Introduction to Stata

Getting Started

Command Syntax
command *varlist*, option

This is the simple command syntax in Stata and more conditions can be added as shown in the examples.

Preamble

mkdir tutorial /* to create a new directory, tutorial */

cd tutorial /* to change directory to tutorial */

This new directory will be stored under C:\Data directory.

As was done in class, because the software is loaded onto a common drive, each time you start a session, you have to ask Stata to read from the correct drive. Since most of you store your data on your h drive, in all your do-files, you have to start it with the following as an example,

cd h:*foldername*

where *foldername* is for the name you give to your data folder.

Setting Memory Size

set memory 300K

By default, Stata allocates 1 MB to data area. If the dataset is big, set a bigger memory for Stata to handle the data properly. The example has set it to 300K but it can be as large as your computer can handle. However note that the more processors you are using concurrently, the less memory you can allocate to Stata. So if you are working on a large data set, it is advisable that you shut down, i.e. exit from all unnecessary processes that would draw memory.

Do-files

do *filename* /* to execute a text file .do that stores sequence of Stata commands. */

Use do-file editor to create do-files in Windows. The first line in the do-file should declare the Stata release under which you wrote the do-file, e.g. version 9.0. It ensures that the do-file will continue to work with future versions of Stata. Place any comments in the do-file between * * delimiters. The document is saved as a *filename*.do

Note: It is a very good habit to make notes on your program using the delimiters **content at the end of a line of a program. This reminds you why you wrote that line.**

Because there is ultimately a row limit to the do-files, you can write your programs on any text processor such as word pad. Here the filename is saved as *filename*.txt. When you call on a do text file, you must tell Stata what format of a file you are asking it to read, so that you have to type the following in the command window;
do *filename*.txt

It is always advisable to keep all your programs with each genesis so that you can track what you did before. You will understand when you leave a program with more than 100 lines you wrote a month back, and rereading a month later spending hours trying to figure out why you did something.

Keeping Logs

log using tutorial 1.log /* to open a log file, tutorial 1.log */

log using *filename*

log on /* to temporarily suspend a log file */

log off /* to resume writing to a log file */

log close /* to close a log file */

translate *filename*.smcl *filename*.log /* to translate a smcl log file to a text log file */

log using *filename*, append /* to append to an existing log file */

log using *filename*, replace /* to replace an existing log file */

cmdlog using *filename* /* to start a command-only log file that contains solely what you type and no results */

cmdlog close /* to close a command-only log file */

All of the output that appears in the Results window can be captured in a log file and by default, it is saved as a SMCL file if no extension is specified. The log file can be saved as a Stata formatted (SMCL) file or a text file (ASCII). A SMCL log file contains the output you saw during the Stata session and can only be viewed in the Viewer in Stata. SMCL files can be translated to ASCII text by translate to be viewed in other word processors. A simple way is to specify log extension when opening the log file.

It is a good practice to always keep logs of everything you do, which means that the following line after changing the session's directory, and setting memory is to create a log. For example, a good sequence in your program is,

set mem 500m
cd h:*foldername*
use auto

Loading Dataset

use C:\\Stata\auto.dta /* to load a dataset in Stata format */

use *filename* /* to read data in Stata format with a .dta extension */

insheet using *filename* /* to read data with tab or comma delimiters, no space delimiters */

infile using *filename* /* to read unformatted data */

infix using *filename* /* to read formatted data */

input x y /* for manual input */

2 3
5 8
end

Data not in Stata format can be easily transferred to a .dta file using StatTransfer. This makes loading the data easier by using simply the use command.

When calling on a stata data file, i.e. any file with *filename*.dta it is not necessary to type .dta since by default, stata presumes you are calling a .dta file.

Data Management

Describing Data

codebook /* to display information about variable's names, labels and values */

count /* to obtain a count */

list /* to list observations of variables */

count if mpg >= 20 /* to count observations in a given condition */

list mpg foreign in 1/10 /* to list variables mpg and foreign in observations 1 to 10 */

describe /* to display a summary of a Stata dataset, describing the variables and other information */

summarize /* to display basic descriptive statistics */

summarize mpg, detail /* to display more descriptive statistics */

table mpg /* to get a frequency table */

tabulate mpg /* to create a frequency distribution table */

tabulate mpg foreign or tab2 mpg foreign /* to create a two-way frequency table */

tab1 mpg foreign /* to create separate one-way frequency table */

correlate mpg weight /* to display a matrix of Pearson correlations for the variables listed */

label /* to change a description of a variable */

inspect mpg /* to display information about the values of variables and is useful for checking data accuracy */

display 1 + 1 /* to act as a calculator */

(You can pick any commands you like to describe your data and put it in your do file.)

The if exp qualifier defines the condition for which the exp is `true'. Logical operators in Stata are:

not ~

equal ==

greater than >

greater than or equal >=

less than <

less than or equal <=

and &

or |

The *in range* qualifier lists values for the subset of cases in the range.

Data Manipulation

generate /* to create a new variable */

replace /* to change an existing variable */

rename name old name new/* to rename an existing variable */

drop /* to delete a variable */

keep /* to keep a variable */

sort variable /* to sort data by a certain variable */

egen / * to generate special new variables */

egen is a powerful command that has many options for creating new variables, e.g. mean, rank, median, sum. See help egen to learn more about it.

Coding for Dummies

tabulate rep78, generate(repair) /* to create five dummy variables called repair1, repair2, repair3, repair4, repair5 */

xi i.rep78 /* to create 4 dummy variables and omitted the dummy variable for group 1 */

The **xi** command can be used as a pre¯x before estimation command, e.g. xi: regress mpg weight i.rep78 means regressing mpg on weight and the 4 dummy variables of rep78.

