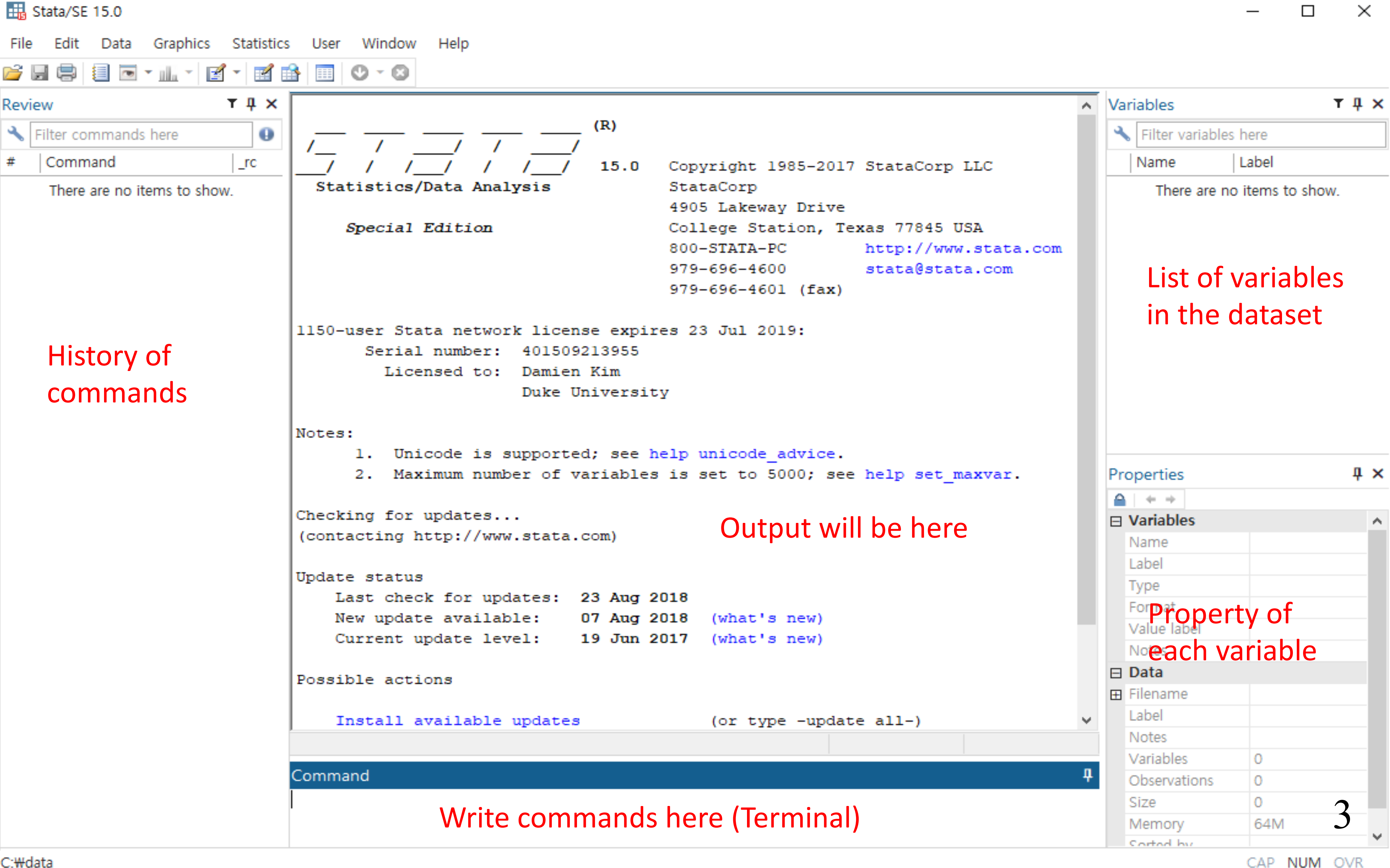
Introduction to STATA

Richard Lombardo
September 2022

Econ 204 STATA Introduction Document:
https://ipl.econ.duke.edu/dthomas/ec204/ho-stata.pdf

Outline

- Basics
- Open a data file and set working directory
- Do file
- Descriptive commands
- Relationship between variables
- Manipulating the data
- Logical statements
- Future references



There are no items to show.

