Programming for Professional Research Using R

Session 1

March 27, 2025

Introduction

Purpose | Learning R

Purpose of this course: Put you on the right track to use R for data or policy analysis.

- As data work has become near-ubiquitous in the research/policy world, so have basic tasks like aggregating, analyzing, summarizing, and visualizing data.
- **The vast majority** of research assistant/analyst (RA) work consists of cleaning and constructing datasets for analysis
- **Entry-level RA positions rarely require complex econometric/regression skills**

Introduction

Purpose | Learning R

You should think of learning R like learning a language.

- Taking a six-hour course won't make you proficient in it
- If you don't practice it, you'll forget it
- Solution — **Find ways to use R in your life, either personally or professionally**

Why R?

Jobs | Beautiful Tables | Beautiful Graphs | Beautiful Maps

- Many entry-level research jobs in policy, economic, development, or political science institutions now expect quantitative work using Stata, R, or Python
- Coding skills make you more valuable in any position — data adds value to nearly every kind of research!

Why R?

Jobs | **Beautiful Tables** | Beautiful Graphs | Beautiful Maps

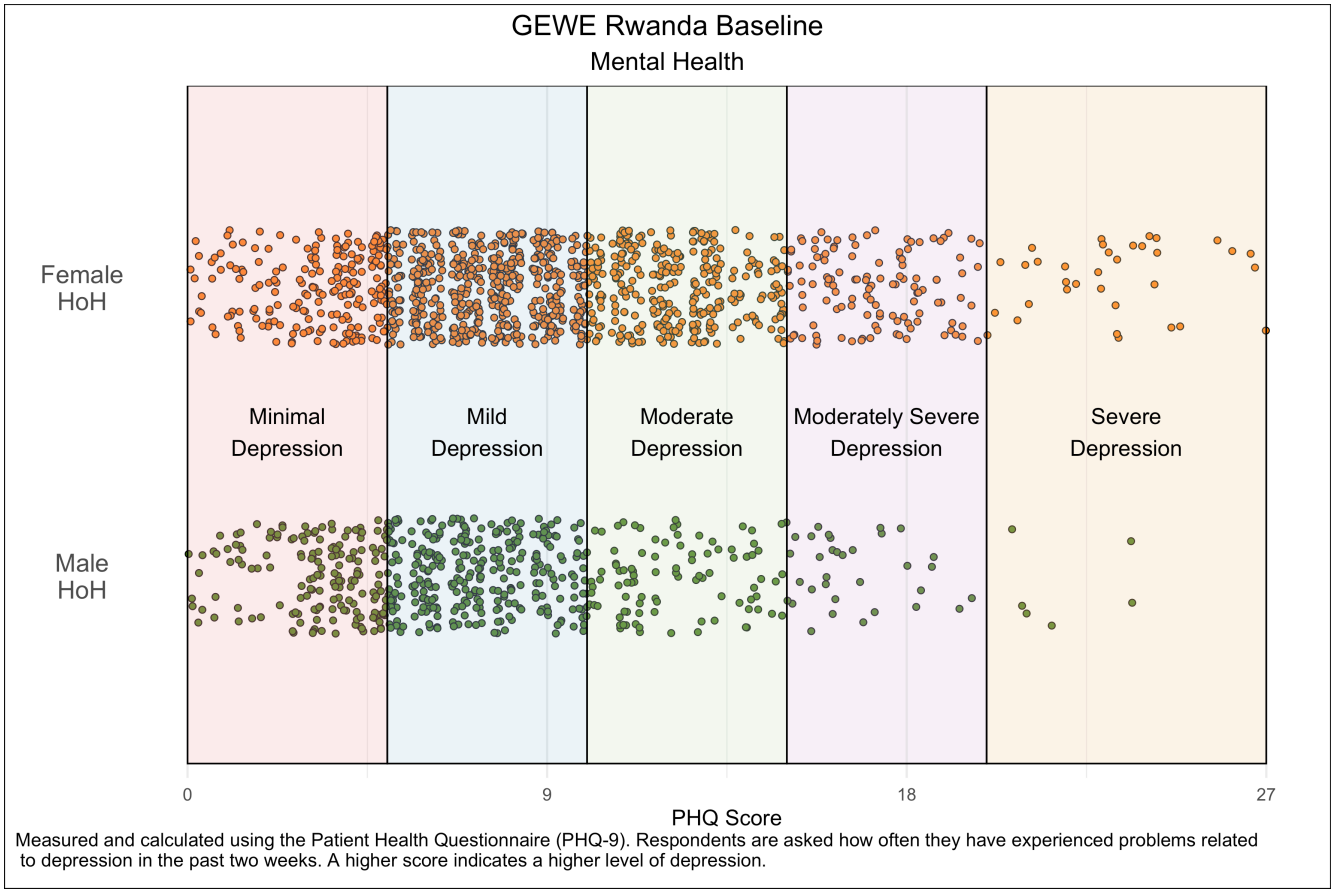
2021 Expected vs. Actual Fantasy Points

