Econ 452 Section 1 - Getting Started with STATA

Connor Cole

October 1, 2015

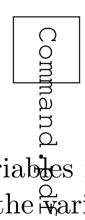
Accessing STATA

Most of the public computers at the University of Michigan have STATA on them, along with a host of other statistical packages. If you are working off campus and would like to use STATA, you should be able to do your work using Virtual Sites.

Please be careful to save your work on MBox or your IFS Space files, links to which should be visible on your Virtual Sites desktop. Any work you save to your 'Desktop' on the computer you link to through a Virtual Sites connection will not be there when you log on next.You may access Virtual Sites here: http://www.itcs.umich.edu/sites/labs/virtual.php

STATA Layout

Opening STATA, you will see a screen with five windows:



1. Variables Window: This window all variables in the dataset by name, including a variety of labels created to make the meanings of the variables clearer. Variable names are how STATA recognizes variables in the data, and how you will refer to variables when manipulating them in the command line. Variable labels offer descriptions that are helpful for the user. Note that double clicking a variable will make it present in the command line.

2. Variable Properties Window: This window reports all information about a selected variable from the variables window, and information about the total dataset. We will generally not use information from this window.

3. Command Line: This window is where you will type all commands you will use to make STATA perform a task, including opening datasets, managing the data, reporting tables or computing statistics.

4. Results Window: This window displays each command you've used and the output. It reports every table, statistic, test, regression output or any other computation with the data other than graphs, which will appear in a separate window.
5. Command Review: This window lists all previous commands you've entered.

Getting Help

If you're having a difficult time with STATA or want to learn more about how to use it than is being covered in this class, there are many resources that can help you get better.

1. The help command: This command explains how to use another STATA command. For example, if we want to better understand the sum command, we can type help sum, and a window comes up describing the sum command, including syntax, a variety of options of ways we could use the command, and examples of how to use the command.
2. The search command: The search command looks through all help topics related to a particular topic. For example, if we would like to know more procedures related to regression in STATA, we could type search regression.
3. STATA Documentation: STATA has a large documentation (which many professors used to have in hard copy in something like 12,000 pages) which is easily accessible from either using the help command and clicking on the blue link under Title, or available online: http://www.stata.com/features/documentation/
4. Other online sources: If you are having a problem with using STATA, there's a high likelihood that someone else has dealt with a similar problem. There's a lot of information online about how to use STATA that spans from basic programming advice to more advanced topics. A good place to start if you haven't used STATA before would be the resources available from UCLA (http://statistics.ats.ucla.edu/stat/stata/) and from the University of Wisconsin (https://www.ssc.wisc.edu/sscc/pubs/sfr-intro.htm), which offer sequential STATA tutorials along with answers to many of the basic frequently asked questions you might have.

   If you are more experienced with STATA and dealing with a more specialized topic, another place you might look at would be the resources on Statalist, (http://www.statalist.org/) an active forum of STATA users where users discuss their techniques for dealing with various problems. Be forewarned, though, you shouldn't post a question to it unless that question hasn't been addressed there or on some other basic frequently asked questions list.

Opening and summarizing a dataset

Opening a dataset is easy in STATA. The most straightforward way to do it is to click on File ¿ Open in STATA and select your dataset. Note that for this technique to work, your data must already be in STATA format. It is straightforward to import data from Excel spreadsheets, CSV files, and certain text files, but we will not be using these features.

A second way to open a dataset is to use the command line. First, you will need to make sure the memory is clear, so type:

```
clear
```

This command removes all data from STATA by clearing the memory of your current work session (note that it doesn't 'clear' the file, which remains unchanged unless it is resaved with changes made to it). Then, type:

```
use "C:/FILE EXTENSION/FILENAME.dta"
```

When you have a dataset open, the variables from that data will be present in the Variables window described earlier. To see a list of the variables, variable properties and the file information, type:

```
describe
```

This command offers a detailed rundown of the nature of the data without statistics about the variables. To see the relevant information for a smaller amount of the data, type:

```
describe variablename1 variablename2
```

Results in the Results Window

When STATA output for a particular command is longer than the screen, STATA will pause and display -more- in the lower left corner of the Results window. To print more output from the command, you can click the `more` button or press space on the keyboard.

