# Introduction to gretl

## Applied Economics

Department of Economics
Universidad Carlos III de Madrid

# Outline


1. What is `gretl`?
2. `gretl` Basics
3. Importing Data
4. Saving as `gretl` File
5. Running a Script
6. First Exercises
7. Commands on Datasets
8. Commands on Variables
9. Graphs

# What is gretl?

- gretl is an acronym for Gnu Regression Econometrics and Time-series Library
- it is free econometrics software
- it has an easy Graphical User Interface (gui)
- it runs least-squares, maximum-likelihood, systems estimators...
- it outputs results to several formats
- very important for us in this course: it admits scripts (sequence of commands saved in a file)

# How do I get gretl?

- already installed in many computer rooms at Carlos III
- can be downloaded from **http://gretl.sourceforge.net** and installed on your personal computer
- it runs on Windows, Mac, Linux

# How do I work with gretl?(1/2)

- the easiest way for beginners is by using its graphical user interface
- you can also use the "console" button of the toolbar: from the prompt (?) you can execute `gretl` commands one line at a time.
- the most efficient way is by using *scripts:*
  1. create a script file, write `gretl` commands– one every line–, and save it
  2. run the script using the `gui`
  3. inspect output
  4. if needed, change script file, save it, and go back to step 2

# How do I work with gretl?(2/2)

you know your way using the gui, but want to know about scripts...

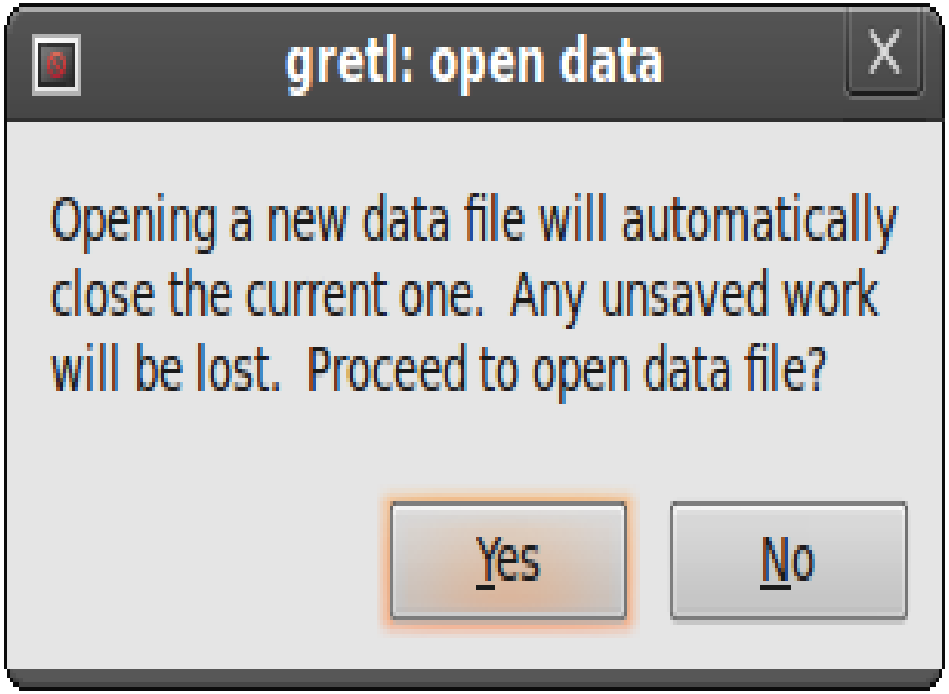
1. actions you do with the gui are stored as script in a filed called session.inp
2. gretl comes with over 70 practice scripts
3. the manual gives good advice and devotes several chapters to provide good programming solutions
4. this course will provide you with tested scripts

- today we are going to use the gui to create a script which opens a dataset in xls format, describes it, edits it, and saves a dataset in gretl

