**SGPE Econometrics Lab 1: Introduction to Stata**
**Sean Brocklebank & Mark Schaffer**
**version of 09.09.2012**

## Introduction

The goal of this lab is to introduce you to Stata, the econometrics package we will be using in the SGPE MSc. The structure of the lab encourages you to experiment, try out different commands, use the online help, etc. By the end of the lab you should be comfortable enough with Stata to be able to do more serious work.

Stata's online help facilities are excellent, and you are strongly encouraged to make use of them. The main help commands are:

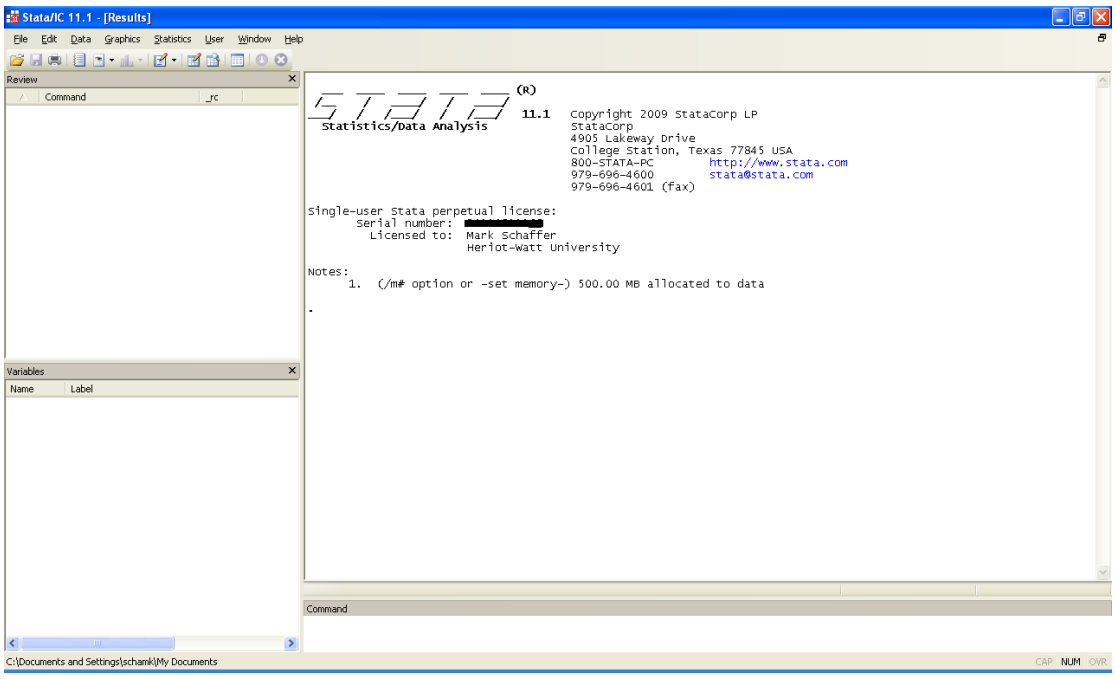
| | |
|---|---|
| **`help`** [keyword] | Provides help on a command |
| **`search`** [keyword] | Searches the main Stata documentation for help |
| **`findit`** [keyword] | Like "search" but also searches the internet |

At the end of this document is a short list of basic Stata commands that you will be using during the lab.

To encourage you to work independently and to exploit these help facilities, this lab is structured so that some details of how to do things are left to you, the list of commands below, and Stata's help facilities.

## Starting Stata

When you start Stata, you will get a screen that looks something like this:



(Your colours and box sizes might be slightly different.) The big box on the right is the **Results window**; this is where you see your estimation results and so forth. The wide box underneath is the **Command window**; this is where you type in your commands.

The other two windows are helpful additions but not used as much: the one on the top left is the **Review window**, which tracks what you have typed into the Command window, and the one on the bottom left is the **Variables window**, which lists the variables that are currently in memory.

In this course, we almost always use the Command window (and something else to be introduced in the next lab, called a "do file"), for executing commands. Stata also has a menu-driven method for executing commands (via the drop-down menus at the top), but we almost never use this. The reason has to do with replication, a key feature of the scientific method; you must always be able to replicate your results. (This is what "do files" are for.)

**Task 1: Load the dataset and calculate some summary statistics**

Today's data come from "A Contribution to the Empirics of Economic Growth," an important 1992 QJE paper by Greg Mankiw, David Romer, and David Weil.