Top 40 Running Backs

Player	Team	Expected FP	Expected FP Rank	Actual FP	Actual FP Rank
Jonathan Taylor	IND	301.20	RB 1	324.20	RB 1
Najee Harris	PIT	292.67	RB 2	228.10	RB 4
Joe Mixon	CIN	251.89	RB 3	254.80	RB 3
Leonard Fournette	TB	232.12	RB 4	221.10	RB 5
Ezekiel Elliott	DAL	225.03	RB 5	214.76	RB 6
Austin Ekeler	LAC	217.88	RB 6	263.90	RB 2
Antonio Gibson	WAS	216.49	RB 7	186.70	RB 11

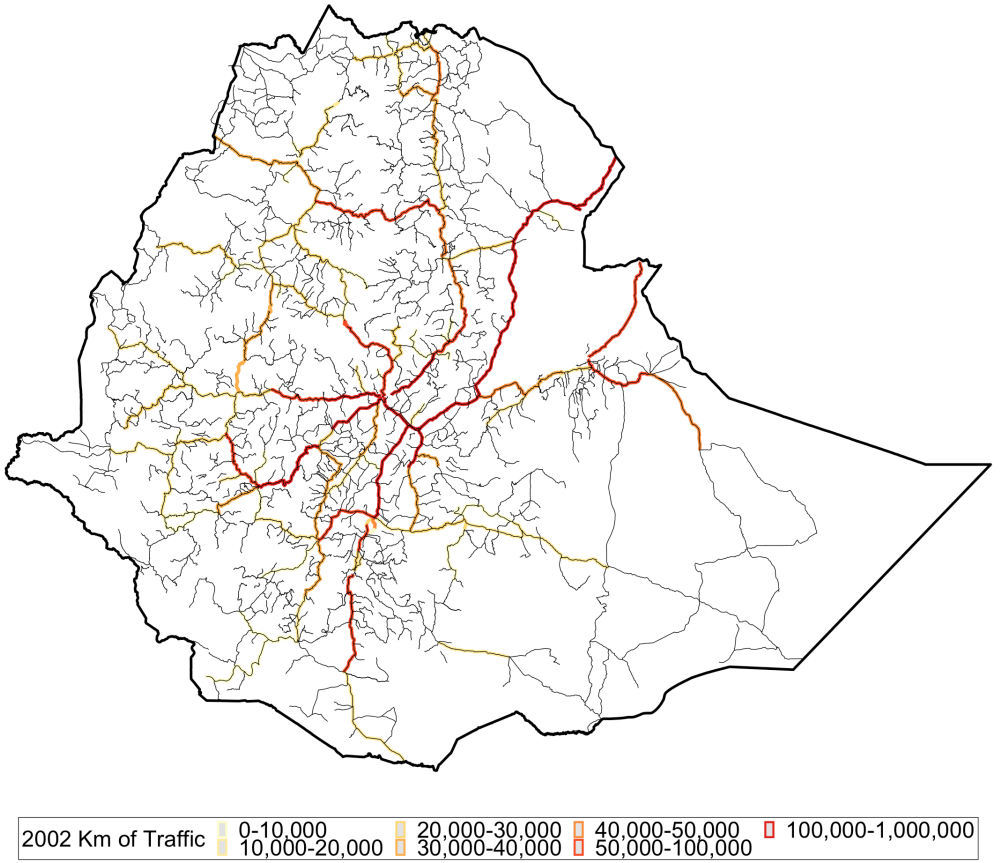
Why R?

Jobs | Beautiful Tables | **Beautiful Graphs** | Beautiful Maps



Why R?

Jobs | Beautiful Tables | Beautiful Graphs | **Beautiful Maps**



Today

- Learn how to:
    - Think as a coder
    - Identify the basic components of data analysis
    - Set up your environment to use R and RStudio
    - Identify and address basic errors in your R setup
- Be introduced to:
    - The R coding language
    - The building blocks of coding in R: scalars, vectors, lists, and data frames

Think As A Coder

Recipe Analogy

- Your data are your ingredients
- Your script is your recipe
- Your output is your... cake? Whatever you're making

What this means:

- **Make sure you have all of the ingredients you need.**
- **Follow every step of the recipe — you can't skip any steps!**
- If you mess up somewhere, you have to start from scratch to make sure that you get the correct outcome.

