Data Management in Stata

Hsueh-Sheng Wu

CFDR Workshop Series

January 25, 2021

BGSU
Center for Family and Demographic Research

Outline

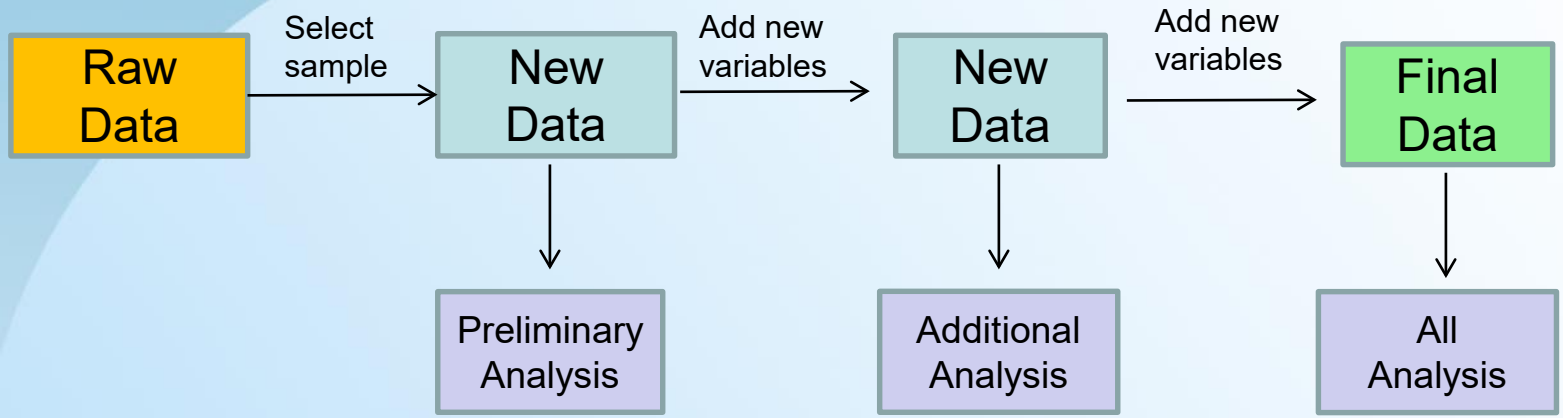
- What are data management problems?
- An introduction to Programmer's File Editor
- Common Stata commands
  - General commands
  - Managing datasets
  - Checking variables
  - Modifying existing variables
  - Generating new variables
- Organize your files
- Conclusions

What Are Data Management Problems?

Researchers may experience data management problems if they:

- Do not know how to create variables or merge files for the analyses
- Cannot find variables or files they need
- Cannot recall why and how hey constructed certain variables
- Have different command files to construct similar data files
- Have redundant data files in the folder
- Mix data construction and analysis in the same command file

What Are Data Management Problems? (Cont.)



- Separate command files for constructing data and for analyzing data. Don't mix these commands together
- This diagram illustrates only three data construction steps along with three data analysis steps
- How many files do you expect to have in the end?
  - An excel file documents the purposes and contents of all command, log, result, and data files
  - A command file for data construction
  - A log file for data construction
  - A final data file
  - Three separate command files for data analyses
  - Three log/result files from data analyses

What Are Data Management Problems? (Cont.)

