

# Introduction to gretl

## Applied Economics

---

Dali Laxton

16.10.2020

# Outline

- 1 What is gretl?
- 2 gretl Basics
- 3 Importing Data
- 4 Saving as gretlFile
- 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?

Handy for remote learning since it is open-source free software anyone can install it on home desktop without the need of license

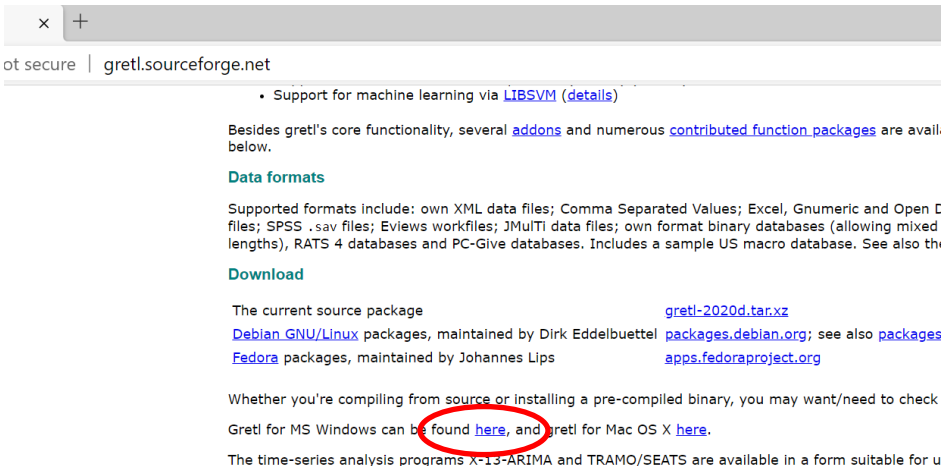
In order to install it go to:

<http://gretl.sourceforge.net>

It runs on Windows, Mac, Linux

# Installation

Open the webpage and go down where Download section is located.  
Click on the circled button for windows users



Not secure | gretl.sourceforge.net

- Support for machine learning via [LIBSVM \(details\)](#)

Besides gretl's core functionality, several [addons](#) and numerous [contributed function packages](#) are available below.

### Data formats

Supported formats include: own XML data files; Comma Separated Values; Excel, Gnumeric and Open Office files; SPSS .sav files; EViews workfiles; JMulTI data files; own format binary databases (allowing mixed lengths), RATS 4 databases and PC-Give databases. Includes a sample US macro database. See also the [Data formats](#) page.

### Download

The current source package [gretl-2020d.tar.xz](#)  
[Debian GNU/Linux](#) packages, maintained by Dirk Eddelbuettel [packages.debian.org](#); see also [packages](#)  
[Fedora](#) packages, maintained by Johannes Lips [apps.fedoraproject.org](#)

Whether you're compiling from source or installing a pre-compiled binary, you may want/need to check Gretl for MS Windows can be found [here](#), and gretl for Mac OS X [here](#).

The time-series analysis programs X-13-ARIMA and TRAMO/SEATS are available in a form suitable for u

# Installation

Choose the .exe file which is appropriate for your system



[\[gretl\\_main\\_page\]](#)

## Gretl: Gnu Regression, Econometrics and Time-series Library

for Microsoft Windows

### System requirements

As of version 1.9.4, gretl requires Windows XP or higher and a processor that supports the SSE2 instruction set. SSE2 support is found in all modern processors; it is absent in AMD CPUs prior to Athlon 64, and in Intel CPUs prior to the Pentium 4. For versions of gretl that will run on older systems, [see below](#).

### Downloads

If you have the rights of a "power-user" or better on Windows, choose a self-installer from the first or second column below; just download and run the exe file. This applies to most people.

If you have no administrator rights on Windows choose a zip archive from the third column; unzip this in any location where you have write permission. *Note:* you must preserve the internal structure of the archive or gretl will not work. In unzipping programs this option may be called "enable folders". The whole archive is in a directory called gretl. For example, if you unzip the archive into a directory named c:\userdata, the gretl GUI program will be at c:\userdata\gretl\gretl.exe.

