ECON 4811: Introduction to Econometrics

Andrea Velasquez
Spring 2018

STATA : A BRIEF INTRODUCTION

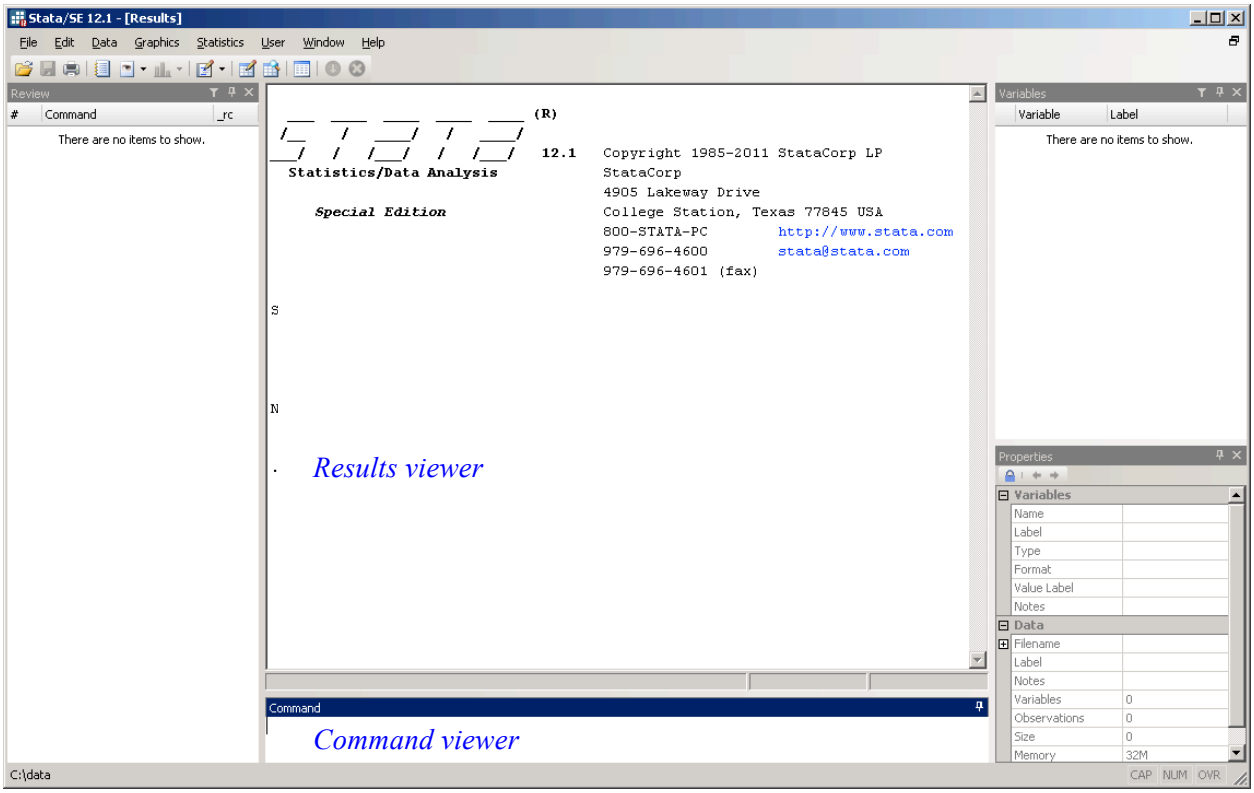
STATA is easy to learn, powerful and very flexible. It will serve you well for this class, statistical research in other courses and your own research papers as well as beyond. It is very widely used and has a huge user base that develops new tools. The product is continually updated to assure that it is on the frontier of econometric practice.

STATA has outstanding on-line help and a menu-driven interface that you can use to learn the command syntax. Honestly if you need to know how to do something in STATA, Google will almost always have the answer. The best text reference is the STATA manual, which is available on-line, and help pages on the STATA website (www.stata.com). There are several other websites that are very helpful. I particularly like examples of commands and programs provided at UCLA (www.ats.ucla.edu/stat/stata) and a tutorial at Princeton (data.princeton.edu/stata). *Statistics with STATA*, by L. C. Hamilton, provides a very accessible introduction to STATA.

