

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

James Sears*

AFRE 891/991 SS 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. [Git\(Hub\) + RStudio](#)
5. [GitHub Desktop](#)
6. [Other Tips and Productivity Tools](#)
7. [Not Covered: Troubleshooting Git Credential Issues in RStudio](#)

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
- [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

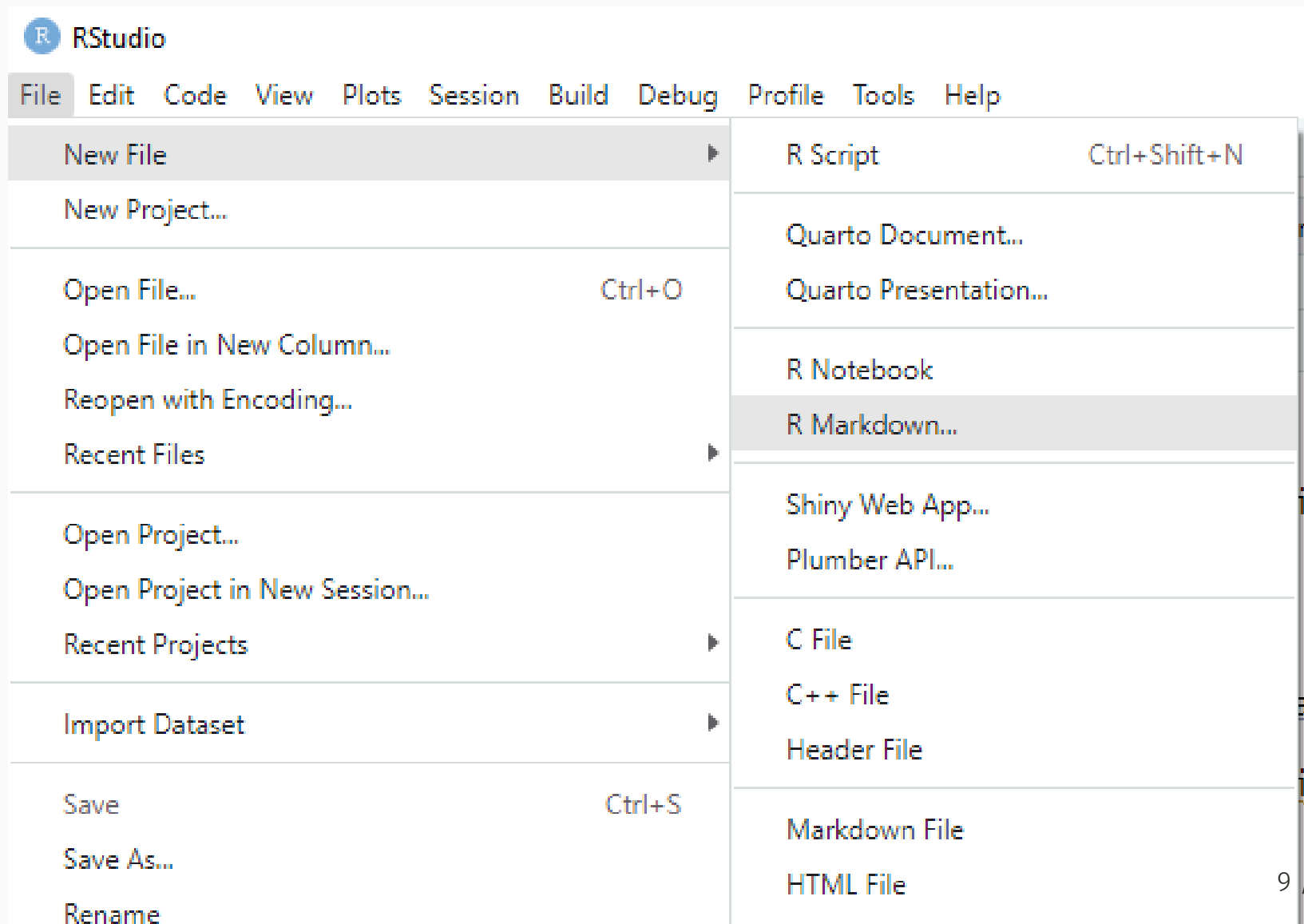
```
install.packages("rmarkdown")
```

☐ Install LaTeX

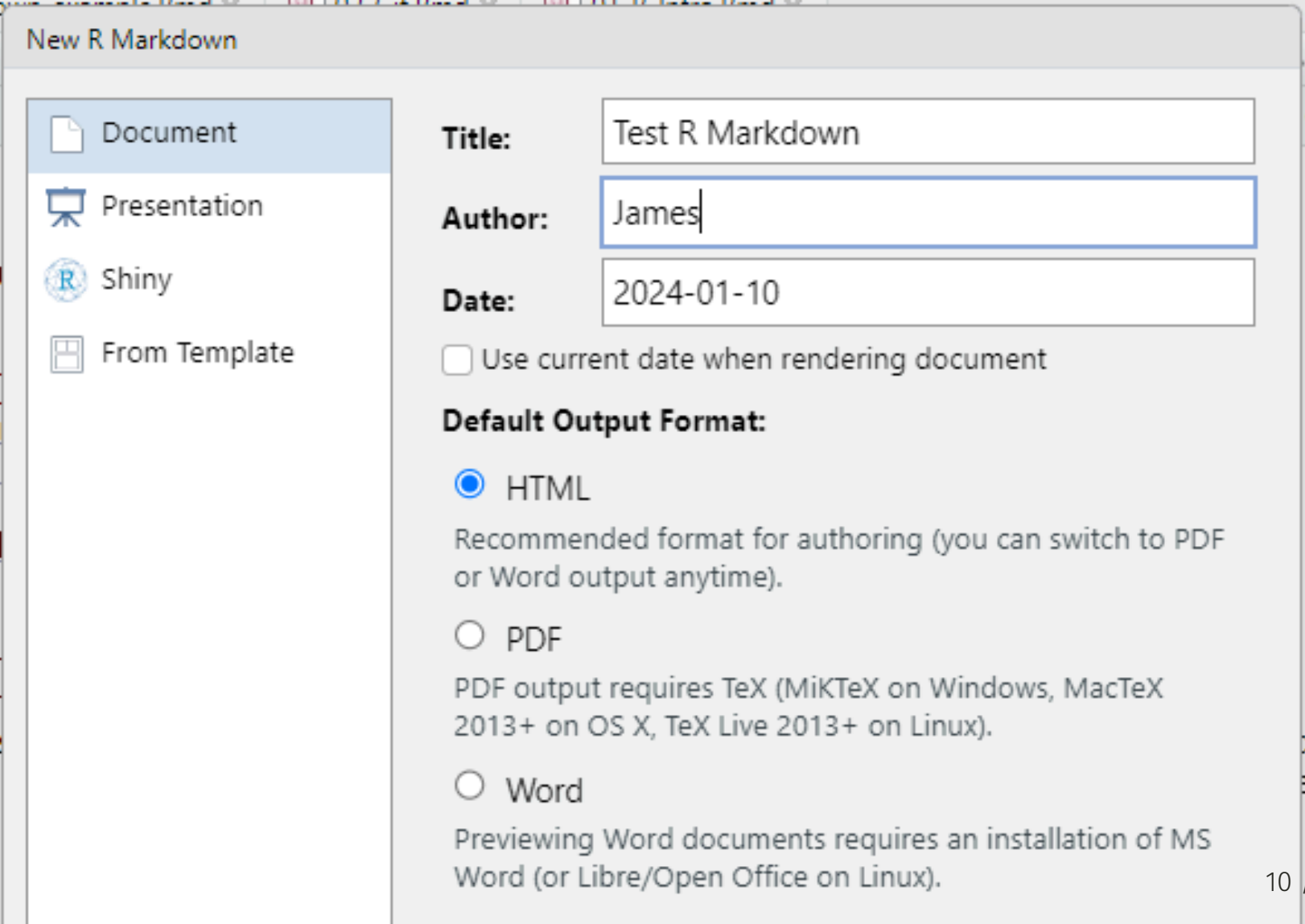
- If just for this, can use [TinyTex](#)

```
# 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

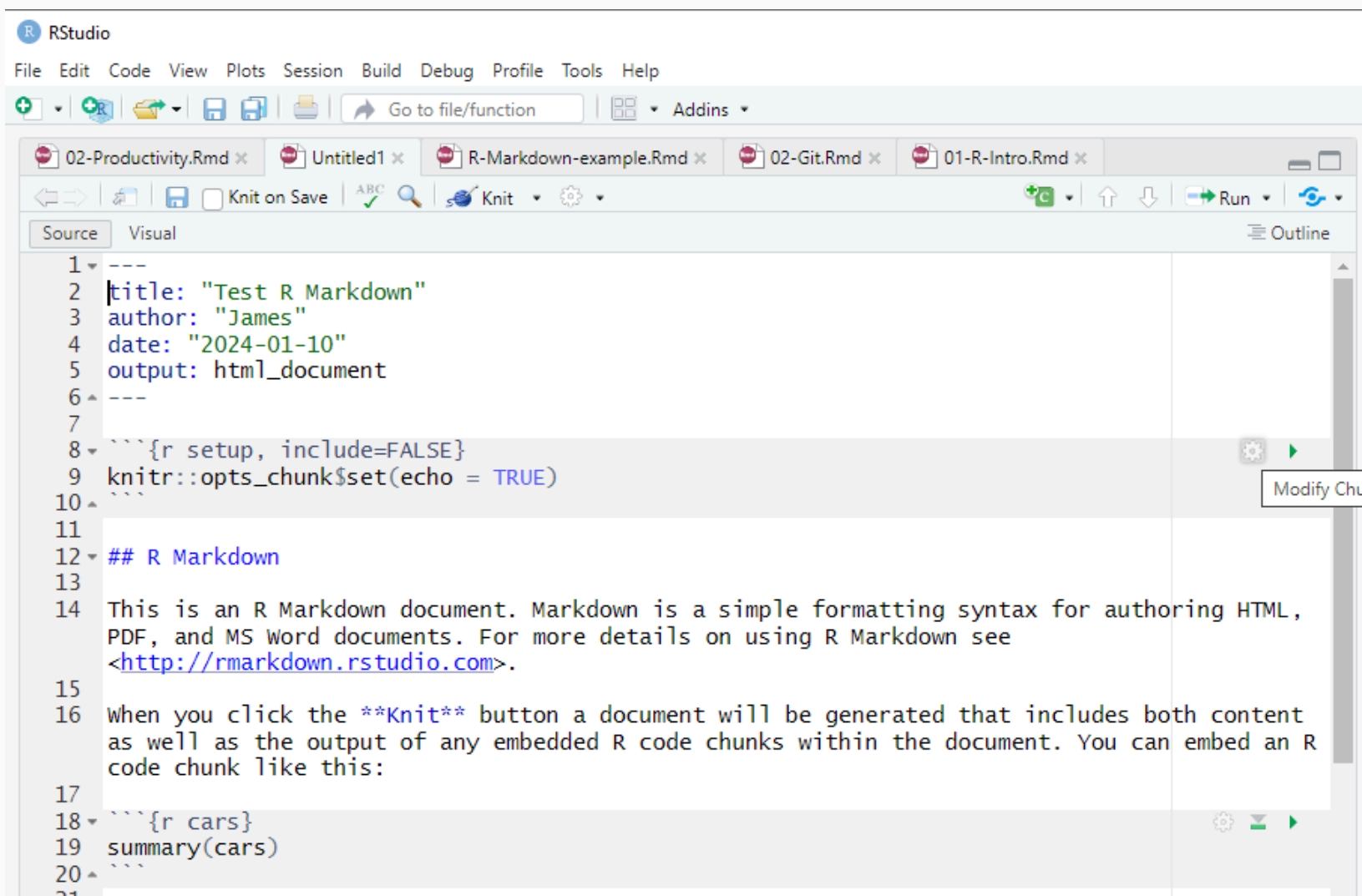
Default Output Format:

☒ 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 superscripts² 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: 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`:



R Markdown: LaTeX

Another advantage of R Markdown is that it integrates \LaTeX functionality for typesetting math.

Add an **inline equation** with $\text{\$TeX\$}$

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

Add multiple rows of LaTeX with

$\text{\$}$

LaTeX lines here

$\text{\$}$

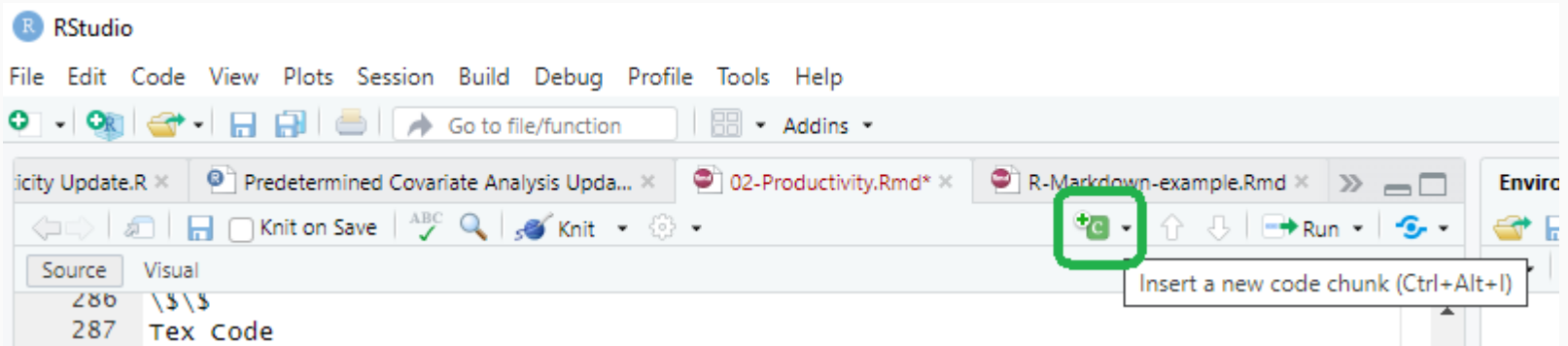
Use the **standard LaTeX commands** for symbols/characters

R Markdown: 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



R Markdown: R Code Chunks

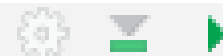
307

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

R Markdown: R 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
```

