

# Lecture 2: R Markdown, Version Control with Git(Hub), and Other Productivity Tools - Part 1

James Sears\*

AFRE 891/991 FS 25

Michigan State University

\*Parts of these slides are adapted from [“Advanced Data Analytics”](#) by Nick Hagerty and [“Data Science for Economists”](#) by Grant McDermott.

# Table of Contents

1. **Prologue**
2. **R Markdown**
3. **Version Control**
4. **GitHub Desktop - Part 1**

# Prologue

# Prologue

Before we dive in, let's double check that we all have

- ☑ Installed [R](#).
- ☑ Installed [RStudio](#).
- ☑ Signed up for an account on [Github](#)
- ☑ Installed [Git](#) and [Github Desktop](#)
- ☑ Log into your Github account on Github Desktop

# R Markdown

# R Markdown

Before we dive into version control, let's chat about **R Markdown**.

R Markdown is a document type that allows for integration of R code and output into a Markdown document.

## Resources:

- Website: [rmarkdown.rstudio.com](https://rmarkdown.rstudio.com)
- [R Markdown Cheatsheet](#)
- Book: [R Markdown: The Definitive Guide](#) (Yihui Xie, JJ Allaire, and Garrett Grolemund)

# R Markdown

Before we dive into version control, let's chat about **R Markdown**.

R Markdown is a document type that allows for integration of R code and output into a Markdown document.

## Other points:

- We'll be completing assignments using R Markdown.
- FWIW, my lecture slides and notes are all written in R Markdown too.  
(E.g. This slide deck is built using the xaringan package with the metropolis theme.)

# R Markdown: Getting Started

☒ Installed [R](#).

☒ Installed [RStudio](#).

☐ Add the `rmarkdown` package

```
install.packages("rmarkdown")
```

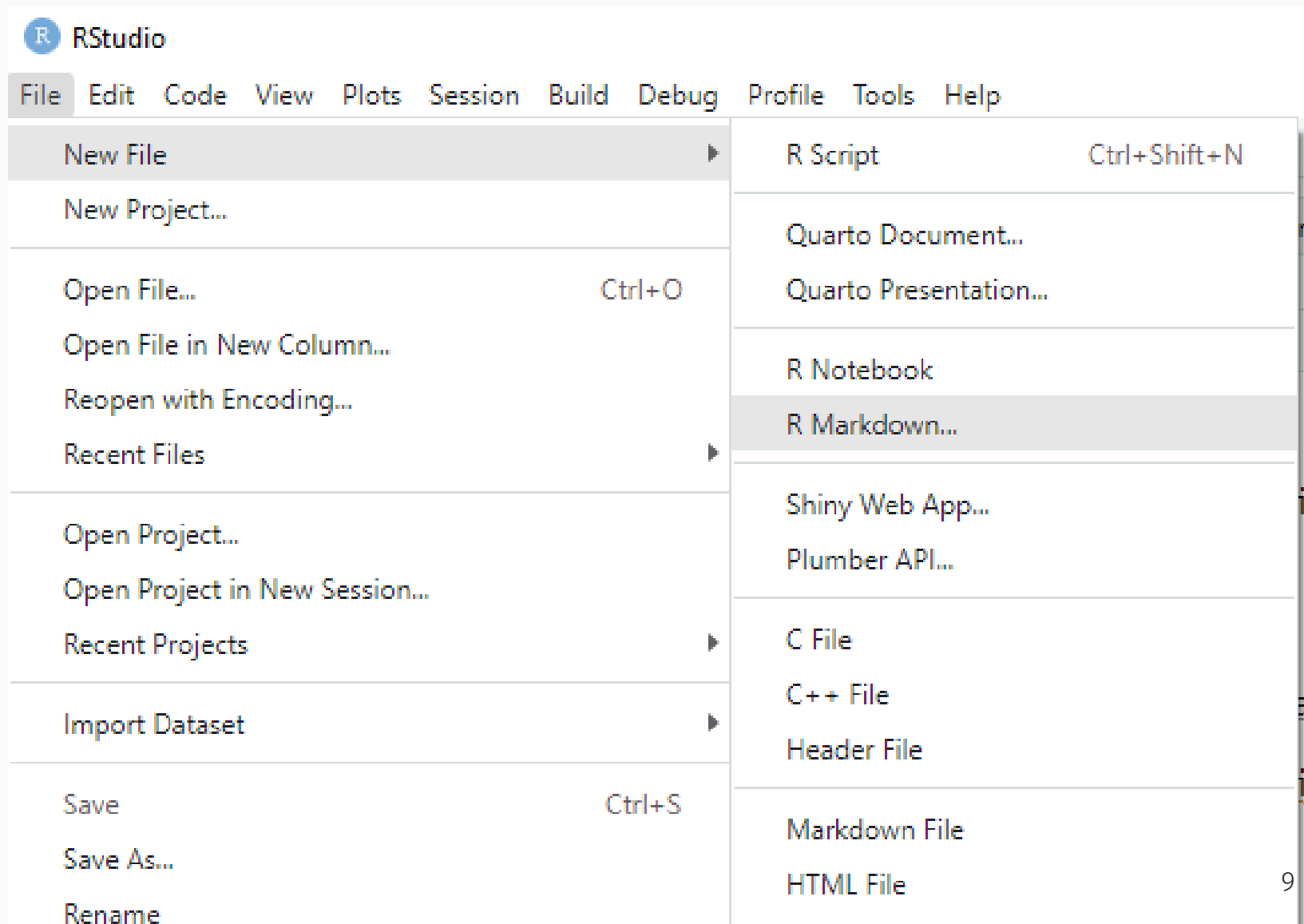
☐ Install LaTeX

- If just for this, can use [TinyTex](#)

```
# Install only if you don't have LaTeX already  
install.packages("tinytex")  
tinytex::install_tinytex()
```



