The Stata Journal (*yyyy*) *vv*, Number *ii*, pp. 1–7

A note on creating inset plots using graph twoway

Matthew Tibbles
mawtibbles@gmail.com

Abstract. Inset plots can be used to "zoom-in" on densely populated areas of a graph or to add extra relevant data in the form of, for example, distribution plots. The standard Stata command for combining plots, `graph combine`, does not, however, permit this type of seamless integration. Each plot within a `graph combine` object is allocated a grid cell, which cannot be placed within another grid cell - at least not without white grid space intruding upon the plot region of the main graph. I present a fairly simple workaround to this issue using reproducible examples. The main idea is to plot insets along a second axis, and to artifically modify the range of this axis in order to constrain the inset plot within a specified area of the main graph. Additional tips are included for producing more intricate, multi-layered inset graphs.

Keywords: st0001, inset plots, scatterplots, histograms, twoway, graph combine, graphics

1 Introduction

Inset plots can be a useful visiual aid in various graphical contexts. The classic example is when data points are densely clustered, though they may also be used to view variable distributions, anamolies (without the effects of visual compression) or additional relevant data. In Stata, it is not possible to superimpose one graph on top of another via `graph combine` because of the underlying grid architecture of Stata graphics. Each graph within a `graph combine` object is allocated a grid cell which cannot be placed within another cell - at least not without white grid space covering either the full vertical or horizontal span of the graph area in which the inset cell is placed. Still, Stata grahpics are highly flexible, permitting a single graph to use up to 9 x, y and z axes (Wiggins 2010). Through the use of `graph twoway`, it is possible, therefore, to add inset plots to a `graph twoway` object by creating and plotting along one more more additional sets of axes.

In what follows, I describe this method for creating inset plots in Stata through reference to general, reproducible examples. I first demonstrate the basic idea, which is to modify the range of the added axis so that the inset plot is constrained within a specified area of the main graph. I then explore the the broader functionality of this method by plotting multiple insets, as well as adding `twoway` objects to link the inset plots to the main graph.

2 The basic idea

As a general working example, I use climate data on 956 U.S. cities. We will examine the relationship between average July temperature, `tempjuly`, and Cooling Degree Days (CDD), `cooldd`, which is the difference between mean July temperature and 65°F. I `separate(tempjuly)` by `region` so that the data can be efficiently plotted by group, which, in turn, accenutates the advantages of inset plotting. Following (Cox 2016), I create local macros to store commands chunks which require multiple reuse.

```
webuse citytemp4, clear
separate(tempjuly), by(region)
label variable tempjuly1 "NE"
label variable tempjuly2 "N Cntrl"
label variable tempjuly3 "South"
label variable tempjuly4 "West"
set scheme sj
local mops msymbol(O D S T) msize(1.2 1.2 1.2 1.2) mfcolor(gs0 gs5 gs10 gs15)    ///
mlcolor(gs0 gs0 gs0 gs0)  mlwidth(.05 .05 .05 .05)
local lops legend(order(0 "Region" 1 2 3 4) rows(1))
local labs xtitle("Cooling Degree Days (CDD)") ytitle("Average July temperature")
graph twoway scatter tempjuly1 tempjuly2 tempjuly3 tempjuly4 cooldd, `mops' `lops' `labs'
```

