A quick introduction to STATA

Data files and other resources for the course book "Introduction to Econometrics" by Stock and Watson is available on: http://wps.aw.com/aw_stock_ie_3/178/45691/11696965.cw/index.html

Load data

- If working directory is set to the folder with the data-file and the file is a .dta file simply write use dataname.dta. For other alternatives see Stata course session 2.

The Data Editor

- Choose **Data Editor** from the menu or enter *edit* in the command window
- Stata is case sensitive
- Numeric and string data are entered in the same way (do not need quotation mark around strings) if you do not have blanks in the strings (e.g. *"string variable"* and *String*).
- Missing values for numerical variables are recorded as '.'. Can enter them by entering period or entering nothing by pressing *Enter*.
- Missing values for string variables are just empty strings. Can enter them by pressing *Enter*. They will be indicated by "".
- Editors initial variables names *var1, var2…*

A variable name must be 1 to 32 characters long (can use letters, digits, and underscores, BUT: no paces or other characters). The first character must be a letter or underscore, but the latter is not recommended.

If you write edit or browse in the command box the spreadsheet window will pop up. You can in the properties window (lower right corner) read (and edit in edit mode) variable name, label and type.

List

List is similar to browse. While the latter shows the data in Stata browser the former shows the data in the result window. To interrupt a Stata command and to return to the state before you used a command use **Break** button or press **Ctrl-Break**.

The viewer

- To open the viwer go to window – viewer – new viewer or press CTRL+7
- The Viewer is a kind of browser (i.e. you can click on the links to execute commands)
- You can open your log-files in the viewer to see previous Stata sessions.

Help

- Type –help commandname- in command window to help on a specific command
- Type –findit keywords- in command window to search for information on a topic.

The command syntax:

The command syntax is almost always on the general form:

[by *varlist*:] *command* [*varlist*] [if *exp*] [in *range*] [**,***options*]

Where:

- *varlist* refers to a list of variables, e.g. *mpg weight length price.*
- *command* refers to the specific command you want to run
- *exp* refers to a logical expression
- *range* refers to a range of line numbers
- *options*, will depend on the command in question. The options must be specified at the end of the command line, after a comma separator.

The brackets indicate that specification is optional. The [by *varlist*:] formulation is optional and specifies that the command is to be repeated for each variable in the variable list. Not all commands can use this formulation.

Some useful commands:

gen	label	regress
egen	describe	rename
save	list	merge
correlate	count	collapse
summarize	mark	test
tabulate	drop	predict
sort	keep	clear

The by syntax:

- NB! To use **by**, the data should be sorted by the **by** -variables (sort varname(s)) or use **bysort**.
- When **by** is used in front of a command - **by** as a prefix - the command is repeated for group of observations for which the values of the variables in *varlist* are the same:
- When **by** is used after a command with a comma – **by()** as an option – informs the command what groups to use:
  command varname [if *exp*] [in *range*] , by (*groupvar*) [*options*]

- Some commands does not allow **by** in front of the command or as an option.

Examples:

- egen avprice1 = mean(price), by(rep78)
- bysort rep78: egen avprice2 = mean(price)
- bysort rep78: gen avprice=sum(price)/_N

- by rep78: gen avprice3=avprice[_N]

Note: _n and _N are useful in many difficult calculations

Logical expressions:

If you decide to use the optional [if exp] specification you must use a special syntax for logical expressions.

- == equals to
- != not equal to
- >= larger than or equal to, etc..
- & and | or

Examples:

- tabulate make rep78 if foreign= =1
- tabulate make rep78 if foreign= =1&price<4000
- tabulate make rep78 if foreign= =1**|** price<4000
- list if make==”VW Rabbit”
- list m* if rep78==3 & price !=.
- list displacement in 3

Range:

- 1 gives observation 1
- -1 gives the last observation
- 1/5 gives the first five observations.
- -5/-1 gives the last 5 observations

Options:

- The options are specific to each command. To look at the options for a command write – help command – and then the options will be listed.

Num(ber)lists:
Often you will find reference to *numlist* in the STATA syntax description. *Numlist* is simply a sequence of numbers, which can be specified in various ways. To get an overview of different ways to specify numlists, type *help numlist*. (NB! Better not to use commas in numlist as commas are not always allowed).

Examples:

- 1 2 to 4 means four numbers, 1, 2, 3, 4;
- 10 15 to 30 means five numbers 10, 15, 20, 25, 30;
- 1/3 three numbers 1, 2, 3, (3/1 – the same in reverse order);
- 4 3:1 means the same as 4 3 to 1;
- -1/2 means four numbers -1, 0, 1, 2
- 2(2)10 means the sequence 2 4 6 8 10;
- -1(.5)2.5 means the numbers -1, -.5, 0, .5, 1, 1.5, 2, 2.5

EXAMPLE GENERATE:

The command syntax is: (found by writing -help generate- in command window)

generate [type] *newvar* [:*lblname*] = exp [if *exp*] [in *range*]

