The Stata Journal

The *Stata Journal* publishes reviewed papers together with shorter notes or comments, regular columns, book reviews, and other material of interest to Stata users. Examples of the types of papers include 1) expository papers that link the use of Stata commands or programs to associated principles, such as those that will serve as tutorials for users first encountering a new field of statistics or a major new technique; 2) papers that go "beyond the Stata manual" in explaining key features or uses of Stata that are of interest to intermediate or advanced users of Stata; 3) papers that discuss new commands or Stata programs of interest either to a wide spectrum of users (e.g., in data management or graphics) or to some large segment of Stata users (e.g., in survey statistics, survival analysis, panel analysis, or limited dependent variable modeling); 4) papers analyzing the statistical properties of new or existing estimators and tests in Stata; 5) papers that could be of interest or usefulness to researchers, especially in fields that are of practical importance but are not often included in texts or other journals, such as the use of Stata in managing datasets, especially large datasets, with advice from hard-won experience; and 6) papers of interest to those who teach, including Stata with topics such as extended examples of techniques and interpretation of results, simulations of statistical concepts, and overviews of subject areas.

The *Stata Journal* is indexed and abstracted by *CompuMath Citation Index*, *Current Contents/Social and Behavioral Sciences*, *RePEc: Research Papers in Economics*, *Science Citation Index Expanded* (also known as *SciSearch*), *Scopus*, and *Social Sciences Citation Index*.

For more information on the *Stata Journal*, including information for authors, see the webpage

http://www.stata-journal.com

Subscriptions are available from StataCorp, 4905 Lakeway Drive, College Station, Texas 77845, telephone 979-696-4600 or 800-STATA-PC, fax 979-696-4601, or online at

http://www.stata.com/bookstore/sj.html

Subscription rates listed below include both a printed and an electronic copy unless otherwise mentioned.

U.S. and Canada		Elsewhere	
Printed & electronic		**Printed & electronic**	
1-year subscription	$ 98	1-year subscription	$138
2-year subscription	$165	2-year subscription	$245
3-year subscription	$225	3-year subscription	$345
1-year student subscription	$ 75	1-year student subscription	$ 99
1-year institutional subscription	$245	1-year institutional subscription	$285
2-year institutional subscription	$445	2-year institutional subscription	$525
3-year institutional subscription	$645	3-year institutional subscription	$765
Electronic only		**Electronic only**	
1-year subscription	$ 75	1-year subscription	$ 75
2-year subscription	$125	2-year subscription	$125
3-year subscription	$165	3-year subscription	$165
1-year student subscription	$ 45	1-year student subscription	$ 45

Back issues of the *Stata Journal* may be ordered online at

http://www.stata.com/bookstore/sjj.html

Individual articles three or more years old may be accessed online without charge. More recent articles may be ordered online.

http://www.stata-journal.com/archives.html

The *Stata Journal* is published quarterly by the Stata Press, College Station, Texas, USA.

Address changes should be sent to the *Stata Journal*, StataCorp, 4905 Lakeway Drive, College Station, TX 77845, USA, or emailed to sj@stata.com.

The *Stata Journal* (ISSN 1536-867X) is a publication of Stata Press. Stata, Stata, Stata Press, Mata, Mata, and NetCourse are registered trademarks of StataCorp LP.

The Stata Journal (2014)
14, Number 1, pp. 230–235

Stata tip 119: Expanding datasets for graphical ends

Nicholas J. Cox
Department of Geography
Durham University
Durham, UK
n.j.cox@durham.ac.uk

Graphical tasks in Stata range from direct plotting of the data in memory (scatterplots, for example) to plotting of results calculated on the fly (histograms, for example). With the first kind of problem, the only issues are graphical. With the second kind of problem, Stata commands need to do some work on your behalf before graphs can be drawn. For a histogram, Stata needs to calculate bar coordinates (heights and base locations) before they can be plotted. Such preliminary work is often needed, although much of the art of graphics programming is to hide it from the user.

When Stata graphical commands are not available for what you want to do, changing the dataset temporarily is often the strategy to adopt. Here I show how using the `expand` (see [D] **expand**) command is a way to overcome some otherwise awkward challenges.

With the familiar auto dataset, we could look at means for, say, `mpg` (miles per gallon) for a one-way breakdown of the data and for a two-way breakdown of the data. Here we classify by the categorical variables `foreign` (whether cars are domestic or foreign, meaning U.S. made) and `rep78` (repair record).

```
. sysuse auto
(1978 Automobile Data)
. graph dot (mean) mpg, over(foreign) ytitle(Mean miles per gallon) nofill
> missing
. graph dot (mean) mpg, over(rep78) over(foreign) ytitle(Mean miles per gallon)
> nofill missing
```

A detail here is specifying the `missing` option. Clearly, if values missing on one or another categorical variable were of no concern to us, we would not do that. Either way, being careful about missing values is a good idea. Figure 1 shows the two graphs side by side.