Figure 1: Scatter plot of average July temperature and CDD by region

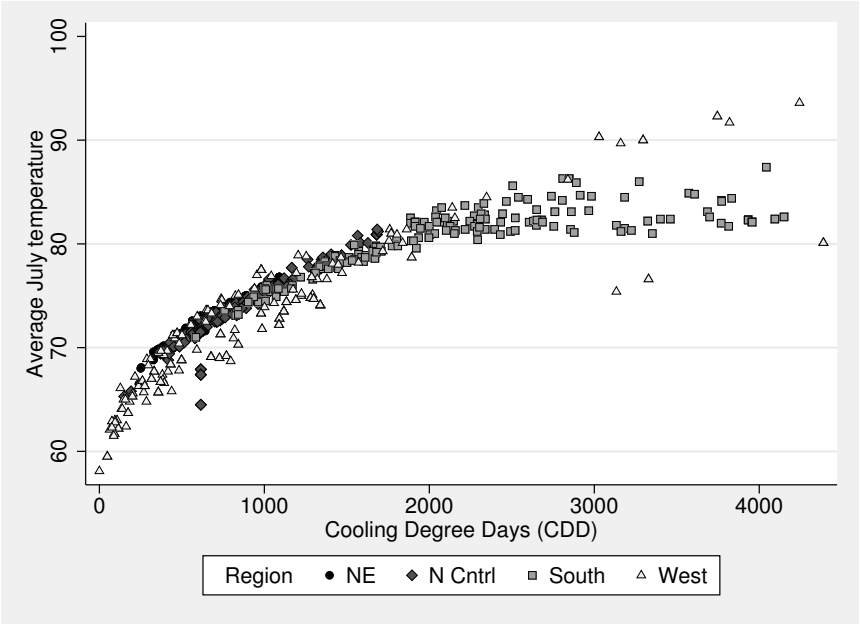


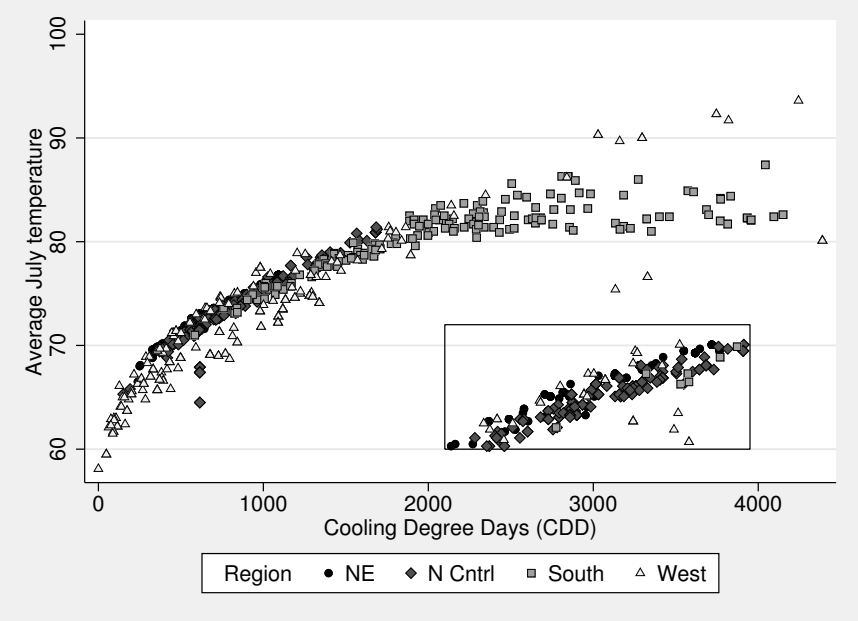
Figure 1 shows the relationship between average July tempature and CDD by region for 956 U.S. cities. North East and North Central cities are densely clustered between 70°F and 75°F on the y axis, while visibility is further impaired by a scattering of South and West cities. To increase visibility, we may want to add an inset plot; one which "zooms-in" on this densely populated area of the graph.

The first step - after adding a second `twoway` scatter plot to the initial `twoway` call - is

