An Introduction to

Computer Tools Course

By: Christina Felfe

Outline

1. Get familiarized with the STATA interface
2. Data management
3. Introduction to Estimation and Post-Estimation Commands
4. Efficient working methods

1. Get familiarized with the Stata

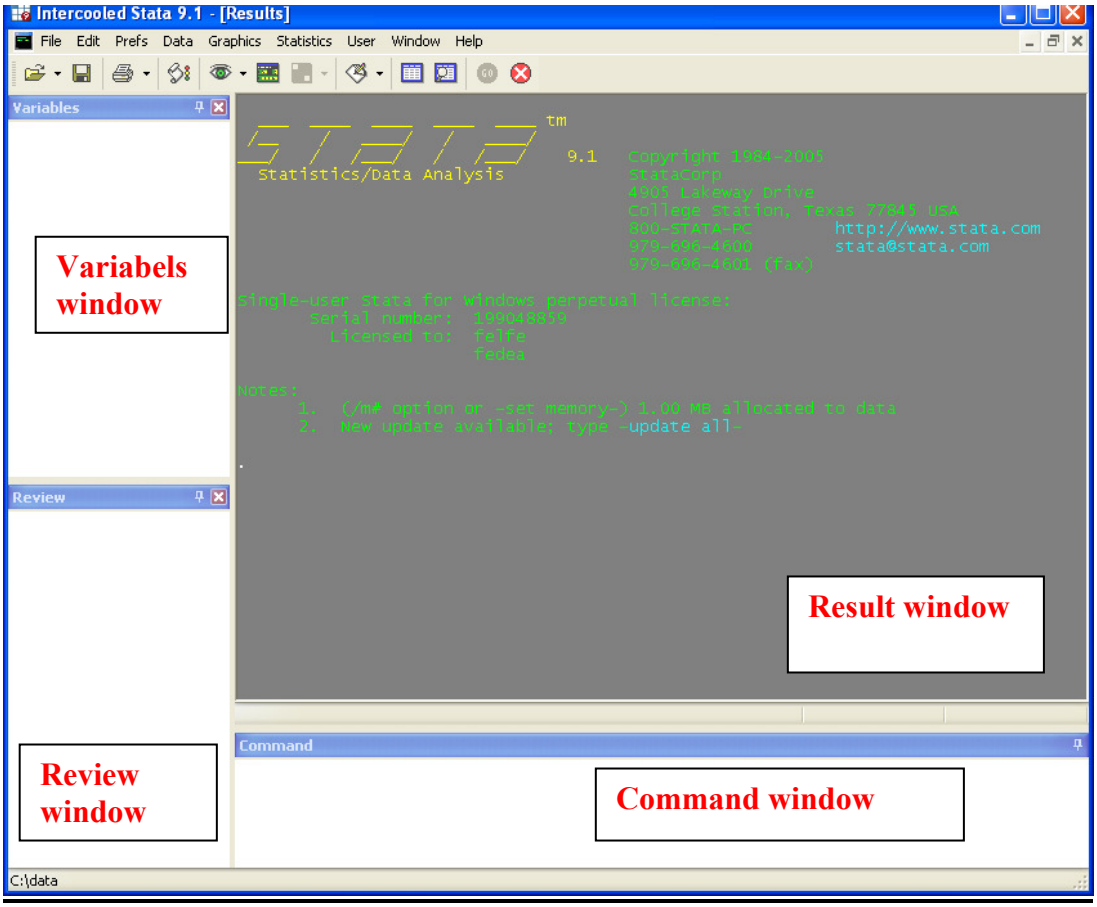
STATA is a general command-drive package for statistical analysis, data management and graphics. It can be used to enter and edit interactively data, and for statistical analysis.

a. Basic features

- STATA is case sensitive (lower case)
- You may abbreviate commands and variables as long as STATA does not get confused
- In the following STATA ***commands*** are typed bold and italic.