# Main Window (1/2)

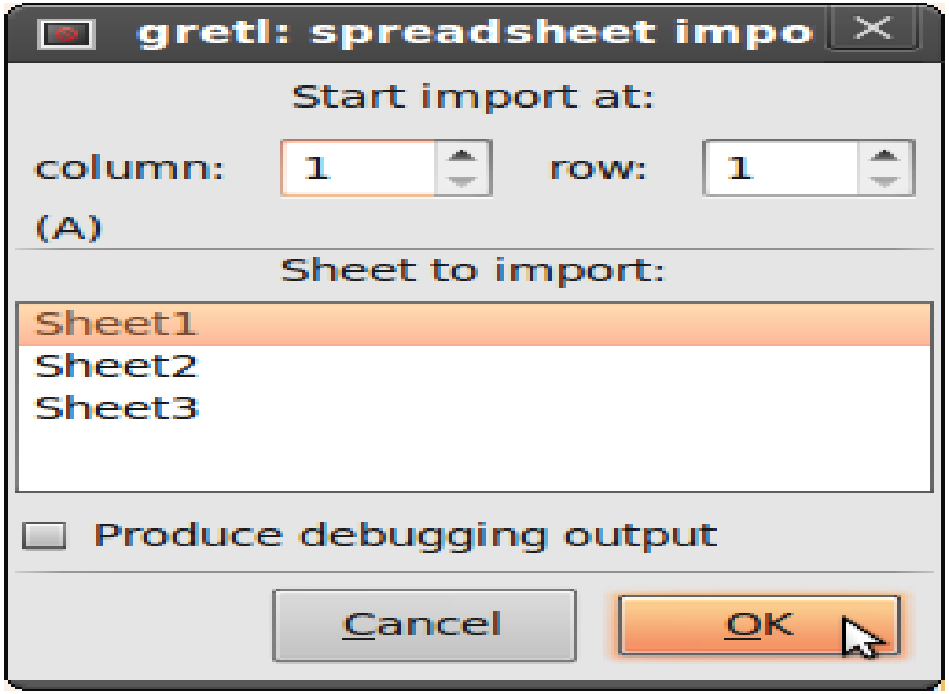
# Importing Example1.xls

this warning only takes place with the `gui`



# Importing Example1.xls

You can import from any of the sheets



# Importing Example1.xls

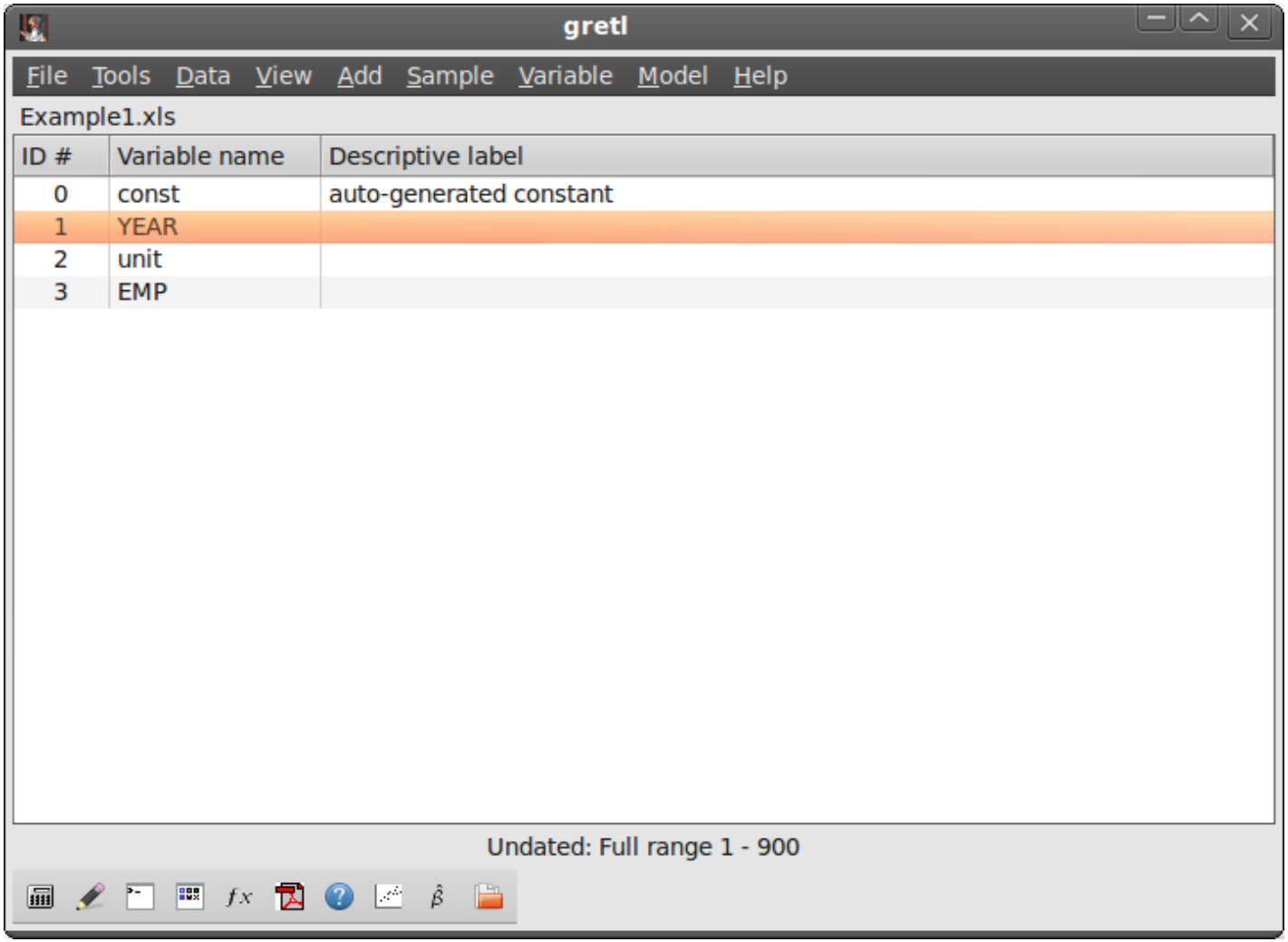