The current "snapshot" of gretl is more up to date than the release: often it will contain bug-fixes but sometimes it will contain newly introduced bugs. To see what's new in the snapshot, take a look at the [Change log](#) (the "in progress" entry).

*Note:* the 32-bit version of gretl will run on 64-bit Windows, but not vice versa. If you're not sure if your Windows installation is 64-bit or not you can check at the [Microsoft knowledge base](#).

*self-installer (32-bit)*   *self-installer (64-bit)*   OR   *zip archive (no admin rights)*

latest release (Aug 6, 2020)   [gretl-2020d-32.exe](#)   [gretl-2020d-64.exe](#)   [gretl-2020d-win32.zip](#)

OR current snapshot   [gretl\\_install-32.exe](#)   [gretl\\_install-64.exe](#)   [gretl-win32.zip](#)

The executables were cross-compiled under GNU/Linux using [mingw-w64](#). The free installer program is courtesy of [Jordan Russell](#).

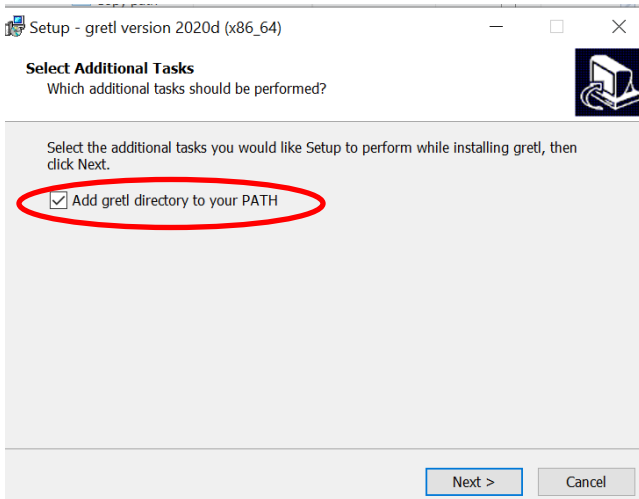
### Optional extras you may wish to install

X-13-ARIMA-SEATS (seasonal adjustment. ARIMA models)

[x13as Install.exe](#) OR (64-bit) [x13as Install-64.exe](#)

# Installation

Run the .exe file and choose default options to proceed. Do not forget to add gretl directory to your PATH



# Start gretl

To start gretl you can just type **gretl** in your search button. Select it and an empty gretl window will pop out. You can either run commands directly in the gretl console or write scripts and save them for future use



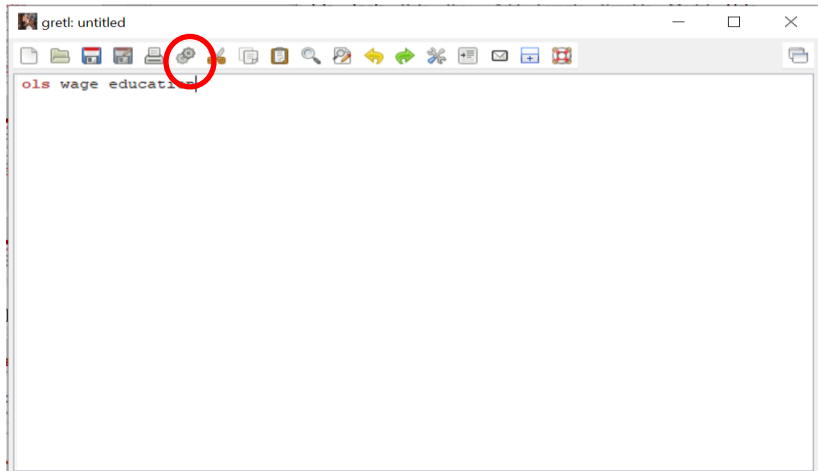


## How do I work with gretl?(1/2)

- 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

## Running commands in the script

Click CTRL + R or click the button shown below. The window with outputs (or errors) will open. If you save these scripts, it will be saved as .inp file.