b. The STATA Interface

Start page



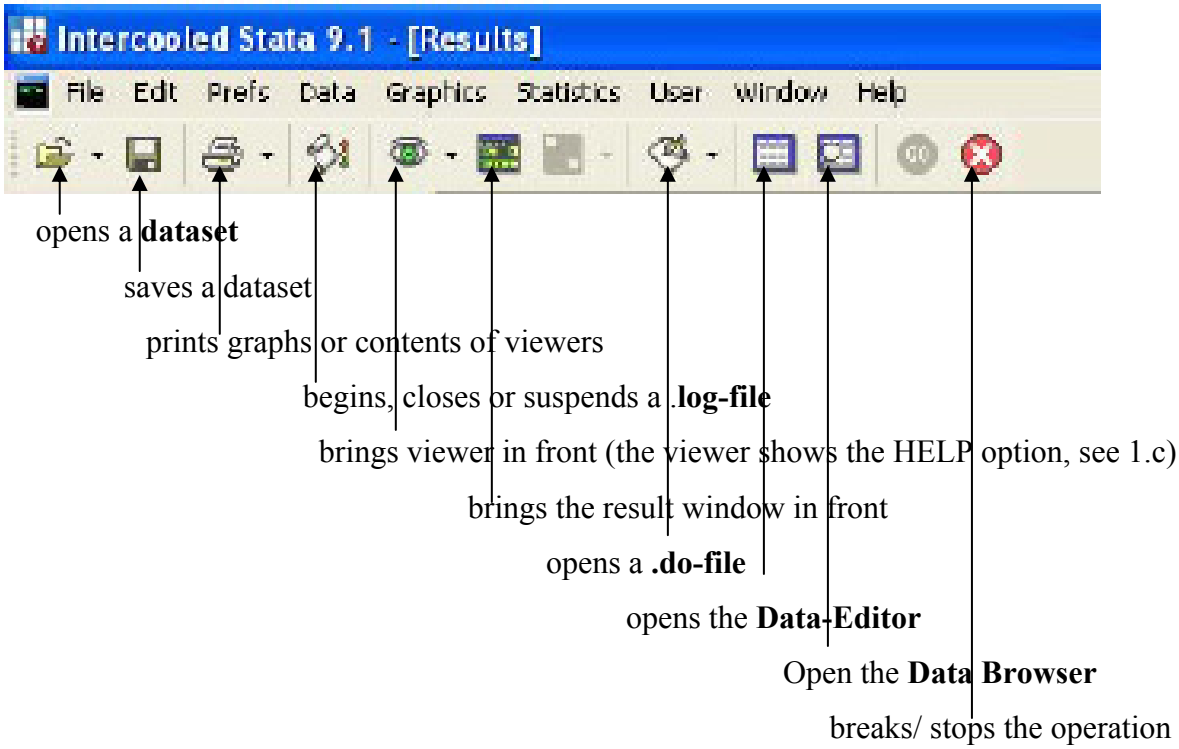
Commands window: In this window you type your commands

Results window: This window shows the continuous output of your commands

Variables window: In this window you can see the variables in the dataset used.

Review windows: In this window you can find the previously used commands, you can click them to use them again or just use the page up/down button in the command widow.

The Tool bar



Dataset: data are saved as .dta files

Data-Editor: here you can manipulate the dataset in spreadsheet format

Data-Browser: here you can only see the dataset in spreadsheet format

.log-file: copy of the result window to a log/txt - file (see chapter 4)

.**do-file:** program from where a list of commands can be executed in Stata (see chapter 4)

Menu bar

File: to open, save, view, launch .do files, save graphs, print graphs or results

Edit: copy and paste options

Prefs: change color scheme, print options, window preferences

Windows: brings various windows in front such as the command, result, variables etc.