`gretl` gives you some info

# Importing Example1.xls

you can import cross-sections, time-series, or panel data

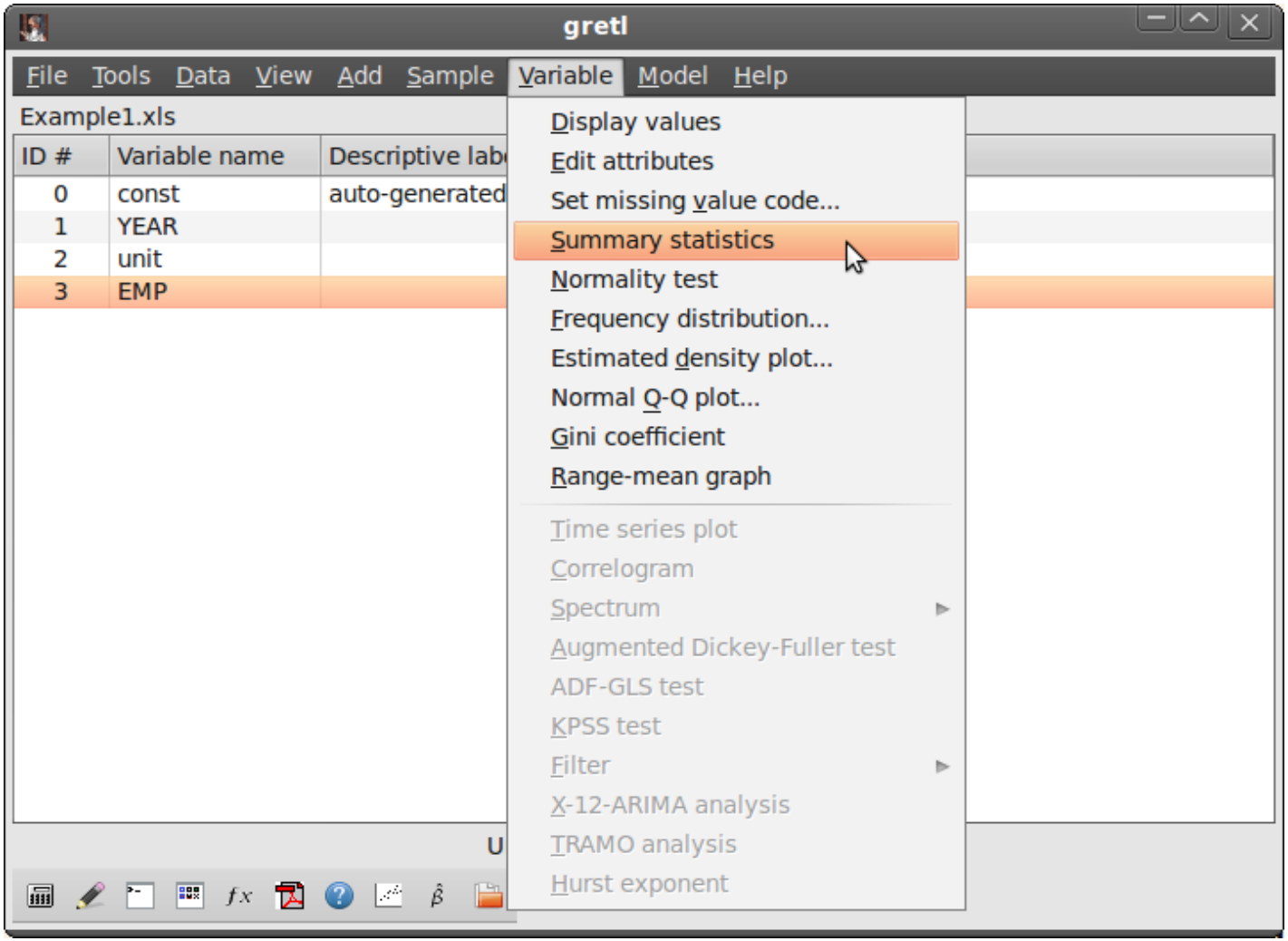
# Importing Example1.xls

now the data is available in `gretl`



