Guide to Stata
~~Econ B003 – Applied Economics~~

- ~~**To configure Stata in your account:**~~
  - ~~Login to WTS (use your Cluster WTS password)~~
  - ~~Double-click in the folder *Applications and Resources*~~
  - ~~Double-click in the folder *Unix Applications*~~
  - ~~Double-click in *Setup Exceed* (type any key)~~
  - ~~Double-click in the Stata icon (use your main UCL password, i.e. e-mail password)~~

~~NOTE: Stata in the WTS environment directly sets up your personal path and there is no need to specify it for any of the files that will be opened or saved during this session (you can see the current working path that Stata is using in the bottom-left corner of the Window)~~

1. OPENING DATA SETS, LOG-FILES, DO-FILES

- **To open a log-file**

```
log using [path for the file]/class1.log, replace
```

  - This command opens a file with extension .log, where it keeps records of what you have done and the results of the commands that you have executed.
  - These files can be opened later on with any editor (MS Word, for example)
  - Before opening any log-file you will need to close any other log-file that you may have opened.

```
log off
log on
log close
log using [path for the file]/class1.log, append
```

- **To open a Stata data set file: .dta**

```
use [path for the file]/income.dta
```

- **To save a Stata data set file once it has been modified**

```
save [path for the file]/income.dta, replace
```

This saves the changes that have been made in the original data set. If you want to keep the original data set, you must type a different name for the data set (e.g. income2.dta). The extension .dta is assumed by Stata.

- **To clear the memory**

```
clear
```

If the data set has not been saved before doing `clear`, the changes will be lost and only the changes done before saving the data set will be saved.

- **Do-files**

Sometimes you will find it useful to write all the commands in a text processor and then ask Stata to execute all the commands that you have written in this file. You can do this using *do-files.* An advantage of these files is that you can use the same file as many times as you want and you do not need to write the commands in the Command window every time you want to execute them. In addition, you can add comment lines to the file to describe what you are doing in each step.
In order to insert a comment, put /* at the beginning of the comment and */ after the comment. Stata will ignore what it is written between these two symbols.

Example: (in a text processor -i.e. MSWord or Notepad- we write)

```
/*do-file for the first tutorial*/
log using class1.dta, replace
use income.dta
describe
summarize
drop if inc==. /*we drop those observations whose income is
missing*/
save income2.dta /*saving with different name*/
clear
log close
```

After saving this file with the extension .do (e.g. class1.do) we write the following command in Stata:

`do` *`path for the file`*`\class1.do`

2. GENERAL SYNTAX OF STATA COMMANDS

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

The expression `by, if` and `in` are optional of all the Stata commands.

- o **by** `varlist:`

This option tells Stata that it has to execute the command separately for each set of observations that share the same value for the variables specified in `varlist.`
Note: in order to use the `by` option you first need to sort the observations of the data set with respect to that variable in `varlist`. This can be done in Stata by typing:

```
sort [varname]
```

- o **in** `range`

This option tells Stata the range of the observations over which we want to apply the command.

`list [variable name] in 5`	lists the 5th observation of the variable
`list [variable name] in 5/10`	lists from 5th to 10th observation
`list [variable name] in –3`	lists the third-from-the-last
`list [variable name] in 5/l`	lists from 5th to last observation

- **if** expression

After **if** a logic expression should be added in order to specify the conditions that we want the observations to satisfy in order to execute the Stata command

The logic operators that can be used to write the logic conditions are the following:

==	equal (note, 2 symbols =)
~=	different
!=	different
&	and
|	or
>= (<=)	greater (less) or equal than
> (<)	greater (less) than
!	not

Note: In Stata missing values are represented with a dot (.), in case we want to execute some commands only for the observations which have missing values for a specific variable.

Examples:

```
list if age==30
list if age==30&inc~=.
list if age=<30|age>40
```

3. DATA MANAGEMENT, GENERATION OF NEW VARIABLES

- **To describe the variables in the current data set**

```
describe
```

This command will let you find out what variables are in the data set and how they are named, with their description, the labels of the variables, the number of variables and observations.

It can be also used to describe only specific variables. For example:

```
describe inc
```

- **To label variables**

```
label variable [variable name] “comment”
```

This command allows you to document your data set so that you can make comments on the variables and give a short description (at most 31 characters long) of the variable.

- **To label the values of the variables**

For example, in the data set of our example, the variable `eth` takes only two possible values 0 and 1, but this variable is not really a numerical variable. It is useful to give a variable value 0 whenever the observation in “Non-white” and 1 whenever it is “White”. Stata can remember which category corresponds to each number by executing the following commands:

```
label define ethlabel 0 “Non-white” 1 “White”
label values eth ethlabel
```

- **To list the values of the variables**

```
label list
```

This command will tell you which values have been assigned to each of the categories of a categorical variable.

