

# Introduction to Social Data Analytics

## Class 7

Arushi Kaushik

arkaushi@ucsd.edu

# Today: Stata

By the end of today's lecture, you should be able to:

- Create, edit, and save Do-files and log files
- Find more information on commands so you can teach yourself
- Load data and conduct some of the most common Stata commands

# Open your Do-file from the pre class 7 exercise

So far you should have three commands listed:

- `clear`
- `import excel`
- `summarize`

# Open your Do-file from the pre class 7 exercise

So far you should have three commands listed:

- `clear`
- `import excel`
- `summarize`

Add the new commands that we go over in class to your Do-file.

# But first: why do we use Do-files?

...instead of typing everything in the Command window?

# But first: why do we use Do-files?

...instead of typing everything in the Command window?

Reproducibility.

# But first: why do we use Do-files?

...instead of typing everything in the Command window?

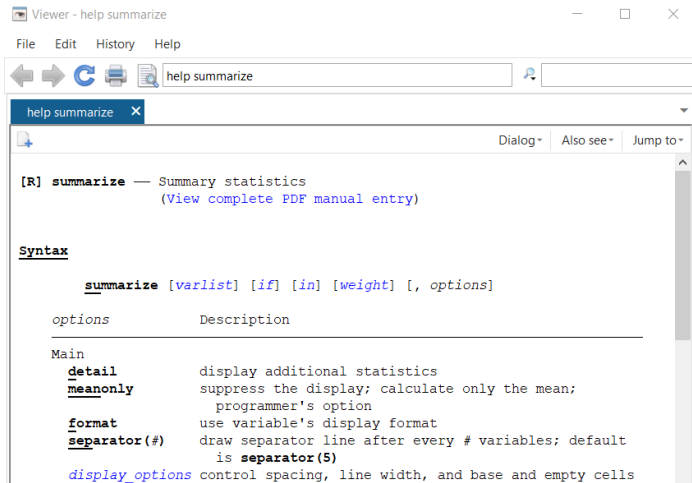
## Reproducibility.

- Anyone anywhere with the same data and Do-file can produce the same results.
- Big time savings when repeating analysis of similar data.

# Command: help

Syntax: `help command`

e.g. `help summarize`



The screenshot shows the RStudio Viewer window titled "Viewer - help summarize". The window contains the help text for the `summarize` function. The text includes the function's purpose, its syntax, and a table of options.

**[R] summarize** — Summary statistics  
([View complete PDF manual entry](#))

**Syntax**

`summarize` [*varlist*] [*if*] [*in*] [*weight*] [, *options*]

<i>options</i>	Description
Main	
<code><u>detail</u></code>	display additional statistics
<code><u>meanonly</u></code>	suppress the display; calculate only the mean; programmer's option
<code><u>format</u></code>	use variable's display format
<code><u>separator</u>(#)</code>	draw separator line after every # variables; default is <code>separator(5)</code>
<code><u>display_options</u></code>	control spacing, line width, and base and empty cells



# Command: `cd` (change directory)

Syntax: `cd path`

e.g. `cd "C:\User\Owner\Documents\Econ 5\Class 7"`

# Command: `cd` (change directory)

Syntax: `cd path`

e.g. `cd "C:\User\Owner\Documents\Econ 5\Class 7"`

- Put this at the beginning of your Do-file
- Change your directory to wherever you want to access and save files

# Command: log

Syntax: log using *filename*, [*options*]

e.g. log using logfile.txt, text replace

# Command: log

Syntax: log using *filename*, [*options*]

e.g. log using logfile.txt, text replace

- The log file will record your session - everything you type and the output

# Command: log

Syntax: log using *filename*, [*options*]

e.g. log using logfile.txt, text replace

- The log file will record your session - everything you type and the output
- This is useful for:
  - Tracking what you do
  - Viewing particularly long output
  - Sending output to collaborators

# Command: log

Syntax: `log` using *filename*, [*options*]

e.g. `log` using `logfile.txt`, `text replace`

- The log file will record your session - everything you type and the output
- This is useful for:
  - Tracking what you do
  - Viewing particularly long output
  - Sending output to collaborators
- At the end of the Do-file, you can stop logging by adding the command `log close`

## Commands: describe, summarize

These commands provide information about variables.

- desc: provides a summary of all variables in memory
- sum: produces summary statistics of all variables in memory

## Commands: describe, summarize

These commands provide information about variables.

