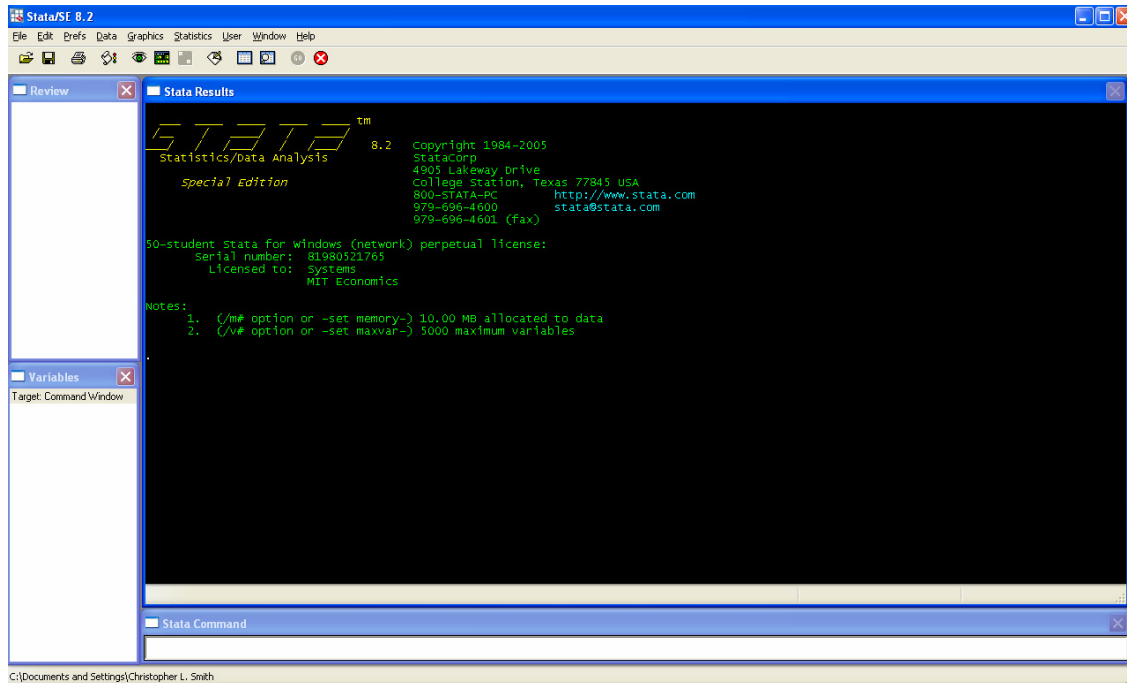




I. BASIC INTRODUCTION

THE VERY BASICS

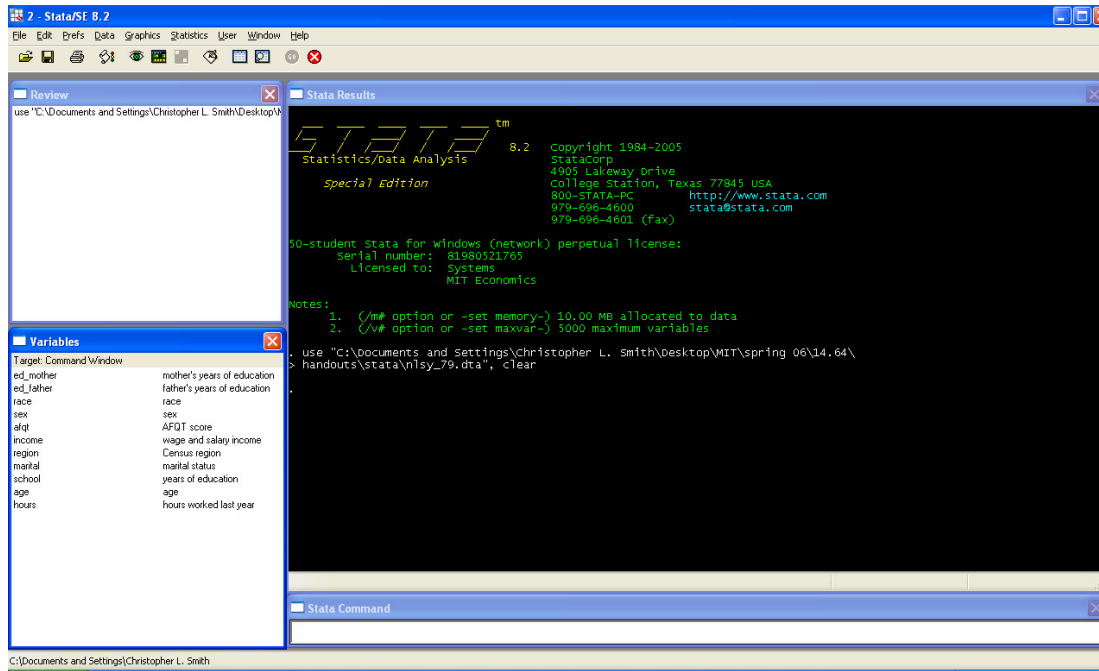
Stata is a statistical program that allows you to analyze data both graphically and quantitatively. Most computers at MIT have Stata version 8.0 or 8.2, although some might use version 9.0; for our purposes, the differences are irrelevant.