Alternatively, if processing the command is taking too long, or if you don't want to read the rest of the output from a particular command, you can either type `q` or click the red button with an 'x' in the middle of it, as those actions will stop the command from running further.

Visualizing Data

You have probably noticed that STATA, unlike Excel, does not offer a spreadsheet of data. Over time, you will find this feature of the way the program works more convenient, but it may be confusing now. In general, you'll be less concerned with individual values of data than with statistics you compute with the whole dataset. However, if you'd like to see all observations, you can type:

```
list
```

If you'd like to see all observations for a particular variable, you can type:

```
list variablename1 variablename2
```

Given the size of the output, this command is often unwieldy, and an easier way of looking at your data might be to look at it as a spreadsheet, which you can do if you type:

```
browse
```

As with the list command, you can also restrict displaying of variables to a smaller by typing:

```
browse variablename1 variablename2
```

You can also accomplish this by clicking on the "Data Editor (Browse)" button on the toolbar.

Do-Files

At first, it will likely seem easier to use the toolbars to manipulate the data and start commands than to memorize commands and write them in yourself. I remember it seemed that way to me at first as well. However, there are two major advantages to learning and memorizing STATA commands. First, it will be considerably faster and easier with time. Second, if you know commands, you will be able to use a critical tool in STATA, the do-file editor. A do-file is a text file that can hold commands in a list and run them as if they were put into the command file. A good thing about do-files is that you can save the list of commands and use it to rerun the set of commands in the future. If you ever work in research, all of your work will be done in do-files so that your work is easy to review for colleagues and people who are reviewing your results.

To start a do-file, go to Window > Do-file Edictor > New Do-file, or click the 'New Do-File Editor' icon on the toolbar. You can then type all of your commands in the do-file and run them by clicking 'Execute (do)' on the do-file editor, or typing ctrl+D

For example, if you wanted to open a dataset, disable the 'more' message and describe the dataset, you would create a do-file and type:

```
clear all

set more off

use "C:/FILE EXTENSION/FILENAME.dta"

describe
```

In addition to saving your do-file, you also might want to save your log-file to see the output from your do-file. A log-file will save all the Results window output from STATA in a text file. To create a log file, go to File > Log > Begin, and choose the location where you want STATA to save the file. You can also type:

```
log using C:/FILE EXTENSION/FILENAME.log
```

To end a log, go to File > Log > Close, or type:

```
log close
```

Data Structure in STATA

Before discussing how to manipulate and examine data in STATA, it is helpful to first talk about how data is organized in STATA. It is helpful to think of each variable as a 'column' in a table where each row is an individual observation. Indeed, this is how the data will appear when using the `browse` command. Thus, operations that we do to variables are essentially operations to columns of the data across observations.

Basic Data Analysis

To see basic descriptive statistics of all variables in the dataset, including the number of observations, the mean, the standard deviation and the minimum and maximum, type:

```
summarize
```

We can see these basic descriptive statistics for individual variables at a time by typing:

```
summarize variablename1 variablename2
```

More basic descriptive statistics, including percentile distributions of the data, are available if we type

```
summarize variablename1 , detail
```

Note that some variables might not have summaries available. This command returns values for variables that have numeric values attached to them. It will not return anything for string variables, that are defined by text and not by numbers, or 'missing' observations, that are recorded in numeric variables as a period.

The `summarize` command obviously work well when dealing with variables that are continuous. However, for variables that are categorical, the 'mean' of a categorical variable that can take on values one through five is not very informative. Instead, it is much more instructive to see a list of all the categorical values that observations take on. To see a list of values, type:

```
tab variablename1
```

If we have two categorical variables, and we'd like to see the number of observations in each of the categories created by the different categories, type:

```
tab variablename1 variablename2
```

You'll note that sometimes categorical variables have text labels attached to them but also appear to have numerical values attached to them. The underlying values of these variables are numeric, but text labels have been added to the numeric values. To see the underlying numeric values that STATA sees for these variables, type:

```
tab variablename1, nolabel
```

Note that the tab command works for both numeric variables and for string variables that are composed of text only.

