Chapter 1: Introduction to STATA

Pre-requisites:

- Basic econometrics and statistics

Contents

1. Introduction

The goal of this chapter is to introduce STATA, the statistical software that we will be using over these notes. In this chapter, we will learn how to use STATA to perform basic data management,

conduct hypothesis testing and run basic regressions. STATA is very popular statistical software that offers a large number of econometric techniques and is widely used among economists and other social scientists. We hope you would appreciate the advantages of STATA!

This course is based on STATA 14 (released in April 2015), although for most of what we are going to cover a previous version would be fine. Notice, however, that starting in version 13, STATA has extended its coverage of treatment effects and power analysis, including two new manuals about these topics. Therefore, it might be worth to consider getting versions 14 or 13 if you expect to work in this area in the near future. We will try to be as general as possible, so readers with previous versions can easily follow the chapter material.

This chapter will demand a fair amount of work with large databases. While we will provide you simplified datasets for pedagogical purposes, in real life "painful" datasets are prevalent. Problem sets will give us the opportunity to expose you to more complex cases in which you would need to think more carefully in common empirical problems like handling missing data, nonresponse, and attrition, just to mention some of the most common ones.

We also want to take this opportunity to provide some ideas about how to organize your empirical work. In practice, complex household surveys are made of several data files that you would need to organize in a single file. Typically, this includes data for different levels: individuals, households and communities. In an educational program, for instance, you will usually have information about students, classroom and schools characteristics for your evaluation. This means that, before running your regression, you will need to merge several files and create variables. This demands some level of organization. This is where do files will play an important role.

At the end of this session, you should be able to:

- Perform basic operations in STATA (read data, create and transform variables, use log and do files, graphs, etc.).
- Run linear regressions (and some extensions) in STATA.
- Carry out hypothesis testing in STATA.
- Be familiar with basic commands for more complex data management and analysis.

2. Introduction to dataset: Mexican ENCEL 2007

The example database to be used in this chapter comes from a subset of variables of the "Survey of Rural Households Evaluation-ENCEL 2007". It contains relevant information for the external evaluation of Oportunidades, the famous conditional cash transfer program in Mexico. The entire database with detailed documentation can be downloaded from the official Oportunidades' website: https://prospera.gob.mx/EVALUACION/es/eval_cuant/bases_cuanti.php.

Oportunidades started in 1997. The program provides financial support for families to keep children in school. The program also incorporates health and nutrition services and conditions. Oportunidades has been subjected to multiple evaluations, taking advantage of the experimental design of the program by collecting repeated observations of the beneficiary families before and after the start of the program.

This experimental design refers to the randomized selection of 506 villages to participate in the program among 6396 locations that met the requirements. Of these villages, 320 were assigned to the

treatment group and 186 were allocated to the control group. In the period 1998-2000, six survey rounds were collected. In 2003, the seventh round was conducted to gather information on the initial 506 villages as well as for 151 additional villages with comparable characteristics but not yet exposed to the program.

The ENCEL 2007 represents the eighth survey round and contains information on the 506 localities initially assigned to treatment program, the 151 added in 2003 and, additionally, 30 cuadruplas of the town of Chiapas and Oaxaca.

Figure 1 summarizes the information of the subsample we will use in this chapter:

Figure 1: Variables in the ENCEL dataset

```
Contains data from C:\Users\Stanislao\Dropbox\Teaching\1. Current\Econometrics\4.
Handbook\2. Data\ENCEL 2007\Final\DataFinal_ENCEL07.dta
  obs:       240,273
 vars:            34                          20 Aug 2013 00:10
 size:    29,793,852 (96.4% of memory free)
------------------------------------------------------------------------------
              storage  display     value
variable name   type   format      label      variable label
------------------------------------------------------------------------------
villid          str9   %9s                    Village ID
hogid           str15  %15s                   Household ID
iid             str17  %17s                   Individual ID
D_HH            byte   %9.0g                  HH has PROGRESA participant:
                                               1=Y, 0=N
HH              byte   %9.0g                  HH head
age             float  %9.0g                  Age: years
sex             byte   %9.0g                  Gender: 1= M, 0=F
labor           float  %9.0g                  labor condition: 1: employees;
                                               0:unoccupied
enroll          byte   %9.0g                  enroll: 1= Y, 0=N
Childlab        byte   %9.0g                  labor condition: 1: Child
                                          employee or job search 0: Non
IncomeLabHH1    float  %9.0g                  HH Primary Monthly Income:
                                          Pesos
ExperienceLab~1 float  %9.0g                  HH Experience: Months
IncomeLabHH2    float  %9.0g                  HH Secondary monthly Income:
                                               Pesos
IncomeLabNH     float  %9.0g                  Non HH Monthly Income: Pesos
IncomeLab       float  %9.0g                  Monthly Income: Pesos
health          byte   %9.0g                  Health insurance: 1= Y, 0= N
famsize         float  %9.0g                  HH size
Lanhead         float  %9.0g                  Language of HH head: 1=
                                               Indigenous, 0=Non Indigenous
agehead         byte   %10.0g                 Age of HH head: years
sexhead         byte   %9.0g       sexhead    Gender of HH head: 1= M, 0=F
pov_HH          byte   %8.0g       mppob      HH Poverty Status: 1= poor,0=
                                               Non poor
HH_AssHo        float  %9.0g                  HH Asset/Housing and
                                               transportation: Pesos
HH_AssPr        float  %9.0g                  HH Asset/Productive Assets:
                                               Pesos
```

```
nprivelem       byte    %8.0g       l301c       Number of private elementary
                                                 schools in the Village
npubelem        byte    %8.0g       l301d       Number of public elementary schools in
                                                 the Village
mwagemale       float   %9.0g       l609        Village Average daily wage of males
                                                 peasant:Pesos
mwagefema       float   %9.0g       l611        Village Average daily wage of females
                                                 peasant:Pesos
mwagechil       float   %9.0g       l613        Village Average daily wage of Child
                                                 peasant:Pesos
p_elec          byte    %9.0g                   Village with some kind of electricity
                                                 system: 1=Y 0=N
p_gas           byte    %9.0g                   Village with piped gas system: 1=Y 0=N
p_dren          byte    %9.0g                   Village with public sewer: 1=Y 0=N
wheat           float   %9.0g       l627        Village price of wheat: Pesos/kg
D               byte    %9.0g                   Village participant: 1=Y 0=N
contrHH         float   %9.0g                   labor condition: 1:contract; 0:Non
                                                 contract
------------------------------------------------------------------------------
```

It is important to notice that the sample we are going to use in this part of the chapter is already clean and organized. Certainly, this is rarely the case in actual empirical work but it would be easier to introduce the basics of STATA in this way. We cover in the Appendix 1.1 of this chapter a more in depth treatment about how to work with more complicated databases.

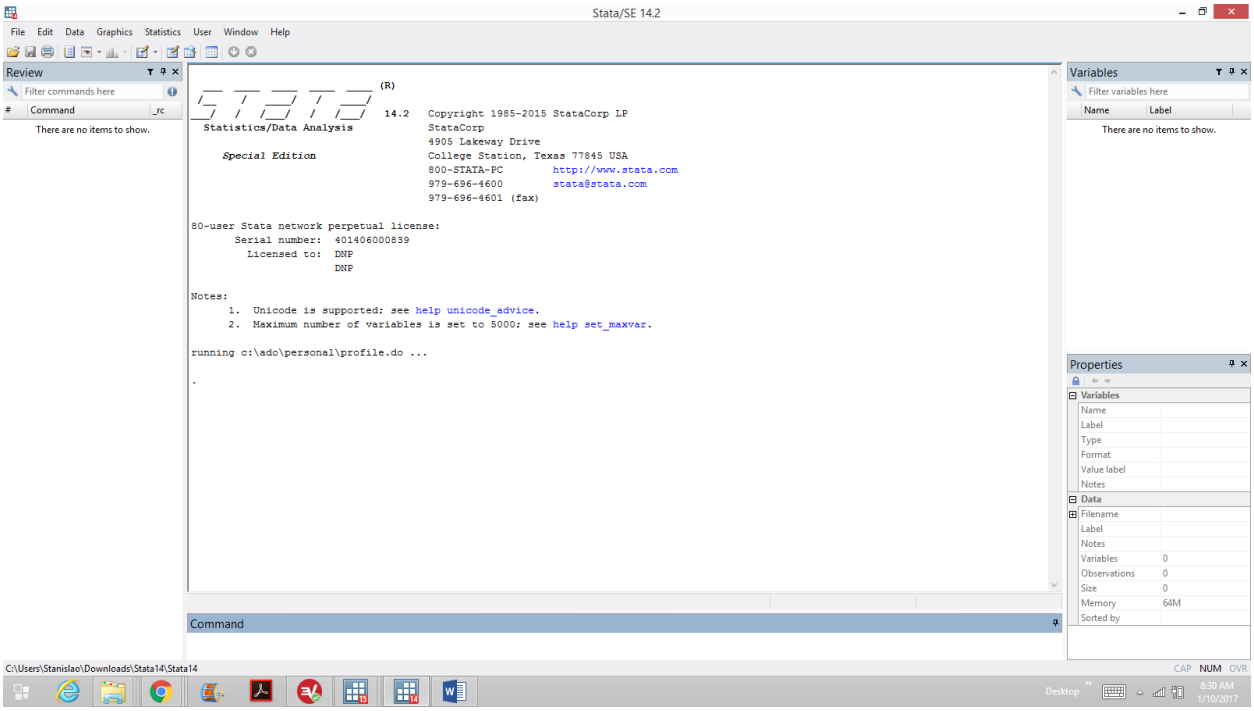
3. STATA basics

3.1. Starting STATA

STATA is an interactive data analysis software. To start STATA, double click on the STATA icon. Once the program is loaded, an initial screen with 5 windows will appear as shown in Figure 2. Check below for an explanation of each window:

- *Command window:* where you tell STATA the operations you would like to perform. You will enter this operation using the syntax or code.
- *Results window:* where results are displayed. You can observe the outcome of any task you ask STATA to perform, but it would not be possible to save the results here. We will explore alternatives to do so later.
- *Review window:* where previous STATA commands are shown. It keeps track of all operations that have been performed during the current session. Another advantage of this window is that provides a shortcut to access to previously run commands.
- *Variables window:* where all the variables in the active data file are shown. Any time a variable is selected, a small arrow will show up, and the user can click on the arrow to make the name of the variable appear in the command window.
- *Properties window:* where the details about the dataset currently in use (including each variable) are provided.