to limit the plotting range on the y axis to between 70°F and 75°F via the `if` qualififer. I additionally limit the x range to 950 CDD at the upper limit in order to exclude a handful of West cities which would otherwise result in visual compression of the points of interest. Next, I create and plot along a new set of axes by specifying `yaxis(2)` and `xaxis(2)`. The "trick" is then to artifically modify the range of each new axis so that the resulting plot is constrained within an area of the main plot. I opt to add the inset at the bottom, right-of-centre. To do so, I limit the y axis range to between 69°F and 90°F via `ylabel(69 90, axis(2))`. The x axis range is limited to between -300 and 1100 CDD via `xlabel(-300 1100, axis(2))`. I then make the inset axes invisible via `yscale(axis(2) off)` and `xscale(axis(2) off)` and specify the (undocumented) `norescaling` option of `graph twoway` in order to preserve the axis range/labels of the main plot. The final step is to add a border around the inset plot using the main graph axes. I do this via `scatteri` - which accepts multiple paired-coordinates as inputs - with the option `recast(line)`.

```
local bops lpattern(solid) lwidth(thin) lcolor(black)
graph twoway scatter tempjuly1 tempjuly2 tempjuly3 tempjuly4  cooldd,          ///
‘mops’ ‘lops’ ‘labs’                                                          ///
|| scatter tempjuly1 tempjuly2 tempjuly3 tempjuly4 cooldd                     ///
if inrange(tempjuly, 70, 75) & cooldd <= 950, ‘mops’                          ///
yaxis(2) xaxis(2)  ylabel(69 90, axis(2)) xlabel(-300 1100, axis(2))          ///
yscale(axis(2)off) xscale(axis(2)off) norescaling                             ///
|| scatteri 60 2100 72 2100 72 3950 60 3950 60 2100, recast(line) ‘bops’
```

Figure 2: Scatter plot of average July temperature and CDD by region; an inset plot is included to aid visibility of a densely clustered group of North and North Central cities where average July temperature falls between 70°F and 75°F.



Finding an appropriate range for the constrained inset axes is a simple though, admittedly, iterative process. In the example above, I opted to add the inset at the bottom

of the main graph. As such, the lower limit on the y axis needed to be approximately equal to the lowest `tempjuly` value found within the inset sample. Conversely, the upper limit needed to be considerably higher than the highest `tempjuly` inset value so as to constrain the inset plot within the bottom third of the main graph. After a couple of near-misses - in which the inset took up too much vertical space - I landed on 69°F and 90°F for the lower and upper limits respectively.

3 Extra details and multiple insets

I now turn to slightly more sophistcated usage of these plots. In the first instance, it is difficult to determine precisely which area the of the main graph is plotted within the inset. Using `scatteri` and option `recast(line)`, I draw a border around the inset area using the main graph axes. I add connecting lines between this border and that of the inset, as well as ticks and labels on the inset plot axes using `scatteri` with option `text`. Now, suppose we have some particular interest in CDD at specific average July temperatures - say, between 70°F and 72°F. We may want to highlight this y range on both the main graph and inset. Using `scatteri` and option `recast(area)`, I shade the horizontal zone between 70°F and 72°F using the x axis of the main graph - via `xaxis(1)` - and the x axis of the inset plot - via `xaxis(2)`. This is done prior to `scatter` so that the shaded zones do not blur the scatter points (Cox 2016). Finally, we may want to plot the distribution of each variable. The standard approach is to use `graph combine` (see Ängquist 2014), however inset plotting offers a slightly more flexible, space-saving alternative. I plot the distribution of `cooldd` as a histogram - via the (undocumented) `twoway__histogram_gen` (see Harrison 2005) - using an additional new set of modified axes. The inset histogram spans the full x axis of the main graph, while the modified y axis constrains the inset within the upper quadrant. The inverse of this logic is used to plot a kernel density of `tempjuly`.

Histogram variables and new macros:

```
forvalues r = 1/4 {
   twoway__histogram_gen cooldd if region ==‘r’,                                ///
   density gen(h‘r’ x‘r’) start(0) width(300)
}
replace x1 = x1 - 50
replace x2 = x2 - 100
replace x3 = x3 - 150
replace x4 = x4 - 200
local l2ops legend(order(0 "Region" 2 3 4 5) rows(1))
local dops lcolor(black) lwidth(vthin)
```

Main graph with shaded zone:

```
graph twoway scatteri 70 -250 72 -250 72 4300 70 4300 70 -250, recast(area)     ///
color(gs14%70) lpattern(solid)  lwidth(vthin) plotregion(margin(l =-2.1))       ///
|| scatter tempjuly1 tempjuly2 tempjuly3 tempjuly4 cooldd,‘mops’ ‘l2ops’ ‘labs’ ///
```

