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. Course Logistics

4. RStudio

5. R Projects and Directory Structure

6. R Basics

7. Assignment 1

Introductions

Student introductions

1. Preferred name
2. Pronouns (if you'd like to)
3. Previous R exposure/experience (if any)
4. 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?
5. One piece of culture you are excited about right now
    - e.g. music, writing, fashion, a meme, sports, etc.

Kevin Wibisono, Teaching Assistant

MS in Data Science, graduating December 2020.

- Pronouns: he/him/his.
- Originated from Jakarta, Indonesia.
- Studied Applied Mathematics and Statistics at National University of Singapore (2015-2019).
- Research Assistant for the Urban Design Lab at Columbia University.

Harold Stolper, instructor

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

Past 5 years:

- Taught quant courses at SIPA as an adjunct.
- Worked as the economist for a non-profit doing research and advocacy to promote upward mobility for low-income NYers.
- Recent focus on documenting racist police enforcement of fare evasion, and other topics related to policing, neighborhood change, and transit accessibility.

Transitioning 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 you install R and RStudio, we'll be working entirely in the RStudio interface.
- 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 pdf lecture slides were created with R Markdown.
    - Subsequent weeks we'll rely on html-based lessons created with R Markdown.
    - Beginning next week, everything **you** submit for this class will be a document generated with R Markdown.

Base R vs. user-defined R packages

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

The default R base package includes much of the basic functionality you will be using.

Then there are R packages developed and shared by others. Some R packages we’ll be using 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 before you can use it you need to load a package every time you open R using the `library()` function.

```
library(tidyverse)
#> -- Attaching packages ------------------------------------------------------
#> v ggplot2 3.3.2     v purrr   0.3.4
#> v tibble  3.0.3     v dplyr   1.0.1
#> v tidyr   1.1.1     v stringr 1.4.0
#> v readr   1.3.1     v forcats 0.5.0
#> -- Conflicts ---------------------------------------------------------------
#> x dplyr::filter() masks stats::filter()
#> x dplyr::lag()    masks stats::lag()
```

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 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 impacts analysis and causal inference
- **Data management:** cleaning and structuring data for analysis
- **Exploratory analysis:** identifying and analyzing key factors in your analysis
- **Explanatory analysis:** estimating relationships between variables to inform policy
- **Data visualization and presentation:** conveying findings to your target audience
- **Policy writing and interpretation:** translating statistical analysis in accessible terms

Course Logistics

Syllabus and file management

Course links:

- All course files will be posted on the course website: https://hreplots.github.io/U6614/

We'll still be using Courseworks for several purposes:

- organizing Zoom links for classes and recitation
- sign-up for TA office hours
- submitting assignments and project deliverables
- course communications and discussion using Piazza (register here)

Synchronous and asynchronous instruction

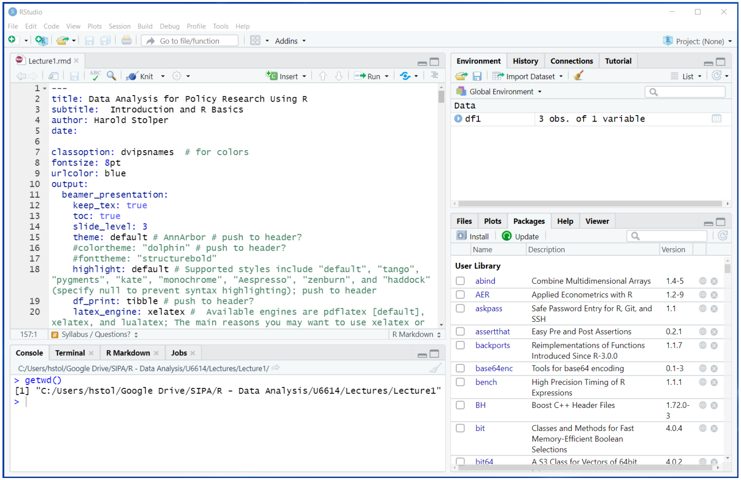
1. Review asynchronous lessons in advance of class
   - will be posted to the course website by Thursday night
   - class meetings will begin with a very short multiple choice quiz on the asynchronous material
2. Synchronous instruction
   - short Zoom quiz
   - discussion of data applications including assignments
   - workshop-style instruction with R
     - prepare for class by downloading the week's R script & data from the course website
     - maintain a logical file structure to organize files (e.g. Lectures/Lecture1)
   - we'll set *community guidelines* for in-class participation/R support next week

Questions?

RStudio

What is RStudio?

"RStudio is an integrated development environment (IDE) for R. It includes a console, syntax-highlighting editor that supports direct code execution, as well as tools for plotting, history, debugging and workspace management."


