

Pre-Semester Course in Statistics

Introduction to STATA

Goethe University Frankfurt

Nan Hu

Outline

- Stata's interface
- Do file
- Using and describing data
- Variable related commands
- Graphs
- T test

1. Stata's interface

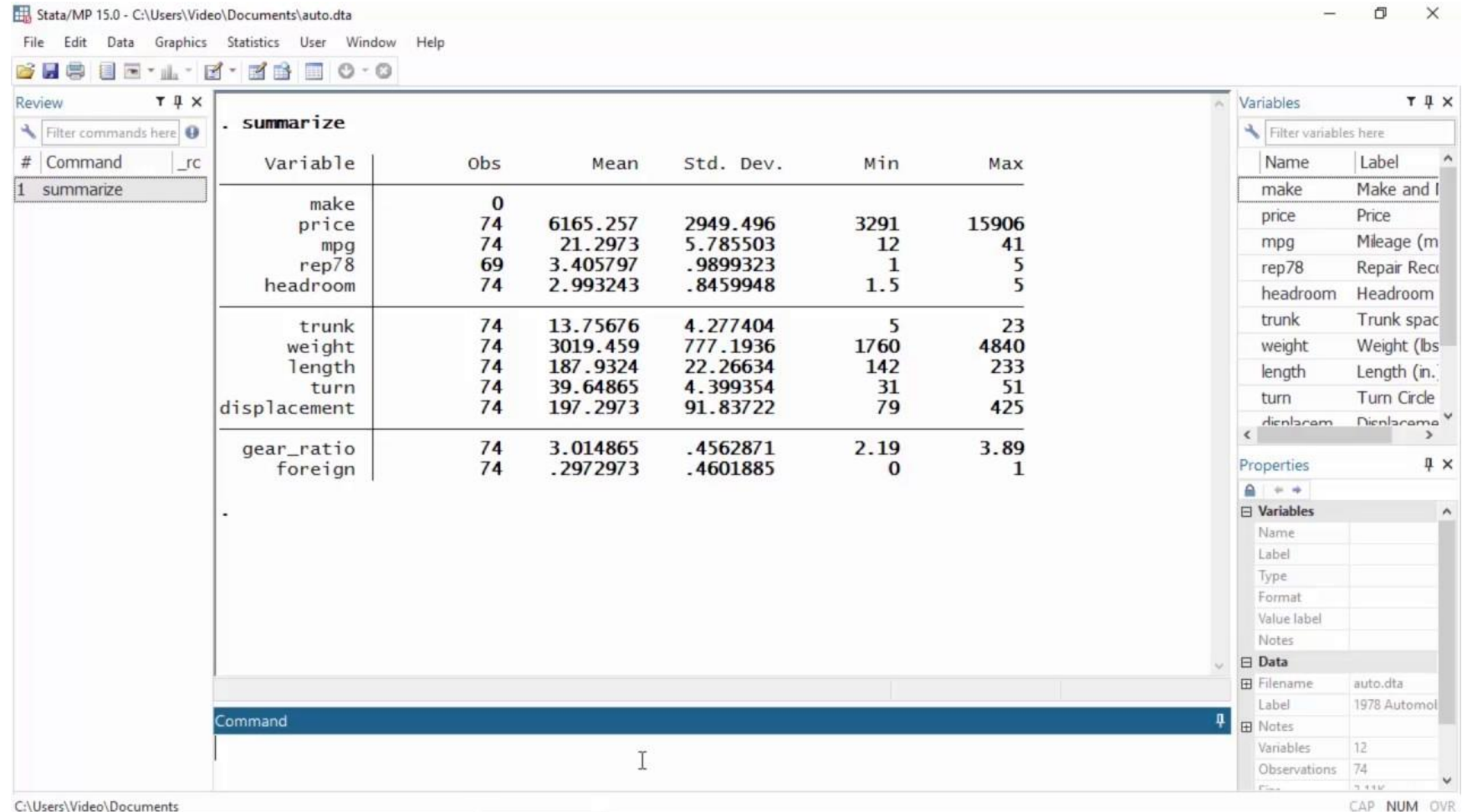
Four windows:

- Results window
- Command window
- Review window
- Variables Window

Two bars:

Menu Bar

Icon bar



The screenshot displays the Stata 15.0 interface with the following components:

- Menu Bar:** File, Edit, Data, Graphics, Statistics, User, Window, Help.
- Icon Bar:** A row of icons for file operations and data management.
- Review Window:** Shows the command `. summarize` and a list of commands.
- Results Window:** Displays the output of the `. summarize` command as a table.
- Command Window:** A text area at the bottom for entering commands.
- Variables Window:** A list of variables with their labels and properties.
- Properties Window:** A panel on the right showing details for the selected variable.