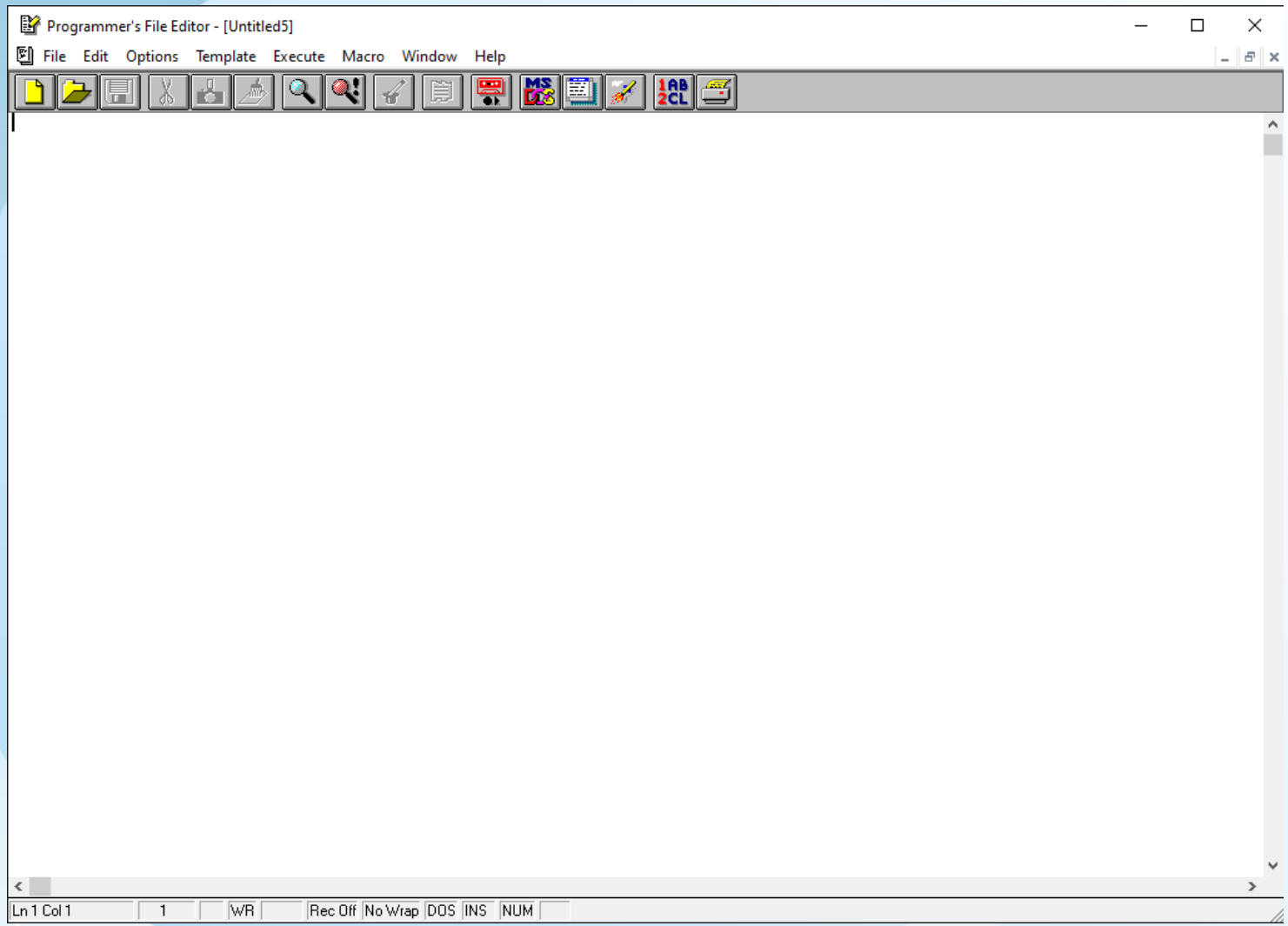
Researches obtain or create different files at each study stage, and it is very easy for them to get lost in these files.

- Planning stage:
  - Journal articles
  - Book chapters
  - Web documents on data and methods
  - Drafts of the research proposal
- Variable construction stage:
  - Stata or SAS command files
  - Log files for Stat users and log and outcome files for SAS users
- Data analysis stage:
  - Stata command and log files
- Reporting results stage:
  - Words files
  - Excel files
  - PowerPoint files

An introduction to Programmer's File Editor (PFE)

- Programmer's File Editor is freeware and can be downloaded from the link https://www.lancaster.ac.uk/~steveb/cpaap/pfe/pfefiles.htm
- PFE can be run from a jump drive, meaning that it does not need to be installed onto a computer
- While using PFE to open a Stata log file, users are given the option to quickly load the file after such the file becomes available, which significantly speeds up the process of modifying the command file and checking results
- PFE can find and replace some special characters, which can come in handy when editing Stata command files
- Researchers can also check out other code editors available for PCs or Macs: https://www.guru99.com/best-free-code-editors-windows-mac.html.