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Lecture1.rmd
Knit
Insert
Run
157:1
Syllabus / Questions?
R Markdown
Console Terminal R Markdown Jobs
Environment History Connections Tutorial
Import Dataset
List
Global Environment
Files Plots Packages Help Viewer
Install Update

```
---
title: Data Analysis for Policy Research Using R
subtitle:  Introduction and R Basics
author: Harold Stolper
date:

classoption: dvipsnames  # for colors
fontsize: 8pt
urlcolor: blue
output:
  beamer_presentation:
    keep_tex: true
    toc: true
    slide_level: 3
    theme: default # AnnArbor # push to header?
    #colortheme: "dolphin" # push to header?
    #fonttheme: "structurebold"
    highlight: default # Supported styles include "default", "tango",
"pygments", "kate", "monochrome", "Aespresso", "zenburn", and "haddock"
(specify null to prevent syntax highlighting); push to header
    df_print: tibble # push to header?
    latex_engine: xelatex #  Available engines are pdflatex [default],
xelatex, and lualatex; The main reasons you may want to use xelatex or
```

```
C:/Users/hstol/Google Drive/SIPA/R - Data Analysis/U6614/Lectures/Lecture1/
> getwd()
[1] "C:/Users/hstol/Google Drive/SIPA/R - Data Analysis/U6614/Lectures/Lecture1"
>
```

```
Data
df1          3 obs. of 1 variable
```

Name	Description	Version
User Library		
abind	Combine Multidimensional Arrays	1.4-5
AER	Applied Econometrics with R	1.2-9
askpass	Safe Password Entry for R, Git, and SSH	1.1
assertthat	Easy Pre and Post Assertions	0.2.1
backports	Reimplementations of Functions Introduced Since R-3.0.0	1.1.7
base64enc	Tools for base64 encoding	0.1-3
bench	High Precision Timing of R Expressions	1.1.1
BH	Boost C++ Header Files	1.72.0-3
bit	Classes and Methods for Fast Memory-Efficient Boolean Selections	4.0.4
bit64	A S3 Class for Vectors of 64bit	4.0.2

R Projects and Directory Structure

Working directory

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

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

```
getwd()
#> [1] "C:/Users/hstol/Google Drive/SIPA/R - Data Analysis/U6614/Lectures/Lectur
```

Function `list.files()` lists all files located in working directory

```
list.files()
```

Note that functions can accept arguments inside of parentheses, but the simple functions shown above do not require any arguments.

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] "C:/Users/hstol/Google Drive/SIPA/R - Data Analysis/U6614/Lectures/Lectur
```

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

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("C:/Users/Harold Stolper/Google Drive/SIPA/R - Data Analysis/Fall 2020/Le
```

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 "lecture2" folder and you want to go up 2 levels to the "Fall 2020" 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 (mine is my name pm)
../	moves up a level
../../	moves up two level

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

What is an "R project"?

- A file that keeps all "project" files organized together:
  - input data, R scripts, analytical results, and figures.
- When you open an R project, your working directory is automatically set to the directory where your R project lives.

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

- We want you to be able to run the R Markdown files (.rmd) used to generate each lecture.
- Sometimes these .rmd files point to certain sub-folders
- You can create or download an R project with directory structure (i.e. organizing files and sub-folders in a particular way).
- That way you'll be able to run .rmd files from your own computer that point to files in sub-folders without making any changes to file-paths.

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

1. Code chunks in R Markdown files
   - *Cmd/Ctrl + Enter*: execute highlighted line(s) within chunk
   - *Cmd/Ctrl + Shift + k*: “knit” entire document
2. R scripts (.r files)
   - **Cmd/Ctrl + Enter**: execute highlighted line(s)
   - **Cmd/Ctrl + Shift + Enter** (without highlighting any lines): run entire script

Assignment in R

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

- `<-` is the assignment operator
    - in other languages `=` is the assignment operator
- code is dense and hard to read, so it's good practice to put a space before and after assignment operator

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

# Create an object b and assign value
b <- "yay!"
b
#> [1] "yay!"
```

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!)

Throughout this course, we’ll be loading data objects to work with as well as assigning values to new objects.

Functions

Functions do things to different objects. They often accept arguments – we say "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 base R functions and handy functions from additional packages.

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.

Assignment 1

Assignment 1: submit an R script via CW before midnight next Monday

See **Assignment1.r** for details.

Complete the assignment by including the necessary code, organized by (sub)question, and use comments for non-code responses.

Submit only your R script through CW using the following file name syntax: **"Assignment1-YOURUNI.r"** via CW

- e.g. "Assignment1-hbs2103.r"

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.
- **Try to keep your code within the margins** to make it more readable.
  - R should know it can keep reading on the next line before executing... unless you break after executable code
- Troubleshooting is a critical skill! Here are some tips and resources:
  - Consult the R script from today's class for examples.
  - Get used to using R's built in documentation by using "?"
  - Use Google liberally to identify functions and find 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.
- Consulting with others is fine! Copying, however, is not the way to learn to code or any language.
  - **Copied assignment submissions will result in a 0 for all parties.**