# R Markdown: Creating a New .Rmd File



# R Markdown: Creating a New .Rmd File

New R Markdown

Document

Presentation

Shiny

From Template

**Title:** Test R Markdown

**Author:** James

**Date:** 2024-01-10

☐ Use current date when rendering document

**Default Output Format:**

☒ HTML  
Recommended format for authoring (you can switch to PDF or Word output anytime).

☐ PDF  
PDF output requires TeX (MiKTeX on Windows, MacTeX 2013+ on OS X, TeX Live 2013+ on Linux).

☐ Word  
Previewing Word documents requires an installation of MS Word (or Libre/Open Office on Linux).

# R Markdown: Creating a New .Rmd File



# R Markdown Components

R Markdown combines

1. **Markdown:** lightweight markup language
2. **LaTeX:** typesetting for math
3. **R:** include code and generate output

Let's do some practice: **open a new .Rmd file** and try adding content as we go

# Markdown

**Markdown** allows for formatting text in a lightweight way

I highly recommend the handy [Markdown Guide](#) for more details

# Markdown: Heading

**Headings** emphasize text and add chunks to your script

Largest heading with one leading # (slide title above)

Second Largest (##)

Third Largest (###)

Getting Smaller... (####)

Normal Text for comparison

# Markdown: Text Format

**Bold text** with `**your text**`

*Italicize* with `*single asterisks*`

Add `code text` with grave accents (the back tick symbol)

- ```
- The other output of the tilde key `~` on keyboard

End a line with two spaces to start a new paragraph

- or leave a line space between sentences

Can also start a new line with backslash (`\`)

# Markdown: Text Format

Add superscripts<sup>2</sup> with ^carets^

Add ~~strikethroughs~~ with ~~double tildes~~

Add a line break (horizontal rule)

---

with \*\*\*



# Markdown: Text Format

Draw **tables** using | and -

```
| Col A | Col B | Col C |
|-----|-----|-----|
| This  | is     | a      |
| Table |        | wow    |
```

# Markdown: Text Format

Draw **tables** using | and -

Col A Col B Col C		
This	is	a
Table		wow

# Markdown: Text Format

You can adjust the **alignment** of table text by adding `:`'s in the second row:

- `:-----` for left-aligned
- `:---:`  for center-aligned
- `-----:` for right-aligned

```
| Column A | Column B | Column C |
|:---:    |:---:    |-----: |
| Col A   | is      | left-aligned |
| Col B   | is      | center-aligned |
| Col C   | is      | right-aligned |
```

# Markdown: Text Format

You can adjust the **alignment** of table text by adding `:`'s in the second row:

- `:-----` for left-aligned
- `:---:---` for center-aligned
- `-----:` for right-aligned

Column A	Column B	Column C
Col A	is	left-aligned
Col B	is	center-aligned
Col C	is	right-aligned

# Markdown: Lists

Add an **ordered list** with **1.**

1. First Item
2. Second Item
3. No need to change the number - keep using 1. It will automatically update.

Add an **unordered list** with **\* or -**

- A thing
- Another related thing
  - Indent to nest
    1. Can mix ordered and unordered

# Markdown: Inputs

Add a **link** with `[]()`

- `[text label](URL)`
- Add direct link with `<link>` <https://www.markdownguide.org>

Add an image with `![]()`

- `![alt text](URL)`

**practice** by adding `images/smile.png`:



# Markdown: LaTeX

Another advantage of Markdown is that it integrates  $\text{\LaTeX}$  functionality for typesetting math.

Add an **inline equation** with  $\$TeX\$$

$$Var(X) = \sum_{i=1}^n \frac{(x_i - \bar{x})^2}{n} \quad Y_{it} = \beta_0 + \beta_1 X_{it} + \epsilon_{it}$$

Add multiple rows of LaTeX with

$\$$

LaTeX lines here

$\$$

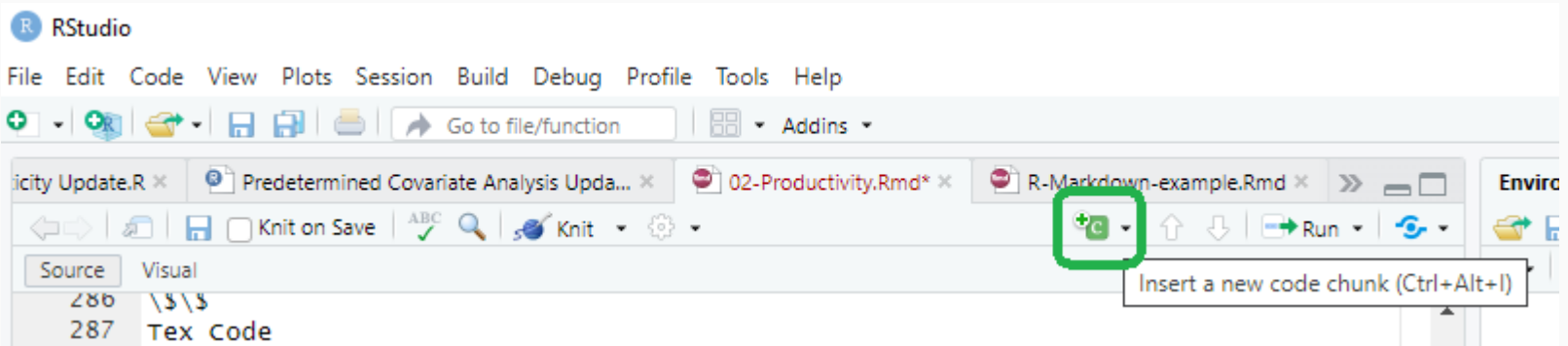
Use the **standard LaTeX commands** for symbols/characters

# Rmd: R Code

R code is primarily executed with **code chunks**

Add a chunk with

- Cmd + Option + I (Ctrl + Alt + I on PC)
- The **Insert** button in the UI
- Manually type





# Rmd: Code Chunks

307

308 ▾ ```{r}

309

310 ▴ ```

311



**Code chunks** allow us to add as many lines of code as we want

- Output will appear underneath after executing the full chunk
- Can customize whether it runs, how output is displayed
- Can run manually
  - Line by line with `Cmd/Ctrl + Enter`
  - Entire chunk with `Run Entire Chunk` button

# Code Chunk Options

You can **add chunk options** in brackets after `r` and separated by commas.

Some commonly-used options include:

- **Chunk label** (`ex_chunk`)
- `include = FALSE` will run the chunk but hide it from the final document
- `eval = FALSE` will display code without evaluating it
- `results = 'hide'` runs code but hides output from the final document