An introduction to Programmer's File Editor (PFE) (Cont.)



Common Stata Commands

When writing a do file to construct data, we can group common Stata commands into five groups :

- General commands
- Managing datasets
- Checking variables
- Modifying existing variables
- Generating new variables

General Commands

Table 1. General Stata Commands	
Key command word and special characters	**Function**
findit	Search Stata documentation
findit [word]	search keywords through Stata documents, which is useful for finding user-written ado file.
help	Display help in Stata
help [command_or_topic_name]	
log	Write the session result to a log file
log using [logname]	Create a log file
log close	Close a log file
set	Overview of system parameters
set maxvar	Set the maxium number of variables to be read in Stata
set more off/on	Tell Stata to pause or not pause for --more-- messages.
Comments	Add comments to programs
*	Used at the start of the command line
//	Used at the end of the command line
/* */	Used for a block fo command line
If	The if expression at the end of the command applies the command to records meeting the specification of the if expression.
if varname == value	The condition is defined as a specific value of a variable
if inlist(variname, value1, value2,…)	The condition is defined as a list of values of a variable
if inrange(variname, value1, value2)	The condition is defined as a range of values of a variable
exit	Stop the do file

BGSU Center for Family and Demographic Research

Managing Datasets

Table 2. Commands for Managing Data	
Key command word	**Purpose**
use	Load Stata dataset
use [filename] [, clear]	
des	Describe data in memory or in file
des	Describe data in memory or in file
des [varlist]	describe variables in memory
duplicates	Identify respondents with the same records
duplicates report [varlist]	Check if there are duplicates records
sort	Sort data
sort [varlist]	Sort the records, based on the ascending order of the values of variables
merge	Merge datasets
merge 1:1 [varlist] using [filename]	One-to-one merge on specified key variables
merge m:1 [varlist] using [filename]	Many-to-one merge on specified key variables
merge 1:m [varlist] using [filename]	one-to-many merge on specified key variables
reshape	change the form of data file
reshape long [stubnames] , i(varlist)	Convert data from wide form to long form
reshape wide [stubnames] , i(varlist)	Convert data from long form to wide form

BGSU Center for Family and Demographic Research

Checking Variables

Table 3. Commands for Checking Variables	
Key command word	**Purpose**
list	List values of variables
list [varlist]	List values of variables
list [varlist], sepby (varlist2)	List values of variables and draw a separator line whenever varlist2 values change
list [varlist] [if]	List values of variables for records meeting the if condition
list [varlist], nol	List values of variables without the value label
tab	Generate one- or two- way tables of frequencies
tab1 [varname], mis	Generate one-way tables of frequencies, including the missing cases
tab2 [varname1] [varname2], mis	Generate two-way tables of frequencies, including the missing cases
summarize	Summary statistics
sum [varlist]	Summary statistics of variables

Modifying Existing Variables

Table 4. Commands for Modifying Existing Variables	
Key command word	**Purpose**
rename	Rename variable
rename old_varname new_varname	Change the name of a variable
rename *, lower	Convert all variable names to lowercase
rename *, upper	Convert all variable names to uppercase
label	Manipulate labels
label variable [varname] ["label"]	Label variable
label define [lblname] # "label" [# "label" ...] [, add modify replace]	Define value label
label values varlist [lblname|.] [, nofix]	Assign value label to variables
label drop {lblname [lblname ...] | _all}	Drop value labels
recast	Change storage type of variable
recast [type] [varlist]	Change storage type of a numeric or string variable
recode	Recode the value of variables
recode varlist (rule) [(rule) ...]	Recode the values of an existing variable
recode varlist (rule) [(rule) ...] [, generate(newvar)]	Generate a new varaible with the recoded values of an existing variabl

Commands for Modifying Existing Variables (Continued)