Variable	Obs	Mean	Std. Dev.	Min	Max
make	0				
price	74	6165.257	2949.496	3291	15906
mpg	74	21.2973	5.785503	12	41
rep78	69	3.405797	.9899323	1	5
headroom	74	2.993243	.8459948	1.5	5
trunk	74	13.75676	4.277404	5	23
weight	74	3019.459	777.1936	1760	4840
length	74	187.9324	22.26634	142	233
turn	74	39.64865	4.399354	31	51
displacement	74	197.2973	91.83722	79	425
gear_ratio	74	3.014865	.4562871	2.19	3.89
foreign	74	.2972973	.4601885	0	1

2. Do file

Create folders to save different files:

- rawdata: raw data
- tempdata: temporary data
- dofile :do file
- logfile: log
- output: export results

2. Do file (example)

Why do-file? It's the programing editor for Stata.

Example:

```
/*This program is written for the econometrics class of spring 2013. */  
clear all  
set mem 100m  
set maxvar 5000  
set more off  
cd C:\ado\personal  
capture log close  
log using $logfile\introduction.log, replace  
.....  
capture log close
```

2. Do file (comment statement)

- At the beginning of the do file, you can explain the name, purpose, and purpose of the program. These are comments.
- Ways to add comments:
- Add a comment to the front: `**`
- Add a comment in the middle of a command: `/*comment content*/`
- Add a comment after the command line: `// comment content`
- Turn a portion of the content into a comment:
`/* Commented out statement*/`

2. Do file : the basic commands

The following are the commonly used commands at the beginning of a dofile:

- `clear all` //Clear the information that was originally left in STATA
- `set maxvar 3000` //Set the maximum number of variables allowed
- `set mem 100m` //Set the size of the memory used
- `set more off` // When reporting results, the default setting is one page

One page display; set more off allows all results to be reported at once

2. Do file : setting the path

- It's a good programming habit to set the path at the beginning of the program.
If the program is run for someone else, he or she just need to modify the path at the beginning.

- Use global command to set the path. For example:

```
global rawdata " C:\ado\personal \rawdata"
```

In this way, when we use \$rawdata, it is equivalent to use the path.

```
use $rawdata\finance.dta, clear
```

- Other examples:

```
gl dofile "C:\ado\personal\dofile"
```

```
gl logfile "C:\ado\personal\logfile"
```

```
gl output "C:\ado\personal\output"
```


2. Do file : the log

- We can to create a log for the program. The commands related to the log are:

Log using `$logfile\introduction.log, replace`

- This command starts with “log” generate a log file called *introduction.log* and store it in C:\ado\personal\logfile.

When this command is executed for the first time, STATA will generate Introduction.log; when this command is executed the second time, since the introduction.log file already exists, STATA cannot generate a file with the same name and will report error.

In this case, use", `replace`“ to overwrite old files with new files. Use", `append`", to attach the new file to the back of the old file.

`capture log close *`

If there is a log file open, it will be turned off; if not, just skip it.

3. Using and describing data (open data)

STATA can use a variety of data, but the commands are different.

- Read csv:

```
import delimited $rawdata\finance.csv, clear
```

- Read xlsx:

```
import excel " $rawdata \finance.xls", sheet("Sheet1") clear
```

- Read dta (Stata format of data)

```
use $rawdata\finance.dta, clear
```

**The option “,clear” means that if there is other data in the database before.

3. Using and describing data

Urban-rural income gap and local government fiscal expenditure

- The file `finance.dta` is the income of 31 provincial units in China in 2008 Financial related data. The source of information is the China Statistical Yearbook.
- The variable `Inequality` measures the proportion of urban and rural income (unit: %) The minimum, mean and maximum values are 232, 312, 427, respectively.
- Variable `Finance` = $\text{fiscal expenditure} / \text{GDP} * 100\%$. Its minimum value, The mean and maximum values are 8.7, 21.6, and 96.15, respectively. This variable reflects local government behavior.
- $N=31$.

3. Using and describing data : describe

- Commonly used commands for describing data are *describe*, *list*, *summarize*, *tabulate* etc.

- First you need to learn to use the help command. It's very useful when we don't know the specifics of a command.

You can use the help plus command name to query the command.

`help describe`

- When using describe, you can add nothing behind to describe the whole dataset

`describe`

Or to describe only a few variables. such as

`describe gdp`

3. Using and describing data : summarize

■ The Summarize command gives the most basic description of the most data variables, too

It is the first step we use to see if there is a problem with the data. such as:

`summarize finance`

- This command gives us the number of observations, mean, minimum, maximum Value, etc.

■ When we want to get more information, we need to add the suffix ",detail"

Such as:

`Summarize finance, detail`

■ This command can be combined with data conditions such as:

`summarize finance if region == 1`

`summarize finance if region != 1`

3. Using and describing data : list

- **list**: lists the data. Similar to edit but shown in the command window

list gdp if region==1

- * Display gdp data with region 1

list gdp if region==1 & gdp>10000

- * Display gdp data with region 1 and gdp > 10000

list gdp region if region==1 | region==2

- * Display data with region 1 or 2

3. Using and describing data : tabulate

■ `tabulate` will report the frequency of each unique values for a variable.

- The command will report that how many provinces are in the eastern, central, and western regions.

`tabulate region`

■ The `tabulate` command also reports the frequency of the combination of two variables.