Figure 2: STATA Main Window



Most of the work in STATA demands the use of commands. To execute a command, STATA offers three possibilities:

I. Typing the command in the command window and then clicking enter. For instance, to estimate the mean of a variable, you can type in the command window:

```
mean varname
```

II. Using the dialog box available in the main menu and choosing the relevant command. For instance, if you want to compute the mean of a variable, you would need to select in the main menu the following:

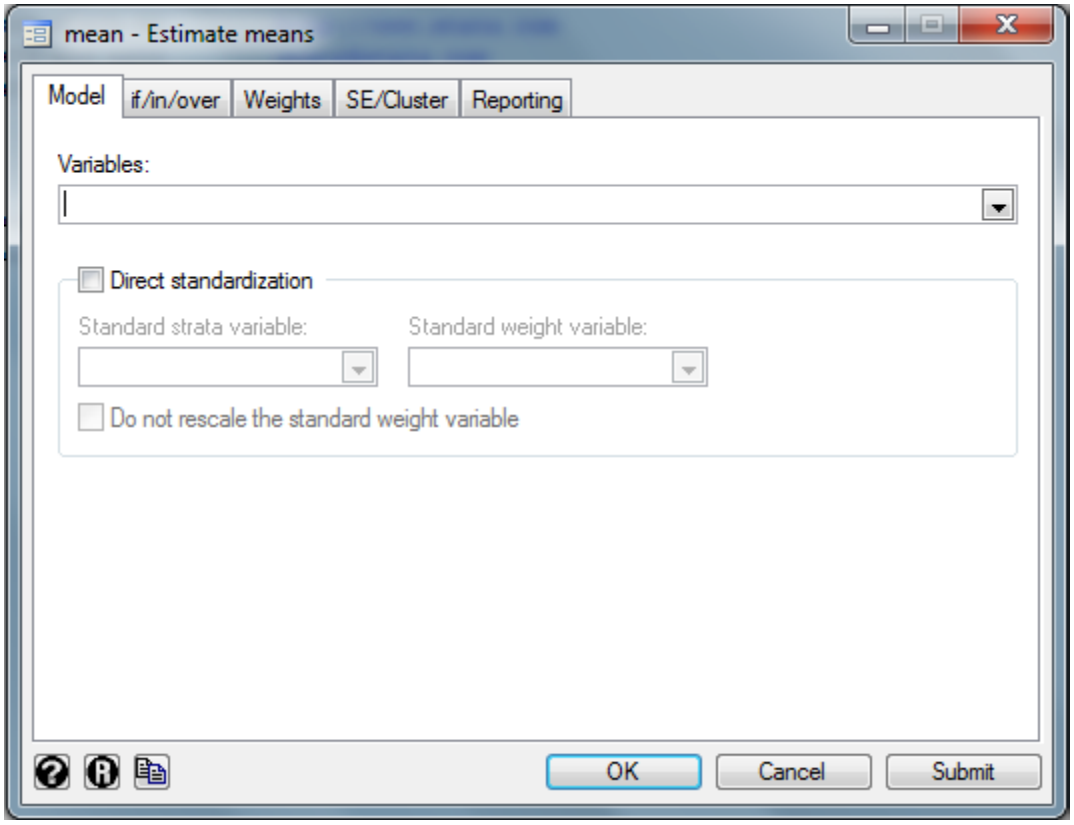
Statistics> Summaries, tables, and tests> Means

Then, a dialog box like the one shown in Figure 3 will appear in your screen. Alternatively, you can type the following line in the command line and you will get the same result:

```
db mean
```

III. Executing a `.do` file, a set of instructions to STATA written in an ASCII file. We will discuss more about `.do` files later, since they would be the recommended way to organize our work.

Figure 3: Example of Dialog Box



The general syntax for STATA commands is the following:

```
[prefix:] command [varlist] [= exp] [if] [in] [weight] [using filename] [, options]
```

The square brackets refer to qualifiers that are usually optional whereas words in typewriter font are to be typed in STATA. Italicized words are to be replaced by users. Check below each element of the general STATA syntax for further explanation:

prefix: It denotes a command that allows for the repetition or modification the input or output of a command.

Example:

```
by region: sum poverty
```

command A STATA command.

Example:

```
tabstat
```

varlist Set of variables over which the command will be applied.

Example:

```
poverty income
```

=exp A mathematical expression.

Example:
`gen incomepc=income/hhsize`

if — It denotes a condition expressed in a mathematical expression.

Example:
`if urban==1`

in — It indicates a range of observations.

Example:
`in 1/100`

weight — It denotes a weighting expression. This is especially relevant for the case of descriptive studies.

Example:
`[pweight=factor]`

filename — Option to indicate the name of a datafile.

Example:
`using (Oportunidades.dta)`

options — denotes one or more options that apply to command.

Example:
`, detail` (after `summarize`)

There are some variations regarding the options available to each command but before discussing these details, let's start with some simple commands to start getting some familiarity with STATA.

3.2. Opening datasets

To start our STATA journey, let's open the simplified version of the ENCEL 2007 dataset. We have called this database `DataFinal_ENCEL07.dta`. In Appendix 1.1 (which we strongly recommend you), we will teach you the details behind the construction of this dataset with the goal of introducing you to more complex data management tasks. Let's keep it simple for now.

There are two common ways to open data sets. In the first case, you just can type the full path and the database extension `.dta`:

```
use "C:\Users\Stanislao\Dropbox\Teaching\1. Current\Econometrics\4.
Handbook\2. Data\ENCEL 2007\Final\DataFinal_ENCEL07.dta", clear
```

The second option is to do it step by step. You can call the working directory (use the `cd` command) and then open the database with `.dta` extension:

```
cd "C:\Users\Stanislao\Dropbox\Teaching\1. Current\Econometrics\4.
Handbook\2. Data\ENCEL 2007\Final"
```

```
use "DataFinal_ENCEL07.dta", clear
```

One common situation is when the database is available in other format than STATA. The command `insheet` allows the user to load into STATA databases in ASCII (text) data format. To illustrate its use, let's consider the file "example.csv". This file contains 13 observations of the original ENCEL dataset. The STATA code is the following:

```
insheet using Example.csv, clear
```

Before continuing, we need to get back to our `DataFinal_ENCEL07.dta`, so let's reload it:

```
use "DataFinal_ENCEL07.dta", clear
```

3.3. Saving datasets

If you want to save changes made in your dataset, you can do it using the command `save`, specifying the `.dta` extension. We save the dataset with a different name:

```
save "Oportunidades.dta", replace
```

It is recommended to use the `replace` option to overwrite the dataset under use.

3.4. Exiting STATA

To leave STATA, you just need to type `exit` in the command window along with the option `clear`. It is important to use this option as STATA will generate an error message in case you have an unsaved dataset still open:

```
exit, clear
```

3.5. Help and search

The help option of STATA provides excellent and useful explanations, syntax and examples of STATA commands. You can call a command using its abbreviated name (underlined in the STATA help command).

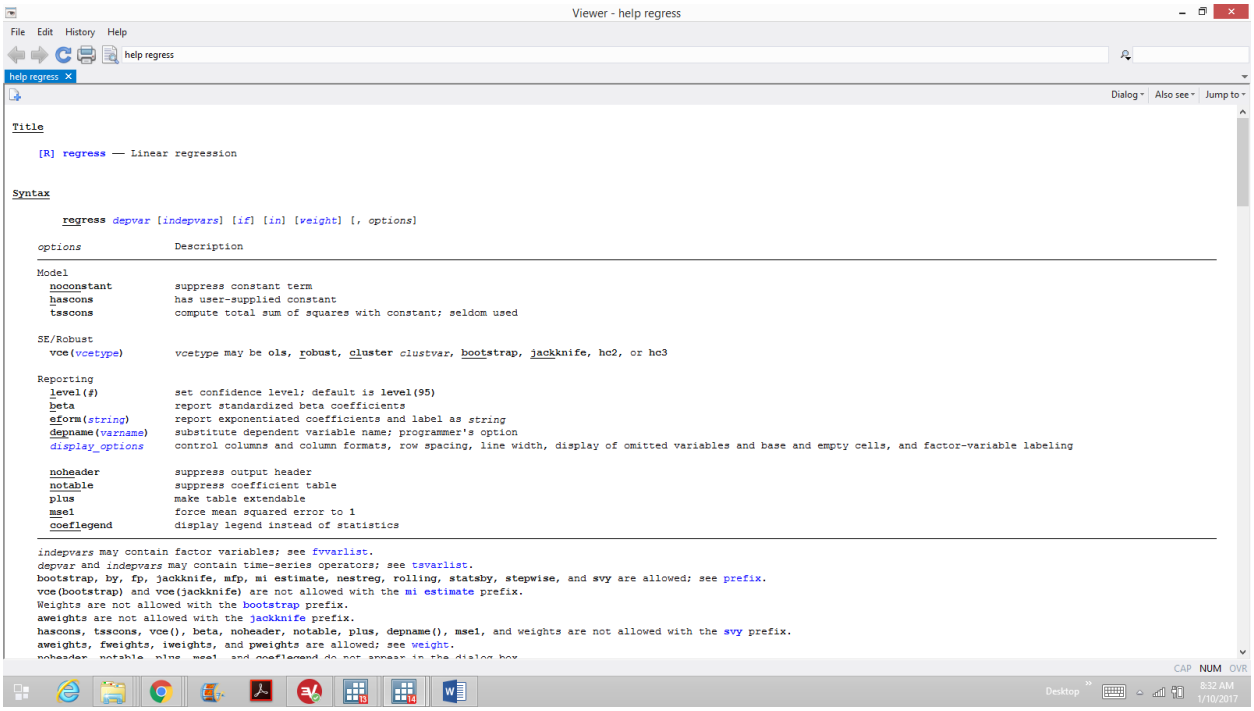
```
help memory

help save

help reg
```

Consider the following 3 examples to illustrate the use of this command. In the first case we ask STATA to provide the help file for the command `memory`. This command is useful for managing the amount of memory allocated to STATA. The second example provides the help file for the command `save` that was already used above whereas the last line requests the help file for the command `regress`. For this last case, you will get a help file like the one on Figure 4.

Figure 4: Help Files in STATA



A useful application to find the name of a command in STATA is via the command `search`. The search, done word for word, is performed over STATA official documents, FAQs, among others. For example, if you do not know which command to use to delete a variable within the database or how to run an ordinary least squares (OLS) regression, search for this information in this way:

```
search eliminate

search ols
```

Consider the case of the search for information regarding OLS. You will find a help file like the one shown in Figure 5.

STATA delivers information regarding OLS regression from different sources, including STATA official commands, help files, STATA journal articles and even online sources. This is usually one of the best ways to search for STATA-related materials.

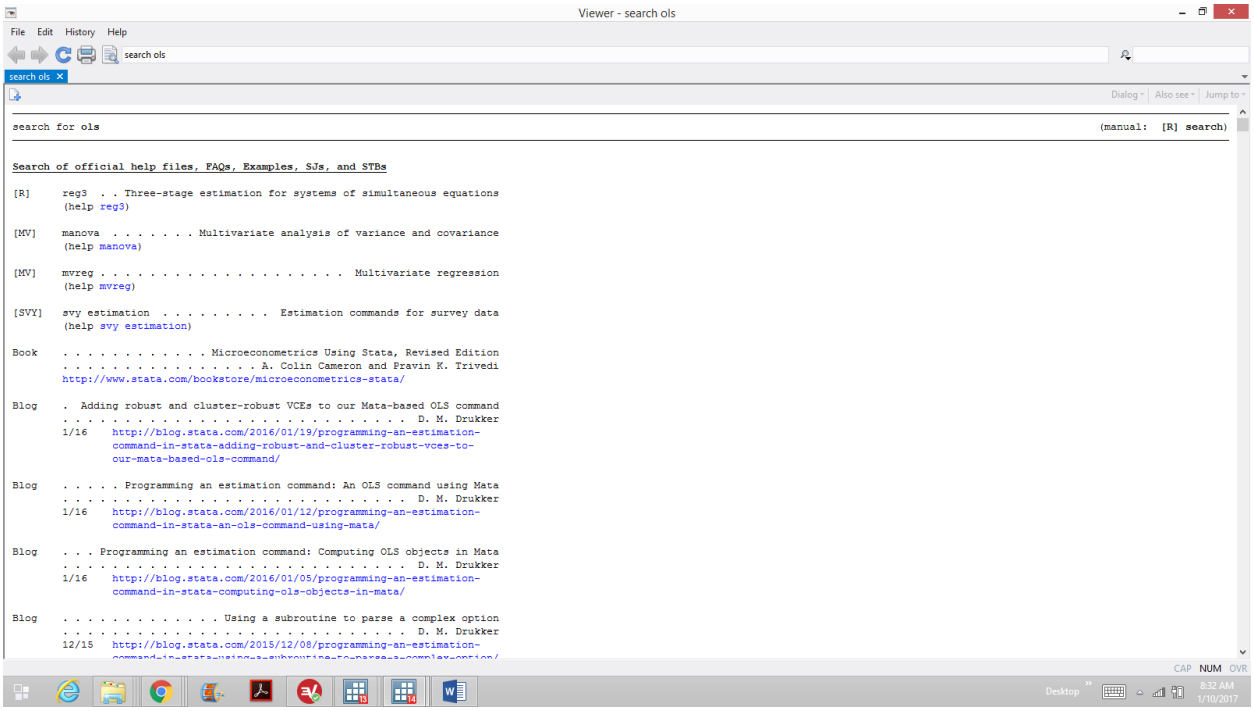
The command `findit` is useful for a broader search and permits to download user-written commands that are not part of the official STATA version. Here a couple of examples:

```
findit xtabond2

findit outreg2
```

In the first case, we asked STATA to find a user-written command `xtabond2`, an extension to the STATA command `xtabond` for dynamic panels. In the second case we request STATA to find the command `outreg2`, a useful command to create edited regression tables.

