The Stata Journal (2020)
**20**, Number 2, pp. 481–488 DOI: 10.1177/1536867X20931007

# Speaking Stata: More ways for rowwise

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

**Abstract.** A previous column (Cox, 2009, *Stata Journal* 9: 137–157) gave a review of methods for working rowwise in Stata. Here *rows* means observations in a dataset, and the concern is calculations in each observation with a bundle of variables. For example, a row mean variable can be generated as the mean of some numeric variables in each observation.

This column is an update. It is briefly flagged that official Stata now has `rowmedian()` and `rowpctile()` functions for `egen`. The main focus is on returning which variable or variables are equal to the maximum or minimum in a row. The twist that requires care is that two or more variables can tie for minimum or maximum. That may entail a decision on what is to be recorded, such as all of them or just the first or last occurrence of an extreme.

**Keywords:** pr0046_1, rows, functions, minimum, maximum, median, loops, egen

## 1 Introduction

The main data structure of Stata is asymmetric: rows of a dataset define observations, and columns define variables. The great majority of Stata commands, starting with workhorses like `summarize`, `regress`, and `graph`, are designed to work with that structure. Such a structure is not eccentric; it has long been regarded as standard or at least default across many statistical programs. Despite that, there has recently been much fuss in some quarters about the benefits of this model, as if they were novel. References are suppressed on this point to protect the innocent and the guilty alike.

Other way round, in a previous column (Cox 2009) I reviewed a variety of ideas and methods for working rowwise. Perhaps the most valuable single suggestion, which is repeated here but not elaborated beyond this paragraph, is that the impulse to work rowwise may signal that your dataset layout (format, if you wish) is not ideal for Stata purposes. Often on Statalist and elsewhere, people have panel or longitudinal data in a wide layout in which observations define individuals (patients, firms, countries, sites, etc.) and different variables define various properties at different times. The answer here is usually to `reshape long` so that times as well as identifiers define observations.

This column is an update on that previous column. First, it is easy and congenial to flag that official Stata added `rowmean()` and `rowpctile()` functions to `egen` in Stata 11 in 2009, just a few months after the column was published. Section 5 of the column discussed an `egen` function for a row median written by me. That section 5 may retain some interest, but in practice users naturally should reach for the official functions (unless, exceptionally, they are on Stata 9 or 10).

I assume that you are broadly familiar with loops in Stata. If you are not, and in any case, the previous column (Cox 2009) may be worth reading first, at least briskly.

## 2 Which variables hold the extremes in a row?

The remainder of this column focuses on a single class of problems, studiously avoided earlier. Suppose within an observation, variables `x1`, `x2`, and `x3` have values 3.14159, 7, and 42, respectively. Which variable holds the minimum? Clearly `x1`. Which holds the maximum? Clearly `x3`. The problem is trivial in this and many other examples, and it is not too difficult to code for such problems.

You can imagine what comes next. What do we do if there are ties? When one fills in surveys, particularly when people are asked to give opinions or beliefs on integer scales, ties are not just common but highly likely. It seems that some people even give the same response (say, 3 3 3 . . . ) on purpose, perhaps out of boredom or as a personal protest of some kind. Here we bite the bullet, grasp the nettle, and seize the bull by its horns and show how to program return of variable names even if ties are present.

As a sandbox dataset, we use what appear to be some real data on percent marks (grades, if you prefer) for a group of students on five mathematics examinations in an unnamed institution in an unnamed year. The anonymity here is all to the good. The data are from Mardia, Kent, and Bibby (1979, 3–4) and have been used by many statistically minded writers, typically for purposes or with methods much more subtle than in this column.

In essence, even if we used the functions `max()` or `min()` to find the maximum or the minimum, we still need to loop over the variables to find precisely which variable or variables are equal to either, so we might as well loop in the first place. Similarly, those familiar with `rowsort` and `rowranks` (Cox 2009) might be tempted to start there, but the approach here is more direct.

