

Stata tip 67: J() now has greater replicating powers

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

The Mata standard function J() was generalized in the Stata update of 25 February 2008. This tip flags its greater replicating powers. Note that J() in Stata's original matrix language remains as it was.

Users of Mata will have become accustomed to the role of J() in creating matrices of constants. For example, once within Mata,

```
: J(5,5,1)
[symmetric]
      1   2   3   4   5
1      1
2      1   1
3      1   1   1
4      1   1   1   1
5      1   1   1   1   1
```

You may be used to thinking of the way J() works like this: I want a 5×5 matrix, all of whose elements are the scalar 1. Another way of thinking about it is this: Give me 5 replicates or copies rowwise and 5 copies columnwise of the scalar 1. The results are identical when scalars are being replicated.

The second way of thinking about it helps in understanding the generalization now in place. What is to be replicated can now be a matrix, naturally including not only scalars but also vectors as special cases.

The help file gives full technical details and a variety of examples, but here is another. My Speaking Stata column in this issue (Cox 2008) mentions the bias on Fisher's z scale when estimating correlation r from sample size n of $2r/(n-1)$. The question is thus how big this is for a variety of values of r and n . We can quickly get a table from Mata:

```
: r = J(5, 1, (.1, .3, .5, .7, .9))
: r
      1   2   3   4   5
1      .1   .3   .5   .7   .9
2      .1   .3   .5   .7   .9
3      .1   .3   .5   .7   .9
4      .1   .3   .5   .7   .9
5      .1   .3   .5   .7   .9
```

```

: n = J(1, 5, (10, 20, 50, 100, 200)')
: n
      1      2      3      4      5
1  10    10    10    10    10
2  20    20    20    20    20
3  50    50    50    50    50
4  100   100   100   100   100
5  200   200   200   200   200

: 2 * r :/ (n :- 1)
      1      2      3      4      5
1  .022222222 .066666667 .111111111 .155555556 .2
2  .0105263158 .0315789474 .0526315789 .0736842105 .0947368421
3  .0040816327 .012244898 .0204081633 .0285714286 .0367346939
4  .002020202 .0060606061 .0101010101 .0141414141 .0181818182
5  .0010050251 .0030150754 .0050251256 .0070351759 .0090452261

```

Notice again how the first two arguments of `J()` are the numbers of replicates or copies, rowwise and columnwise, and not necessarily the numbers of rows and columns in the resulting matrix.

Reference

Cox, N. J. 2008. Speaking Stata: Correlation with confidence, or Fisher's *z* revisited. *Stata Journal* 8: 413–439.

Software Updates

st0123_1: Maximum likelihood and two-step estimation of an ordered-probit selection model. R. Chiburis and M. Lokshin. *Stata Journal* 7: 167–182.

A bug caused `predict`, used after `heckman`, to produce an error message when the `yif()` option was specified. This has been fixed.

st0140_1: `fuzzy`: A program for performing qualitative comparative analyses (QCA) in Stata. K. C. Longest and S. Vaisey. *Stata Journal* 8: 79–104.

Various minor fixes have been made to this package.

The default method of handling missing data has been changed. Originally, users needed to decide whether to include observations that were missing on the outcome set in the analyses, because some QCA analyses do not require full listwise deletion. Now analyses are carried out only for observations with full information on pertinent variables, including the outcome. This change places `fuzzy` in line with the majority of Stata programs and forces users to make informed decisions about including observations with missing values (i.e., explicitly recode them to a particular value).

`fuzzy.ado`: The created `bestfit` variable was being replaced when subsequent calls to `fuzzy` were made, even when `bestfit` had been renamed with a suffix (e.g., `bestfit2`). This problem has been fixed so that the program only automatically overwrites a variable explicitly named `bestfit`.

`coverage.ado`: A problem occurred when a user invoked this program on a very small dataset. The program became confused when there was only one variable named with a leading lowercase letter that was similar to one of the assigned letter labels in the program, and this lowercase letter was one of the final solution configurations. This has been fixed.

`yvo.ado` and `cmvom.ado`: A more informative error message is now issued if fewer than two configurations are entered.

`setgen.ado`: The `crisp()` option did not function properly for variables with negative medians. This has been fixed.