Stata Handout

1. Starting Stata

If you use Stata for Windows, starting Stata is straightforward. You just have to double-click on the **wstata** (or **stata**) icon.

If you use Stata for Unix, type at the athena* command prompt:

athena%* add stata
athena%* xstata

Then four windows will appear on your monitor. You will type your commands in the **Stata Command** window. The **Review** window records your commands and the **Variables** window lists the variables in your data set (including the one that you created).

2. Do-files

Problem sets usually require you to turn in your do-file. Do-files simply contain commands that you want Stata to execute. Without do-files, you have to type a command, wait for an execution, and then type another command. This can be inconvenient once your commands become more elaborate. In addition, it is important to create do-files to keep track of what you did. If you don't have a do-file when you want to repeat the same analysis again or modify your past analysis, you will have to type the same commands again.

The way do-files work is very easy: we first create a file that contains all the commands that Stata should execute, and then tell Stata to execute them. So we are just telling Stata what to do all in advance, instead of one by one.

*Athena is MIT's UNIX-based computing environment. OCW does not provide access to it.

To create a do-file, click on the do file button or type

doedit

Then you will see a simple text file. When you finish entering commands, save the do-file. Here you should make sure it is saved in the current directly. Stata looks for data and do-files in the current directory and save new data files there. You can find the location of the current directory at the bottom left hand corner of the window (something like **C:\Data** in Windows). You can change the location of the current directory by the **cd** command, as in **cd temp**, for example (assuming that you have already created a folder called **temp** in **Data** directory). When you have finished typing your code, you can run it by selecting it and hitting the do button. You can run the entire file by typing

do *filename*

filename can be any name. Do-files have **.do** as an extension, but **.do** is assumed when you enter **do** or **doedit**, so you don't have to explicitly spell out your filename as **do sample.do**; **do sample** is sufficient. If you want to modify your do-file, type **doedit *filename***.

3. Log files

You can record all results that appear in the **Results** window in a log file. There are two versions of log-files: one is Stata formatted (smcl) and another one is a text file. You can read log-files written in smcl only in the viewer of Stata and smcl-style logs contain some annoying style, so I recommend that you use a text file version of log-file. To start recording your output, type

log using *filename*.log

Typing **.log** at the end is important. If you don't add the extension, Stata creates smcl-style log files. When you are done with the analysis, type

log close

Of course, you can include these commands (**log using…**and **log close)** in your do-file. When you do so, it is safe to enter **log using …, replace** (see below). Alternatively, you can start a log, then run your do-file that does not contain the **log using.. command**.

If you make a mistake in your do-file and receive an error message, Stata stops the procedures. Stata does not automatically close the log, so if this happens, close the log first (you can check if the log is open by looking at the bottom of the **Stata Results** window – if it's open, it says **log on**) and then type

log using *filename*.log, replace

This command overwrites the existing log file. In some cases, you may want to add more results to the existing log file. Then you should make sure the log is closed and type

log using *filename*.log, append

Since text log-files are just text files, you can read and print them using text editors (such as emacs in Unix or Notepad in Windows). Log files are stored in the current directly.

4. Some Additional Commands

Some basic commands that you use to do Problem set 2 are included in the questions. Here are some other commands and facts that you may need to know.

(1) Creating variables

To create a variable, type

gen *varname=expression*

For example, to create a standard normal variable and call it **x**,

gen x=invnorm(uniform())

Note that, obviously, you cannot create a variable that has the same name as the existing variable. Suppose that you first set the number of observations to 50 and then create a variable called **x** by typing **gen x=invnorm(uniform())**. You are done with obs=50, so you change the number of observation to 100 (**set obs 100**) and again type **gen x=invnorm(uniform())**, then you will get an error message. In this case, you have three options: (1) clear the data, (2) remove variable **x**, or (3) or explicitly tell Stata to create a new variable with the same name (i.e. overwrite **x**). To do (1), use command **clear** (see below). To do (2), just type **drop x**. In order to do (3) (i.e. when you want to overwrite **x**), type **replace x=invnorm(uniform())**.

(2) Checking the data

When you want to see the variable that you created, type

br

(equivalent to typing **browse**).

(3) Clearing the memory and dropping variables

Once you are done with the data, you can clear the data. To do so, type

clear

Note that this will erase all the variables in the data. If you want to remove some variables in the data and keep all the other variables, type

drop *varname*

(4) Graphs

To draw a graph and save it, type

histogram *varname*, saving(*graphname,* replace) [Stata 8]

For example, if the name of the variable is **x** and if you want to call the graph **graph1**, the above command would look like;

histogram x, bin(100) normal saving(graph1, replace)

I would add **replace** in **saving(..)** for the same reason that **replace** is added in the **log using..** command. The **normal** command adds a nice normal curve and is optional. The **bin( )** command is also optional and you can specify different numbers. You probably want to add the **bin( )** command only when the sample size is very large and the graph does not look very smooth. (Note that in Stata 7, you can increase your bin only up to 50.)

To look at the graph after your do-file is done,

graph use graph1 [Stata 8]

Note that depending on the version of Stata, you have to use a slightly different command. To print the graph, click on the **File** in the header bar (when the graph is still open) and then choose **Print Graph**.

(5) Exiting Stata

When you are done, make sure your data is cleared (**clear**), then just type **exit**.

17.871 Political Science Laboratory
Spring 2012