Figure 5: Searching for Help in STATA



Finally, if we are interested in finding a variable (or variables) in our database, the command `lookfor` searches for the names and labels of the variables within this database. Since we changed the dataset uploaded in STATA to illustrate the use of the `insheet` command, we need to load the dataset `DataFinal_ENCEL07.dta` again to continue with our STATA intro. Now, imagine we want to find a variable related to poverty status or family assets inside the database of Oportunidades. This can be done as follows:

```
lookfor poverty

lookfor asset
```

In the previous example, we ask STATA to look for the words “poverty” and “asset” in the loaded dataset.

4. Data Management: Working with Datasets

An important part of any impact evaluation is the management of a large flux of information. Starting this section, we will be discussing commands that are useful for performing this kind of task.

Let’s upload the dataset again:

```
use "DataFinal_ENCEL07.dta", clear
```

4.1. Describing datasets

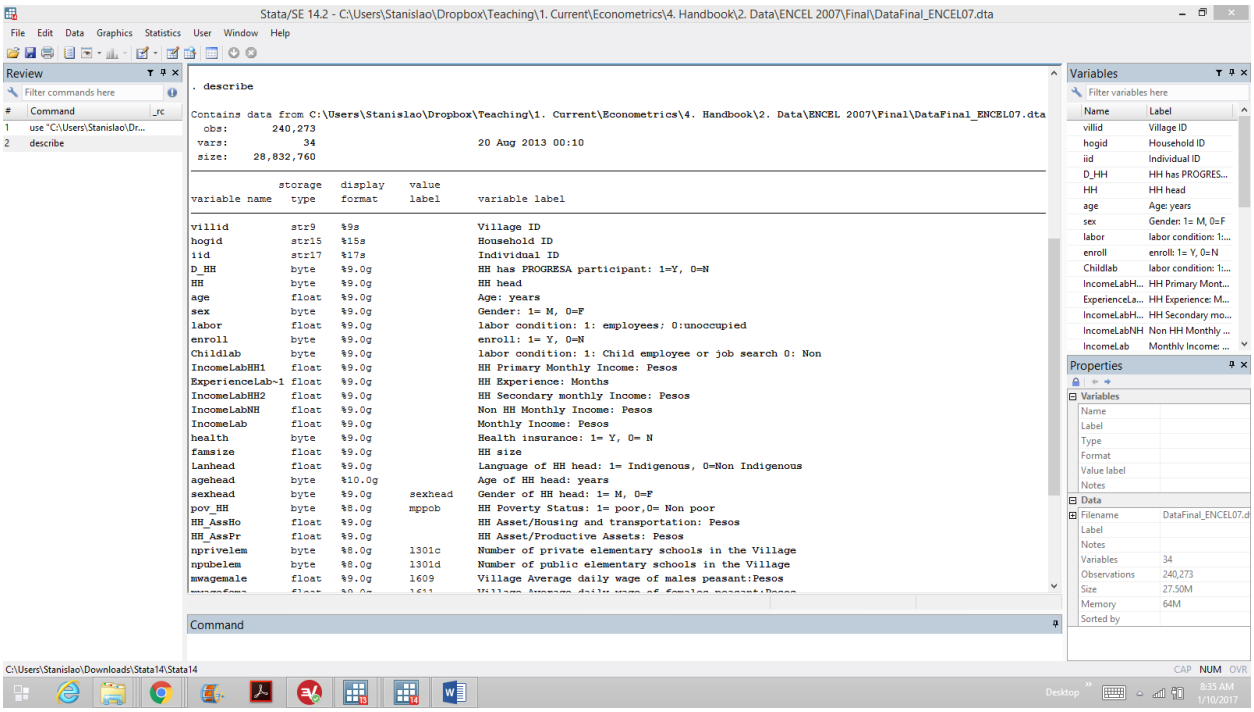
Let’s start with the basics. The first step of any empirical work is to carefully study the information that was made available to you. Therefore, you will need to list all variables in your dataset and see

what kind of information is provided in your database (typically name, size, number of observations). This can be done using the `describe` command.

```
describe
```

You will obtain something like this in the results window:

Figure 6: Describing a dataset



A first look of the STATA output already gives you an important amount of information. The number of observations, variables and size of the dataset are provided at the top of the screen. Then, each variable is listed including information regarding its name, the type, format, and labels. Sometimes the number of variables is very large. In that case, you can shorten the list of variables specifying the variables to be listed as follows:

i. variable by variable,
ii. by typing the first and last of the list, separated by a "-" and,
iii. exploiting some common factor between them (use an asterisk *).

Consider the following example:

```
describe famsize sexhead pov_HH

describe villid - sex

describe p_*
```

The first line restricts the description of the dataset to 3 variables: family size, sex of household head and the household poverty status. The second line describes all the variables between the village id and

the sex of the individual. Finally, the third line considers all the variables whose names start with "p_". In our case, three variables satisfy this condition.

On occasions you may want to see the actual observations rather than the variables. You can list the data into variables using the command `list`. If the command is typed alone it shows all observations for all variables, so that is not recommended. Usually we will just want see data on certain variables. One way to do this is by quoting the number of variables, number of observations to be listed and/or using conditionals.

```
list famsize sexhead in 1/10

list pov_HH if (villid=="01001106" & agehead>30)
```

In the first case, we ask STATA to show the first ten observations for the variables family size (`famsize`) and sex of household head (`sexhead`). You will get something like this:

```
. list famsize sexhead in 1/10
     +--------------------+
     | famsize   sexhead |
     |--------------------|
  1. |       6    Female |
  2. |       6    Female |
  3. |       6      Male |
  4. |       5      Male |
  5. |       6      Male |
     |--------------------|
  6. |       5      Male |
  7. |       6      Male |
  8. |       5      Male |
  9. |       6      Male |
 10. |       6      Male |
     +--------------------+
```

In the second case, we ask STATA to list all the observations by household poverty status in one particular village for those households with household head older than 30 years old.

In the previous step we already introduced some operators available in STATA (== for equal and > for greater than). Some common operators available in STATA are the following:

Relational Operators	Logic Operators
> (greater than)	
< (less than)	**~ (not)**
== (equal)	**| (or)**
> =(greater than or equal to)	**& (and)**
< =(less than or equal to)	
!= or ~= (not equal)	

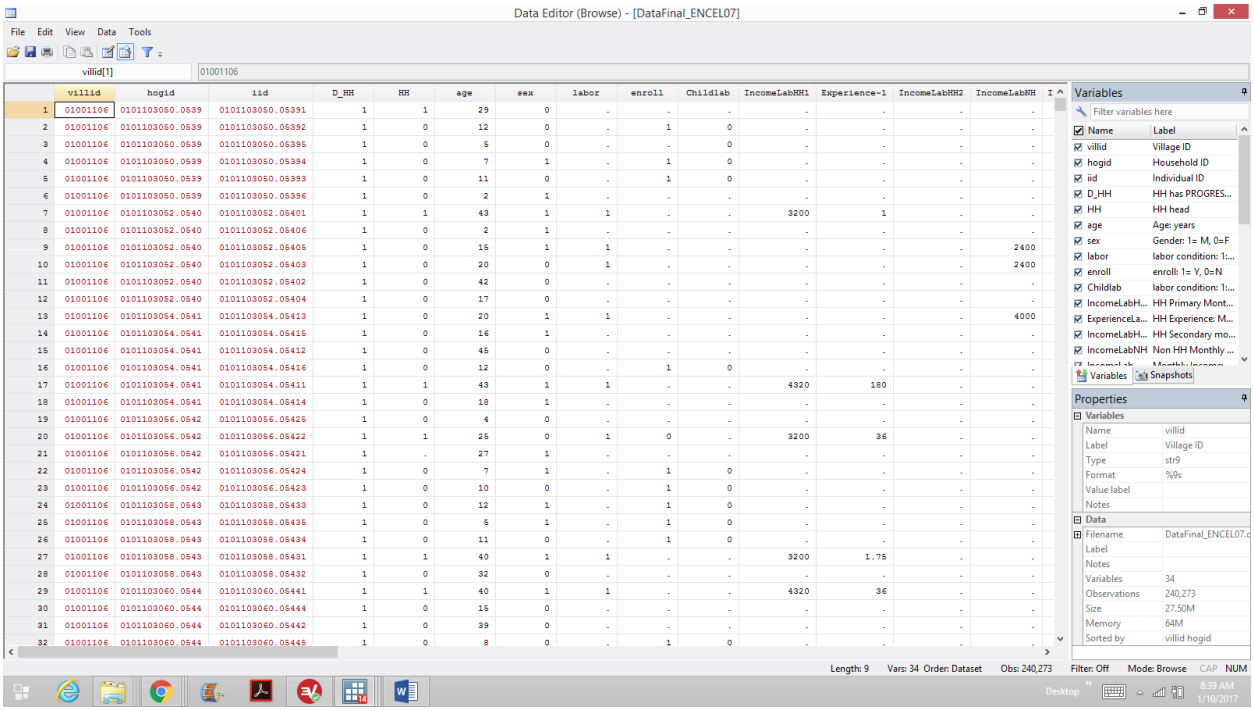
4.2. Sorting and counting observations from a dataset

For many data management operations, you will be asked to sort your data. Before doing that, let's see how our dataset is originally sorted before any change. To complete this task, you have to type in the command window `browse`. Check how the dataset is sorted and identify which variable is used to sort the observations. To finally sort the database based on some variables (or variable) of interest, use the command `sort`.

```
sort villid hogid
```

After this operation, your dataset is sorted according to the village and household id as it is shown in the image bellow:

Figure 7: Sorting a dataset



A useful command to count the observations within a database is `count`. Without any additional criteria, this command delivers the total number of observations in the dataset.

```
count
```

STATA reports that 240,273 observations are available in the database. Although this information is useful, in some cases we want to know how many observations of a given characteristic are available. In that scenario, one would want to control for some argument that restrict the data. For instance, you may want to consider only observation from households in which the household head is older than 35 years. The code will be the following:

```
count if agehead>35
```

In this case, STATA reports that 191,538 observations comply with the requested criteria.

4.3. Summary statistics

So far, attention was given to general operations to characterize the information available in the dataset. We are now in the position of computing basic statistics that would help us in the empirical analysis.

To get basic summary statistics (mean and standard deviation), you can use the command `summarize`, listing the variables to be studied. If we want to analyze other statistics such as the median, percentiles, kurtosis, skewness, among others, we would need to add the `detail` option.

```
summarize famsize sexhead

summarize famsize sexhead, detail
```

In the first case, we ask STATA to provide summary statistics for variables family size (`famsize`) and sex of the household head (`sexhead`). The number of observations, mean, standard deviation and minimum and maximum value are reported.In the second line we ask STATA the same thing but adding the option `detail` to provide more statistics (percentiles, variance, skewness and kurtosis besides the ones mentioned above). In this last case, we will get the following result:

```
. summarize famsize sexhead, detail

                          HH size
-------------------------------------------------------------
      Percentiles      Smallest
 1%            2              1
 5%            3              1
10%            4              1       Obs                 240273
25%            5              1       Sum of Wgt.         240273

50%            7                      Mean           7.383281
                        Largest       Std. Dev.      3.576342
75%            9             53
90%           12             53       Variance       12.79022
95%           14             53       Skewness       1.706424
99%           18             53       Kurtosis       13.43941

                Gender of HH head: 1= M, 0=F
-------------------------------------------------------------
      Percentiles      Smallest
 1%            0              0
 5%            0              0
10%            0              0       Obs                 240271
25%            1              0       Sum of Wgt.         240271

50%            1                      Mean           .8143305
                        Largest       Std. Dev.      .3888406
75%            1              1
90%            1              1       Variance        .151197
95%            1              1       Skewness      -1.616761
```

```
    99%            1               1       Kurtosis       3.613916
```

We learn that, on average, households in the area of influence of Oportunidades have 7.38 members. The smallest household in the sample has 2 members whereas the largest contains 53, a very weird number that should be just a typo given that observations in the 95% percentile contain 14 members. The standard deviation is 3.58, the variance is equal to 12.79 and the kurtosis is 13.44. The gender of household head is a dichotomous variable with a mean of 0.81. In other words, about 81% of households in the sample have a male household head. As well as in the case of the previous variable, computations for the standard deviation, variance and kurtosis are provided.

