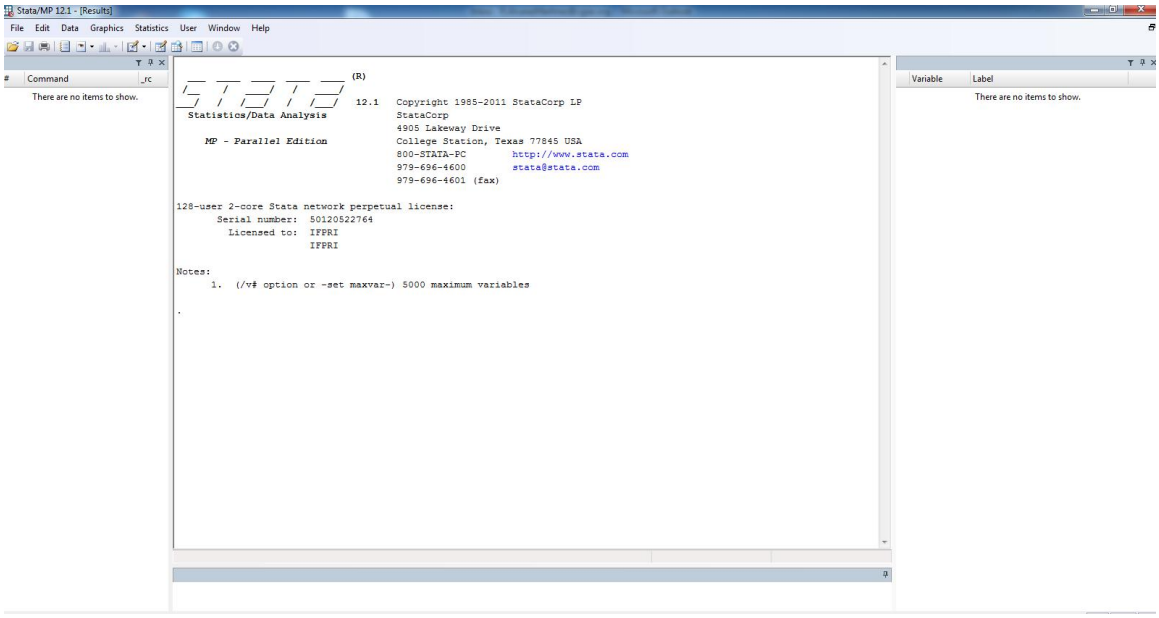
STATA 12

1. Introduction



Stata windows
Review: where past commands appear
Variables: variables list of the dataset in use appear here on the variable window
Stata Command: where you type your commands
Stata Results: where results are displayed

Stata toolbar

Stata files
Databases, extension ".dta"
Do files: collect all the commands in a program or script, extension ".do"
Log files: store the commands in addition to the output from the commands, extension ".log"
Both the do and the log files can be read as text files. Stata has a do file editor where you can write and save your programs.

Basic syntax

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

2. Loading, inputting and saving data

- clear: clears the memory.

clear

- set more on/off: tells Stata to pause/not to pause.
  set more off

- use: loads a Stata-format dataset.
  use filename [, clear]
  use "c:\example1.dta", clear

- save: guarda una base de datos.
  save filename [, replace]
  save "c:\example1.dta", replace
  saveold "c:\example1.dta", replace

Note: If you have all your files in a particular folder, you can specify the route, that is:
cd: display your current directory
cd "c:\<file directory>": set the working directory or change the current directory
dir: list the names of files in the specified directory
use example1.dta, clear: opens the data files that are already in Stata format (with extension .dta)

- edit: brings up a spreadsheet-style data editor for entering new data and editing existing data.
  edit [varlist] [if exp] [in range] [, nolabel]
  edit
  edit, nol

- insheet: is intended for reading files created by spreadsheet or database programs. Reads text (ASCII) files where there is one observation per line and the values are separated by tabs or commas.
  insheet [varlist] using filename [, options]
  insheet using example1.txt, clear
  see browse below

3. Description of database

- describe: displays a summary of the contents of the dataset.
  describe [varlist]
  describe

- codebook: displays a codebook for all the variables in the dataset.
  codebook [varlist]
  codebook codjoven educa incomea

4. Labels

- rename: changes the name of an existing variable.
  rename old_varname new_varname
  rename codjoven code

