

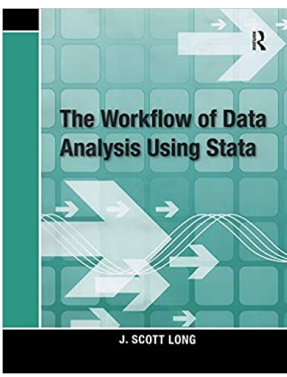
# Facilitating Reproducible Research

Wassim Tarraf, PhD  
Analyses Core Seminar  
MCUAAAR 5  
May 20<sup>th</sup>, 2019

# Plan

1. Discuss workflow
2. Integrate Open Science workflow
3. Challenges
4. Applied demonstration
  - a. git & GitHub
  - b. RStudio

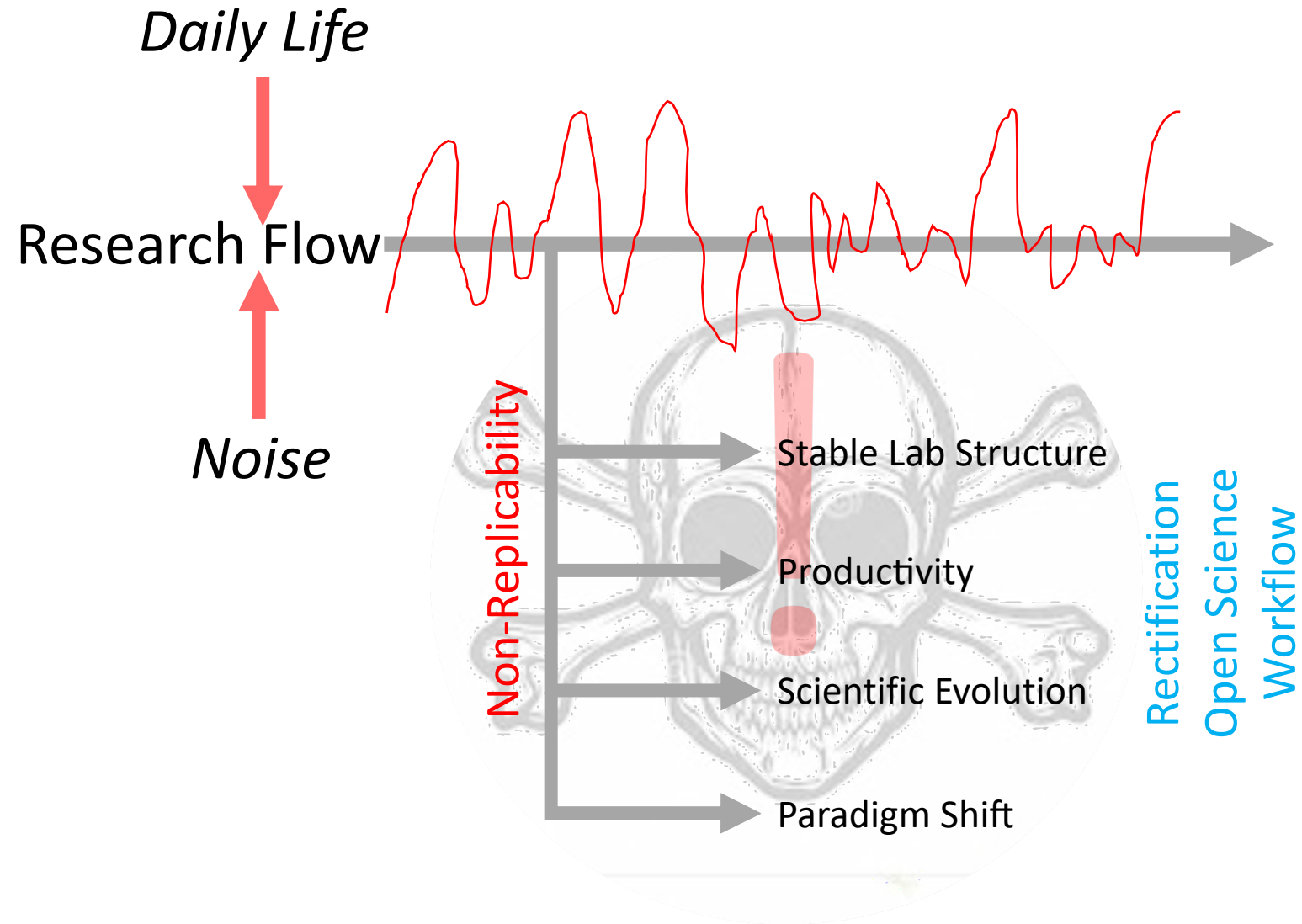
# Defining a research workflow (Scott Long)



System for:

- “Planning, organizing, and documenting” scientific process
- Establishing and fostering collaborations
- Managing and sharing data
- Analyzing data
- Disseminating findings
- Archiving process for replication

# Non-replicability



# What is the nature of your workflow?

- Non-systematic
  - Semi-systematic or ad-hoc
    - > reactionary, responsive to errors
  - Carefully planned
- 
- Can you improve workflow?
    - Initial time investment
    - Longer term improvement in efficiency and return on investment

