Data Analysis for Policy Research Using R

Introduction and R Basics

Harold Stolper

1. Introductions

2. What is R and How Will We Use It?

3. Prerequisites, R setup, course goals

4. R Projects and Directory Structure

5. R Basics

6. Assignments and other course responsibilities

7. Questions about the syllabus and course expectations?

Introductions

Student introductions

1. Preferred name (and pronouns, if comfortable sharing)
2. Previous R exposure/experience
3. And answer one of these two questions:
   - What's something you would eventually like to learn how to do in R?
   - What's something that you have observed or think is important that people in your field aren't paying attention to?
4. (optional) One piece of culture you are excited about right now
   - e.g. music, writing, fashion, a meme, other art, sports, etc.

Teaching Assistants

- Neil Ke-Jian Chin (morning section)
- Tirtha Patel (afternoon section)

Harold Stolper, instructor

Graduated from SIPA many moons ago, returned to Columbia for my PhD in economics.

Recent background:

- This is my 8th year teaching quant courses at SIPA.
- Previously worked as the economist for a non-profit doing research and advocacy to promote upward mobility for low-income NYers.
- Recent focus on NYPD subway fare evasion enforcement and over-policing for communities of color, and other topics related to urban inequality (e.g. policing disparities, neighborhood change, transit, access to education).

Transitioned from Stata to R after years of using and teaching Stata.

What is R and How Will We Use It?

What is R?

- R is “an alternative to traditional statistical packages such as SPSS, SAS, and Stata such that it is an extensible, open-source language and computing environment for Windows, Macintosh, UNIX, and Linux platforms.” (ICPSR)
- “R is an integrated suite of software facilities for data manipulation, calculation and graphical display.” (R-project.org)

How will we use R?

- RStudio is a powerful user interface for R.
  - After installing R and RStudio, we'll be working entirely in the RStudio interface.
  - We'll be working with R scripts every week! (an R script is more or less a text file that RStudio recognizes as R code)

- R Markdown files are used in RStudio to "both save and execute code and generate high quality reports that can be shared with an audience."
  - These lecture slides were created using R Markdown.
  - Beginning with Assignment 2, everything you *submit* for this class will be a document generated with R Markdown.
  - But your workflow should always begin with an R script before writing up your work using R Markdown.

Base R vs. user-defined R packages

R uses “packages” as a bundle of code, data and documentation.

There are default base packages that come ready-to-use with R. Some examples:

- `base`
- `stats`
- `utils`

Then there are R packages developed and shared by others. Some popular R packages include:

- `tidyverse`
- `ggplot2`

More about these in later weeks…

Installing and loading R packages

You only need to install a package once. To install an R package use `install.package()` function.

```
install.packages("tidyverse")
```

But you need to load a package every time you open R using the `library()` function.

```
library(tidyverse)
#> -- Attaching packages --------------------------------------- tidyverse 1.3.2
#> v ggplot2 3.3.6     v purrr   0.3.4
#> v tibble  3.1.8     v dplyr   1.0.9
#> v tidyr   1.2.0     v stringr 1.4.0
#> v readr   2.1.2     v forcats 0.5.1
#> -- Conflicts ------------------------------------------ tidyverse_conflicts()
#> x dplyr::filter() masks stats::filter()
#> x dplyr::lag()    masks stats::lag()
```

- Base packages don't need to be installed or loaded.

What can you do with R+RStudio+RMarkdown?

Things you can also do using Stata:

- Data cleaning and manipulation
- Statistical analysis and plots

Things you generally can't do in Stata:

- Generate reports and presentations
- Generate interactive content
    - Maps
    - Graphs
    - Dashboards

What will we be doing in this class?

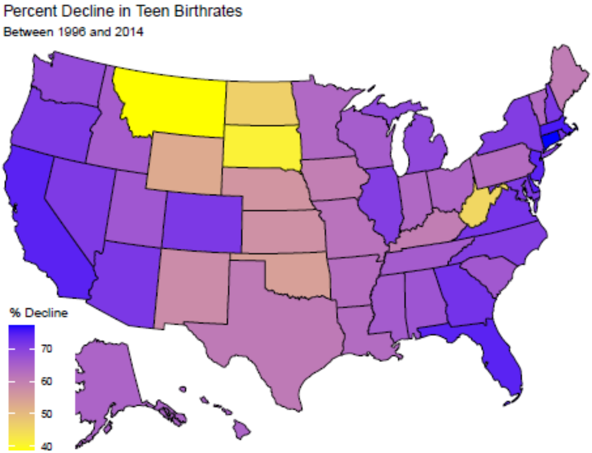
We'll be learning how to use R to explore data to inform policy.