STATA also incorporates an easy way to obtain frequency distributions of a variable using the command `tabulate`. You can delimitate the variables to study as we have seen before. Additionally, it is possible to construct frequency tables for two variables.

Let's assume that you want to compute a table of frequencies for the `sexhead` variable. You may also want to consider the case in which the analysis is restricted to some sub-group or sub-population. For instance, you may want to restrict the analysis to poor households. The STATA code for both cases is given below:

```
tabulate sexhead

tabulate sexhead if pov_HH==1
```

According to the results, 81.4% of household are headed by a male. This result is similar to the one obtained using the command `summarize`. We obtain similar results (81.7%) if we restrict the analysis to poor households.

We may want to calculate a frequency table with 2 variables. For instance, one may be interested in knowing the percentage of poor individuals who live in households with a male head. One can use the following STATA routine:

```
tabulate sexhead pov_HH, cell
```

For the this case, we get the following:

```
. tabulate sexhead pov_HH, cell

+-----------------+
| Key             |
|-----------------|
|    frequency    |
| cell percentage |
+-----------------+

 Gender of | HH Poverty Status: 1=
  HH head: |   poor,0= Non poor
 1= M, 0=F |  no pobre      pobre |     Total
-----------+----------------------+----------
    Female |     8,225     36,276 |    44,501
```

```
           |      3.43      15.15 |     18.58
-----------+----------------------+----------
      Male |    33,253    161,754 |   195,007
           |     13.88      67.54 |     81.42
-----------+----------------------+----------
     Total |    41,478    198,030 |   239,508
           |     17.32      82.68 |    100.00
```

From this simple example, we learn that 198,030 individuals in the sample live in poor households and that number represents the 82.7% of the total number of observations. Of those, 161,754 individuals live in households with male heads (67.5% of the total) whereas only 36,276 in households with female head (15.2%). A similar comparison can be done for non-poor households or by starting by looking at the gender of the household head instead of poverty status as we did.

Although this command is really useful, sometimes one needs to obtain other types of statistics. In this case, it is recommended to use the command `table`. You can display statistics such as mean, standard deviation, medians, and percentiles, just to mention a few of the options available. More importantly, we can display statistics for two variables within the same table. Let's consider the following examples:

```
table D, c(mean pov_HH sd pov_HH mean agehead sd agehead)

table D, c(mean pov_HH sd pov_HH) format(%9.3f) center

table D pov_HH, c(mean agehead)
```

In the first case, we compute means and standard deviations of two variables -household poverty (`pov_HH`) and household head age (`agehead`)- according to whether a village is participating in the program (`D`). As we see, the means and standard deviations look similar in treated and non-treated villages. In both types of villages, the average level of household poverty is around 80% whereas the household head age is about 50 years. The standard deviations of both variables are also alike between both groups. The STATA output is given bellow:

```
. table D, c(mean pov_HH sd pov_HH mean agehead sd agehead)
-------------------------------------------------------------------------
Village   |
participa |
nt: 1=Y   |
0=N       |  mean(pov_HH)     sd(pov_HH)  mean(agehead)    sd(agehead)
----------+--------------------------------------------------------------
        0 |       .807899       .3940106      49.597431       13.59404
        1 |       .827088       .3781722      48.599201       14.47525
-------------------------------------------------------------------------
```

The second example provides the same results but only for the mean and standard deviation of the household poverty variable. The main difference is that we are telling STATA to restrict the number of

decimals to 3, instead of the original 6 provided by default. The final example considers two variables (whether the village is treated and the household poverty level) and asks STATA to compute the average of household head age for each of the 4 combinations of these 2 variables. The result suggests that the average age of household heads are similar among the 4 groups.

5. Data Management: Changing Datasets

5.1. Creating new variables

Changing or creating new variables is a standard routine in empirical research. In STATA, the command `generate` creates new variables, while the command `replace` helps to modify existing variables. The following routine uses both commands to create a variable called `oldhead` which takes the value of 1 if the age of the household head is more than 35 years and the value of 0 otherwise.

```
generate oldhead=1 if agehead>35

replace oldhead=0 if agehead<=35
```

Similarly, the same variable can be generated using an useful shorthand for dichotomous cases as follows:

```
generate oldhead1=(agehead>35)
```

An extension to the command `generate` is the command `egen` that creates variables but from pre-determined statistics. It is also useful to create variables by groups using the option `by`. To illustrate is use, let's consider two basic examples. The first example creates a variable with the average age of the household head for the whole sample, while the second creates one with an average for each village. Use the command `browse` to see how the results differ in both cases. The STATA code is the following:

```
egen avage= mean(agehead)

egen avage_vil= mean(agehead), by (villid)
```

5.2. Recoding and labeling variables

Another useful command in STATA is `recode` to recode the values within a categorical variable. For example, the variable sex of the household head is encoded so that men take value of 1 and women the value of 0. Suppose now that we want to code women with a value equal to 2 such that we can generate a new dichotomous variable (`sex`) equal to 2 for women and 1 for men. Notice that once this variable is recoded we can add tags to these new values. Another example is the following: imagine that you want to create a new variable (`famscale`) in order to categorize differently (in only 4 categories) the family size variable (`famsize`). Both cases are presented below.

```
recode sexhead (0 = 2 "Women") (1 = 1 "Men"), generate(sex1)

recode famsize (1 = 1) (2/5=2) (6/10=3) (11/100=4), generate(famscale)
```

In order to describe and facilitate understanding of the variables in our dataset, we often label them using the `label` command. The first thing you can label is the database itself. Then, we can label the variable of interest as well as the values within this variable. Let's use the following example:

```
label data "Database of Oportunidades CCT"

label variable oldhead "Family Head over 35 years old"

label define oldlabel 0 "Under_35" 1 "Over_35"

label values oldhead oldlabel
```

The first line in the example above assigns a label to the Oportunidades database. The second one creates a label for the variable `oldlabel` whereas the third one assigns labels to each category of the `oldhead` variable. The last line assigns value labels to this variable.

5.3. Dropping variables and observations

To remove or stop using variables in STATA, use the commands `keep` and `drop` respectively. You can remove or stop using observations in a given variable using conditionals. Consider the following example:

```
drop oldhead1 sex1

drop if famsize>30

keep if famsize<=20
```

In the first case, we drop two variables from our data set. In the second case, all the observations from households with more than 30 members are dropped from the sample whereas in the last one only observations from households with 20 or less members are kept.

5.4. Working with string variables

Finally, a pair of useful commands are `destring` and `trosting`. They convert string variables in numeric variables and vice-versa. They are especially useful when dealing with identifiers in surveys. We discuss how to use of this command in detail in Appendix 1.1. In the meantime, we can use the following examples to illustrate its use:

```
destring villid, generate(village)

tostring famsize, generate(fam_size)
```

In the first line in STATA code above, we convert a string variable into a numeric one under a new variable named `village`. In the second line we do the opposite and create a string variable from a numeric one.

6. Data Management: Combining Data Sets

In most empirical studies is common to work with information of different types, sometimes coming from different levels (district, school, classroom and students) or from different places (provinces or

towns). Therefore, we need to know how to merge these different datasets into a single data file that can be used to the analysis of interest. In this section we discuss how to do so.

Since the dataset has been modified in the previous section, it is worth to reload the original dataset and discard the current modified version:

```
use "DataFinal_ENCEL07.dta", clear
```

6.1. Merging datasets

To work with variables from two related databases based on a unique variable (known as an identifier), STATA has the command `merge` that can de used to join two or more databases in just one. To do so, both databases must be properly ordered based on the variable (or variables) we use as identifiers using the command `sort`. We will always have two different datasets: the main base (**master file**) to which we are going to add variables coming from a secondary base (known as the **using file**). STATA creates one variable `_merge` (created by default) which gives the number of observations that were merged. Let's use a simple example to illustrate the use of this command.

```
sort villid hogid

merge villid hogid using "asset_data.dta"

tab _m
```

The tables below show (in a simplified form) how the command works in our example. In (1) we have the master database `DataFinal.dta` which includes a household id, sex and age of the household head. In (2), we have the database `asset_data.dta` which includes the household id and a dummy equal to one if the household has at least one asset. The command `merge` adds (2) to (1) using the variable `hogid` as identifier in the two bases. In (3) show the results after merging the databases and the variable `_merge`:

In memory master (1)			+	In filename.dta using (2)		=	Merged result (3)				
hogid	**sex**	**age**		**hogid**	**asset**		**hogid**	**sex**	**age**	**asset**	**_m**
101103058	hombre	40		101103058	Without		101103058	hombre	40	Without	3
101103050	mujer	29		101103050	Without		101103050	mujer	29	Without	3
101103052	hombre	43		101103052	Without		101103052	hombre	43	Without	3
101103060	hombre	40		101103060	Without		101103060	hombre	40	Without	3
101103062	hombre	39		101103062	Without		101103062	hombre	39	Without	3
101103064	mujer	45		101103064	Without		101103064	mujer	45	Without	3
101103072	hombre	24		101103072	Without		101103072	hombre	24	Without	3
101103068	hombre	22		101103068	Without		101103068	hombre	22	Without	3
101103070	mujer	40		101103070	Without		101103070	mujer	40	Without	3
101103054	hombre	43		101103054	Without		101103054	hombre	43	Without	3

6.2. Appending datasets

On the other hand, if you want to add two databases with the same variables (or nearly the same variables) but with different observations, you can use the command `append`. For example, if we have a database that has information for 698 villas and we want to add another database with information collected for families of 2 villages (which were mistakenly not included), you can use the following routine.

```
append using "additional.dta"
```

6.3. Creating datasets with aggregate information

Imagine you have a database with observations defined at individual level that you want to merge with one with information defined at village level. If you want to collapse the individual database to create a new dataset with aggregate averages (or some other statistic) that you can later merge to another dataset, you will need to use the `collapse` command. For example, in the database we have information for families within villages. Suppose now that we are interested in village level information, so we need to aggregate the variables to that level. Notice that the mean is not always of interest, so the `collapse` command includes other relevant statistics such as "sum", "max", among others. We proceed using the following routine:

```
collapse (mean) famsize agehead (sum) asset, by(villid)
```

In the picture bellow we see a simplified version of this exercise, taking only 3 villages and three variables. In (1) we have household level data and as a result, in (2) we obtain the average at village level. Notice that we need specify the "mean" to designate that we want to compute averages at village level.

Individual level (1)

hogid	village	famsize	sex	agehead
101103050.1	1001106	6	mujer	29
101103052.1	1001106	6	hombre	43
.....	.....	.....	.....	....
101103066.1	1001106	6	hombre	42
101103068.1	1001106	3	hombre	22
101103070.1	1001106	6	mujer	40
101103072.1	1001106	5	hombre	24
101103002.1	1001315	6	hombre	29
.....	.....	.....	.....	....
101103018.1	1001315	4	hombre	25
101103024.1	1001315	4	mujer	36

102103002.1	1002008	4	hombre	29
102103004.1	1002008	6	mujer	45
.....	.....	.....	.....	....
102103018.1	1002008	9	hombre	47
102103020.1	1002008	5	hombre	41
102103022.1	1002008	6	hombre	40
102103024.1	1002008	6	hombre	38

Village level (2)

village	famsize	sex	agehead
1001106	5.3	0.67	36.0
1001315	5.7	0.67	39.6
1002008	5.7	0.92	40.8

7. Data Management: Using .log and .do files

When empirical work is complex, it is better to organize your work in a way that you can ensure replicability. STATA makes that simple by using a text with a set of instructions known as do-files.