R Markdown: R Code Chunk Options

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

Some commonly-used options include:

- `echo = FALSE` runs the code but hides the chunk from the final document
- `error = FALSE` (`warning = FALSE`) will hide error (warning) messages generated by the code
- LOTS of options for output figures: figure size (`fig.width`, `fig.height`, `fig.dim`), output document scale (`out.width`, `out.height`), alignment (`fig.align`), caption (`fig.cap`)

Learn more [about chunk options here](#)

R Markdown: R Code

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

`< r <`

```
four = 2+2
```

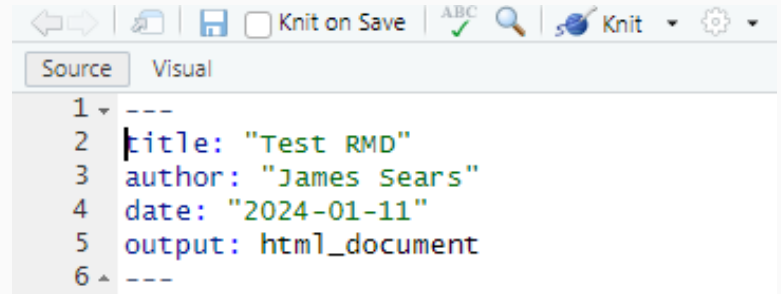
This can output in line with text: $2 + 2 = 4$

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 options¹: HTML
(`html_document`), PDF
(`pdf_document`), LaTeX
(`latex_document`), or Word
(`word_document`)



