Notes on Stata: Part II

Thomas Lumley

October 27, 2003

These notes follow on from Part I by describing how to get various summary statistics and do some data manipulations. As in part 1, `this font` is used for things you type or results that Stata prints, and this font is used for menu and dialog items.

Basic summary statistics: The `summarize` command gives most of what you need

```
summ ozone temp wind
```

It lists the number of non-missing observations, the mean and standard deviation, the minimum and maximum. Adding the `detail` option gives many other statistics, including the kurtosis and skewness and the median and quartiles.

The median and other percentiles can be computed with the `centile` command

```
centile ozone, c(90 95 97)
```

Statistics such as the interquartile range or Pearson's skewness coefficient need some extra processing. You can use Stata as a calculator with the `display` command: here we compute the upper and lower quartiles and then the interquartile range

```
centile ozone, c(25 75)
display 63.75-18
```

Similarly, we can obtain the mean, median and standard deviation and compute the Pearson coefficient of skewness

```
*mean (42.129) and std deviation (32.988)
summ ozone
```

```
* median (31.5)
centile ozone, c(50)
*combine them: note that * is multiplication
display 3 * (42.129 - 31.5)/32.988
```

Defining right-censored data: Stata has quite flexible analysis of right-censored data, but it can handle only one censored variable at a time. The first step is to declare a censored variable, which is done with `stset`

```
label define inrem 1 "no" 0 "yes"
infile ptid nadirpsa pretxpsa ps bss grade age obstime inrem:inrem using
        http://courses.washington.edu/b517/datasets/psa.txt
stset obstime, fail(inrem)
```

For simple right-censored data you need to specify the observation time, and then the `fail` option with the failure indicator (1 for event, 0 for censoring). The `stset` command will give a message with some summary information[1] It is now possible to get graphs, quantiles, and other summaries of the survival distribution. Note that you don't specify the outcome variable explicitly in any of these commands. Stata will use the variable specified by `stset`.

```
* quartiles, by tumour grade.
stsum, by(grade)

* survival function graph
sts graph
* same, by bone scan score
sts graph, by(bss)

* hazard rate graph
sts graph, hazard
```

Scatterplots and smoothers These are most easily done with the menus in Stata 8. To use different plotting symbols for different subgroups or to draw stratified lowess smooths use Graphics | Overlaid twoway graphs. You will need a separate overlaid plot for each subgroup. For example, to plot age and FEV1 for smokers and non-smokers in different colors and to add a separate lowess smooth for smokers and non-smokers takes four plots. Begin by creating jittered versions of age

```
infile seq id age fev height sex smoke using http://courses.washington.edu/b517/datasets/fev.t
gen agej= age + uniform()*2/3 -1/3
```

[1]There are some quite complicated options for `stset`, mostly for handling data with multiple records per person. The summary information is very useful in these more complex situations

1. In plot 1, X is `agej`, Y is `fev`, if is `smoke==1`
2. In plot 2, set type to scatter, then X is `agej`, Y is `fev`, if is `smoke==2`.
3. In plot 3, set type to lowess, then X is `age`, Y is `fev`, if is `smoke==1`.
4. In plot 4, set type to lowess, then X is `age`, Y is `fev`, if is `smoke==2`.

This creates the Stata command

```
twoway (scatter fev agej if smoke==1) (scatter fev agej if smoke==2)
       (lowess fev age  if smoke==1) (lowess fev age if smoke==2)
```

I wouldn't try to construct this by hand, but I might edit it by hand (use the PageUp key to get the last command back again) if I wanted to replace age with height (rather than going through the menus again).

Repeated measurements In the faculty salary data we had multiple measurements on each person and needed to compute statistics within a person, eg finding starting salary or percentage increases. We did this using the `by` command and the magic variables `_1` and `_n`.

The starting salary for each person was created by

```
* create a variable that is missing for everyone
gen startsal = .
* sort by person and then by year within each person
sort id year
* within each person the starting salary is now the first observation
quietly by id: replace startsal = salary[_1]
```

The notation `salary[_1]` refers to the value of `salary` for the first observation in the file, but the `by id:` prefix means that we consider each `id` (each person) to be a separate file, so that `salary[_1]` is the value of `salary` for the first observation for that person.

We also wanted to compute previous year's salary

```
* create a variable that is missing for everyone
gen prevsal = .
* sort by person and then by year within each person
sort id year
* within each person the starting salary is now the first observation
quietly by id: replace prevsal = salary[_n-1]
```

The notation `salary[_n-1]` means the value of `salary` in the previous record in the file. Since the records are sorted by id and year, that is the previous year's record for the same person. Since we are considering each id to be a separate file, there is no previous record for the first year for each person, so the first year for each person has `prevsal` missing.

Recoding With the Seattle air pollution data we want to define a variable that indicates the winter heating season, say November to March. There are many ways to do this. Four of them are:

```
gen heat1=mo
recode heat1 1/3=1 4/10=0 11/12=1

gen heat2= (mo<=3) | (mo >=11)

gen heat3= (mo==1) | (mo==2) | (mo==3) | (mo==11) | (mo==12)

gen heat4 = (mo<4)
replace heat4=1 if (mo>10)
```

Another common recoding task is dividing a variable into groups at specified cutpoints. The `egen cut` command simplifies this

```
egen pmgroup= cut(pm1), at(0 20 35 100) label
```

Note that you must provide lower and upper limits for all categories, not just the cut points between categories. The `label` option asks Stata to define sensible value labels for the new variable.

A variant of the same command can be used to divide a variable into groups with equal numbers of observations in each group:

```
egen bili4 = cut(bili), groups(4) label
```

creates four groups for serum bilirubin, each with (approximately) the same number of subjects.

Stata for statistical tables: When computing power and sample size and confidence intervals you may need probabilities from various mathematical distributions. These are not quantities that you would compute for each record in the data set; you just want one number.

- Probability that a Normal distribution is less than 2 standard errors above its mean? `display norm(2)`
- What threshold (in standard errors) is a Normal distribution below 90% of the time? `display invorm(0.90)`
- Probability of 5 or more successes in 7 attempts each with probability 0.6? `display Binomial(7,5, 0.6)`