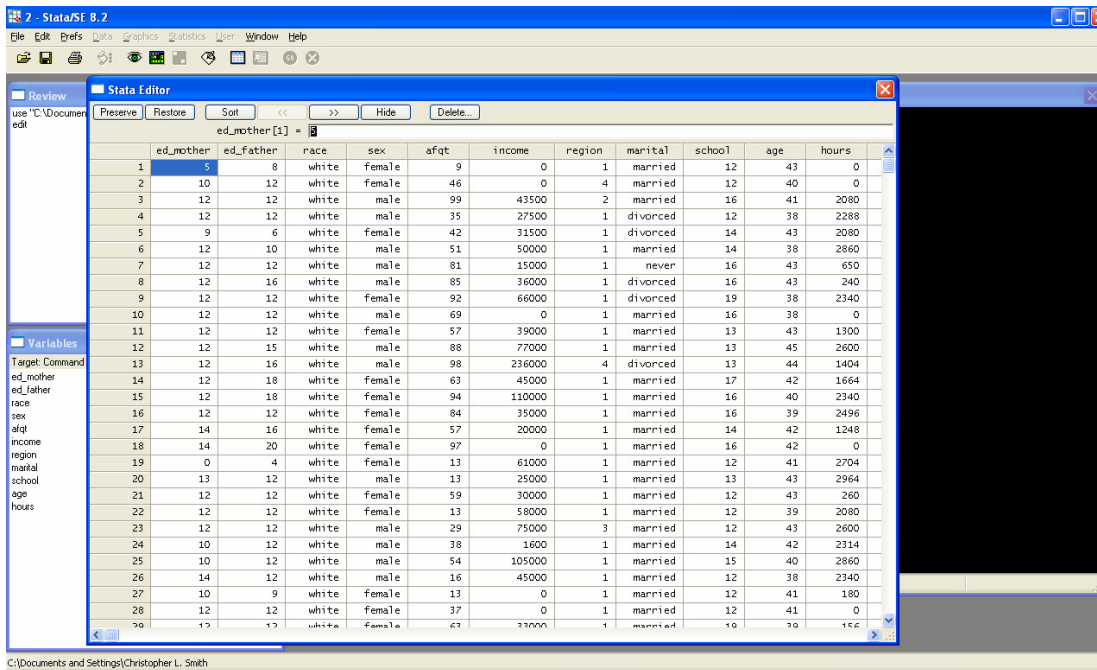
When you first run Stata, you should get a screen similar to the one above. There are four panels that we will use often. The first is *review*, which tells us what commands we have run, and lets us re-run old commands without retyping them. The second is *variables*, which will display a list of variables in our data set. The third is *Stata Command*, into which we type commands that we want Stata to run. The fourth is *Stata Results*, which will display the results from running commands.

The first thing that we want to do is load data into Stata. There are (at least) two ways to do this. The first is to click the folder icon (or File→open) and find the data file. (Note: stata files have .dta as a file extension). The second way is to open the “data editor window,” which looks like: . Clicking on that icon will open a spreadsheet-like window, which is similar to an Excel spreadsheet. If you had data from Excel that you wanted to import, for instance, you could copy your data from Excel and paste it into this data editor. Saving the data (File→save or ) saves it in Stata format (as a .dta file).