That means we'll be spending a lot of time working through R, but also thinking about how/when to use the methods and concepts we've learned in Quant I and II:

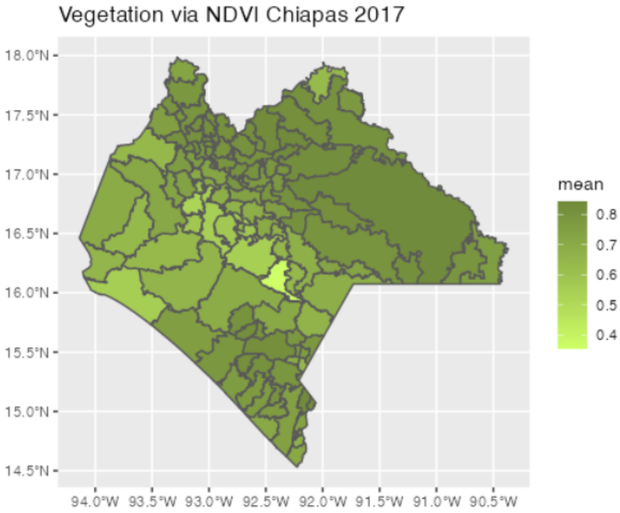
- **Research design:** understanding how data structure and methods impact analysis and causal inference
- **Data wrangling:** cleaning and structuring messy data for analysis
- **Exploratory analysis:** identifying and analyzing key factors in your analysis
- **Explanatory analysis:** estimating relationships between variables to inform policy
- **Data visualization & presentation:** conveying findings to your target audience
- **Policy writing & interpretation:** translating statistical analysis in accessible terms
- **Data advocacy**: thinking critically about using *data for good*

Examples from student project work

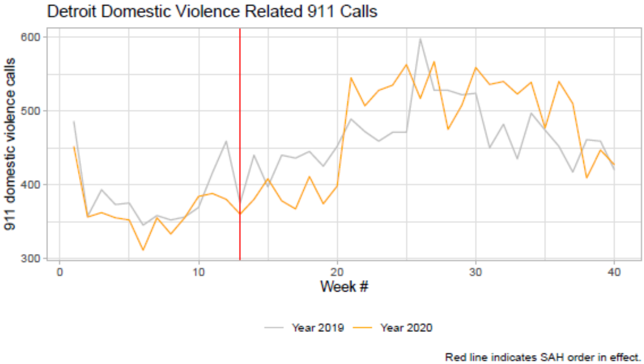
Figure III: Change In Birthrate by State



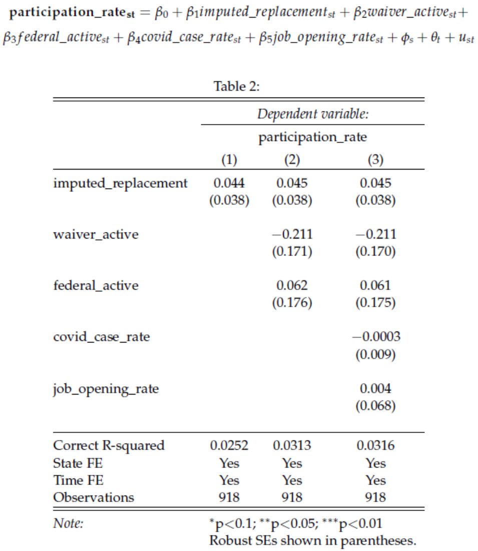
Examples from student project work



Examples from student project work



Examples from student project work



$$\mathbf{participation\_rate_{st}} = \beta_0 + \beta_1 imputed\_replacement_{st} + \beta_2 waiver\_active_{st} + \beta_3 federal\_active_{st} + \beta_4 covid\_case\_rate_{st} + \beta_5 job\_opening\_rate_{st} + \phi_s + \theta_t + u_{st}$$

Table 2:

	Dependent variable:		
	participation_rate		
	(1)	(2)	(3)
imputed_replacement	0.044 (0.038)	0.045 (0.038)	0.045 (0.038)
waiver_active		−0.211 (0.171)	−0.211 (0.170)
federal_active		0.062 (0.176)	0.061 (0.175)
covid_case_rate			−0.0003 (0.009)
job_opening_rate			0.004 (0.068)
Correct R-squared	0.0252	0.0313	0.0316
State FE	Yes	Yes	Yes
Time FE	Yes	Yes	Yes
Observations	918	918	918
Note:	*p<0.1; **p<0.05; ***p<0.01 Robust SEs shown in parentheses.		