The basic data consist of 12 variables (we will make more variables later):

| | |
|---|---|
| `c_index` | numeric codes from 1-121 assigned to the countries |
| `c_name` | full country names |
| `c_code` | three-letter country codes (used for labelling graphs) |
| `cont` | continent |
| `nonoil` | dummy; =1 if the country does *not* export oil |
| `inter` | dummy; =1 if the country has good-quality data |
| `oecd` | dummy; =1 if the country was in the OECD |
| `gdp60` | GDP per member of the working age population in 1960 |
| `gdp85` | GDP per member of the working age population in 1985 |
| `pop` | average population growth over 1960-1985 |
| `igdp` | average investment rate over 1960-1985 |
| `school` | average secondary school enrolment of the working population over the years 1960-1985 |

To load the data, do the following:

1. In Windows, go to "My Documents" and create a folder called "Econometrics" with a subfolder called "Lab1" (no spaces!). This should create a folder with the path M:\Econometrics\Lab1. This is where you will keep your copies of the lab data and files.

2. Next, go to the shared network drive where all the SGPE coursework is kept. In \coursework\SGPE\Econometrics\Lab1 you should find several files: "mrw1992.dta", "mrw1992.pdf", and this document.

3. Copy the file "mrw1992.dta" into M:\Econometrics\Lab1. You can also copy the PDF file if you want – it's the paper by Mankiw et al. QJE paper.

4. Open Stata (all further commands are in Stata).

5. Type this into the command window:

   ```
   cd M:\Econometrics\Lab1
   ```

   "cd" will "change directory". This is useful because it Stata will know where to look for the data loaded in the next command. Now type:

   ```
   use mrw1992
   ```

   As long as you've copied the file mrw1992.dta into the correct directory, you should now see a list of variables appear in the Variables Window at the bottom left.

6. "Describe" the dataset using the **desc** command by typing this in the command window:

   ```
   desc
   ```

7. Calculate summary statistics for the variables in the dataset by typing this in the command window:

   ```
   sum
   ```

8. Do the same, but only for the gdp60 and gdp85

   ```
   sum gdp60 gdp85
   ```

9. Do the same, but with fewer letters (note: the * below is a wildcard; **gdp*** means "everything beginning with the letters "gdp").

   ```
   sum gdp*
   ```

10. Do the same and get a full set of detailed statistics by using the "detail" option.

    ```
    sum gdp*, detail
    ```

This is a key feature of Stata syntax – options are separated from the main command by using a comma.

*Tip*: if you want to re-execute a command in the Command window, hit <Page-Up> until it reappears. You can edit it if you want – this is the easiest way to do #9 above.

**Task 2: More about dataset and variables**

Look again at the summary statistics from the **sum** command used in Task 1. You will see that there are 121 observations for four of the variables but only 118

observations for **school**, 116 observations for **gdp60**, 108 observations for **gdp85**, 107 observations for **pop** and – apparently – no observations for the variables **c_name** and **c_code**.

There are two things going on here:

The variables **c_name** and **c_code** are "string", i.e., text variables. They are *not* numeric, and hence there are no summary statistics for them. They do, however, exist – in fact, this is a dataset about different countries, so this variable actually identifies the observations uniquely. You can see them by using the **list** command. Type this into the Command window:

```
list c_*
```

The variable **pop** (the average population growth rate) is available for 107 observations only. Type this into the Command window:

```
list c_* pop
```

and you will see dots ("**.**") for the missing values.

Stata's treatment of missing values is somewhat idiosyncratic – a missing value is equivalent to $+\infty$ (positive infinity). This will be important when it comes to manipulating data.

You can list the variables of interest using the list command:

```
list c_name gdp* pop igdp school
```

but it may be more convenient to use browse the data using the Data Editor. This is one of the few commands that is best launched from the menu bar at the top: Data > Data Editor > Data Editor (Browse). (Or just click on the icon at the top that has the tiny magnifying glass on it.)

### Task 3: Tabulating categorical data

**cont** is a variable that takes only 6 values representing the continent on which each of the countries is located.

**oecd** is a dummy variable that takes the values 0 or 1, depending on whether or not the country was a member of the Organization for Economic Cooperation and Development during the years 1960-1985.