Identify the Basic Components of Data Analysis

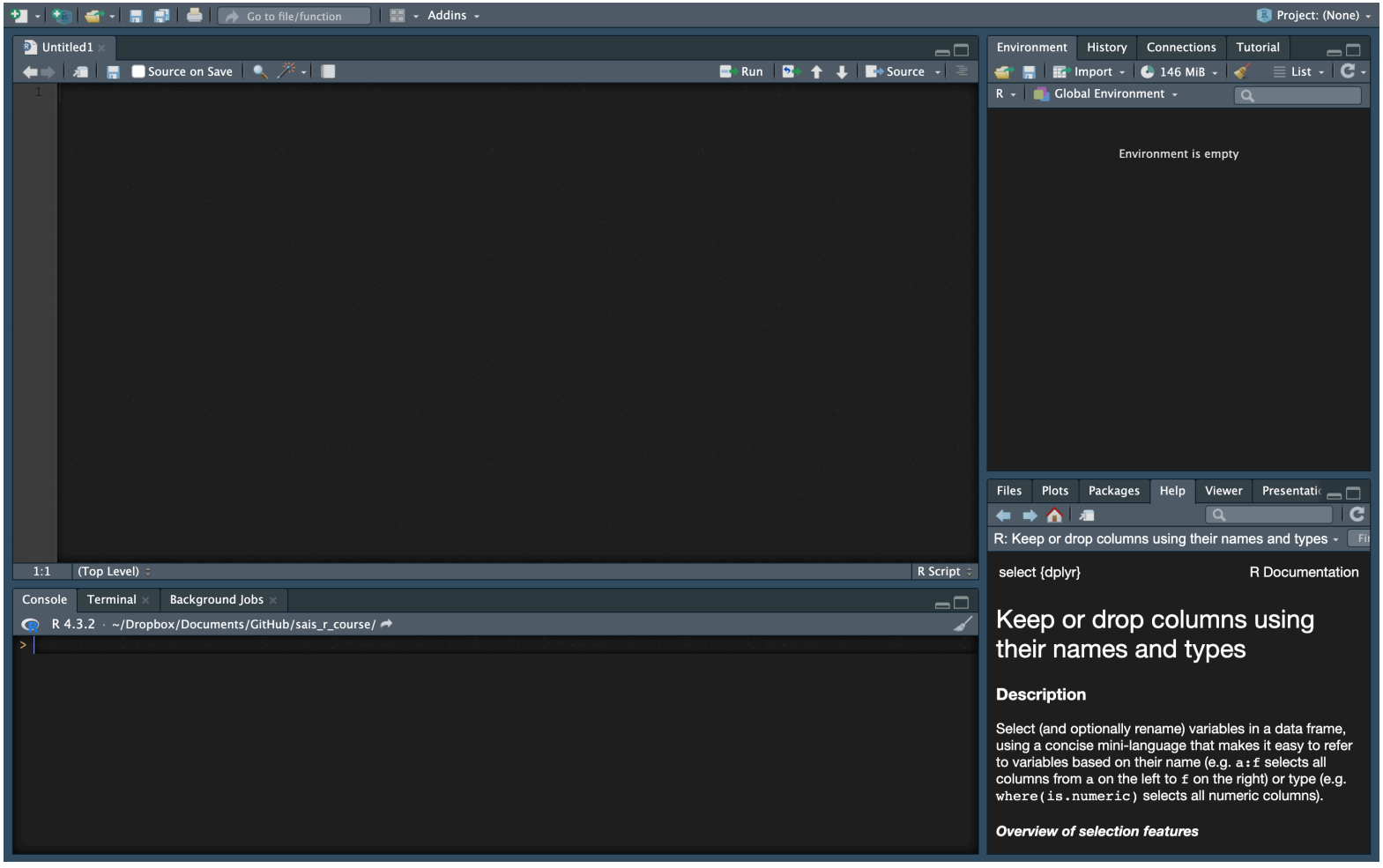
Three main components to your data analysis:

- Data
- Code
- Outputs

Questions to ask:

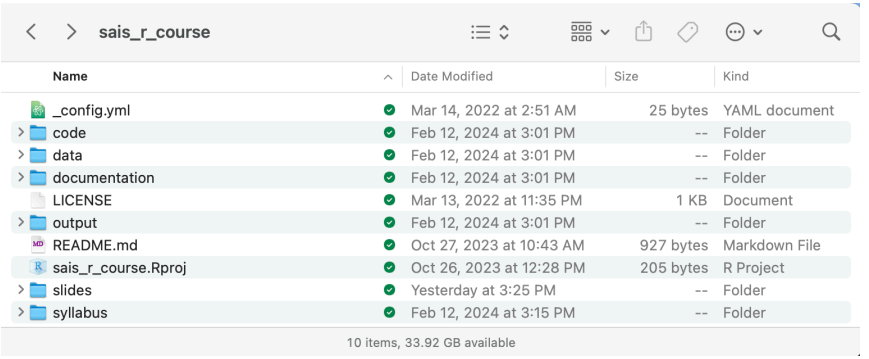
- Where is the file stored?
- In which format is the file?
- How easy is it for you/other people to access it?
- Is it intuitive for you to navigate to the file?

Your RStudio Environment



Setting Up Your RStudio Environment

Setting Up Your Data Structure | Setting Up Your Script



- Code
- Data
  - Raw Data
  - Intermediate Data
- Outputs
- (Optional) Documentation
- (Optional) Presentations/Slides

Setting Up Your RStudio Environment

Setting Up Your Data Structure | **Setting Up Your Script**

Defining your code's purpose — What are its inputs? What are its outputs?

```
## Programming for Professional Research Using R -- Session 1

### Session Description

  # Basic R Use - Introduction to R and RStudio, How to import/export data
  # Using Basic Tidyverse Functions - how to subset, mutate, summarize, and reshape data
  # Using R in a Collaborative Setting - Introduction to downloading and publishing data using
  # GitHub
```

Set up your packages — Which user-created functions do you plan to use?

```
## 1. Setup ----

  ### Packages Random

# NOTE -- Unlike using library(), the 'pacman::p_load()' function installs the package if it is
# already not present in the user's R environment.

if(!require(pacman)) install.packages("pacman")

pacman::p_load(tidyverse, data.table, janitor, usethis)
```

Import your data — Which data frames do you need to fulfill the script's purpose?

```
## 2. Import Data ----

# This is session 1 so we're going to import the data from Dropbox:

if(!(file.exists("data/final/wvs_values_norms_data.csv"))) { # Checks whether the data has been
                                                             # downloaded already
    usethis::use_zip(
        "https://www.dropbox.com/scl/fo/vnxjbqyq1g9z368coh1vq/h?rlkey=zc66o2ll7613b5e9ipp915ynk6dl=1"
    )

}

norms_values_data <- data.table::fread( # Other options are base R's read.csv() and data.table::fread()
    "data/final/wvs_values_norms_data.csv", na.strings = ""
)
```

R Basics

Scalars | Vectors | More Vectors | Selecting Vector Elements

This is a scalar:

```
a ← 2
```

It's an object (a) with a single value (2). Notice that we assign the value 2 to a using an arrow ←.

R Basics

These are vectors:

```
vector ← c(1, 2, 5)
vector
```

```
## [1] 1 2 5
```

```
vector ← 2:6
vector
```

```
## [1] 2 3 4 5 6
```

R Basics

These are more vectors:

```
vector ← seq(2, 3, by = 0.5)
vector
```

```
## [1] 2.0 2.5 3.0
```

Vectors take multiple values. Notice that vectors have a specific **order**: `c(1, 2)` is not the same as `c(2, 1)`.

R Basics

Scalars | Vectors | More Vectors | Selecting Vector Elements

```
## By Position
x[4]       # Fourth element
x[-4]      # Everything but the fourth element
x[2:4]     # Elements two to four
x[-(2:4)]  # Everything but elements two to four
x[c(1, 5)] # Elements one and five
```

```
# By Value
x[x == 10]           # Elements which are equal to 10
x[x < 0]             # Elements that are less than zero
x[x %in% c(1, 2, 5)] # Elements in the set 1, 2, 5
```

R Basics

Lists | Some Classes | Data Frames | Assignment vs. Printing

This is a list:

```
y ← list("a", 1, "b")
```

Lists are different from vectors in one key manner: **they can take objects of different classes**. Vectors can only take objects of the same class.

