Data Science and Society with R

DSDA/STAT 1010 - Quarto Book

Jun Yan

2025-08-26

Table of contents

Preliminaries

Tip

Welcome! This Quarto book hosts all lecture notes, in-class activities, and weekly reading for DSDA/STAT 1010. It is designed for first-year students with no prerequisites.

Adapting to Rapid Skill Acquisition

In this course, students are expected to rapidly acquire new skills, a critical aspect of data science. To emphasize this, consider this insightful quote from VanderPlas (2016):

> When a technologically-minded person is asked to help a friend, family member, or colleague with a computer problem, most of the time it's less a matter of knowing the answer as much as knowing how to quickly find an unknown answer. In data science it's the same: searchable web resources such as online documentation, mailing-list threads, and StackOverflow answers contain a wealth of information, even (especially?) if it is a topic you've found yourself searching before. Being an effective practitioner of data science is less about memorizing the tool or command you should use for every possible situation, and more about learning to effectively find the information you don't know, whether through a web search engine or another means.

This quote captures the essence of what we aim to develop in our students: the ability to swiftly navigate and utilize the vast resources available to solve complex problems in data science. Examples tasks are: install needed software (or even hardware); search and find solutions to encountered problems.

Course tools

- **R & RStudio** for analysis
- **Quarto** for reproducible documents and dashboards
- **Git & GitHub** for version control and project management
- **Command line** for automation and efficiency

Policies & syllabus

See the course syllabus on HuskyCT.

Key reminders: academic integrity, no AI-generated text in graded submissions, and professional email etiquette.

1 Know Your Computer

Note

Learning goals 1) Recognize how files, folders, and paths work across macOS, Windows, and Linux.
2) Use a terminal to navigate and run simple commands.
3) Install R, RStudio, and Quarto; verify your setup.

1.1 Operating Systems

Most students are familiar with **Windows**, but data science workflows often run on **macOS** or **Linux**. To keep everyone on the same page for command-line work, we will treat macOS and Linux as a single "Unix-like" family and help Windows users bridge to the same environment via WSL (Windows Subsystem for Linux).

Note

What is an Operating System?
An operating system (OS) manages your computer's resources (CPU, memory, storage, files, and processes) and provides interfaces (GUI and terminal) for people and programs to interact with hardware.

1.1.1 The big three

- **Windows** — Ubiquitous on personal laptops; historically less aligned with Unix tooling, but excellent with WSL.
- **macOS** — Unix-based; ships with a terminal and many developer tools out of the box.
- **Linux** — Open-source family used on servers, clusters, and cloud VMs; many distributions (Ubuntu, Fedora, Debian) share common command-line tools.

Tip

Why this matters in data science
Reproducibility and collaboration require knowing your OS, versions, and paths. Most research servers and HPC clusters run Linux. Learning a command-line interface (CLI) gives you a common language across systems.

1.1.2 Quick checks: what am I running?

Open a terminal and run **one** of the following:

```
# macOS / Linux / WSL
uname -a
```

```
# Windows (PowerShell)
ver
```

If you see Linux details on a Windows laptop, you are inside WSL.

1.1.3 File systems and paths

- **Unix-like path style**: `/home/alex/project/data.csv`
- **Windows path style**: `C:\\Users\\Alex\\project\\data.csv`

On macOS/Linux, your home is typically `/Users/<name>` (macOS) or `/home/<name>` (Linux). On Windows, it is usually `C:\\Users\\<name>`.

Important

Naming habits that save you pain
Avoid spaces in file and folder names. Prefer `kebab-case` or `snake-case`. Keep a project's scripts, data, and reports together.

1.1.4 Windows Subsystem for Linux (WSL)

WSL lets you run a real Linux environment (e.g., Ubuntu) inside Windows, so your terminal commands match those of macOS/Linux users. This is the recommended setup for Windows in this course.

1.1.4.1 Install WSL (Windows 11 or Windows 10 2004)

1. Open **PowerShell as Administrator**.
2. Run:

```
wsl --install
```

3. When prompted, choose **Ubuntu** and set a Linux username and password.
4. (Optional) Ensure WSL2 is the default:

```
wsl --set-default-version 2
```

5. Verify:

```
wsl --status
wsl -l -v
```

You should see `Ubuntu` listed and version `2`.