- desc: provides a summary of all variables in memory
- sum: produces summary statistics of all variables in memory

You can use these commands for specific variables instead of every variable in memory. Try:

- desc id
- sum class adult
- sum id - adult survive



## Commands: describe, summarize

These commands provide information about variables.

- desc: provides a summary of all variables in memory
- sum: produces summary statistics of all variables in memory

You can use these commands for specific variables instead of every variable in memory. Try:

- desc id
- sum class adult
- sum id - adult survive

What percent of Titanic passengers survived?

## Command: tabulate

Syntax: `tab varlist`

This command determines the frequency that a variable takes each value.

## Command: tabulate

Syntax: `tab varlist`

This command determines the frequency that a variable takes each value. Which of the following variable types is `tab` useful for?

- numerical
- categorical
- logical

## Command: tabulate

Syntax: `tab varlist`

This command determines the frequency that a variable takes each value. Which of the following variable types is `tab` useful for?

- numerical
- categorical
- logical

Try the following:

- `tab male`
- `tab male survive`
- `tab male survive, row`

## Command: tabulate

Syntax: `tab varlist`

This command determines the frequency that a variable takes each value. Which of the following variable types is `tab` useful for?

- numerical
- categorical
- logical

Try the following:

- `tab male`
- `tab male survive`
- `tab male survive, row`

Were men or women more likely to survive the sinking of the Titanic?

# Command: `set more off`

Suppose you run a command that produces very long output, like `tab id` (try it!). This command enables you to view all output without having to press enter repeatedly.

## Command: `set more off`

Suppose you run a command that produces very long output, like `tab id` (try it!). This command enables you to view all output without having to press enter repeatedly.

You can make this change permanent with `set more off, permanent`

## Commands: generate, replace

Syntax: `gen newvar = exp`

Syntax: `replace oldvar = exp`

These commands are used to create variables.



# Commands: generate, replace

Syntax: `gen newvar = exp`

Syntax: `replace oldvar = exp`

These commands are used to create variables. Try it:

- `gen notmale = 0`
- `replace notmale = 1 if male == 0`

## Commands: generate, replace

Syntax: `gen newvar = exp`

Syntax: `replace oldvar = exp`

These commands are used to create variables. Try it:

- `gen notmale = 0`
- `replace notmale = 1 if male == 0`

Notice the single versus double equals signs.

- `=` assigns values to a variable
- `==` tests if a variable takes certain values (a logical test)

Commands: rename, label variable

Syntax: `rename old_varname new_varname`

## Commands: rename, label variable

Syntax: `rename old_varname new_varname`

Rename the variable `notmale` as `female`:

- `rename notmale female`

## Commands: rename, label variable

Syntax: `rename old_varname new_varname`

Rename the variable `notmale` as `female`:

- `rename notmale female`

Labeling variables is useful when the name of the variable doesn't fully communicate what the variable represents.

Syntax: `label variable varname "label"`

## Commands: rename, label variable

Syntax: `rename old_varname new_varname`

Rename the variable `notmale` as `female`:

- `rename notmale female`

Labeling variables is useful when the name of the variable doesn't fully communicate what the variable represents.

Syntax: `label variable varname "label"`

Try it:

- `label var female "0 if male, 1 otherwise"`

## Commands: rename, label variable

Syntax: `rename old_varname new_varname`

Rename the variable `notmale` as `female`:

- `rename notmale female`

Labeling variables is useful when the name of the variable doesn't fully communicate what the variable represents.

Syntax: `label variable varname "label"`

Try it:

- `label var female "0 if male, 1 otherwise"`

## Commands: save, use

Stata has its own data file type (.dta). We save and load data in this format with these commands.



## Commands: save, use

Stata has its own data file type (.dta). We save and load data in this format with these commands. Try:

- `save titanic, replace`
- `use titanic, clear`

# Commands: save, use

Stata has its own data file type (.dta). We save and load data in this format with these commands. Try:

- `save titanic, replace`
- `use titanic, clear`

`replace` is necessary to overwrite existing files.

## Commands: save, use

Stata has its own data file type (.dta). We save and load data in this format with these commands. Try:

- `save titanic, replace`
- `use titanic, clear`

`replace` is necessary to overwrite existing files.

`clear` is necessary to load new data when existing data is present.

# Commands introduced in this class:

- clear
- import excel
- help
- cd
- log using *filename.txt*,  
text replace
- log close
- describe
- summarize
- tabulate
- set more off
- generate
- replace
- rename
- label variable
- save *filename*, replace
- use *filename*, clear