Basics of STATA

This handout is intended as an introduction to STATA. STATA is available on the PCs in the computer lab as well as on the Unix system. Throughout, bold type will refer to STATA commands, while file names, variables names, etc. will be in normal type.

1 Data files

Variables within a data set are typically organized in columns, while rows represent different observations of a given variable. An important feature of data sets is their format. Our data sets will come either in the ASCII (text) format or in STATA format. Both formats are compatible with STATA.

Data can either be stored in a separate file - which we will call DATA - or typed in when using STATA in the interactive mode. Obviously, we won't be typing in long data sets each time we want to analyze them, so we will prefer to store our data in a separate file. In STATA, text format data files have the suffix .RAW while STATA format data files will bear the suffix .DTA (text format data sets may bear another suffix, such as .TXT). So assume that we have a data file, named either DATA.RAW or DATA.DTA.

2 Loading data into STATA

Launching STATA from the Windows menu allows you to enter the interactive mode of the program, which means that you can type commands which will be executed one by one. The prompt '.' (dot) indicates that you are within STATA. The first thing you will have to do is to enter the data in STATA's memory:

- If the data are in text (ASCII) format, the command to perform this is:

INFILE VAR1 VAR2 **USING** c:\path\DATA

where VAR1 and VAR2 (and possibly VAR3...) are names you will give to the variables (columns) which make up DATA. You must specify drive (in this example c:) and the path to the directory where the DATA file is stored (here path). The maximum length of a name is 8 characters.

- If the data are in STATA format, the command is:

USE c:\path\DATA

Unlike text files, STATA format data files already contain variable names, so you should not respecify these.

You can create a STATA format data file from a text format file by first loading the text format data using the INFILE command, and then typing:

SAVE c:\path\DATA

This will create a file called DATA.DTA in your directory.

3 Getting started with data analysis

We are now ready to start analyzing our data set. The first thing we might want to do is to make sure we have loaded the right file, and to get a rough idea of its components. A convenient way to obtain this information is the following command:

DESCRIBE

This gives you some information on the data which STATA has in its memory (number of variables, number of observations, names of variables, etc...).

We may also want to view the data directly on screen:

LIST VAR1 VAR2 displays the variables named VAR1, VAR2.

LIST displays all the variables in STATA's current memory.

LIST IN 1/10 displays the first 10 observations of all the variables in the data set.

Going yet further, we may be interested in summary statistics about the data:

SUMMARIZE provides statistics such as mean, standard deviation of the variables in STATA's memory.

Other potentially useful commands are the following:

SORT VAR1 reorganizes the data in such a way that VAR1 will appear in ascending order. For example, if we have a sample of individuals, we may want to organize our data in ascending order of their income or education levels. Never use this when dealing with time series !

TABULATE VAR1 VAR2 provides frequency tables; if VAR1 is an age group and VAR2 is the education level, it will tell you how many individuals of each age group have a given education level.

CORRELATE VAR1 VAR2 VAR3 provides the autocorrelation table of the listed variables.

GRAPH VAR1 VAR2 provides a scatter plot of the data with VAR2 on the x-axis and VAR1 on the y-axis.

GENERATE NEWVAR=VAR1+VAR2 generates a new variable called NEWVAR, which is the sum of VAR1 and VAR2, and stores it in the sessions memory. Of course, you can create any combination of any number of variables using +, -, *, / etc...

DROP VAR1 removes VAR1 from the sessions memory.

REPLACE VAR1=VAR1+VAR2 replaces the values of VAR1 with the the sum of its old values plus VAR2. This is equivalent to (1) **GENERATE** NEWVAR=VAR1+VAR2; (2) **DROP** VAR1; (3) **RENAME** NEWVAR VAR1.