# Decision points for open science workflows

1. Creating own workflow or using existing *Workflow Management Systems* ✓ - some recommendations
2. Choosing a data repository ✗
3. Deciding on a source code repository ✓ - some recommendations
4. Choosing a system to “Package, Access, and Execute Data and Code” ✓ - some recommendations
5. Choosing a document repository (free or fee for service) ✗
6. Licensing and Privacy ✗

# What does an open science workflow get you?

- Primary Benefit is Facilitating replication and strengthening the evidence base
  - Internally: Streamlined analytic process
  - Externally: Improved collaboration
  - Efficiencies:
    - Better framework for fixing and recovering from errors
    - Enhanced throughput
      - Use of past processes to inform future work
      - More natural evolution of scientific product
      - Progeny

# Example of a complication

<u>Outcome</u>	<u>S1</u>		<u>S2</u>		<u>S3</u>		<u>S4</u>
Cognitive Function	As is		Z-Score		Group Specific Z-Scores		Threshold Based Grouping
<u>Covariates</u>							
Age	Continuous		4 Groups				
Race/Ethnicity	As is		Regrouped				
Education	Continuous		5 Groups				
CESD	Continuous		Global Threshold Grouping		Group Specific Thresholds		

$$5 * 2 * 2 * 2 * 3 = 120 \text{ possibilities}$$



# Replication

- Workflow effectiveness -> enables replication
  - Be planful starting today
  - Universal concern with replication in scientific fields
- Easy metric for success in creating Open Science workflow
  - Use existing gauge – your “manuscript” is ready when it is ready for peer-review/wider readership
  - Your workflow is effective and replicable if your scientific process is ready for public view

# Replication is complicated

Ask yourself: Can someone else use my project files to discern my intent, clearly see my presuppositions and guiding assumptions, make sense of my process, understand the reasoning for my decisions, and reproduce my findings

Answer is in Documentation:

- Detailing of process
- Explicit choice of tools that facilitate public documentation of scientific process
- Protection against document leaks; version control

# Some Criteria to consider when picking a framework

- How simple is it to use?
  - Critical in the beginning
- Is it suitable for your personal needs
- Does it enhance current workflow
- Is it sustainable as a “longer-term” solution
- Can it be scaled to expected growth (multiple projects, lab needs, collaborations)
- Does it contribute to standardizing critical production elements
- Does it help with automating repetitive tasks

# Collaborations

- Adds complication to any process
- Collaboration can be a hazard for breakages in workflow
- **Unless** system includes:
  - Clear role definitions
  - Standards for interacting and feeding into the established system
  - Mechanisms for coordination
  - Enforcement rules

# Challenges

Individual research needs:

- Incentive structure not yet established
  - rewards for “openness” not yet fully recognized
- Time costs
  - To set up the system
  - Be productive within the system
- Other systemic constraints (e.g. data restrictions)

# Make the workflow **WORK**

1. Start now!!
2. Gain skills incrementally.
  - Establish habits
  - Integrate complex processes over time
3. Don't design or attempt to change quickly or under time constraint
4. Many viable workflows:
  - Find one that might work (borrow from other efficient users) with your style and personality
  - Make it your own and instill it in your lab members
  - Be flexible to change; be open to having graduate students, post docs, and collaborators show you new ways

# Considerations for open science workflows

1. Creating own workflow or using existing Workflow Management Systems ✓
2. Choosing a Data Repository ✗
3. Deciding on a Source Code Repository ✓
4. Choosing a system to “Package, Access, and Execute Data and Code” ✓
5. Choosing a Document Repository (free or fee for service) ✗
6. Licensing and Privacy ✗

# Things you can do right away

- Start now!
- Keep in mind that:
  - Reproducible does not mean perfect
  - Improving a system is a lot easier than falling behind
- Create a simple set of rules that initially bind you, your lab members and trainees, and eventually your collaborators to the process
- Associate with (attend conferences, follow on social media) and seek help from (correspond directly) with others who work within a similar framework



# Three simple steps to start now

1. Create an account and commit to using a version control system for documenting code
  - I will do a demonstration on how to do so using Git
2. Commit to documentation now
  - Make this part of your and your lab members daily writing routines
  - Have others look at your documentation the same way you have them inspect your scientific writing
3. Adopt practices that allow for replication
  - I will show an example with RStudio and Rmarkdown
  - Other software allow similar processes