The screenshot shows the RStudio Source editor with the 'Source' tab selected. The editor displays the header of an R Markdown file, which is enclosed in a dashed line. The header contains the following text: `title: "Test RMD"`, `author: "James Sears"`, `date: "2024-01-11"`, and `output: html_document`. The line numbers 1 through 6 are visible on the left side of the editor.

```
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. 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

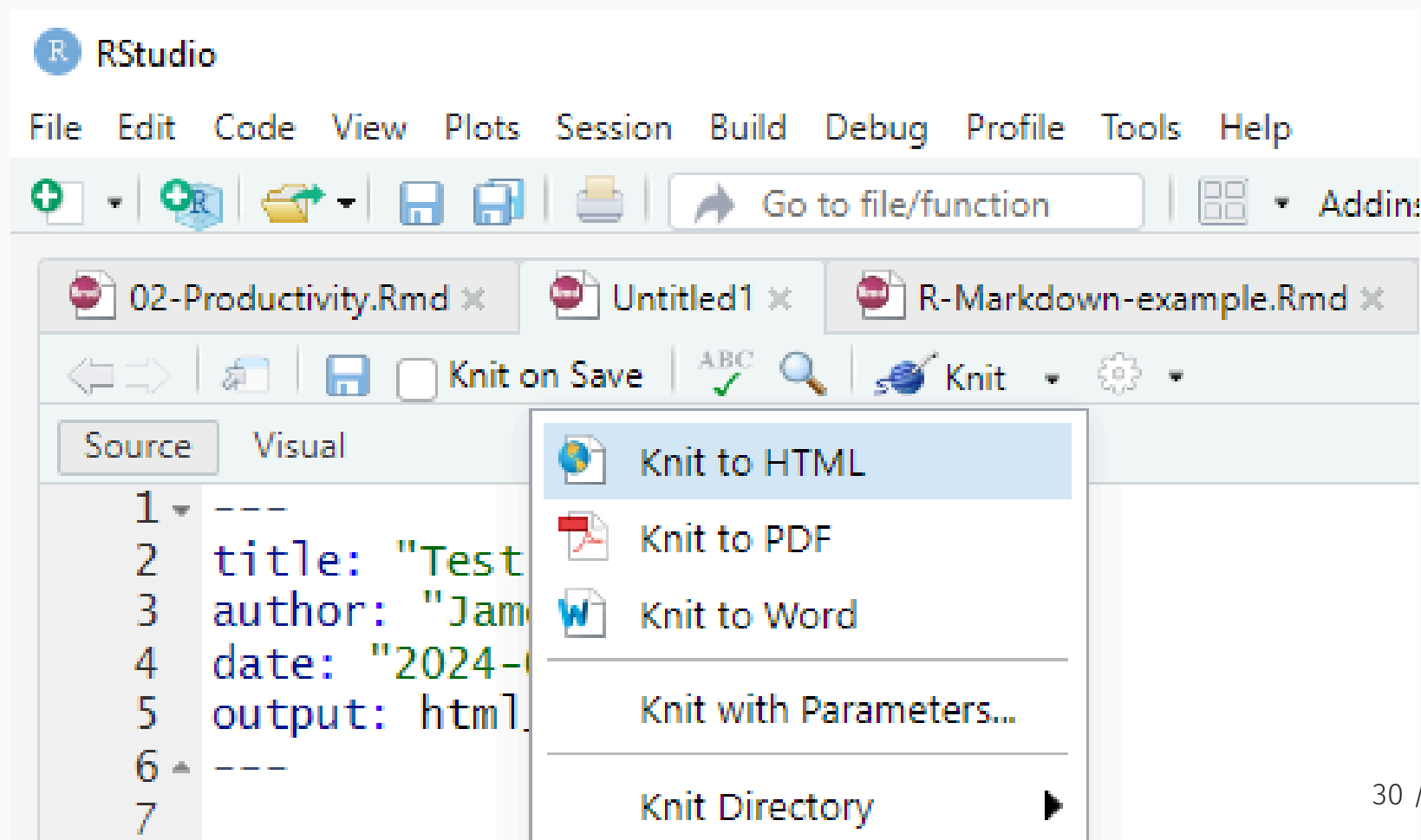
3. Contents

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, author, date, a section header, an introductory paragraph, a paragraph about the Knit button, and a code chunk with its output.

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

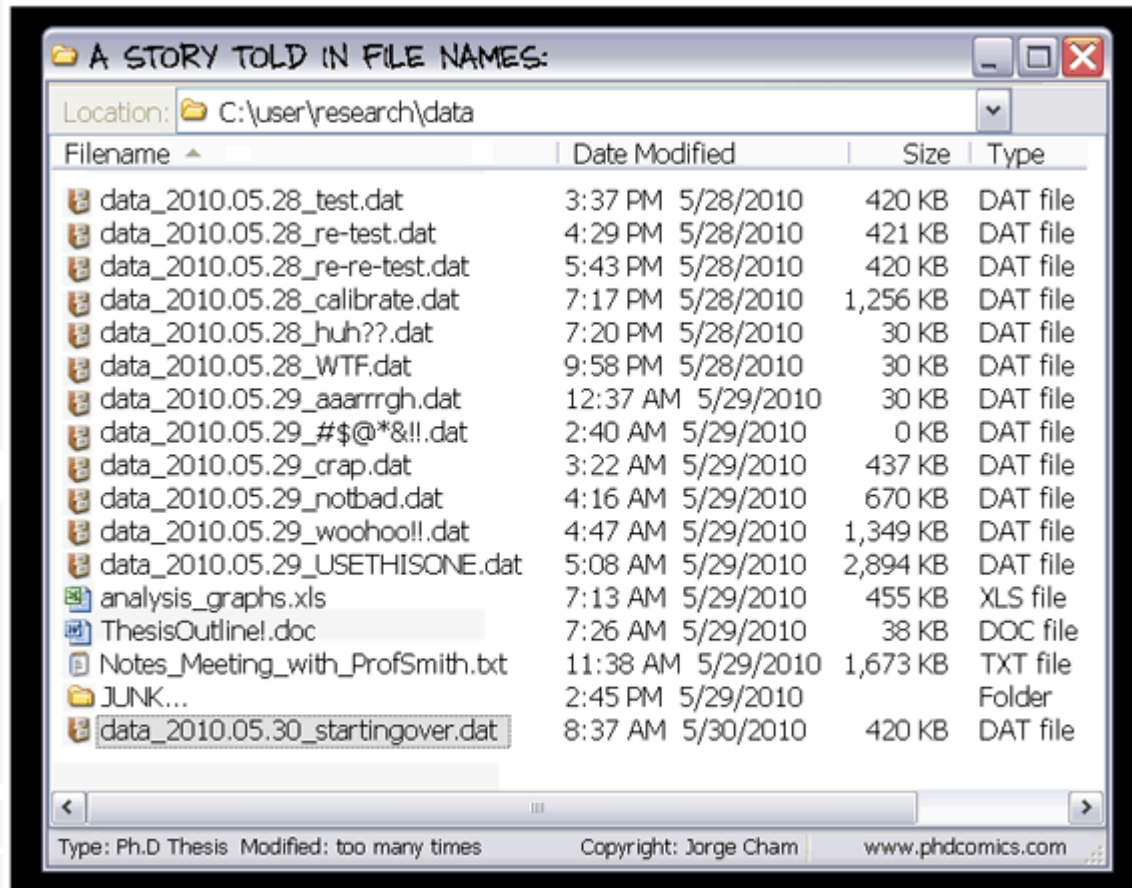
1. Create a new R Markdown file named "R-Markdown-Ex.RMD"
2. In the setup chunk, load the `dslabs` and `tidyverse` packages
 - Use the `data()` function to read in the `divorce_margarine` dataset
3. Add a header labeled "Correlation vs. Causation" and a text explanation below for why we often want to 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 favourite 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.²
- 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.

2: Democratic databases: Science on GitHub (Nature) (Perkel, 2016).

Git(Hub) and RStudio

Seamless Integration

One of the (many) great features of RStudio is how well it integrates version control into your everyday workflow.

- Even though Git is a completely separate program to R, they feel like part of the same "thing" in RStudio.
- This next section is about learning the basic Git(Hub) commands and the recipe for successful project integration with RStudio.

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.

Link a GitHub Repo to an RStudio

The starting point for our workflow is to link a GitHub repository (i.e. "repo") to an RStudio Project. Here are the steps we're going to follow:

1. Create the repo on GitHub and initialize with a README.
2. Copy the HTTPS/SSH link (the green "Clone or Download" button).¹
3. Open up RStudio.
4. Navigate to **File -> New Project -> Version Control -> Git**.
5. Paste your copied link into the "Repository URL:" box.
6. Choose the project path ("Create project as subdirectory of:") and click **Create Project**.

Now, I want you to practice by these steps by creating your own repo on GitHub — call it "test" — and cloning it via an RStudio Project.

Create a Repository on GitHub (Repo)

github.com/searsjm?tab=repositories

searsjm

Overview Repositories 4 Projects Packages Stars

Your repository "searsjm/test" was successfully deleted.

Find a repository...

searsjm.github.io Public
Personal Website
CSS GNU General Public License v3.0 Updated on Jun 12, 2023

R-Notebook-Test Public
Jupyter Notebook BSD 3-Clause "New" or "Revised" License Updated on Jun 27, 2020

datahub Public
Forked from [vinitra-zz/datahub](#)

James Sears
searsjm

- New repository
- Import repository
- New codespace
- New gist
- New organization
- New project

Create a Repository on GitHub (Repo)

← → ↻ 🏠 github.com/new

☰ New repository 🔍 Type to search | >_ |

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk ().*

Repository template

No template ▾

Start your repository with a template repository's contents.

Owner * **Repository name ***

searsjm ▾ /

✔ test is available.

Great repository names are short and memorable. Need inspiration? How about [fuzzy-dollop](#) ?

Description (optional)

☒ **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐ **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

☒ **Add a README file**
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: None ▾

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: MIT License ▾

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

Create a Repository on GitHub (Repo)

The screenshot shows the GitHub interface for a repository named 'test' by user 'searsjm'. The repository is public and has 1 branch (main) and 0 tags. It contains 1 commit (36b4f4e) with files LICENSE and README.md. The README file is selected, showing the title 'test' and the description 'Test Repository'. The right sidebar shows repository statistics: 0 stars, 1 watching, and 0 forks. It also includes links for 'About', 'Releases', and 'Packages'.

searsjm / test

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

test Public Pin Unwatch 1 Fork 0 Star 0

main 1 Branch 0 Tags Go to file t + <> Code

searsjm Initial commit 36b4f4e · now 1 Commits

LICENSE	Initial commit	now
README.md	Initial commit	now

README License

test

Test Repository

About

Test Repository

- Readme
- Activity
- 0 stars
- 1 watching
- 0 forks

Releases

No releases published
[Create a new release](#)

Packages

No packages published
[Publish your first package](#)

Create a Repository on GitHub (Repo)

The screenshot shows the GitHub interface for a repository named 'test' by user 'searsjm'. The repository is public and has 1 branch (main) and 0 tags. The 'Code' dropdown menu is open, displaying options to clone the repository using HTTPS, SSH, or GitHub CLI. The HTTPS URL is highlighted: `https://github.com/searsjm/test.git`. Below the URL, it says 'Use Git or checkout with SVN using the web URL.' Other options include 'Open with GitHub Desktop' and 'Download ZIP'. The repository's README is visible in the background, showing the title 'test' and the description 'Test Repository'.

searsjm / test

Type to search

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

test Public

Pin Unwatch 1 Fork 0 Star 0

main 1 Branch 0 Tags

searsjm Initial commit

LICENSE Initial commit

README.md Initial commit

README License

test

Test Repository

Go to file

Local Codespaces

Clone

HTTPS SSH GitHub CLI

`https://github.com/searsjm/test.git`

Use Git or checkout with SVN using the web URL.

Open with GitHub Desktop

Download ZIP

About

Test Repository

Readme

Activity

0 stars

1 watching

0 forks

Releases

No releases published

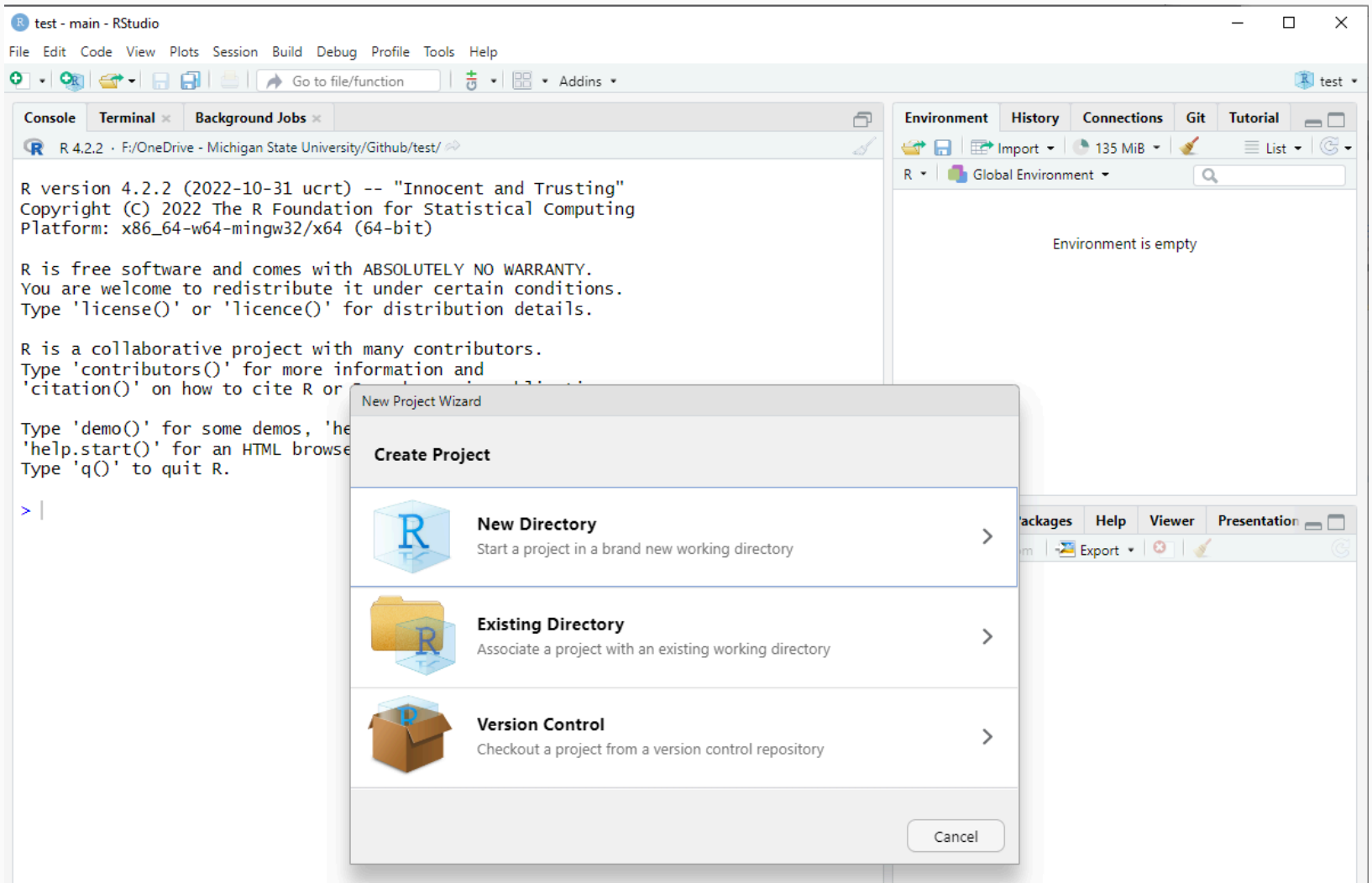
[Create a new release](#)

Packages

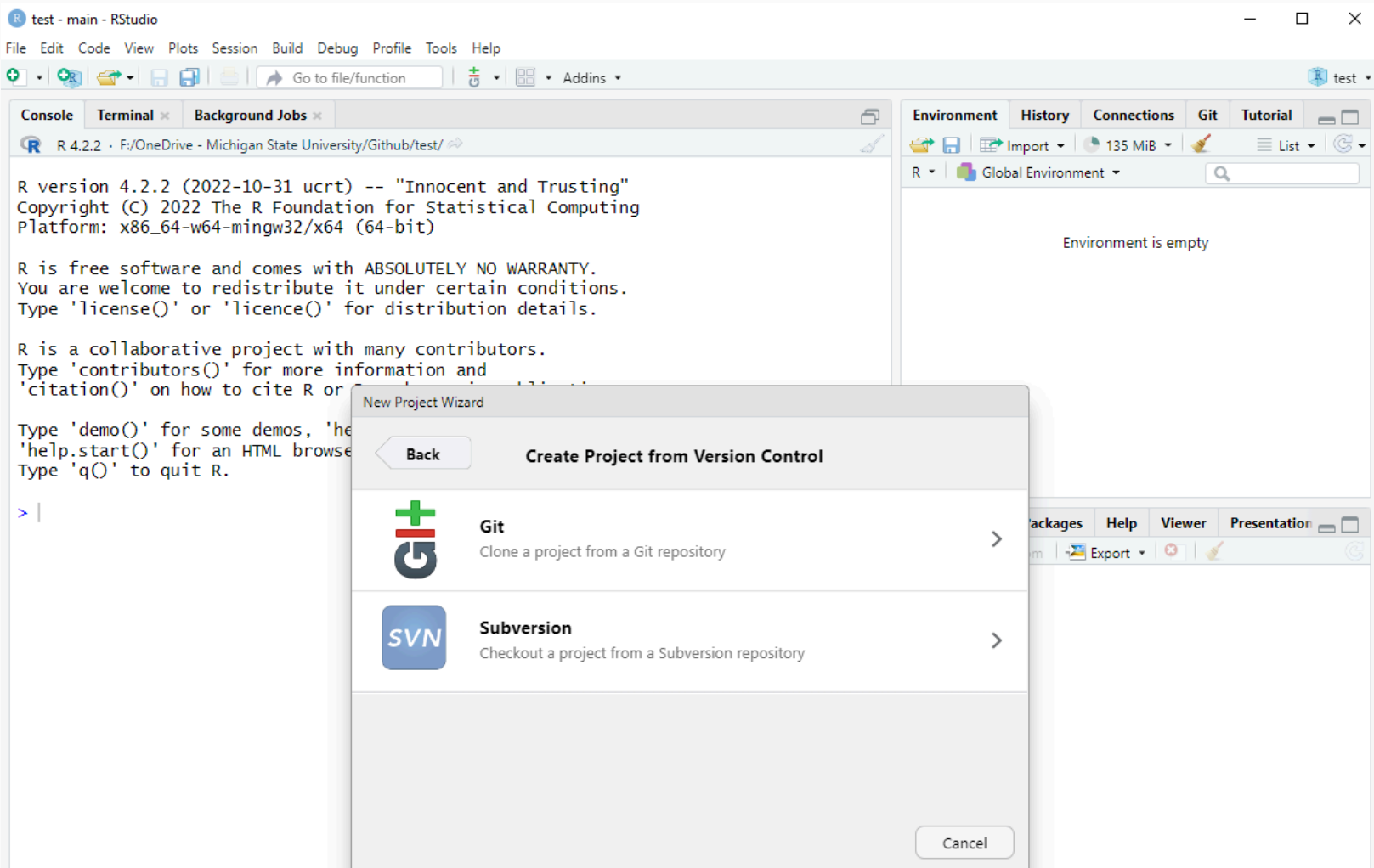
No packages published

[Publish your first package](#)

Copy Repo Link




Add Repo into RStudio



Add Repo into RStudio

New Project Wizard

[Back](#) **Clone Git Repository**



Repository URL:

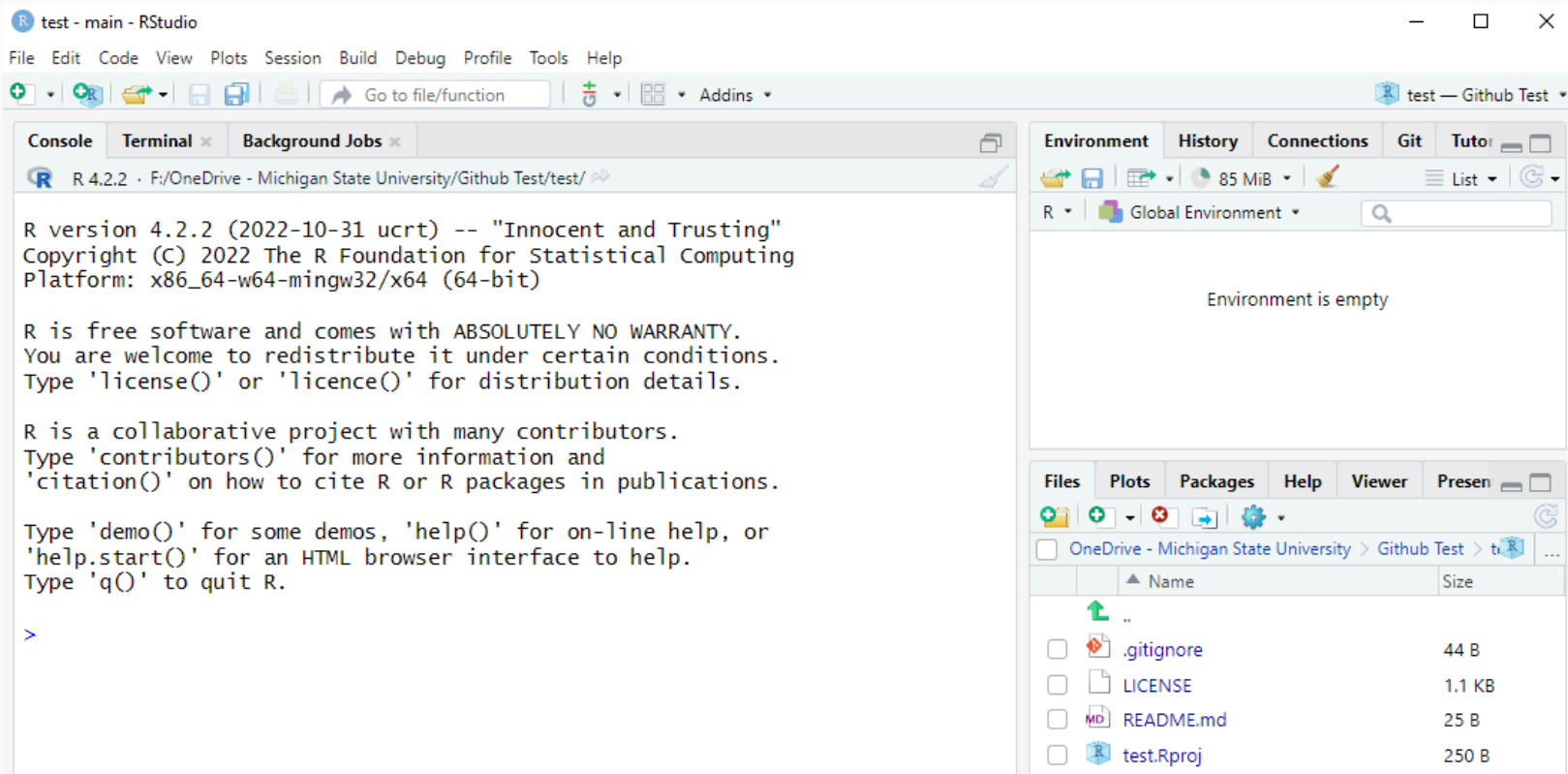
Project directory name:

Create project as subdirectory of:
 [Browse...](#)

☐ Open in new session

[Create Project](#) [Cancel](#)

Add Repo into RStudio



The screenshot shows the RStudio interface with the following components:

- Console:** Displays the R version 4.2.2 (2022-10-31 ucrt) -- "Innocent and Trusting" and copyright information. It also shows the R startup message: "R is free software and comes with ABSOLUTELY NO WARRANTY. You are welcome to redistribute it under certain conditions. Type 'license()' or 'licence()' for distribution details. R is a collaborative project with many contributors. Type 'contributors()' for more information and 'citation()' on how to cite R or R packages in publications. Type 'demo()' for some demos, 'help()' for on-line help, or 'help.start()' for an HTML browser interface to help. Type 'q()' to quit R."
- Files Pane:** Shows the directory structure of the project. The path is `OneDrive - Michigan State University > Github Test > test/`. The files listed are:

Name	Size
..	
.gitignore	44 B
LICENSE	1.1 KB
README.md	25 B
test.Rproj	250 B
- Environment Pane:** Shows the Global Environment, which is currently empty.

Making Local Changes

Look at the **top-right panel** in your RStudio IDE. Do you see the **Git** tab?

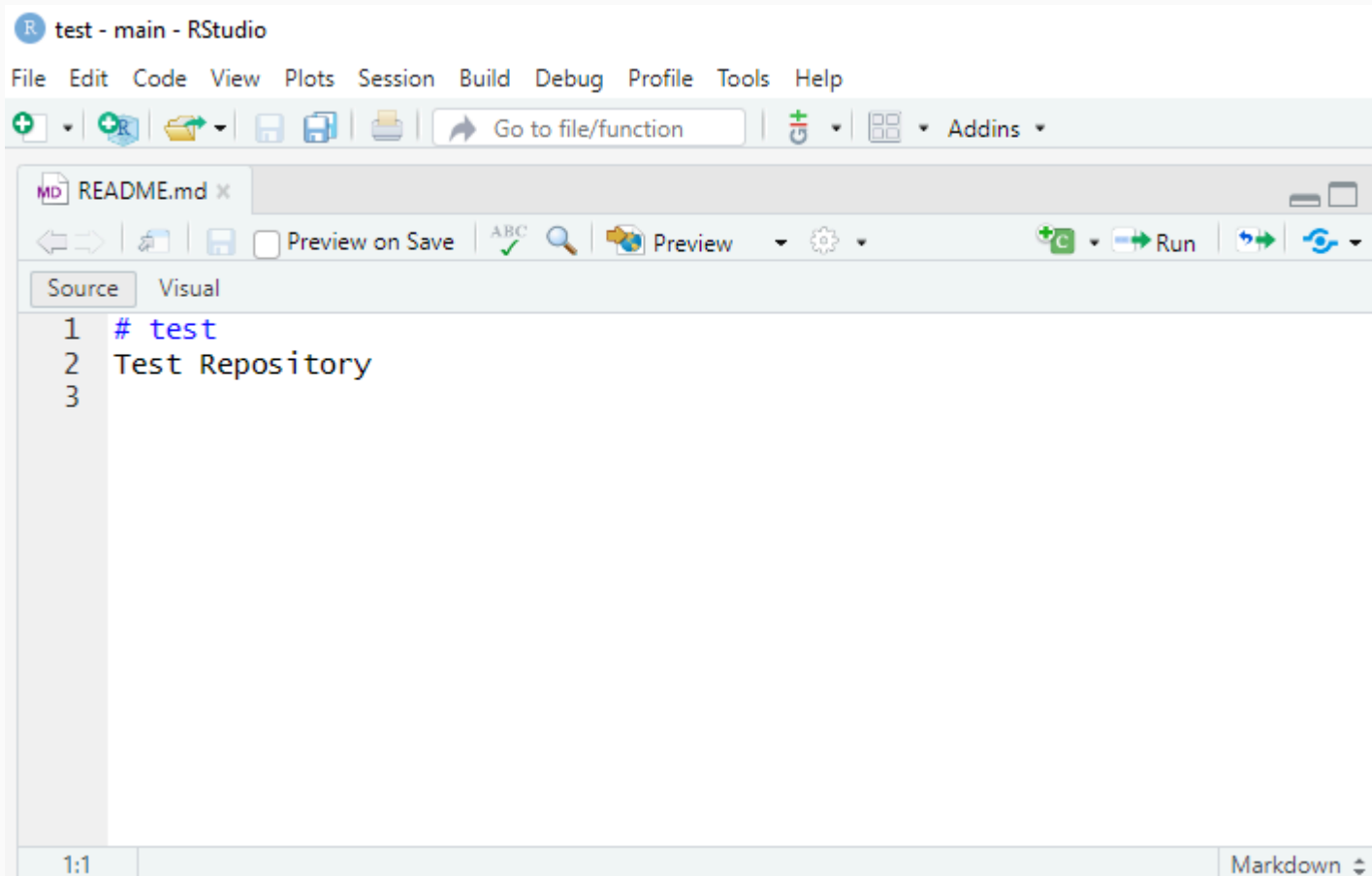
- Click on it.
- There should already be some files in there, which we'll ignore for the moment.¹

Now open up the README file (see the "Files" tab in the bottom-right panel).

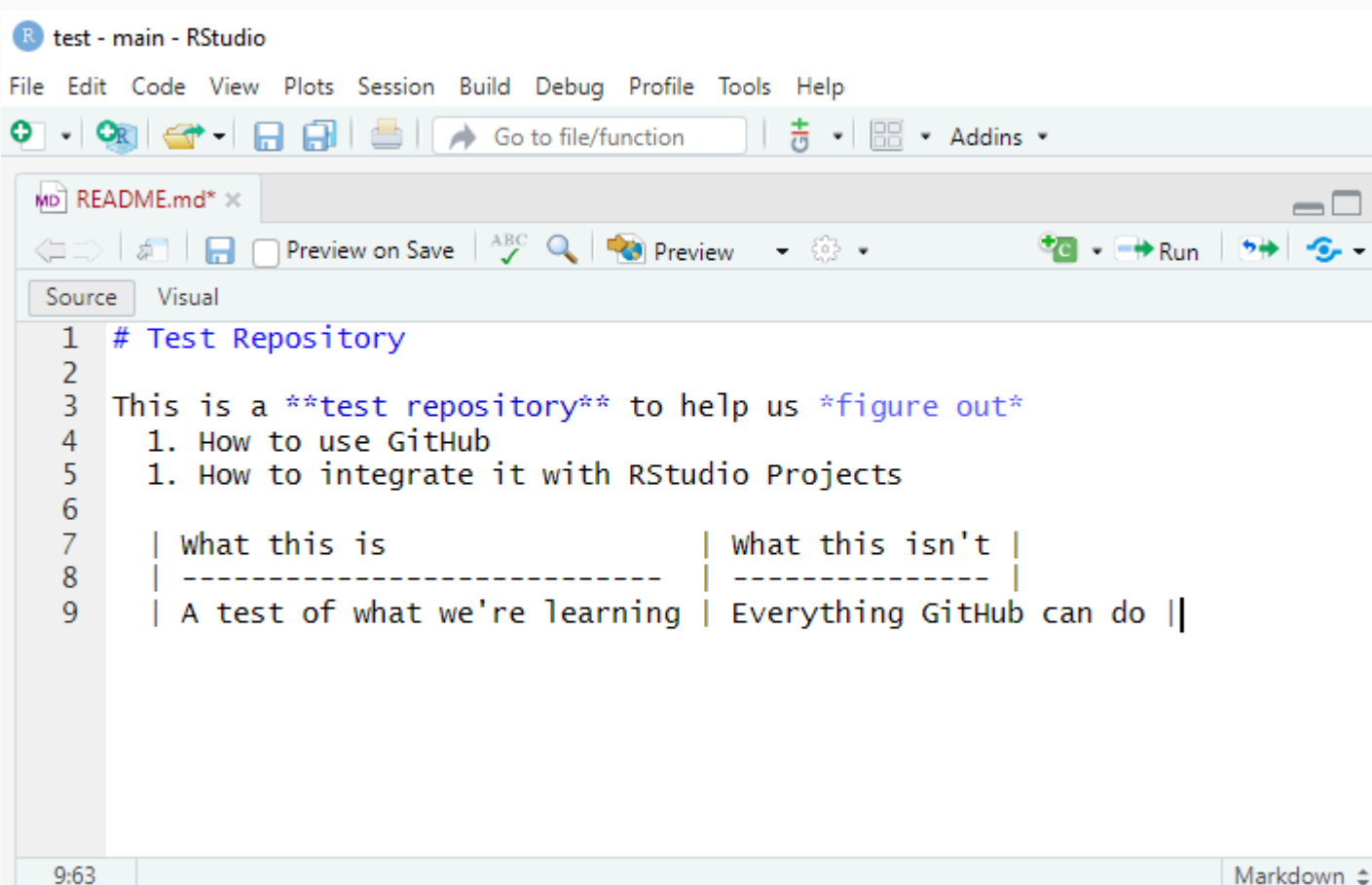
- Add some text and save the README.
- Do you see any changes in the **Git** panel? Good. (Raise your hand if not.)

¹ They're important, but not for the purposes of this section.

Making Local Changes



Making Local Changes



The screenshot shows the RStudio interface with the 'test - main' project. The menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, and Help. The toolbar contains icons for file operations and a search bar. The editor window displays a file named 'README.md' with the following content:

```
1 # Test Repository
2
3 This is a test repository to help us figure out
4   1. How to use GitHub
5   1. How to integrate it with RStudio Projects
6
7   | What this is | What this isn't |
8   | ----- | ----- |
9   | A test of what we're learning | Everything GitHub can do ||
```

The status bar at the bottom shows the time 9:63 and the file type Markdown.

Main Git operations

Now that you've **cloned** your first repo and made some local changes, it's time to learn the **four main Git operations**.

1. **Stage** (or "add")

- Tell Git that you want to add changes to the repo history (file edits, additions, deletions, etc.)

2. **Commit**

- Tell Git that, yes, you are sure these changes should be part of the repo history.

3. **Pull**

- Get any new changes made on the GitHub repo (i.e. the upstream remote), either by your collaborators or you on another machine.

4. **Push**

- Push any (committed) local changes to the GitHub repo

Main Git operations

1. **Stage** (or "add")

- Tell Git that you want to add changes to the repo history (file edits, additions, deletions, etc.)

2. **Commit**

- Tell Git that, yes, you are sure these changes should be part of the repo history.

3. **Pull**

- Get any new changes made on the GitHub repo (i.e. the upstream remote), either by your collaborators or you on another machine.

4. **Push**

- Push any (committed) local changes to the GitHub repo

For the moment, it will be useful to group the first two operations and last two operations together. (They are often combined in practice too, although you'll soon get a sense of when and why they should be split up.)

Stage and Commit

The screenshot shows the RStudio interface with a project named "test - main". The main editor displays a README.md file with the following content:

```
1 # Test Repository
2
3 This is a test repository to help us figure out
4 1. How to use GitHub
5 1. How to integrate it with RStudio Projects
6
7 | what this is | what this isn't |
8 | ----- | ----- |
9 | A test of what we're learning | Everything GitHub can do|
```

The right-hand pane shows the Git status. The "Staged" tab is active, displaying a list of files with their status icons:

Staged	Status	File
<input type="checkbox"/>	?	.gitignore
<input type="checkbox"/>	M	README.md
<input type="checkbox"/>	?	test.Rproj

A tooltip for the "Commit" button indicates "Commit pending changes (Ctrl+Alt+M)". Below the Git pane, the "Files" tab shows the project structure:

Name	Size
..	
.gitignore	44 B

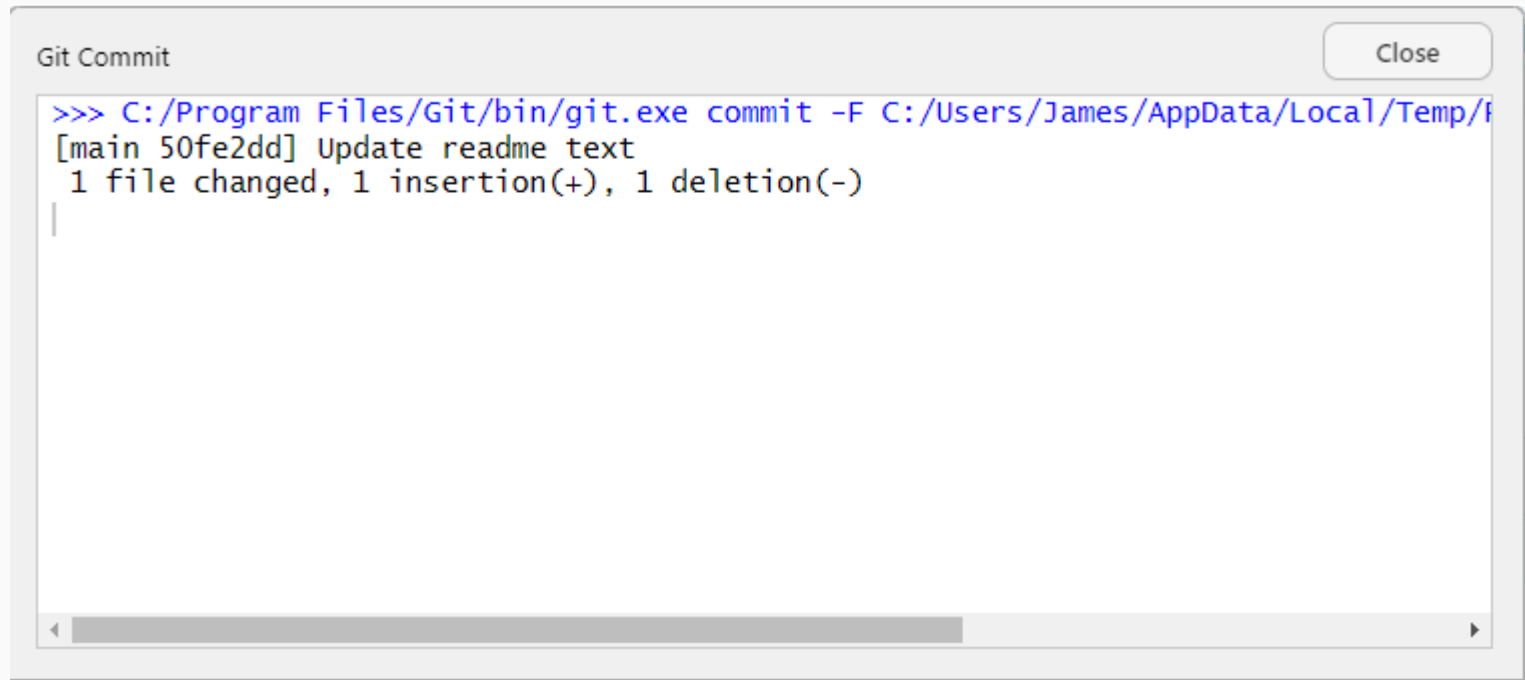
Stage and Commit

The screenshot shows the RStudio 'Review Changes' window. At the top, there are tabs for 'Changes' and 'History', and a dropdown for the current branch, 'main'. To the right of these are buttons for 'Stage' (checked), 'Revert', and 'Ignore'. Further right are 'Pull' and 'Push' buttons. Below the tabs, a message states: 'Your branch is ahead of 'origin/main' by 1 commit.'

The main area is divided into two panes. The left pane is a table with columns 'Staged', 'Status', and 'Path'. It lists three files: '.gitignore' (unstaged, yellow question mark), 'README.md' (staged, blue 'M'), and 'test.Rproj' (unstaged, yellow question mark). The right pane is for the 'Commit message', showing 'Update readme text' with a character count of 18. Below the message input is an 'Amend previous commit' checkbox and a 'Commit' button.

At the bottom, there is a 'Show' section with radio buttons for 'Staged' (selected) and 'Unstaged', a 'Context' dropdown set to '5 lines', an 'Ignore Whitespace' checkbox, and an 'Unstage All' button. Below this is a diff view showing changes to 'README.md'. The diff shows a deletion of a line (line 3) and an addition of a new line (line 3). The added line is highlighted in green and contains the text: 'This is a **test repository** to help us *figure out*'. The diff view also shows a list of topics: '1. How to use GitHub' and '1. How to integrate it with RStudio Projects', followed by a comparison of 'what this is' and 'what this isn't'.

Stage and Commit



A screenshot of a 'Git Commit' dialog box. The window has a title bar 'Git Commit' and a 'Close' button in the top right corner. The main area contains a text field with the following text:
>>> C:/Program Files/Git/bin/git.exe commit -F C:/Users/James/AppData/Local/Temp/f
[main 50fe2dd] Update readme text
1 file changed, 1 insertion(+), 1 deletion(-)
Below the text field is a horizontal scrollbar.

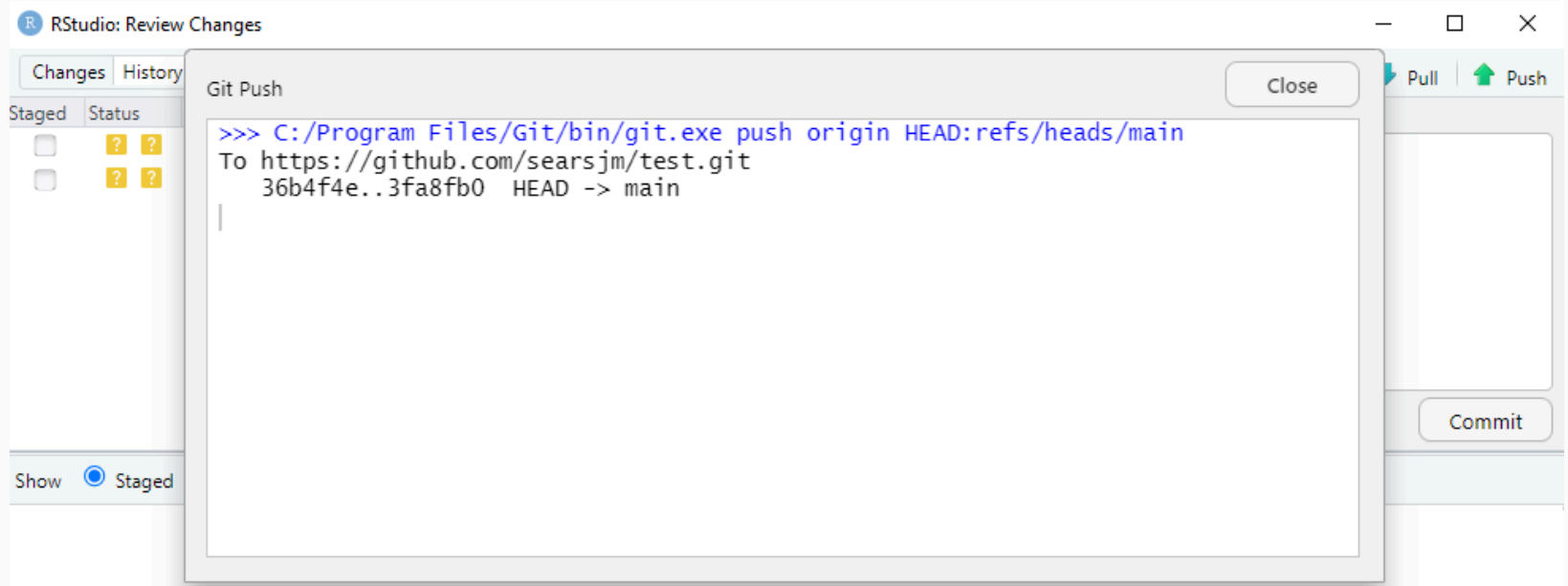
```
>>> C:/Program Files/Git/bin/git.exe commit -F C:/Users/James/AppData/Local/Temp/f
[main 50fe2dd] Update readme text
1 file changed, 1 insertion(+), 1 deletion(-)
```

Pull

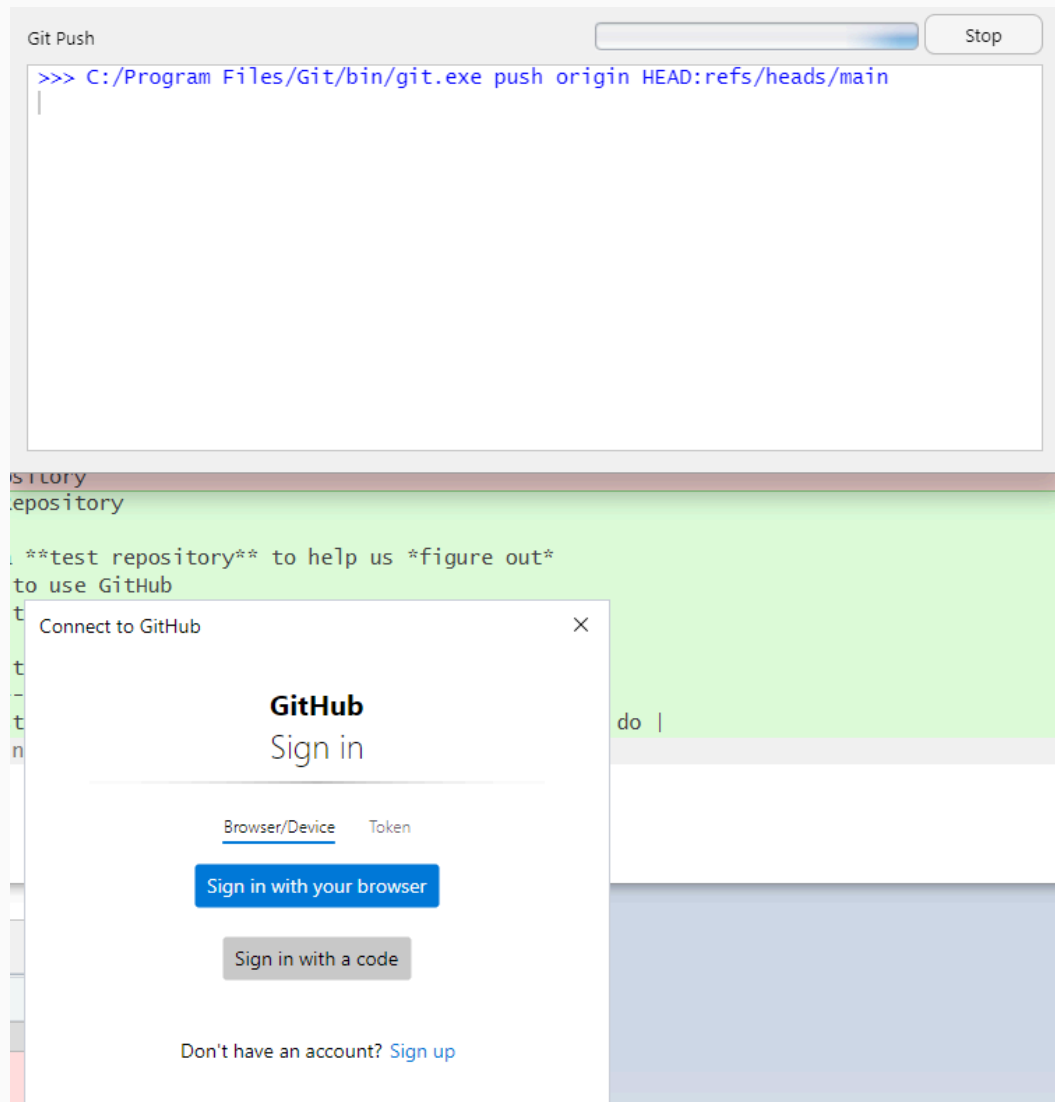
The screenshot shows the RStudio interface with a 'Git Pull' dialog box open. The dialog box contains the command `>>> C:/Program Files/Git/bin/git.exe pull` and the message 'Already up to date.' with a 'Close' button. In the background, the RStudio 'Changes' pane shows a file named 'Test Repository' with a commit message '# test'. The file editor shows the content of 'Test Repository' with a green background. The content is as follows:

```
1 # Test Repository
2
3 This is a **test repository** to help us *figure out*
4 1. How to use GitHub
5 1. How to integrate it with RStudio Projects
6
7 | What this is | What this isn't |
8 | ----- | ----- |
9 | A test of what we're learning | Everything GitHub can do |
No newline at end of file
```

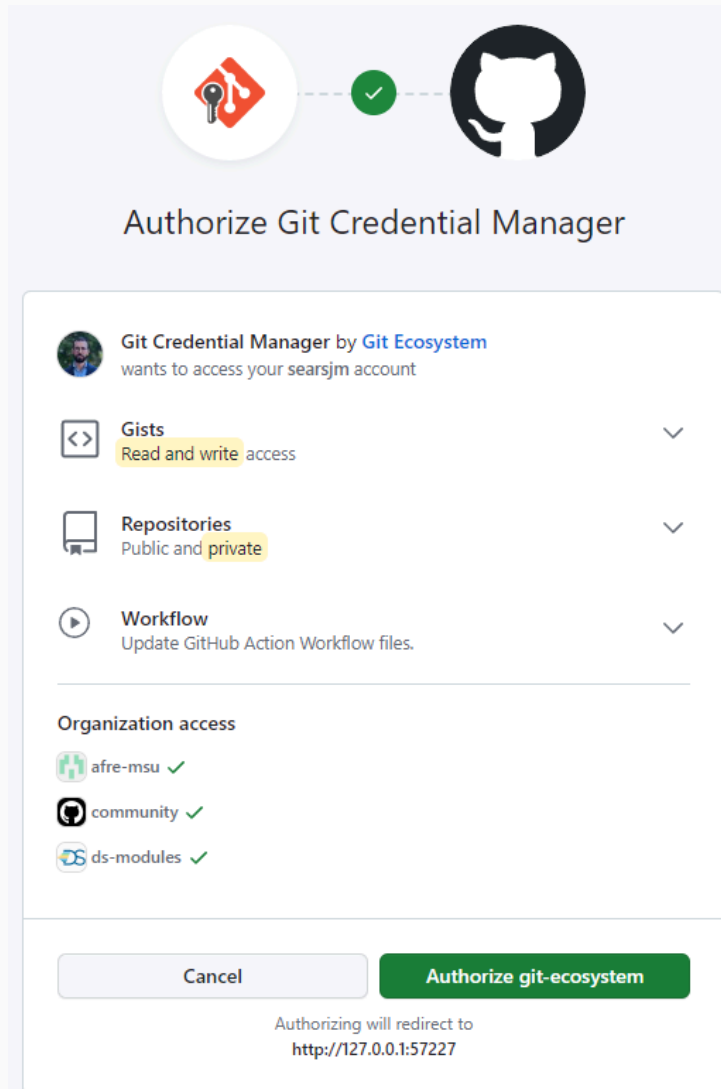
Push



Sign RStudio into Github



Sign RStudio into Github



Changes Now Visible on Github

searsjm / test

Q Type to search

>

+

<> Code

Issues

Pull requests

Actions

Projects

Wiki

Security

Insights

Settings

test

Public

Pin

Unwatch 1

Fork 0

Star 0

main

1 Branch

0 Tags

Go to file

t

+

<> Code

searsjm Update Readme

52525f2 · now

3 Commits

LICENSE

Initial commit

42 minutes ago

README.md

Update Readme

now

README

MIT license

Test Repository

This is a **test repository** to help us *figure out*

1. How to use GitHub

2. How to integrate it with RStudio Projects

What this is	What this isn't
A test of what we're learning	Everything GitHub can do

About

Test Repository

Readme

MIT license

Activity

0 stars

1 watching

0 forks

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

Changes Now Visible Locally Too

OneDrive - Michigan State University > Github > test

Name	Status	Date modified	Type	Size
.git	✓	1/15/2024 4:35 PM	File folder	
.Rproj.user	✓	1/15/2024 3:47 PM	File folder	
.gitignore	✓	1/15/2024 3:47 PM	Text Document	1 KB
.Rhistory	✓	1/15/2024 4:35 PM	RHISTORY File	0 KB
LICENSE	✓	1/15/2024 3:47 PM	File	2 KB
README.md	✓	1/15/2024 4:32 PM	MD File	1 KB
test.Rproj	✓	1/15/2024 4:32 PM	R Project	1 KB

README.md — Github\test — F:\OneDrive - Michigan State University\Github\test — Atom

File Edit View Selection Find Packages Help

Project



Your project is currently empty

Add folders

Reopen a project

Sunsetting Atom

README.md — Course...

README2.md

README.md — Github...

