Applied Quantitative Methods for the Social Sciences II
MA in Social Sciences, UC3M, Spring 2026

Appendix to Assignment 1: Code editors and alternatives to RStudio

In this document I list the introduction to two alternatives to RStudio, in case you want to try them out (but there is no need to):

1. Positron is a bit better version of RStudio, but of the same nature: an integrated software (an IDE, or 'integrated development environment') where you edit R code, run it, see graphs, etc. Also valid for Python.
2. Sublime Text is a *code editor*, so a more complex tool to handle plain text files in general. You can customize it for any language you use (R, Latex, Markdown, Python, etc). It is the one I use.

Contents

1 Positron

As a potential alternative to RStudio, I recommend using **Positron** (https://positron.posit.co) as your IDE. Positron is a next-generation data science IDE built by Posit (the same company behind RStudio). It is based on VS Code and designed for R and Python.

1.1 Installing Positron

1. Download Positron from https://positron.posit.co
2. Install it like any other application
3. On first launch, Positron will detect your R installation automatically

1.2 Differences for Assignment 1

The following table maps the RStudio instructions in this assignment to their Positron equivalents:

RStudio	**Positron**
File → New Project → Version Control → Git	File → New Folder, then open the folder. Use the built-in terminal (`Ctrl+‘`) to run `git clone`.
Git pane (top-right)	Source Control panel in the left sidebar (branch icon), or `Ctrl+Shift+G`
Check box next to file to stage it	Click the `+` icon next to the file in the Source Control panel
Click “Commit” button	Click the checkmark icon in the Source Control panel, type your message in the text box
Click “Push” button	Click the `...` menu in Source Control → Push, or use the terminal: `git push`
Click “History” (clock icon)	Use the terminal: `git log --oneline`, or install the “Git Graph” extension for a visual history
Files pane (bottom-right)	Explorer panel in the left sidebar (top icon), or `Ctrl+Shift+E`
File → New File → R Script	File → New File, then select “R File”
Tools → Global Options → Git/SVN	Git is detected automatically. If not, open Settings (`Ctrl+,`) and search for “git.path”

1.3 Key Advantages of Positron

- **Integrated terminal**: Positron has a built-in terminal (`Ctrl+‘`) where you can run Git commands directly—no need to switch to a separate application
- **Better Git integration**: The Source Control panel shows diffs, staged changes, and commit history in one place
- **Extensions**: You can install extensions (e.g., “Git Graph” for visual commit history, “R” for enhanced R support) from the Extensions panel
- **Multiple languages**: Positron works equally well with R and Python, which is useful if you work across both

Note: All the command line instructions in this assignment work identically regardless of whether you use RStudio, Positron, or a standalone terminal. The differences only apply when using the graphical interface.

2 Sublime Text 4

Sublime Text is a fast, lightweight code editor. Unlike RStudio or Positron, it is not an IDE—it does not come with an R console, file browser, or plot viewer out of the box. Instead, you write R code in Sublime Text and send it to a separate R console running alongside it. This minimal setup requires only two packages.

2.1 Step 1: Install Sublime Text 4

Download from https://www.sublimetext.com and install it.

2.2 Step 2: Install Package Control

Package Control is Sublime Text's package manager. To install it:

1. Open the Command Palette: `Ctrl+Shift+P` (Windows/Linux) or `Cmd+Shift+P` (Mac)
2. Type "Install Package Control" and press Enter
3. Wait for the confirmation message

2.3 Step 3: Install R-IDE and SendCode

Using Package Control, install two packages:

1. Open the Command Palette (`Ctrl+Shift+P` / `Cmd+Shift+P`)
2. Type "Package Control: Install Package" and press Enter
3. Search for **R-IDE** and install it (provides R syntax highlighting, code completions, and function signatures)
4. Repeat the process and install **SendCode** (sends code from the editor to an external R console)

2.4 Step 4: Configure SendCode

SendCode needs to know where to send your code. Open its settings:

1. Go to Preferences → Package Settings → SendCode → Settings
2. In the right-hand pane (user settings), paste the following:

```
{
    "prog": "Terminal"
}
```

On **Mac**, set `"prog"` to `"Terminal"` to send code to the built-in Terminal (where you will run R), or to `"iTerm"` if you use iTerm2. On **Windows**, set it to `"Cmder"`, `"ConEmu"`, or another terminal emulator. On **Linux**, set it to `"tmux"` or `"linux-terminal"`.

2.5 Step 5: The Workflow

1. Open a terminal window and start R by typing `R` and pressing Enter
2. In Sublime Text, open your `.R` file
3. Place your cursor on a line of code and press `Ctrl+Enter` (Windows/Linux) or `Cmd+Enter` (Mac)—SendCode sends that line to the R console and moves to the next line
4. To send a selection, highlight multiple lines and press the same shortcut

That is the entire setup. You edit in Sublime Text, execute in the R console, and switch between them as needed. For Git operations, use the terminal.