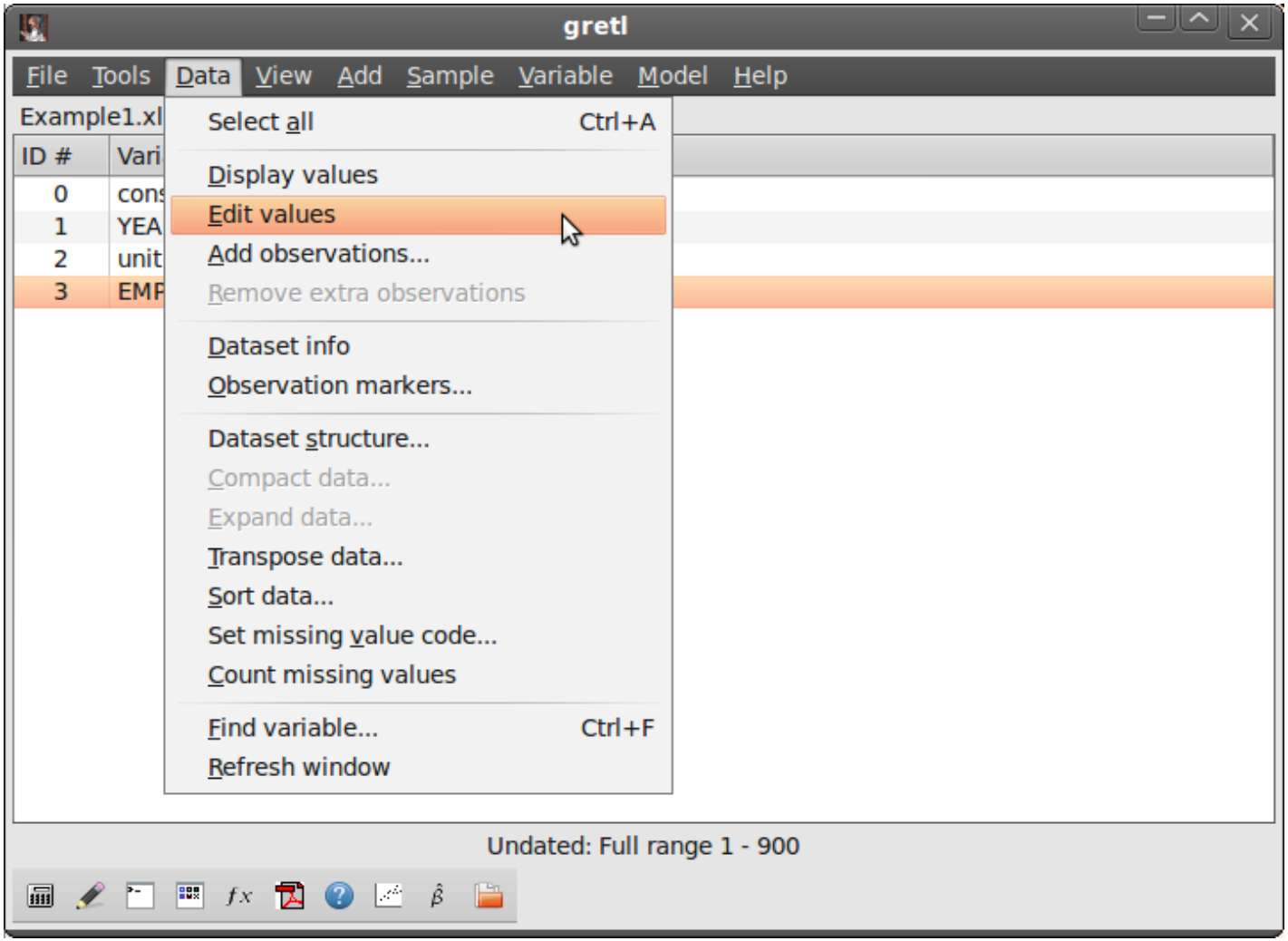
# Describing a variable in a dataset

After selecting one variable, Variable/Summary statistics



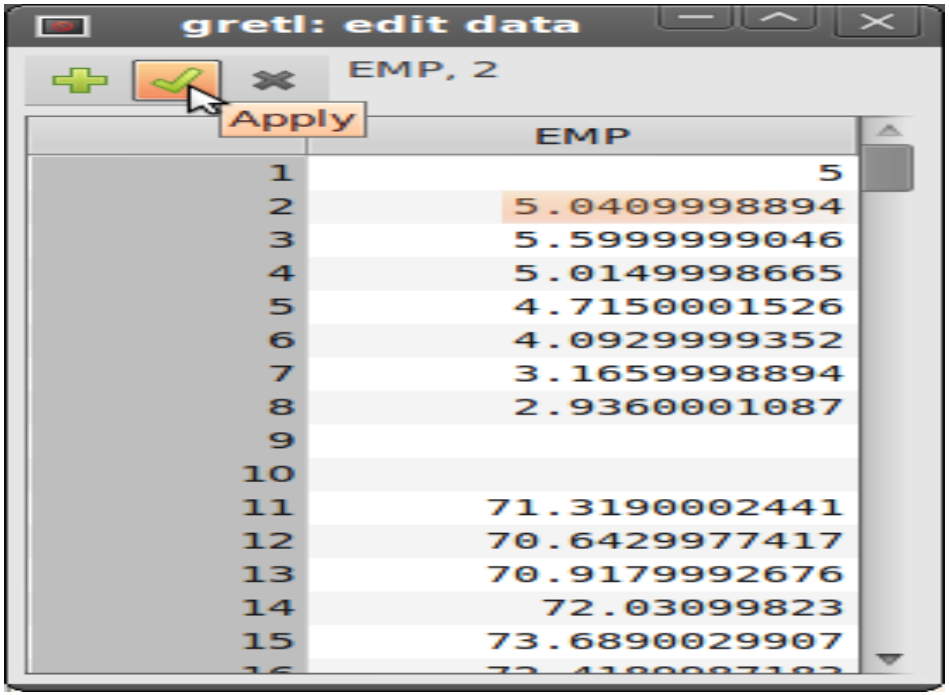
# Editing a variable in a dataset

## After