ECON 340
Economics Research Methods

Div Bhagia

Lecture 6: Getting Started with R

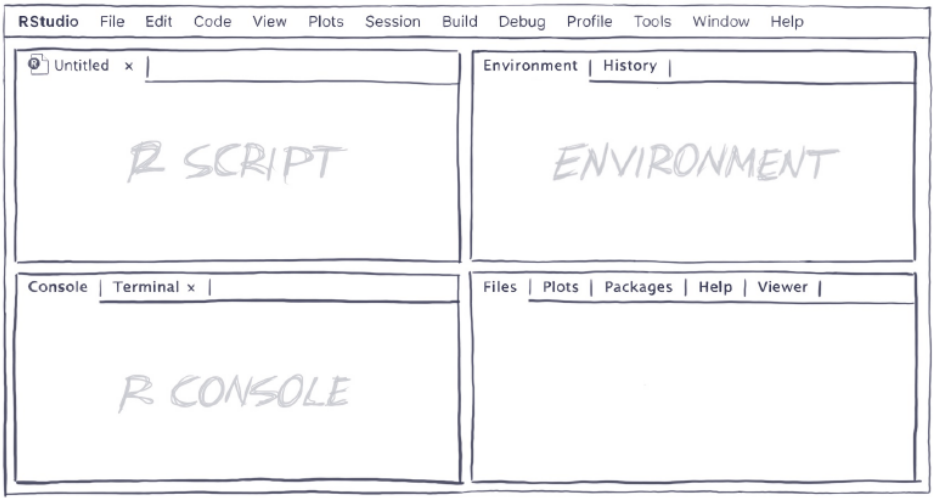
Before we begin

- Make sure you have R and RStudio installed on your computer
- Let's create a new folder on our computers, you can call it `Econ340_R`
- Now let's download the dataset "caschool.csv" from our Dropbox folder and save it in this folder.

About R and R Studio

- R is an *open-source* language designed for statistical computing
- Numerous add-on packages are available for R
- Once you install R, you will have base packages installed. We will also use other packages and install them as we go.
- RStudio IDE is a set of integrated tools designed to use R more easily

Interface


RStudio File Edit Code View Plots Session Build Debug Profile Tools Window Help
Untitled x
R SCRIPT
Environment | History |
ENVIRONMENT
Console | Terminal x |
R CONSOLE
Files | Plots | Packages | Help | Viewer |

Getting Started

- To get started, let's create a new R Script:
  File → New File → R Script
- This should create a new untitled file in your window.
- Save this file to the `Econ340_R` folder:
  File → Save as → Type *getting_started* → Save
- We will write all of our code in this script and execute it using the Run button on the top right

Installing Packages

- Let us now install our first package `TidyVerse`
- TidyVerse is a collection of R packages that share an underlying design philosophy, grammar, and data structures
- You only need to install a package once, so no need to do it again if you installed `TidyVerse` before
- To install a package: Tools → Install Packages → Install from Repository (CRAN) → Type *TidyVerse*

Loading Packages

- Once a package is installed you need to load it before you can use it, so at the top of your R script put the following command:

```
library(tidyverse)
```

File Management

- It is good practice to keep all files related to a project in one folder
- Essentially you want R to use this folder as your *working directory*
- Working directory is the folder where R will save files and retrieve files from

How to set the working directory

Three ways to set the working directory:

- Option 1: Initialize R from this folder: click on the R script *in* the folder to open RStudio
  - Note: This doesn't work if RStudio is already running so quit it and then restart it again by clicking on the R script in the folder.
- Option 2: Open the folder under *Files* on bottom-left in R Studio and click on: More → Set As Working Directory
- Option 3: Manually tell R the directory (path) of this folder

Option 3: Manually setting the path

```
# Manually set directory
setwd("/Users/dbhagia/myfolder")
```

- On a Mac, you can right click on any folder and click *Get Info* to get the path
- On a Windows computer, you can find the location in the address bar on the top (replace "\" with "/")
- Alternatively, you can find your path by opening the folder under Files on bottom-left in R Studio and clicking: More → Copy Folder Path to Clipboard

