

Statistics II

Week 1

Content for Today

1. Introduction to `RMarkdown`
2. GitHub overview and setup
3. Assignment peer-review with `ghclass`
4. Assignment workflow overview

Labs

The optional drop-in lab sessions exist primarily to help with the coding application of the materials presented in the lecture.

Each session will begin with a brief overview of the week's content and a few minutes for questions, followed by coding exercises that will help you to prepare for the assignments and develop your coding skills.

Please attend the lab you're assigned to.

Slides and scripts will be available here:

<https://github.com/seramirezruiz/stats-ii-lab>

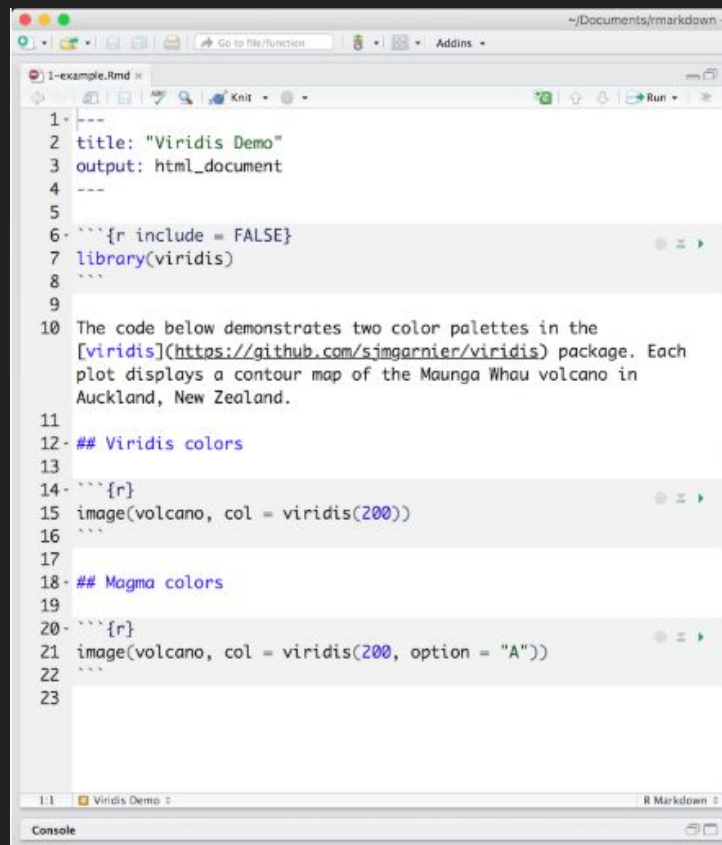
RMarkdown

Introduction to Rmarkdown

RMarkdown is an authoring framework for data science. A single RMarkdown file can be used to:

- Save and execute code
- Generate high quality reports that can be shared with an audience

We will use RMarkdown to submit our weekly assignments and review our peers' work.

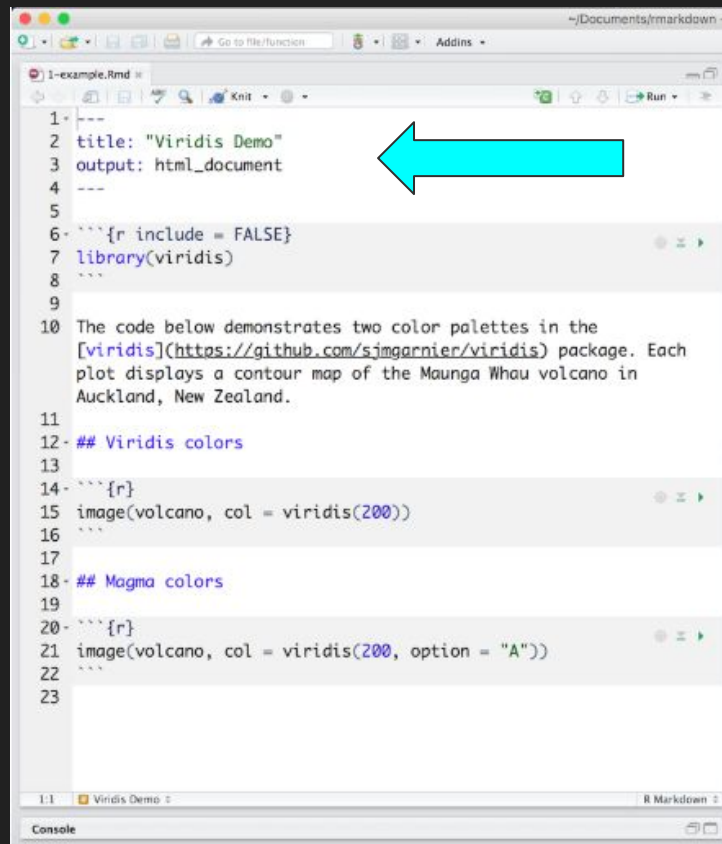


```
1 ---
2 title: "Viridis Demo"
3 output: html_document
4 ---
5
6 ```{r include = FALSE}
7 library(viridis)
8 ```
9
10 The code below demonstrates two color palettes in the
11 [viridis](https://github.com/sjmgarnier/viridis) package. Each
12 plot displays a contour map of the Maunga Whau volcano in
13 Auckland, New Zealand.
14
15 ## Viridis colors
16
17 ```{r}
18 image(volcano, col = viridis(200))
19 ```
20
21 ## Magma colors
22
23 ```{r}
24 image(volcano, col = viridis(200, option = "A"))
25 ```
26
27 
```