- label: attaches a value label and a label to a variable.
  label define lblname # "label" [# "label" ...] [, add modify replace nofix]
  label list
  label drop {lblname [lblname ...] | _all}
  label values varname lblname
  label variable varname "label"

  label de marital 1 "single" 2 "married" 3 "common law" 4 "other"
  label val marital marital
  label l
  bro
  label var educa “education level”

5. Listing

- format: allows you to specify the display format for variables. The internal precision of the variables is unaffected.
  format varlist %fmt
  where %fmt = %9.2f, %10s

  format incomea %9.2f

- list: displays the values of variables.
  [by varlist:] list [varlist] [if exp] [in range] [, nolabel]
  list incomea male city if city==1|city==2, nol

- browse: is like edit except that it will not allow changing the data.
  browse [varlist] [if exp] [in range] [,nolabel]
  bro in 1/10

- gsort: arranges the observations to be in ascending or descending order of the specified varnames.
  gsort [+|-] varname [[+|-] varname ...] [, generate(newvar) mfirst]
  gsort –income age, g(order)
  bro income age order

6. Create and delete variables

- generate: creates a new variable.
  generate newvar = exp [if exp] [in range]

```
gen manuel=_n
bro income age order manuel
gen manuel2=edad*edad if edad>20 & city!=1
bro age city manuel2
```

- egen: extension of generate
  egen newvar = fcn (varlist) [if exp] [in range] [,options]
  where fcn = (r)sum, (r)mean, (r)max, (r)min, group
  where options = by (varlist)

```
egen manuel3=mean(age), by(city)
bro city age manuel3
egen manuel4=rsum(income incomea) if group==1
bro group income incomea manuel4, nol
```

Note: A missing value "." will be treated as 0 in a sum and will not be considered when estimating an average.

- replace: changes the contents of an existing variable.
  replace oldvar = exp [if exp] [in range]

```
gen manuel5=income
replace manuel5=. if factor==.
bro income manuel5 factor
```

- recode: changes the values of an existing variable according to the rules specified.
  recode varlist (rule) [(rule) ...] [, generate(newvar)]

```
gen manuel6=marital
recode 3=2 4=3
bro marital manuel6
```

- drop: eliminates variables or observations from the data in memory.
  drop varlist
  drop if exp [in range]

```
drop manuel-manuel6
```

- keep: works the same as drop except that you specify the variables or observations to be kept rather than those to be deleted.
  keep varlist
  keep if exp [in range]

```
keep if group==1
```

- expand: replaces each observation in the current dataset with n copies of the observation
  expand [=]exp [if exp] [in range] [, generate(newvar)]

```
expand 2 if city==2
gsort codjoven
bro if city==1
```

7. Merge, append and delete databases

- merge: joins corresponding observations from the dataset currently in memory (called the master dataset) with those from the Stata-format dataset stored as filename (called the using dataset) into single observations.

  merge [varlist] using filename [, update]

  The variable _merge is added to the data containing:
  _merge =1 obs. from master data
  _merge =2 obs. from using data
  _merge =3 obs. from both, master agrees with using
  _merge =4 obs. from both, missing in master updated
  _merge =5 obs. from both, master disagrees with using

```
use example2.dta, clear
bro
keep codjoven city course sons
sort codjoven city
save temporal.dta, replace

use example1.dta, clear
sort codjoven city
merge codjoven city using temporal.dta
tab _merge
drop _merge
bro
```

- append: appends a Stata-format dataset stored on disk to the end of the dataset in memory.
  append using filename

```
use example1.dta, clear
append using example3.dta
bro
```

- erase: erases files stored on disk.

```
erase temporal.dta
```

8. Descriptive Statistics

- do: execute the commands stored in filename just as if they were entered from the keyboard.

```
do stata.do
```

- log: allows you to make a full record of your Stata session. A log is a file containing what you type and Stata's output.
  log using filename [, replace append]
  log close

  log using example1.log, replace
  log close

- summarize: calculates and displays a variety of univariate summary statistics.
  summarize [varlist] [if exp] [in range] [weight] [, options]
  gsort group
  by group: sum income incomea

- tabulate: produces one- and two-way tables of frequency counts along with various measures of association.
  [by varlist:] tabulate varname [weight] [if exp] [in range] [, nolabel]
  [by varlist:] tabulate varname1 varname2 [weight] [if exp] [in range] [, column row nolabel]
  tab marital, g(manuel)
  tab manuel1