The dataset can be read in from the media for this column.

```
. use mathsmarks

. describe

Contains data from mathsmarks.dta
  obs:            88
 vars:             5                          27 Jan 2020 18:32
────────────────────────────────────────────────────────────────────────────────
              storage   display    value
variable name   type    format     label      variable label
────────────────────────────────────────────────────────────────────────────────
mechanics       byte    %10.0g
vectors         byte    %10.0g
algebra         byte    %10.0g
analysis        byte    %10.0g
statistics      byte    %10.0g
────────────────────────────────────────────────────────────────────────────────
Sorted by:
```

Evidently, we have splendidly evocative variable names like `mechanics` and `statistics`, rather than tediously unoriginal names like `x1` and `x2`.

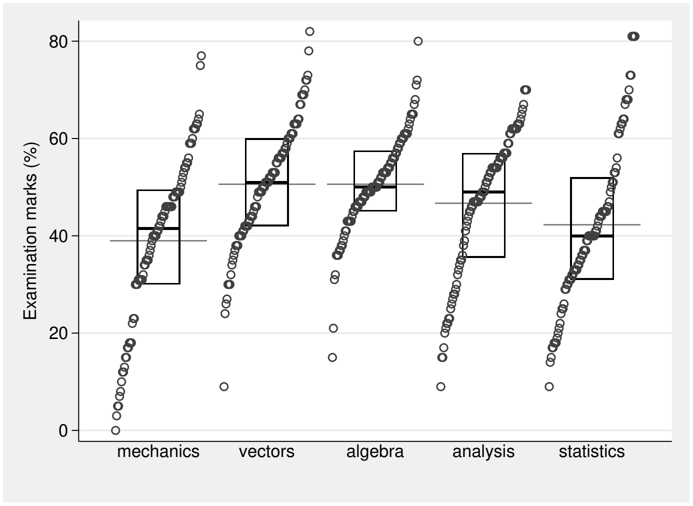

Figure 1. Distributions of the sandbox dataset. Quantiles, median and quartile boxes, and extra reference lines showing means.

No harm comes from looking at a graph. Figure 1 is a quantile-box plot (Parzen 1979). Each set of marks is plotted in ascending order (a quantile plot) with cumulative probability scales implicit on the horizontal axis. Boxes show median and quartiles, as in a conventional box plot. Plotting all the data points underlines the key point about such a box: 50% of the data points for each variable lie within each box (and so 50% lie outside). Finally, horizontal reference lines show the mean for each variable. The details of how to get such a plot are suppressed for a forthcoming column. Anyone intrigued by the result can download the program and help file using (Cox 2003)

```
. ssc install stripplot
```

Apart from allowing the usual kind of assessment of the distributions here—very simply, the variables are well behaved and, as advertised, percent marks—the graph serves as propaganda for a point of view. There is often, especially with small or modest datasets, scope for showing all the data without making the display too busy or too complex.

Here is a chunk of code confronting the ties issue that will be followed by commentary.

```
. generate which_max = ""
(88 missing values generated)
. generate which_min = ""
(88 missing values generated)
. generate max = -1
. generate min = 101
. foreach v of var mech-stat {
  2.   replace which_max = "`v´" if `v´ > max
  3.   replace which_max = which_max + " `v´" if `v´ == max
  4.   replace max = `v´ if `v´ > max
  5.   replace which_min = "`v´" if `v´ < min
  6.   replace which_min = which_min + " `v´" if `v´ == min
  7.   replace min = `v´ if `v´ < min
  8. }
```

The general logic is first to initialize variables to hold the minimum and maximum values and the names of the variables in which they occur. Taste can supervene here in detail. Some people would use the name and values of the first variable in the loop, on the grounds that they could hold the minimum, the maximum, or both; and if they do not, we will change our minds accordingly. Remember that in different observations, a particular variable may hold both row minimums and row maximums; remember also that the loop rowwise includes implicitly a loop across observations. I have no feelings or recommendations against using the first name or the first value in initialization. Indeed, it is my practice sometimes, as a matter of caprice.