selecting variable, Data/Edit Values



# Editing a variable in a dataset

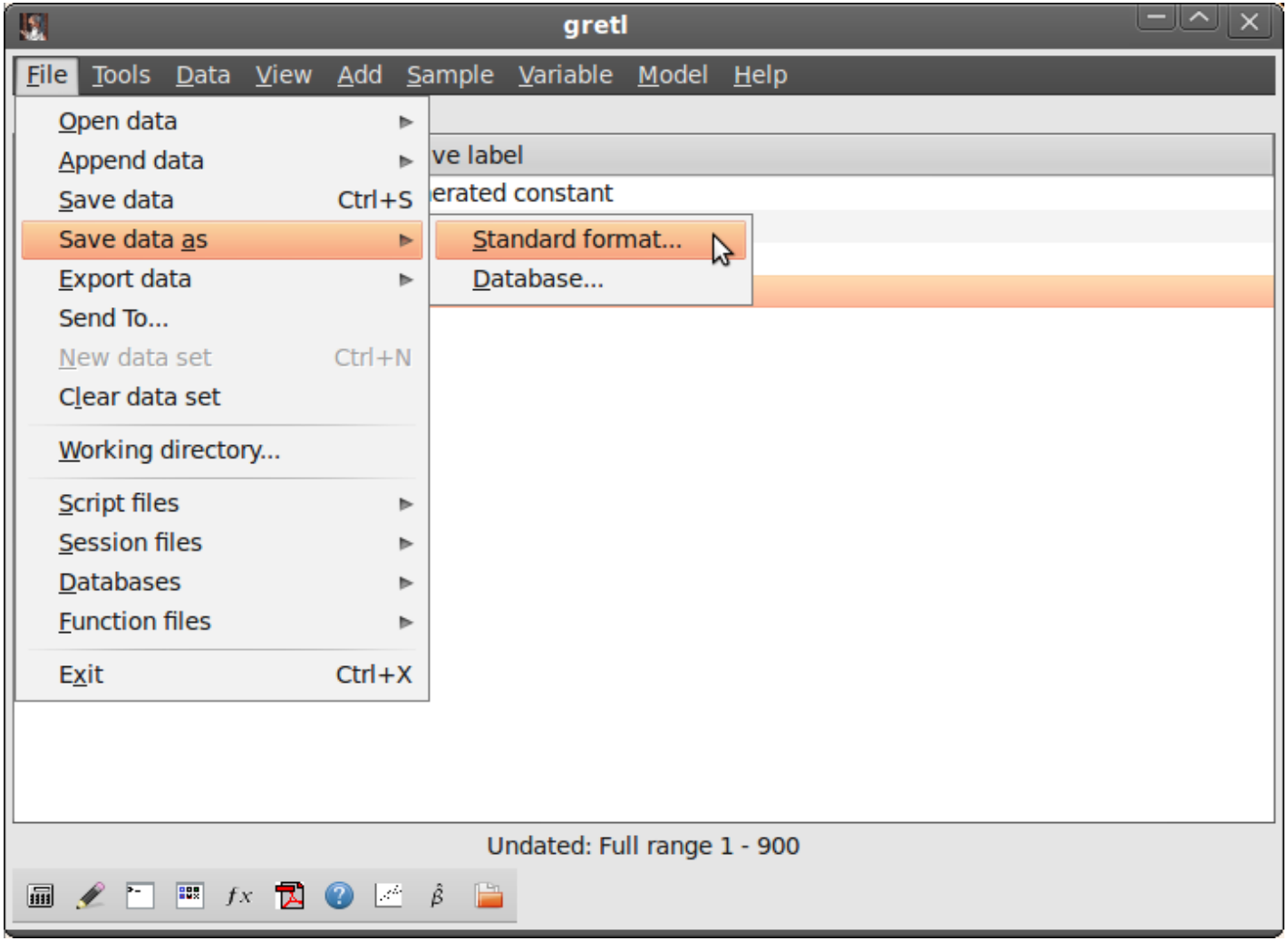
we add 5 in the first observation




(to display EMP and check editing has been succesful, double-click on EMP in main Window)