Contents in Rmarkdown

RMarkdown files support three types of content:

- **YAML headers** surrounded by `---`
Meta-data that guides the file build-up process.
(not used in assignments, since submissions are anonymous)



```
1 ---
2 title: "Viridis Demo"
3 output: html_document
4 ---
5
6 ```{r include = FALSE}
7 library(viridis)
8 ```
9
10 The code below demonstrates two color palettes in the
11 [viridis](https://github.com/sjmgarnier/viridis) package. Each
12 plot displays a contour map of the Maunga Whau volcano in
13 Auckland, New Zealand.
14
15 ## Viridis colors
16
17 ```{r}
18 image(volcano, col = viridis(200))
19 ```
20
21 ## Magma colors
22
23 ```{r}
24 image(volcano, col = viridis(200, option = "A"))
25 ```
```

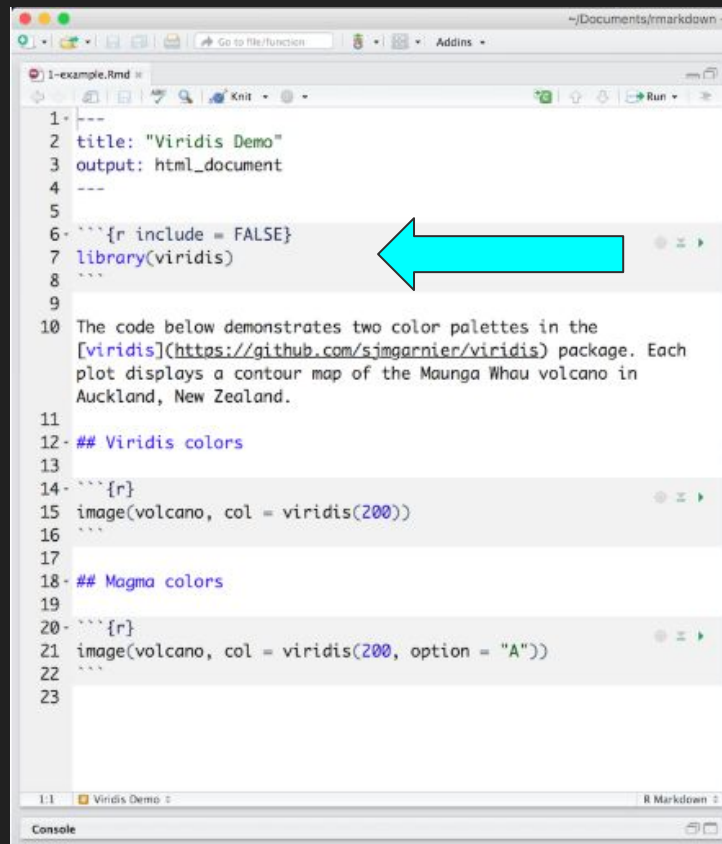
Contents in Rmarkdown

RMarkdown files support three types of content:

- R code chunks surrounded by ````\n\n`
Chunks take code as an input. It works in the same way your .R scripts did for Stats I.