The natural way to summarize categorical data is to tabulate the frequencies. Try the following:

```
tab oecd

tab cont
```

The tab command can be used for two categorical variables to generate a two-way table of frequencies (a "crosstab"). Try the experimenting with following combinations of options. **row** and **col** request row and column percentages, respectively.

```
tab cont oecd
tab cont oecd, row
tab cont oecd, col
tab cont oecd, row col
```

**Task 4: Looking at the data**

It's *very* important to look at your data.

The basic univariate plot is the histogram. Try looking at a histogram for the school enrolment variable

```
hist school
```

Now try investment

```
hist igdp
```

Now try the population growth rate

```
hist pop
```

Now try GDP per worker in 1985

```
hist gdp85
```

Now try GDP per worker in 1960

```
hist gdp60
```

Notice that the last histogram is dominated by one very wealthy outlier in the far right of the distribution. We'll come back to the issue of how to find and deal with outliers later, but for now notice that we can get a bit more information out of this histogram by using an "**if**" statement to restrict ourselves to observations below a GDP per worker of $15,000 (more on **if** statements in the next section)

```
hist gdp60 if gdp60<15000
```

A good way to compare different subsets of your data is the box plot. A box plot displays boxes bordered at the 25$^{th}$ and 75$^{th}$ percentiles of the variable with a median line at the 50$^{th}$ percentile. The box plot also has "whiskers"; there are various ways of doing these, but the one used by default in Stata is for the whiskers to extend 1.5 times the interquartile range beyond the outer edges of the 2$^{nd}$ and 3$^{rd}$ quartiles (note: the interquartile range is the value at the 75$^{th}$ percentile minus the value at the 25$^{th}$ percentile).

But maybe it's better to just *look* at a box plot. Type this

```
graph box pop, over(cont)
```

This will give you information about the distribution of population growth rates across continents. Can you understand the syntax of the command? What if we wanted to look at school enrolment rates instead of population growth rates? Then we would type this:

```
graph box school, over(cont)
```

You can see that enrolment rates don't vary as much as population growth rates. What about investment rates? Let's look:

```
graph box igdp, over(cont)
```

What if we wanted to look at investment rates in the OECD vs. Non-OECD samples. How would we do that? Like this:

```
graph box igdp, over(oecd)
```

But we're not just interested in univariate plots. We're also interested in how variables relate to each other. The basic bivariate data view is the scatterplot. To see how GDP per worker in 1985 relates to average population growth in the previous 25 years, type this:

```
scatter gdp85 pop
```

We might also be interested in the relationship between GDP per worker in 1985 and the average investment rate over the previous 25 years

```
scatter gdp85 igdp
```

In fact, there are a large number of scatter plots that we can imagine making even with just five variables. A useful command for getting to know your data is **`graph matrix`**, which allows you to see lots of scatterplots at once. Type the following into the command window:

```
graph matrix gdp60 gdp85 pop igdp school
```

Notice the **`gdp60`** vs. **`gdp85`** plots in the top left. These have the predicted upward slope, but they also have a conspicuous outlier

```
scatter gdp85 gdp60
```

Which is that wealthy country off in the right of the data? There are several ways to find out, but one way is to use the **`mlabel()`** option, which puts marker labels on the individual data points. Remember that to use an option, we put a comma after the command. To us this particular option, we put a label variable inside the parentheses; we'll use **`c_code`**, which provides a three-letter country code for every country in our dataset

```
scatter gdp85 gdp60, mlabel(c_code)
```

Now we can see that the rich outlier was Kuwait. If you previously read through the Mankiw, Romer and Weil paper, you may remember that they didn't use oil exporting countries for most of their analysis. The argument that they gave was that the productivity of these countries was dominated by extraction rather than the returns to investment, so we should not expect them to conform to the Solow model. In the next section we'll see how we can exclude these observations.

**Task 5: Using logical conditions to examine subsets of observations**

Let's look at the difference between the oil exporting countries and the others. We can get separate summary statistics for the two groups by using an **if** statement to condition on the **nonoil** variable (recall that non-exporters are 1s and exporters are 0s)

```
sum gdp* if nonoil==1
sum gdp* if nonoil==0
```

Note the double **==**. This is the logical operator. Don't confuse it with the single **=**, which is the assignment operator.

Also note the position of the comma. In Stata, an **if** condition goes BEFORE the comma; options for the command go AFTER the comma.

Looking at the output, we can see that the oil exporters have a much higher mean and variance of GDP per worker than the non-exporters (especially in 1960).