- Assume that the provinces are divided into two groups: high income and low income:

`gen highincome= 1 if GDP> a specific value`

`replace highincome=0 if GDP<specific value`

`tabulate region highincome`

- Get the interaction analysis table of region and highincome

3. Using and describing data : table

■ table: provide data table

`table region` // region frequency table

`table region, contents(n ineq)`

- frequency table with no missing values with ineq

`table region, c(n ineq mean ineq)`

- Classification shows other statistics for ineq

4. Variable related commands: gen and egen

■ gen

■ egen: often used with functions

- `gen ineq1=ineq*100` //Generate 100 times ineq
- `gen lgdp=log(gdp)` // Generate log (gdp)
- `egen maxgdp=max(gdp)` //maximum
- `egen mgdp=mean(gdp)` //average

■ The main difference between gen egen is reflected in the sum function:

- `gen cum_gdp=sum(gdp)` //added gdp
- `egen tot_gdp=sum(gdp)` //summed gdp
- `list cum_gdp tot_gdp`

4. Variable related commands: rename

- In addition to generating new variables, there are quite a few common commands such as rename
 - rename variable
 - `rename inequality ineq`

4. Variable related commands: label

- `label var` tags variables
- `label define` tags the value of a variable
- `label value` for this set of label descriptions

`label var gdp "province GDP per capita"`

- Add the label "province GDP per capita" to the variable gdp

`label define lregion 1 "east" 2 "central" 3 "west"`

- Label the region variable value, note that the value of the variable is still 1, 2, 3

`label values region lregion`

- The display of the variable value changes to east, central, west

4. Variable related commands: count

- count: How many observations are reported in the report?

`count if wage>30000`

- Report the number of observations with wage>3000. Note: missing numbers " ." are counted as infinity.

`count if wage>30000 & wage~=.`

- Report the number of observations if wage>3000 and wage is not missing. Note: != means not equal

4. Variable related commands: order

■ `order`: re-arrange the location of variables in dataset. The variables in `order` will be moved to the beginning columns.

```
gen ineq1=ineq*100
```

```
gen cum_ineq=sum(ineq) //added gdp
```

```
egen tot_ineq=sum(ineq) //summed gdp
```

```
order ineq ineq1 cum_ineq tot_ineq
```

- To compare the values of the four variables, put them together.

4. Variable related commands: sort, by

- sort: rearranges the observations in ascending order
- by: classification for operation

sort region // sort regions in order of 1, 2, 3

by region: egen mn_ineq=mean(ineq)

- Generate the average values within each region and then sort them

by region, sort : egen mn_ineq1=mean(ineq)

The result is the same with the two lines of commands above.

4. Variable related commands: keep, drop

- drop: delete variables or observations
- keep: keep variables or observations

drop wage // delete the variable wage

drop if wage<10000

// Delete the observation value of wage<10000

- keep gdp ineq // keep only the variable gdp ineq
- keep if gdp>10000 //retain observations of gdp>10000

4. Variable related commands: preserve, restore

■ `preserve...restore`: very useful when we want to change the data structure temporarily to obtain certain results and restore back to original data.

The command does not cause any damage to the data.

`preserve`

`drop if province=="Tibet"`

`reg ineq1 finance`

`restore`

- The command doesn't destroy the original database, and also avoid to generate cached data files.

4. Variable related commands: append and merge

- append combines two data files vertically (add new obs)
- merge combines two data files horizontally (add new variables)

4. Variable related command: replace

- Assume that the provinces are divided into two groups: high income and low income:

`gen highincome= 1 if GDP> 10000`

- In the command above, when generating highincome, we have only told Stata the value for highincome if gdp meets the above conditionsof ; for non-conforming situation, the highincome value is assigned the default ".".

To change this state, we need to use the command ", `replace`"

`replace highincome=0 if GDP<10000`

Summary: Common commands

- describe: describe the data
- edit: Data editing with the data editor
- list: lists the data. Similar to edit but shown in the command window
- summarize: the number of observations, the average, the standard of the variable

Difference, minimum and maximum. Abbreviation sum

- tabulate: joint list
- table: data list
- scalar: define scalar data
- display: display the calculation result, abbreviation di

Summary: Common commands

- rename: variable rename
- label var: tag the variable
- label define: tag the variable value
- label value: for this set of label descriptions
- count: observed value
- sort: rearranges the observations in ascending order
- order: variable ordering
- gen and egen: generate new variables
- replace: variable value replacement

Summary: Common commands

- drop: delete variables or observations
- keep: keep variables or observations
- append: longitudinally stitching data with the same result
(observation value stitching)
- merge: splicing two data files horizontally
- save: save data
- clear: clear memory

5. Graphs

■ Univariate drawing

- graph
- histogram
- kdensity

■ Two-variable drawing

- graph twoway (scatter/line...)

We will replicate all graphs in lecture slides in *introduction.do*.

6. T test

- We will replicate the t test results for the example *Silver content of Byzantine coins*

use "coins.dta", replace

tabstat x1 x4, statistics(mean median sd variance)

ztest x1=6.5

ttest x1=x4, unpaired