# Get git

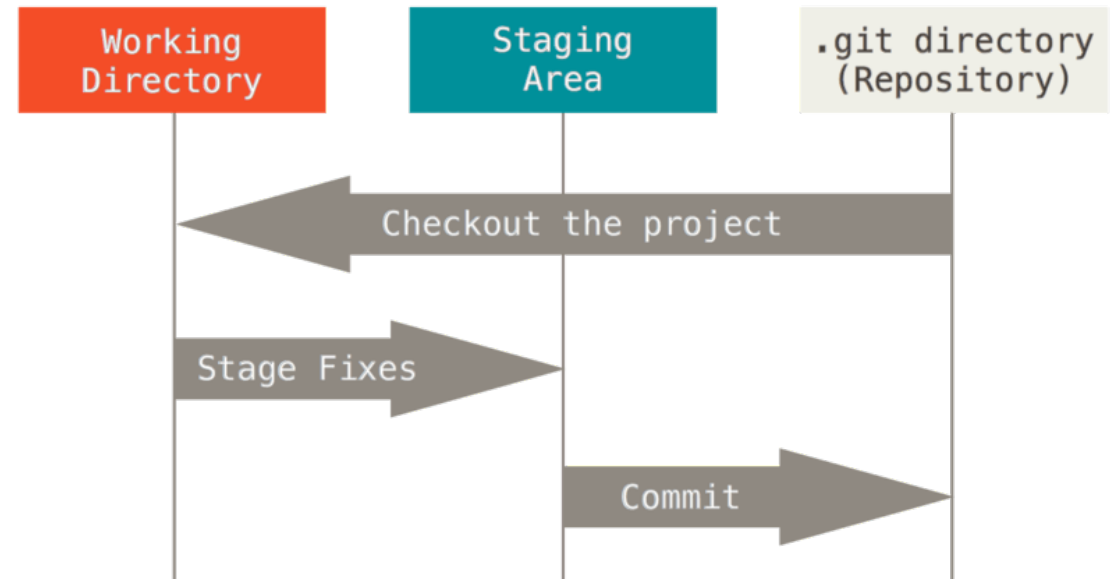
## Download git:

Instructions on how to do so for Linux, macOS, and Windows are available [here](#)

## What is git:








Distributed version control system:

1. Do the work locally
2. Stage it (add needed changes)
3. Commit it to your repo












# From local to repo

## Local project architecture

Name		Date Modified
▼  cog-func-diabetes-2020	✓	Yesterday at 4:18 PM
▶  Abstract	✓	Yesterday at 3:31 PM
▶  Analyses	✓	Yesterday at 3:24 PM
▶  Archive	✓	Yesterday at 3:30 PM
▶  Literature	✓	Yesterday at 3:29 PM
▶  Manuscript	✓	Yesterday at 3:28 PM
 README.md	✓	Yesterday at 4:18 PM

## GitHub repo

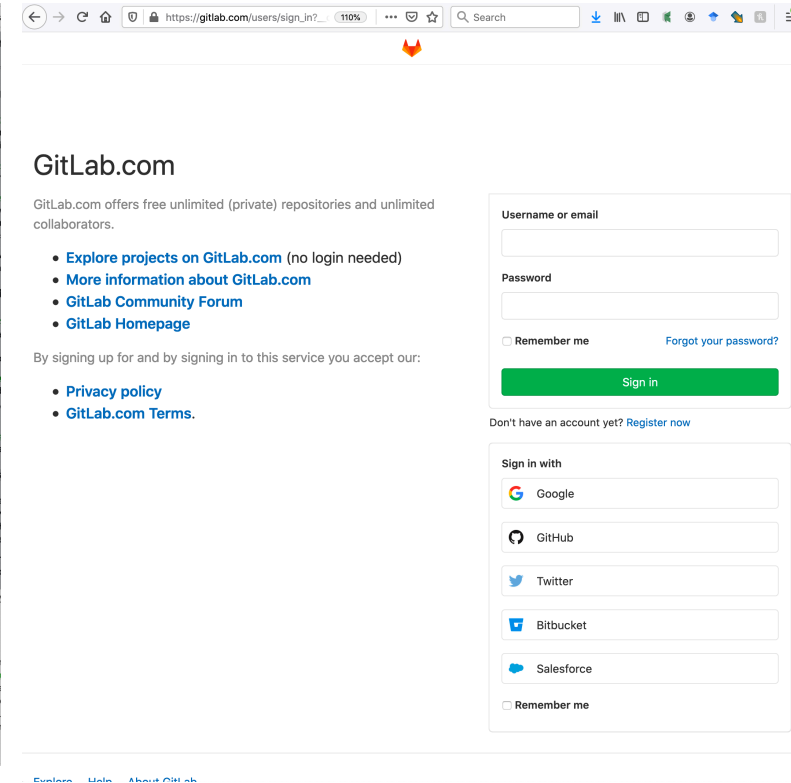
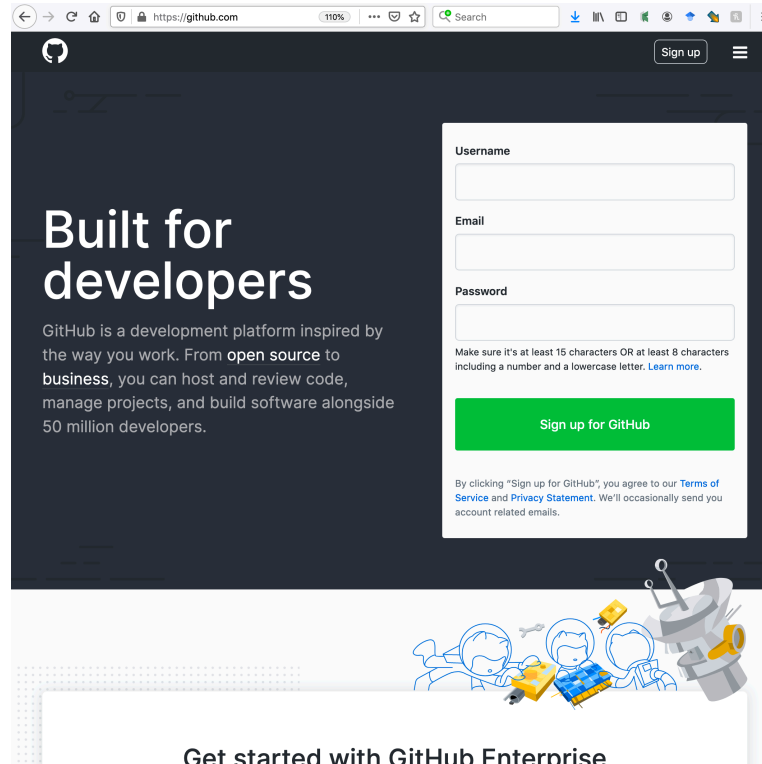
3 commits			1 branch			0 packages			0 releases		
Branch: master ▼			New pull request			Create new file			Upload files		
						Find file			Clone or download ▼		
 wtarraaf Create README.md						Latest commit 5d77dc9 23 hours ago					
 Abstract			Commit Project						23 hours ago		
 Analyses			Commit Project						23 hours ago		
 Archive			Commit Project						23 hours ago		
 Literature			Commit Project						23 hours ago		
 Manuscript			Commit Project						23 hours ago		
 .DS_Store			Commit Project						23 hours ago		
 .gitignore			.gitignore						23 hours ago		
 README.md			Create README.md						23 hours ago		