- table: calculates and displays tables of statistics.
  table rowvar [colvar [supercolvar]] [weight] [if exp] [in range] [, contents(clist) by (superrow_varlist) col row scol format(%fmt) center]
  where clist = freq, sum varname, mean varname, median varname, sd varname, max varname, min varname

  table city if group==1, row
  table city [pw=factor] if group==1, row
  table city group, c(mean age sd age) col row f(%9.1f)

- correlate: displays the correlation or covariance matrix for varlist. Observations are excluded from the calculation due to missing values on a casewise basis.
  correlate [varlist] [if exp] [in range]
  pwcorr [varlist] [if exp] [in range][, obs sig]

  sum factor age income
  corr factor age income
  pwcorr factor age income, obs

- collapse: converts the data in memory into a dataset of means, sums, medians, etc.
  collapse [(stat) varlist [[(stat)] ...] [weight] [if exp] [in range] [, by(varlist)]
  where stat = mean, sum, median, sd, max, min

  collapse (mean) income age (median) income2=income, by(city)

- ttest: performs one-sample, two-sample, and paired t tests on the equality of means.
  Hypothesis:
  - If the mean of one variable is equal to a value
  ttest varname = # [if exp] [in range] [,level(#)]
  - If the mean of two variables are equal
  ttest varname1 = varname2 [if exp] [in range] [,level(#)]
  - If the mean of one variable is different between groups
  ttest varname [if exp] [in range], by(groupvar) [level(#)]

```
ttest income=180, level(99)
ttest income=incomea
ttest income, by(male)
```

- graph: draws scatterplots, line plots, etc.
  [graph] twoway plot [if exp] [in range] [, twoway_options]

plot type	
scatter	scatterplot
line	lineplot
connected	connected-line plot

```
gen lincome=ln(income)
gen lincomea=ln(incomea)
twoway scatter lincome lincomea, t1("Graph 1")
```

- pctile: creates a new variable containing the percentiles of another variable.
  pctile newvar = exp [weight] [if exp] [in range] [, nquantiles(#)]

```
pctile pcut = income, n(5)
bro pcut
```

- xtile: creates a new variable that categorizes another variable by its quantiles. You can also specify the cutpoints from another variable.
  xtile newvar= exp [weight] [if exp] [in range][, nquantiles(#) | cutpoints(varname)]

```
xtile quantil = income, n(5)
xtile quantil2 = income, c(pcut)
tab quantil
```

- foreach: loop over items; sets local macro *lname* to each element of the list and executes the commands enclosed in braces.
  foreach lname {in|of listtype} list {
  commands referring to `lname'
  }

where listtype is:
any_list
local
global
varlist
newlist
numlist

```
foreach var of varlist age incomea {
        egen m`var'= mean (`var')
}
```

- forvalues: loop over consecutive values; repeatedly sets local macro *lname* to each element of *range* and executes the commands enclosed in braces.
  forvalues lname = range {
  commands referring to `lname'
  }

```
forvalues i = 1/5 {
        egen mq`i'= mean(income) if quantil==`i'
}
```

9. Matrices

- set matsize: sets the maximum number of rows and columns of a matrix (also the maximum number of variables in a regression). Upper limit is 11,000.

```
set matsize 800
```

- mkmat: stores the variables listed in varlist in _N x 1 column vectors of the same name. Optionally, if matrix() is specified, they can be stored as a single _N x k matrix.
  mkmat varlist [if exp] [in range] [, matrix(matname)]

```
mkmat income age
mat list income
mkmat age male marital, mat(X)
mat list X
```

- svmat: takes a matrix and stores its columns as new variables.
  svmat [type] matname [, names(varnames)]

```
svmat double X, n(x)
sum age x1
```

10. Linear regression

- regress: estimates a linear regression $Y = \beta X + \varepsilon$.