This note is intended to help you get started with STATA. It is not a substitute for your own experimentation; nor is it a substitute for using the on-line help and reading the manual. It should, however, provide you with the tools you will need to *begin* working with STATA; you should aim to develop some familiarity with the product and understand the logic underlying it. So that they stand out, the STATA commands in this note are written in **bold**: when you use STATA, all commands are in **lowercase**. (Everything in STATA is case sensitive including variable names.) In this note, examples are in italics.

Starting up STATA

Invoke STATA and you will have a screen with 5 windows.



The large window in the center is the *Results viewer* where your commands are echoed and results appear. You type in commands in the window below it, the *Command viewer*. A history of your commands is recorded in the left window. Information about data in memory appears in the right windows; a list of variables in the upper window and information about specific variables in the lower window. A *Menu* is at the top left.

Type **help** in the Command window. A window will open and you can select from the help items. You can also click on Help in the menu bar that runs along the top of the STATA screen. Click on PDF Documentation to load the STATA manuals. They are all hyperlinked and easy to use. You can search for a command by clicking on Search or type **search <topic>** in the command window where <topic> is some topic you want to learn about. For example, **search *regression*** will provide you with a list of ways to run a regression in STATA.

To use data

Go to the class webpage. Click on the *Problem sets, data, exams and solutions* link and click on the dataset that you want (click on the file with extension .dta which is a STATA dataset). You will be asked where you want that file to be located. You may get the file with the expenditure data –which is called **expend.dta**. Assuming that you copy the dataset into your current directory then you can bring the data into memory with the **use** command: **use *expend***. Since the default extension for a STATA dataset is .dta, you do not need to type that in. If the data are not in the current directory, you need to navigate to the directory or include the path after **use,** e.g. **use */path/dataset***.

If you get a message that data are already in memory, then you must include the option **clear**, i.e. **use *expend*, clear** which will clear all data from memory prior to loading expend.dta. STATA cannot handle more than one datasets in memory at any point in time. Use **save** to save your data. If the data exist and you want to overwrite those data, use **save <datasetname>, replace**. E.g. **save *expend*, replace**.

There are many ways to import data into STATA. Click on File in the menu in the top left corner and then click on import. You can also read data from a text file. Type **help infile**.

Looking at the data

To see what variables are in a dataset, use **describe** which displays a list of all variables and their attributes including the variable label which tells you about the variable. This information is displayed in the Variable viewer (in the top right of the STATA window). Describe also tells you how many observations (individual records) are in the dataset.

To look at the *value of each observation*, you might **list** the variable(s). For example, in the Command viewer, type **list *hhsize***. Instead of typing out the name of the variable, you can type **list** in the Command viewer and then click on the variable in the Variable viewer and the name will appear after list. Then hit return. You can list all variables by saying **list** and hitting return. (In general, for any command, if you do not specify a set of variables, STATA will execute the command on all variables.) You can also list a range of variables. Say you have var1 var2 var3 var4 in the dataset, then var1-var3 will list var1 var2 var3. The order of variables is given by the storage order (as given in the list of variables in the Variables viewer or when you **describe** the dataset).

You can also **browse** and **edit** the data. Click on Data in the menu in the top left and click on Data Editor.

Hit the spacebar when the screen fills and you will get the next screen. Hit CTRL-C or BREAK if you want to stop the list (or any other STATA function from continuing to display results). You can retrieve the last command in the Command window by hitting PageUp or CTRL-R. Click on a command in the Review window and you will get the same effect.

Exploratory data analysis: univariate summary statistics

STATA will **summarize** the data. Type **summarize <**variable> (or just **summarize** for all variables in the dataset). In the Results window, you will see the number of observations, mean, standard deviation, minimum and maximum value of each variable. If you want order statistics, then include the option **,detail** (after the list of variables) and you will also get the median, bottom and top quartile, a couple of other order statistics (percentiles) as well as the smallest and largest 5 observations in the dataset. For example, **summarize *tot_exp*, detail** will display summary statistics for the variable tot_exp.