7.1. Using .do files

One way to create a do-file is via the Do-file Editor. In STATA, we can open a do-file selecting **Window> Do-file Editor> New Do-file** in the toolbar. Then, we just need to type the instructions and finally save them into the do-file.

A do-file is usually run from the Do-file Editor. Select **Window> Do-file Editor> New Do-file**, and then select **File> Open**. An advantage of the Do-file Editor is that you can select only part of the do-file by selecting **Execute Selection Quietly** (Ctrl + D) in the window of the Do-file Editor. A typical do-file looks like Figure 8.

If you paid attention to our previous steps, you will notice that this do-file includes the steps we have been discussing so far. This is a great thing and something you will appreciate during this course.

7.2. Using .log files

By default, STATA outputs are displayed in the screen. However, it is advisable to store the results in another separate file, so that these can then be revised without seeing the Do-file Editor. The STATA output file is called log file, which is saved with the extension `.log`. The STATA command to open or to

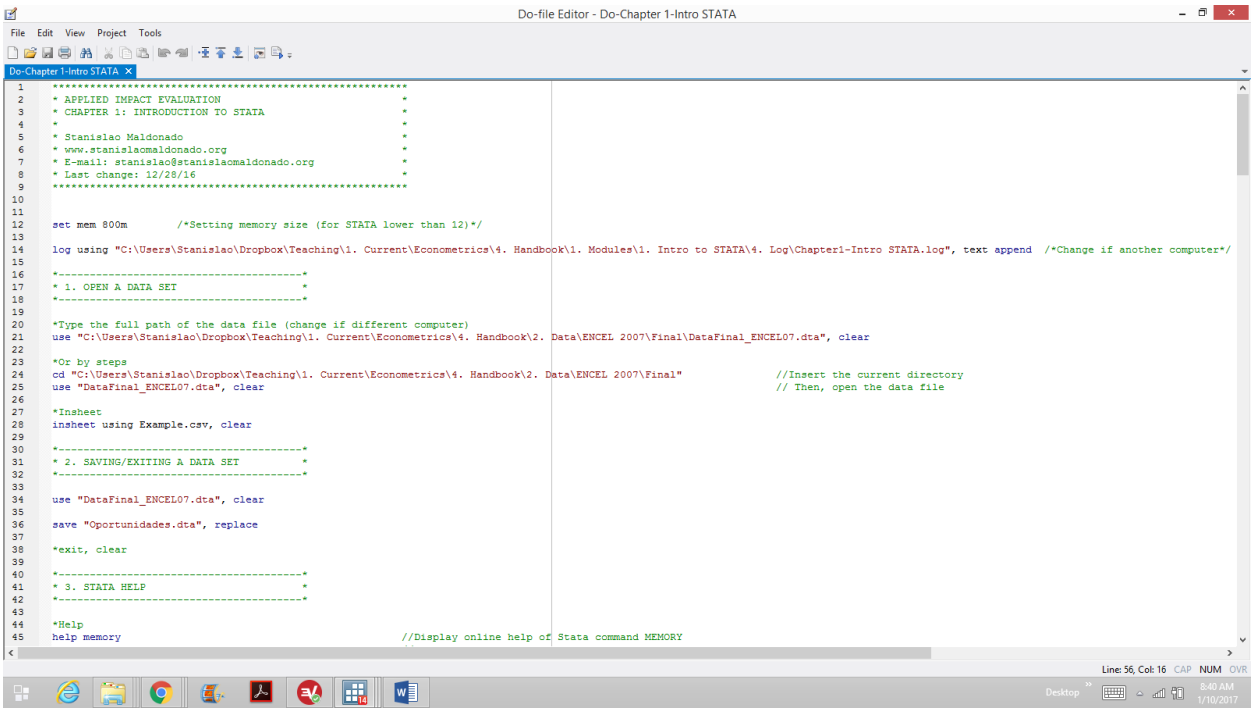
create a log file is `log`. Once all the information you want to be saved is complete, you can finish your work by typing `log close` (or, if you are using a do-file, by including this text at the end of the section of your do-file you want to save). A useful option is `replace text` which allows overwriting information in the existing log file.

```
log using example, text replace

*[Stata commands]

log close
```

Figure 8: Do-file



In STATA will usually do the following 3 steps:

1. Create or change a do-file
2. Running a STATA do-file
3. Read the results of the log file with a text editor.

An example of a `.do` and `.log` files can be found online in the virtual course platform. It contains all the steps followed in this chapter.

8. Graphs in STATA

It is advisable to study your variables before you plot them. STATA offers a lot of interesting options for graphs, so you can produce figures, charts or tables of very high quality. There are some nice examples of cool graphs in STATA you can find online. Here, we only introduce the basics. We will cover more graphical tools over the course.

Since our original dataset have been modified due to the previous steps, it is good to upload it again.

8.1. Histograms

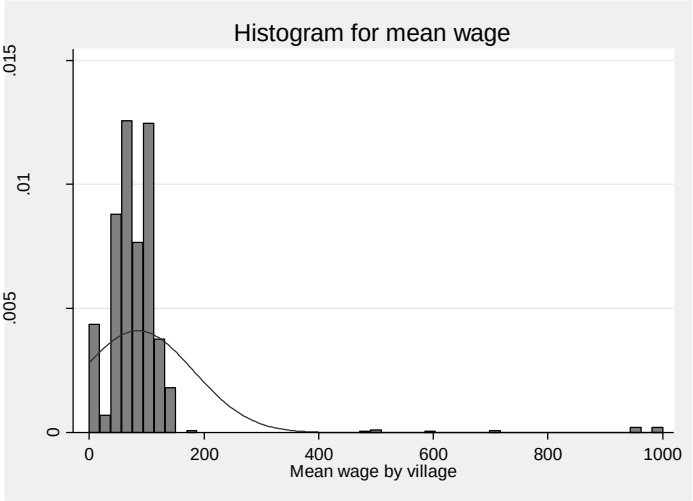
A histogram is a graphical representation (in bars) of a variable, wherein the surface of each bar is proportional to the frequency of the values represented. The histogram is used to obtain a "first glance" of the overall data distribution. The basic STATA command for this is `histogram`. Its main options are `width (#)` to adjust the width of the bar and `bin (#)` to adjust the number of bars. The default number of bins is $min(\sqrt{N}, \frac{10lnN}{ln10})$. An interesting option is to overlay a graph of a normal density to see whether it shows a good fit with the empirical data. Let's use the following example:

```
histogram mwagemale

histogram mwagemale, bin(10)

histogram mwagemale, normal xtitle(Mean wage by village) title(Histogram for
mean wage) scheme(sj)
```

Using our Oportunidades dataset, we can study the average village wage using histograms. The first line in the previous box produces a very simple histogram of the average village wage. The second line uses a number of bins equal to 10 instead of the default option. Finally, the third line add options to the basic graph to add features like a title, a label for the variable and the inclusion of a normal curve. STATA offers a lot of options for making nicer graphs. We refer the interested reader to Mitchell (2012). In the graph bellow, we provide an histogram with a title and a label for the wage variable as well as a normal curve.

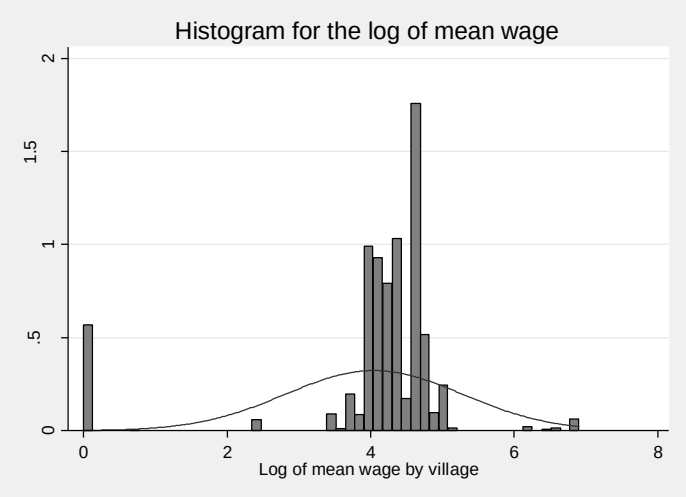


As seen in the histogram, the distribution is asymmetric with enough mass in the left side, as would be expected of any income variable. A useful option is to use the logarithm of wages which shows a better fit using the command `generate` in the following way:

```
gen logwage=log(1+mwagemale)
```

Notice that we computed the log of one plus the mean wage. We do this since those individuals with zero income would be get a missing value if we compute the logarithm instead. We can draw the histogram for the logarithm of the mean wage now:

```
histogram logwage, normal xtitle(Log of mean wage by village) title(Histogram
for the log of mean wage) scheme(sj)
```



8.2. Kernel densities

An alternative and very popular way of describing the distribution of wages across villages is via the use of kernel densities. The main advantage over the histogram is that rather than giving equal weight to each bar, the kernel gives more weight to data that are close to the point of reference. The estimated kernel density $f(x)$ evaluated at $x = x_0$ is

$$f(x_0) = \frac{1}{Nh}\sum_{i=1}^{N} K\left(\frac{x - x_0}{h}\right)$$

Where $f(x)$ denotes the density, $K(.)$ is the kernel function that gives more weight to point x_i near to point x_0. The function $K(z)$ fulfills the condition of being symmetrical around zero, integrates to one and $K(z) = 0$ if $|z| > z_0$ or $z \to 0$ as $z \to \infty$. Since we know that we need to choose a kernel function and a bandwidth (h) to evaluate $f(x_0)$ to be plotted against x_0, we can define it depending on our application. Otherwise STATA would choose the Epanechnikov kernel function by default. This Kernel is written as:

$$K(z)\begin{cases}\frac{3}{4}\frac{(1 - z^2/5)}{\sqrt{5}} & \text{si } |z| < \sqrt{5} \\ 0 & \text{otherwise}\end{cases}$$

Other choices are available for the kernel function (such as a uniform, triangular, Gaussian, among others). The default bandwidth is $h = 0.9m/n^{1/5}$,, where $m = \min(s_x, iqr_x/1.349)$ and iqr_x is inter-quantile range. The `bwidth` option (#) enables different values for the bandwidth (h).

```
kdensity mwagemale

kdensity mwagemale, kernel(gaussian)
```

We plot the kernel density using the same example of average village-level salary. The first line draws a very simple kernel density of average wages with an Epanechnikov function and the optimal bandwidth. The second line uses a Gaussian kernel instead. As you may notice, the choice of kernel does not play an important role.

We also test different bandwidths (h). In the first line, we choose h=0.5 and in the second line h=10. You can observe that when h is very large, the kernel density tends to be very soft, while if h is very small tends to be fit better the data. This illustrates the trade-off between bias and variance which is implied in choosing the bandwidth. As in the histogram case, the density tends to concentrate enough mass points in the left side.