# Github (or Gitlab)

## Create a GitHub or GitLab account

What are these:  
Platforms for hosting (mostly)  
software based on git  
Offers:

[distributed version control](#) – peer-to-peer – each user has local copy and access to the full history of code (or other documents)  
Mostly used in open source projects  
Offers [functionalities for code management](#) (branching, merging, forking, cloning, etc..)



# Get started with Git and GitHub

1. Log in to <https://github.com> or to <https://gitlab.com>
2. Create a new repo by clicking on the green “New Repository” button
3. Name your repository - I usually use the same project name that I’ve created locally
4. Choose whether you want the repo to be public or private
5. Initialize without a README or gitignore file (we will come back to this in a bit)

## Create a new repository

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

Owner



wtarraf ▾

Repository name \*

mcuaaar\_demo ✓

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

Description (optional)

Demonstration MCUAAAR 5 Reproducible Research



**Public**

Anyone can see this repository. You choose who can commit.



**Private**

You choose who can see and commit to this repository.

Skip this step if you’re importing an existing repository.



**Initialize this repository with a README**

This will let you immediately clone the repository to your computer.


Add .gitignore: **None** ▾

Add a license: **None** ▾



Create repository

# Connect local project to GitHub repo

 **wtarraf / mcuaaar\_demo** Private

Unwatch 1 Star 0 Fork 0

<> Code

Issues 0

Pull requests 0

Actions

Projects 0

Security 0

Insights

Settings

### Quick setup — if you've done this kind of thing before


Set up in Desktop

 or 

HTTPS

SSH

https://github.com/wtarraf/mcuaaar\_demo.git



Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

Copy the https that was produced

# Segway to RStudio

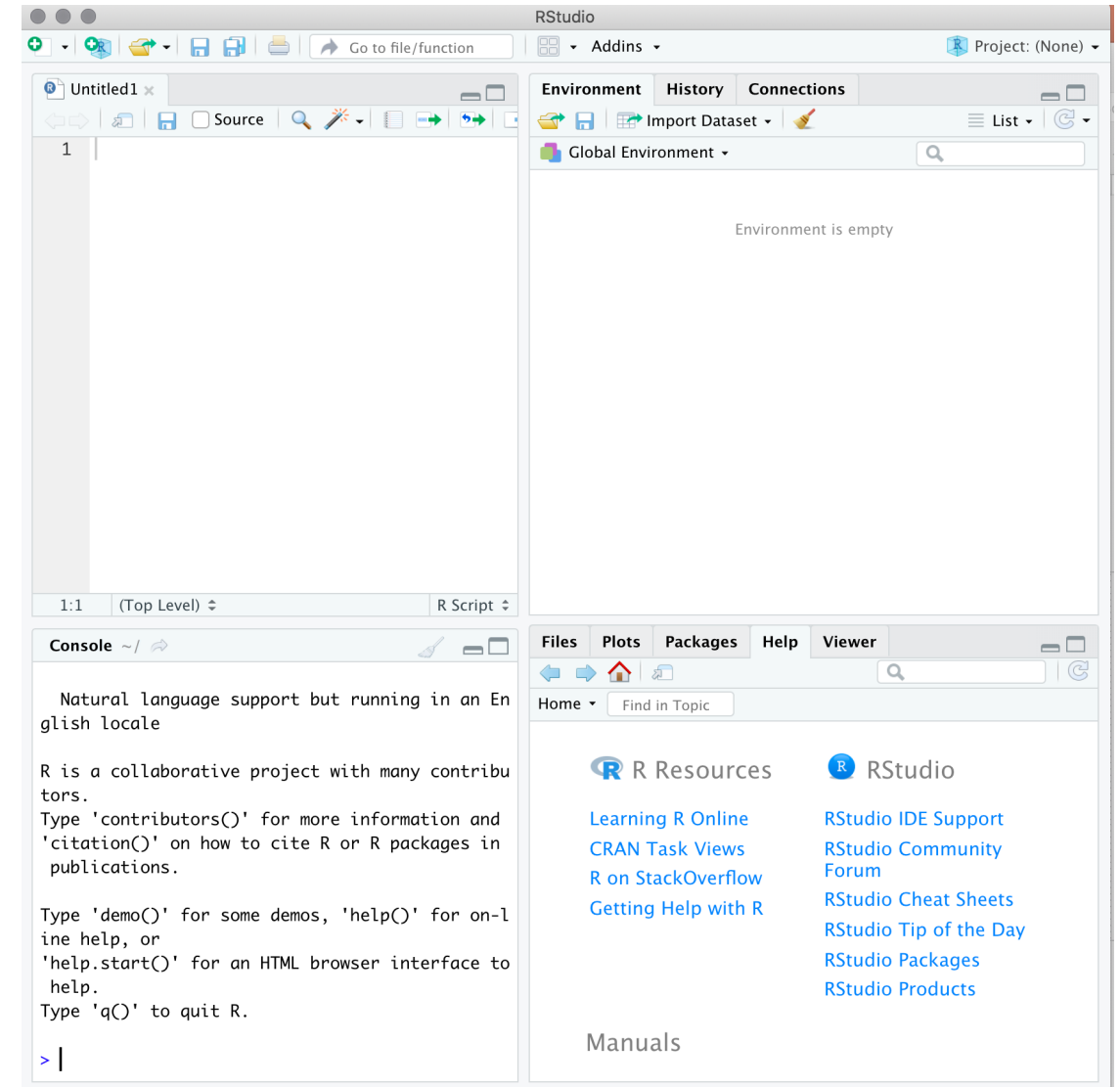


## [Download R](#)

Free programming  
language and statistical  
computing environment