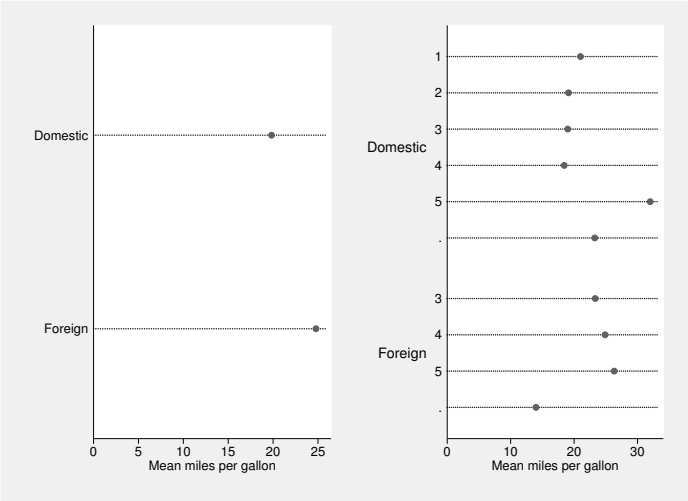

Figure 1. Two displays of mean miles per gallon. The left-hand graph is a poor use of space, but including its means in the right-hand graph would make sense.

Few researchers might want to show the left-hand graph of figure 1 because it conveys too little information. But many might want to enhance the right-hand graph with the summaries it shows. So how do we do that? The two graphs show different reductions of the data, so we need to ensure that both reductions are accessible at the same time. After saving a copy of the dataset first, we can double the dataset with `expand`:

```
. preserve
. expand 2
(74 observations created)
```

`expand` here adds an extra copy of the data as extra observations so that the first half of the observations is the original dataset and the second half is new but identical. Knowing that `rep78` has values 1 to 5 and that there are some missings (`.`), we could lump all the values together using any other integer or extended missing value.

```
. replace rep78 = .z if _n > _N/2
(74 real changes made, 74 to missing)
. label define rep78 .z "all"
. label values rep78 rep78
. graph dot (mean) mpg, over(rep78) over(foreign) ytitle(Mean miles per gallon)
> nofill missing
```

Figure 2 shows the result. In effect, we did a two-way breakdown with one half of the dataset and a one-way breakdown with the other half, but given the way we structured the data, Stata ended up doing both at once.

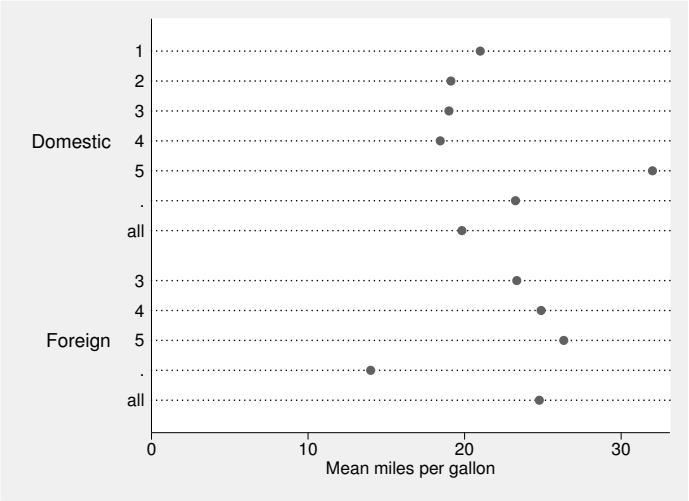

Figure 2. Extended summary of mean miles per gallon combining one-way and two-way breakdowns.

The odd-looking choice of `.z` to show results for "all" deserves comment. You can attach a value label to it, as just shown, so it will not be plotted as such. One particular reason for choosing `.z` is that it is larger than anything else, even when system missings (`.`) are present. Hence, it will be plotted at the end of each block of lines in the graph. Similarly, if you were using `graph bar` or `graph hbar` to draw bar charts, it would be plotted at the end of each block of bars. Either way, the `missing` option is now essential. It is possible that you are already using `.z` as a value. In that case, or for other reasons, you could instead choose any integer that ensures results for "all" are plotted at the beginning of each block of lines or bars. In this example, 0 or any negative integer would work fine.

Figure 2 is only a start. We might want to improve it in some way—say, by using different marker symbols for the "all" category—but we will leave the example there.

The same dataset can be used to show another application of `expand` to solve a similar graphical challenge. Imagine starting again and drawing some box plots. This time, we choose to be indifferent to the missing values.

```
. sysuse auto, clear
(1978 Automobile Data)
. graph box mpg, over(rep78)
. graph box mpg, over(foreign)
```

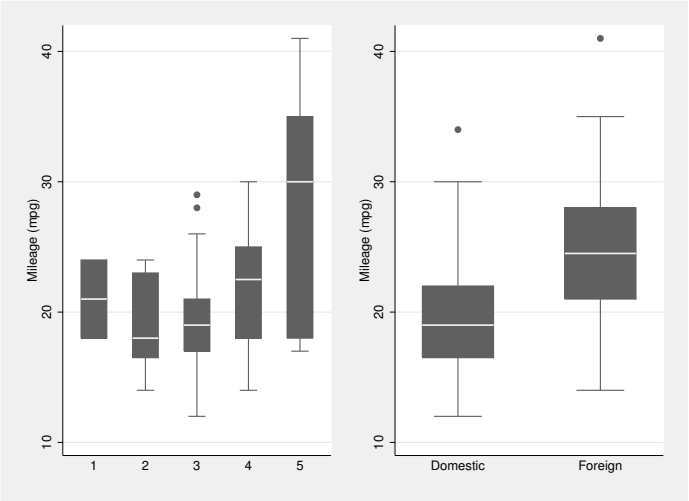

Figure 3. Box plots for miles per gallon classified by repair record (left) and domestic or foreign (right). The two graphs were produced separately and put together with `graph combine`.