In general, options in STATA commands are written at the end of the command usually following a comma. There are, however, some important exceptions. For example, in some cases you will want to calculate statistics with weights. There are many different types of weights and STATA treats them differently. Type **help weights** for information. If you want to calculate weighted statistics, then you need to include the option **[pweight=**variable name] (if you are using 'sampling' weights) or [**fweight**=variable name] (if you are using frequency weights). For example, **summarize *pce* [fweight=*hhsize*], detail**

If the variable is discrete (or categorical) then it will typically take on only a limited number of values. In this case, you might find a frequency tabulation more useful than measures of central tendency: use **tabulate**. The table reports the absolute frequency (Freq.), relative frequency (Percent) and cumulative relative frequency (Cum.). The **plot** option will print out a histogram type plot alongside the table. If you have missing observations, then they are suppressed from the frequency count; if you want to include them, then add the **missing** option. You can also use **tabulate** to create cross-tabulations of two (or more) discrete variables.

Say you have two variables, one discrete and one continuous (such as hhsize and tot_exp). You want summary statistics of tot_exp for each hhsize: you have at least two options. First, **sort** by the discrete variable (hhsize) and then precede the **summarize** command with **by <varname>:** e.g. first **sort *hhsize***. Next, **by *hhsize*: summarize *tot_exp***. A short cut allows you to do this in one step using **bys** which will sort the variable for you when it calculates the summary statistics. That is, **bys *hhsize*: summarize *tot_exp*.**

A second option is to use **tabsum** (for calculation of means) or **table** (for other statistics). Type help summarize, help tabulate, help tabsum and help table for more information on these commands.

Selecting subsets of data

Sorting is immensely useful: if you plan to do several analyses on subsets of a particular dataset, then sorting may be a good strategy. Virtually all the STATA utilities permit analysis of subsets of the dataset with the **by** option. Note, however, that to use the **by** option, the data *must* be sorted by that variable. If you use the shortcut **bys variable: command** then the data will be sorted by variable prior to running the command.

If you want to restrict attention to a particular subset of observations, then you might use the **in** or **if** option. For example, say you want to restrict attention to the first ten observations: to list them, use **list** with the option **in** 1/10. In general, the option is **in** start_obs/stop_obs. Alternatively, you may be interested only in those households with no more than 4 members. To list them, **list** variable **if** hhsize <=4. You can use any relational operator here: (> < >= <= not equal is ~= and equal is == which distinguishes it from the algebraic = sign). So if you want to list tot_exp if hhsize is equal to 1, then **list tot_exp if hhsize==1**.

Keeping a copy of results

If you want to keep a copy (on disk or paper) of your session (including the commands you type and the output) then you need to open a log file. Assuming you can write into your current directory, **log using filename**, will open a file called **filename.log**. You can temporarily suspend the log, **log off**, and restart it, **log on**. If you want to close the log, **log close.** If the file exists, you will need to use the **,replace** option**.** To append results to an existing log file use the **,append** option. Notice that your commands are included in the log file. This helps you keep track of what you have done. Notice, also, that the command generated by your use of the menus is listed in the command viewer and on the log file. This is one way to learn about STATA's commands.

There are many options for how the output are displayed. STATA writes output in a STATA markup language (to make it pretty). If you want the output in plain text, use the **,text** option. Type **help log.** Alternatively click on File in the command menu and then click on Log.

Exploratory data analysis: graphics

It is very easy to get simple graphical descriptions of data using STATA. To make pictures very pretty takes more time: you should look at the options available to you (by using the menu, reading the manual or using help). I will go over only the simplest plots (which we will use) and a couple of potentially useful options.