## [Download RStudio](#) Desktop

Free integrated  
development environment  
(IDE) for R

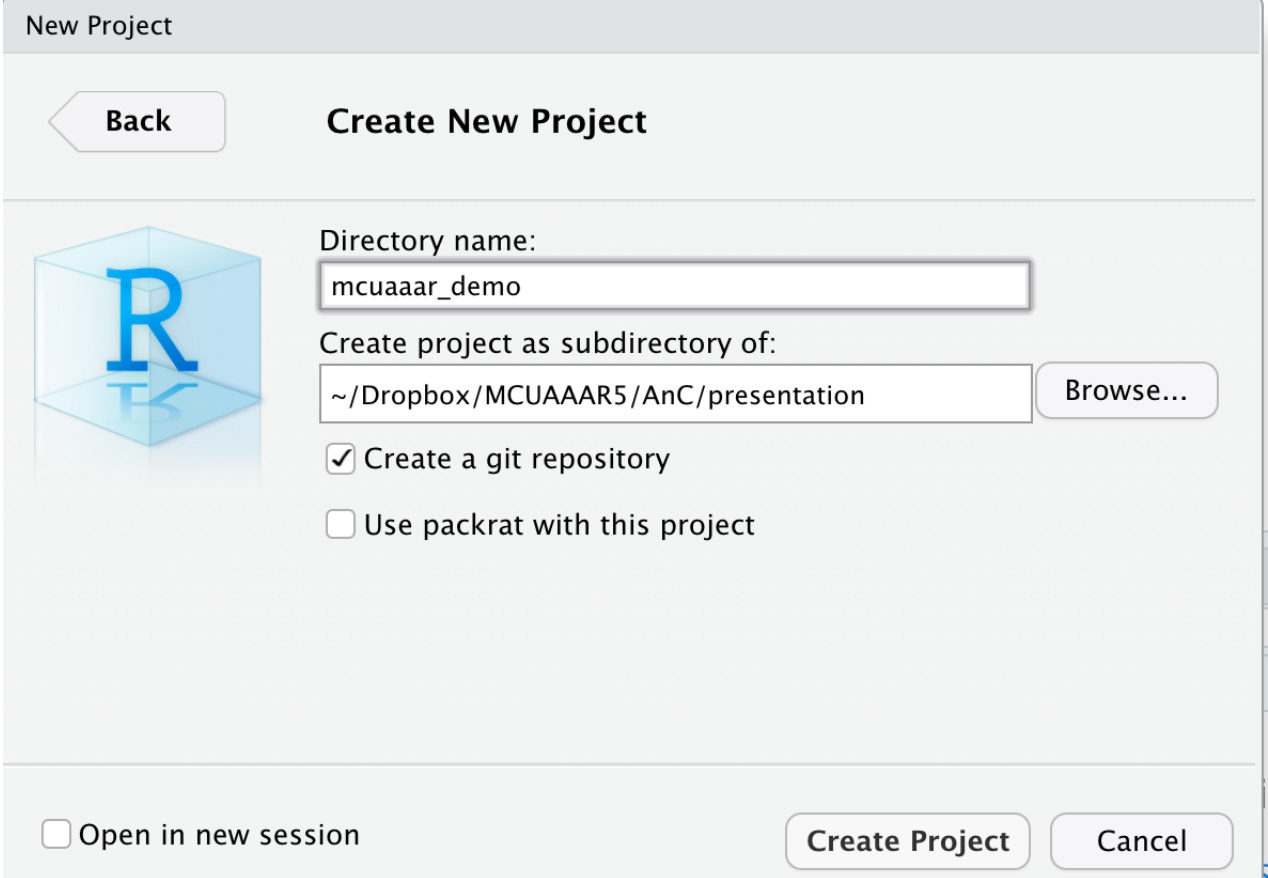


# Working with Rstudio and GitHub

File ->

New Project

Check – create a git repository



The image shows the 'New Project' dialog box in RStudio. The title bar says 'New Project'. Inside, there's a 'Back' button and a 'Create New Project' section. On the left is the R logo. The 'Directory name:' field contains 'mcuaaar\_demo'. The 'Create project as subdirectory of:' field contains '~/Dropbox/MCUAAAR5/AnC/presentation', with a 'Browse...' button to its right. There are two checkboxes: 'Create a git repository' (checked) and 'Use packrat with this project' (unchecked). At the bottom, there's an 'Open in new session' checkbox (unchecked), a 'Create Project' button, and a 'Cancel' button.

New Project

Back Create New Project

Directory name:  
mcuaaar\_demo

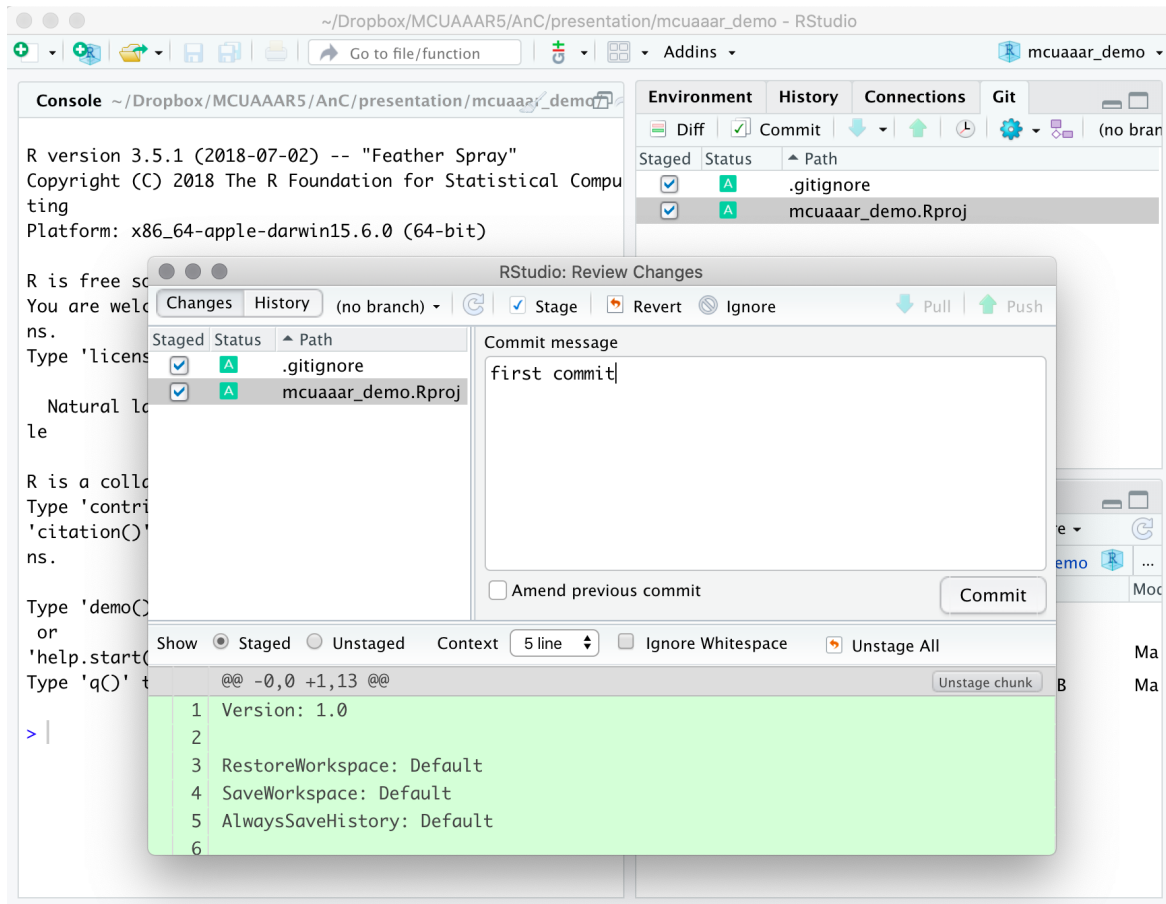
Create project as subdirectory of:  
~/Dropbox/MCUAAAR5/AnC/presentation Browse...

