Getting Started with Data (in R)

Jonathan Moreno-Medina

ECO3253, UTSA

Fall 2022

Why R again?

Why are we learning R? I wanted to learn about social issues...

This class *is* about social issues in economics. But what are those social issues?

- Economic mobility
- Effects of education
- Pollution and climate change
- and so on...

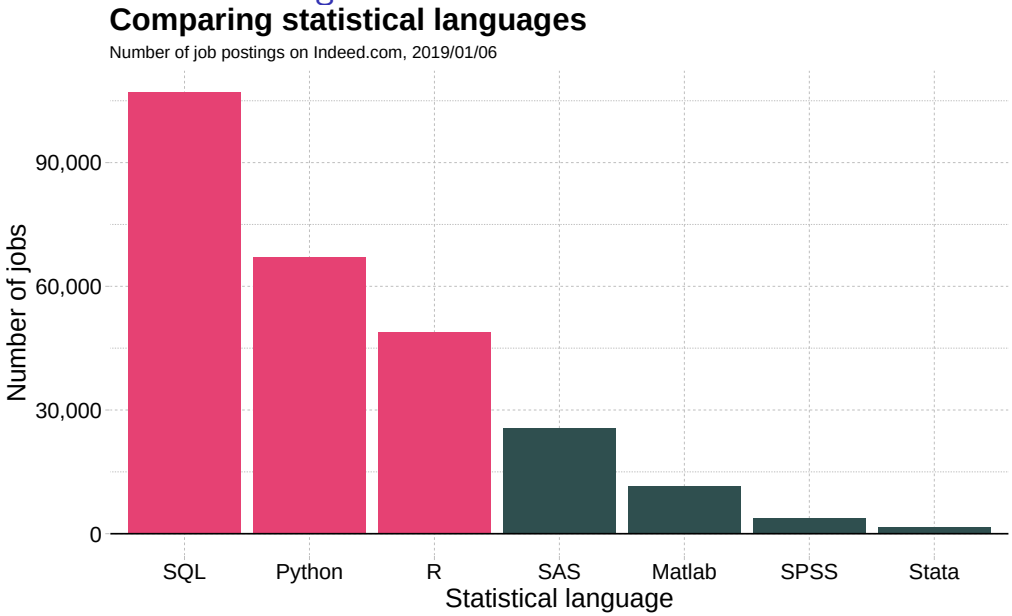
How can we know if going to school increases wages? Or if economic mobility is low or high?

We need to *analyze* data! We can do that analysis by hand... but that would be very time consuming. Or we can use a super calculator with amazing capabilities to explore data, maps, etc: enter **R** and **R Studio**.

Ok, but why R?

- **R** is free and open source!
- **R** has a vibrant online community!
- **R** is very flexible and powerful — adaptable to nearly any task (*e.g.*, correlations, econometrics, spatial data analysis, machine learning, web scraping, data cleaning, website building, teaching.)
- Employers like **R** over alternatives

Added benefits of learning R



Getting Started with Data in R

Basic questions

Before we can start exploring data in R, there are some key concepts to understand first:

1. What are R and RStudio?
2. How do I code in R?
3. What are R packages?

We will cover those 3 points today, and you will start exploring your first dataset based on what we covered!

What are R and RStudio?

We will use R via RStudio. They are not the same!

- R is like a car's engine.
- RStudio is like a car's dashboard.

R: Engine	RStudio: Dashboard
	

What are R and RStudio?

- R: programming language that runs computations
- RStudio: *integrated development environment (IDE)* - interface that add many convenient features and tools.