start a chunk: ````{r}`

end a chunk: `````



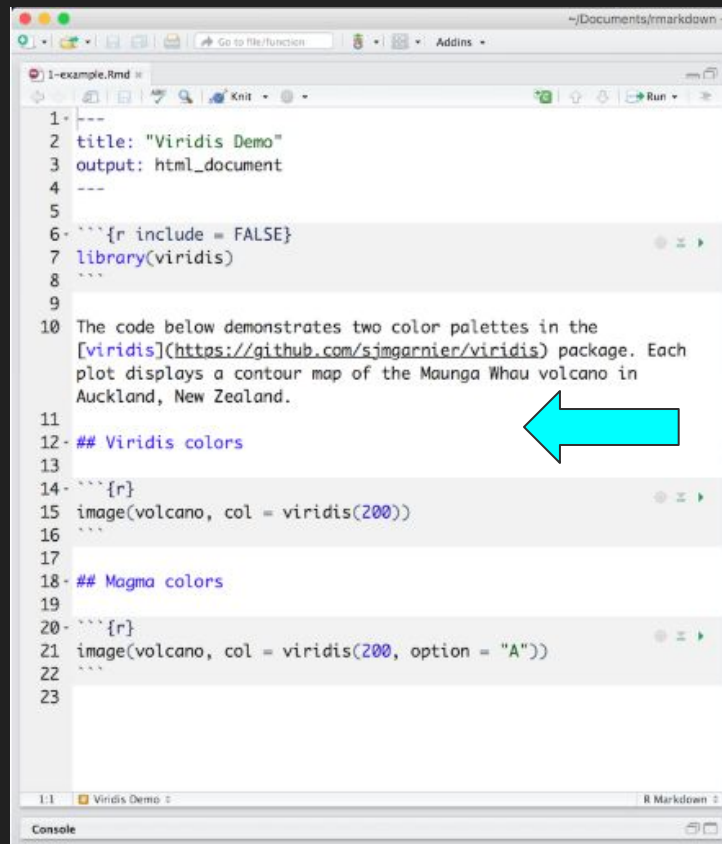
```
1 ---
2 title: "Viridis Demo"
3 output: html_document
4 ---
5
6 ```{r include = FALSE}
7 library(viridis)
8 ```
9
10 The code below demonstrates two color palettes in the
11 [viridis](https://github.com/sjmgarnier/viridis) package. Each
12 plot displays a contour map of the Maunga Whau volcano in
13 Auckland, New Zealand.
14
15 ## Viridis colors
16
17 ```{r}
18 image(volcano, col = viridis(200))
19 ```
20
21 ## Magma colors
22
23 ```{r}
24 image(volcano, col = viridis(200, option = "A"))
25 ```
26
27 
```

Contents in Rmarkdown

RMarkdown files support three types of content:

- Text mixed with simple text formatting
Takes text as input.

[RMarkdown Cheatsheet](#)



```
1 ---
2 title: "Viridis Demo"
3 output: html_document
4 ---
5
6 ```{r include = FALSE}
7 library(viridis)
8 ```
9
10 The code below demonstrates two color palettes in the
11 [viridis](https://github.com/sjmgarnier/viridis) package. Each
12 plot displays a contour map of the Maunga Whau volcano in
13 Auckland, New Zealand.
14
15 ## Viridis colors
16
17 ```{r}
18 image(volcano, col = viridis(200))
19 ```
20
21 ## Magma colors
22
23 ```{r}
24 image(volcano, col = viridis(200, option = "A"))
25 ```
26
```


GitHub

GitHub

GitHub is a version control platform that is industry-standard in any field where people code (though it can also technically be used with any kind of document).

It allows you to collaborate with others on the same piece of code, and keep records of the changes you've made.

Can be used through the terminal, directly in RStudio, or with the **desktop client** (which is the method we'll use in class).