- [type] is in bracket and is thus optional. It is only necessary if you want to create a string variable or have a desired given precision.
- *newvar* is the name of the new variable
- [:lbname] is optional and allows you to specify a variable label that describes the content of the new variable.
- =exp is the definition of the variable
- [if exp] is the logical expression telling to which observations the command will be executed

Some useful commands:

- **Collapse** takes a dataset and creates a new dataset containing summary statistics (sums, means, medians, etc) of the original one. Command syntax:
  collapse *clist* [*weight*] [if *exp*] [in *range*] [,by(*varlist*) cw fast]

  Example:

  collapse (mean) price mpg (median) medprice=price medmpg=mpg, by(rep78)

- **Merge** merges datasets (i.e. join corresponding observations from the dataset currently in memory (master dataset) with those from the Stata-format stored as *filename* (using dataset) into a single observations.

  merge [*varlist*] using *filename* [, nolabel update replace nokeep _merge(*varname*)]

  nokeep causes to ignore observations in the using dataset that do not have a corresponding observations in the master dataset. By default they are kept and marked _merge=2.

- **Egen** (extension to generate) creates a new variable equal to *function(arguments).* Depending on the function *arguments* refers to an expression, *varlist*, or *numlist*.

  [by]: egen [*type*] *newvar* = *fcn(arguments)* [if *exp*] [in *range*] [,options]

  Note: If you first take the command -sort rep78- and then try the commands

  - gen sumrep1 = sum(rep78)
  - egen sumrep2 = sum(rep78)

  You see that they give different solutions. The first gives the running sum while the second gives the overall sum.

Using log-files:

This facility allows you to print or save all commands you used for a session with STATA. It is particularly useful when you hand in written papers in class, so that the teacher can see how you obtained your results. To start logging a session, type -log using sessionname-, where *sessionname* is the name you decide for the session.

When the session is completed, type -log close-. In the results window you will now be told where the log file is saved. When you want to view or print the log file you type: -view address\sessionname.smcl-, or use the Menu.

If you want to do some calculations that should not be logged type – log off- and then when you want to start logging again write –log on- in the command window.

Linear regressions:

To fit simple or multiple linear regressions, use the *regress* command which carries out an OLS regression of the variable *depvar* on list of variables. When if *expression* is used, then the regression is estimated using observations for which *expression* is true. The option *robust* tells STATA to use the heteroskedasticity-robust formula for the standard errors of the coefficient estimators. The option *noconstant* tells STATA not to include a constant (intercept) in the regression. Level (#) specifies the confidence level, in percent, for confidence intervals. The default is level (95).

regress *depvar* [*varlist*] [*weight*] [if *exp*] [in *range*] [,level (#)]

When **predict** command follows the regress command, the OLS predicted values or residuals are calculated and saved under the name *newvarname*. When the option *residuals* is used, the residuals are computed; otherwise the predicted values are computed.

predict [*type*] *newvarname* [if *exp*] [in *range*] [, *statistic*] ,

where statistics can be:

- xb x_j b, fitted values (the default), calculates linear prediction
- residuals residuals
- stdp standard error of the prediction (of the fitted value)
- stdf standard error of the forecast (of the future value)
- stdr standard error of the residuals

The command **test** is used to test hypotheses about regression coefficients. It can be used to test many types of hypotheses. The most common use of this command is to carry out a joint test that several coefficients are equal to zero. Used this way, the form of the command is test list of variables where the test is to be carried out on the coefficients corresponding to the variables given in list of variables.

The command **ttest** is used to test a hypothesis about the mean or the difference between two means. The command has several forms. Here are a few:

```
ttest varname = # [if expression] [, level(#) ]
```

Tests the null hypothesis that the population mean of the series *varname* is equal to #. When if expression is used, then the test is computed using observations for which expression is true. The option level(#) is the desired level of the confidence interval. If this option is not used, then a confidence level of 95% is used.

Example:

- `ttest mpg=0 if (rep78<2)`

Test that the mean of two variables is equal to eachother:

- `ttest varname1 = varname2 [if expression] [, level(#) unpaired unequal ]`

Tests the the null hypothesis that the population mean of series varname1 is equal to the population mean of series varname2. The option unpaired means that the observations are not paired (they are not panel data), and the option unequal means that the population variances may not be the same.

The do-file editor

Often you will need to type a sequence of commands several times. In this case you may place commands in a text file, so that Stata can read it and execute each command in sequence. You should then use the do-file editor (press the short cut with a picture of an envelope). In the do-file you can write in multiple lines and run them in a sequence. You can save the do-file for later use.

Loops:

Often you will want to specify loops in the do-file editor. As an **exercise**, suppose you have generated variables; year1, year2, year3, …, year5.

```
set obs 5
forvalues i = 1/5 {
egen year`i' = fill(2000 2001)
  }
```

Then you want to transform these variables to string from real numbers. You can then type

```
forvalues x = 1/5 {
        generate y`x' = string(year`x')
}
```

STATA will then perform this command successively for `x' running from 1 to 5. Note that all the definitions for *numlist* can be used with this command. To test out a loop, try the following command (you can use the command window for this one-liner)

```
forvalues x = 2/20 {
display  "I will do `x' attempts to do my homework properly"
}
```