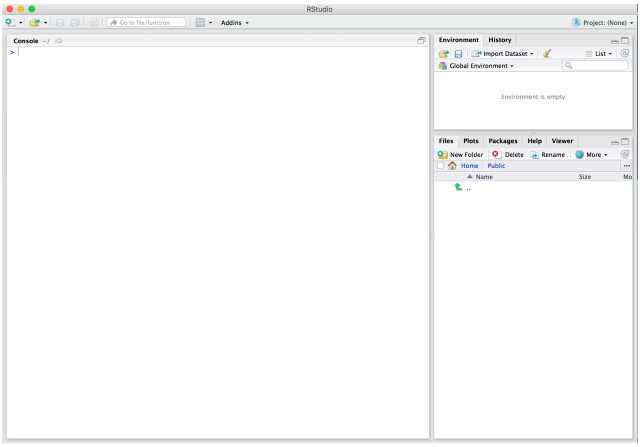
So just as having access to a speedometer, rearview mirrors, and a navigation system makes driving much easier, using RStudio's interface makes using R much easier as well.

R and RStudio: In your computer or in the cloud?

To use R and RStudio, you can:

- install it in your computer (see the book) or ...
- run it on someone else's computer (the cloud!). We will do that, so you don't have to worry about installations.

Using R via RStudio



When open RStudio, you should see this ()

3 panes (panels dividing the screen):

1. The *Console pane*
2. *Files pane*
3. *Environment pane*

We will soon learn what they are for.

Simple coding in R

How do I code in R?

"OK. Now how do I use R?" Unlike other statistical software programs like Excel, STATA, or SAS that provide point and click interfaces, R is an interpreted language

Meaning you have to enter in R commands written in R code. In other words, you have to code/program in R.

That sounds scary if you have not programmed, but turns out **programming in R is fairly simple**

Basic programming concepts and terminology - 1

You'll "learn by doing." Whenever you see the following, it means `computer_code`, not normal text.

How will you learn? **Do the HOMEWORKS with time**... I cannot emphasize this enough!

- Basics:
  - *Console*: Where you enter in commands.
  - *Running code*: The act of telling R to perform an action by giving it commands in the console.
  - *Objects*: Where values are saved in R. In order to do useful and interesting things in R, we will want to *assign* a name to an object. For example we could do the following assignments: `x <- 44 - 20` and `three <- 3`. This would allow us to run `x + three` which would return 27.
  - *Data types*: Integers, doubles/numerics, logicals, and characters.

Basic programming concepts and terminology - 2

- *Vectors*: A series of values. These are created using the `c()` function, where `c()` stands for "combine" or "concatenate." For example: `c(6, 11, 13, 31, 90, 92)`.
- *Factors*: *Categorical data* are represented in R as factors.
- *Data frames*: Data frames are like rectangular spreadsheets: they are representations of datasets in R where the rows correspond to *observations* and the columns correspond to *variables* that describe the observations. We will use this a lot!

Basic programming concepts and terminology - 3

- *Conditionals*:
  - Testing for equality in R using == (and not = which is typically used for assignment). Ex: `2 + 1 == 3` compares `2 + 1` to 3 and is correct R code, while `2 + 1 = 3` will return an error.
  - Boolean algebra: `TRUE/FALSE` statements and mathematical operators such as < (less than), <= (less than or equal), and `!=` (not equal to).
  - Logical operators: `&` representing "and" as well as | representing "or." Ex: `(2 + 1 == 3) & (2 + 1 == 4)` returns `FALSE` since both clauses are not `TRUE` (only the first clause is `TRUE`). On the other hand, `(2 + 1 == 3) | (2 + 1 == 4)` returns `TRUE` since at least one of the two clauses is `TRUE`.
- *Functions*, also called *commands*: Functions perform tasks in R. They take in inputs called *arguments* and return outputs. You can either manually specify a function's arguments or use the function's *default values*.

Errors, warnings, and messages

What is scare for most new R and RStudio users? *errors*, *warnings*, and *messages*!

R reports errors, warnings, and messages in a glaring red font, which makes it seem like it is scolding you. However, seeing red text in the console is not always bad.

R will show red text in the console pane in three different situations:

Errors

- **Errors**: When red text is a legitimate error, it will be prefaced with "Error in..." and try to explain what went wrong.
- Generally when there's an error, the code will not run.
  - For example, we'll see in Subsection @ref(package-use) if you see `Error in ggplot(...) : could not find function "ggplot"`: it means that the `ggplot()` function is not accessible because the package that contains the function (`ggplot2`) was not loaded with `library(ggplot2)`. Thus you cannot use the `ggplot()` function without the `ggplot2` package being loaded first.

