# “Code and Data for the Social Sciences: A Practitioner’s Guide”

by Matthew Gentzkow and Jesse M. Shapiro

Jaime Arellano-Bover

*Applied Micro in Stata Workshop*

RoME, 2022

RoME Rome Masters in Economics

# Motivation and Purpose

- Modern Economics empirical projects are large-scale complex enterprises
  - Thousands of lines of code
  - Gigabytes of data
  - Many collaborators
  - Duration of multiple years

- How to best manage code and data in such projects?
  - These skills typically taught in Computer Science, but not in Economics programs
  - Econ researchers typically learn basics of programming as they go
  - But intuitive self-taught approach can hit limits and lead to common problems…

RoME Rome Masters in Economics

- In trying to replicate the estimates from an early draft of a paper, we discover that the code that produced the estimates no longer works because it calls files that have since been moved. When we finally track down the files and get the code running, the results are different from the earlier ones.

- A referee suggests changing our sample definition. The code that defines the sample has been copied and pasted throughout our project directory, and making the change requires updating dozens of files. In doing this, we realize that we were actually using different definitions in different places, so some of our results are based on inconsistent samples.

- We are keen to build on work a research assistant did over the summer. We open her directory and discover hundreds of code and data files. Despite the fact that the code is full of long, detailed comments, just figuring out which files to run in which order to reproduce the data and results takes days of work. Updating the code to extend the analysis proves all but impossible. In the end, we give up and rewrite all of the code from scratch.

# Motivation and Purpose

- Goal of this handbook:

- ***Translate insights from experts in code and data into practical terms for empirical social scientists***

RoME Rome Masters in Economics

# List of topics (more in the handbook)

1. Automation
2. Version Control
3. Directories
4. Keys

RoME Rome Masters in Economics

# AUTOMATION

# Example research project

- Effect of introduction of TV in the US on sale of potato chips

- Raw data: Excel file with two worksheets
  1. “tv”: for each county in the US, year of TV introduction
  2. “chips”: sales of chips by county by year 1940-1970

- Goal: Estimate panel regression
  - log chip sales on a dummy variable for TV being available with county and year fixed effects

RoME Rome Masters in Economics

# Possible way to proceed

1. Open file in Excel, “Save as” to save worksheets as text files
2. Open Stata and, using command line, write necessary commands to load, reshape, and merge text files
3. From command line, run the regression and copy regression output
4. Paste regression output into a Word file
5. Write discussion of the results, save document. Submit to journal

- Is this a good idea?

# “Interactive” mode of research is bad

- We all learn that data building and statistical analysis should be stored in scripts (i.e., `.do` files in Stata, `.m` files in Matlab, `.r` files in R…)
- Why?

1. **Replicability**
   - If a year from now we want to reproduce our regression, we need a record of the precise steps that were taken, both in data building and in estimation.

2. **Efficiency**
   - If we need to implement a change in the analysis, repeating all steps each time is time-consuming.

# Project directory after writing .do files

```
chips.csv          mergefiles.do          tv_potato_submission.pdf
cleandata.do       regressions_alt.do     tv_potato.tex
extract0B.xls      regressions_alt.log    tv.csv
fig1.eps           regressions.do         tvdata.dta
fig2.eps           regressions.log
figures.do         tables.txt
```

- Better than interactive mode, but how long would it take you to reverse-engineer how to replicate the findings in `tv_potato_submission.pdf` ?
  - (Even if you are the only author, it's surprising how little one can remember of what you at *t*-6 months knew)

RoME Rome Masters in Economics

# Write a master file!

- A master file works like a roadmap
- It details the order in which to run the whole project directory
- *Best*: automate everything, across different software, so that master file can be run from computer command line
- *Second best*: automate all Stata analysis (what we'll do)

```
---- rundirectory.bat ----
stattransfer export_to_csv.stc
statase -b mergefiles.do
statase -b cleandata.do
statase -b regressions.do
statase -b figures.do
pdflatex tv_potato.tex
```

RoME Rome Masters in Economics

# VERSION CONTROL

# Our project folder after working on it a while

```
cleandata_022113.do      cleandata_022613.do      regressions.log
cleandata_022113a.do     cleandata_022613_jms.do  regressions_022413.do
chips.csv                tvdata.dta               regressions_022713_mg.do
regressions_022413.log
```