One more thing... You can combine these commands with logical qualifiers such as if, & (and) and or. For example, you can use:

LIST VAR1 **IF** VAR1>100 **&** VAR2==1

which will display VAR1 whenever the value of this variable is greater than 100 and the value of VAR2 is equal to 1. Note that a single equal sign corresponds to a variables name (as in the **GENERATE** command) while two equal signs are needed when dealing with a given value of a variable (as in VAR2==1).

More commands are of course available; you can get a complete list by typing HELP under STATA, or by consulting the users manual. The best way to learn STATA is through practice.

4 Introduction to statistical analysis using STATA

Least squares regression is one of the essential statistical methods we will be studying in the course. A discussed in the first lecture, this consists of minimizing the vertical distance between the scattered data points and the line we are trying to fit through them. Suppose that we wish to predict VAR1 using VAR2 and VAR3:

VAR1 = β_1 + β_2×VAR2 + β_3×VAR3 + ε

To do this, and to produce many of the useful statistics that go with it, STATA has a very convenient command:

REGRESS VAR1 VAR2 VAR3

You will immediately obtain estimated values for b1 , b2 , b3 , as well as their standard errors, confidence intervals and other useful statistics which have been or will be introduced in class.

To obtain fitted values or regression residuals from this regression, type:

PREDICT FITTED stores the fitted values from the regression in a data column (variable) called FITTED, and keeps it in memory.

PREDICT RESID, RESIDUALS stores the residuals from the regression in a data column (variable) called RESID, and keeps it in memory.

As the course progresses, we will discuss and implement many more statistical procedures; the corresponding STATA commands will be introduced in section or through email.

5 Writing programs and getting output files

The interactive mode requires you to enter commands one at a time, and to get results one at a time. This may be inconvenient if you have many commands to run, or if you make mistakes which require you to type a whole chain of commands over again. In this case, you may prefer to write a program which contains all the commands in order of desired execution. You can then correct, modify and run the program whenever need arises. The easiest way to write programs is the following: Write the program in the Windows editor (notepad or any text editor), one command on each line, and save the program file in a convenient directory. This file must bear the suffix .DO. For example, it can be called PROGRAM1.DO. An example of a possible STATA program is the following:

```
INFILE VAR1 VAR2 VAR3 USING c:\path\DATA
SUMMARIZE
GENERATE VAR4=VAR1-VAR2
REGRESS VAR1 VAR2 VAR4
PREDICT FITTED
LIST FITTED IN 1/20
```

Save this program under the name PROGRAM1.DO, then enter STATA and run the program by going to the file menu and choosing **DO**. This will perform all the commands of PROGRAM1 in the order you typed them in, and provide you with the output on the output window.

Another convenient tool is to store the output of your work (regression results, statistics, transformed data ...) in an output file which can then be printed. Sometimes, the output will be too long to fit on a single screen (as in our example above), so it is convenient to store it in a text (ASCII) file, which you can later view and print. To do this, you can start your session or your program with:

LOG USING c:\path\OUTPUT

This will create an output file in text format, called OUTPUT.LOG, stored in the subdirectory from which you invoked STATA. This file will contain all the results from your session, as well as the commands you typed in the interactive mode. To access this file, simply type EDIT OUTPUT.LOG at the DOS prompt, or use a word processor. Note that the LOG command will not keep track of graphs (which you can print directly from STATA using the **PRINT GRAPH** command). Note also:

LOG CLOSE stops logging a session and closes the text file containing the log.

LOG OFF temporarily stops the logging session without closing the log file.

LOG ON resumes logging on the open log file.

One more thing ! You must save your data before STATA lets you exit:

SAVE c:\path\DATA will save the data in the sessions memory (including new variables, transformed variables...) in a .DTA file (STATA format), on the c: drive (or any other drive you specify) in the path directory.

You cannot use an existing name for the data file, so it is a good idea to delete useless data files from your subdirectory, and keep only the initial data set and/or useful transformed data sets. Its also a good idea to delete old output files, especially if the LOG command is written in your program (again, STATA won't overwrite).