We can see that a lot of our previously outlier-dominated graphs become a lot more reasonable when we look only at non-oil exporters

```
graph matrix gdp60 gdp85 pop igdp school if nonoil==1
```

Zooming in on the GDP per worker 1960 vs. 1985 graph, we can see that we have a new set of outliers—one "bad" and a few "good"

```
scatter gdp85 gdp60 if nonoil==1, mlabel(c_code)
```

Now we can see that the "bad" (i.e., negative growth) outlier is Venezuela. Venezuela is an oil exporting country, but exports are apparently not a big enough fraction of their exports to meet the MRW criteria for exclusion. The "good" (i.e., lots of growth) outliers were the growth miracles of Japan, Singapore and Hong Kong. But it's very difficult to see some of the labels for points which are not outliers, particularly the points bunched together at the origin (poor countries which remained poor). In the next section, we'll try to spread some of these points apart by taking natural logarithms of GDP per worker.

But first we need to decide what to do about the oil exporters. Since MRW use only the **nonoil** sample for their entire analysis, we will follow their lead and omit oil exporters from ours. There are a few ways to do this, but the easiest is to use a logical operator to drop the data

```
drop if nonoil==0
```

Q: How would you summarize GDP per worker for all observations where **igdp** is below its median value?

(A: First use **sum** with the **detail** option to get the 50% percentile, i.e., the median. Then use this value in the **if** condition for your second **sum** command.)

**Task 6: Creating new variables**

You create new variables in Stata using the **gen** ("generate") command.

A legal name for a new variable must start with an alphabetic character, and can also contain numbers as well as the underscore character "_".

Important: Stata is *case-sensitive*. Stata commands are always lower case, and if you use uppercase or mixed case, it won't know what you are talking about. Similarly, if you create a variable that has upper case in it, you must use *exactly* that mix of upper and lower case to refer to the variables. For this reason you are probably best off if you use only lower case (lower case is easier to type than all upper case, and is what most people do).

Probably the most common new variable we'll be creating throughout the year will be logs. Create new logs of variables as follows:

```
gen lngdp60 = ln(gdp60)
gen lngdp85 = ln(gdp85)
```

The syntax here is [generate] [name] [=] [calculation or logical statement]. We could have put nearly anything in the "name" slot. If we had written **gen Darth_Vader = ln(gdp60)**, then we would have had a log GDP variable named **Darth_Vader**. But that may not have been as clear, so we gave the new variables more functional names.

Note the use of the single **=**. This is the assignment operator. Don't confuse it with the double **==**, which is the logical operator – see above.

Now we can graph the data in logs

```
scatter lngdp85 lngdp60, mlabel(c_code)
```

See how the observations have been spread apart? It's still a bit tricky to see some of them, though. We can make this better by getting rid of the blue dots with **msymbol(i)**, shrinking the font size of the labels with **mlabsize(vsmall)**, and then centering the text where the dots used to be with **mlabpos(center)**

```
scatter lngdp85 lngdp60, mlabel(c_code) mlabpos(center)
msymbol(i) mlabsize(vsmall)
```

Another variable we'll want to generate is the average growth rate over the 25 year sample period. We can do this easily with the log variables we've just made:

```
gen growth = (lngdp85 – lngdp60) / 25
```

We now want to label this variable. One way to do this is to right-click on the on growth and click "Edit Variable Label…", but you can also just paste this into the command window:

```
label variable growth "average growth rate"
```

In the next section, we'll play with this growth rate a bit to see where it comes from and remind ourselves of some of the basic features of bivariate versus multiple regression.

**Task 7: Regression and correlation**

Let's look at the relationship between the log of GDP per worker in 1960 and growth over the subsequent 25 years

```
scatter growth lngdp60
```

It's not obvious (to me anyway) how strong the relationship is between these two variables (or even what sign it has), so let's run a regression.

```
reg growth lngdp60
```

Ok, so the coefficient we get is about .0038 with a p-value of .06. What does it mean? It means that the value is positive and marginally statistically significant. What does that imply about convergence in the level of GDP per capita?

*…(this is where you should think for a few seconds before reading on)…*

It implies not only that countries are not converging, but that they are diverging—countries with higher initial GDP per worker also had higher growth rates over the next 25 years. That's bad!

