)

Specify 25 contour levels

twoway contourline $z$ $y$ $x$, levels(25)

As above, but with colored contour lines

twoway contourline $z$ $y$ $x$, levels(25) colorlines

As above, but with an alternative method of selecting line colors

twoway contourline $z$ $y$ $x$, levels(25) colorlines crule(chue)

Reverse the order cutpoints are listed in the legend

twoway contourline $z$ $y$ $x$, levels(25) colorlines reversekey

Add “My Title” as the title of the graph

twoway contourline $z$ $y$ $x$, title("My Title")

Add “My note” as a note on the graph

twoway contourline $z$ $y$ $x$, title("My Title") note("My note")

**Menu**

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

```
twoway contourline z y x [if] [in] [, options]
```

<table>
<thead>
<tr>
<th>options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ccuts(numlist)</code></td>
<td>list of values for contour lines or cuts</td>
</tr>
<tr>
<td><code>levels(#)</code></td>
<td>number of contour levels</td>
</tr>
<tr>
<td><code>minmax</code></td>
<td>include contour lines for minimum and maximum of z</td>
</tr>
<tr>
<td><code>format(%fmt)</code></td>
<td>display format for <code>ccuts()</code> or <code>levels()</code></td>
</tr>
<tr>
<td><code>colorlines</code></td>
<td>display contour lines in different colors</td>
</tr>
<tr>
<td><code>crule(crule)</code></td>
<td>rule for creating contour-line colors</td>
</tr>
<tr>
<td><code>scolor(colorstyle)</code></td>
<td>starting color for contour rule</td>
</tr>
<tr>
<td><code>ecolor(colorstyle)</code></td>
<td>ending color for contour rule</td>
</tr>
<tr>
<td><code>ccolors(colorstylelist)</code></td>
<td>list of colors for contour lines</td>
</tr>
<tr>
<td><code>clwidths(linewidthstylelist)</code></td>
<td>list of widths for contour lines</td>
</tr>
<tr>
<td><code>reversekey</code></td>
<td>reverse the order of the keys in contour-line legend</td>
</tr>
<tr>
<td><code>interp(interpmethod)</code></td>
<td>interpolation method if (z, y, x) does not fill a regular grid</td>
</tr>
<tr>
<td><code>twoway_options</code></td>
<td>titles, legends, axes, added lines and text, by, regions, name, aspect ratio, etc.</td>
</tr>
</tbody>
</table>

**crule**

<table>
<thead>
<tr>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>hue</code></td>
<td>use equally spaced hues between <code>scolor()</code> and <code>ecolor()</code>; the default</td>
</tr>
<tr>
<td><code>chue</code></td>
<td>use equally spaced hues between <code>scolor()</code> and <code>ecolor()</code>; unlike <code>hue</code>, it uses $360 + \text{hue}$ of the <code>ecolor()</code> if the hue of the <code>ecolor()</code> is less than the hue of the <code>scolor()</code></td>
</tr>
<tr>
<td><code>intensity</code></td>
<td>use equally spaced intensities with <code>ecolor()</code> as the base; <code>scolor()</code> is ignored</td>
</tr>
<tr>
<td><code>linear</code></td>
<td>use equally spaced interpolations of the RGB values between <code>scolor()</code> and <code>ecolor()</code></td>
</tr>
</tbody>
</table>

**interpmethod**

<table>
<thead>
<tr>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><code>thinplatespline</code></td>
<td>thin-plate-spline interpolation; the default</td>
</tr>
<tr>
<td><code>shepard</code></td>
<td>Shepard interpolation</td>
</tr>
<tr>
<td><code>none</code></td>
<td>no interpolation; plot data as is</td>
</tr>
</tbody>
</table>

**Options**

`ccuts()`, `levels()`, `minmax`, and `format()` determine how many contours are created and the values of those contours.

`ccuts(numlist)` specifies the z values for the contour lines. Contour lines are drawn at each value of `numlist`.

`levels(#)` specifies the number of contour lines to create; # – 1 contour lines will be created.
Minmax is a modifier of `levels()` and specifies that contour lines be drawn for the minimum and maximum values of $z$. By default, lines are drawn only for the cut values implied by levels, not the full range of $z$.