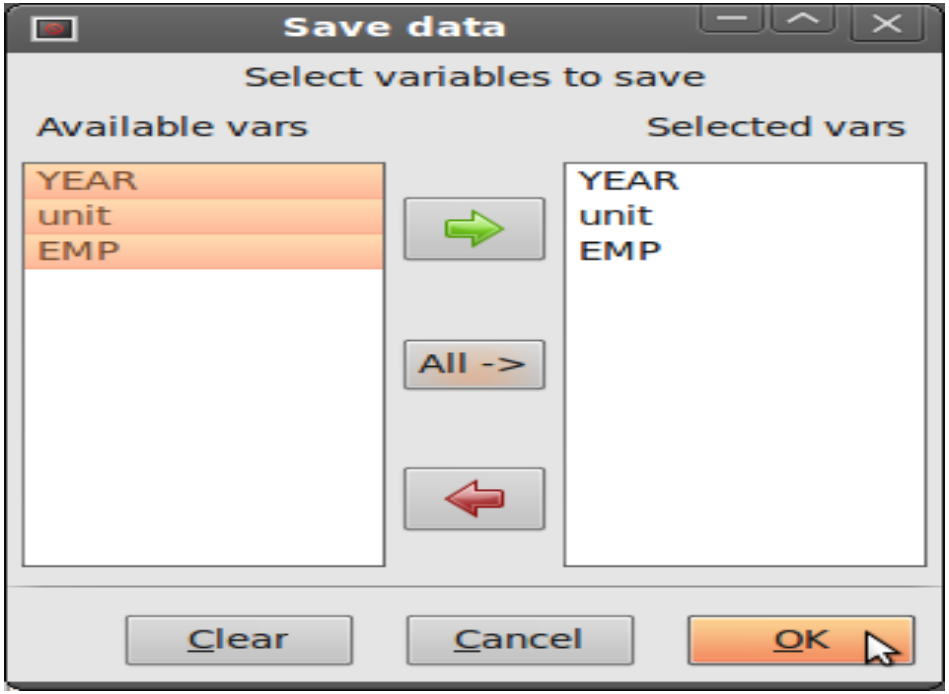
# Saving as a new gretl File

File > Save Data as > Standard format



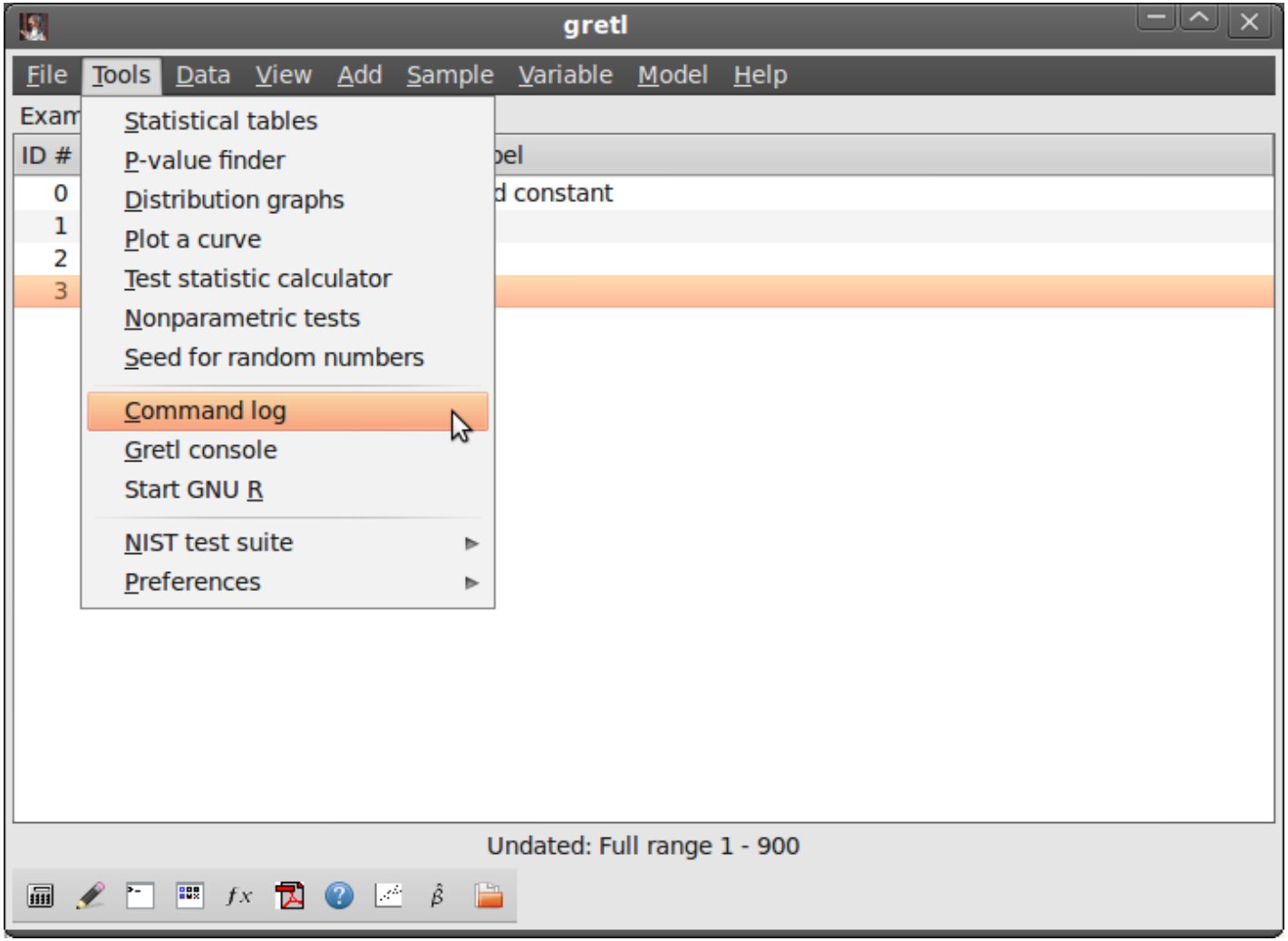
# Saving as a new gretl File

you can select a subset of the variables



# Looking at the Session Script

## Tools > Command log



# Looking at the session script

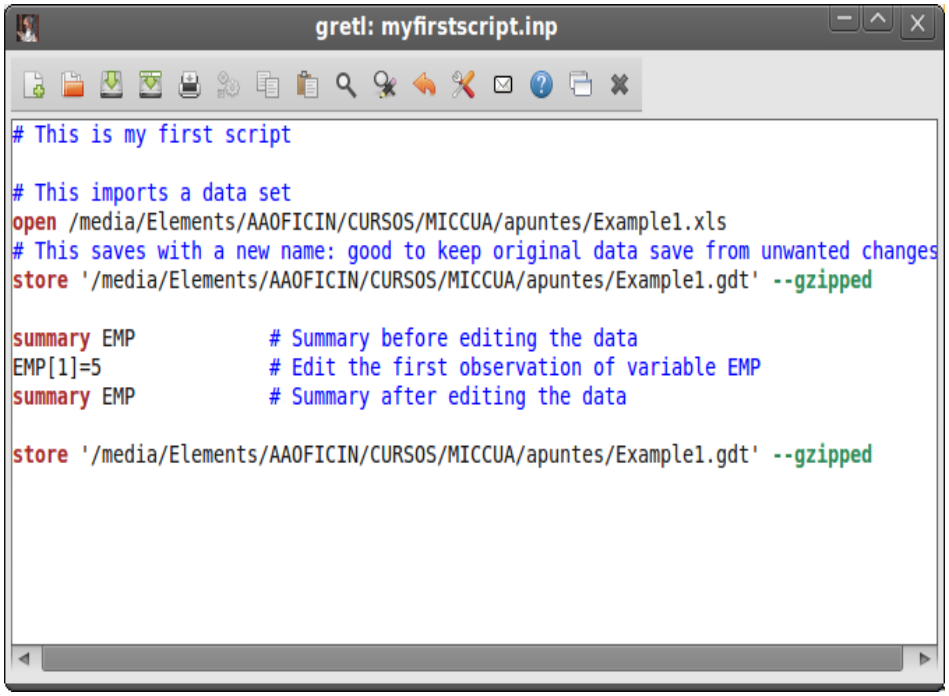
note that editing is not recorded and storage is commented out

gretl: command log

```
# Log started 2011-01-19 18:33
# Record of session commands.  Please note that this will
# likely require editing if it is to be run as a script.
open /media/Elements/AAOFICIN/CURSOS/MICCUA/apuntes/Example1.xls
summary EMP
# store '/media/Elements/AAOFICIN/CURSOS/MICCUA/apuntes/Example1.gdt' --gzipped
```

# Changing the script

- type the commands you want to execute in the box using one line for each command
- to save the file, use the “save" button at the top of the box.
- to run the program, click your mouse on the “gear" button.



# More on scripts

- using File/Script files/New script you open the command script editor
- If you have a very long command that exceeds one line, use the backslash (\) as a continuation command
- using scripts (and the console) requires you to use the correct language syntax
- `gretl`'s language is case sensitive: `gretl` considers x to be different from X
- you can find all the commands in the `gretl` command reference (the fourth botton from the right hand side of the toolbar
- at the console window, you can type `help`

# Exercise 1

1. Load the file food.gdt. The data set contains two variables named x and y. The variable y is weekly expenditures on food in a household and x is weekly income measured in $100 increments.
2. Using GUI, add logs of both variables.
3. Using GUI, summarize y for households with income below the median.
4. Using GUI, plot y against x.
5. Write all the commands in a script file and save it.
6. Open the script and run the commands.
7. Save the new dataset.