For purposes of this tutorial, let's use a dataset from the 1979 cohort National Longitudinal Survey of Youth (interviewed in 2002), which has information on earnings, labor supply, education, and some background characteristics. It's available from my website: http://web.mit.edu/clsmith/Public/stata_tutorial/nlsy_79.dta. Download it and open the data in Stata as described above. When you do, the screen should look like this:



Now, the *Variables* window has the name and description of all variables in our data set. Clicking on the data editor () allows us to take a look at our data and browse through it:

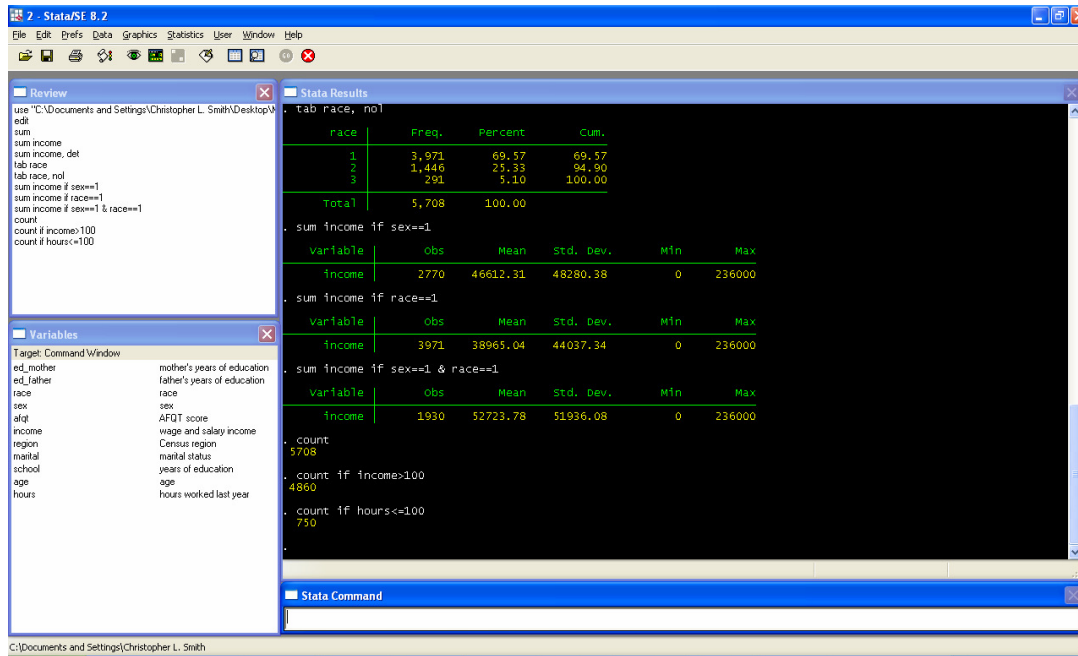


Let's close that, and try out some basic Stata commands.

To enter commands in Stata, you type them into the *Stata Command* window and hit enter. Try typing the following commands:

sum	(provides summary statistics of all variables)
sum income	(provides summary statistics of only income)
sum income, det	(provides more detailed summary statistics, such as upper and lower quantiles)
tab sex	(provides data count by sex)
tab sex, nol	(in the actual data set, the variable sex is labeled 1 if male, 2 if female. tab sex, nol provides counts as before, but this time labeled as "1" and "2" rather than "male" and "female")
tab race	
tab race, nol	
sum income if sex==1	(summarizes income for males only; note that the equal sign is actually <i>two</i> equal signs in succession)
sum income if race==1	(summarizes income for whites only)
sum income if race==1 & sex==1	
count	(simply counts the number of observations)
count if income>50	(counts number of observations with income greater than 50)
count if hours<=100	(counts number of observations with hours less than or equal to 100)
desc	(describes your dataset by telling you the number of observations, variables, and dataset size)

If you note, the *Review* window now lists each command that you've previously run:



You can click any command in the *Review* window to re-run the command again. Also, note that you do not need to type variable names each time you invoke them in a command. Instead of manually typing `sum income`, for instance, you can type `sum` and then click on the “income” variable in the *Variables* window – and `income` will be inserted into your command. Another useful shortcut: `control+r` enters the last command you ran in the *Stata Command* window. Pushing `control+r` multiple times recalls earlier commands.

GENERATING NEW VARIABLES

Suppose we want to generate a new variable that is the natural log of each individual’s imputed wage (where imputed wage is `income` divided by `hours`). To do this, simply type:

`gen lwage=ln(income/hours)` (note that we’re only using one equal sign here)

To create a dummy variable that is 1 if male, 0 if female, type:

`gen male=(sex==1)` (note the first equal sign is single, the second is double. Whenever we’re telling Stata that a variable *must* equal another, we use one equal sign; when we’re asking Stata to evaluate some relationship [whether something equals something else] we use two equal signs)

Alternatively, we could type:

```
gen male=. (creates the variable male, but all values for male  
are missing – that's what the period implies)  
replace male=1 if sex==1  
replace male=0 if sex==2
```

To create a variable that is 1 if income is greater than \$5,000, type:

```
gen high_income=(income>5000)
```

Which is the same as:

```
gen high_income=1  
replace high_income=0 if income<=5000
```

WARNING: Stata treats missing observations as the highest possible value in your data. In our dataset, lwage is missing for any observations that had 0 income or 0 hours reported (if income was 0, the calculated wage would be 0, and it would be impossible to take the log; if hours was 0, the wages couldn't be calculated, and nor could log wage). If you type count if lwage>2, for instance, then the count given by Stata will include all individuals with lwage>2 IN ADDITION TO all individuals with a missing lwage.