R Basics

Lists | Some Classes | Data Frames | Assignment vs. Printing

Here are some classes in R:

```
a ← "a"     # Character
b ← 1       # Integer
c ← 1.2     # Numeric
d ← 1 + 2i  # Complex
e ← TRUE    # Logical
```

R Basics

This is a data frame:

```
df ← data.frame(
    first_name = c("Mark", "Mary", "July"),
    last_name  = c("Smith", "John", "Sanchez"),
    age        = c(21, 34, 55)
)
```

At their core, data frames are just a group of named vectors of the same length. In practice, we visualize data frames as **tables** where each vector is a **column**, or variable, and each group of nth elements of the vectors is a **row**, or observation.

R Basics

KEY — We can classify the code we write in our script into two general categories:

1- Assignment code. We are **assigning** a value to an object:

```
a ← 2
```

2- Printing code. We are **printing** an object's value:

```
a
```

```
## [1] 2
```

```
print(a)
```

```
## [1] 2
```

Only 2- results in something appearing in your console. When you assign a value to an object, nothing prints in the console.

R Basics

NOTE — R works in a manner that allows to write a specific function over multiple lines. This is called a **code chunk**.

This:

```
a ← mean(c(seq(1, 4, by = 0.5)))

a
```

```
## [1] 2.5
```

Is the same as this:

```
a ← mean(
    c(
        seq(1, 4, by = 0.5)
    )
)

a
```

```
## [1] 2.5
```

R Basics

Functions | Packages

Like in mathematics, a function is an expression or rule that takes an input, or *argument*, and returns an *output*. A function is any `f` such that `f(x) = y`.

In R, functions take the following form:

```
# The function is called
sum_function ← function(x, y) { # x and y are the function's arguments
    x + y # Within the brackets {}, we define the function's output
}

sum_function(2, 3)
```

```
## [1] 5
```

The vast majority of using R is done using functions, either included with R or written by external users. For instance, to find the sum of 2 and 3, we can use the "base R" function `sum()`.

R Basics

Functions | Packages

A package is a bundle of functions created by external users. They innovate and, in many cases, improve upon base R functions. Most work in R is done using functions from external packages.

We will look at how to install and load packages shortly. Examples of commonly-used packages are `dplyr`, `tidyr`, `stringr`, `ggplot2`, and `data.table`.

R Basics

You can click anywhere in the code chunk and click "run" or Cmd+Enter (Mac)/Ctrl+Enter (Windows), and the whole chunk will run.

KEY — If you only select a portion of the code chunk and run that, then R will identify the chunk as unfinished and refuse to let you do anything else until you've "completed" it.

Coding Set Up

Packages are groups of user-created functions that help us accomplish tasks that would be harder/impossible using base R functions.

Easy

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

Better

The `pacman` package installs packages if they aren't installed yet, loads them otherwise

```
if(!require(pacman)) install.packages("pacman")
pacman::p_load(tidyverse)
```

File paths help R identify where the files you want to use are located.

You want your code to be **reproducible** and **easy to use by other people**

Simple solution: Create an `.rproj` file that people can open to access your R environment

Better solution:

```
# Set User (this allows us to use fixed file paths but to adapt them
# for multiple possible users)
      # 1 - Marc-Andrea Fiorina
      # 2 - Enter here if needed

user ← 1

if(user == 1) {
    # Absolute file path
    main_filepath ← "/Users/marc-andreafiorina/Dropbox/SAIS R Course/"
}

# Notice the relative file paths
data_filepath ← paste0(main_filepath, "data/")
```

Importing Data

Easiest file type to import into R is a .csv file. But you can also import .xlsx, .dta (Stata), etc.

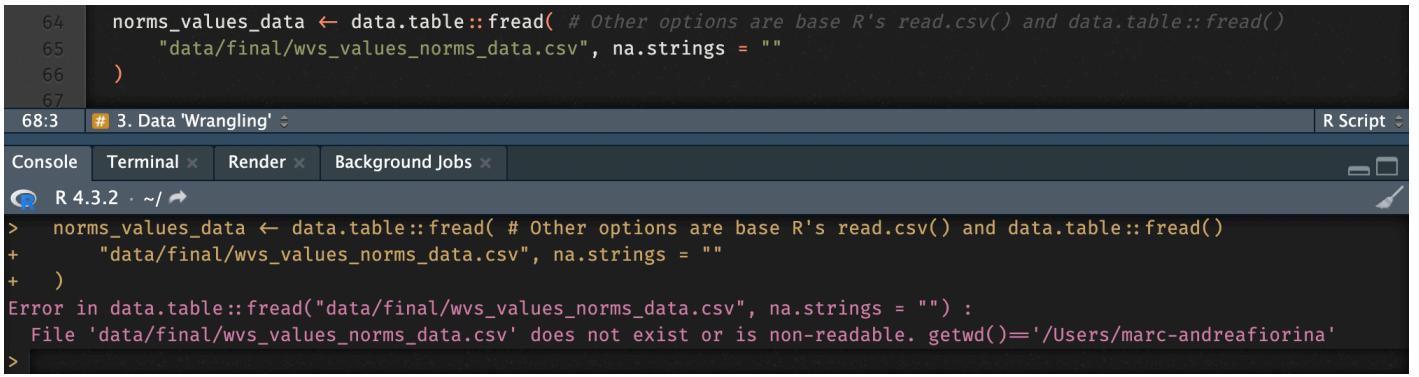
- Easy → `read.csv()`
- Harder (faster) → `data.table::fread()`