Conditioning Statements

So far we have only considered commands that calculate statistics for all observations in a dataset. Often, we're also interested in statistics for subgroups; for example, you might want to know the mean of earnings for individuals with a college education and compare it to individuals who do not have a college education. The ease of doing these sorts of calculations in STATA is one of the features that makes it much more convenient than doing similar work in, for example, Excel.

In STATA, if we wanted to do some command operation for a subgroup, we would put the relevant command for whatever variable of interest we are considering (e.g. earnings, in the previous example) and then, at the end, include an additional 'if' statement that clarifies the subgroup at which we want to look by describing that group with the other variables. For example, if we wanted to get some basic descriptive statistics for a variable for the subgroup of observations where a second variable is greater than or equal to 2 we would write:

```
summarize variablename1 if variablename2=>2
```

If we wanted to calculate similar statistics for the subgroup where the second variable is equal to two, we would write:

```
summarize variablename1 if variablename2==2
```

Note that the equals sign appears twice in the 'if' statement for this kind of conditioning.

If we wanted to calculate similar statistics for the subgroup where the second variable is not equal to two, we would write:

```
summarize variablename1 if variablename2!=2
```

If we wanted to calculate similar statistics for the subgroup where the second variable is less than two, we would write:

```
summarize variablename1 if variablename2<2
```

We can can further select smaller subgroups by conditioning on multiple variables at the same time and combining the conditioning statements with either an 'and' sign (&) or an 'or' (|) sign.

If we wanted to calculate statistics for the subgroup where the second variable is less than two and some third variable is equal to 0, we would write:

```
summarize variablename1 if variablename2<2 & variablename3==0
```

It is good programming practice to use parentheses to separate conditioning statements if one conditioning argument may include multiple elements.

Note that an alternative way of conditioning would be to create a 'dummy variable,' or a variable that takes on values 0 and 1, that defines the group of interest, and perform operations on that subgroup.

Dropping Observations

In our problem sets, we will often drop observations if they are missing values for variables of interest (note that some researchers, instead of dropping missing observations, will sometimes 'impute' values to missing variables). Note that dropping an observation is equivalent to dropping a row in the data. A missing observation in STATA is recorded, in the variables we use, as a "." (our data has no string variables that are composed of text information only. For these variables, a missing observation is recorded as a "") Thus, dropping an observation because it is missing some variable would be coded as:

```
drop if variablename1==.
```

Variable / Dataset Management

Simple Commands

When you open a dataset, there are a set of variables listed in the variables window. Often, in the course of analyzing the data, you will want to create additional variables that are functions of the existing variables. First, we consider 'simple' operations. If we wanted to add two variables together to create an additional variable, we would write:

```
generate newvariable = variablename1 + variablename2
```

We could then easily drop this variable by writing:

```
drop newvariable
```

Labeling variables

As was discussed previously, the categorical variables in the dataset have value labels attached to them.

The 'Egen' Command

The previous operations make new variables row by row. The 'egen' command opens up all sorts of calculations of new variables on the basis of all the values in either a row or a column (or whatever section of rows we are conditioning on). For example, if we had data on income and sex, we could use the egen command to calculate standard deviations of income for men and women separately and save it in an additional variable that gives the relevant for standard deviation for each variable. Or, we might want to add up all the values of some variable in a column and save it as an additional variable, or calculate the mean of all the variables in a column. Typing in help egen will give all various functions you can perform on data in a column using the egen function. In general, you will use the egen function by typing `egen newvariable =` and then putting in the relevant function, with the variables it takes as an argument in parentheses. Some functions of interest will be:

If you want to calculate the mean of the values in a column and save it as an additional variable, you would type:

```
egen newvar = mean(variablename1)
```

If you want to calculate the value of some percentile of the distribution of values in a column, for example the median, and save it an additional variable you would type:

```
egen newvar = pctile(variablename1), p(50)
```