We could just juxtapose these box plots, as in figure 3, but now there are two related small problems. The different sizes (bar widths) of the boxes are disconcerting, and the display for repair record takes up too much space for the information given. We could tinker with the sizes of the graphs, but we will explore a solution based on an initial `expand`.

We `expand` the data and produce a combined categorical variable. We cannot just copy `foreign`: its values of 0 and 1 overlap with those of 1 to 5 for `rep78`. Adding 6 to `foreign` will circumvent this problem.

```
. preserve
. expand 2
(74 observations created)
. generate x = cond(_n < _N/2, rep78, 6 + foreign)
(5 missing values generated)
. label define x 6 "Domestic" 7 "Foreign"
. label values x x
```

We now want something like `graph box mpg, over(x)`. With some more work, we can go further. `graph` will use different colors, markers, and so forth if it is plotting different variables. The `separate` command is ideal for producing two or more response variables (graphically, y variables) from one.

```
. separate mpg, by(x > 5)
              storage   display    value
variable name   type    format     label      variable label
------------------------------------------------------------------------------
mpg0            byte    %8.0g                 mpg, !(x > 5)
mpg1            byte    %8.0g                 mpg, x > 5
. graph box mpg?, over(x, relabel(3 `" "3" "Repair record" "´)) nofill
> legend(off) box(1, fcolor(white)) box(2, fcolor(gs4))
> medline(lc(gs8)) ytitle("`: var label mpg´") ylabel(, angle(h))
```

Figure 4 shows the result. Small details include using `relabel()` to provide a label for repair record, centered appropriately; tinkering with default colors for the box display; and getting an axis title directly from the variable label of `mpg`. (Start with `macro` (see [P] **macro**) if you want more information on the syntax used for the last detail.)

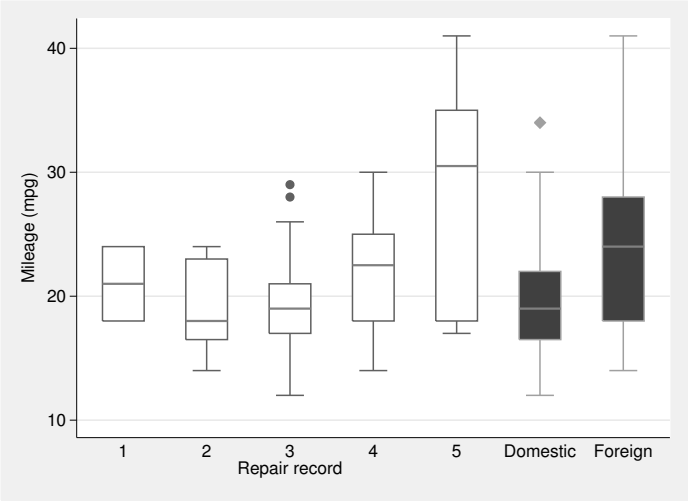

Figure 4. Box plots for miles per gallon classified by repair record (left) and domestic or foreign (right). The two graphs were produced in one call after rearranging the data.

This tip aims at giving you just a taste of what can be done with `expand` to solve some graphical problems, but more can be said.

Recall that each example using `expand` first used `preserve`, saving whatever data were in memory. If you did this within a do-file or program, then by default, there would be an automatic `restore`. With a `save`d copy of the dataset, you can afford to be cavalier about changing the data.

Some graphical problems might require `expand 3` or even more copies of the data. If that is so, know that `expand` adds observations to the existing dataset observation by observation rather than in blocks. Thus with `expand 3`, the original dataset would be followed by two copies of the first observation, two copies of the second observation, and so forth. If you have an identifier for each observation, such as `make` in the auto dataset, an identifier distinguishing different copies of the original would be useful, as in

```
. by make, sort: generate block = _n
```

If you did not have such an identifier, you would need to create one before the `expand`, by typing

```
. generate long id = _n
```

All of this is naturally based on the assumption that memory is available to allow the `expand` command to run. You could ease any memory problem by ensuring that variables or observations not needed for graphics were dropped with the `drop` command first. If that is insufficient, you might be able to use similar ideas but ones based on adding extra observations containing appropriate summaries. That could be very easy or very difficult. In the first example here, adding two observations with the means of `mpg` for the two categories of `foreign` would be enough. In the second example, adding enough information to produce the same box plots would be much more challenging.