R Workflow: Projects, File Paths, and Reading Data

BMSC 620

Emile Latour

2026-02-03

Roadmap for Today's R Content

Building on last class: File paths and reproducibility

Today's goals:

1. **R Projects** - Your solution to file path headaches
2. **The `here` package** - Robust file referencing
3. **Reading data** - Getting your data into R
   - CSV files with `readr`
   - Excel files with `readxl`

By the end, you'll be able to load the body temperature dataset we'll use for hypothesis testing!

The Problem: File Paths Break

Recall: Why do file paths break?

Last class we talked about how this breaks:

```r
# This works on MY computer
data <- read_csv("C:/Users/Emile/Documents/BMSC620/data/BodyTemps.csv")
```

Problems:

- Hard-coded path specific to one computer
- Won’t work if you move the folder
- Won’t work on collaborator’s computer
- Won’t work if you rename folders
- Makes your code **not reproducible**

Reproducibility

- Research **data** and **code** (and **documentation**) can reach the same results regardless of who is running the code
  - This can also refer to future or past you!
- We want to set up our work so the entire folder can be moved around and work in its new location

What we want: Portable, reproducible code

Ideal scenario:

- Your entire project folder can be moved anywhere
- Code still works without editing paths
- Collaborators can run your code immediately
- Future you can run it on a new computer

The solution: R Projects + `here` package

R Projects: Your Working Directory Solution

What is an R Project?

An **R Project** is a way to designate a working directory for your analysis.

When you create an R Project:

- RStudio creates a `.Rproj` file in your folder
- That folder becomes the “root” of your working directory
- RStudio knows where you are and where to find files
- Each project has its own independent environment

Best practice

Create a **separate R Project for every analysis** (and every class!)

Why use R Projects?

Organization:

- Keeps all files for one project together
- Easy to see what belongs to what analysis

Reproducibility:

- Paths are relative to the project folder
- Project can be moved anywhere and still work
- Easy to share with collaborators

Workflow:

- Can have multiple RStudio sessions open (different projects)
- Each session is independent
- Easy to switch between projects

The nice thing about R projects

- 5 minute video explaining some of the nice features of R projects

https://rfortherestofus.com/2022/10/rstudio-projects

Recommended folder structure

When you create a project, organize your files:

My typical folder structure

```
MyProject/
├── MyProject.Rproj       # R Project file
├── data/                 # data files (never edited by hand)
├── code/                 # R, qmd, html
├── docs/                 # notes, instructions, PDFs, references
├── figures/              # saved figures
├── deliverables/         # sent to collaborators, by date
└── admin/                # budgets, admin
```

For this class, I recommend:

```
BMSC620/
├── BMSC620.Rproj         # R Project file
├── data/                 # Datasets I provide
├── homework/             # Your homework files, one folder for each HW
├── notes/                # Your class notes (hmtl, pdf, etc.)
├── practice/             # Practice exercises
└── misc/                 # And other folders if you want
```

How to create an R Project

Option 1: New project in existing folder

(recommended if you already organized folders)

1. `File → New Project...`
2. Choose `Existing Directory`
3. Navigate to your class folder
4. **Check "Open in new session"**
5. Click `Create Project`

Option 2: New project in new folder

1. `File → New Project...`
2. Choose `New Directory`
3. Choose `New Project`
4. Name your project and choose where to save it
5. **Check "Open in new session"**
6. Click `Create Project`

Always check "Open in new session"

This keeps your current work separate from the new project. Good habit for managing multiple projects!

Live demonstration

Let me show you how to:

1. Create an R Project for our class
2. Set up the folder structure
3. Open the project
4. **Note: Watch for the “Open in new session” checkbox**

Note

We’re creating a **“regular” R Project**, not a “Quarto Project”

- Regular projects are simpler and work perfectly for our needs
- Once you’re comfortable, you can explore Quarto Projects later

Your turn: Create your class project

Task: Create an R Project for BMSC 620

Steps:

1. Decide where on your computer you want your class folder
2. Create folders: `data`, `homework`, `notes`, `practice`
3. In RStudio: `File → New Project... → Existing Directory`
4. Navigate to your class folder
5. **✓ Check "Open in new session"**
6. Click `Create Project`

You should now see a `.Rproj` file in your folder!