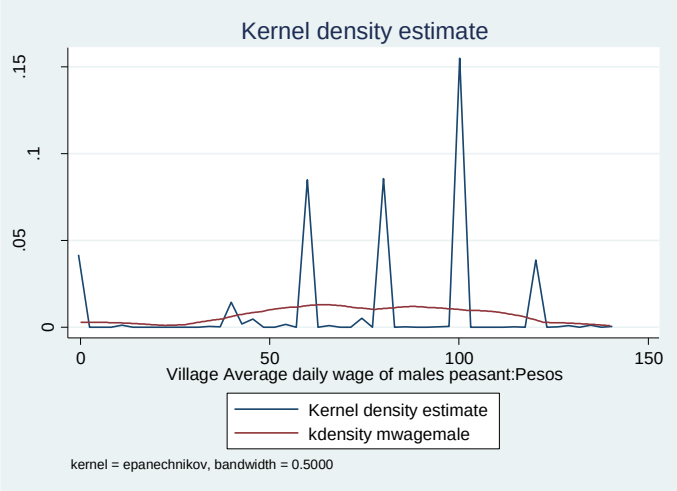
```
kdensity mwagemale, bwidth(0.5)

kdensity mwagemale, bwidth(10)
```

We have drawn two different kernel densities in the previous step. Sometimes, researchers want to have both densities in the same graph. In this case, the option `addplot` can be used. Below an example:

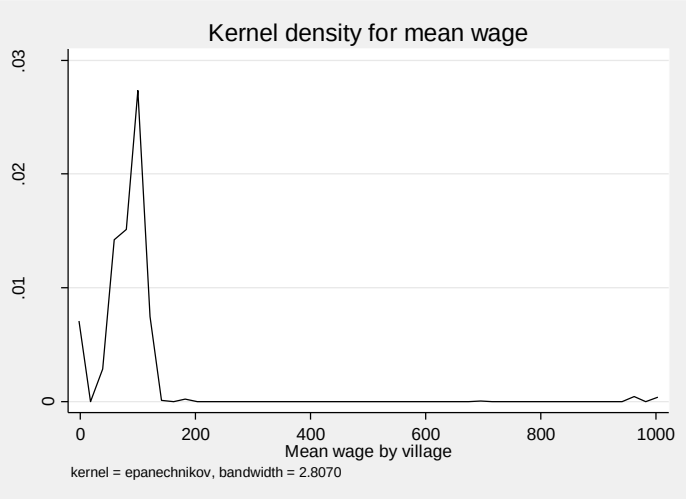
```
kdensity mwagemale if mwagemale<150, bwidth(0.5) addplot(kdensity mwagemale
if mwagemale<150, bwidth(10))
```

To facilitate the comparison, we restrict the analysis to cases of average wages lower than 150 pesos.



As in the previous case, we can exploit the STATA formatting options to draw nicer graphs.

```
kdensity mwagemale, xtitle(Mean wage by village) title(Kernel density for
mean wage) scheme(sj)
```

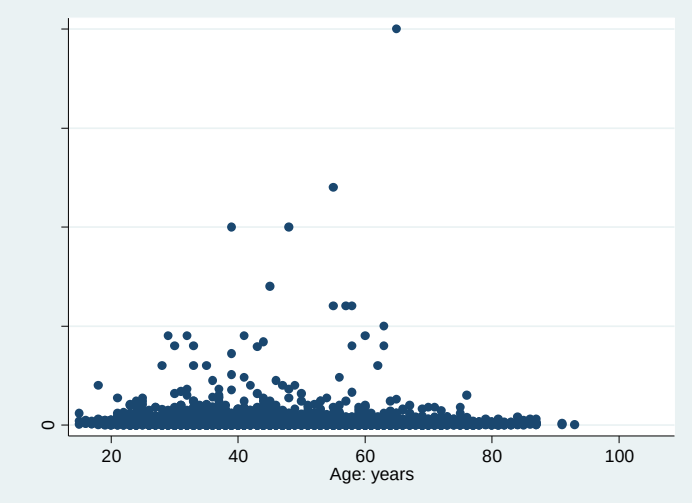


8.3. Scatter plots

A very common and useful task is to plot correlations among variables to see the level of association among them. In STATA this can be done using the command `scatter` for two variables (`twoway`). Let's, for instance, plot the relationship between monthly income and household head age. We can use the following routine.

```
twoway (scatter IncomeLab age if HH==1)
```

We restrict the sample only to household heads. The graph does not show a clear relationship between these two variables as you may have expected, but we are interested only in teaching you how to use STATA for this task rather than exploring the information available to us.

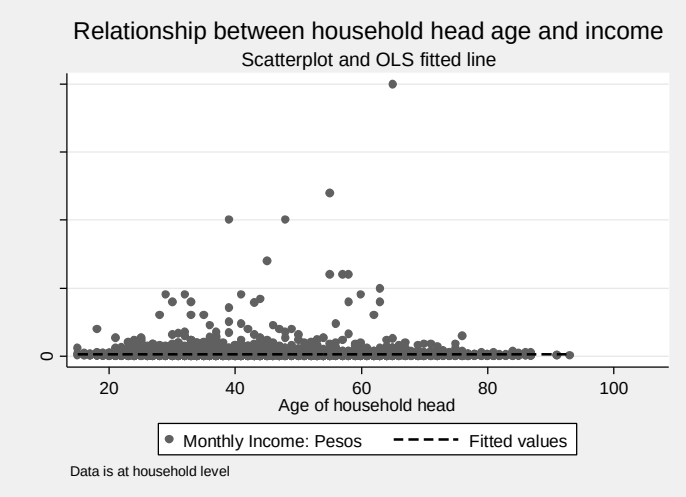


We can also add titles and other features to our graph.

```
twoway (scatter IncomeLab age if HH==1), ytitle(Monthly income)
xtitle(Household head age)
```

We can also fit a line that minimizes the distance between the points in the previous graph in order to evaluate whether the relationship between incomes and age is linear. We use the following code:

```
twoway (scatter IncomeLab age if HH==1) (lfit IncomeLab age if HH==1,
lwidth(medthick)), ytitle(Monthly income) ///
      xtitle(Age of household head) title(Relationship between household head
age and income) subtitle(Scatterplot and OLS fitted line) ///
      note(Data is at household level) scheme(sj)
```



The results do not show a clear pattern. We have added some extra features like subtitle and a note. We also use the option `lwidth(medthick)` to specify the shape of the line for fitted values. The interested reader can review Mitchell (2012) for an explanation of the options available for graphs in STATA.

9. Data Management: Macros and Looping Commands

9.1. Global and locals

A macro is a string of characters that represents another string of characters such that makes it shorter, easier to read and easy to adapt to similar problems. A macro can be `local` or `global`. A global macro is accessible through a do-file or during a STATA session. A local macro is only accessible within a given do-file or during an interactive session.

To define a global macro we use the command `global`. To access to what was stored in the global macro, we type the `$` symbol immediately before the name of the macro. For example, suppose you keep a list of variables into a global macro which we will call `glo_list`. Then, we can use it in various applications with other command such as `describe`, `summarize`, even with other variables. Here some examples:

```
global glo_list famsize pov_HH

describe $glo_list

summarize mwagemale $glo_list
```

The first line of the example above creates a global with a list of two variables (family size and household poverty level). We then ask STATA to describe the variables in the global. The final line asks STATA to summarize the information of a list of variables included in the global plus the mean of male wage (`mwagemale`). An advantage of the global macro is that we can use the same macro several times because it is preserved throughout a do-file.

A local macro is defined with the command `local`. To access to what was stored in the local macro, we type the macro name with single quotes. Notice that a local only will work if we call it at the same time with the other instructions, otherwise it won't. The latter problem does not happen using the `global` command.

```
local lo_list mwagemale agehead

twoway (scatter `lo_list')
```

In the example above, we create a local variable in the same way we did the global before. Then, we ask STATA to create a scatter plot between the average male wage and the age of household head.

9.2. Loops

Loops provide a way to repeat the same command many times. STATA has three loops built in: `foreach`, `forvalues` and `while`. The command `foreach` constructs a loop over the elements of a list, where the list can be a list of variable names (possibly given in a macro) or a list of numbers. The command `forvalues` builds a loop over consecutive values of numbers. A `while` loop continues until a user-specified condition is not met.

For example, suppose we want to compute multiple correlations between wages and a list of variables but one by one so that we know the size of the correlation of each case. Using the `foreach` command we can create a list of local variables to call `xvar` against which we compute (one by one) the correlation with the wage variable.

```
foreach varx of varlist sex pov_HH agehead {

display "A loop to compute correlation beetween average wage and " "`varx'"

correlate mwagemale `varx'

}
```

Similarly, suppose now you want to compare the means for certain variables (poverty, family size and average wages) for treated and untreated villages. As we know that the treatment status is defined in the variable `D`, which takes values of 1 and 0 for treated and non-treated villages respectively. We can use the command `forvalues` to include this variable as a conditional as follows:

```
forvalues i=0/1  {

summarize pov_HH mwagemale famsize if D==`i'

}
```

This same exercise can be done but using the `while` command, in which we first define a local variable `i = 1` and repeat the command (for instance, using the command `summarize`) until the condition accompanying the command `while` is no longer satisfied. Note that, when we define a local for each iteration, the value of `i` is increased by one.

```
local i 0

while `i'<=1  {

summarize pov_HH mwagemale famsize if D==`i'

local i = `i' + 1

}
```

10. Basic Regression Analysis

A first approach to empirical data analysis is linear regression. The goal of a linear regression is to estimate the parameters of a linear conditional mean:

$$E(y|x) = X'\beta = \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k$$

Where y is a dependent variable to be explained, x_k is a column of n entries for the k regressor, β_k are k regressor parameters, which are interpreted as marginal effects.

$$\frac{\partial E(y|x)}{\partial x_k} = \beta_k$$

It is possible in STATA to run this regression and obtain these marginal effects, as well as a variety of other results. The command to do regressions in STATA is `regress`. To illustrate the use of this command, we use poverty as the dependent variable and run it against a list of regressors. We start by using a very simple specification using village treatment status `D` as an independent variable as follows:

```
regress pov_HH D
```

The output is the following:

```
. regress pov_HH D

      Source |       SS       df       MS              Number of obs =  239508
-------------+------------------------------           F(  1,239506) =    8.47
       Model |  1.21331974     1  1.21331974           Prob > F      =  0.0036
    Residual |  34293.6259239 506   .14318483          R-squared     =  0.0000
-------------+------------------------------           Adj R-squared =  0.0000
       Total |  34294.8392239 507  .143189298          Root MSE      =   .3784

------------------------------------------------------------------------------
      pov_HH |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
           D |   .0191883   .0065917     2.91   0.004     .0062687    .0321078
       _cons |   .8078995   .0065455   123.43   0.000     .7950704    .8207285
------------------------------------------------------------------------------
```

10.1. Interpreting STATA regression output

Since we are going to be running regressions all the time, it is worth to carefully analyze the regression output. In general, a regression output contains three basic elements:

a) ANOVA table

The **analysis of variance (ANOVA)** is a very popular technique to analyze the sources of variability of a particular variable and decompose the contribution of each factor in explaining this variability. Specifically, the total variability of a variable is decomposed in the variability due to the model (the set of explanatory or independent variables) and the residual (all the other factors not accounted in the model that also explain de dependent variable). Recall the definition of the variance:

$$\text{var}(y) = \frac{1}{n-1}\sum(y_i - \bar{y})^2;$$

where the expression $\sum(y_i - \bar{y})^2$ is known as the **sum of squares (SS)** and $n-1$ is the **degree of freedom (df)**. The ANOVA analysis is based on the partitioning of the total SS into the model and the residual components. The same logic applies to the df. The ratio of the SS and the df for each component is known as the **mean square (MS)**. STATA provides an ANOVA table by default after `regress`.

```
      Source |       SS       df       MS
-------------+------------------------------
       Model |  1.21331974     1   1.21331974
    Residual |  34293.6259239 506   .14318483
-------------+------------------------------
       Total |  34294.8392239 507  .143189298
```

In the previous example, the model is composed of a single variable (`D`). The SS associated to the model is `1.21331974`. Since df is 1, the MS has the same value. Notice that the degree of freedom is always the number of coefficients (including the intercept) minus 1. Since in this example we have two coefficients, the df is equal to 1. The same logic applies to the case of the residual. The results suggest that treatment status explains very little of the observed variability in household poverty status.

b) Overall model fit

The regression output includes a set of measures of the overall fit of the econometric model and related information. STATA report the number of observations used in the regression (239,508 in our case), the **F-statistic**, the **coefficient of determination** (or **R-squared**) and the **root of the mean square error (RMSE)**.

The **F-statistic** is a test statistic with an F-distribution under the null hypothesis. We will discuss hypothesis testing in the next section in detail.