Histograms

histogram <variable_name> will draw a histogram with 5 bins and the axes identifying the largest and smallest values. You can change the number of bins with the **bin(b)** option, where b is the number of bins you want (and must be less than 50). You can also define the axes with the **xlabel(value1 value2 value3)** and/or **ylabel(value1 value2)** options. The x-axis will identify three values and the y-axis 2 values in this case. If you want the absolute frequency (instead of the relative frequency) on the y-axis, use the **freq** option. For example, **histogram *tot_exp* , bin(*10*) xlabel(*0 2000 5000 10000 15000*) ylabel(*0 20 40*) freq**

Two-way graphs

There are a lot of ways to display the relationship between two (or more) variables in STATA. For example, **twoway (scatter *variable1 variable2*)** will display a scatter plot of variable1 and variable2 . **twoway (connect *variable1 variable2*)** will connect the points with a line. You can control the symbol used for observations, the way they are connected, the labels on axes. You can draw bar graphs, pie charts and star graphs and you can have matrices of scatterplots. You can also combine several graphs together. See the graphics chapter in the manual or use the Graph option on the command menu.

Printing output

Use the File Print in the command menu to print a log file or a graph.

Relationships among variables

To compute correlations among a group of variables, **correlate** variables. Clearly you must have at least two variables. If you do not specify any variable names, then the correlations among all variables in your dataset will be calculated. The **covariance** option will result in covariances being calculated instead of correlations. If you use the **means** option, then the output will include the mean, standard deviation, minimum and maximum of each variable.

Regression analysis

It is extremely easy to run regressions with STATA. **regress** <y x> will produce estimates of the regression $y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$ You may enter as many independent variables, x, as you like.

For example, **regress** ***food_exp tot_exp hhsize***

The output includes a column of the explained sum of squares (SS column), residual sum of squares and total sum of squares. The means of these statistics are in the third column. Some summary statistics are printed in the final column of the upper panel; these include R^2, adjusted R^2 and the root mean squared error (MSE). The estimated coefficient is in the first column next to each variable name.

You can use the **by**, **in** and **if** options to subset the data. The **[pweight=**variable name] option permits you to run a weighted regression.

You will often want to look at predicted values of the dependent variable. After running a regression, **predict *variable_name*** will create a variable called variable_name which contains the predicted value of the dependent variable. To obtain residuals, use **predict *variable_name*, residual**. You can then use **summarize** or the graph options to examine these data.

For example, after

regress ***food_exp tot_exp***

you can

predict *yhat*
predict *uhat*, residual
sum *yhat uhat*, detail
sort tot_exp
twoway (scatter food_exp tot_exp) (connect yhat tot_exp)

which runs the regression of food expenditure on total expenditure and household size. The next command predicts the value of food expenditure for each observation in the dataset (predicted value of the dependent variable) and the third command calculates the residual for each observation. The fourth command summarizes those new variables. The fifth command sorts the dataset by tot_exp from lowest to highest value. The sixth command creates a graph with a scatter plot of the bivariate relationship between food expenditure and total expenditure along with the predicted regression line overlaid on the scatter plot.

One of the key strengths of STATA is that it provides many options for estimating regression-type models. A small sub-set of those options are introduced in the next sub-section.

Extensions to the regression model

Quantile regression

The linear regression model provides an estimate of the conditional mean of the dependent variable, that is the mean of the dependent variable conditional on the values of the covariates. In some cases, an investigator in interested in other parts of the conditional distribution of the dependent variable. Quantile regression can be estimated with **qreg** ***y x*** **, q(*quantile*).** The median regression (quantile=50) is the default and provides the least absolute deviation estimator, e.g. **qreg** ***food_exp tot_exp , q(50)*** where the option ***, q(50)*** is unnecessary because it is the default.

Hypothesis testing

It is straightforward to test hypotheses about coefficients in regression models. Consider a model that relates food expenditure to total expenditure:

regress ***food_exp tot_exp***

To test whether the coefficient on tot_exp is equal to 1, use the test command

test tot_exp=1

Extend the model to include the square of total expenditure, tot_expsq, as a covariate:

regress ***food_exp tot_exp tot_expsq***

To test whether both tot exp and tot_expsq are jointly significant, use this form of the test command

test tot_exp , tot_expsq

Estimation of variance of coefficient estimates

The optimality of ordinary least squares estimates relies on several assumptions. One is that errors in the regression are drawn from a distribution with a finite variance (homoskedastic). You can test this assumption using the ***estat httest*** command after estimation of the model. For example:

regress ***food_exp tot_exp tot_expsq***
estat httest, fstat

will report an F test statistic to test the assumption that the errors are homoskedastic.