Caprice or not, a flag should be waved vigorously here. A `float` storage type, the default, works fine for this example for new variables. The original data are stored as `byte`, as `describe` told us. In other problems, be careful to think about whether a `double` or `long` is needed.

Here the code deliberately uses impossible values to initialize. We know that the marks range from 0 to 100 in principle and a quick plot, such as figure 1, or a `summarize` confirms that no marks occur outside that range. So, −1 as an initial value for maximum is safe because we expect to find at least one higher value. Similarly, 101 is safe for minimum because we expect to find at least one lower value. But why, you may ask, not use, say, 0 and 100 as initial values? There is not much in it, but I have sometimes found that initializing with impossible values makes logic bugs a little easier to catch. If the initial values are unchanged by a look at the data, either the data are not as expected or the code is.

The overarching logic inside the loop has one theme. If the value we are looking at now is greater than the maximum so far, it is the new maximum. Symmetrically, if the value we are looking at now is less than the minimum so far, it is the new minimum.

The tricky detail inside the loop concerns the handling of ties. If the value we are looking at now is as extreme as any seen yet, so equal to the maximum or minimum so far, then we add its name to the name or names we have stored so far. Again, this addition does or does not occur within individual observations, depending on the values of the variables.

Note a crucial tiny point: two or more variable names should be separated by spaces. Other punctuation such as commas is not of order but may not be so convenient. We need not worry about being forced into a dead end with no retreat, because a fresh minimum or maximum will lead to overwriting the names tentatively stored.

I would slap a `quietly` on the entire loop, once I am happy that it is working.

Let's look at the results.

```
. tab1 which_max  which_min
-> tabulation of which_max
          which_max |      Freq.     Percent        Cum.
--------------------+-----------------------------------
            algebra |         19       21.59       21.59
           analysis |         17       19.32       40.91
analysis statistics |          1        1.14       42.05
          mechanics |          7        7.95       50.00
  mechanics algebra |          1        1.14       51.14
  mechanics vectors |          1        1.14       52.27
         statistics |         13       14.77       67.05
            vectors |         28       31.82       98.86
    vectors algebra |          1        1.14      100.00
--------------------+-----------------------------------
              Total |         88      100.00
-> tabulation of which_min
                   which_min |      Freq.     Percent        Cum.
-----------------------------+-----------------------------------
                     algebra |          1        1.14        1.14
            algebra analysis |          1        1.14        2.27
                    analysis |         12       13.64       15.91
                   mechanics |         34       38.64       54.55
           mechanics vectors |          2        2.27       56.82
mechanics vectors statistics |          1        1.14       57.95
                  statistics |         33       37.50       95.45
                     vectors |          4        4.55      100.00
-----------------------------+-----------------------------------
                       Total |         88      100.00
```

The results are not too complicated. Four students had two subjects at which they were equally good, and four had two or three subjects at which they were equally bad. Knowing that ties are uncommon, we might be confident at setting aside the ties and focusing on rank order. Because the `which*` variables are string, and spaces were used for separation, a convenient function for finding single subjects is `wordcount()`. Stata's rule that variable names must be single words frustrates some goals but provides the needed simplicity for this purpose. `wordcount()` is in my personal list of functions experienced users should know (Cox 2011). As alluded to earlier, commas or other punctuation apart from spaces do not allow this simple device.

```
. tabulate which_max if wordcount(which_max) == 1, sort
          which_max |      Freq.     Percent        Cum.
--------------------+-----------------------------------
            vectors |         28       33.33       33.33
            algebra |         19       22.62       55.95
           analysis |         17       20.24       76.19
         statistics |         13       15.48       91.67
          mechanics |          7        8.33      100.00
--------------------+-----------------------------------
              Total |         84      100.00
```

```
. tabulate which_min if wordcount(which_min) == 1, sort

                              which_min |      Freq.     Percent        Cum.
----------------------------------------+-----------------------------------
                              mechanics |         34       40.48       40.48
                             statistics |         33       39.29       79.76
                               analysis |         12       14.29       94.05
                                vectors |          4        4.76       98.81
                                algebra |          1        1.19      100.00
----------------------------------------+-----------------------------------
                                  Total |         84      100.00
```