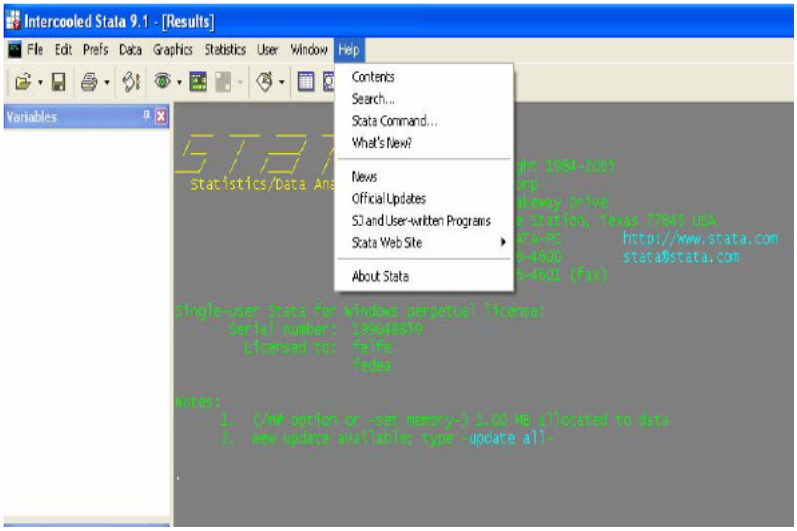
Help: contents, search for term or command, updates from the STATA website

Data/ Graphics/ Statistics/ User:

instead of typing the commands for data manipulation, estimations, creating graphs or statistics in the command window you can click several buttons

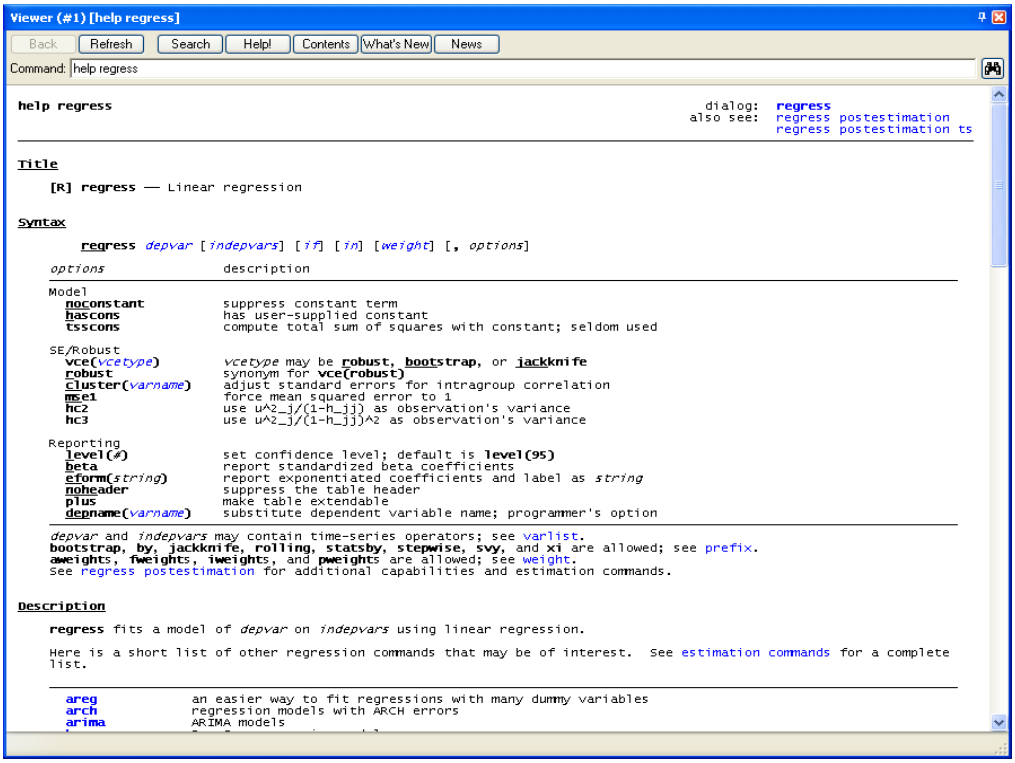
c. The Help function

The HELP Button provides an essential support for the work with STATA.



