# basic organization and documentation

adam okulicz-kozaryn

adam.okulicz.kozaryn@gmail.com

this version: Thursday 17th February, 2022 14:29

# outline

directory (folder) and file (data and code (dofiles)) structure

code structure (within one file)

naming, labeling

# outline

directory (folder) and file (data and code (dofiles)) structure

code structure (within one file)

naming, labeling

# replication: raw− >clean− >analysis

- always keep raw data intact
- then manipulate it and save, even several times
- will have few dats at different stages
- can begin stata session at any stage
- blackboard: draw workflow

# files in general singularity rule

- **always one version of a dofile or datafile in one place**
- if you have 2 versions of the same file
  - ○ sooner or later there will be problems!
  - ○ you will update/change one, but forget the other one, etc
- exception is backup; but you never edit the backup!

# code in general singularity rule: branching

- just like with files, so with code:
- **have the same chunk of code only in one place**
- if same code repeats across multiple dofiles:
  - then build hierarchy: parent-children
  - parent does basic and generic
  - children pick up same data from parent and diverge
- eg use same data for many projects
  - parent dofile makes it ready for multiple papers
  - proces raw data into friendly shape
    (recode, label, calculate new vars, etc)
  - and then always just start from there for each new project
- blackboard: draw diagram/flow chart (next slide)

# code and data: hierarchy and branching

- never overwrite the original datafile, and have datafiles at different stages esp if data complex:
- ◦ rawFile$-$ $>$file1$-$ $>$file2 –and those are produced by: dofile0$-$ $>$dofile1$-$ $>$dofile2 (or subsequent sects in one dofile)
- dofile0 will common for all projects
- ◦ dofile0 may have 2 children: dofile1A and dofile1B
- likewise, rawFile may have 2 diff children file1A and file1B

# backup

- **backup all files at least once a week**–computers break regularly; flash drives break really often
- have automatic system for backups (i use cron)
  - otherwise you'll forget
- just keep copy of everything in the cloud, goog, amzn, etc

# outline

directory (folder) and file (data and code (dofiles)) structure

code structure (within one file)

naming, labeling

## sections, subsections

- dofile should have a multi-layerd structure
  - like chapters, sections, sub-sect in book
- for different levels, use different kinds of comments: box, block, one line, horizontal line, etc

  type them in dofiles and scroll down to already existing
  - now i just use '***', '**', '*', '//'
  - i used to use '//——' (still in dofile)
- definitely use “FIXME” “LATER” “KLUDGE” etc

# outline

directory (folder) and file (data and code (dofiles)) structure

code structure (within one file)

naming, labeling

# general

- naming and labeling looks like waste of time
- but at the end saves time
- importantly, it prevents mistakes/misinterpretations
  - especially, if a project is big and/or you share it with others
  - or if it takes long time

# variable names, labels, and value labels

- variable name is...a variable name, eg educ
- var lab describes var, eg “highest degree completed”
- value label describes values that a variable takes on
  - (output of codebook, or tab and tab,nola), eg:
  - “primary school” 1
  - “high school” 2
  - “college or university”3
- dofile

# labels tips

- give vars short names eg inc
- but labels should be descriptive eg “2004 hh income”
- labels prevent confusion later and for others
- they automatically appear on graphs, regressions, etc.
- use lookfor, esp if you have many vars
- be lazy (remember it’s our core value)
- ○ only label what’s necessary
- ○ indeed, only keep data and variables that are necessary
- ○ you have the code, so you can always add back in later