```
1 # Test Repository
2
3 This is a **test repository** to help us *figure out*
4   1. How to use GitHub
5   1. How to integrate it with RStudio Projects
6
7 | What this is | What this isn't |
8 | ----- | ----- |
9 | A test of what we're learning | Everything GitHub can do |
```


Recap

Here's a step-by-step summary of what we just did.

- Made same changes to a file and saved them locally.
- **Staged** these local changes.
- **Committed** these local changes to our Git history with a helpful message.
- **Pulled** from the GitHub repo just in case anyone else made changes too (not expected here, but good practice).
- **Pushed** our changes to the GitHub repo.

Aside: Always pull from the upstream repo *before* you push any changes. Seriously, do this even on solo projects; making it a habit will save you headaches down the road.

Recap

Here's a step-by-step summary of what we just did.

- Made same changes to a file and saved them locally.
- **Staged** these local changes.
- **Committed** these local changes to our Git history with a helpful message.
- **Pulled** from the GitHub repo just in case anyone else made changes too (not expected here, but good practice).
- **Pushed** our changes to the GitHub repo.

PS — You were likely challenged for your GitHub credentials at some point. Learn how to cache these [here](#).

Why this Workflow: GitHub

Creating the repo on GitHub first means that it will **always be "upstream"** of your (and any other) local copies.

- In effect, this allows GitHub to act as the **central node** in the distributed VC network.
- Especially valuable when you are collaborating on a project with others — more on this later — but also has advantages when you are working alone.
- If you would like to move an existing project to GitHub, my advice is still to create an empty repo there first, clone it locally, and then copy all your files across.

Why this Workflow: RStudio

RStudio Projects are great.

- They interact seamlessly with Git(Hub), as we've just seen.
- They also solve absolute vs. relative path problems, since the .Rproj file acts as an anchor point for all other files in the repo.¹

¹ Calling files from their full `YourComputer/YourName/Documents/Special-Subfolder/etc` paths in your scripts is the enemy of reproducibility!

GitHub Desktop

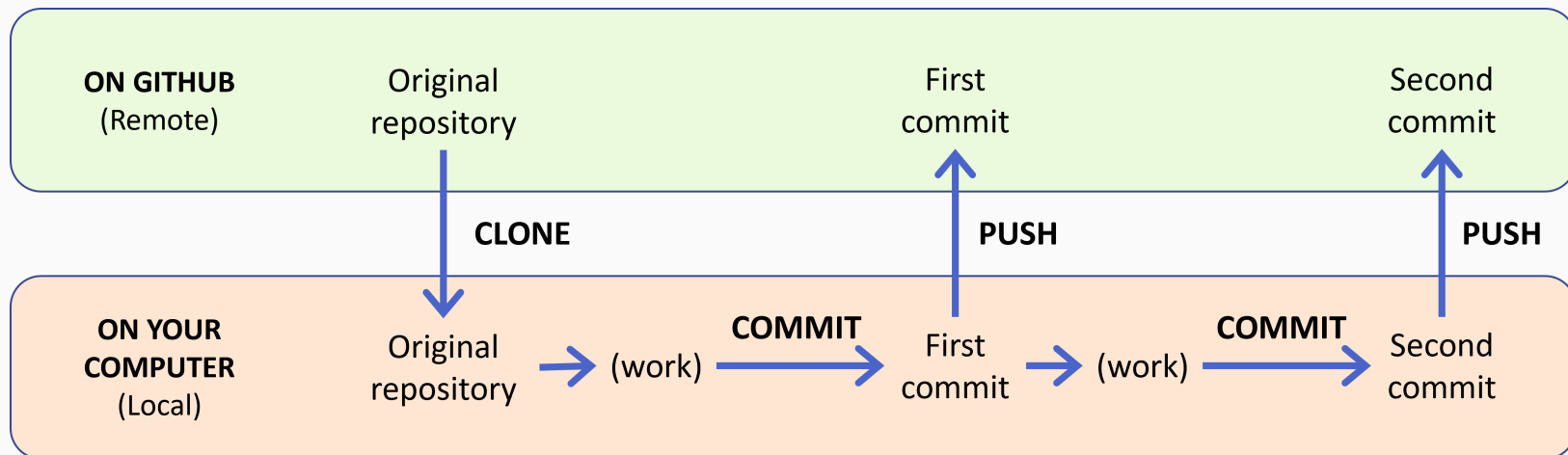
Version Control with GitHub Desktop

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**.

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

Github Desktop Workflow

With GitHub Desktop, we can maintain a similar workflow



Github Desktop Workflow

With GitHub Desktop, we can maintain a similar workflow

1. Create a new repository on GitHub.com
2. **Clone** the repository to your local machine
3. Do some work (i.e. edit the repository)
4. **Commit** changes to the repository
5. **Push** your commit to GitHub

Clone Repository

The screenshot shows the GitHub interface for a repository named 'test' by user 'searsjm'. The repository is public and has 1 branch and 0 tags. The 'Code' button is highlighted, and its dropdown menu is open, showing options to clone the repository using HTTPS, SSH, or GitHub CLI. The HTTPS URL is `https://github.com/searsjm/test.git`. Other options include 'Open with GitHub Desktop' and 'Download ZIP'. The repository's README is visible in the background, titled 'Test Repository'.

searsjm / test

Type to search

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

test Public

Pin Unwatch 1 Fork 0 Star 0

main 1 Branch 0 Tags

searsjm Update readme text

LICENSE Initial commit

README.md Update readme

README MIT license

Test Repository

This is a test repository to help us figure out

Go to file + <> Code

Local Codespaces

Clone

HTTPS SSH GitHub CLI

`https://github.com/searsjm/test.git`

Use Git or checkout with SVN using the web URL.

Open with GitHub Desktop

Download ZIP

About

Test Repository

Readme

MIT license

Activity

0 stars

1 watching

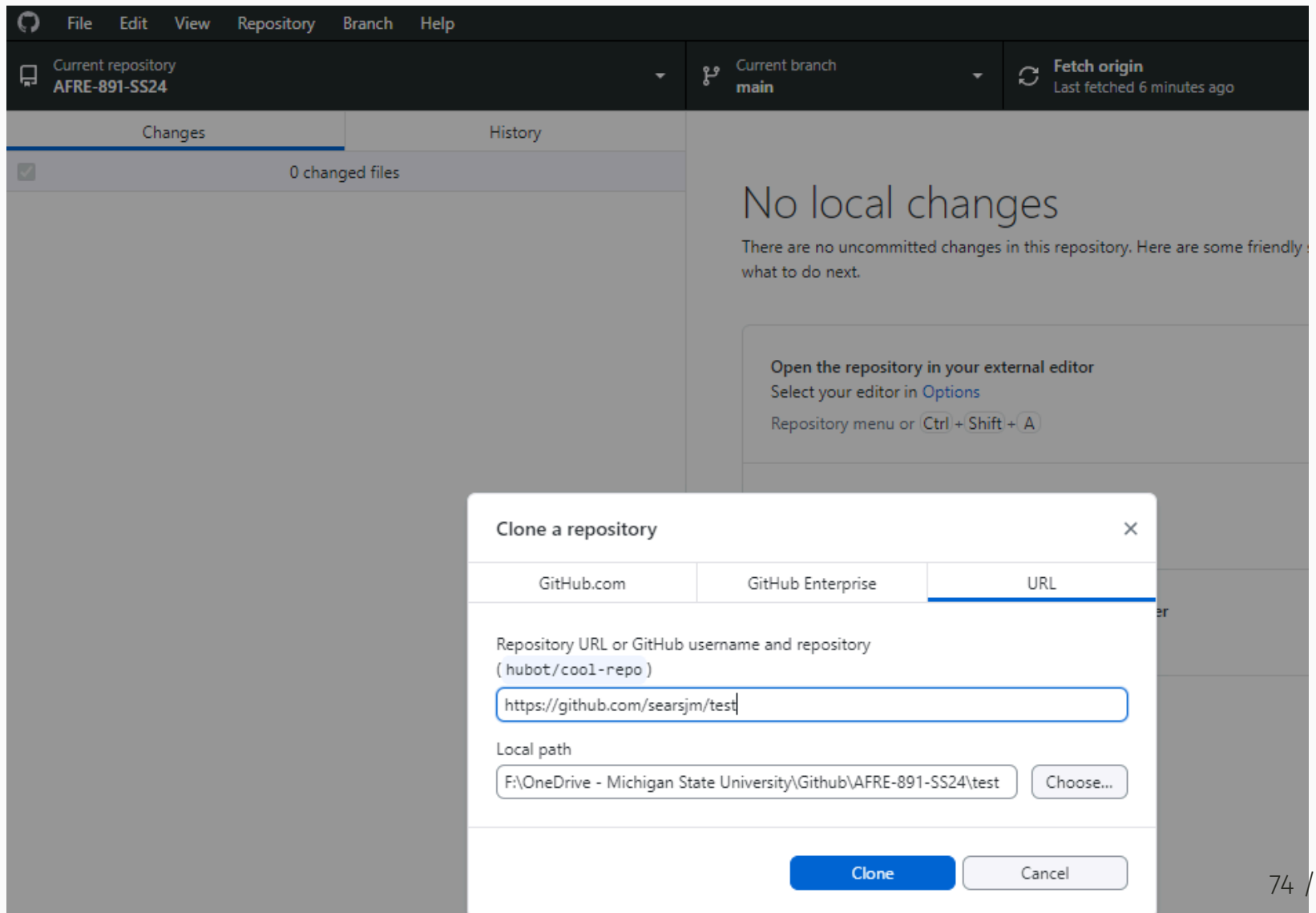
0 forks

Releases

No releases published

[Create a new release](#)

Clone Repository



Clone Repository

FileEditViewRepositoryBranchHelp

Current repository
test

Current branch
main

Fetch origin
Never fetched

ChangesHistory

No branches to compare

Initial commit

James Sears • 3 minutes ago

Initial commit

James Sears 36b4f4e ± 2 changed files +23 -0

LICENSE

README.md

@@ -0,0 +1,2 @@

1 +# test

2 +Test Repository

Fetch Origin + Pull to Stay Current

The screenshot displays the RStudio Git interface. At the top, the menu bar includes File, Edit, View, Repository, Branch, and Help. Below the menu, the status bar shows the current repository is 'test', the current branch is 'main', and a 'Pull origin' button with a downward arrow and the text 'Last fetched just now'. The main workspace is divided into three panes. The left pane, titled 'Changes 1', shows a list of changes: a checked box next to '1 changed file' and a checked box next to 'README.md'. The middle pane, titled 'History', is currently empty. The right pane, titled 'README.md', shows the content of the file with line numbers 1 through 9. The content is as follows:

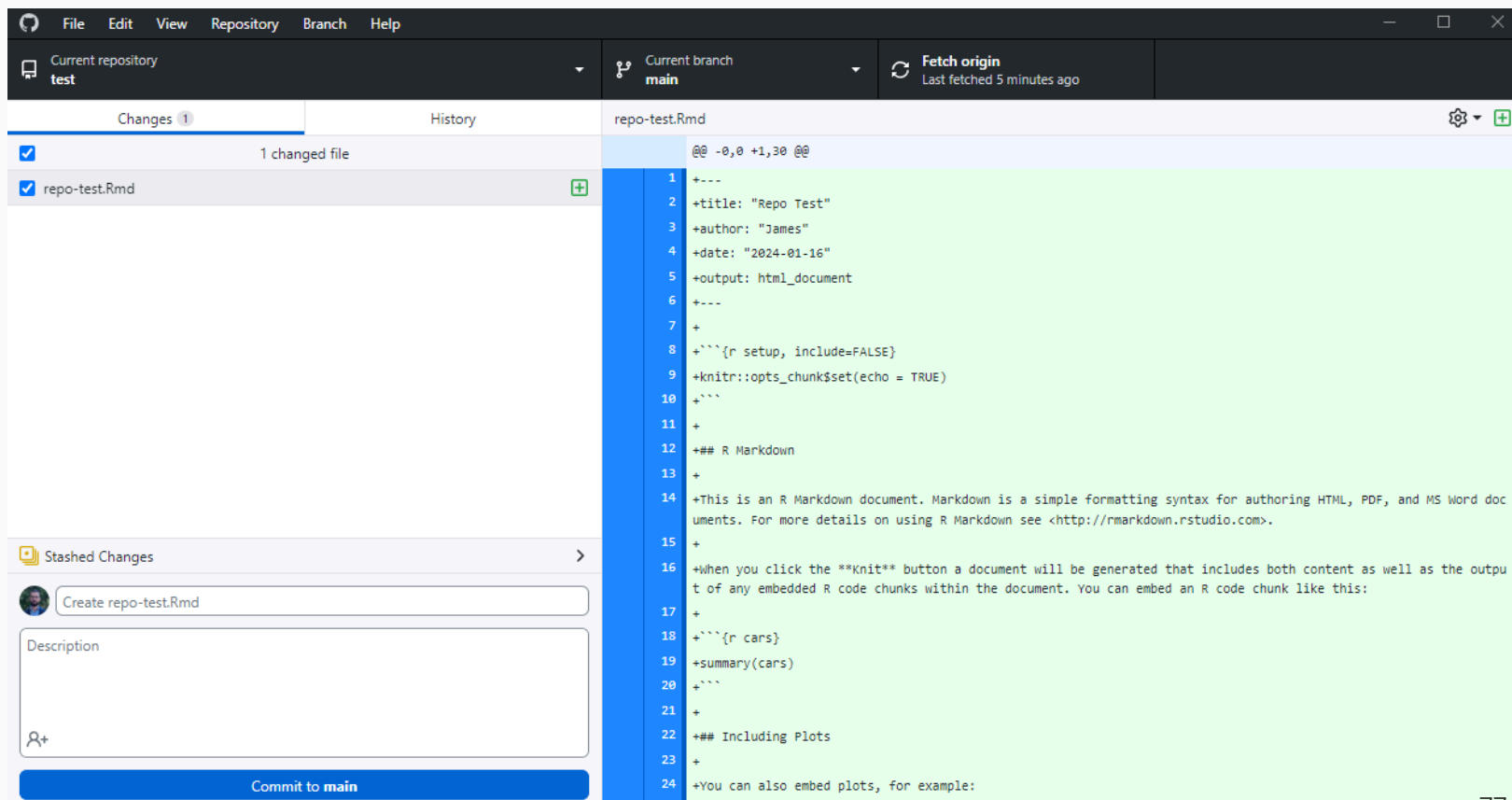
```
@@ -1,2 +1,9 @@
1  -# test
2  -Test Repository
3  +# Test Repository
4  +
5  +This is a **test repository** to help us *figure out*
6  + 1. How to use GitHub
7  + 1. How to integrate it with RStudio Projects
8  +
9  + | What this is | What this isn't |
10 + | ----- | ----- |
11 + | A test of what we're learning | Everything GitHub can do |
```

At the bottom of the interface, there is a section for committing changes. It includes a profile picture icon, a text input field with the placeholder 'Update README.md', a larger text input field with the placeholder 'Description', and a small icon of a person with a plus sign. Below these fields is a blue button labeled 'Commit to main'.

Make Local Changes to Repo

Save a new .Rmd file named `repo-test` into the "test" folder

- GitHub Desktop automatically stages the changes



Commit Changes

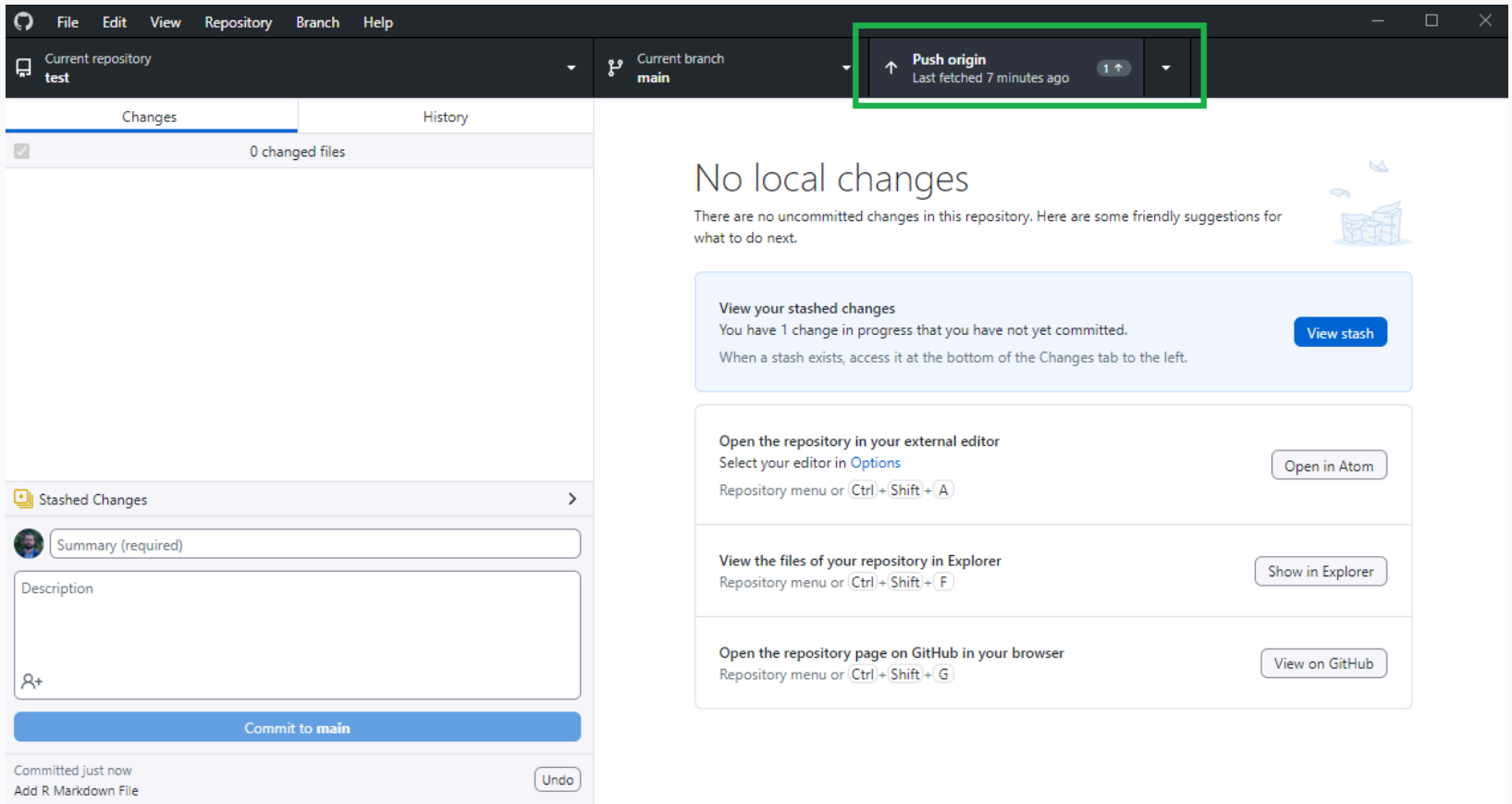
Add commit title and description, then commit to GitHub

The screenshot shows the RStudio interface with the 'Commit to main' dialog open. The dialog has a 'Changes' tab showing '1 changed file' and 'repo-test.Rmd'. Below this is a 'Stashed Changes' section with a text input field containing 'Added an R Markdown test file' and a 'Commit to main' button.



The 'repo-test.Rmd' file content is displayed in the editor:


```
@@ -0,0 +1,30 @@
1 +---
2 +title: "Repo Test"
3 +author: "James"
4 +date: "2024-01-16"
5 +output: html_document
6 +---
7 +
8 +```{r setup, include=FALSE}
9 +knitr::opts_chunk$set(echo = TRUE)
10 +```
11 +
12 +## R Markdown
13 +
14 +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>.
15 +
16 +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:
17 +
18 +```{r cars}
19 +summary(cars)
20 +```
21 +
22 +## Including Plots
23 +
24 +You can also embed plots, for example:
```

Push Changes to GitHub Desktop




Repo now Matches Local Version


 searsjm / test


Type  to search

[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)


 **test** Public Pin Unwatch 1




[main](#) [1 Branch](#) [0 Tags](#) [+](#) [Code](#)

 searsjm

Add R Markdown File 

b41ce7b · 1 minute ago

 6 Commits

 LICENSE	Initial commit	4 hours ago
 README.md	Update readme text	3 hours ago
 repo-test.Rmd	Add R Markdown File	1 minute ago

Other Tips and Productivity Tools

Productivity Miscellanea

What follows are miscellaneous things that I find **improve my productivity**

- **Synced Cloud Storage** (SpartanDrive or Dropbox/Box)
- **Overleaf** for LaTeX collaboration
- **Connected Papers** for literature networks

Synced Cloud Storage and OneDrive

Synced Cloud Storage is hugely beneficial if you work across **multiple computers** or **with many collaborators**.

- Make sure each computer has the most up-to-date version of all your files
- Renders flash drives almost entirely obsolete!

Synced Cloud Storage and OneDrive

One easy way to do this: **SpartanDrive/OneDrive**

All faculty + staff get **5 TB of free storage** on **SpartanDrive (MSU's version of OneDrive)**.

Pros

- Free
- Syncable desktop apps
- Part of the MSU Office365 ecosystem

Cons

- Part of the MSU Office365 ecosystem
- Limited storage (5TB, 250gb max filesize)
- Sometimes finicky

Synced Cloud Storage

Alternatives to SpartanDrive:

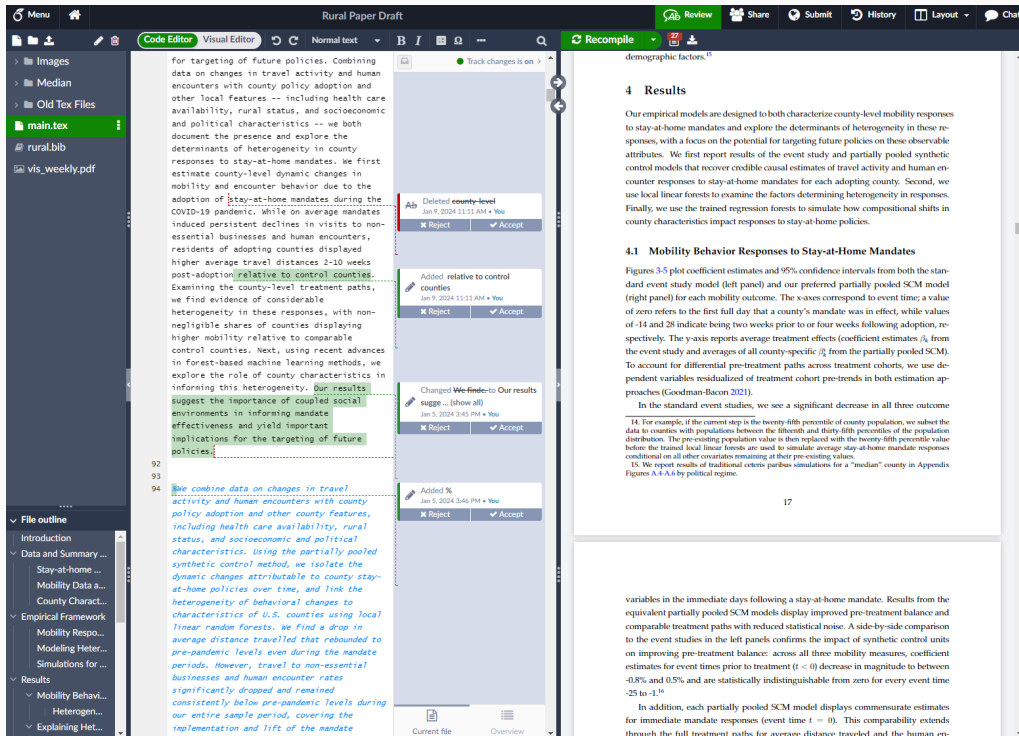
- **Dropbox**: 2GB free (\$10/mo for 2TB)
- **Box**: 10GB free (\$10/mo for 100GB)

Ultimate choice of platform may depend on coauthors + your current choice, but it's a good idea to **download the desktop app** to keep your files synced + backed up!

- Also gives basic version control

Overleaf

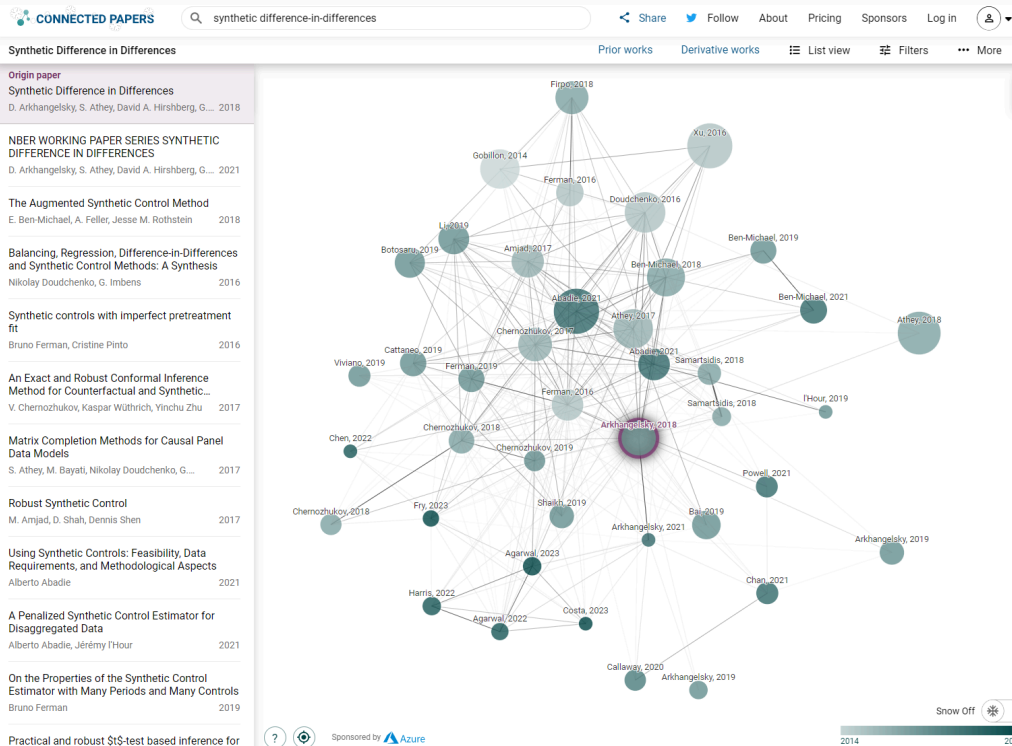
If you typeset using LaTeX, **Overleaf** streamlines access and collaboration



Free Version: remotely host all files, access for you + 1 collaborator

Paid (\$7.40/mo): track changes + full document history, Git(hub) + Dropbox integration, up to 6 collaborators

Connected Papers



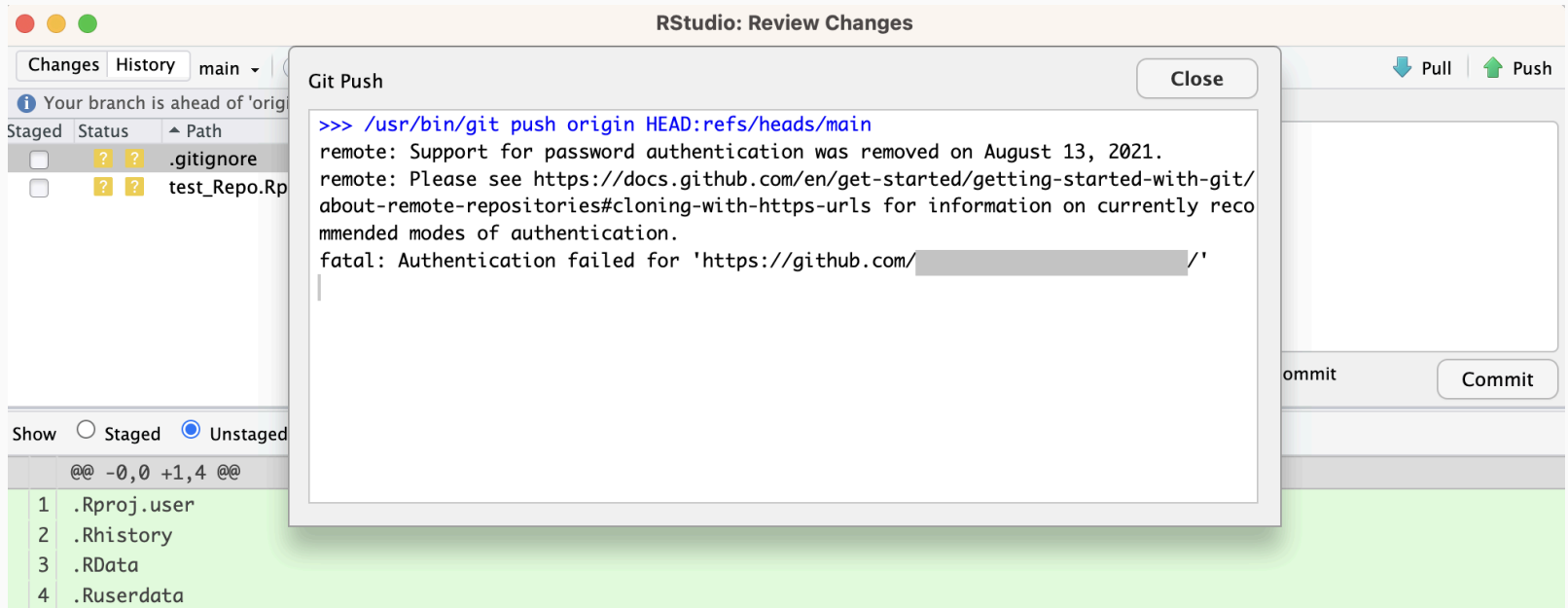
- Graphical representation of paper networks
 - Visualize literature as directed graph
 - 5 free per month (unlimited for \$6/mo)

Your Productivity Tips + Tricks?

Troubleshooting Git Credential Issues in RStudio

Troubleshooting Git Credential Issues

Do you get a password authentication error when trying to push to GitHub from RStudio?



Troubleshooting Steps

To begin, install the `usethis` package:

```
pacman:p_load(usethis)
```

```
# or:
```

```
# install.packages("usethis")
```

```
# followed by
```

```
# library(usethis)
```

Troubleshooting Steps

Next, run the following:

```
library("usethis")  
git_sitrep()
```

View the output: do you see any Xs or lines saying "lacks recommended scopes", "error", or "can't retrieve"?

Create a Personal Access Token (PAT)

To fix these errors, we'll create a **Personal Access Token (PAT)** on GitHub.

The below code will open a browser window - follow the steps to create a PAT

```
usethis::create_github_token()
```

New personal access token (classic)

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Note

DESCRIBE THE TOKEN'S USE CASE

What's this token for?

Expiration *

No expiration  The token will never expire!

GitHub strongly recommends that you set an expiration date for your token to help keep your information secure.

[Learn more](#)

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes](#).

- | | |
|---|--------------------------------------|
| <input checked="" type="checkbox"/> repo | Full control of private repositories |
| <input checked="" type="checkbox"/> repo:status | Access commit status |

Add a PAT to RStudio

Back in RStudio, run

```
gitcreds::gitcreds_set()
```

Follow the prompts to add/replace existing credentials, pasting in the PAT.

Check PAT

Once added, run `git_sitrep` one last time to verify.

If all looks good, try pushing again!

```
> git_sitrep()
Git config (global)
• Name: 'James Sears'
• Email: 'james.sears.m@gmail.com'
• Global (user-level) gitignore file: <unset>
• Vaccinated: FALSE
i See `?git_vaccinate` to learn more
• Default Git protocol: 'https'
• Default initial branch name: 'master'
GitHub
• Default GitHub host: 'https://github.com'
• Personal access token for 'https://github.com': '<discovered>'
• GitHub user: 'searsjm'
• Token scopes: 'gist, repo, user, workflow'
• Email(s): 'james.sears.m@gmail.com (primary)', 'searsja1@msu.edu'
Git repo for current project
i No active usethis project
>
```

Table of Contents

1. [Prologue](#)
2. [R Markdown](#)
3. [Version Control](#)
4. [Git\(Hub\) + RStudio](#)
5. [GitHub Desktop](#)
6. [Other Tips and Productivity Tools](#)
7. [Not Covered: Troubleshooting Git Credential Issues in RStudio](#)