- There are good reasons to store multiple versions of the same file
  - Discard changes, compare different ways of estimating something or defining variables
- But the method of using dates is bad!
  - Ambiguity on when to have a new version, unclear which file enters the project's workflow, or how

RoME Rome Masters in Economics

# Solution: Use *Version Control*

- Version control software tracks successive versions of a given piece of code:
  1. Set up a repository (a folder) on your computer or the cloud
  2. Each time you want to modify a script, you "check it out" of the repository
  3. After you are done changing it, you check it back in
- One file, no need to change names, no need to add dates
- Software remembers every version that was ever checked in
- You can always check the history of changes and go back to an old version

# Version Control

- Many options for to implement version control (and newer ones since when document was written in 2014)
- I use a combination of a GitHub account and the software Sourcetree (we'll see it during the workshop)
- Version control is not only useful for code and data, but also for drafts of paper! (a paper in LaTeX is a bunch of code after all)
  - Avoid the shameful file: `thesis_final_final_edits_reallyfinal.tex`

# DIRECTORIES

# Separate directories by function

- Instead of having a single directory, better to separate into two high-level directories:

1. `/build`: contains code to build a usable Stata file from the raw inputs
2. `/analysis`: code to take the Stata file and turn it into figures and tables for the paper

- Within each directory, there is a consistent structure:
  - inputs
  - outputs
  - code
  - temporary or intermediate files

- Benefits
  - separate pipelines ensures dataset consistency
  - more efficient to work on/debug one part of the project without touching the other

```
---C:/build---
/input
    extract0B.xls

/code
    rundirectory.bat
    export_to_csv.stc
    mergefiles.do

/output
    tvdata.dta

/temp
    chips.csv
    tv.csv
```

```
---C:/analysis---
/input
    tvdata.dta (link to C:/build/output)

/code
    rundirectory.bat
    regressions.do
    regressions_alt.do

/output
    fig1.eps
    fig2.eps
    tables.txt

/temp
    regressions.log
    regressions_alt.log
```

RoME Rome Masters in Economics

# KEYS

# New table with county population as control for potato chip consumption

| county | state | cnty_pop | state_pop | region |
|---|---|---|---|---|
| 36037 | NY | 3817735 | 43320903 | 1 |
| 36038 | NY | 422999 | 43320903 | 1 |
| 36039 | NY | 324920 | . | 1 |
| 36040 | . | 143432 | 43320903 | 1 |
| . | NY | . | 43320903 | 1 |
| 37001 | VA | 3228290 | 7173000 | 3 |
| 37002 | VA | 449499 | 7173000 | 3 |
| 37003 | VA | 383888 | 7173000 | 4 |
| 37004 | VA | 483829 | 7173000 | 3 |

- A mess…
  - NY state population missing in third row
  - state missing in fourth row
  - VA region should be constant but is not
- These data are unusable, we cannot understand what they represent or how they were constructed

# Unique keys and relational databases

- A county *table* and a state *table*
- Each table has a *key:*
  - i.e., variable(s) that uniquely identifies an element (row)
  - variables that form the key are never missing and are never duplicated
- Each variable in a table is an attribute of the table's elements
  - e.g., state population is a property of a state, so it cannot live in the county table

| county | state | population |
|---|---|---|
| 36037 | NY | 3817735 |
| 36038 | NY | 422999 |
| 36039 | NY | 324920 |
| 36040 | NY | 143432 |
| 37001 | VA | 3228290 |
| 37002 | VA | 449499 |
| 37003 | VA | 383888 |
| 37004 | VA | 483829 |

| state | population | region |
|---|---|---|
| NY | 43320903 | 1 |
| VA | 7173000 | 3 |

RoME Rome Masters in Economics

# Unique keys and relational databases

- Data stored in this form is considered *normalized*
- It's good practice to keep data normalized as long as possible through your project's workflow

- At the minimum, when manipulating a dataset you should always check you know what its unique identifier is
  - In Stata this is easy to check with the command `isid`

RoME Rome Masters in Economics

# The End. Read the full document!

- Well-written and entertaining
- It provides more useful information and examples than what I summarized here
- More on coding best practices and project management that will make your life easier
- You can find it on workshop website

RoME Rome Masters in Economics