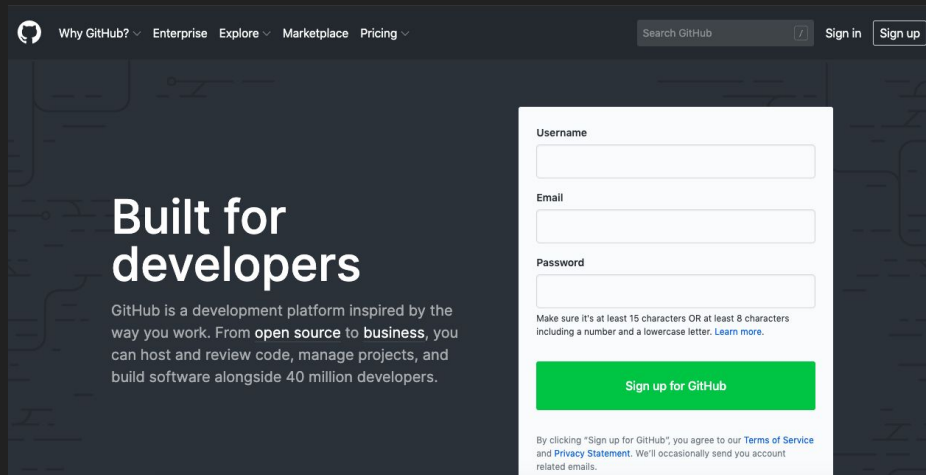
GitHub: Basic Concepts

- **Repository/repo:** Basically a project folder where your files are stored. Git will track changes to anything in this folder.
- **Clone:** make a copy of a repo so you can access it on your local machine.
- **Commit:** When you make changes, you commit them to “save” them.
- **Push:** After you make a commit, you have to push them to the repo. Otherwise they are just saved on your local machine.
- **Pull:** You can pull from the repo to access any changes made by other collaborators. This will give you the most recent version.
- **README:** File with information about the project.
- **.gitignore:** Files that you do not want Git to track (like data you don't want to upload)
- **Branch:** A copy of the repo you can make changes on without affecting others.
- **Merge:** If there are conflicts between your local copy and the copy in the repo, sometimes you may need to open the file and choose which version of the code to keep in order to merge the two versions.