Table 4. Commands for Modifying Existing Variables (Continued)

Key command word	Purpose
tostring	Convert numeric variables to string variables
tostring varlist , generate(newvarlist)	Generate a new string variable from an existing numeric variable
tostring varlist , replace	Change an existing numeric variable into a string variable
destring	Convert string variables to numeric variables. This command works only for the string variable does not have text strings.
destring varlist , generate(newvarlist)	Generate a new numeric variable from an existing string variable. This command works only for the string variable does not have text strings.
destring varlist , replace	Change an existing string variable into a numerica variable
mvencode	changes missing values in the specified varlist to numeric values.
mvencode varlist , mv(#|mvc=# [\ mvc=#...] [\ else=#])	Recode missing values of an existing variables into numeric values
mvdecode	Change numeric values to missing values
mvencode varlist , mv(#|mvc=# [\ mvc=#...] [\ else=#])	Recode numeric values of an existing variables into missing values

BGSU Center for Family and Demographic Research

Generating New Variables

Table 5. Commands for Generating New Variables	
Key command word	**Purpose**
clonevar	Clone existing variable
clonevar newvar = varname	generates newvar as an exact copy of an existing variable, varname, with the same storage type, values, and display format as varname.
gen	Create or change contents of variable
gen newvar = _n	generate a new variable from the current observation number.
gen newvar = _N	generate a new variable indicating the total number of observation
gen [type] [newvar] =exp	
tab1 (varname), gen(newvar)	Generate new dummy variables for the values of a variable
gen newvarr=substr(oldvar,value1,value2)	generate a new variable using a substring from a string variable
gen newvar =sum(oldvar)	Create variable containing the running sum of an existing variable

Generating New Variables (Cont.)

Table 5. Commands for Generating New Variables (Continued)

Key command word	Purpose
egen	Extensions to generate
egen [type] newvar = fcn(arguments)	The basic syntax of egen commands
fcn(arguments):	
anycount(varlist), values	returns the number of variables in varlist for which values are equal to any integer value in a supplied numlist.
anyvalue(varname), values(integer numlist)	It takes the value of varname if varname is equal to any integer value in a supplied numlist and is missing otherwise.
cut(varname), {at(#,#,...,#)	It sets the thresholds of a numerical variable and then generates a new variable
concat(varlist), punct(pchars)	It concatenates varlist to produce a string variable.
rowmiss(varlist)	It gives the number of missing values in varlist for each observation (row).
rownonmiss(varlist)	It gives the number of nonmissing values in varlist for each observation (row).
by varname1, sort: egen newvar = sum(varname2)	Create variable containing the running sum of an existing variable (varname2), conditioned on the value of varname1