1.1.4.2 Start using WSL

- Launch the **Ubuntu** app (or run `wsl` in PowerShell).
- You are now at a Linux shell (bash). Try:

```
pwd          # current directory in the Linux filesystem
ls           # list files
whoami       # your Linux username
```

1.1.4.3 Sharing files between Windows and WSL

- Windows drives are mounted inside Linux at `/mnt/c`, `/mnt/d`, …
  Example: `C:\\Users\\Alex\\project` appears at `/mnt/c/Users/Alex/project` inside WSL.
- Your Linux home is separate (e.g., `/home/alex`).

Tip

Best practice
Keep course projects *inside your Linux home* (e.g., `/home/<name>/dsda1010`) to avoid path and permissions surprises.

1.1.4.4 Line endings and Git on Windows/WSL

Configure Git once to avoid CRLF/LF confusion:

```
git config --global core.autocrlf input   # recommended in WSL/macOS/Linux
git config --global init.defaultBranch main
```

Add a `.gitattributes` to normalize endings:

```
* text=auto
*.qmd text eol=lf
*.R   text eol=lf
*.md  text eol=lf
*.yml text eol=lf
```

1.1.5 Terminal quickstart by OS

- **macOS**: Open **Terminal** (or **iTerm2**). You are in a Unix shell. Commands from the book work as-is.
- **Linux**: Open your terminal (GNOME Terminal, Konsole, etc.). You are already in a Unix shell.
- **Windows (WSL)**: Open the **Ubuntu** app (WSL). You are now in a Linux shell that matches macOS/Linux.

Try these commands everywhere:

```
pwd            # show current path
ls -lah        # list with details and human-readable sizes
mkdir lab      # make a folder
cd lab         # enter it
curl --version
R --version    # verify R once installed later
```

1.1.6 Common pitfalls and fixes

- **"Command not found"** — The program is not installed, or your PATH does not include it.
- **Permission denied** — You may be in a protected folder; work in your home directory.
- **Strange characters in filenames** — Avoid spaces and punctuation; stick to letters, numbers, dashes.
- **Mixing Windows and WSL paths** — Prefer working inside your WSL home. If you must access Windows files, use `/mnt/c/...`.

1.1.7 Hands-on check (to do now)

1. Open your terminal (macOS/Linux/WSL).

2. Confirm your OS with `uname -a` (or `ver` in PowerShell).

3. Create a course folder and a notes file:

```
mkdir -p ~/dsda1010/week1
cd ~/dsda1010/week1
echo "Week 1 notes" > notes.txt
ls -l
```

1.2 Files, folders, and paths

- **Home folder:** Your personal workspace.

- **Absolute vs relative paths:**

```
# absolute (macOS/Linux)
/Users/alex/projects/dsda1010

# absolute (Windows)
C:\Users\alex\projects\dsda1010

# relative (from a project root)
../data/nyc311.csv
```

- **Good habits**
  - Avoid spaces in file/folder names (`use-kebab-case`).
  - Keep project files together (R scripts, data, and reports).

1.3 Extensions and line endings

- Common types: `.qmd`, `.R`, `.csv`, `.tsv`, `.parquet`, `.md`.
- Text vs binary: keep data in text formats when possible.
- Line endings: Git normalizes these; we set `.gitattributes` to help.

1.4 Terminals and shells

Open a terminal and try:

```
pwd          # print working directory
ls           # list files
cd ..        # move up one directory
mkdir lab    # make a folder
cd lab         # go into the newly created folder
```

Tip

Use Tab to autocomplete file and folder names.

1.5 Install R, RStudio, and Quarto

1. Install **R** from CRAN.

2. Install **RStudio Desktop**.

3. Install **Quarto**.

4. Verify:

```
R --version
quarto --version
```

In RStudio, create a new Quarto document and render it.

1.6 Your first Quarto project

From a terminal:

```
mkdir dsda1010
cd dsda1010
# Edit hw-template.qmd
quarto render hw-template.qmd
```

Open the folder in RStudio and inspect the generated files.

1.7 Troubleshooting checklist

- PATH issues: can the terminal find `R` and `quarto`?
- Permissions: can you write to your project folder?
- Antivirus or VPN blocking downloads?

! Important

If your machine is shared or locked down, use the **HPC workstations** or **RStudio Server** if provided by the department. Ask the instructor for access details.

References

VanderPlas, J. (2016). *Python data science handbook: Essential tools for working with data.* O'Reilly Media, Inc.