[by varlist:] regress depvar [varlist] [weight] [if exp] [in range] [, level(#) noconstant robust]

options:

level(#)	specifies the confidence level, in percent, for confidence intervals of the coefficients
noconstant	suppresses the constant term (intercept) in the regression
robust	specifies that the Huber/White/sandwich estimator of variance is to be used in place of the traditional calculation

This command generates automatically:

e(b)	row vector of the estimated coefficients
e(V)	variance-covariance matrix of the estimated coefficients

Special function:

e(sample)	only considers those observations included in the estimation

```
use example4.dta, clear

log using example1.log, append
gen lincome=ln(income)
tab size, g(size)
reg lincome age male school size2 size3 size4 size5
sum age male school size2 size3 size4 size5 if e(sample)
reg lincome age male school size2 size3 size4 size5, r
log close

reg lincome age male school size2 size3 size4 size5
mat betas = e(b)'
mat list betas

set more off
reg lincome age male school size2 size3 size4 size5 if male==1
gen age2=age if e(sample)
gsort age2
gen order=_n if e(sample)
mat beta = J(183,1,1)
for num 1/183: reg lincome school age size2 size3 size4 size5 if male==1 & order>=X
 & order <=X+999 \ mat betaX=e(b) \ matrix beta[X,1]=betaX[1,1]
mat list beta
```

- lincom: computes point estimates, standard errors, t and Z statistics, p-values, and confidence intervals for linear combinations of coefficients after any estimation.
  lincom exp [, level(#)]

```
reg lincome age male school size2 size3 size4 size5
lincom size3 - size2
reg lincome age male school size1 size3 size4 size5
```

- predict: calculates predictions, residuals, and influence statistics after estimation.
  predict [type] newvarname [if exp] [in range] [, statistic]

statistic:	
xb	linear prediction (default)
residuals	residual
stdp	standard error of linear prediction

```
reg lincome age male school size2 size3 size4 size5
predict lincomep
predict lincomep2 if e(sample)
predict resid, r
sum lincomep lincomep2 resid
bro lincome lincomep2 resid if e(sample)

gen lincome2=lincome if e(sample)
sort lincome2
gen x=_n if e(sample)
twoway line graph lincome2 lincomep2 x


reg lincome age male school size2 size3 size4 size5
collapse lincome age male school size2 size3 size4 size5 if e(sample)
predict total
replace male=0
predict females
replace male=1
predict males
sum total females males
```

- stepwise: performs stepwise estimation.
  sw estimation_command depvar [varlist] [weight] [if exp] [in range] , {pr(#)}

pr(#)	specifies the significance level for removal from the model; terms with p>=pr() are eligible for removal

```
use example4.dta, clear
gen lincome=ln(income)
tab size, g(size)
tab actecon, g(act)
sw reg ling edad sexo school tam2-tam5 act2-act11 sindic, pr(0.05)
```

- outreg: formats regression output as it is presented in most documents: t-statistics or standard errors in parentheses under each coefficient, asterisks indicating coefficients statistically different from zero, and summary statistics like R-squared at the bottom. The formatted output is written to a tab- or comma-separated ASCII file, which can then be loaded into word processing or spreadsheet programs to be converted to a table.
  outreg [varlist] using filename [, se bdec(#) tdec(#) noaster nocons nonobs adjr2 nonotes append replace]

se	specifies that standard errors rather than t-statistics are reported
bdec(#)	specifies the number of decimal places reported for coefficient estimates
tdec(#)	specifies the number of decimal places reported for t-statistics
noaster	specifies that no asterisks denoting 1% and 5% confidence intervals be reported
nocons	specifies that the intercept (constant) coefficient estimate not be reported
nonobs	specifies that the number of observations in the estimation not be reported
adjr2	specifies that the adjusted R-squared be reported rather than the regular R-squared
nonotes	specifies that notes explaining the t-statistics (or standard errors) and asterisks not be included
append	specifies that new estimation output be appended to an existing output file
replace	specifies that it is okay to replace filename if it already exists

Note: to install this command: go to the Help icon; search for "outreg", click on sg97; click on install.

```
reg lincome age male school size2 size3 size4 size5
outreg using out1.out, b(4) nol replace
```

Basic notes on Stata language

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

Operators:

Arithmetic	Logic	Relations
+ Addition	~ No	> Greater than
- Subtraction	| Or	< Less than
* Multiplication	& And	>= Greater or equal
/ Division		<= Less or equal
^ Power		== Equal
		!= Not equal

Mathematical Functions:

abs(x)	- absolute value
sin(x)	- sine
cos(x)	- cosine
tan(x)	- tangent
exp(x)	- exponential
ln(x)	- natural logarithm
log(x)	- natural logarithm
sqrt(x)	- square root

Random numbers:
uniform() returns uniformly distributed pseudo-random numbers on the interval

Other functions:
real() converts a string into a numeric value or returns a missing value