The by construct

The by construct causes command to be repeated for each unique set of values of the variables in the *varlist*. It can be used as a by prefix or an option depending on command. The data has to be sorted first or use bysort in the prefix.

```
bysort foreign: egen pmean = mean(price)
egen ppmean = mean(price), by(foreign)
list pmean ppmean in 1/10 /* to show that the above two commands are the same */
```

From _n to _N

```
generate id = n /* n is Stata notation for the current observation number */
generate nt = N /* N is Stata notation for the total number of observations. */
```

_n and _N combined with the by construct can produce some very useful results. One use is to check duplicate data. This can be done with the following commands in the auto dataset:

```
sort make
list if make == make[ _n+1]
```

(In the data, there should be just one observation for each type of cars included. It is useful when there is a unique identifer for each observation in the data.)

Merging Datasets

To merge different datasets together, there are three commands: **append**, **merge** and **joinby**. The **append** command combines datasets vertically. The **merge** command combines datasets horizontally. The **joinby** command combines datasets horizontally but form all pairwise combinations within group. There are options to replace the values when there are observations in both files. Check the help merge and help joinby to learn more. After merging the datasets, use save *filename* command to save the merged datasets for later use. If the *filename* already exists, use save filename, replace to replace the existing dataset.

Statistical Procedures

Regression

```
regress mpg foreign weight /* to regress mpg (dependent variable) on foreign and weight (independent variables) */
```

This is the command for the ordinary least squares estimation. This should always be the first technique you use. You will be ask to write this program in Matlab once we have complete multiple variable regressions.

Formatting Regression Output

The command outreg is used to create regression output table similar to the journal article. There are many options in specifying the information displayed in the text ¯le. Check help outreg for more information. A common specification in economics:

outreg using tutorial 1, se bracket 3aster /* to open a text file, tutorial 1.out for the formatted regression output with the regression coefficients and the standard errors in bracket and asterisks for 1%, 5% and 10% significance levels */

Other Estimation commands

The following is a selection of other estimation commands in Stata.

anova (analysis of variance and covariance)

arch (autoregressive conditional heteroskedasticity family of estimators)

arima (autoregressive integrated moving average models)

glm (generalized linear models)

ivreg (instrumental variable and two-stage least squares regression)

logit (maximum-likelihood logit regression)

nl (non-linear least squares)

Stata has a whole slew of commands for all kinds of regression techniques. The best way to learn about them, is always first to read econometric books, be they textbooks, or cookbooks or journal articles, so that you understand first what those techniques do. Without full understanding of the techniques, you will not be able to use the output.

Graph

For Stata 8 users, the graph options are much easier as Stata has developed a drag and drop input system which no commands are needed to specify.

graph mpg weight /* to create a scatterplot between the variables */

sort foreign

graph mpg weight, by(foreign) total /* to create separate scatterplots for foreign and local cars */

graph weight, histogram bin(7) normal /* to create a histogram with seven bars, the normal option superimposes a normal curve on the graph */

graph weight, histogram bin(7) title("Histogram 1") /* same graph as before but with a title added */

graph weight, histogram bin(7)ylabel xlabel normal /* same graph as before but with axes label */

graph weight, histogram bin(7)ylabel xlabel (15 20 30 40) normal /* same graph as before but specify the x axis label*/

predict mpghat /* to compute a predicted mpg for each observation */

graph mpg mpghat weight if foreign == 0, connect(.l) symbol(Oi) /* to graph mpg vs. weight and mpghat vs. weight for local cars; connect(.l) means do not connect the mpg vs. weight points but connect (with straight line) the mpghat vs. weight points; symbol(Oi) means using big circles for the mpg vs. weight points but use invisible symbpl for the mpghat vs. weight points */

Graph Types

histogram

box

bar

oneway

twoway

matrix (half)

star

Symbol Options

O large circle

S large square

T large triangle

o small circle

d small diamond

p small plus

. dot

i invisible

[*varname*] variable to be used as text

[_n] use observation number as symbol

Line Options

. do not connect

l draw straight lines between points

L draw straight lines between ascending x points

m connect median bands using straight lines

s connect median bands using cubical splines

Line Pattern Options

l solid line (default)

(underscore) a long dash

- (hyphen) a medium dash

. a short dash (almost a dot)

Add [pattern] after the line type

Title and Axes Options

title("text")

t1itle("text") /* Title on top of the graph */

t2itle("text")

b1title("text") /* Title below the graph */

b2title("text")

l1title("text") /* Title on the left of the graph

l2title("text")

r1title("text") /* Title on the right of the graph */

r2title("text")

xlabel , ylabel , rlabel , tlabel

xline , yline

Multiple Graphs

graph price, bin(4) normal title(Prices) saving(part1)

graph length weight, title(Length vs. Weight) saving(part2)

graph using part1 part2, title(Both Graphs) saving(part3)

Graphs can be printed from Stata directly (File option) or copied (Edit option) and pasted to other word processors.

Stata Resources

Help within the Stata interface

help command can be used from the command line or from the **Help** window, e.g. help egen, help joinby. **search** command searches for information in Stata maunals, FAQs and Stata Technical Bulletins. findit command combines search and net search.

Tutorial

Stata has built-in tutorial and typing tutorial will bring up information about the tutorials. tutorial intro will bring up the tutorial on stata introduction.

When it comes to using statistical software, nothing beats hand-on trial. Only then would you learn the intricacies of the software.

You will have a stata program assignment soon.