Importing Data

- We can import data from all kinds of format in R
- Some of the common formats in which data are stored are .xls, .xlsx, .csv
- .xls and .xlsx are Microsoft Excel's native formats, however often data is stored in .csv files as the they are simpler
- Data in R format has extension .rda or .Rdata

So Far

- You should have a folder on your computer with "caschool.csv" and "getting_started.R".
- In your R script, you shoud have the following commands

```
library(tidyverse)
```

- Run your code and make sure you do not get an error
- You can select it and Run or use the dropdown next to Run and click Run All or use Source. *(If you just click Run, it will only execute the current line)*

Importing Data

To import the dataset:

```
data <- read.csv("caschool.csv")
```

This data set consists of information on 420 elementary school districts in California from 1998-1999.

You can find the description of the variables in the accompanying codebook.

Assignment Operator

- The symbol $<-$ stands for the assignment operator
- You can use shorcut Alt + - (Windows) or Option + - (Mac)

```
data <- read.csv("caschool.csv")
```

- In the above code, we created a new object `data` and assigned the dataset we loaded using `read.csv()` to this object
- If your command ran succesfully, a new object called `data` should appear under the `Environment` (top-right)

Assignment Operator

```
umm <- 2
umm <- "Hello"
umm
```

```
## [1] "Hello"
```

- The first line of code above creates a new object `umm` and assigns value 2 to it.
- The next line of code takes the existing object `umm` and assigns a new value to it.
- Third line displays what is stored in object `umm`

Exploring Data

- Before doing anything too complicated, let's get a feel of the data
- You can double click on the object `data` (or whatever you named it) under the `Environment` (top-right)
- Or single click on the arrow on left of data to see the structure of the data (alternatively use `str()` command)
- Some variables are stored as characters (`chr`), some as integers (`int`)

Exploring Data

To see the list of variables:

```
ls(data)
```

To summarize all variables in a dataset:

```
summary(data)
```

To summarize a particular variable:

```
summary(data$avginc)
```

R Syntax

- To call a variable we need to use `data$var_name` as multiple data objects can be loaded in R at the same time
- Say we want to find the average math score

```
# Average math score
mean(data$math_scr)
```

```
## [1] 653.3426
```

- Can write comments using #
- Try `median()` and `sd()` as well

Help in R

- To learn more about any function or it's arguments, one can type `?function_name` in the console.
- For instance,

```
?mean
```

- This will open up the documentation for this section in the Help window (bottom-right).

More on R Syntax

Nearly everything we do in R fits into one of three categories:

- Create or overwrite an object (using the assignment operator $<-$)
- Apply functions to objects
- Look at objects

When we input arguments in the order the function lists them, no need to explicitly specify what argument we are referring to.

Exercise for you

- Create a new object called `mean_comp` that contains the mean of variable `computer`
- Create a new variable in the data frame `data` called `mean_comp` that contains the mean of `computer`

Creating Objects

- You can create a new vector that contains numbers 1-5:

```
x1 <- c(1, 2, 3, 4, 5)
```

- To create `x2` that contains 0 and 100 on the left and right of `x1`, respectively

```
x2 <- c(0, 1, 2, 3, 4, 5, 100)
```

- Alternatively

```
x2 <- c(0, x1, 100)
```

Another exercise for you

- Use `?mean()` to figure out what the following code does:

```
mean(x2, 0.25)
```

More on R Syntax

- Note that `mean(x2, 0.25)` is equivalent to

```
mean(x = x2, trim = 0.25)
```

```
## [1] 3
```

- We can even write:

```
mean(trim = 0.25, x = x2)
```

```
## [1] 3
```

If we follow the order, no need to explicitly refer to which argument and we can simply write `mean(x2, 0.25)`.

Few Last Words

- Best way to learn a programming language is by using it
- It is definitely challenging with a steep learning curve, but it is rewarding in the end
- Internet is your friend (like you didn't know that!)
- ChatGPT is helpful, but you still need to understand the programming language to reap its benefits.
- Have fun while you are at it!
- Next class: really cool things you can do in R :)