History of commands

```
  ___  ____  ____  ____  ____ (R)
 /__    /   ____/   /   ____/
___/   /   /___/   /   /___/   15.0   Copyright 1985-2017 StataCorp LLC
  Statistics/Data Analysis            StataCorp
                                      4905 Lakeway Drive
     Special Edition                  College Station, Texas 77845 USA
                                      800-STATA-PC        http://www.stata.com
                                      979-696-4600        stata@stata.com
                                      979-696-4601 (fax)

1150-user Stata network license expires 23 Jul 2019:
       Serial number:  401509213955
         Licensed to:  Damien Kim
                       Duke University

Notes:
      1.  Unicode is supported; see help unicode_advice.
      2.  Maximum number of variables is set to 5000; see help set_maxvar.

Checking for updates...
(contacting http://www.stata.com)

Update status
    Last check for updates:  23 Aug 2018
    New update available:    07 Aug 2018  (what's new)
    Current update level:    19 Jun 2017  (what's new)

Possible actions

    Install available updates             (or type -update all-)
```

Output will be here

Write commands here (Terminal)

There are no items to show.

List of variables in the dataset

Variables
Name
Label
Type
Format
Value label
Notes

Data
Filename
Label
Notes
Variables 0
Observations 0
Size 0
Memory 64M
Sorted by

Property of each variable

C:₩data

CAP NUM OVR

Open a data file

- STATA datasets are saved as “.dta” files

- STATA lets you analyze and manipulate data, so we first need to load in data!

- To do this, you need to tell STATA where to look in your computer’s files for the dataset
    - “Setting your working directory”

Set working directory

- Figure out which folder your data file is in
  - Let’s try using “expend.dta.” I saved the file into my “Downloads” folder

- Type into the STATA terminal “cd FILEPATH HERE” to tell STATA this is the folder you want to work in
  - Windows: “cd c:\Downloads” or “cd Downloads”
  - Mac: “cd /Users/richa/Downloads” or “cd Downloads”

- You need to set working directory every time you read in data

Open a data file (cont.)

- After setting our working directory to the folder that the data is in, we now tell STATA the name of the dataset to use

Command: use *filename.dta*
Example: use expend.dta
Purpose: Loads in the dataset *expend.dta*

- If we already have a dataset loaded into STATA, then you type instead: use expend.dta, clear
  - Warning: This does not save the changes made to the old dataset
  - We could have instead used "clear" or "clear all" to close any open datasets in STATA

Save changes to a data file

- Let’s try saving expend.dta

Command: save *filename.dta*
Example: save expend.dta
Purpose: Saves the dataset *expend.dta*

- This code doesn’t work! STATA says this file already exists. We have 2 options:
  - Choose a different filename: save expend2.dta
  - Replace the existing file: save expend.dta, replace

3 main types of files

- Data file (STATA datasets like expend.dta)

- Do file
  - List of “commands”, or the specific code to tell STATA to do something to your data
  - Allows you to keep track of what you’ve done

- Log file
  - Documents your results
  - Outputs the result of the commands in Do-File

What is a do file??

- Do-file is a text file containing commands
  - Reminder: Commands are specific code telling STATA to do something to our data
    - “cd Downloads” or “use expend.dta”

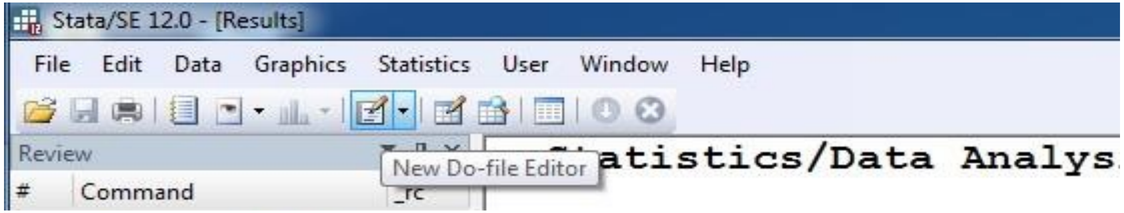
- If we need to do similar analyses over and over, we want to keep a record of what we did