Inset plot with shaded zone:

```
|| scatteri 70 380 72 380 72 959 70 959 70 380, recast(area) color(gs14%70)     ///
lpattern(solid) lwidth(vthin) yaxis(2) xaxis(2)  ylabel(69 90, axis(2))         ///
xlabel(-300 1100, axis(2)) yscale(axis(2)off) xscale(axis(2)off) norescaling    ///
```

```
|| scatteri 74.9 380 75.95 380 75.95 959 74.9 959 74.9 380, recast(area)       ///
yaxis(2) xaxis(2) color(white) lpattern(solid) lwidth(vthin)                  ///
|| scatter tempjuly1 tempjuly2 tempjuly3 tempjuly4 cooldd                     ///
if inrange(tempjuly, 70, 75) & cooldd <= 950, ‘mops’ ‘l2ops’  yaxis(2) xaxis(2) ///
|| scatteri 60 2000 72 2000 72 3925 60 3925 60 2000, recast(line) ‘bops’      ///
```

Main graph inset box and connecting lines:

```
|| scatteri 70 500 76 500 76 950 70 950 70 500, recast(line) ‘bops’           ///
|| scatteri 70 500 60 2000, recast(line) ‘bops’                               ///
|| scatteri 76 950 72 3925, recast(line) ‘bops’                               ///
```

Inset axis ticks and labels:

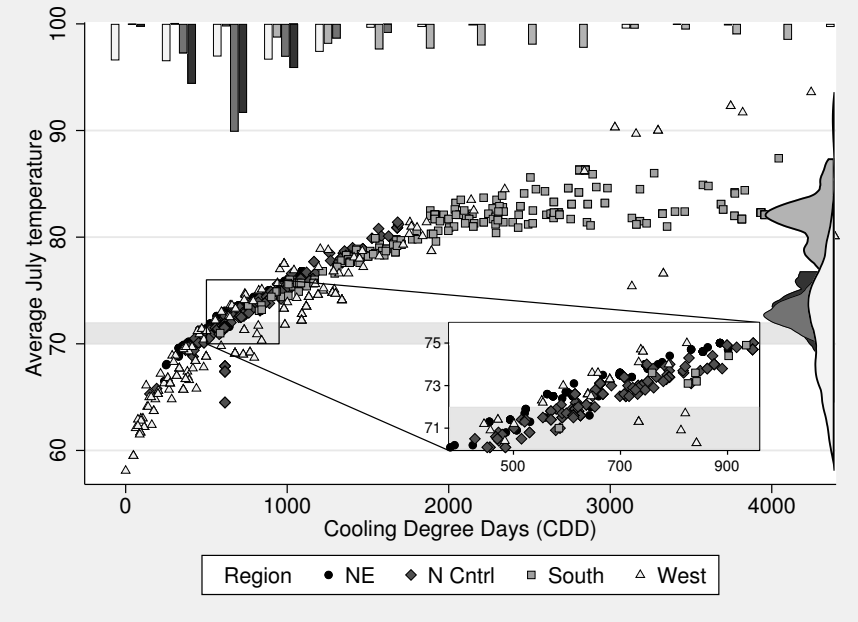
```
|| pci 71 371 71 380, xaxis(2) yaxis(2) ‘bops’                                ///
text(71 350 "71", size(4.5pt) xaxis(2) yaxis(2) just(left))                   ///
|| pci 73 371 73 380, xaxis(2) yaxis(2) ‘bops’                                ///
text(73 350 "73", size(4.5pt) xaxis(2) yaxis(2) just(left))                   ///
|| pci 75 371 75 380, xaxis(2) yaxis(2) ‘bops’                                ///
text(75 350 "75", size(4.5pt) xaxis(2) yaxis(2) just(left))                   ///
|| pci 69.75 500 69.95 500, xaxis(2) yaxis(2) ‘bops’                         ///
text(69.25 500 "500", size(4.5pt) xaxis(2) yaxis(2) just(left))               ///
|| pci 69.75 700 69.95 700, xaxis(2) yaxis(2) ‘bops’                         ///
text(69.25 700 "700", size(4.5pt) xaxis(2) yaxis(2) just(left))               ///
|| pci 69.75 900 69.95 900, xaxis(2) yaxis(2) ‘bops’                         ///
text(69.25 900 "900", size(4.5pt) xaxis(2) yaxis(2) just(left))               ///
```