```
52
53 ```{r sum, echo = FALSE, warning = FALSE}
54 2+2
55
56 ```
57
```

# Code Chunk Options

You can also **change the color** of code chunks and/or output in the rendered document through code chunk options<sup>1</sup>

- `class.source` to change the **code chunk**
- `output.source` to change the **output**

## default

```
1 + 2 * 3 ^ 4
```

```
## [1] 163
```

## bg-primary

```
1 + 2 * 3 ^ 4
```

```
## [1] 163
```

<sup>1</sup> We'll chat more about this when we get to web scraping, but what's actually happening here is that we're harnessing some built-in CSS classes to change the backgrounds. This also means that, using CSS, you can define custom classes and format things however you'd like.

# Code Chunk Options

You can also **change the color** of code chunks and/or output in the rendered document through code chunk options<sup>1</sup>

- `class.source` to change the **code chunk**
- `output.source` to change the **output**

**bg-success**

```
1 + 2 * 3 ^ 4
```

```
## [1] 163
```

**bg-info**

```
1 + 2 * 3 ^ 4
```

```
## [1] 163
```

<sup>1</sup> We'll chat more about this when we get to web scraping, but what's actually happening here is that we're harnessing some built-in CSS classes to change the backgrounds. This also means that, using CSS, you can define custom classes and format things however you'd like.

# Code Chunk Options

You can also **change the color** of code chunks and/or output in the rendered document through code chunk options<sup>1</sup>

- `class.source` to change the **code chunk**
- `output.source` to change the **output**

**bg-warning**

```
1 + 2 * 3 ^ 4
```

```
## [1] 163
```

**bg-danger**

```
1 + 2 * 3 ^ 4
```

```
## [1] 163
```

<sup>1</sup> We'll chat more about this when we get to web scraping, but what's actually happening here is that we're harnessing some built-in CSS classes to change the backgrounds. This also means that, using CSS, you can define custom classes and format things however you'd like.

# Rmd: Inline Code

You can call R objects from earlier chunks **inline** with

```
- r -
```

```
four = 2+2
```

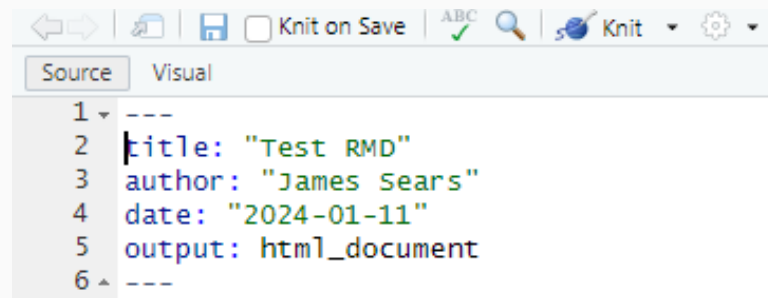
This can output in line with text:  $2 + 2 = 4$

# R Markdown File Organization

# 1. Header

RStudio automatically builds the R Markdown file from a template, which begins with a **header**

- Title
- Author
- Date
- Output Format
  - Main options<sup>1</sup>: HTML (`html_document`), PDF (`pdf_document`), LaTeX (`latex_document`), or Word (`word_document`)<sup>2</sup>

A screenshot of the RStudio Source pane. The pane has two tabs: 'Source' and 'Visual'. The 'Source' tab is active, showing a code editor with line numbers 1 through 6. The code is a YAML header for an R Markdown document. Line 1 is a separator '---'. Line 2 is 'title: "Test RMD"'. Line 3 is 'author: "James Sears"'. Line 4 is 'date: "2024-01-11"'. Line 5 is 'output: html\_document'. Line 6 is another separator '---'. The toolbar at the top includes icons for navigation, saving, and a 'Knit' button with a dropdown arrow.