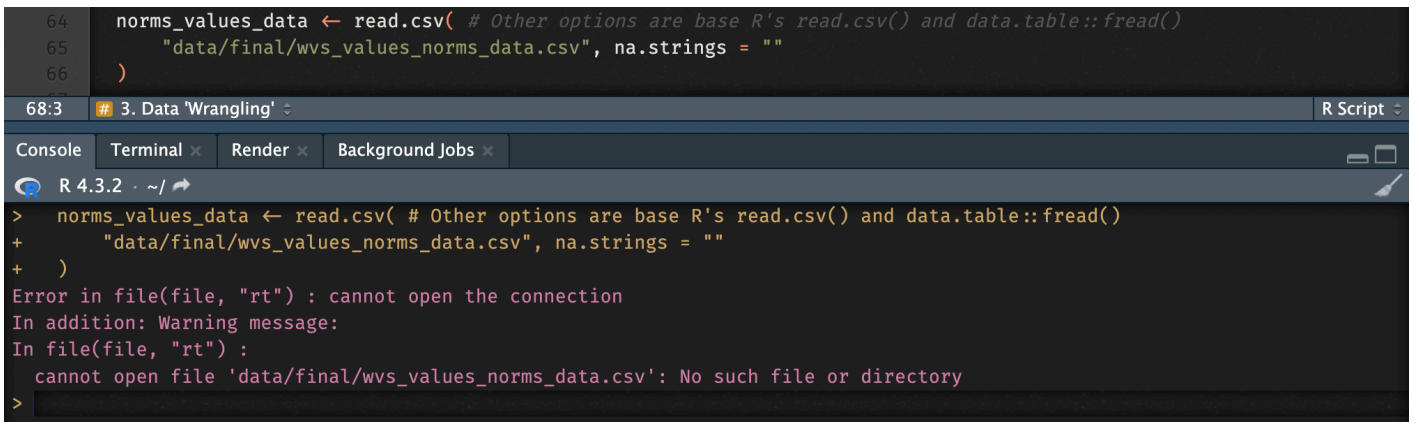
```
norms_values_data ← data.table::fread(
    "raw/wvs_values_norms_data.csv",
    na.strings = ""
)
```

Note the construction of the "harder" method, with the " `::` ". This is used to refer to the package from which the function originates. The structure is `package::function()`.

Basic Issues in RStudio

Data Not Loading





Basic Issues in RStudio

Data Not Loading

Check:

- Working directory — `getwd()` or check the top of the RStudio console.
- File path — are there any typos? Is your file where you expect it to be?

Solutions:

- If you were provided with an `.Rproj` file alongside your script, make sure that you opened the project.
- Modify the working directory using `setwd()` or correct the file path if need be!

Basic Issues in RStudio

Function Not Found

Basic Issues in RStudio

Function Not Found

Check:

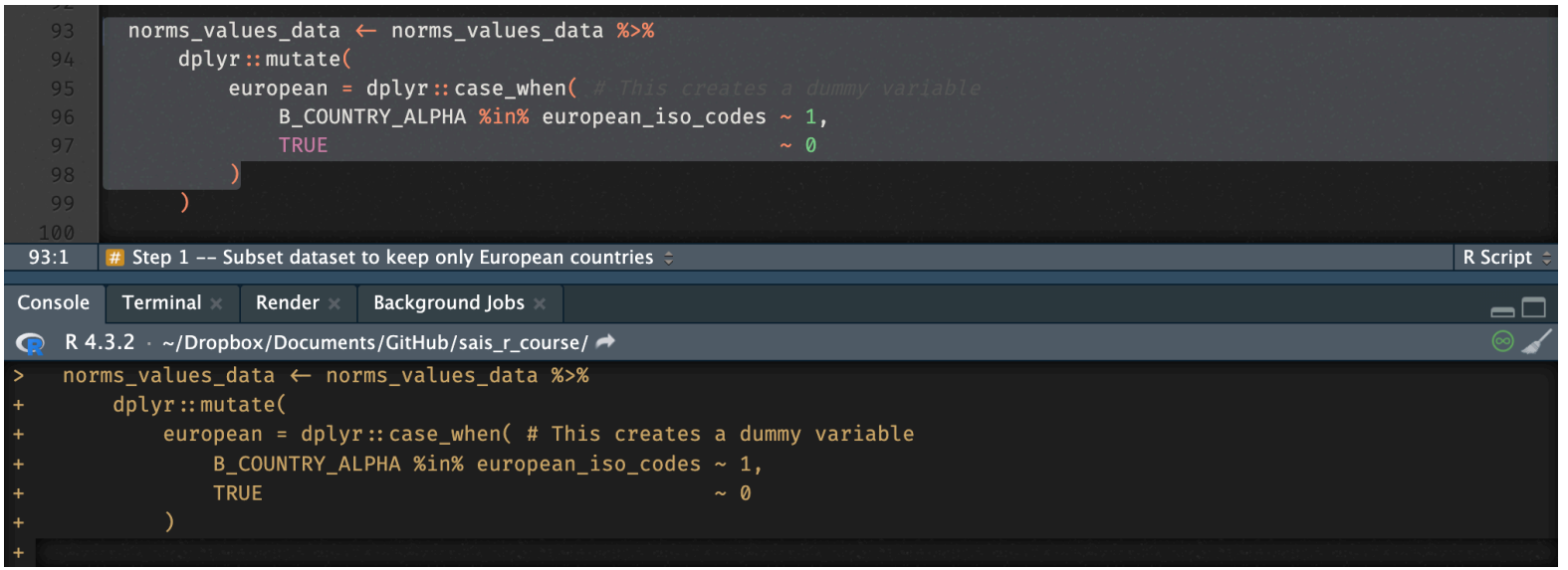
- From which package the function comes. You can do this using `?? FUNCTION NAME` (e.g. `?? tabyl`) or through a Google search.
- That the package is (1) installed in your environment and (2) loaded. Having the package installed isn't sufficient!

Solutions:

- If the package isn't installed, use `install.packages("tabyl")`. If the package isn't loaded, use `library(tabyl)` or `pacman::p_load(tabyl)`.

Basic Issues in RStudio

Code Not Running



Basic Issues in RStudio

Code Not Running

Check:

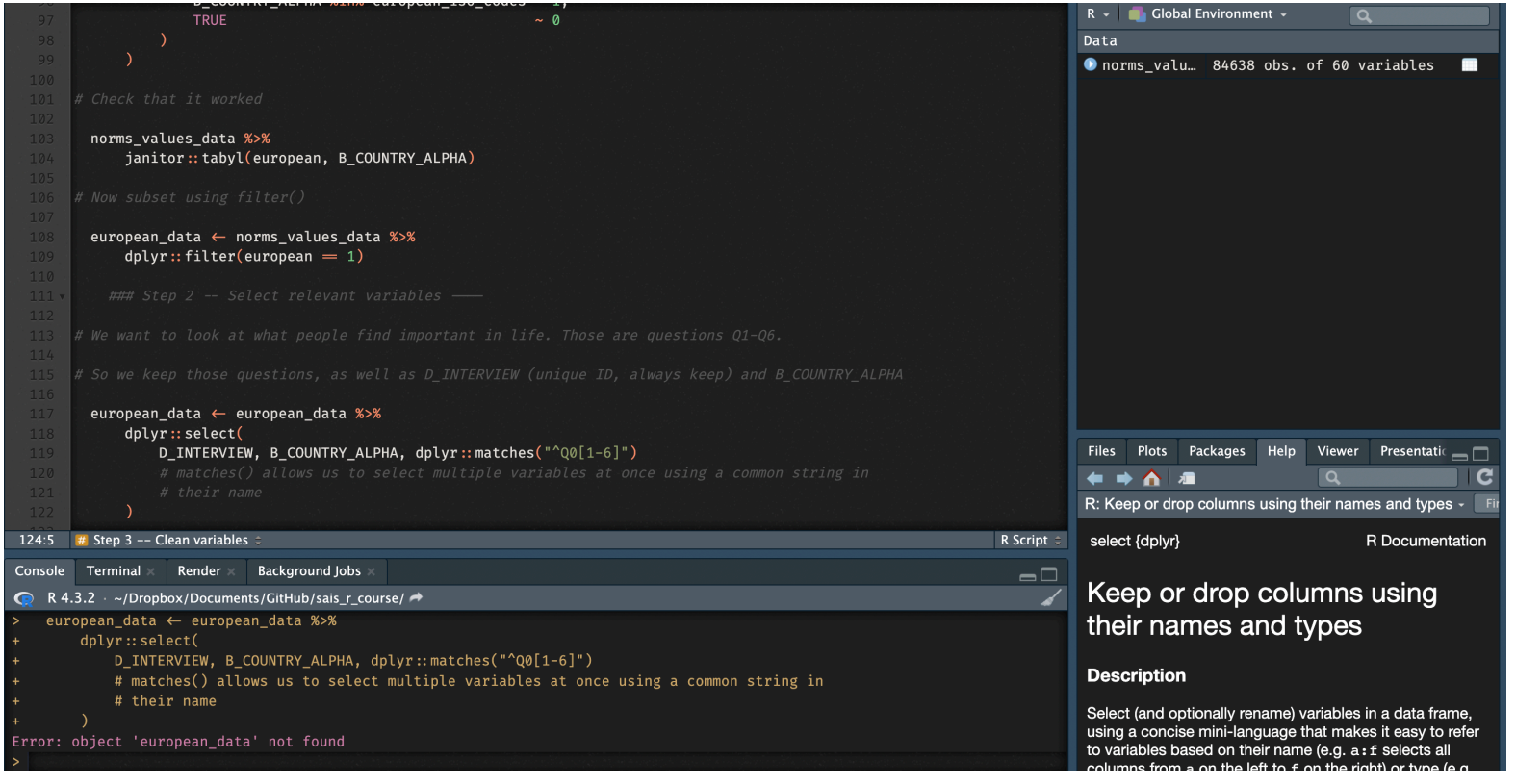
- That you didn't miss a parenthesis ( )) or bracket (}) in your code! This is the most common reason.
- If you missed it, the console will show a + at the start of the console line instead of the expected >.