Search: If you don't know the STATA command you can type here a keyword

Command: If you know the STATA command but want to learn more about its use or further alternative parameters.



The help site has always the same structure: First you find the syntax of the command, second the description of the command and all its parameters, third examples and last you can find related commands. Whenever an expression is written in blue you can click it to find further information about it.

2. Data Management

a. How to get the data into STATA

First of all you should make sure that you are in the working directory by typing:

cd "I:\EST3\GPEM\Stata_Course"

When loading data in STATA we have to consider its format. There are three different commands for different formats:

1. *.dta:

In this case you can use directly the following command:

use nameofdataset.***dta***

2. *.txt

Now you can read the data using the following command:

infile variable1 variable2 …. variableK ***using*** nameofdataset.***txt***

Note: If the dataset a comma or tab delimited ASCI file:

insheet variable1 variable2 …. variableK ***using*** nameofdataset.***txt***

Note: The ASCII-file cannot contain the names of the variables, but only the values.
In case there is already one dataset open you should first ***clear*** the memory

Note: You can make use of the program Stat/Transfer where you can convert different kind of datasets into .dta format and then you load them by using the *use* nameofdataset.*dta*

b. Data preparation

General Command structure

The syntax is the same for all commands

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

where

by varlist	to group data (before you have to sort the data according to the varlist by the command ***sort*** varlist)
command	you find a list of commands below
in	to restrict the sample to a certain range (e.g. in 4, restricts to case 4, in 1/4, which shows cases 1 to 4)
options	various options possible depending on the command (see help) e.g. ***,replace*** replaces a variable//dataset etc. in case it already exists
if	you can restrict the sample by the following expressions:

Arithmetic	Logical	(numeric and string)
+ addition	~ not	> greater than
- subtraction	| or	< less than
* multiplication	& and	>= greater or equal
/ division		<= less than or equal
^ power		== equal
		!= not equal

Furthermore you might want to use some system variables, here a selection:

Command	System variable
_n	Index of current observation
_N	Total number of observations
_all	All variables
_b	Vector of regression coefficients
_se	Vector of standard errors of regression coefficients

Commands for a descriptive statistics

describe	provides basic information about the file and the variables (number of observations, number and names of the variables, format of the variables, labels of variables if attached)
browse	opens the data-browser where you can see the dataset
	displays all the data about the listed variables on the screen
summarize varlist	provides a summary statistic about the variables listed (frequencies, mean, min, max)
	Note: a possible option is ***, detail*** which provides more details such as percentiles, kurtosis etc.
tabulate varlist	shows all different values of the variables with their frequencies
table varlist	displays higher dimensional tables (see help for options)
correlate varlist	gives the correlation coefficient between the variables named

Commands for data manipulation

generate varname=x	to create a variable
replace varname=x	to replace the value of a variable
label variable varname	to give a variable a certain description

Note: these commands follow the common syntax
[*by* varlist:] *command* [varlist] [*if* exp] [*in* range], [*options*] and thus can be applied to a restricted sample or as an extended version. For the construction of a variable you might want to use certain mathematical or statistical functions. In the list below you find a selection.