If the assumption is rejected, the standard errors on your coefficient estimates are likely to be wrong (typically too conservative). You can use a robust (Huber-White or sandwich) estimator to calculate the standard errors by specifiying the ***vce(robust)*** option in your regression statement

regress ***food_exp tot_exp tot_expsq, vce(robust)***

OLS also assumes that residuals are uncorrelated with one another. That assumption may be violated (say because the residuals share some common factors (are clustered) or because of serial correlation in the residuals). One approach to taking clustered unobserved factors into account in the estimation is to use:

regress ***food_exp tot_exp tot_expsq, vce(cluster groupvar)***

where ***groupvar*** is a variable that identifies the groups or clusters of residuals.

In some cases, you may prefer to use a non-parametric approach to estimate the variances of the coefficient estimates such as the jackknife or bootstrap. They are options for vce:

regress ***food_exp tot_exp tot_expsq, vce(jackknife)***

and

regress ***food_exp tot_exp tot_expsq, vce(bootstrap)***

respectively.

Instrumental variable estimation

Consider the model in which $y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$ where x_i and ε_i are potentially correlated. The OLS estimates of β_0 and β_1 will be biased. If there is an instrument, z_i, that is correlated with x_i and not correlated with ε_i

then the instrumental variables estimate of β_0 and β_1 will be unbiased. Use **ivregress** to estimate this model using instrumental variables. To use the method of two stage least squares, estimate **ivregress *2sls* y (x=z)** . You can have many covariates, x, as you like. Say you have x_1, which you know is not correlated with ε, and x_2 which is potentially correlated with ε. Then you do not want to predict x_1 in the model and you can estimate **ivregress *2sls y x1 (x2=z)*** . If you add the option **, first** after the ivregress command, STATA will print the first stage estimates. That is **ivregress *2sls y x1 (x2=z), first*** . The first stage estimates are the same as the results from estimating the model **reg *x2 z*** .

Panel data methods

Another approach to handling correlations between covariates and unobserved heterogeneity is to estimate a model with fixed effects. In this case, you can use **xtreg** with the **fe** (for fixed effects) option. There are several other options including **re** for random effects. For example, to estimate a model with fixed effects, $y_{it} = \beta_0 + \beta_1 x_{it} + \mu_i + \varepsilon_{it}$ where μ_i is a fixed effect for all observations in some group denoted by, say, individual, and represented by the subscript i in the model, estimate **xtreg *y x , fe i(individual)***

Manipulating data

All these utilities are very useful. You will, however, also want to create new variables, manipulate them, edit values of variables and drop variables.

You may **drop** any variables by typing **drop** variable_names. If you want to drop all variables, use the STATA reserved variable, **_all**. If you want to **keep** a few variables, it would be more efficient to list those variables **keep** variable_names.

If you want to generate a new variable, you need to **generate** new_variable = some manipulation of old_variables. For example, to create the square of total expenditure (tot_exp2 = tot_exp*tot_exp), **generate *tot_exp2 = tot_exp*tot_exp*** or **generate *tot_exp2 = tot_exp^2* .** In general, you may add [+], subtract [-], multiply [*] or divide [/] variables. For any variable (var_name) you can also compute the absolute value [abs(var_name)], logarithm [log(var_name)], exponent [exp(var_name)], square root [sqrt(var_name)] and integer value [int(var_name)]. You can, of course, combine all these operators and also use the **if** and **in** options if you want to work on a subset of the data.

You cannot generate a variable that already exists. If you want to change an existing variable, you must **replace** it. For example, say you want to divide tot_exp2 by 1000, then **replace *tot_exp2 = tot_exp2/1000***. Again, you can use the **if** and **in o**ptions if you only want to replace particular observations: this is very useful for fixing errors in the data. (See also the **edit** command). As another example, say you want to create a variable which takes the value 1 if household size is greater than 4 and it takes the value 0 otherwise. You can **generate** hh_gt4 which is 1 **if** hhsize > 4. Then **replace** hh_gt4 which is 0 **if** hh_gt4==. Notice that hh_gt4 is set to the missing value, '.' when hhsize is 1 through 4 and you use the double equals to replace those dots with zeroes. Thus:

generate *hh_gt4*=1 if *hhsize*>4
replace *hh_gt4*=0 if *hh_gt4*==.