To delete a variable, type:

```
drop male
```

RUNNING REGRESSIONS

Let's regress log wages on education, experience (age-education-6), experience squared, AFQT score (purportedly a test score measure that is similar to an IQ score) and dummies for sex and race. We already have wages, but we need to create the experience and experience squared variable, as well as dummy variables for sex and race.

```
gen exp=age-school-6  
gen exp2=exp^2  
gen male=(sex==1)  
gen white=(race==1)  
gen black=(race==2) (race=3 is the excluded group, "other")
```

```
reg lwage school exp exp2 afqt male white black
```

which should give you the familiar Stata output. If you wanted to restrict your regression to whites, then you would type:

```
reg lwage school exp exp2 afqt male if white==1
```

instead of generating all of those dummy variables, a faster way to run the same regression is:

```
xi: reg lwage school exp exp2 afqt i.sex i.race
```

Here, xi is first used to let Stata know that you will want it to generate and use dummy variables in the regression. i.sex tells Stata that you want it to include a dummy variable for every unique value of sex (i.e. a dummy variable for male or female), and i.race indicates the same thing for race.

Suppose you want to run the regression first on males, then on females:

```
sort sex  
by sex: reg lwage school exp exp2
```

(whenever we want to use a by command, which we will in the next step, we first need to sort by the variable of the by statement)
(note: we cannot use xi with a by command)

Or, if you want to run a regression separately for each combination of sex/race:


```
sort sex race  
by sex race: reg lwage school exp exp2
```

(Stata first sorts on sex, then on race)

SAVING YOUR OUTPUT

For now, the easiest way to save your regression output is either to 1) highlight the output, copy it to the clipboard (using the “copy table” option under edit – NOT the regular control+c copy command), and paste it into word, excel, etc., or 2) create a log file, run your commands, and close the log file.

Log files are simply files which record all commands you type while the log is open, as well as the output from those commands. Then, you can open the log in Stata, notepad, emacs, Word, etc. and reformat your output.

To open a log, click on . Stata then asks you to choose a location on your computer to save your log. Once you’ve done that, the log file is open; everything you now type (and resulting output) will be recorded to the log. To close the log, click on the same icon. You can open previously created log files in Stata by clicking the same button when no log is open and choosing a pre-existing log file (or, just open the same file in Word, notepad, or another text editor).

There are more efficient ways of outputting regression results (outreg, for instance), but we need not go into that yet.


MAKING PLOTS

Syntax for creating attractive graphs, diagrams, and figures in Stata is unfortunately complex and picky (i.e. you need to know the precise syntax for changing colors, changing sizes, changing labels, changing positions of text, changing line width, altering the legend, etc.) It's worth learning the commands eventually, if you plan on doing much work in Stata, but the most user-friendly way of making graphs is simply to use graphing options under the graphics menu. If you've got the data set you want to use already loaded in to Stata, just choose the graph you want to make under Graphics → Easy Graphs – or for more flexibility, choose Graphics → Twoway Graphs.

To export the graph, simply click inside the graph window and copy (control+c or Edit → Copy Graph). Now just paste it into Word, Excel, an image editing program, etc.

CREATING A DO FILE

We've discussed how to execute commands in Stata from the command line. If you've got lots of commands you want to run in sequence – as a program – the more efficient way to do it is to create and run a do file. A do file lists all of the commands you want Stata to run; once it's created, you tell Stata to run the do file, and it implements the commands in order. You can write a do file in any text editor (and save it with the .do extension), or use Stata's do file editor, which is accessible from Window → Do-File Editor.

Simply type in all the commands you want Stata to execute, and hit . Your program should now execute in the *Stata Commands* window. You should use do files whenever possible, as they provide hard evidence of what commands were used to generate your output, they allow you to access a previously run program and make minor changes, etc.

Here's an example of a do file that does most of what we've just described. To run this do file, you could either copy and paste it into a do-file editor and execute – or type `do filename.do` from the *Command* window in Stata.