Solutions:

- Type gibberish and/or the missing parenthesis/bracket until the > reappears. More likely, you'll have to rerun the code chunk to make sure it works!

Basic Issues in RStudio

Object Not Found



Basic Issues in RStudio

Object Not Found

Check:

- Whether the object exists in your environment. More likely than not, you either misspelt the name of the object, or you skipped the code that creates it (remember that a script is like a recipe and steps can't be skipped!).

Solutions:

- Backtrack in your code and run the chunk that creates the object.
- If a typo is at fault, correct the typo.

Practical Exercise — Using the World Values Survey Dataset

World Values Survey

Background

"The survey, which started in 1981, seeks to use the most rigorous, high-quality research designs in each country. The WVS consists of nationally representative surveys conducted in almost 100 countries which contain almost 90 percent of the world's population, using a common questionnaire. [...] WVS seeks to help scientists and policy makers understand changes in the beliefs, values and motivations of people throughout the world."

Survey Contents

- Social values, attitudes & stereotypes
- Societal well-being
- Social capital, trust and organizational membership
- Economic values
- Corruption
- Migration
- Post-materialist index
- Science & technology
- Religious values
- Security
- Ethical values & norms
- Political interest and political participation
- Political culture and political regimes
- Demography

Today's practical component

1— Download the World Values Survey (Wave 7) at this link: https://www.worldvaluessurvey.org/WVSDocumentationWV7.jsp

- You can theoretically download any of the provided file types, but the easiest to use will be the .csv file (WVS Cross-National Wave 7 csv v6 0.zip).
- You'll have to agree to WVS's conditions of use and provide contact information to download this data.

2— Set up your data analysis folder. Make sure that it has the following components:

- data
  - raw
  - intermediate
- code
- outputs
- documentation

3— Place the data and documentation files that you find appropriate for your analysis in the corresponding folders.

4— Open RStudio and click on "New Project" in the "File" dropdown menu. Navigate to your data folder and select it.

Today's practical component

5— Set up your R Script. It should have the following components:

- Package setup. For now, have the script load the `dplyr`, `data.table`, and `janitor` packages.
- Data import. Import your data into the script using `read.csv()` or `fread()`. Remember to give your data frame an intuitive name!

6— Test your R Script. Type and run the following line in your code (**replacing `[[DATASETNAME]]` with the name you gave your dataset**): `[[DATASETNAME]] %>% head()`. If this works without issue, you're free to go!

Links

Thomas Mock, “A Gentle Introduction to Tidy Statistics in R” (blog post and video)

Hadley Wickham & Garrett Grolemund, R for Data Science, Second Edition

RStudio, RStudio Cheatsheets