Opening a project

To work on your project in the future:

Option 1: Double-click the `.Rproj` file

- Opens RStudio with that project loaded
- Working directory is automatically set

Option 2: In RStudio, click the project dropdown (top right)

- Shows recent projects
- Easy to switch between projects

Workflow tip

Always open RStudio by opening your project file, not just opening RStudio directly!

The `here` Package: Robust File Paths

The problem here solves

Even with an R Project, you still need to reference files:

```r
# These might work differently depending on file type
data <- read_csv("data/BodyTemps.csv")
```

The issue:

- `.qmd` files and `.R` files handle working directories differently
- Can lead to confusion about where files are
- `here` package makes this consistent and reliable

What does `here` do?

The `here` package **always starts at your project root** (where the `.Rproj` file is)

```r
library(here)
here()  # Shows your project root directory
```

```
[1] "/Users/latour/Library/CloudStorage/Dropbox/teaching/BMSC_620_W26"
```

Benefits:

- Works the same in `.qmd` and `.R` files
- Paths are relative to project root
- Very clear where files are located
- Essential for reproducibility!

Using `here()` to reference files

Basic syntax:

```r
here("folder_name", "filename")
```

Examples:

```r
# Data file in the data folder
here("data", "BodyTemps.csv")

# Output file
here("output", "my_plot.png")

# Nested folders
here("data", "raw", "survey_data.xlsx")
```

The `here()` function builds the full file path for you!

here + readr: Reading CSV files

To load a CSV file:

```r
library(tidyverse)  # includes readr package
library(here)

# Read CSV file from data folder
body_temps <- read_csv(here("data", "BodyTemperatures.csv"))
```

What's happening:

- `here("data", "BodyTemperatures.csv")` creates the full path
- `read_csv()` reads the CSV file into R
- Data is stored in `body_temps` object

here + readxl: Reading Excel files

To load an Excel file:

```r
library(readxl)
library(here)

# Read Excel file from data folder
body_temps <- read_excel(here("data", "BodyTemperatures.xlsx"))
```

Note: `readxl` is not part of the tidyverse, so install separately:

```r
install.packages("readxl")
```

Common data reading functions

Function	File type	Package	Notes
`read_csv()`	`.csv`	`readr` (tidyverse)	Recommended for CSV files
`read_excel()`	`.xlsx`, `.xls`	`readxl`	For Excel files
`read.csv()`	`.csv`	base R	Older base R function
`read_delim()`	tab, other delimiters	`readr`	For other delimited files
`read_sas()`	`.sas7bdat`	`haven`	For SAS files

For this class: You'll mostly use `read_csv()` and occasionally `read_excel()`

Putting it all together

Complete workflow for loading data:

```r
# 1. Load packages
library(tidyverse)  # includes read_csv()
library(here)

# 2. Load data using here
body_temps <- read_csv(here("data", "BodyTemperatures.csv"))

# 3. Check the data
glimpse(body_temps)
```

This code will work:

- On any computer
- After moving your project folder
- For your collaborators
- Years from now

Resources and Best Practices

Key resources

R Projects:

- RStudio Projects and Working Directories: A Beginner's Guide
- Using RStudio Projects
- Video: The Basics of Projects in RStudio

`here` package:

- `here` package
- Ode to the here package (Jenny Bryan)

Best practices summary

Always do these three things

1. **Use R Projects** for every analysis
2. **Use `here()`** for all file paths
3. **Organize your files** in a clear folder structure

This makes your code:

- Reproducible
- Portable
- Shareable
- Future-proof

Your future self (and collaborators) will thank you!

Practice: Load the body temperature data

Your task:

1. Make sure your R Project is open
2. Download `BodyTemperatures.csv` from Canvas
3. Save it in your `data` folder
4. Create a new `.qmd` file
5. Load the data using `here()`

Code to try:

```r
library(tidyverse)
library(here)

body_temps <- read_csv(here("data", "BodyTemperatures.csv"))
glimpse(body_temps)
```

Questions?

We'll use this workflow throughout the course!

Coming up: We'll use this body temperature dataset for hypothesis testing examples.

Artwork by @allison_horst