Organize Your Files

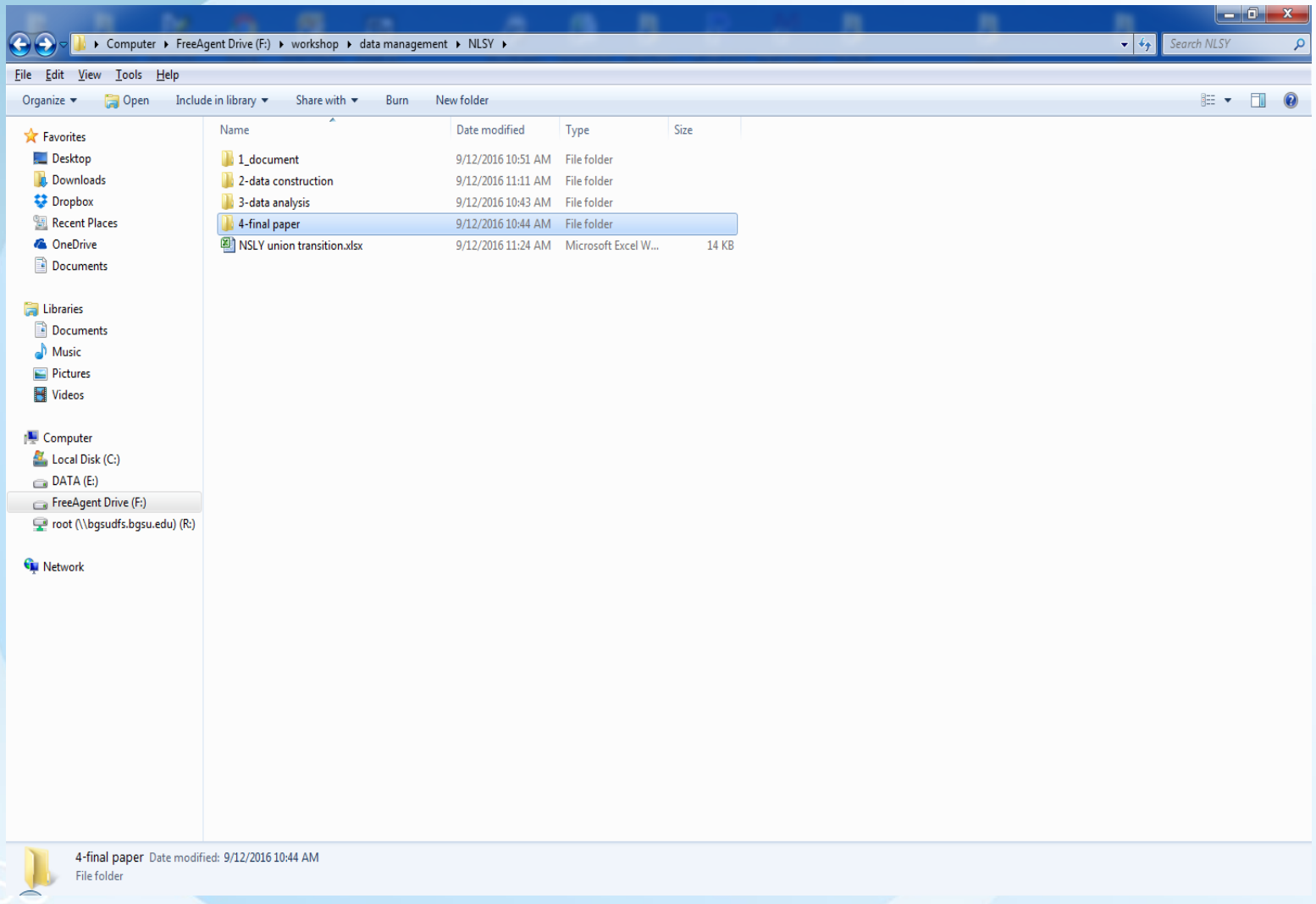
- Create a folder system to help you organize files for the same project

- Create a spreadsheet to keep track of your files
  - Name the command, log, result, and data files wisely, so you can easily find the correct files
  - Remember to include the version number or the date in the file name
  - Remember to merge and purge command files for data construction

Organize Your Files (Cont.)

- Always write separate command files for constructing variables and for data analysis
- Command files should have their corresponding log/result files
- Because data construction has its continuity, you usually just need one command file linking the original data to data file that you are currently working. Don't generate a new data file each time you modify a command file.
- Because results of data analysis are determined by the data used, you do not need to merge the command files for data analysis if they use different data sets.
- Keep only the variables and observations that you need in your data files
- Document the purpose, data file, log file, result file, problems, decisions, thoughts, doubts in the command files
- Document how new variables are created from the original variables
- When constructing variables, you should not overwrite the original variables
- You should have "permanent file" and "working file" when working on your project

Create A Folder System



Conclusions

- Data management problems influence not only your research but also other people's.
- Data management problems include both how to create data and how to organize files related with data construction
- Create separate command files for data construction and data analyses. Document everything in your command files
- Each command file should have its corresponding log file.
- Included detailed comments in the command files to remind you how and why you create or modify the variables.
- After you complete a task, merge the command file with the previous ones.
- Use consistent names for command, log, result, and data files.
- Creating an Excel file significantly helps keep track of all the files
- If you have any questions about data management problems, feel free to contact Hsueh-Sheng Wu @372-3119 or wuh@bgsu.edu