Command	**Mathematical & Statistical Function**
ln (varx)	Natural logarithm of variable x
sqrt (varx)	Square root of variable x
sin (varx)	Sinus of variable x
cos (varx)	Cosinus of variable x
exp (varx)	e^{x} of variable x
abs (varx)	Absolute value of variable x
norm (varx)	cumulative standard normal of variable x

normden (varx)	standard normal density of variable x
uniform ()	uniform(0,1) random number
mix (varx1 varx2 … varxK)	returns the minimum of the variables x1-xK
max (varx1 varx2 … varxK)	Returns the maximum of the variables x1-xK
cond (exp, varx varz)	When the expression is true it returns variable x, else variable z
sign (varx)	Returns 1 if x>0, 0 if x=0 and 1 if x<0

Note: Depending on the mathematical/statistical function you want to use you might have to use an extension of generate:

egen varname=***function***(varx) to create a variable which is the mean, median,…of another variable

Below a selection of functions where you have to use egen:

Command	**Mathematical & Statistical Function**
mean (varx)	Mean of variable x
median (varx)	Median of variable x
sum (varx)	Sum of variable x (Note: gen varz=sum(varx) → varz= cumulative sum)
rank (varx)	Rank of variable x, highest value gets value1
count(varx)	The number of nonmissing observations of x

Data elimination

Once we have create the variables we might want to get rid of some of them. These we can either do by eliminating the variables that we are not interested anymore (var1 … varn)

drop var2 var1 …varn

or by keeping the ones that we still want to use (var1 var2 … varn)

keep var1 var2 … varn

Furthermore we can change the name of a variable (from oldname to newname)

rename oldname newname

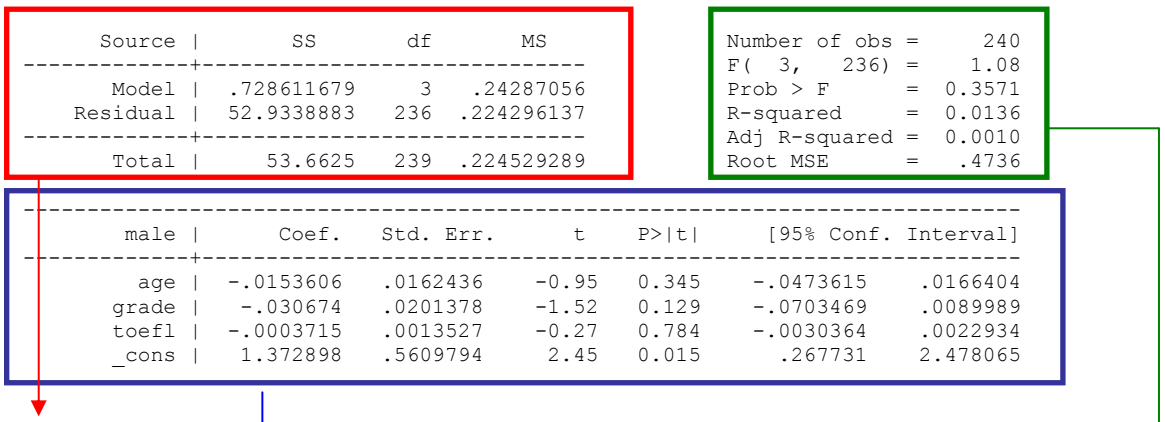
3. Estimation and Post-Estimation commands

a. Estimation commands

The command for a simple OLS regression where y is regressed on several control variables x1-xK is the following:

regress y x1 x2 … xK

The output on the result window looks the following:



```
      Source |       SS       df       MS
-------------+------------------------------
       Model |  .728611679     3   .24287056
    Residual |  52.9338883   236  .224296137
-------------+------------------------------
       Total |     53.6625   239  .224529289

Number of obs =     240
F(  3,   236) =    1.08
Prob > F      =  0.3571
R-squared     =  0.0136
Adj R-squared =  0.0010
Root MSE      =  .4736

------------------------------------------------------------------------------
        male |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         age |  -.0153606   .0162436    -0.95   0.345    -.0473615    .0166404
       grade |   -.030674   .0201378    -1.52   0.129    -.0703469    .0089989
       toefl |  -.0003715   .0013527    -0.27   0.784    -.0030364    .0022934
       _cons |   1.372898   .5609794     2.45   0.015      .267731    2.478065
```

Anova block: This block shows the sum of squares for both the explained part of the model and the residuals. df shows the degree of freedom and MS the mean squared error with respect to the degrees of freedom

Modellfit block: This block shows the number of investigated units, the F-Statistic and the adjusted R squared value.