Prerequisites, R setup, course goals

Questions for you

https://pollev.com/haroldpoll

R Projects and Directory Structure

Working directory

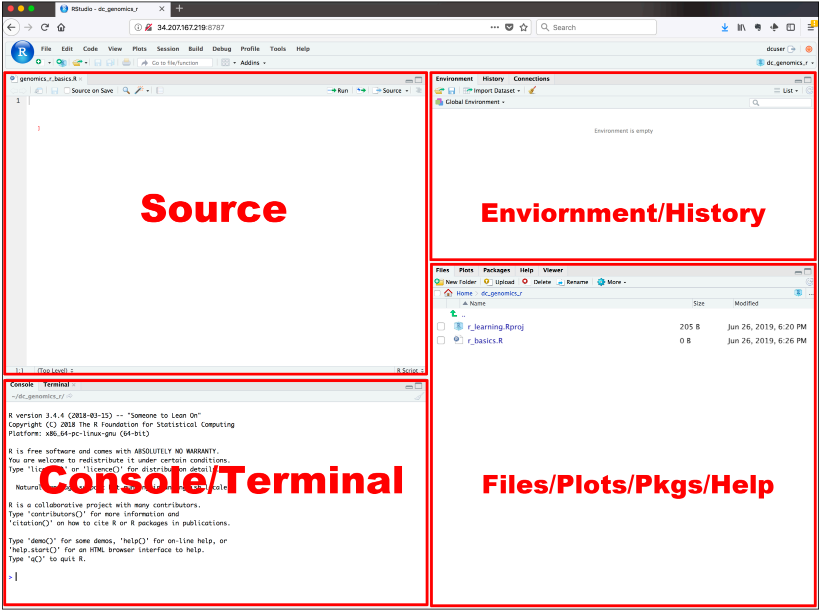
R looks for files in your **working directory**

Function `getwd()` shows the current working directory (also shown at the top of the RStudio console.)

```
getwd()
#> [1] "I:/My Drive/Teaching/U6614-drive/Lectures-drive/Lecture1"
```

You can see all files located in your working directory in the “Files/Plots/Packages/Help/Viewer” pane (by default in the lower right)

RStudio interface



Name	Size	Modified
..		
r_learning.Rproj	205 B	Jun 26, 2019, 6:20 PM
r_basics.R	0 B	Jun 26, 2019, 6:26 PM

```
R version 3.4.4 (2018-03-15) -- "Someone to Lean On"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> |
```

So what is the working directory?

When you run code from the **R Console** or an **R Script**, or from **code chunks** in an R Markdown file (.rmd), the working directory is:

- the folder your file is saved in, or…
- if you are working within an **R Project**, the working directory is the main directory for the project (more on that shortly!)

```
getwd()
#> [1] "I:/My Drive/Teaching/U6614-drive/Lectures-drive/Lecture1"
```

- For this class we'll be using R projects to keep organized.

What is an R project? Why are we using them?

What is an “R project”?

- A file that keeps all “project” files organized together:
  - e.g. input data, R scripts, saved output, figures.
- When you open an R project, your working directory is automatically set to the directory where your R project (and related files) lives.

Why will we be asking you to create and work with R projects?

- Projects are important for keeping organized and avoiding file path errors
- You should have a project for every week that will include R scripts from class, data files, and also R Markdown files that you’ll create for assignments.

The path is how we refer to a directory

Absolute file path: a complete list of directories needed to locate a file or folder.

```
setwd("I:/My Drive/Teaching/U6614-drive/Lectures-drive/Lecture1")
```

Relative file path: a way of indicating a given file location relative to your working directory (note that they might be the same!)

- Assuming your current working directory is in the “lecture1” folder and you want to go up 2 levels to the “U6614-drive” folder, your relative file path would look something like this:

```
setwd("../../")
```

File path shortcuts:

Key	Description
~	tilde is a shortcut for user’s home directory
../	moves up a level
../../	moves up two level

*Note that R does not accept backslashes here, only forward slashes.

R Basics

Executing code in R

Three ways to execute commands in R

1. **Console:** type/paste commands to run “on the fly”
2. **R scripts** (.r files)
   - Just a text file full of R commands
   - Can execute one command at a time, several commands at a time, or the entire script
3. **Code chunks** in R Markdown (.rmd files)
   - Can execute one command at a time, one chunk at a time, or “knit” the entire file into a document (e.g. html or pdf)

Shortcuts for executing commands

Three ways to execute commands in R

1. Type/paste commands directly into the "console" (and press ENTER)
2. R scripts (.R files)
    - **Cmd/Ctrl + Enter**: execute highlighted line(s)
    - **Cmd/Ctrl + Shift + Enter** (without highlighting any lines): run entire script
3. 'code chunks' in RMarkdown (.rmd files)
    - **Cmd/Ctrl + Enter**: execute highlighted line(s) within chunk
    - **Cmd/Ctrl + Shift + k**: "knit" entire document

Assignment

Assignment means assigning a value/set of values to an "object"

- `<-` is the assignment operator
  - in other languages `=` is the assignment operator
- good practice to put a space before and after assignment operator

```r
# Create an object a and assign value
a <- 5
a
#> [1] 5

# Create an object b and assign value
b <- "I'm so excited to be here in the lab again!"
b
#> [1] "I'm so excited to be here in the lab again!"
```

Note 1: comments start with one or more `#` symbols

Note 2: R is caps sensitive!