1. Create your GitHub account

<https://github.com/>

1. Select the free version
2. Skip the extra questions
3. Verify your email address

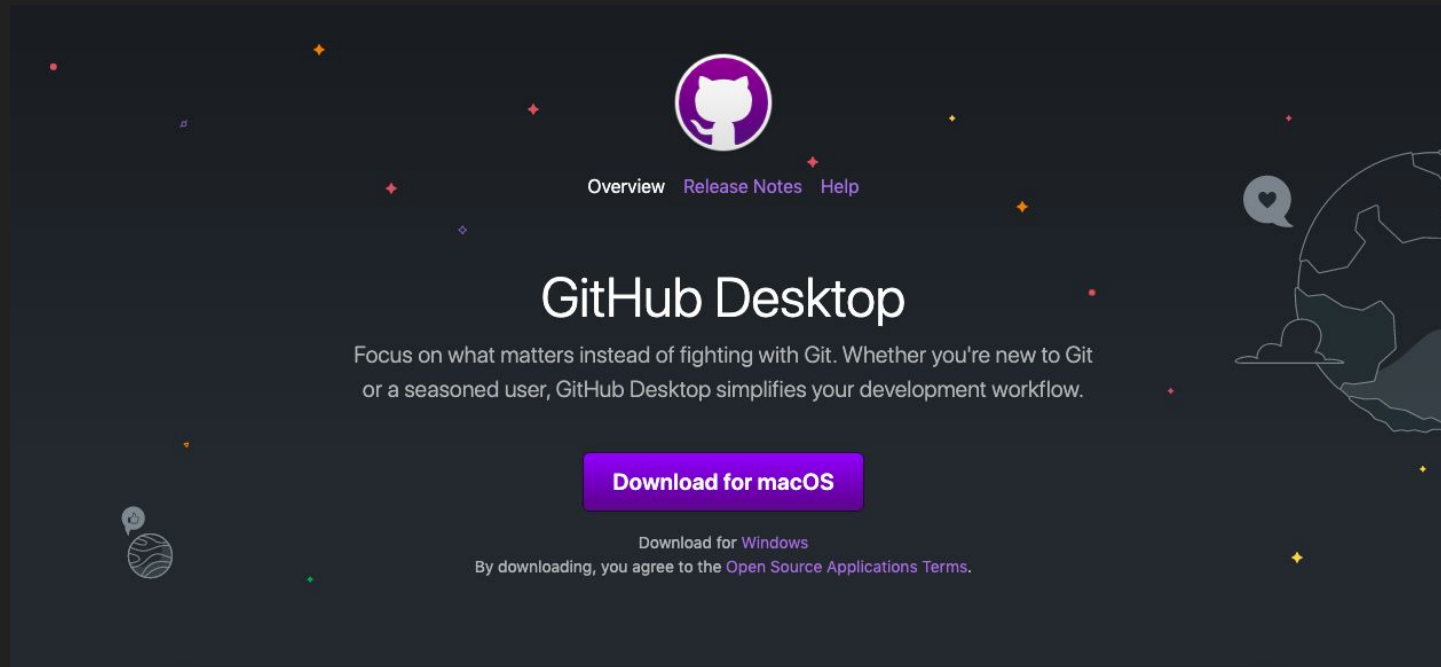


The screenshot shows the GitHub homepage with a sign-up overlay. The overlay contains the following elements:

- Navigation:** Links for 'Why GitHub?', 'Enterprise', 'Explore', 'Marketplace', and 'Pricing' at the top left. A search bar and 'Sign in' / 'Sign up' buttons at the top right.
- Main Content:** A large heading 'Built for developers' followed by a paragraph: 'GitHub is a development platform inspired by the way you work. From **open source** to **business**, you can host and review code, manage projects, and build software alongside 40 million developers.'
- Sign-up Form:** A white box on the right with three input fields: 'Username', 'Email', and 'Password'. Below the 'Password' field is a note: 'Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter. [Learn more.](#)'
- Buttons:** A large green button labeled 'Sign up for GitHub'.
- Footer:** A small line of text at the bottom: 'By clicking "Sign up for GitHub", you agree to our [Terms of Service](#) and [Privacy Statement](#). We'll occasionally send you account related emails.'

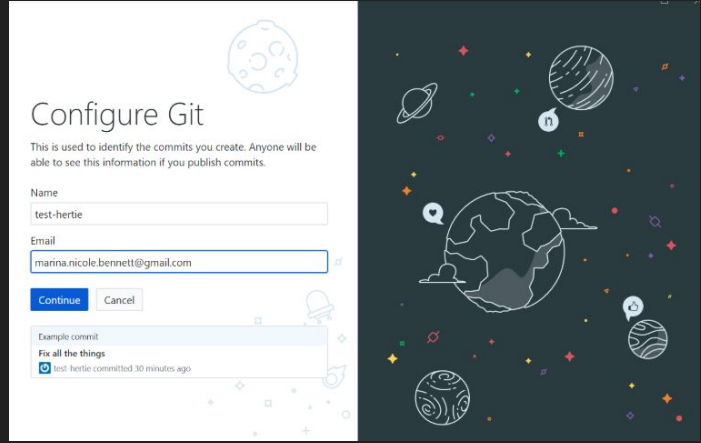
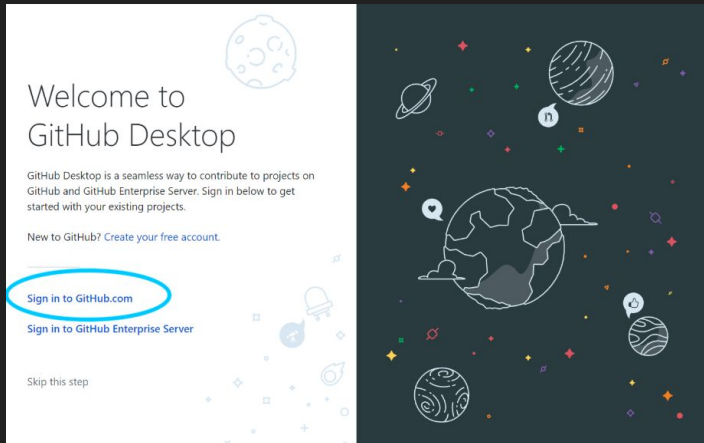
2. Download GitHub Desktop

<https://desktop.github.com/>



3. Login

Log in with your GitHub name and email



Peer Review with `ghclass`

Assignment Schedule

Your submission:

New assignment posted Tuesdays by 1pm

Submit solutions the following Sunday by 11:59pm

Reviews:

Assignment of two peers' submissions to review by the following Monday at 1pm

Submit your feedback the next day, Tuesday, by 11:50pm

Author feedback:

Receive feedback from peers on Wednesday (we think!)