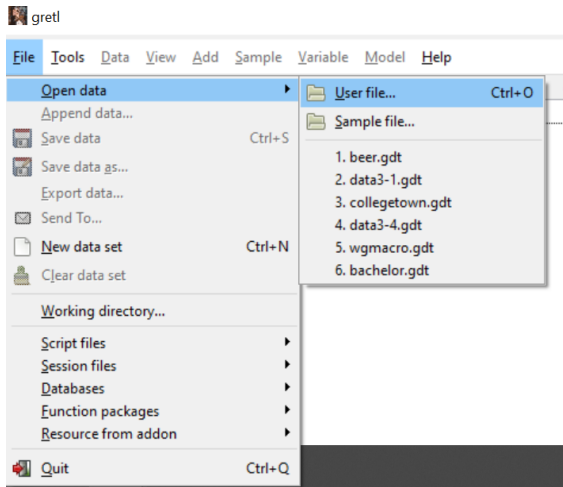
## How do I work with gretl?(2/2)

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 led 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

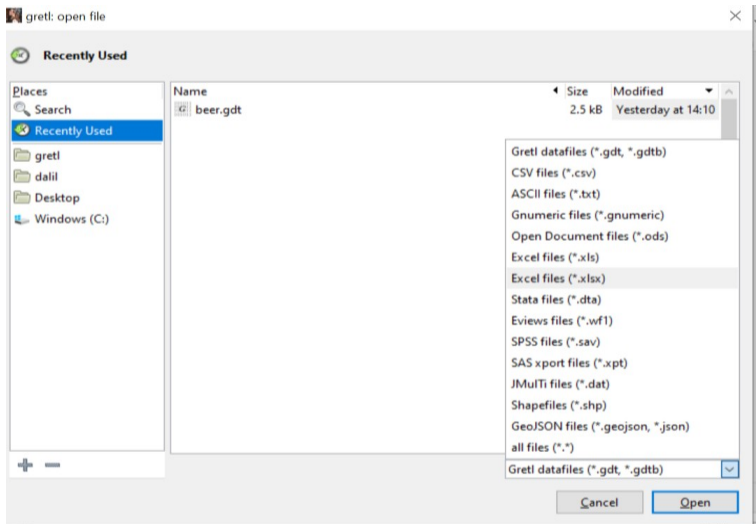
## Opening datasets

- ❑ Gretl comes with inbuilt datasets which you can play with
- ❑ Go File -> Open data -> Sample file and this will give you lots of datasets, or alternatively, go to specific data file which you want to access in your computer by choosing User file instead of sample file



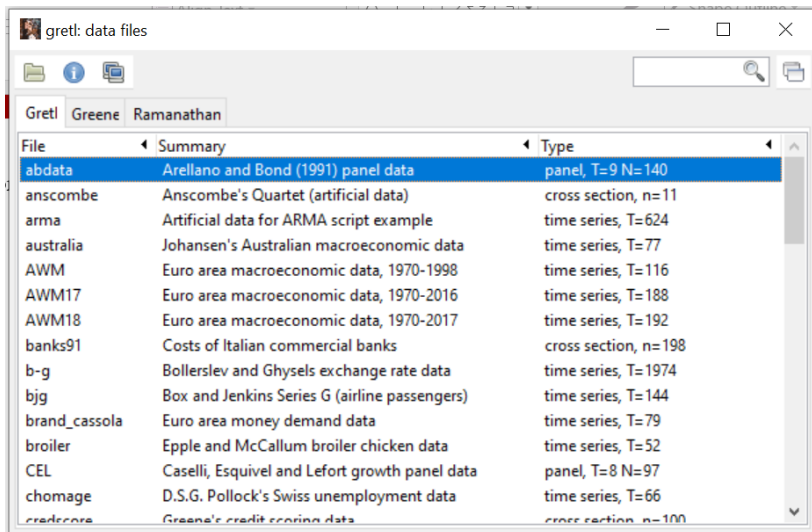
## Opening datasets

- If you want to open file which has a different extension, for example excel, when you go to User File, do not forget to set it for excel searching, otherwise nothing will be shown in the window



## Opening datasets

- For now let's open a **Sample file** the very first one offered 'abdata' by double-clicking it with a mouse