* CLS Feb 17 2006

* the asterix denotes a comment, which Stata will ignore and not execute. It's a good idea to label your do files with date and identifying info.

set more off

*sometimes when output fills multiple screens, you would need to push a keyboard button for the screen to advance. This prevents the need to do so.

log using "C:\Documents and Settings\Christopher L. Smith\Desktop\MIT\spring 06\14.64\handouts\stata\nlsy_79.log", replace

*tells Stata where to save the log file, and what to name it. ,replace indicates that if the file already exists, it should be replaced.

delimit ;

*the character given after # delimit tells Stata which character delimits a line break (if this command isn't listed, then Stata assumes the next line is the next command

clear;

*gets rid of any data currently in memory (Stata can only handle one dataset at a time)

use "C:\Documents and Settings\Christopher L. Smith\Desktop\MIT\spring 06\14.64\handouts\stata\nlsy_79.dta";

gen exp=age-school-6;

gen exp2=exp^2;

sort sex race;

by sex race: reg lwage school exp exp2;

log close;

lwage	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
school	.1090275	.0115691	9.42	0.000	.0863359	.1317192
exp	-.0451554	.0533297	-0.85	0.397	-.1497566	.0594458
exp2	.0012102	.0012352	0.98	0.327	-.0012124	.0036329
_cons	1.503755	.6486586	2.32	0.021	.231473	2.776038

-> sex = female, race = black

Source	SS	df	MS	Number of obs = 604		
Model	52.6634111	3	17.5544704	F(3, 600)	= 23.87	
Residual	441.280888	600	.735468147	Prob > F	= 0.0000	
				R-squared	= 0.1066	
				Adj R-squared	= 0.1022	
Total	493.944299	603	.819144775	Root MSE	= .85759	

lwage	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
school	.1327571	.023502	5.65	0.000	.0866009	.1789133
exp	.2350035	.1145493	2.05	0.041	.0100372	.4599697
exp2	-.0057203	.0026462	-2.16	0.031	-.0109172	-.0005233
_cons	-1.776259	1.388938	-1.28	0.201	-4.50403	.9515112

-> sex = female, race = other

Source	SS	df	MS	Number of obs = 120		
Model	20.7421304	3	6.91404348	F(3, 116)	= 16.15	
Residual	49.6501192	116	.428018269	Prob > F	= 0.0000	
				R-squared	= 0.2947	
				Adj R-squared	= 0.2764	
Total	70.3922496	119	.591531509	Root MSE	= .65423	

lwage	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
school	.1921169	.0302773	6.35	0.000	.1321489	.2520848
exp	-.0544888	.120322	-0.45	0.651	-.2928017	.1838241
exp2	.0028811	.0026561	1.08	0.280	-.0023796	.0081417
_cons	-.1217879	1.495472	-0.08	0.935	-3.083758	2.840182

```
. xi: reg lwage school exp exp2 i.race i.sex;
i.race      _Irace_1-3      (naturally coded; _Irace_1 omitted)
i.sex      _Isex_1-2      (naturally coded; _Isex_1 omitted)
```

Source	SS	df	MS	Number of obs = 4803		
Model	609.307633	6	101.551272	F(6, 4796)	= 157.16	
Residual	3098.96873	4796	.646156949	Prob > F	= 0.0000	
				R-squared	= 0.1643	
				Adj R-squared	= 0.1633	
Total	3708.27636	4802	.77223581	Root MSE	= .80384	

lwage	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
school	.1303099	.0070965	18.36	0.000	.1163975	.1442223
exp	.0198683	.0328834	0.60	0.546	-.0445982	.0843348
exp2	-.0002638	.0007567	-0.35	0.727	-.0017472	.0012196
_Irace_2	-.2526366	.0273265	-9.25	0.000	-.3062091	-.1990641
_Irace_3	-.0344723	.0545512	-0.63	0.527	-.1414176	.0724731
_Isex_2	-.3242104	.023259	-13.94	0.000	-.3698087	-.2786121
_cons	.854366	.4030377	2.12	0.034	.0642272	1.644505

```
. log close;
log: C:\Documents and Settings\Christopher L. Smith\Desktop\MIT\spring 06\14.64\handouts\stata\nlsy_7
> 9.log
log type: text
closed on: 17 Feb 2006, 17:15:30
```

ADDITIONAL REFERENCE

This basic introduction should be enough for doing simple data manipulation, running regressions, and making simple graphics. The best way to learn programming in Stata is to read other peoples' code and figure out what commands they used and why – and (less fun, but still instructional) trial and error. For a review of additional useful Stata commands, see: http://www.ugr.es/~dpto_prev/Introduccion%20a%20Stata8.pdf