Submit your feedback on the reviews of your assignment by Sunday 11:59pm

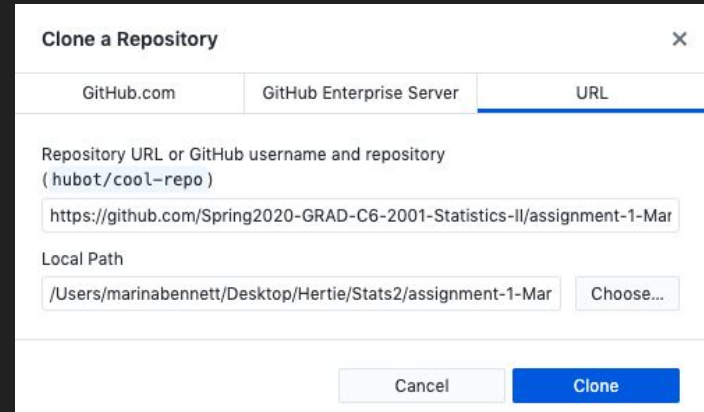
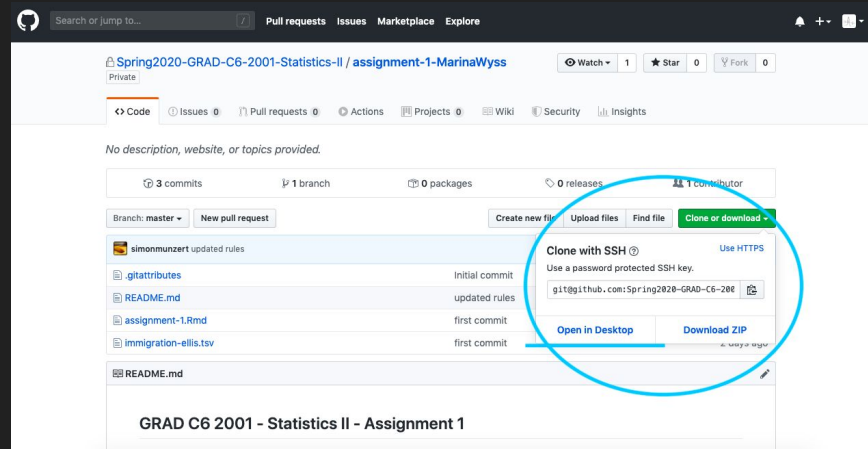
ghclass

Each week the professors will invite you to **two new** private repos each week - one for submissions, and one for reviews.

The submission one will be available on Tuesday, the review one will be available the following Monday.

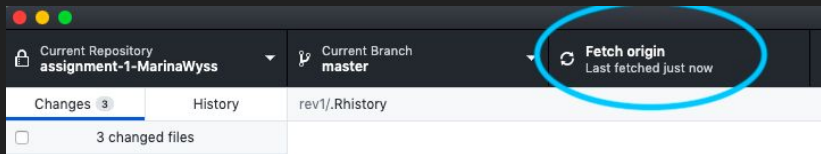
The first step is to **clone** repos so you can access them on your desktop and commit changes.

Save them to a folder on your desktop where you want your assignments.



ghclass - Your submission

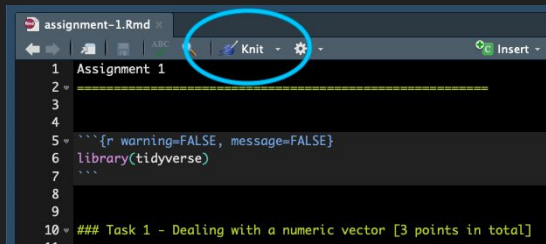
Open GitHub desktop, select your assignment repo, click **fetch**, then click **pull** (pull will only show up if there is something new in the repo, so you can skip this part if the button keeps saying 'fetch').



The assignment will now be in your assignment **desktop folder**.

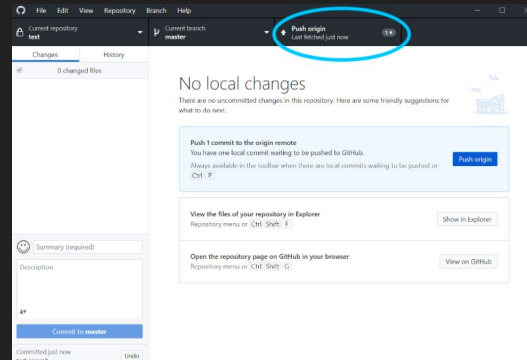
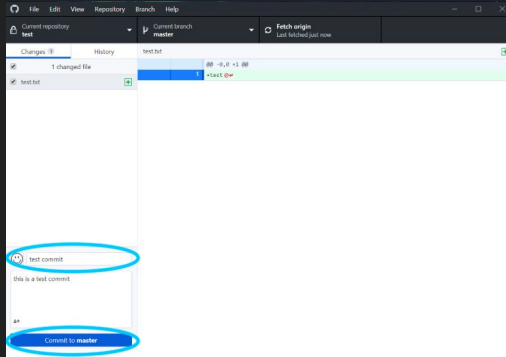
Fill out the .Rmd form, and when you're done, **knit** it to save it as a pretty html file.