☒ Create a git repository  
☐ Use packrat with this project

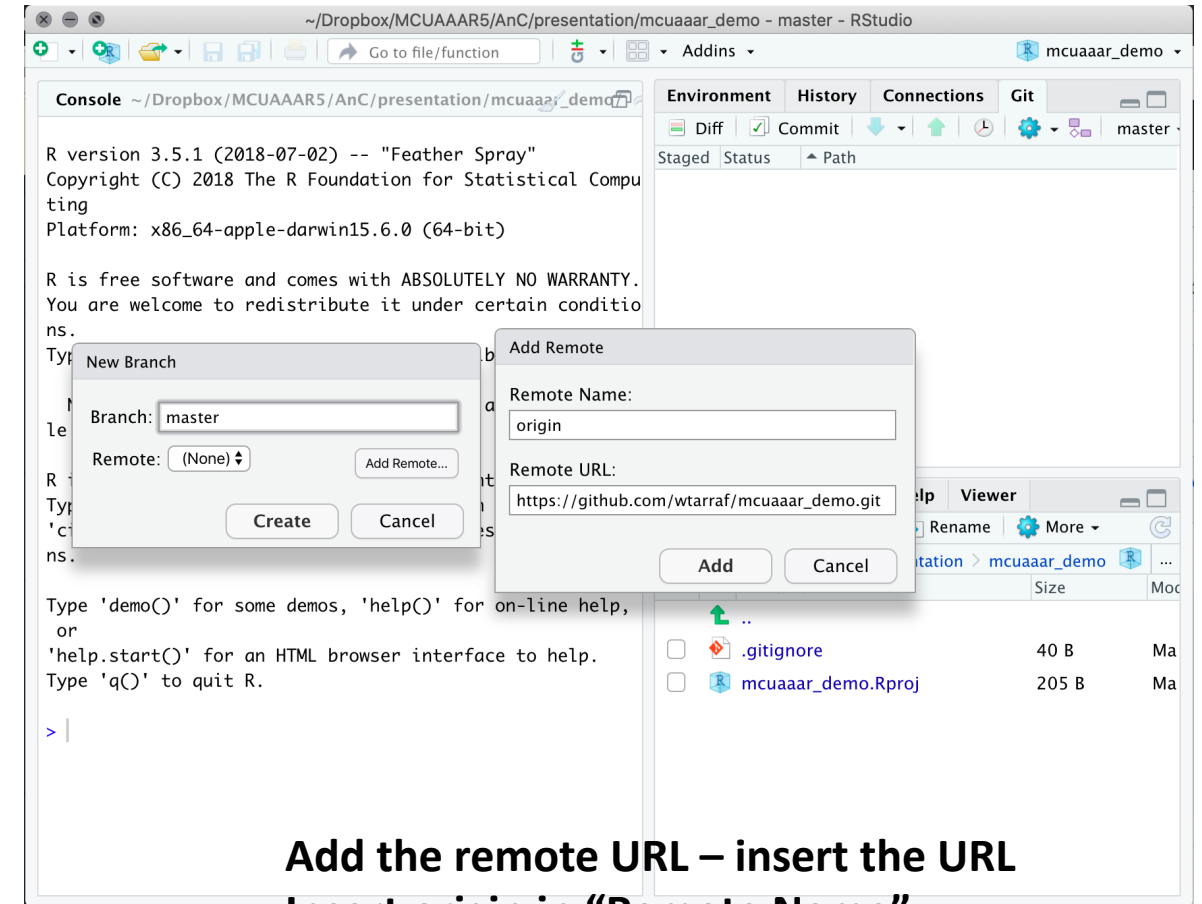
☐ Open in new session Create Project Cancel