- Also helps you to collaborate with others

- Easier to debug when we run into errors

Basic do file

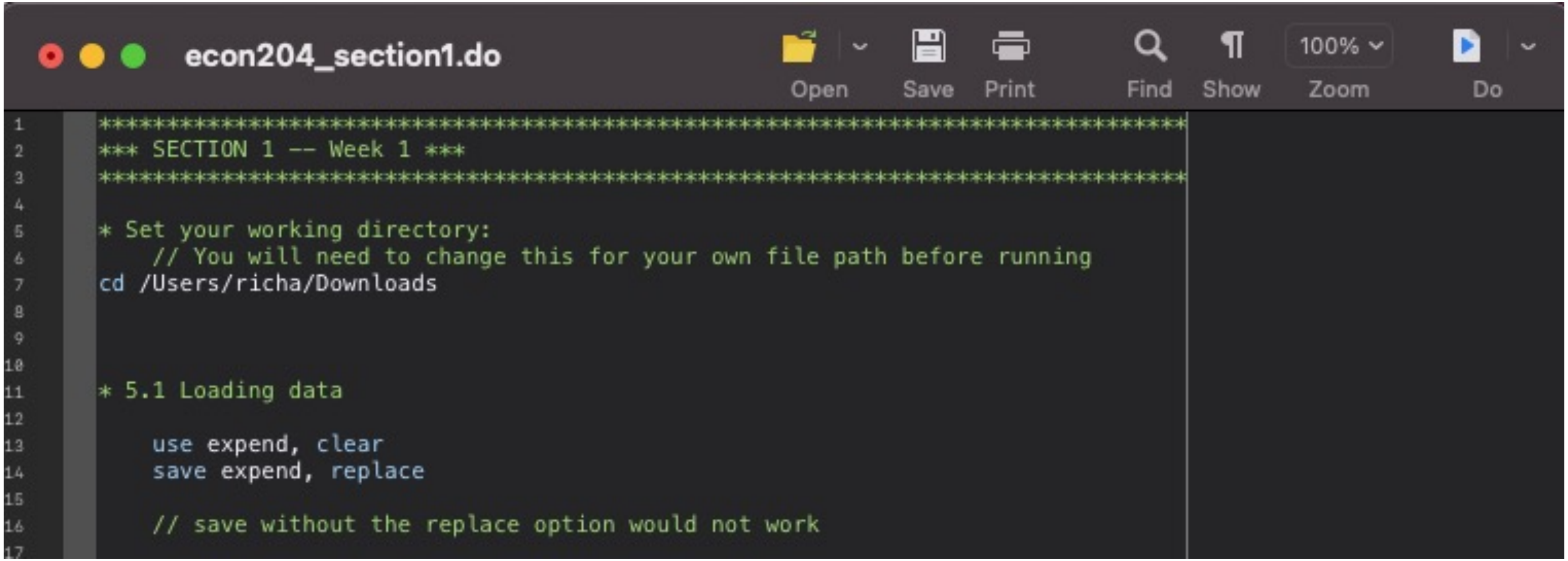
- Step 1: Create a do-file



- Step 2: Change working directory to where your data file is
  - I put this line at the top of all of my do-files, since you need to set working directory first

- Step 3: Save the do-file with a name you like

- Step 4: Edit the do file



```
********************************************************************************
*** SECTION 1 -- Week 1 ***
********************************************************************************

* Set your working directory:
    // You will need to change this for your own file path before running
cd /Users/richa/Downloads



* 5.1 Loading data

    use expend, clear
    save expend, replace

    // save without the replace option would not work
```

Descriptive commands

- Let’s begin looking at what is in *expend.dta*

Command: describe *variablenames*
Example: describe tot_exp
Purpose: Shows you what the variable is and what type of variable it is (string, float [i.e. a number], etc.)

- Simply using “describe” gives this info for all variables in the dataset

Descriptive commands (cont.)

Command: list *variablenames*
Example: list tot_exp
Purpose: Lists all of the values of this variable across all observations

- Again, simply using "list" gives this info for all variables in the dataset

- For very large datasets, we often only want to see some observations:
  - To see the first 10 values of tot_exp, "list tot_exp in 1/10"
  - To see the next 10, "list tot_exp in 11/20"

Descriptive commands (cont.)

Command: sum *variablenames*

Example: sum tot_exp

Purpose: Gives some summary statistics for this variable

- Again, simply using “sum” gives this info for all variables in the dataset

- sum tot_exp only gives us some statistics (no median??)
  - To see more stats, add the detail option “sum tot_exp, detail”
  - In STATA, options are code after the comma that impacts the command

HELP: the most useful command

- What if I don’t remember what the option is called? Or, I don’t know what a command does?
- Type “help *commandname*” into the terminal (or Google) for documentation (explanation of the command)
  - “help sum”

Help (cont.)