```
1 ---
2 title: "Test RMD"
3 author: "James Sears"
4 date: "2024-01-11"
5 output: html_document
6 ---
```

1: See [CH 3 of "R Markdown: The Definitive Guide"](#) for more on how to customize [output formats](#)

2: For **better formatted Word output** with greater customisability, use the **officedown** package's `rdox_document` format.



## 2. R Setup

By default, RStudio adds a **setup** code chunk next.

```
8  ```{r setup, include=FALSE}
9  knitr::opts_chunk$set(echo = TRUE)
10 ```
11
```

- Can set global options
- Useful as your preamble
- For **R Notebooks**, this will automatically be run and is the only place where you can change your working directory

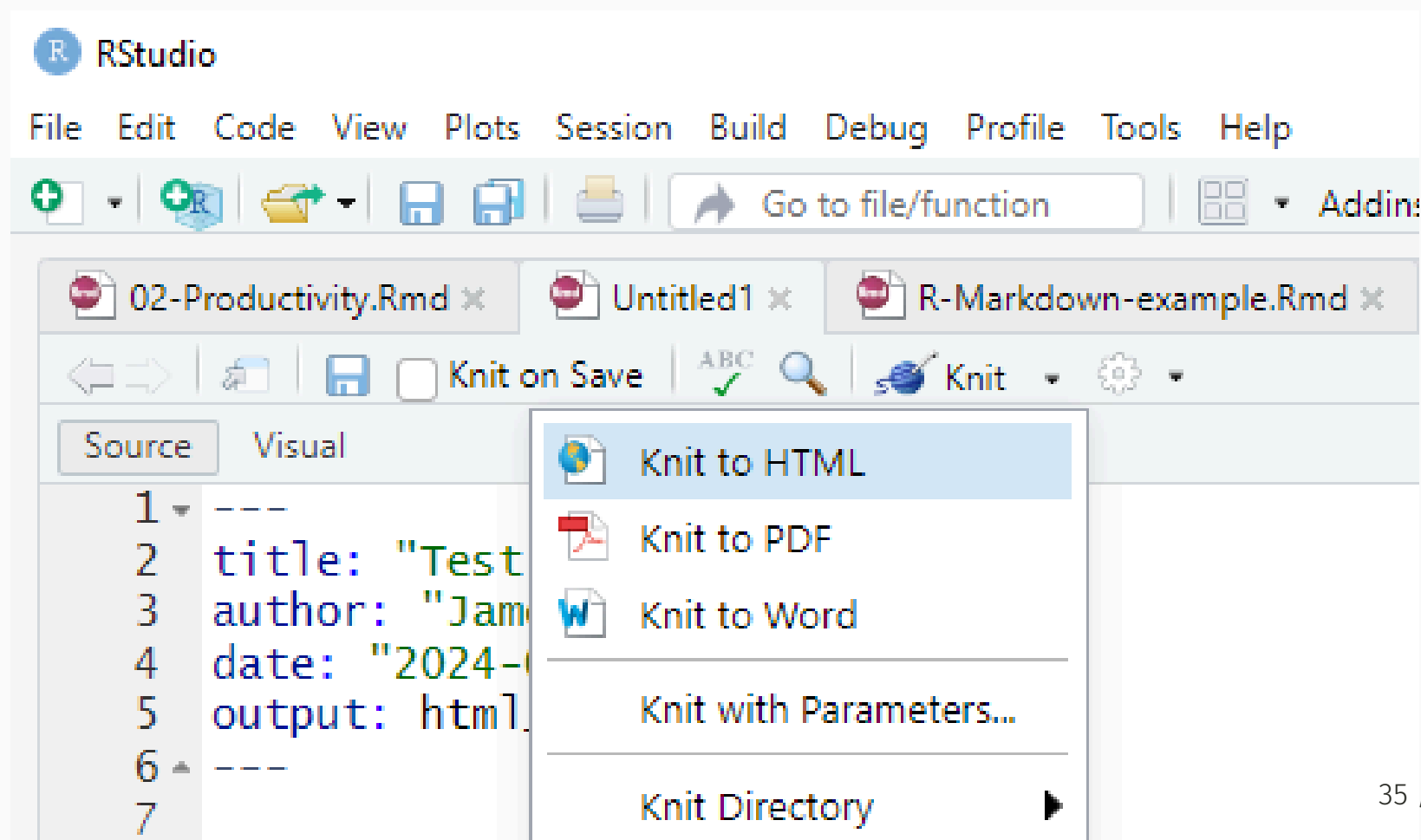
# 3. Contents

From here on you can build the report/notebook as needed for the task.


- Add any writing and outside graphics or **bibTeX citations**
- Add code chunks to carry out desired analysis
- Employ sections and formatting to structure the document as desired

# Compiling/Knitting

When you are ready to compile your final document, use the **Knit** button or **Ctrl/Cmd + Shift + K**



# R Markdown: Knit to Compile Output



The screenshot shows a web browser window displaying an R Markdown document. The browser's address bar shows the file path: `F:/OneDrive - Michigan State University/Teaching/MSU 2023-2024/AFRE 891 SS23/Lecture-Slides/02-Productivity/output/R-Markdown-example.html`. The browser's tab is labeled "R-Markdown-example.html". The