The **coefficient of determination** is a measure of how well the observations fit the model or, in other words, the fraction of the variance in the dependent variable that can be explained by the independent variables. The following is its basic formula:

$$R^2 = 1 - \frac{SS_{\text{model}}}{SS_{\text{total}}}$$

STATA also offers an adjusted version of the R-squared. This version penalizes the inclusion of covariates into the model. The formula is the following:

$$AdjR^2 = 1 - \frac{1 - R^2 / n - 1}{n - k - 1};$$

where k is the number of predictors. In this particular example, both R-squared and its adjusted version suggest that the model explains very little –if anything- of the variability of poverty status (0.0000).

Finally, the RMSE is defined as follows:

$$RMSE = \sqrt{\frac{\sum (y_i - y_i)^2}{n}}.$$

The RMSE is the standard deviation of the error term. It is a measure of the fit of a model. The number itself does not say that much unless is compared to an alternative model. In this particular example, its value is `.3784`.

```
Number of obs =  239508
F(  1,239506) =    8.47
Prob > F      =  0.0036
R-squared     =  0.0000
Adj R-squared =  0.0000
Root MSE      =   .3784
```

c) Parameter estimates

The final element in the STATA regression output presents the parameter estimates. It includes the regression coefficients, the standard errors, the t-statistic, the p-value and the 95% confidence interval.

The **regression coefficients** just represent the values of the regression equation for predicting the dependent variables from the independent variables. In this case, the coefficient associated to the village treatment status is `.0191883`. Since this explanatory variable is dichotomous, this coefficient is interpreted as how less likely of being poor households are in treated villages with respect to non-treated villages. We will discuss in the next chapter how to interpret regression coefficients for a variety of cases.

The second column presents the **standard errors** for each coefficient. As in the case of the RMSE, the standard error is a measure of how precise an estimator is. In this example, the standard error for D is equal to `.0065917`.

The third column presents the **t-statistic** for the test of whether a coefficient is statistically different from zero and the fourth column reports the associated **p-values** used in testing the null hypothesis that the coefficient is equal to zero. We know from basic statistics that a t-statistic of 1.96 is

associated to a p-value of 0.05. In our particular case, the t-statistic is `2.91` and it is significance at the usual confidence level of 95%. Accordingly, the p-value is `0.004`.

The last two columns report the **confidence interval** considering at 95% level. Notice that zero is not included in the interval, suggesting that the coefficient is statistically significant.

```
------------------------------------------------------------------------------
      pov_HH |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
           D |   .0191883   .0065917     2.91   0.004     .0062687    .0321078
       _cons |   .8078995   .0065455   123.43   0.000     .7950704    .8207285
------------------------------------------------------------------------------
```

10.2. Tables for regression output

After each regression it is possible to save the main results using the command `estimate store` and then assigning a name to the saved data for each regression. We then can recover the saved results in a single table. This is extremely useful in terms of comparing several econometric specifications for the same outcome of interest.

In the example below we run four regressions having poverty status as a dependent variable and whether the village is treated by Oportunidades as independent variable. In the first three regressions we add a new independent variable in each, while in the fourth one we compute robust standard errors using the option `vce(robust)`. The STATA code would be the following:

```
regress pov_HH D

estimates store reg_1

regress pov_HH D mwagemale

estimates store reg_2

regress pov_HH D mwagemale agehead

estimates store reg_3

regress pov_HH D mwagemale agehead, vce (robust)

estimates store reg_4
```

STATA provides the command `estimates table` that allows in a simple way to add the saved regression results and display them together. Among the options for this command we have the possibility of reporting some statistics such as R2, F-statistic, number of observations, just to mention a few.

```
estimates table reg_1 reg_2 reg_3 reg_4, b(%9.4f) se stats(N r2 F)
```

The STATA output is the following:

```
. estimates table reg_1 reg_2 reg_3 reg_4, b(%9.4f) se stats(N r2 F)

----------------------------------------------------------------
```

```
    Variable |   reg_1       reg_2       reg_3       reg_4
-------------+----------------------------------------------
           D |    0.0192      0.0276      0.0275      0.0275
             |    0.0066      0.0067      0.0067      0.0071
  mwagemale |               -0.0001     -0.0001     -0.0001
             |               0.0000      0.0000      0.0000
    agehead |                          -0.0004     -0.0004
             |                           0.0001      0.0000
       _cons |    0.8079      0.8131      0.8346      0.8346
             |    0.0065      0.0067      0.0072      0.0075
-------------+----------------------------------------------
           N |    239508      236242      236030      236030
          r2 |    0.0000      0.0013      0.0016      0.0016
           F |    8.4738    155.2522    126.0833    145.1623
-------------------------------------------------------------
                                                 legend: b/se
```

We won't expend time discussing the details behind the interpretation of these results. We will go over these issues in the next chapter. For the sake of completeness, it is worth to mention that we asked STATA to include the number of observations, R2 and F-test in the table. We also asked STATA to present the estimates of the standard errors and to restrict the number of decimals to 4.

Sometimes, copying the results to a spreadsheet for posterior edition can be difficult. Fortunately, STATA users have written several commands to automatize this process. A useful command to save the results in an excel file is the command `xml_tab`. This command is not part of the basic STATA package and therefore must be installed (using `findit` command). As in the previous case, this command allows a number of options that includes different statistics, titles, notes, and much more. Here just a basic examples. The interested reader should consult Lokshin and Sajaia (2008) for more details.

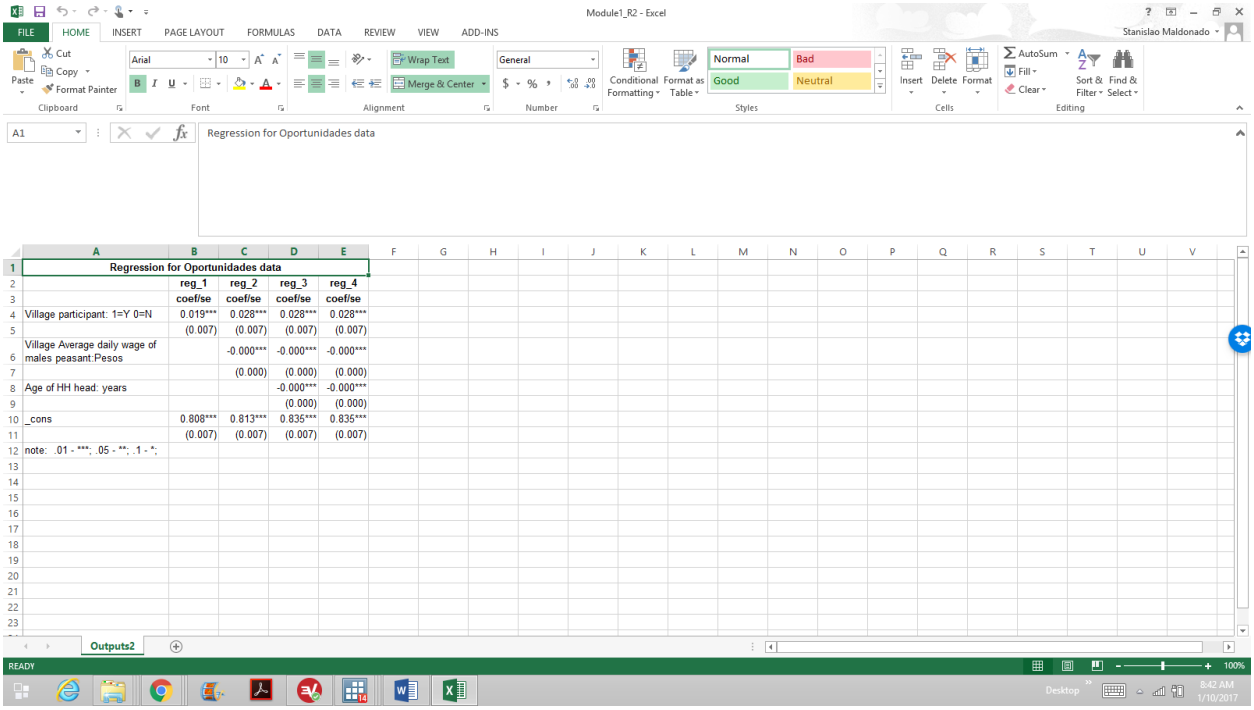
```
findit xml_tab

xml_tab reg_1 reg_2, save ("C:\Users\Stanislao\Dropbox\Teaching\1.
Current\Econometrics\4. Handbook\2. Data\ENCEL
2007\Final\Modulo1_R1.xml")replace title ("Regression for Oportunidades
data") sd below sheet("Outputs1")

xml_tab reg_1 reg_2 reg_3 reg_4, save("C:\Users\Stanislao\Dropbox\Teaching\1.
Current\Econometrics\4. Handbook\2. Data\ENCEL
2007\Final\Module1_R2.xml")replace title ("Regression for Oportunidades
data") sd below sheet("Outputs2")
```

The excel file with the results looks like for the second example:

Figure 9: Excel File Results



Regression for Oportunidades data				
	reg_1	reg_2	reg_3	reg_4
	coef/se	coef/se	coef/se	coef/se
Village participant: 1=Y 0=N	0.019***	0.028***	0.028***	0.028***
	(0.007)	(0.007)	(0.007)	(0.007)
Village Average daily wage of males peasant:Pesos		-0.000***	-0.000***	-0.000***
		(0.000)	(0.000)	(0.000)
Age of HH head: years			-0.000***	-0.000***
			(0.000)	(0.000)
_cons	0.808***	0.813***	0.835***	0.835***
	(0.007)	(0.007)	(0.007)	(0.007)
note: .01 - ***; .05 - **; .1 - *;				

Notice that the results are similar to those reported above with the advantage that now it is possible to edit the results is a faster manner. Notice that they differ in the number of decimals.

10.3. Post-estimation options

The command `regress` provides additional results that are not listed in the screen of the regression results. One way to see and access them is by using the command `ereturn list`. One can also save them as scalars, locals or matrices so you can work with them later. For example, let's save the number of observations as scalar. The STATA code is the following:

```
ereturn list

scalar obs=e(N)

display obs
```

The first line reports the additional results mentioned above for the case of the last econometric specification. We ask the readers to look carefully at the STATA output for a more detailed view of the advantages of this command. The second line creates a scalar with the number of observations. Once this result is saved, it can be displayed on the STATA results window using the command `display` as in the third line of the code above.

STATA also records results using matrices by default. For instance, the regression coefficients are saved in a matrix. We can have access to these results using the following code:

```
matrix list e(b)
```

Alternatively, we can directly save the results of a regression into a matrix. We show you how to do it in the example below, including how to recover one particular coefficient:

```
matrix betha=e(b)

display betha [1,3]
```

In this case, we ask STATA to display the third coefficient, the one associated to the household head age (`agehead`) variable. We recommend the reader to try with other coefficients.

Once a regression is done, it is possible to recover its predicted values. For example, you can calculate the predicted dependent variable (y = x'β) or predicted errors (e = y-x'β) using the command `predict` and the various options provided by it. From there you can work with these variables. For example, you may be interested in knowing whether the predicted errors have mean zero or whether they follow a normal distribution. The STATA code would be the following:

```
predict yhatpov_HH

summarize yhatpov_HH

predict double resid, residuals

summarize resid
```

In the example above, we ask STATA to predict the dependent variable and the residuals. In the case of the predicted error, it has zero mean as expected for an OLS regression.

The marginal effects, which in the case of the general linear model are the estimated parameters, can be retrieved with the `mfx` command. It should be noted that for other estimation methods such as discrete choice models (logit or probit), the marginal effects are unlikely to be the same as the estimated parameter values. The STATA code is the following:

```
mfx, varlist(D mwagemale) eyex
```

The output is the following:

```
. mfx, varlist(D mwagemale) eyex

Elasticities after regress
      y  = Fitted values (predict)
         =  .82861501
------------------------------------------------------------------------------
variable |      ey/ex    Std. Err.     z    P>|z|  [    95% C.I.   ]      X
---------+--------------------------------------------------------------------
       D |   .0327543      .00845    3.88   0.000   .016201  .049308   .986455
mwagem~e |   -.014037      .00076  -18.39   0.000  -.015533 -.012541   85.0806
------------------------------------------------------------------------------
```

This command provides the predicted value for the dependent variable (poverty) and the marginal effects, standard errors and confidence intervals for the variables of interest. Notice that we are using the option `eyex` which computes the results in form of elasticities. We will cover more about these issues in the next chapter.