- **To summarize the variables**

```
summarize
```

This command gives a series of summary statistics of all the data in memory. In particular, it gives you the number of observations, means, variances and ranges of all the loaded variables.

In order to get more detailed statistics, such as the percentiles of each variable and higher moments (skewness and kurtosis) the following command should be used.

```
summarize, detail
```

You can also obtain the summary statistics for only a list of variables

```
summarize inc eth
sum inc eth
```

- **To list the observations**

```
list [name of variable] [if expression] [in range]
```

Examples:

```
list
```

(this command lists all the observations of all the variables in the data set)

```
list inc
```

(this command lists all the observations of the variable income)

- **To drop and keep observations/variables**

  keep [*name of variable*] [**if** *expression*] [**in** range]
  drop [*name of variable*] [**if** *expression*] [**in** range]

- **To generate new variables and replace existing ones**

  generate [*new name of variable*]=expression **if** *exp* **in** *exp*

  This command allows one to generate a new variable from the variables that already exist and you can also specify the conditions with **if** and the range with **in**.

  replace [*name of variable*]=expression **if** *exp* **in** *exp*

  This command allows one to modify a variable that already exists in memory.

  Examples:

```
generate lninc=log(inc)

generate d=1 if inc>1000
replace d=0 if d==.
```

The same result could be obtained if we use the following command in Stata:

```
generate d=(inc>1000)
```

(this command generates a variable **d** that takes value 1 if the expression inside the brackets is true and 0 if it is false)

4. HISTOGRAMS, PLOTS AND GRAPHS

- **To draw a histogram**

  graph [*varname*], histogram bin(#)

  # is the number if intervals we want to specify (#=5 is the default)

  Note that we would have obtained the same by typing

  graph [*varname*], bin(#)

because when we only have one variable Stata graphs a histogram as a default.

If you include `normal` at the end of the command
`graph [`*`varname`*`], histogram bin(#) normal`
a line for the normal distribution appears so that you can compare whether your distribution looks like a normal or it is very different.
Examples:

- `graph inc, histogram bin(25)`
- `sort eth`
  `graph inc, by(eth) histogram bin(25)`
  (this graphs two histograms. One for each possible values of eth)

- **Scatter plot of two variables**

```
plot var1 var2
```

, where `var1` is the y-axis variable and `var2` is the x-axis variable.

Otherwise, you can use the following command will produce a better quality graph:

```
graph var1 var2
```

5. TABLES OF FREQUENCIES AND CORRELATION

- **To tabulate variables: one-way frequency tables**

This command allows one to create tables showing the frequency of each possible value of the variable that has been specified in the command.

`tabulate [`*`varname`*`]` **`if`**`[]` **`in`**`[]`

- **To tabulate variables: two-way frequency tables**

This command allows one to create tables showing the frequency for all the possible values of two variables simultaneously.

`tabulate var1 var2` **`if`**`[]` **`in`**`[]`

Example:

```
tabulate d eth


tabulate d eth, column

tabulate d eth, row

tabulate d eth if inc~=., column
```

- **Correlation and covariances between two variables**

  ```
  correlate
  ```
  (this command computes the correlation coefficient between all the possible pairs of variables in memory)

  ```
  correlate var1 var2
  ```
  (this command computes the correlation coefficient between the two variables specified)

  ```
  correlate var1 var2, covariance
  ```
  (computes the covariance between the two variables instead of the correlation coefficient)

6. REGRESSION ANALYSIS

You can do the regression analysis by

```
regress dep_var x1 x2 x3
```

This command executes the regression of the dependent variable (`dep_var`) on the independent variables or regressors (`x1,x2,x3..`). Thus, the first variable in the list of variables is the dependent variable, and then we write the regressors.
Stata automatically adds a constant.
The option `,noconstant` has to be added to the command as follows if we do not want to include a constant in the regression:

```
        regress dep_var x1 x2 x3, noconstant
```

Stata automatically shows many statistics relating to the regression (number of observations, R^2, t-statistic for each coefficient of the regression)

Statistical Tests for the values of the coefficients in the regression:
After regressing a particular model with the command `regress`

Individual tests:

`test` *`name of the variable`*`=value`

(this command tests the null hypothesis that the coefficient of the variable specified is statistically equal to the `value`)

Joint tests:

```
test var1=value1
test var2=value2, accumulate
```

(this command tests the joint null hypothesis that the coefficient of the variable `var1` is equal to the `value1` and that the coefficient of `var2` is equal to the `value2`)