Let's look one more time at the scatterplot, but this time we'll put a slope in the data to show the regression we've estimated. We can graph the relationship between two variables using the `lfit` command. You can make a basic `lfit` graph like this:

```
twoway lfit growth lngdp60
```

But it's really much better to see the data and the linear fit simultaneously. To do this, we need to make better use of the `twoway` graphing command to simultaneously plot a scatter of `gdp60` vs `growth` and a linear fit of the relationship

```
twoway (scatter growth lngdp60) (lfit growth lngdp60)
```

Hopefully the syntax of this command is clear enough: `twoway` indicates that we are graphing more than one relationship, and the stuff in the parentheses tells you what

the relationships are. Note that the "two" in **twoway** refers to the number of axes. You can actually put in three (or more) relationships at once.

Now look at the graph: what does it mean? Are poor countries doomed?

No. Before we can conclude anything about the relationship between growth and the initial level of GDP, we need to control for other relevant variables.

Let's look again at the relationships among our primary variables of interest. Last time we made scatterplots using graph matrix, but we can also look at the relationships between these variables by examining their correlation matrices. You can get these with the **corr** command:

```
corr growth lngdp60 lngdp85 pop igdp school
```

Now you can see that growth isn't just correlated with GDP in 1960, it's also correlated with school enrolment, investment rates and negatively with population growth rates. Furthermore, GDP in 1960 is correlated strongly with these three variables, so it's possible that the relationship that we think we saw between growth and initial GDP was actually a result of these other—unseen—relationships. We can check this by running a multiple regression

```
reg growth lngdp60 pop igdp school
```

Look at that! The relationship between growth and initial GDP per worker has completely reversed: it used to be significantly positive, and now it's highly significantly negative. This means that the positive relationship between growth and initial GDP we saw when we ran a bivariate regression was all wrong; it just looked positive because the poor countries initially were also the ones with low investment and few schools. They grew slowly because they underinvested in physical and human capital. This means that there is convergence, but it is only conditional convergence.

**Guide to Commands, Lab 1**

**Miscellaneous terminology**

Your dataset is like a matrix of rows and columns.
A *variable* is a column in your dataset.
An *observation* is a row in your dataset.
In Stata, datasets are loaded in memory before they can be worked with.
In what follows below, [] indicates contents that you can change. Don't actually type the [ or the ] when you want to execute the command!

Important: Stata is case-sensitive. All Stata commands are in lower case. Variables can be in lower, upper, or mixed case, but when typing a variable name it has to match eXaCtLy.

**Help commands**

| | |
|---|---|
| **help** [keyword] | Provides help on the command or topic |
| **search** [keyword1 keyword2 …] | Searches Stata documentation for relevant entries on keyword1, keyword2, etc. Any number of keywords can be used. |
| **findit** [keyword1 keyword2 …] | Like "search" but searches both Stata documentation AND the internet (where user-written add-ins can be found). |

**Inputting data**

| | |
|---|---|
| **use** [filename] | Loads the data in filename.dta. You don't have to include ".dta". |
| **use** [filename] **, clear** | The "clear" option says that it's OK to replace the data currently in memory. |
| **sysuse** [filename]<br><br>**sysuse** [filename] **, clear** | Like "use", but loads data in a sample dataset provided by StataCorp. |
| **webuse** [filename]<br><br>**webuse** [filename] **, clear** | Like "use", but loads data from an internet source. |
| **clear** | Clears memory, i.e., drops all variables in the dataset. (Don't worry – it does not delete the dataset from where it is stored on the network or hard drive.) |

**Describing data**

| | |
|---|---|
| **desc** | Describes the data in memory. |
| **list** | Lists the data in memory. |
| **List** [var1 var2 …] | As above, but lists just the variables var1, var2, … |
| **list** [var1 var2 …] **if** [cond] | As above, but lists just the variables var1, var2, …, for which individual observations satisfy a the condition in "cond". |
| **sum** | Short for "summarize". Provides basic descriptive statistics for all variables in memory: number of observations, means, SDs, mins and maxes. |
| **sum** [var1 var2 …] | As above, but summarizes just the variables var1, var2, … |
| **sum** [var1 var2 …] **, detail** | As above, but also provides median, skewness, and other detailed statistics. |
| **sum** [var1 var2 …] **if** [cond] | As above, but summarizes just the variables var1, var2, …, for which individual observations satisfy a the condition in "cond". |
| **tab** [var1] | If var1 is a categorical variable, provides a one-way table of frequencies. |
| **tab** [var1] **, miss** | As above, but also display the frequencies of missing values. Note the "," – this is standard Stata syntax for separating the main command (tab) from the option (miss). |
| **tab** [var1 var2] | If var1 and var2 are categorical variables, provides a two-way table of frequencies ("cross-tab"). |
| **tab** [var1 var2] **, miss** | As above, but also display the frequencies of missing values. |
| **tab** [var1 var2] **, row col** | As above, but also display percentages by row and column. |