Do not change the file names!



ghclass - Your submission

Open GitHub desktop again, select the files you want to commit (don't commit useless ones like .Rhistory, .DS_Store, etc.), **add a commit message**, and **click commit**, then click **push**. Your assignment has been submitted.



You can repeat the edit - commit - push process to make changes to your submission as many times as you'd like (before the deadline).

Check on GitHub.com to see that your changes made it and you pushed correctly.

ghclass - Reviews

The following Monday, you will have access to a review repo that will contain folders for two of your peers' assignments to review. **Repeat the cloning process** to save to your computer.

Grade their work and fill out your scores in the provided .Rmd document, including in the top header with the q* scores. Only include numbers (no brackets, no 4/4, etc.)

Knit the document (which is a .md file format), **commit**, and **push** your changes to GitHub.

ghclass - Rating Reviews

As a final step, you are asked to provide feedback on the quality of the peer-grading of your assignment.

Open GitHub desktop and navigate to the repo where **you created your assignment**, and click **fetch** and **pull**. Two new folders will now be in your original assignment desktop folder with your peers' reviews.

Complete the **author rating form**, knit the final document, and commit and push your changes to the .Rmd and .md files from within GitHub desktop.

Assignment Workflow: Rmarkdown, Github, and ghclass

- I. Weekly assignment submission*
- II. Peer-review process
- III. Rating reviewer's feedback

*Students are expected to submit both the .Rmd and a knitted .html versions of their weekly assignments

1. **Clone** the assignment repo for the week
 - assignment-x-username
2. Read instructions in README.md
3. Complete assignment-x.Rmd
4. Knit to html in RStudio (it will automatically update the .Rmd)
5. **Commit** changes in GitHub Desktop
6. **Push** your submission

Assignment Workflow: Rmarkdown, Github, and ghclass

- I. Weekly assignment submission
- II. Peer-review process*
- III. Rating reviewer's feedback

*You will have access to this repo on Monday following the submission deadline

**You should have one folder per peer with their .Rmd and .html submissions, plus the reviewer-feedback.Rmd file

1. **Clone** the review repo for the week*
assignment-x-review-username
2. Explore your peers' submissions**
3. Provide and save your feedback in the reviewer-feedback.Rmd file of each peer. Knit the .md file (note: different file type from the assignment)
4. **Commit** changes in GitHub Desktop
5. **Push** your submission

Assignment Workflow: Rmarkdown, Github, and ghclass

- I. Weekly assignment submission
- II. Peer-review process
- III. Rating reviewer's feedback

*You should have one folder per peer review

1. **Pull** the changes to your assignment-x-username folder from GitHub Desktop*
2. Explore the feedback offered by your peers
3. Rate the feedback and save the author-rating-form.Rmd file of each peer. Knit the .md file
4. **Commit** changes in GitHub Desktop
5. **Push** your submission

Helpful Hints

What to do when you get stuck on coding problems

First, don't panic. Then:

1. Check your code (missing parentheses, packages, stray commas, etc.)
2. Google the error message
3. Search on <https://stackoverflow.com/> or look on YouTube
4. Ask for help (from stackoverflow, friends, or your TA)

Further resources

For learning Git and `ghclass`:

- GitHub desktop: <https://tinyurl.com/wu9rjru>
- `ghclass` documentation: <https://tinyurl.com/wfrjol4>

R and RMarkdown:

- Reminder of the basics: <https://tinyurl.com/vkebh2f>
- RMarkdown: The definitive guide <https://tinyurl.com/y4tyfqmq>
- Data wrangling with `dplyr`: <https://tinyurl.com/vyrv596>
- `dplyr` video tutorial: <https://www.youtube.com/watch?v=jWjqLW-u3hc>