An alternative that has the same effect would be to replace hh_gt4 if hhsize is less than or equal to 4:
replace *hh_gt4*=0 if *hhsize<=4*

There is a very useful shorthand built into STATA

generate *hh_gt4*=(*hhsize*>4)

which has the same effect as the commands above. In this case, if the statement in the parentheses is true, hh_gt4 is set to 1 and if the statement is not true, hh_gt4 is set to zero.

You can use the same type of conditional statements to **drop** or **keep** observations from the dataset. For example, **drop if *hhsize > 4*** will drop all households with more than four members.

Finally, **assert** is an extremely useful tool in STATA. If you assert some expression is true, STATA will tell you if that is correct. For example, **assert *hhsize <= 4*** will be true after you have executed the DROP statement in the last paragraph. This is a good way to check that you have done to the data what you think you did. It is also good way to check any data you use (and manipulate) follow the patterns that you are expecting.

Keeping track of your work

Using the command line (or menus) is useful when you first use STATA. However, it quickly becomes very tedious. First, it is useful to have a record of what you did so that you can refer to it in the future. Second, you often want to repeat analyses with different subsamples or variables and it is a pain to write out each command. More generally, it is really important to keep track of your work so that you can always retrace your steps.

With this in mind, I encourage you to keep your commands in a program. STATA calls such a program a Do file. Create mypgm.do with a text editor and from the command window type **do *mypgm***. (You do not have to include the extension .do as it is implied. If you choose to call your do programs something else, include the extension in the filename. If the program is not in the current directory, include the path.) The program will execute as if you entered each command one by one.

You will notice that if you use the menu to create a command, the command is displayed in the results menu. By logging your output (e.g**. log using *filename*, text**) you will keep a list of everything you have done. If you edit that file, you can easily turn it into a do file.

Documenting your programs

A long stream of STATA commands is going to be difficult for you or anyone else to read. Help yourself and anyone who will read your do file by explaining what it does as clearly and concisely as possible. At the beginning of your do file, you should write an explanation of the purpose of the do file and describe the sources of data, the output you create and so on. For each block of commands in the body of the do file, explain what the block of commands will do so that the reader knows what to expect. If you have a really complicated command or some strange looking logic, explain it to the reader before you use it.

Precede any comments in your program with a * which indicates to STATA that the rest of the line should be ignored. If you want STATA to ignore a block of commands use /* to denote the beginning of the block and */ to denote the end of the block. You can also use this to split commands across lines. STATA thinks a command ends when it sees a carriage return. You can override that by putting /* at the end of the first line of the command and then */ at the beginning of the second line. You have told STATA to ignore the carriage return. You can achieve the same goal by changing the indicator for the end of a command from carriage return to something else using the #delimit statement. For example #delimit ; tells STATA that a ; denotes the end of each command. This is most useful when you create your own programs.

Documenting your data

If you create a dataset, you should document the data. First, you can label any variable in a dataset using **label variable "<variable label>".** You can label a dataset using **label data "<date label>".** In both cases, you can write whatever you like in the <…> between the double apostrophes.

Sometimes it is convenient to label the values of a variable. For example, the variable gender may be 1 if a person in a study is male and 3 if the person is female. It's a bit hard to remember which is male and which is female. First define the labelname using **label define labelname value "text".** For example

label define *lblgender 1 "1.Male" 3 "3.Female"*

Next assign the values of labelname to the variable. For example

label values gender lblgender

Now **tabulate gender** and you will see 1.Male and 3.Female instead of 1 and 3, respectively. For more information on labels, see help label.

This is intended to be a brief introduction to some features of STATA that you will use. To be sure it only scratches the surface of the power of the product.

The best way to learn how to use STATA is to just use it. With the menus, on-line help and this handout, you should have enough to get going. Don't be afraid to experiment. When in doubt, consult on-line help, the manual, or your friend. Or just make a guess.