Data Management and Visualization, Part I

Andie Creel

1 Goal

The goal of this lecture and the next is:

1) Learn how to set up an R Project
2) Learn best practices for file structures
3) Importing data
4) Exporting results
5) Basic data visualization with ggplot

2 Creating an R Project

2.1 What's an R Project?

An R project contains multiple scripts, your data, results you export, and any other files you have associated with that project.

R projects are an extremely useful organizational tool, and you should use them for every research project.

They act as an address for your computer to recognize that everything in that project stays together.

2.2 New R Project

1) Open R Studio
2) File/New Project... (alternatively, little clear box with R button)
3) Navigate to the file you want the r project to exist in. **Make sure you can find this again so that we can drag folders into it.**
4) Give it a short, informative name
5) Done! Notice you're top right hand corner.. you're R Studio session has loaded your R Project

Do this together.

3 File Structure

- File structure can make or break a research project
  - Acts as an outline that gives you forward momentum
  - Allows your collaborators to clearly follow what you did
  - Allows you to return to a project after and quickly remember what step you were on and what remains to be done
- There are best practices for file structure that are worth following now

3.1 Best practices

Matt Wibbenmeyer (fellow at RFF) wrote a Notes on Project Management and Collaboration for RFF RAs and interns that I still use. It is worth reading and you can find it here. I share it with all my RAs and collaborators.

There will be your `project/` folder. This is the file your R Project files lives. Everything else will be in this folder or sub folders. I typically have the following sub folders.

- `data/`
  - `raw_data/`: The original data. Do not edit this! You never know when what you originally thought was a good idea was actually bad. Always keep a copy of your original data here.
  - `clean_data/`: Where you store the clean data sets that you'll use for your models
- `scripts/`: all of the r scripts you write
  - `1_data_cleaning.R`
  - `2_descriptives.R`
  - `3_analysis.R`
  - `4_figures.R`
- `results/`: where you store final graphs and tables
- `manuscript/`: where you store your writing that (figures crossed) becomes your paper
- `presentations/`: Any presentations you prepare for the research project

In our new r project, let's create a clean and raw ***`data`****,* ***`script`****, and* ***`results`*** *sub folder. You can do this with the file-green-plus button in the files tap (bottom right corner).*

4 Importing Data

You can import a lot of different data sets in a lot of different ways. The most common data sets I import are CSVs.

4.1 Note on Excel Files

I try to avoid going back and forth between excel and R. Excel is useful for data entry (especially when you're collecting your data in the field). But once you're done with data collection, I would avoid going between R and Excel.

My advice would be to export your excel sheet to a csv. CSVs are more memory efficient and easier to read into R. However, if you edit the Excel file remember you'll need to re-export your csv.

Code for if you do need to read in an excel file instead of a csv.

```
# ------------------------------------------------------------------------
# Install (only once) and load the readxl package
# ------------------------------------------------------------------------
# install.packages("readxl")
library(readxl)

# ------------------------------------------------------------------------
# read in an excel sheet
# ------------------------------------------------------------------------
myData <- read_excel("raw_data/an_excel_file.xls",
                     sheet = "the_name_of_the_sheet_I_am_importing")
```

4.2 Code for reading in a CSV so you have it

There are a million ways to read in CSVs. In this pdf is the code so you have it.

```
# ------------------------------------------------------------------------
# Base r
#   Pro: no need to load package
#   Con: less efficient, slower, and worse at getting variable types right
#   Use case: when you have a small and simple data set
# ------------------------------------------------------------------------
myData <- read.csv("/raw_data/an_imaginary_csv.csv")

# ------------------------------------------------------------------------
# Tidyverse package: readr
#   Pro: faster, intuitive at predicting variables types
#   Con: Requires a package
#   Use case: almost all the time
# ------------------------------------------------------------------------
# install.packages("readr")
library(readr)

myData <- read_csv("/raw_data/an_imaginary_csv.csv")

# ------------------------------------------------------------------------
# Another package: vroom
#   Pro: excellent for big data
#   Con: a bit clunkier than readr
#   Use case:  big data
# ------------------------------------------------------------------------
# install.packages("vroom")
library(vroom)

myData <- vroom("/raw_data/an_imaginary_csv.csv")
```

5 Today's Example

5.1 Data Download and Clean

First step: download csv and drag it into `raw_data` folder.

- Here is the link: https://github.com/a5creel/intro_to_programming/blob/main/lecture_material/4_data_manage_vis/raw_data/mpg.csv
- Click the download button

First, pull data into the ***`raw_data`*** *file we created while going through file structure. Next, create a script called* ***`1_data_clean.R`*** *in the* ***`scripts`*** *folder*

```
# Andie Creel / Goal: data cleaning / Started: January 11 2023

# ------------------------------------------------------------------------
# load libraries
```

```r
# ------------------------------------------------------------------------
library(readr) # reading in csv
library(dplyr) # data cleaning
# library(ggplot2) # if we're fighting to get data loaded

# ------------------------------------------------------------------------
# read in: reading in data is slow, so i always call it _og in case
#     I regret something later and want to return to this step
# ------------------------------------------------------------------------
myData_og <- read_csv("data/raw_data/mpg.csv")

# Look at the dataset so we know what variables we have etc
# View(myData_og)

# alternatives in case this isn't working
# myData_og <- read_csv("https://raw.githubusercontent.com/a5creel/intro_to_programming/main/lecture_ma

# data(mpg)
# myData_og <- mpg
# rm(mpg)

# ------------------------------------------------------------------------
# clean data: what we learned this morning
#   - the variables you do or don't need should be informed by your research Q
# ------------------------------------------------------------------------

# our research Q is:
#   - only interested in Ford, dodge and toyota
#   - not about the type of fuel "fl"
myData <- myData_og %>%
  filter(manufacturer == "ford" | manufacturer == "dodge" | manufacturer == "toyota") %>%
  select(-fl) %>%
  mutate(cyl = as.factor(cyl)) # data manipulation: factor datatypes are discrete categories



# write clean data so it's saved as a seperate file we can load
write_csv(myData, "data/clean_data/my_clean_data.csv")
```

- writing clean data can save memory and be fast for when you're doing analysis
- Normally, you'd do some more exploratory analysis in another file
- You'd also do more data cleaning than this (refer to data cleaning lectures!)
- But this is a basic example of how to get the project workflow set up
- We will pick up from here for data vis!