Coefficient block: This block provides the estimation results for all control variables, incl. the coefficients, the standard errors, the value of the t-statistic and the p-value as well as the confidence interval.

For further advanced estimation procedures the following commands might be helpful:

Regression procedures	**Description**
rreg	Robust regression
ivreg	Instrumental variables regression
heckman	Heckman's selection model
probit/logit	Probit/Logit analysis
mlogit	Multinomial logit
tobit	Censored-normal and Tobit regression
arima*	Autoregressive integrated MA models
arch*	AR conditional heteroscedasiticity estimators
xt*...	Panel analysis
st*…	Survival time data
* data must be time-series/panel data/ survival data; use the command tsset panelvar timevar or stset.	

b. Post estimation commands

After having done a regression you might want to use the results for further analysis, e.g. some significance test or graphical analysis (see 3.c). In the following table you can find some useful commands to save regression results, to calculate marginal effects or to do certain types of tests. For more detailed information see the HELP function.

Post estimation command	**Description**
predict name, options	saves predictions (xb), residuals (res), influence statistics, etc.
predictnl name, options	point estimates, standard errors, testing, and inference for generalized predictions
mfx	marginal effects or elasticities in a nonlinear equation (e.g. probit, logit etc.)
estimate	cataloging estimation results
estimate store	saves estimation results
estat	AIC, BIC, VCE, and estimation sample summary
mat name= ***e(…)***	saves the estimation results such as the coefficients (e(b)), variance covariance matrix (e(V)
test	Wald tests for simple and composite linear hypotheses
testnl	Wald tests of nonlinear hypotheses
lrtest	Likelihood-ratio test (to conduct the test, both the unrestricted and the restricted models must be fitted using the ML method)
hausman	Hausman's specification test: tests if an efficient and a consistent estimator are significantly different.

c. Graphs

An easy way to create graphs is to use the graph option in the menu. There you can find the whole variety of graphs offered by STATA.
There are different graph commands in STATA:

graph plottype [var1] [var2]
graph twoway plottype [var1] [var2]

The first command draws (in the announced plot-type) a one dimensional graph of the means of the variable listed. The second graph command shows the two-dimensional relationship between the variables listed. [var1] appears on the y-axis, and [var2] on the x-axis of the graph. The default extension for Stata graph files is .gph which Stata will automatically add to the file name. You can select the font, colors, line thickness and other options by selecting Graph Preferences in the Prefs menu.
The following table gives an overview of the type of graphs you can draw in STATA