Inset histogram and kernel density:

```
|| bar h1 x1, barw(50) fcolor(gs0) ‘dops’ yaxis(3) xaxis(3)                   ///
ylabel(0.0075 0.005, axis(3)) xlabel(0 4000, axis(3))                         ///
yscale(reverse axis(3) off)  xscale(alt axis(3) off)                          ///
|| bar h2 x2, barw(50) fcolor(gs5)  ‘dops’  yaxis(3)  xaxis(3)               ///
|| bar h3 x3, barw(50) fcolor(gs10)  ‘dops’ yaxis(3)  xaxis(3)               ///
|| bar h4 x4, barw(50) fcolor(gs15)  ‘dops’  yaxis(3)  xaxis(3)              ///
plotregion(margin(t = -1.95))                                                 ///
|| kdensity tempjuly if region == 1, recast(area) horizontal  fcolor(gs0%90)  ///
‘dops’  yaxis(4) xaxis(4)   ylabel(60 100, axis(4)) xlabel(0 2, axis(4))      ///
 xscale(axis(4) reverse off)  yscale(axis(4) off)                             ///
|| kdensity tempjuly if region == 2, recast(area) horizontal  fcolor(gs5%90)  ///
‘dops’ yaxis(4)  xaxis(4)                                                     ///
|| kdensity tempjuly if region == 3, recast(area) horizontal fcolor(gs10%90)  ///
 lcolor(black) yaxis(4)  xaxis(4)                                             ///
|| kdensity tempjuly if region == 4, recast(area) horizontal  fcolor(gs15%90) ///
lcolor(black) yaxis(4) xaxis(4) plotregion(margin(r =-1.95))
```

Admittedly, this final example conveys probably too much information. Yet the broader - and, for our purposes, more important - take-away is that it is possible to produce this type of intricate, mutli-layered inset graph using native Stata commands. Before closing this discussion, I would be remiss if I did not draw attention to the excellent `addplot` package by Ben Jann, which permits users to add plots to existing `twoway` objects (Jann 2015). The advantage of this is that large and somewhat unwieldy code chunks - such that is needed to produce the example below - can be broken up into more mangable, cleaner-looking segments.

Figure 3: Scatter plot of average July temperature and CDD by region with inset scatter, histogram and kernel density plots



4 Final thoughts

I have described a fairly straightforward method for creating highly flexible inset plots in Stata. The main idea - to artifically modify axis ranges so as to constrain the inset within a specified area of a larger graph - is, perhaps, not an ideal solution. But it appears to be the best available. The examples I have presented above barely touch the surface of the full functionality of these plots. Indeed, to do so would be to provide an exhaustive account of every `twoway` *plottype* and possible combination of *plottypes*. That is to say, these inset plots are simply `twoway` objects which have been manipulated to seamlessly appear within another `twoway` object. If Stata can plot it, then Stata can inset it.

5 References

Ängquist, L. 2014. gr0057. Stata tip 117: graph combine—Combining graphs. *Stata Journal* 14: 221–225.

Cox, N. J. 2016. gr0067. Speaking Stata: Shading zones on time series and other plots. *Stata Journal* 14: 805–812.

Harrison, D. A. 2005. gr0014. Stata tip 20: Generating histogram bin variables. *Stata Journal* 5: 280–281.

Jann, B. 2015. gr0065. A note on adding objects to an existing twoway graph. *Stata Journal* 15: 751–755.

Wiggins, V. 2010. gr0047. Stata tip 93: Handling multiple y axes on twoway graphs. *Stata Journal* 10: 689–690.

About the authors

Some background information about the first author.

Some background information about the second author.