It looks bad for mechanics and statistics. Teachers (some of us) and past or present students (all of us) know the Law of Interpreting Exam Results: it is always the teachers' fault. Good marks? The teaching was too elementary, the exam was too easy, or you were lucky that so many smart students chose your class. Bad marks? The teaching was poor or the exam was too hard. Naturally, there are no students who are not smart.

Back to the problem: let's imagine a variant in which we are happy to record the maximum or maximum the first time it occurs. There might be some rationale for this, say, that variables have some time order and "first occurrence" is intrinsically interesting. Or it is just a variant on the problem, and we are thinking through the logic of the code.

The code is simpler, and we will not show results here, partly to save space and partly because there is no information that the variables in the example dataset have any kind of natural order. The initializations are the same; it is just the loop that differs.

```
. foreach v of var mech-stat {
  2.   replace which_max = "`v´" if `v´ > max
  3.   replace max = `v´ if `v´ > max
  4.   replace which_min = "`v´" if `v´ < min
  5.   replace min = `v´ if `v´ < min
  6. }
```

In other words, the idea that the first variable met is the only variable that counts means that we ignore ties, ties being with a maximum or minimum previously observed.

As a final flourish, what about the opposite problem, in which only the last occurrence is remembered?

```
. foreach v of var mech-stat {
  2.   replace which_max = "`v´" if `v´ >= max
  3.   replace max = `v´ if `v´ > max
  4.   replace which_min = "`v´" if `v´ <= min
  5.   replace min = `v´ if `v´ < min
  6. }
```

Here we overwrite names if we see a new instance of the maximum or minimum so far. Hence, we exit the loop with the last occurrence of a maximum or minimum recorded as the only occurrence remembered with a name.

## 3 Conclusion

Apart from the belated update flagging the introduction of `egen` functions `rowmedian()` and `rowpctile()` in Stata 11, this column has focused on one further problem. We wish to record not just the row maximum and row minimum across a bunch of numeric variables but also the variables in which they occur. The aim calls for a loop across variables, use of string variables to store the results, and careful attention to the possibility of tied values.

## 4 References

Cox, N. J. 2003. stripplot: Stata module for strip plots (one-way dot plots). Statistical Software Components S433401, Department of Economics, Boston College. https://ideas.repec.org/c/boc/bocode/s433401.html.

———. 2009. Speaking Stata: Rowwise. *Stata Journal* 9: 137–157. https://doi.org/10.1177/1536867X0900900107.

———. 2011. Speaking Stata: Fun and fluency with functions. *Stata Journal* 11: 460–471. https://doi.org/10.1177/1536867X1101100308.

Mardia, K. V., J. T. Kent, and J. M. Bibby. 1979. *Multivariate Analysis*. London: Academic Press.

Parzen, E. 1979. Nonparametric statistical data modeling. *Journal of the American Statistical Association* 74: 105–121. https://doi.org/10.2307/2286734.

### About the author

Nicholas Cox is a statistically minded geographer at Durham University. He contributes talks, postings, FAQs, and programs to the Stata user community. He has also coauthored 16 commands in official Stata. He was an author of several inserts in the *Stata Technical Bulletin* and is an editor of the *Stata Journal*. His "Speaking Stata" articles on graphics from 2004 to 2013 have been collected as *Speaking Stata Graphics* (2014, College Station, TX: Stata Press).