document content includes a title "Test R Markdown", a subtitle "James", a date "2024-01-10", and a section header "R Markdown". The text explains that this is an R Markdown document and provides a link to <http://rmarkdown.rstudio.com>. It also mentions the "Knit" button and how it generates a document with both content and R code output. Below the text, there is a code chunk containing the R code `summary(cars)`. The output of this code is displayed as a table with two columns: "speed" and "dist". The table shows summary statistics for the "cars" dataset, including minimum, 1st quartile, median, mean, 3rd quartile, and maximum values for both speed and distance.

Test R Markdown

James

2024-01-10

## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

##	speed	dist
## Min.	: 4.0	Min. : 2.00
## 1st Qu.	:12.0	1st Qu.: 26.00
## Median	:15.0	Median : 36.00
## Mean	:15.4	Mean : 42.98
## 3rd Qu.	:19.0	3rd Qu.: 56.00
## Max.	:25.0	Max. :120.00

# Markdown Practice!

# Markdown Practice

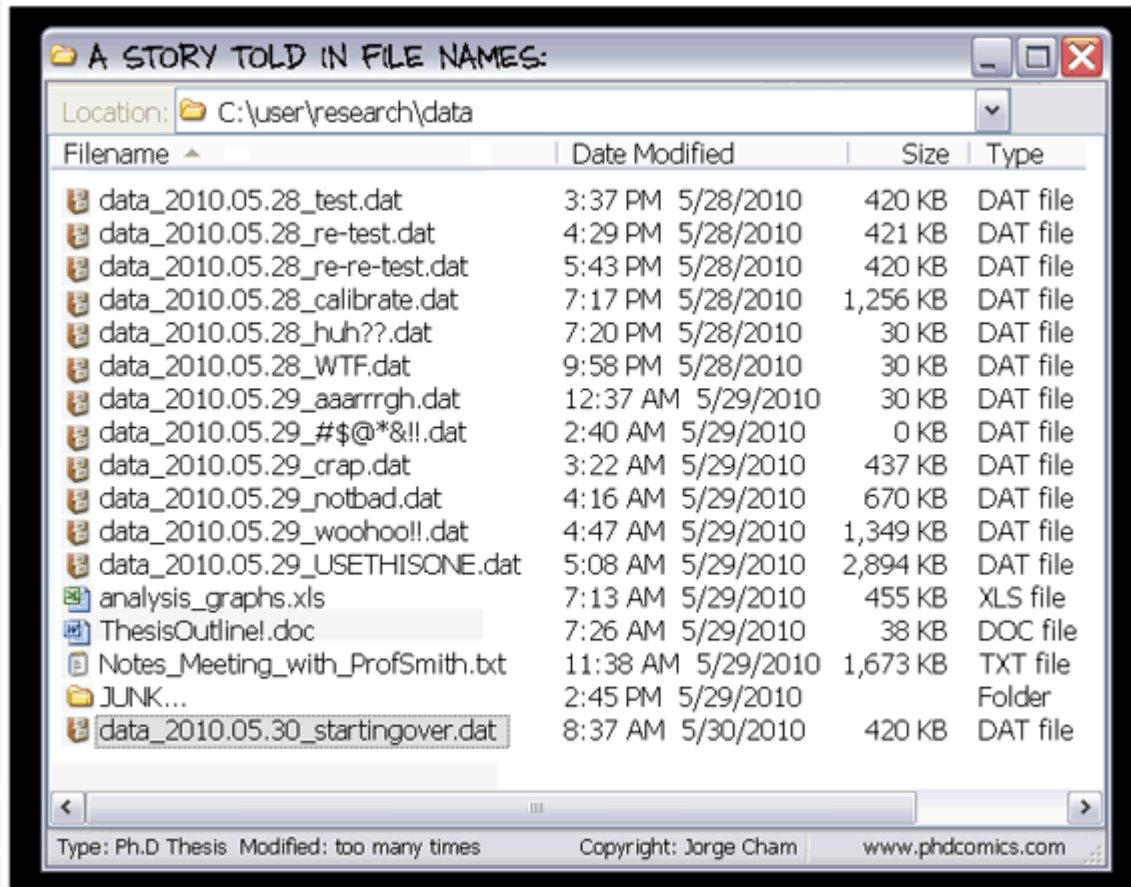
1. Create a new R Markdown file named "R-Markdown-Ex.Rmd"
2. In the setup chunk, load the **dslabs** and **tidyverse** packages
  - Use the `data()` function to read in the `divorce_margarine` dataset
3. Add a header labeled "Correlation vs. Causation" and a text explanation below for why we often want to differentiate between the two
4. Add a code chunk with the label `plot`
  - Type the following code:

```
ggplot(divorce_margarine) +  
  geom_point(aes(x = margarine_consumption_per_capita,  
                 y = divorce_rate_maine)) +  
  labs(title = "Relationship between Margarine Consumption and  
             Divorce Rates in Maine",  
        subtitle = "2000-2009",  
        x = "Margarine Consumption per Capita",  
        y = "Divorce Rate")
```

5. Knit and save a PDF/HTML copy of the file to the "output" folder

# Version Control

# Why Use Version Control





# Goals of Version Control

While building project folders with the above naming conventions is *fun*, a good **version control system** can solve this problem.

- Save each set of changes sequentially
- Keep track of different versions of a file
- Merge changes from multiple versions/sources

# Git(Hub) Solves this Problem

## Git

- **Git** is a **distributed version control system**
  - Each team member has a **local copy** of files on their computer
- Imagine if Dropbox and the "Track changes" feature in MS Word had a baby. Git would be that baby.
- In fact, it's even better than that because Git is optimised for the things that economists and data scientists spend a lot of time working on (e.g. code).
- There is a learning curve, but I promise you it's worth it.

# Git(Hub) Solves this Problem

## GitHub

- It's important to realise that **Git** and **GitHub** are distinct things.
- **GitHub** is an **online hosting platform** that provides an array of services built on top of the **Git** system. (Similar platforms include Bitbucket and GitLab.)
- Just like we don't *need* **Rstudio** to run **R** code, we don't *need* **GitHub** to use **Git**... but it will make our lives so much easier.

# Git(Hub) for Scientific Research

## From software development...

- **Git** and **GitHub**'s role in global software development is not in question.
- There's a high probability that your favorite app, program or package is built using Git-based tools. (RStudio is a case in point.)

## ... to scientific research

- Benefits of VC and collaboration tools aside, Git(Hub) helps to operationalise the ideals of open science and reproducibility.<sup>2</sup>
- Journals have increasingly strict requirements regarding reproducibility and data access. GH makes this easy (DOI integration, off-the-shelf licenses, etc.)
- I host [teaching materials](#) on GH. I even use it to host and maintain my [website](#) for free.  
2: [Democratic databases: Science on GitHub \(Nature\)](#) (Perkel, 2016).

# Using GitHub

There are a couple of different main ways that we could use GitHub:

## 1. Through [github.com](https://github.com) Only

- **Pros:** doesn't require any software/local repo copies
- **Cons:** much more time-intensive and not automated (the whole point of this thing!)

## 2. Through the command line ([GitHub CLI](#))

- **Pros:** fully programmatic, requires no additional software