11. Hypothesis Testing

11.1. T-tests

A fundamental part of any statistical analysis is hypothesis testing to evaluate the statistical significance of a variable or set of variables. In the case of a single variable, we can use the command `ttest` that provides a t-student test. In this case, we work under the null hypothesis that the variable parameter x_k ($\beta_{mwagemale}$ in this case) is zero. If we reject this hypothesis, we can say that the variable is statistically significant.

The Student t-test has the following statistics:

$$t = \frac{\beta}{s/\sqrt{n}} \sim t_{n-1}$$

Let's implement a t-test in STATA. Assume you want to test whether the household poverty level is different according to treatment status. The STATA code is the following:

```
ttest pov_HH, by(D) unequal
```

STATA delivers the following table:

```
. ttest pov_HH, by(D) unequal

Two-sample t test with unequal variances
------------------------------------------------------------------------------
   Group |     Obs        Mean    Std. Err.   Std. Dev.   [95% Conf. Interval]
---------+--------------------------------------------------------------------
       0 |    3342    .8078995    .0068156    .3940106    .7945363    .8212627
       1 |  236166    .8270877    .0007782    .3781722    .8255625    .8286129
---------+--------------------------------------------------------------------
combined |  239508      .82682    .0007732    .3784036    .8253045    .8283354
---------+--------------------------------------------------------------------
    diff |           -.0191883    .0068599               -.0326382   -.0057384
------------------------------------------------------------------------------
    diff = mean(0) - mean(1)                                      t =  -2.7972
Ho: diff = 0                     Satterthwaite's degrees of freedom =  3428.67

    Ha: diff < 0                 Ha: diff != 0                 Ha: diff > 0
 Pr(T < t) = 0.0026         Pr(|T| > |t|) = 0.0052          Pr(T > t) = 0.9974
```

Hypothesis testing is a regular business in empirical work, so it is better to look carefully at this output. The table reports the means, standard deviations and confidence intervals for both groups. The null hypothesis is whether both means are the same. This hypothesis is rejected with a p-value of `0.0052`. The associated t-statistic is `-2.7972`. Notice that we make the assumption of unequal variances at the moment of running the test. Whether this assumption is credible, it is a matter of justification by the researcher.

It is interesting to compare this result to the coefficient reported in the first column of Figure 9. As you can see, the numbers are the same; they only differ in the sign of the coefficient. The reason why this is the case would be discussed in the next chapter.

The previous result is only true for the simple case of a regression with no control variables. When other variables are included, this might not hold anymore. The advantage of regression is that more sophisticated tests can be run since it allows for the possibility of controlling for other covariates. We can use the command `test` to evaluate the same hypothesis. Since this command uses an F distribution, it is better to introduce the concept before proceed with this discussion.

11.2. F-test

A researcher may want to analyze the joint significance of a set of variables in a regression, in which case an F-test will be necessary. As in the previous case, we assume that under the null hypothesis the coefficients are equal to zero and if we reject this, we won't be able to reject the joint significance of these variables.

The F-test has the following statistic:

$$F = \frac{SSR_R - SSR_{UR}}{SSR_{UR}} \frac{n-k}{k-1} \sim F_{k-1,n-k};$$

where SSR_R and SSR_{UR} are respectively the sum of square residuals for the restricted and unrestricted model, n the sample size and k the number of coefficients.

To illustrate the use of the command `test`, assume we want to the test the joint null hypothesis of a list of coefficients associated to different variables. The STATA code would be the following:

```
test D mwagemale agehead
```

The STATA output is below:

```
. quietly regress pov_HH D mwagemale agehead, vce (robust)
. test D mwagemale agehead

 ( 1)  D = 0
 ( 2)  mwagemale = 0
 ( 3)  agehead = 0

       F(  3,236026) =  145.16
            Prob > F =    0.0000
```

The F-statistic is 145.16 which imply that the null hypothesis is strongly rejected. Notice that we previously run a regression using the option `quietly`. This option permits to execute the command `regress` without reporting the results in the STATA results window.

Let's return to the previous question about the statistical significance of the treatment variable in the multiple regression above. The STATA code would be the following:

```
test _b[D]=0
```

The result is the following:

```
. test _b[D]=0

 ( 1)  D = 0
```

```
       F(  1,236026) =   15.04
            Prob > F =    0.0001
```

We test the null hypothesis that the coefficient of the treatment variable is equal to zero. As in the case of the t-test, we strongly reject the null of the coefficient being equal to zero. However, we wanted to recover the t-statistic rather than the F-statistic computed above. We can recover the t-statistic taking advantage of a known statistical property that establishes that the square root of F-statistic is equal to the t-statistic. We can do this in STATA using the following routine:

```
scalar t=r(F)^(1/2)

display t
```

Recall that STATA save results in scalar and matrices. We can create a scalar called `t` using the value of the F-statistic saved after applying the command `test`. We take the square root of this value and display the result in the STATA window results using the command `display`. The result is the following:

```
. display t

3.8783229
```

The reader can compare this result to the t-statistic reported for the coefficient of D in the previous regression. To facilitate the comparison, we replicate the results here:

```
. regress pov_HH D mwagemale agehead, vce (robust)

Linear regression                               Number of obs =  236030
                                                F(  3,236026) =  145.16
                                                Prob > F      =  0.0000
                                                R-squared     =  0.0016
                                                Root MSE      =  .37655

------------------------------------------------------------------------------
             |               Robust
      pov_HH |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
           D |   .0275134   .0070941     3.88   0.000      .013609    .0414177
   mwagemale |  -.0001367   7.42e-06   -18.42   0.000    -.0001513   -.0001222
     agehead |  -.0004427   .0000484    -9.15   0.000    -.0005376   -.0003479
       _cons |   .8346395   .0074654   111.80   0.000     .8200075    .8492714
------------------------------------------------------------------------------
```

As you can notice, the t-statistic reported by the table above is 3.88, exactly the same as the one we computed manually after applying the `test` command.

12. Organizing Empirical Work

Ideally, all scientific work should follow a set of standardized methods to ensure replicability of results. Therefore, one need to make sure that every step is clearly organized and documented. This implies the organization of your files in a methodologically sound manner in which all steps along the project are organized. This can be done under various techniques. We recommend the record of all the do files in which the routines for each step of the empirical work are saved.

There is no single way to organize an empirical project, so no general rules are available. However, some basic tips can be helpful when one work in a complex project. We propose a set of general rules based on our previous experience running impact evaluations. You are free to follow your own path.

- Organize your folders in a clear manner. Take special care with folders containing databases, do-files and estimation results. Keep the original names of the original databases; store them by source and year for easy econometric handling. Keep survey documentation (survey manuals, questionnaires, and other guides) in the same folders for quick reference.
- Record all your command routines in `.do` files. Applied research is usually characterized by the intensive use of .do files for each process and database. Therefore, it is recommended to use a way to "map" all of them, so the researcher can easily return to an empirical project if additional results or revisions are required. We usually call this .do file the `Master.do`. We use a `Master.do` to record and document the general structure of the set of .do files used in an empirical project from the first steps devoted to data management to more complex tasks such as econometric analysis and robustness checks. Bellow there is an example of how you can organize your work using a `Master.do`.

```
*******************************************************************************
*              APPLIED IMPACT EVALUATION: CHAPTER 1- INTRO TO STATA           *
*              MASTER DO                                                      *
*              www.stanislaomaldonado.org/aie/                                *
*******************************************************************************
clear
set mem 600m

/* COMMENT:
This do-file has been written as a guide for the econometric analysis performed for the Guided
Example 1: Exploring Regression Analysis. Each step is detailed in order to facilitate the
replication in the future.*/

*------------------*
* (1) GENERAL ROUTE:
*------------------*
/*COMMENT:
A local with the route is created indicating the place where the folder called "1. Intro to
STATA" is included. This route should be changed in case you were working using other computer.
*/
global path = "C:\Users\Stanislao\Dropbox\Teaching\1. Current\Econometrics\4. Handbook\1.
Modules\1. Intro to STATA"

*-------------------------------------------------------------------------------------*
* (2) MERGING FILES AND CREATING VARIABLES FROM ORIGINAL SURVEYS
*-------------------------------------------------------------------------------------*
/*COMMENT:
This section runs a set of do-files designed in order to compute basic variables */
local path = "$path"

              *************Active if is needed**************
              *do "`path'/do/Preparation.do" "`path'"
              ******************************************

*-----------------------------------------------------*
* (3) DESCRIPTIVE STATISTICS AND ECONOMETRIC ANALYSIS
*-----------------------------------------------------*
/*COMMENT:
This section runs a do-file for making descriptive statistics and runs a do-file for making the
econometric analysis.*/
              ***********Active if is needed***********
              do "`path'/do/Econometrics.do" "`path'"
              ****************************************
```

```
*End of this do-file
```

- As you can notice, in this fictitious `Master.do`, we record three basic steps. Firstly, we create a route or path using a global. This is a key step since it is very common for a researcher to work in different computers, so creating a path will save a lot of time because you won't need to change the path of your files anytime you start to work in a new computer. The second step basically takes the original survey files and creates a clean dataset to be used in the empirical work. Finally, the last steps contain the do-file that includes the econometric routines and save the results in a specific format.
- Do not store changes on the original databases, just create temporary datasets. This is critical since it is extremely helpful to guarantee replicability.
- Protect your final database. It is recommended to use the command `datasignature` to do so. This command assigns a signature to the database, so one can be sure that the same dataset is used. One advantage of this procedure is that keeps track of changes made to the database.

13. STATA Resources

There are many useful online resources for learning STATA. Perhaps the most complete are the materials provided by UCLA Academic Technology Services:

http://www.ats.ucla.edu/stat/stata/

A natural place to look for resources about STATA, is the webpage of STATA Corporation:

http://www.stata.com

Here is a list of some of the available resources there:

- Stata Technical Support FAQs, StataCorp LP, USA
- Answers to the most frequently asked questions in statistics, data management, graphics, and operating system issues.
- Statalist FAQ, StataCorp LP, USA
- Information on subscribing, unsubscribing, and using the free Stata discussion list at Harvard University.
- Stata video tutorials, StataCorp LP, USA
- Weekly video postings by StataCorp showing how to do common tasks in Stata.
- Stata training, StataCorp LP, USA
- Public training courses, on-site training courses, NetCourses, and more.

Some blogs also have useful information. Perhaps, a very useful one is the associated to the *Mostly Harmless Econometric* book written by Josh Angrist and Jorn Pischke:

http://www.mostlyharmlesseconometrics.com/blog/

14. Further Readings

There are a lot of books on STATA out there. STATA Corporation edits books based on this software. You can have a look at their webpage to get an idea regarding the books they produce:

http://www.stata-press.com/

Besides the regular books on STATA, some economists have published books on econometrics using STATA that you may find very useful for your work:

Baum, Cristopher (2006). *An Introduction to Modern Econometrics using STATA*. STATA Press.

Cameron, A. Colin and Pravin Trivedi (2009). *Microeconometrics using STATA*. STATA Press.

Baum (2006) offers a standard introduction to most of the standard econometric methods. Cameron and Trivedi (2009) is more ambitious and covers more advanced microeconometric techniques in a detailed manner. More advanced readers may also find useful the following book:

Baum, Cristopher (2009). *An Introduction to Programing in STATA*. STATA Press.

STATA is a powerful tool for graphical analysis. Readers may find the following book useful:

Mitchell, Michael (2012). *A Visual Guide to STATA Graphics*. Third Edition. STATA Press.

Besides books, STATA edits a per-reviewed journal called *STATA Journal*. You will need a subscription, but old articles are available online for free:

http://www.stata-journal.com/

For instance, we have used in this chapter the following article:

Lokshin, Michael and Zurab Sajaia (2008). "Creating print-ready tables in STATA," *The STATA Journal*, 8, Number 3, pp. 374-389.