Warnings

- **Warnings**: When red text is a warning, it will be prefaced with "Warning:" and R will try to explain why there's a warning.
- Generally your code will still work, but with some caveats. For example, we will see later that if you create a scatterplot based on a dataset where one of the values is missing, you will see this warning: `Warning: Removed 1 rows containing missing values (geom_point)`. R will still produce the scatterplot with all the remaining values, but it is warning you that one of the points isn't there.

Messages

- **Messages**: When red text doesn't start with either "Error" or "Warning", it's *just a friendly message*.
- You'll see these messages when you load *R packages* in the upcoming section or when you read data saved in spreadsheet files with the `read_csv()` function. These are helpful diagnostic messages and they don't stop your code from working. Additionally, you'll see these messages when you install packages too using `install.packages()`.

What to do if you see red text?

Don't panic!. It doesn't necessarily mean anything is wrong. Rather:

- If the text starts with "Error", figure out what's causing it. Think of errors as a red traffic light: something is wrong!
- If the text starts with "Warning", figure out if it's something to worry about. For instance, if you get a warning about missing values in a scatterplot and you know there are missing values, you're fine. If that's surprising, look at your data and see what's missing. Think of warnings as a yellow traffic light: everything is working fine, but watch out/pay attention.
- Otherwise the text is just a message. Read it, wave back at R, and thank it for talking to you. Think of messages as a green traffic light: everything is working fine.

Tips on learning to code

Learning to code/program is like learning a foreign language: can be daunting and frustrating at first. But as learning a foreign language, effort + not afraid to make mistakes = anybody can learn.

Tips:

- **Remember that computers are not actually that smart**: You have to tell a computer everything it needs to do. Instructions should not have mistakes nor be ambiguous.
- **Take the "copy, paste, and tweak" approach**: Especially when learning your first programming language, it is often much easier to taking existing code that you know works and modify it to suit your ends, rather than trying to write new code from scratch.
- **The best way to learn to code is by doing**: Do the homeworks with time! This are opportunities for you to try and get familiar with coding.
- **Practice is key**: Just as the only method to improving your foreign language skills is through practice, practice, and practice.

R Packages

What are R packages?

Usual confusion with many new R users: packages. They give extra functionality to R (extra functions, data, and documentation). They are written by a world-wide community of R users and can be downloaded for free from the internet.

For example, among the many packages we will in the course are the `ggplot2` package for data visualization. We will see more on how to use `ggplot2` and `dplyr` package (helpful to edit and modify data - or wrangling).

R packages = apps for a phone:

R: A new phone	R Packages: Apps you can download
	

Packages as Apps

How do you open an app like Instagram on your phone?

1. *Install the app*: New phone does not have Instagram app -> you need to download it. You do this **once** and you're set. You might do this again in the future any time there is an update to the app.
2. *Open the app*: After you've installed Instagram, you need to open the app.

Once Instagram is open on your phone, you can then proceed to share your photo. Almost the same for an R package. You need to:

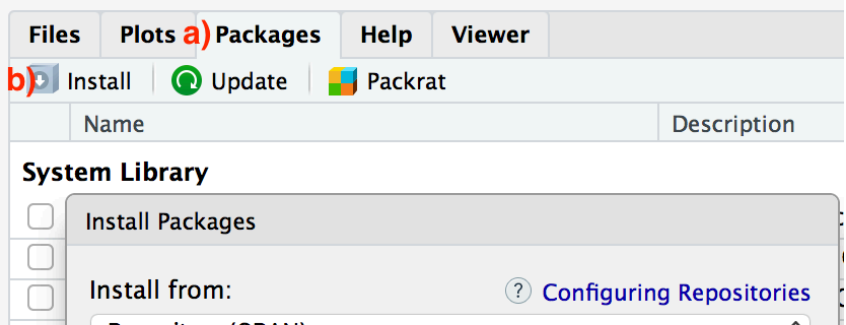
1. *Install the package*: This is like installing an app on your phone. Most packages are not installed by default when you install R and RStudio. If you want to use a package for the first time, you need to install it first. Once you've installed a package, unlikely you need to install again.
2. *"Load" the package*: "Loading" a package is like opening an app on your phone. Packages are not "loaded" by default when you start RStudio on your computer; you need to "load" each package you want to use every time you start RStudio.