- **Cons:** fully programmatic!

# Using GitHub

There are a couple of different main ways that we could use GitHub:

## 3. Integrated with RStudio

- **Pros:** fully integrates RStudio projects
- **Cons:** limited to R projects, doesn't play as nicely with GitHub Classroom

## 4. Through the GitHub Desktop App

- **Pros:** Intuitive GUI, sync any kinds of files/projects
- **Cons:** requires an extra piece of software, [GitHub Desktop](#)

We're going to focus primarily on **4**, but the back of the deck will contain slides working through **3** if you want to experiment with that too.

# GitHub Desktop

# Version Control with GitHub Desktop

Although GitHub integration with RStudio has lots of functionality, there are times where we want to keep track of files and projects **outside of RStudio**.

- For example, when you want version control of projects that **don't only use R** (or don't use it at all)

This is where **GitHub Desktop** comes in.



# Version Control with GitHub Desktop

This next section is about learning the basic Git(Hub) commands and the recipe for successful version control with GitHub Desktop.

I also want to bookmark a general point that we'll revisit many times during this course:

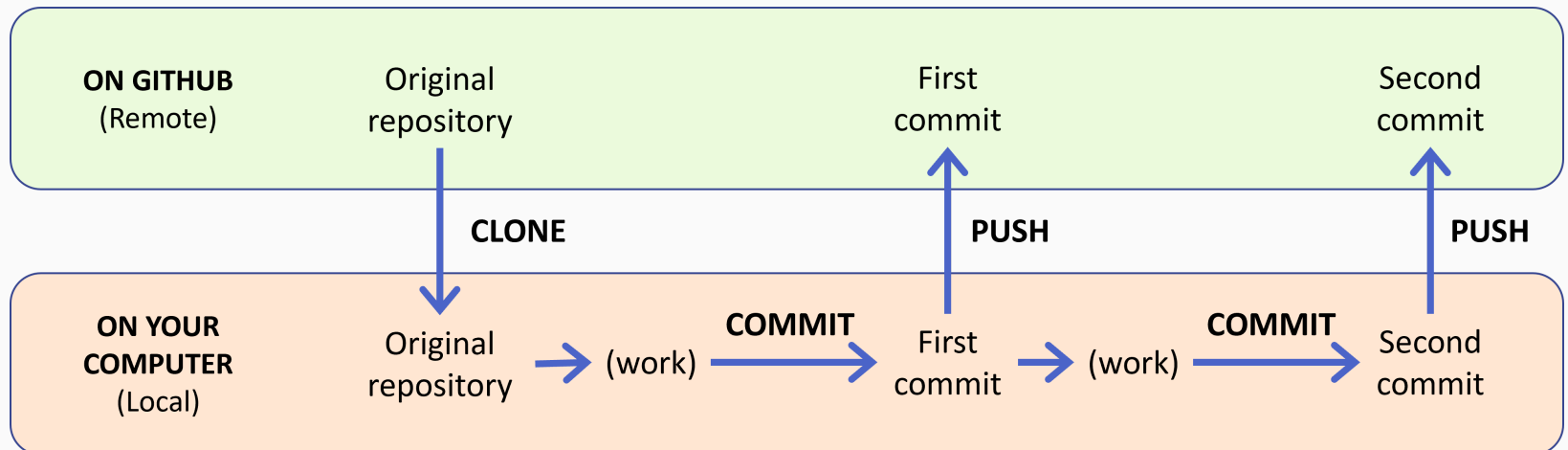
- The tools that we're using all form part of a coherent **data science ecosystem**.
- Greatly reduces the cognitive overhead ("aggregation") associated with traditional workflows, where you have to juggle multiple programs and languages at the same time.

# Github Desktop Workflow

With GitHub, if we were working

1. On our own
2. From a single computer

we could just follow the below workflow:



Since we're collaborating with others/potentially across machines, we'll also add in a few more actions.

# Main Git operations

The first Git operation in the workflow is **Cloning**



# (Git) Cloning

No, not *that* kind of cloning.

**Cloning:** making a local copy of a **GitHub Repository** (repo for short).

In order to clone a repo, we first need a repo to clone. Let's start by cloning our Course repo!

We can do this

1. From the repo page on github.com (direct link or SSH)
2. From GitHub Desktop directly

# Course Repo Cloning (github.com)

Let's start with the first approach. With GitHub Desktop installed/open, navigate to our class web page repo.

The screenshot shows the GitHub interface for the repository 'afre-msu / AFRE-891-991-FS25'. The repository is public and has 0 stars, 0 forks, and 0 watchers. The main branch is 'main'. The repository description is 'Course Repository for AFRE 891/991, Advanced Data Analytics, Fall Semester 2025'. The repository contains a README, a License, and two folders: 'Course Logistics' and 'Lecture Slides'. The 'Course Logistics' folder contains a 'LICENSE' file and a 'README.md' file. The 'Lecture Slides' folder contains a 'README.md' file. The repository was last updated by 'searsjm' last week.