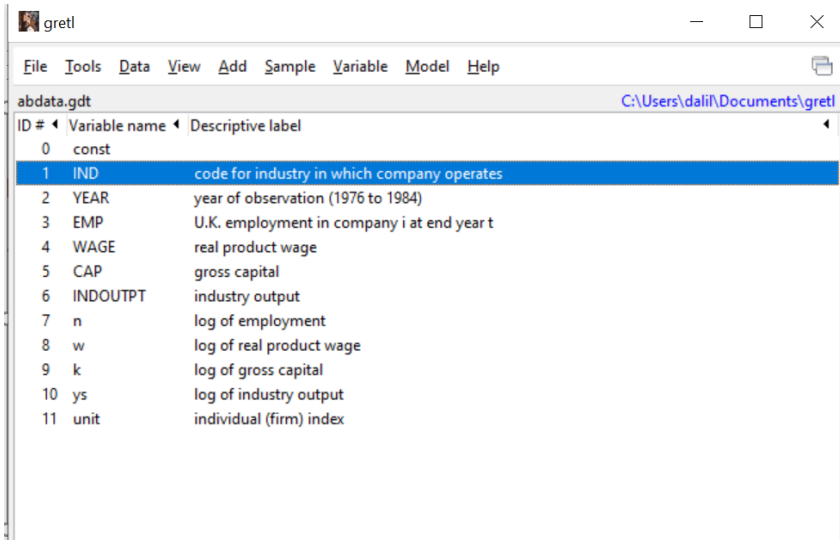


The screenshot shows the 'gretl: data files' window. At the top, there are window control buttons and a search bar. Below that, there are tabs for 'Gretl', 'Greene', and 'Ramanathan'. The main area is a table with three columns: 'File', 'Summary', and 'Type'. The 'abdata' file is selected and highlighted in blue.

File	Summary	Type
abdata	Arellano and Bond (1991) panel data	panel, T=9 N= 140
anscombe	Anscombe's Quartet (artificial data)	cross section, n= 11
arma	Artificial data for ARMA script example	time series, T=624
australia	Johansen's Australian macroeconomic data	time series, T=77
AWM	Euro area macroeconomic data, 1970-1998	time series, T=116
AWM17	Euro area macroeconomic data, 1970-2016	time series, T=188
AWM18	Euro area macroeconomic data, 1970-2017	time series, T=192
banks91	Costs of Italian commercial banks	cross section, n= 198
b-g	Bollerslev and Ghysels exchange rate data	time series, T=1974
bjg	Box and Jenkins Series G (airline passengers)	time series, T=144
brand_cassola	Euro area money demand data	time series, T=79
broiler	Epple and McCallum broiler chicken data	time series, T=52
CEL	Caselli, Esquivel and Lefort growth panel data	panel, T=8 N=97
chomage	D.S.G. Pollock's Swiss unemployment data	time series, T=66
credscore	Greene's credit scoring data	cross section, n=100

## Playing with data

The following window will pop out. You can now check the values of each variable by double clicking them. You can edit them etc.

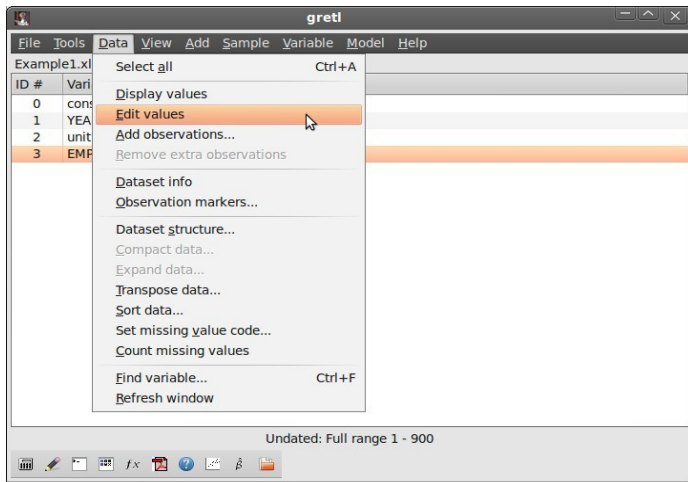


The screenshot shows the gretl software interface. The window title is "gretl" and the file name is "abdata.gdt". The menu bar includes "File", "Tools", "Data", "View", "Add", "Sample", "Variable", "Model", and "Help". The main area displays a list of variables with their IDs, names, and descriptive labels. The variable "IND" is highlighted in blue.

