The Stata Journal (2006)
6, Number 1, pp. 147–148

Stata tip 29: For all times and all places

Charles H. Franklin
Department of Political Science
University of Wisconsin–Madison
Madison, WI
chfrankl@wisc.edu

According to the Data Management Reference Manual, the cross command is “rarely used”; see [d] cross. This comment understates the command’s usefulness. For example, the fillin command uses cross (Cox 2005). Here is one further circumstance in which it proves extremely useful, allowing a simple solution to an otherwise awkward problem.

In pooled time-series cross-sectional data, we require that some number of units (geographic locations, patients, television markets) be observed over some period (daily from March to November, say). We thus need a data structure in which each unit is represented at each time point. If the data come in this complete form, then no problem arises. But when aggregating from lower-level observations, some dates, and possibly some units, are often missing. This missingness could be because no measurement was taken or because an event that is being counted simply did not occur on that date and so no record or observation was generated. In the aggregated Stata data file, no observation will appear for these dates or units. Inserting observations for the missing dates or units is awkward, but the cross command, followed by merge, makes the solution simple.

To illustrate with a real example: in the Wisconsin Advertising Project, we have coded 1.06 million political advertisements broadcast during the 2004 U.S. presidential campaign, using data provided by Nielsen Monitor-Plus. These ads are distributed across 210 media markets. Each time an ad is broadcast, it generates an observation in our dataset. The data are then aggregated to the media market to produce a daily count of the total advertising in each market. Such aggregation is simple in Stata. Variables repad and demad are coded 1 if the ad supported the Republican or Democratic candidate, respectively, and 0 otherwise. The sum is thus simply the count of the number of ads supporting each candidate.

```
clear
use allads
sort market date
collapse (sum) repad demad, by(market date) fast
save marketcounts, replace
```

This do-file produced no observation if no ads ran in a market on a particular date, which is common in these data. We want a dataset that includes every date for each of the 210 markets, with a value of 0 if no ad ran in a market on a date.

We can use cross to create a dataset that has one observation for each market for each of the 245 days included in our study. The file dmacodelist.dta contains

© 2006 StataCorp LP
one observation for each of the 210 markets: dma stands for “designated market area”, Nielsen’s term for television markets. First, we create a Stata dataset with 245 observations, one for each day of our study (March 3–November 2). Then we convert this information to a Stata date.

```stata
clear
set obs 245
gen date = _n + mdy(03,02,2004)
format date %d
```

Now use `cross` to generate the dataset with all dates for all markets:

```stata
cross using dmacodelist
sort market date
save alldates, replace
```

The file `alldates.dta` contains one observation for each market and for each date. The last step is to merge the aggregated `marketcount.dta` dataset with `alldates.dta` and replace missing values with zeros.

```stata
clear
use marketcounts
sort market date
merge market date using alldates
assert _merge != 1
replace demad = 0 if demad == .
replace repad = 0 if repad == .
```

The merge should produce no values of `_merge` that are 1, meaning observations found only in `marketcounts`, so the `assert` command checks this: the do-file will stop if the assertion is false (see Gould 2003 on `assert`). The `repads` and `demads` will be missing in the merged data only if no ad was broadcast, so replacing missing values for these variables with zeros will result in the desired dataset.

Thus the `cross` command offers an efficient solution to this type of problem. Those who often aggregate low-level data to create time-series cross-sectional structures will find this command handy.

**References**