`format(\%fmt)` specifies the display format used to create the labels in the contour legend for the contour lines.

`ccuts()` and `levels()` are different ways of specifying the contour cuts and may not be combined.

`colorlines`, `crule()`, `scolor()`, `ecolor()`, `ccolors()`, and `clwidths()` determine the colors and width that are used for each contour line.

`colorlines` specifies that the contour lines be drawn in different colors. Unless the `ccolors()` option is specified, the colors are determined by `crule()`.

`crule(crule)` specifies the rule used to set the colors for the contour lines. Valid `crules` are `hue`, `chue`, `intensity`, and `linear`. The default is `crule(hue)`.

`scolor(colorstyle)` specifies the starting color for the rule. See `colorstyle`.

`ecolor(colorstyle)` specifies the ending color for the rule. See `colorstyle`.

`ccolors(colorstylelist)` specifies a list of `colorstyles` for each contour line. If RGB, CMYK, HSV, or intensity-adjusted (for example, `red*.3`) colorstyle is specified, they should be placed in quotes. Examples of valid `ccolors()` options include `ccolors(red green magenta)` and `ccolors(red "55 132 22" ".3 .9 .3 hsv" blue)`. See `colorstyle`.

`clwidths(linewidthstylelist)` specifies a list of `linewidthstyles`, one for each contour line. See `linewidthstyle`.

`reversekey` specifies that the order of the keys in the contour-line legend be reversed. By default, the keys are ordered from top to bottom, starting with the key for the highest values of $z$. See `plegend_option` in `legend_options`.

`interp(interpmethod)` specifies the interpolation method to use if $z$, $y$, and $x$ do not fill a regular grid. Variables $z$, $y$, and $x$ fill a regular grid if for every combination of nonmissing ($y$, $x$), there is at least one nonmissing $z$ corresponding to the pair in the dataset. For example, the following dataset forms a $2 \times 2$ grid.

```
. input z y x
  z y x
  1. 1 1 1
  2. 2 4 1
  3. 3 4 1
  4. 1 1 2
  5. 1 4 2
  6. end
```

If there is more than one $z$ value corresponding to a pair of ($y$, $x$), the smallest $z$ value is used in plotting. In the above example, there are two $z$ values corresponding to pair (4, 1), and the smallest value, 2, is used.

```
. input z y x
  z y x
  1. 1 1 1
  2. 2 2 1
  3. 1 1 2
  4. end
```

does not fill a regular grid because there is no $z$ value corresponding to the pair (2, 2).
`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`).

Remarks and examples

Remarks are presented under the following headings:

- Controlling the number of contour lines and their values
- Controlling the colors of the contour lines
- Choose the interpolation method

Controlling the number of contour lines and their values

We could draw a contour-line plot with default values by typing

```stata
   . use https://www.stata-press.com/data/r16/sandstone
   (Subsea elevation of Lamont sandstone in an area of Ohio)
   . twoway contourline depth northing easting
```

We add the `colorlines` option to display the values of cuts in the contour legend. We also include the `levels()` option to create \(#-1\) contour lines equally spaced between `min(depth)` and `max(depth)`. 
The ccuts() option gives you the finest control over creating contour lines. Here we use it to draw a contour-line plot with six cuts at 7500, 7600, 7700, 7800, 7900, and 8000.

Controlling the colors of the contour lines

`crule()`, `scolor()`, and `ecolor()` control the colors for each contour line.

draws a contour-line plot with lines of nine equally spaced z values between `min(depth)` and `max(depth)`. The starting color for lines is green and the ending color for lines is red. Also, the legend labels’ display format is `%9.1f`.

ccolors() specifies a list of colors to be used for each contour line.

```
. twoway contourline depth northing easting, ccuts(7600(100)8000)
> colorlines ccolors(red green magenta blue yellow)
```

Choose the interpolation method


Also see

- [G-2] graph twoway contour — Twoway contour plot with area shading
- [G-2] graph twoway line — Twoway line plots
- [G-2] graph twoway connected — Twoway connected plots