- What if I want to do something but not sure the name of the command?

- Type “search *fill in*” with what you want to do
    - Ex. You want to know the command for regressions, so type “search regression”
    - Google also works!

Descriptive commands (cont.)

Command: tab *variablename*
Example: tab hhsize
Purpose: Gives a frequency table of this variable

- This works best for “categorical” variables or variables that don’t take on many different values
  - Difficult to interpret “tab tot_exp” since most values only appear once

- You can also easily do a two-way frequency table:
  - “tab hhsize n_child”

Relationship between variables

Command: twoway scatter *variablenames*
Example: twoway scatter food_exp tot_exp
Purpose: Produces a scatter plot of the 2 variables

- Useful to do this before looking at correlations or regressions to see if there are any major outliers (and to visually understand your data)

Relationship between variables (cont.)

Command: corr *variablenames*
Example: corr food_exp tot_exp
Purpose: Calculates the correlation between each pair of listed variables

- Simply typing "corr" estimates the correlation between all pairs of (non-string) variables in our dataset

- If you want the covariance instead, use the covariance option
  - "corr food_exp tot_exp, covariance"

Relationship between variables (cont.)

Command: reg *depvar indepvars*
Example: reg food_exp tot_exp
Purpose: Calculates regression coefficient estimates, standard errors, and useful regression stats

- The first variable listed after “reg” is the dependent variable
  - All variables after are used as covariates / independent variables
- Can do multiple covariates by simply adding more variables
  - “reg food_exp tot_exp hhsize”

Manipulating the data

Command: gen *newvar* = *function of old vars*
Example: gen lnexp = ln(tot_exp)
Purpose: Makes a new variable into the dataset

- Can't use the same name as an existing variable

- What if I want to change an existing variable instead of making a new one?
  - "replace lnexp = tot_exp"
  - Can't undo easily! That's why do-files are helpful

Logical statements

- We want a variable that is 1 if *hhsize* is >= 5 and 0 if *hhsize* is less than 5
- We do this with "if" statements! First, make the new variable *hh_gt5* equal to 1 ***if*** *hhsize* is greater than or equal to 5
    - "gen hh_gt5 = 1 if hhsize > = 5"

Logical statements (cont.)

- For observations with *hhsize* less than 5, *hh_gt5* is now missing! We want it to be 0 instead:
    - “replace hh_gt5 = 0 if hhsize < 4”

- Could also have done “replace hh_gt5 = 0 if hh_gt5 == .”
    - “.” is how STATA denotes missing numeric (non-string) values
    - To test for equality within if statements, you need to use 2 equal signs (this is how STATA knows to check if something is true or false)

Manipulating the data (cont.)

- Back to changing our current STATA dataset…

- Let’s now rename the variable *hhsize* into *householdsize*

Command: rename *oldvarname newvarname*
Example: rename hhsize householdsize
Purpose: Change variable name

Manipulating the data (cont.)

- Let’s now change the variable label for *householdsize*

Command: label variable *varname* “FILL IN LABEL”
Example: label variable householdsize "Number of people in household”
Purpose: Changes the variable label that you see when you describe the data

- It doesn’t matter if variable already did or did not have a variable!

Manipulating the data (cont.)

- Suppose we no longer want the *householdsize* variable

Command: drop *varname*

Example: drop householdsize

Purpose: Removes a variable from current dataset

Logical statements (cont.)

- Before, we only had one condition on variable ("does hh have at least 5 ppl")
- We may have multiple conditions ("does hh have at least 5 ppl AND more than 2500 in tot_exp?")
- STATA:
  - and is "&"
  - or is "|"
  - Not is "!"

Logical statements (cont.)

- We want variable to be 1 if *hhsize* is at least 5 AND *tot_exp* at least 2500. Variable is 0 if first is not true OR second is not true.

- gen hh_big_and_rich = 1 if hhsize >= 5 & tot_exp >= 2500

- replace hh_big_and_rich = 0 if hhsize < 5 | tot_exp < 2500

- Could also do:
    - replace hh_big_and_rich = 0 if hh_big_and_rich == .

Future references

- Links for STATA (https://ipl.econ.duke.edu/dthomas/ec208d/statalinks.html)
- Econometrics Academy (https://sites.google.com/site/econometricsacademy/econometrics-software/stata)
- Econometrics by simulation (http://www.econometricsbysimulation.com/)