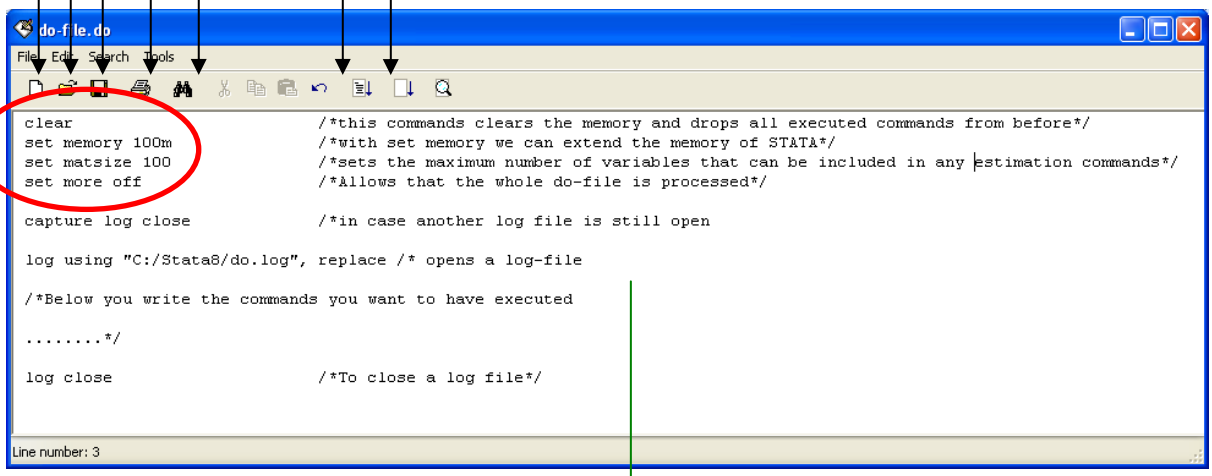
Graphics*	**Description**
twoway[plottype] vary varx plottype : ***scatter, line, bar*** etc.	2-dimensional family of plots, all of which fit on numeric y and x scales
graph bar (mean)vary***, over***(varx)	vertical bar charts (y axis is numerical, and the x axis is categorical
graph dot (mean)vary***, over***(varx)	horizontal dot charts (categorical axis is presented vertically, the numerical axis is presented horizontally)
graph pie vary***, over***(varx)	pie charts
graph matrix	matrix plot of two-way scattergrams
graph save graphname	saves a graph under "graphname" with the default extension .gph
graph use graphname.gph	redisplays the graph "graphname"
graph combine graphname1 graphname2	combines several graphs into one

4. Efficient working methods

a. do-file

In STATA we have the option to write a program of commands which we can "do" or run from STATA. A do-file allows us to execute all commands at once by just clicking the "run"-button. Below you can file an example where you can see a do-file with the basic preamble.

New do-file
Open a folder where you can find an old do-file
Save the do-file
Print the do-file
Find in the do-file
Un-do
Run the commands of the do-file



Basic Preambel: First you should eliminate all variables or observations from the memory by typing ***clear***. By ***set memory*** # you can increase the memory to run the operations wanted and by ***set matsize*** # you can set the number of variables that STATA should handle in a regression. The command ***set more off*** makes STATA running through all commands without stopping. The /*...*/ allows writing a comment that is not processed by STATA

b. log-file

The log file saves the STATA output in a txt file. It is recommendable to create always a log-file since this way you can have a look at all previous results at any time later you want. You open a log-file by ***log using*** "…/logfilename.log", ***replace*** and you close it by typing ***log close***.

c. Basic Programming

Macros

Macros store information as strings, in another word you can collect several variables under a common macroname

global macroname var1 var2 … varn
local macroname var1 var2 … varn

Both macros are working exactly the same way (you assign strings to specified macro names and can call them later by calling it `macroname'). The main difference is that a global macro may be used by any program whereas a local macro is only for private use of the program in which you define it.

Loops

while

evaluates an expression/condition and, if it is true, executes the command enclosed in the braces. Further whiles may be nested within a while.

Example: Generates 100 random variables that are uniformly distributed

```
local i = 1
while `i' < 100 {
        gen u`i' = uniform()
local i = `i' + 1
}
```

forvalue

This is the fastest way to execute a block of codes for different numeric values given in the local macro.

```
forvalues i = 1(1)100 {
    generate x`i' = uniform()
}
```

foreach

This is a fast way to execute repeatedly a certain command (enclosed in brackets) for a list of variables that before have been set in a local macro.

local varlist var1 var2
tsset timevar
foreach v of local varlist {
    gen lagged`***v'***=**l.**`***v'*** ***/*l.*** is a lag operator*/
}

if

The if command allows you to set several conditions according to which one is true a certain command will be executed.

if expression1 {
        command
}
else if expression2{
        command
}
else {
        command
}

The if command evaluates expression1 (e.g. var1>0). If the result is true, then it executes the command inside the first set of brackets. If not, it evaluates expression2 (e.g. vr1<0) and executes the second block of commands if true. If it is also not true, it will execute the last set of commands.