Let's now show you how to perform these two steps for the `ggplot2` package for data

Package installation

There are two ways to install an R package. For example, to install the `ggplot2` package:

1. **Easy way**: In the Files pane of RStudio:
   a) Click on the "Packages" tab
   b) Click on "Install"
   c) Type the name of the package under "Packages (separate multiple with space or comma):" In this case, type `ggplot2`
   d) Click "Install"



Excercise - Install Packages

Install packages `dplyr`, `nycflights13`, and `knitr` packages. This will install the earlier mentioned `dplyr` package, the `nycflights13` package containing data on all domestic flights leaving a NYC airport in 2013, and the `knitr` package for writing reports in R.

Package loading

Recall: after installing a package, you need to "load" it (open it). We do this by using the `library()` command. For example, to load the `ggplot2` package, run the following code in the Console pane. What do we mean by "run the following code"? Either type or copy & paste the following code into the Console pane and then hit the enter key.

After running above code, do you see blinking cursor returns next to the > "prompt" sign? YES: success! `ggplot2` package is now loaded and ready to use; NO: got a red "error message" that reads...

```
Error in library(ggplot2) : there is no package called 'ggplot2'
```

... it means that you didn't successfully install it. In that case, go back to the previous subsection "Package installation" and install it.

Excercise - Load Packages

Load packages `dplyr`, `nycflights13`, and `knitr` packages as well by repeating the above steps.

Package use

One extremely common mistake new R users make when wanting to use particular packages is that they forget to "load" them first by using the `library()` command we just saw. Remember: *you have to load each package you want to use every time you start RStudio.* If you don't first "load" a package, but attempt to use one of its features, you'll see an error message similar to:

```
Error: could not find function
```

R is telling you that you are trying to use a function in a package that has not yet been "loaded." Almost all new users forget do this when starting out, and it is a little annoying to get used to. However, you'll remember with pratice.

Hands-on exercise!

Explore your first dataset

Let's put everything we've learned so far into practice and start exploring some real data! These "spreadsheet"-type datasets are called *data frames* in R; we will focus on working with data saved as data frames throughout this course.

Step 1: Load all the packages needed for this exercise (assuming you've already installed them).

`nycflights13` package

Are there ways that we can avoid having to deal with these flight delays? We'd all like to arrive at our destinations on time whenever possible. This package contains five data sets saved in five separate data frames with information about all domestic flights departing from New York City in 2013. These include Newark Liberty International (EWR), John F. Kennedy International (JFK), and LaGuardia (LGA) airports:

- `flights`: Information on all 336,776 flights
- `airlines`: A table matching airline names and their two letter IATA airline codes (also known as carrier codes) for 16 airline companies
- `planes`: Information about each of 3,322 physical aircraft used.
- `weather`: Hourly meteorological data for each of the three NYC airports. This

Conclusion

We've given you what we feel are the most essential concepts to know before you can start exploring data in R. Is this chapter exhaustive? Absolutely not. To try to include everything in this chapter would make the chapter so large it wouldn't be useful!

Additional resources

If you are completely new to the world of coding, R, and RStudio and feel you could benefit from a more detailed introduction, we suggest you check out ModernDive co-author Chester Ismay's Getting used to R, RStudio, and R Markdown short book [@usedtor2016], which includes screencast recordings that you can follow along and pause as you learn. Furthermore, there is an introduction to R Markdown, a tool used for reproducible research in R.