# Exercise 2

The file BWGHT.csv contains information about infants' birth weight, sex and race, the income of their faimilies, their parents' education and the number of cigarettes the mother smoked per day during pregnancy (*cigs*).

1. Import from gretl the file BWGHT.csv
2. Compute the average birth weight for all the babies in the sample, and separately for boys and girls.
3. Compute the proportion of mothers that smoked during pregnancy.
4. Which familiy variables do you think are related to children's birth weight? Compute the correlation among *bwght* and those variables.
5. Save the new database in gretl format.
6. Write down all the previous commands in a script and save it. Close your session, open the script and run all the commands together.

# Basic commands for data management (1/2)

## Commands on the entire data

- `open`: opens a data file replacing any data file already open
- `smpl`: defines the sample range
- `dataset`: sorts/clears/transposes/adds observations and more
- `setobs`: declares the structure of the data (cross-section, time-series, panel)
- `append`: appends the content of a data to the current dataset
- `store`: saves the data into a file

# Basic commands for data management (2/2)

## Basic commands on variables

- `genr`: creates a new variable
- `delete`: removes variables
- `setinfo`: sets attributes of a variable
- `rename`: renames a variable
- `summary`: shows summary statistics for variables
- `print`: lists the values of variables

# open *dataname* --www --sheet="*name*" --coloffset=# --rowoffset=#

- opens a dataset replacing any already loaded data
- --www opens database in `gretl` server
- with spreadsheets, it selects the worksheeet, and the first column and row
- the first row must contain valid variable names. In the case of an ASCII or CSV import, if the file contains no row with variable names the program will automatically add names, v1, v2 and so on.

open C:\there\mydata.xls --sheet="mysheet" --coloffset=3 --rowoffset=2

- opens worksheet mysheet from C:\there\mydata.xls
- reads the data from the fourth column and third row