Objects and assignment

R stores information in objects (like all “object-oriented” programming languages).

Some objects:

- numbers
- character strings
- vectors
- matrices
- lists
- functions
- plots
- data frames (the datasets of R!)

Functions

Functions do things to different objects. They often accept arguments – we “pass” arguments to functions.

Functions are also objects themselves that can be “called” to do things like:

- calculate and display statistics
- generate output
- display part or all of objects (e.g. show some data)
- manipulate objects (e.g. create a new column of data)
- extract information from objects (e.g. the number of rows of data)

Base R includes lots of functions. We’ll be working with additional packages that include some handy functions.

Let's jump in!

Our goals for today's R workshop example are very modest:

- Create an R project including R script.
- Look around and get our bearings.
- Install and load a package (gapminder).
- Use base R functions to inspect a data frame included with this package.
- Use some functions to perform some very basic analysis.
- Assign results from our analysis to new objects and display them.

Assignments and other course responsibilities

Assignment 1: submit an R script via CW by 11:59pm next Monday

Create an R script called assignment1.r that includes code and answers (as comments) for the following:

0. Create a new R project called assignment1.
   - This is for your internal project management, do not submit your R project file.
1. Load the gapminder data using the library function.
   - You'll need to install the gapminder package if you didn't follow along in class today.
2. Show the data structure of the gapminder data frame in the gapminder package.
3. What is the average gdpPercap across all observations in the data frame?
   - Use ?gapminder to access gapminder documentation and find the units for gdpPercap.
   - How would you interpret this mean? i.e. what is it the mean of?
4. Plot year (x-axis) vs. gdpPercap (y-axis).
   - Describe what the plot says about economic growth over time.
5. Create a barplot showing the number of observations in each continent.
   - Start by using the table function with continents as its argument.
   - Next pass the object created by this function to the barplot function.

General assignment guidance

- Use blank spaces liberally, code is hard to read and spaces help!
- Use comments liberally throughout your R script to describe your steps.
- Troubleshooting is a skill that will take time to develop! Here are some tips and resources:
  - Consult the R script from class for examples.
  - Get used to using R's built in documentation by using `?`
  - Use Google liberally for examples that work.
  - When you're stuck, focus on finding examples to get your own code to work, even if you don't feel comfortable with all the syntax just yet.
  - Execute your code line by line as you go, this will help you isolate the source of any errors.
- Consulting with others is good! Copying, however, is not the way to learn to code.
  - **Copied assignment submissions will result in a 0 for all parties.**

Use Ed Discussion for help!

- If you need help troubleshooting errors, posting to Ed Discussion is usually a great place to start (in addition to office hours).
- Your post should provide a reproducible example, including code and a screenshot of the output/error message if applicable.
- Don't be bashful about posting for help troubleshooting code or setup issues... if you're running intro trouble, odds are somebody else is too!
- A teaching team member will reply soon, and you are also encouraged to reply to each others' posts if you have insight about how to resolve the issue (also an easy way to boost your participation).

Quizzes on pre-class lessons

Starting next class, every class will begin with a short Courseworks quiz covering the pre-class lessons:

- Typically 10 multiple choice questions in total, ~5 minutes to complete
- Will include 1 question on the Data Primer covering the data to be used next class
- Academic integrity: quizzes are closed-note (and Courseworks Quizzes shows a log of your quiz activity)
- I try not to emphasize detailed memorization for the quizzes but inevitably there is some
  - As long as you are thoughtfully engaging with the pre-class material in advance then you're on the right track. Don't be discouraged if, as a new R user, it takes a bit of time to see your quiz scores improve.

Attendance and participation

Recitation

- **You must be able to attend one of the Thursday recitation times**
  - Early in the semester recitation will be used to review code from class and prep for assignments
  - Later in the semester this time will be used for extra office hours and project meetings
  - Email me right away if you want to attend a different recitation time than the one for the section you are registered for

Attendance and participation

Zoom

- If you are feeling unwell or facing unexpected challenges outside of school, please email me right away to briefly explain your situation and request a Zoom link for class if necessary.
- Whenever feasible, Zoom links and/or recordings will be provided for students with excused absences. We will try to record at least one of the two weekly class meetings (and recitations)–not necessarily both–so you will either have the opportunity to attend remotely in real-time, or view the recording after.

Participation

- Keep in mind that participation–in-class and through Ed Discussion–is worth 10% of your total grade. We want to create our own data community with engagement from everybody in ways they are comfortable with.

Questions about the syllabus and course expectations?