**Manipulating data**

| | |
|---|---|
| **gen** [var] **=** [expression] | Creates a new variable called "var" that is equal to [expression]. Important: the single "=" here is the assignment operator. Don't confuse it with the double "==" logical operator! |
| **replace** [var] **=** [expression] | Replaces the contents of the existing variable "var" with contents specified by [expression]. |
| **replace** [var] **=** [expression] **if** [cond] | As above, but replaces "var" only for observations that satisfy condition [cond]. |

**Estimation commands**

| | |
|---|---|
| **corr** [var1 var2 …] | Provides a correlation matrix for variables var1, var2, …. Note: the correlation matrix is *casewise* – if any variable in [var1, var2, …] is missing, the *entire observation* is omitted from the calculation. |
| **corr** [var1 var2 …] **, cov** | As above, but reported a covariance matrix instead. Note the "," – this is standard Stata syntax for separating the main command (corr) from the option (cov). |
| **corr** [var1 var2 …] **if** [cond] **, cov** | As above, but use only observations satisfying the logical condition [cond]. |
| **reg var1 var2 var3** … | Run an OLS regression with var1 as the dependent variable and var2, var3, …, as independent variables (regressors). |
| **reg var1 var2 var3** … **if** [cond] | As above, but use only observations satisfying the logical condition [cond]. |

**Graph commands**

| | |
|---|---|
| **hist** [var1] | Creates a histogram of var1, with the density on the vertical axis. |
| **hist** [var1] **, percent** | As above, but with percentages on the vertical axis. |
| **graph box** [var1]**, over(**[var2]**)** | Create a box plot of the variable var1, separated by var2 |
| **scatter** [var1] [var2] | Creates a scatterplot with var1 on the vertical axis and var2 on the horizontal axis. |
| **graph matrix** [var1] [var2] [var3] [var4] […] | Creates a matrix of scatterplots |
| **twoway lfit** [var1] [var2] | Plots a linear fit with var1 on the vertical axis and var2 on the horizontal axis |
| **twoway (scatter** [var1] [var2]**) (lfit** [var1] [var2]**)** | Plots a linear fit on top of a scatterplot |

**Examples of arithmetic expressions in generating or replacing variables**

| | |
|---|---|
| **var1 + var2** | Addition |
| **var1 * var2** | Multiplication |
| **var1 – var2** | Subtraction |
| **var1 / var2** | Division |
| **var1^2** | Raised to the power 2 |
| **ln(var1)** | Natural log of var1 |
| **exp(var1)** | *e* to the power var1. |

**Examples of conditions (logical expressions)**

| | |
|---|---|
| … **if** [var1] **==0** | Do … only for observations where variable var1 is zero. Note the double "==". Don't confuse this with the single "=" used in creating or replacing variables! |
| … **if** [var1] **!= 0** | Do … only for observations where variable var1 is NOT zero. The "!" is the logical "not" operator. |
| … **if** [var1] **> 0** | Do … only for observations where variable var1 is strictly greater zero OR var1 is missing (remember, a missing value in Stata is like positive infinity). |
| … **if** [var1] **> 0 &** [var1] **< 50** | Do … only for observations where variable var1 is strictly greater zero AND var1 is strictly less than 50. |
| … **if** [var1] **>= 0 &** [var1] **<= 50** | Do … only for observations where variable var1 is greater than or equal to zero AND var1 is less than or equal to 50. |
| … **if** [var1] **== 0 \|** [var2] **== 0** | Do … only for observations where variable var1 is zero OR var2 is zero. On most keyboards, the "\|" is next to z. |
| **>**<br>**<**<br>**>=**<br>**<=**<br>**==**<br>**!=** | Logical operators.<br><br>Note: to avoid ambiguity in logical expressions, you can surround them with parentheses, i.e.,<br><br>… if ([var1] >= 0) & ([var1] <= 50)<br><br>means the same thing as in the example above. |