ID #	Variable name	Descriptive label
0	const	
1	IND	code for industry in which company operates
2	YEAR	year of observation (1976 to 1984)
3	EMP	U.K. employment in company i at end year t
4	WAGE	real product wage
5	CAP	gross capital
6	INDOUTPT	industry output
7	n	log of employment
8	w	log of real product wage
9	k	log of gross capital
10	ys	log of industry output
11	unit	individual (firm) index

# Operations on variables

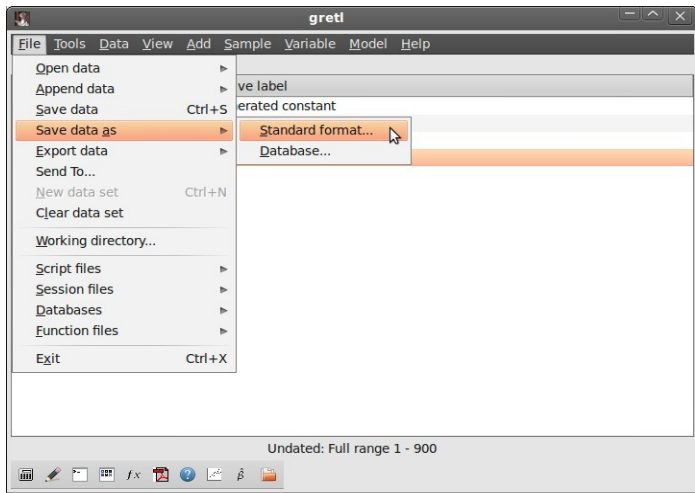
- ❑ See summary statistics of variables by highlighting them and clicking Variable -> summary statistics in the upper panel
  - ❑ Change values of variables by highlighting it and clicking Data -> Edit values
- Edit values





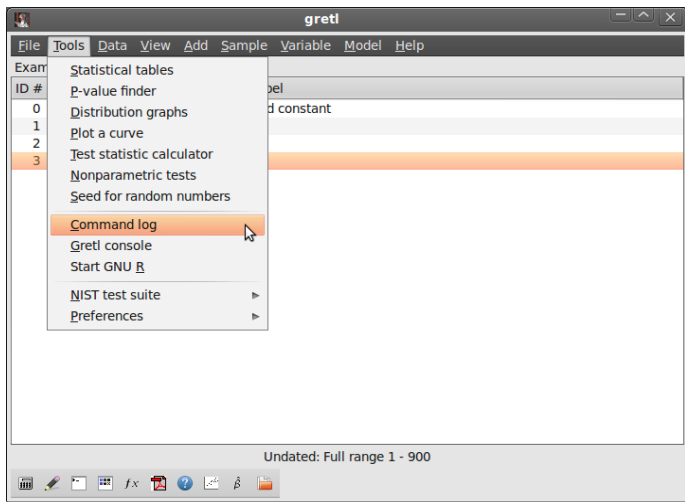
## Saving as a new gretl File

You can open any other type of file using gretl (excel), do some operations on them and then save as a gretl file (.gdt) by clicking: File -> Save Data as -> Standard format



# Looking at the Session Script

Tools > Command log displays the operations that we have done so far



## 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 after clicking Help -> command reference
- at the console window, you can type help

# Exercise 1

- Load the file `engel.gdt`. The data set contains two variables named `income` and `foodexp`. The variable `foodexp` is annual expenditures on food in a household and `income` is annual income measured in \$100 increments.
- Using GUI, display logs of both variables.
- Using GUI, summarize `foodexp` for households with income below the median.
- Using GUI, plot `foodexp` against `income`.
- Write all the commands in a script file and save it. Open the script and run the commands.
- Save the new dataset.

## Exercise 2

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

- 1 Import from `gretl` the file `BWEIGHT.xlsx`
- 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 family 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 gretlserver
- with spreadsheets, it selects the worksheet, 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 --col set=3  
--row set=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 randomly 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 data le [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:  $\wedge$ ,  $*$ ,  $/$ ,  $+$ ,  $-$
- boolean operators:  $!$  (negation),  $\&\&$  (AND),  $\|\|$  (OR),  $>$ ,  $<$ ,  $=$ ,  $>=$ ,  $<=$ ,  $\neq$
- look at the `gret!` 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