# smpl (#*start* #*end* | *condition* −−restrict | # −−random | full) −−replace −−balanced

- *condition* −−restrict: restricts the sample to observations that satisfy the *condition*
- # −−random: # cases are ramdomly selected
- full: restores the full data range
- sample restrictions are by default cumulative: −−replace turns off all previous restrictions

## Examples (using Example2.xls)

- smpl *YEAR*!=1976 −−restrict
- smpl *EMP* > 3 −−restrict −−replace
- smpl 50 −−random

# dataset (addobs # | transpose | sortby *varname* | resample # | clear)

- addobs: adds extra observations at the end
- transpose: transposes current data set.
- sortby: sorts data by varname (dsortby: descending order) (a list of variables can be provided; available only for undated data).
- resample: random sampling (Constructs a new dataset by random sampling, with replacement, of the rows of the current dataset. The original dataset can be retrieved via the command smpl full).
- clear: clears out current data

## Examples

- dataset sortby EMP
- dataset resample 500
- dataset clear

# setobs #*freq* #*start* (−−cross-section | −−time-series | −−stacked-cross-section | −−stacked-time-series)

- #*freq* represents frequency in time-series data
- in panel, #*freq* is units in stacked cross-sections or periods in stacked time series
- for cross-sections, #*freq*=1
- #start=1 for panels and cross-sections
- in time series, #start is the starting date

# setobs *unitvar timevar* ––panel-vars

- imposes a panel interpretation
- sorts data as stacked time series, by ascending values of *unitvar*

## Examples

- setobs 1 1 ––cross-section
- setobs 20 1:1 ––stacked-time-series
- setobs *unit year* ––panel-vars

# append *newdata* −−time-series

- opens a data file and appends the content to the current dataset
- First case: additional observations for existing variables
- Second case: new variables (best if # obs compatible)
- Third case: appends a time series in a panel

## First Case

- open C:\there\Example2.xls −−sheet="first100"
- append C:\there\Example2.xls −−sheet="moreunits"
- appends worksheet *moreunits* from C:\there\Example2.xls

## Second Case

- append C:\there\Example2.xls −−sheet="wages"
- appends worksheet *wages* from C:\there\Example2.xls

# append

## Third Case

- You have a panel and you want to add a variable which is available in time-series form. For example, you want to add annual CPI data to a panel in order to deflate nominal income figures.
- open the data: C:\there\Example2.xls −−sheet="first100"
- you need to have a panel: setobs *unit year* −panel-vars
- append C:\there\Example2.xls −−sheet="cpi"

# store *datafile* [*varlist*] −−gzipped −−overwrite

- by default data saved in `gretl` format
- also exports to csv (using −−csv) and other formats

store C:\there\mydata.gdt

- saves current data to C:\there\mydata.gdt

# [genr] *newvar = formula*

- a formula is a well-formed function of variables
- the range over which the result is written depends on the current sample
- arithmetical operators: ^, *, /, + , -
- boolean operators: ! (negation), && (AND), || (OR), >, <, =, >= , <= , !=
- look at the `gretl` Function Reference (Help/Function Reference) for built-in functions

## Examples

- genr y = 3 + 2 * x1 + 5 * x2 + error
- D1976 = (YEAR = 1976)
- genr avgy = mean(y)

# delete [ *varlist* ] −−db

- removes listed variables
- if no *varlist* is given, it deletes the last (highest numbered) variable from the dataset
- −−db: deletes variables from a gretl database

# setinfo *varname* −d "*thislabel*" −n "*thisname*" −−discrete −−continuous

- −d "*thislabel*": *thislabel* is set as the variable's descriptive label
- −n "*thisname*": *thisname* is used in place of the variable's name in graphs
- −−discrete: marks variable as discrete (by default variables are continuous)

## Examples

- setinfo x1 −d "Description of x1" −n "Graph name"
- setinfo z −−discrete

## rename *varname newname*

- changes the name of the variable
- names must be of 15 characters maximum
- they must start with a letter
- they must be composed of only letters, digits, and the underscore character

### Example in bwght.gdt

- rename bwght birth_weight

# summary [ *varlist* ] −−simple −−by=*byvar*

- prints summary statistics for variables in *varlist*
- if *varlist* is omitted, it prints statistics for all variables
- −−simple: only prints the mean, minimum, maximum and standard deviation
- −−by=*byvar*: statistics are printed for sub-samples defined by the values taken on by *byvar*

Example in bwght.gdt

- summary bwght
- summary bwght −−simple
- summary bwght −−simple −−by=male
- summary bwght −−simple −−by=parity

# print [*varlist*] −−byobs −−no-dates

- prints the values of the variables in *varlist*
- if no list is given, prints the values of all variables
- −−byobs: data are printed by observation, not by variable
- you can also print strings

## Examples

- print bwght male −−byobs
- print bwght ; male −−byobs :
- print "This is a comment"

# A (very) brief graph menu

- gnuplot *yvars xvars*: xy graphs
- scatters *yvar* ; *xvarlist*: pairwise scatterplots
- freq *yvars*, or using the `gui` Variable/Frequency Distribution: histograms