**Repository Details:**

- Repository: **afre-msu / AFRE-891-991-FS25** (Public)
- Stars: 0
- Forks: 0
- Watchers: 0

**Files and Folders:**

Item	Description	Last Updated
Course Logistics	Add Course Logistics, Lecture 0	last week
Lecture Slides	Add Lecture 1 Slides	last week
LICENSE	Add Course Logistics, Lecture 0	last week
README.md	Update README.md	last week

**About:**

- Course Repository for AFRE 891/991, Advanced Data Analytics, Fall Semester 2025
- Readme
- View license
- Activity
- Custom properties
- 0 stars
- 0 watching
- 0 forks
- Report repository

# Course Repo Cloning

Click the green **Code** button and then "Open with GitHub Desktop"

The screenshot shows the GitHub interface for the repository 'AFRE-891-991-FS25'. The repository is public and has a 'main' branch. The file list on the left includes 'Course Logistics', 'Lecture Slides', 'LICENSE', and 'README.md'. The 'README' file is selected. A green arrow points to the 'Code' button in the top right. Another green arrow points to the 'Open with GitHub Desktop' option in the 'Clone' dropdown menu. The dropdown menu also shows options for 'Local' and 'Codespaces', and provides the HTTPS URL for cloning: 'https://github.com/afre-msu/AFRE-891-991-FS25.git'. The page title at the bottom is 'Course Materials for AFRE 891/991'.

AFRE-891-991-FS25 Public

Edit Pins Watch 0

main

searsjm Update README.md

- Course Logistics
- Lecture Slides
- LICENSE
- README.md

README License

Local Codespaces

Clone

HTTPS SSH GitHub CLI

`https://github.com/afre-msu/AFRE-891-991-FS25.git`

Clone using the web URL.

Open with GitHub Desktop

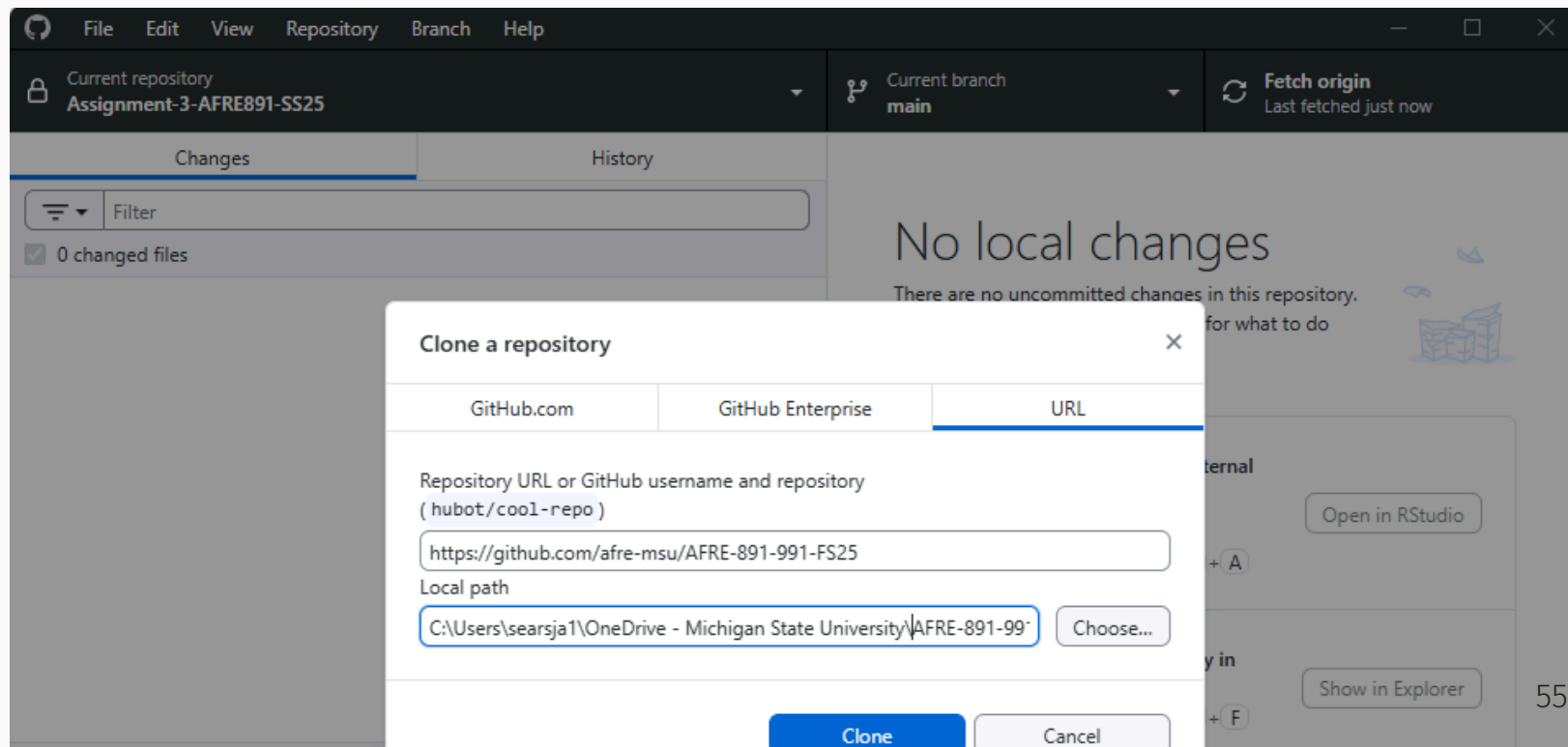
Download ZIP

## Course Materials for AFRE 891/991

# Course Repo Cloning

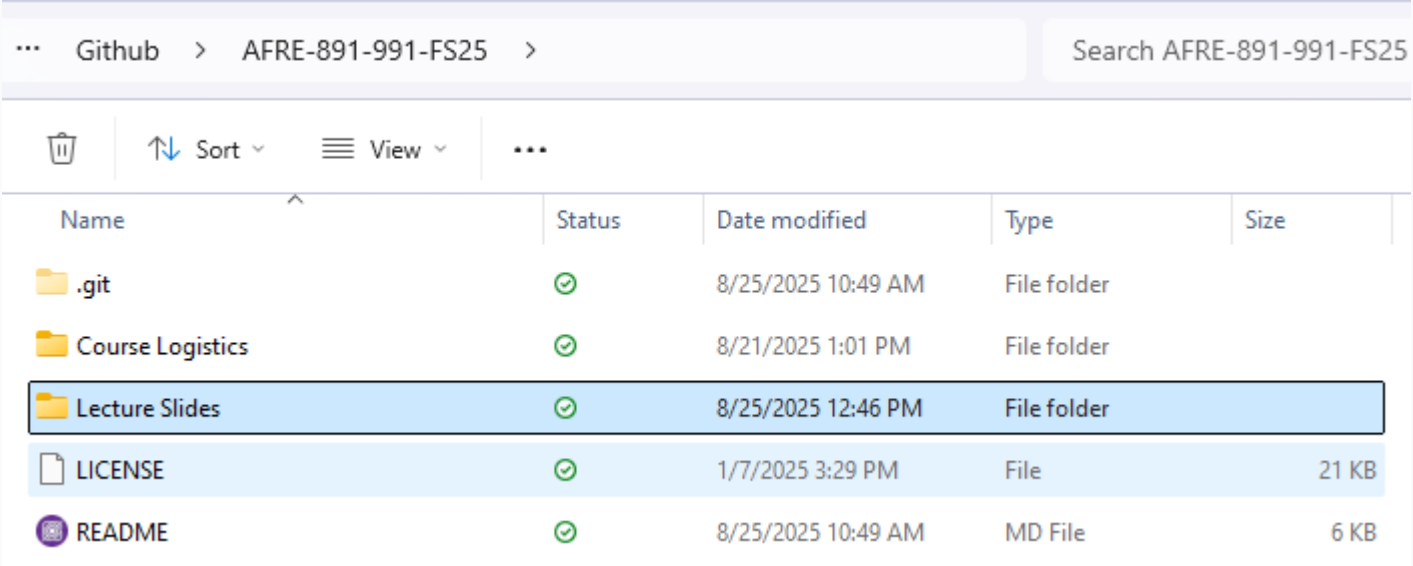
Allow your web browser to open the link. This will redirect you to GitHub Desktop, automatically adding in the repo URL.

Next, choose where you want the local copy of the repo file's saved and hit **Clone**



# Course Repo Cloning

Wait a little bit, and navigate to the local path you gave it. Voila!



The screenshot shows a file explorer window for a repository named 'AFRE-891-991-FS25' on Github. The interface includes a search bar, a toolbar with icons for trash, sort, view, and a menu, and a table of files and folders. The 'Lecture Slides' folder is highlighted.

Name	Status	Date modified	Type	Size
.git	✓	8/25/2025 10:49 AM	File folder	
Course Logistics	✓	8/21/2025 1:01 PM	File folder	
Lecture Slides	✓	8/25/2025 12:46 PM	File folder	
LICENSE	✓	1/7/2025 3:29 PM	File	21 KB
README	✓	8/25/2025 10:49 AM	MD File	6 KB



# Table of Contents

1. **Prologue**
2. **R Markdown**
3. **Version Control**
4. **GitHub Desktop - Part 1**