# Initialize



**Create a first commit**



**Add the remote URL – insert the URL**

**Insert origin in “Remote Name”**

**Press Add**

**Insert master in “branch”**

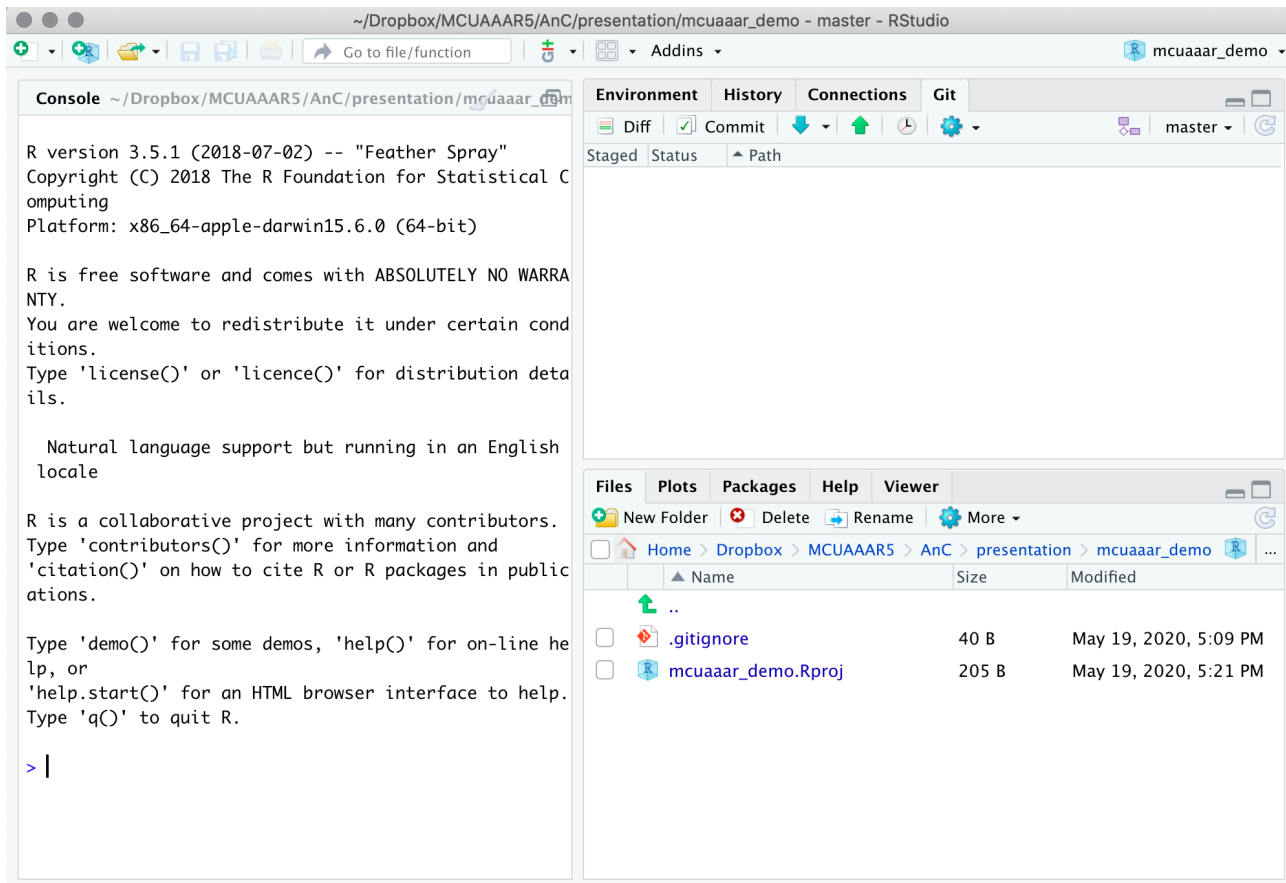
**Press Create**

**Check sync branch with remote**

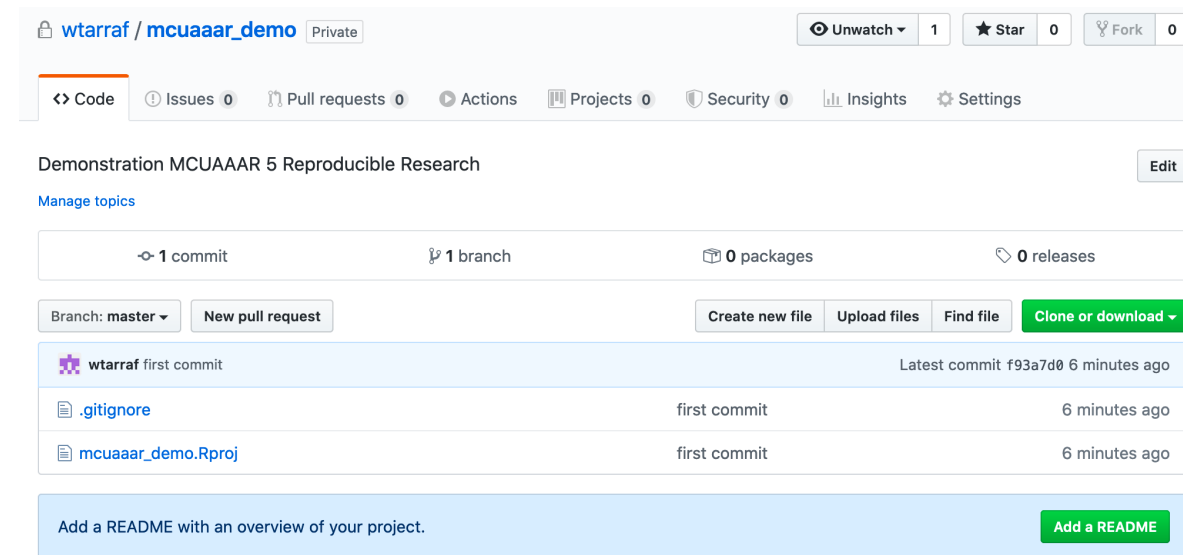
**Choose Overwrite**

# You are ready

## Local project

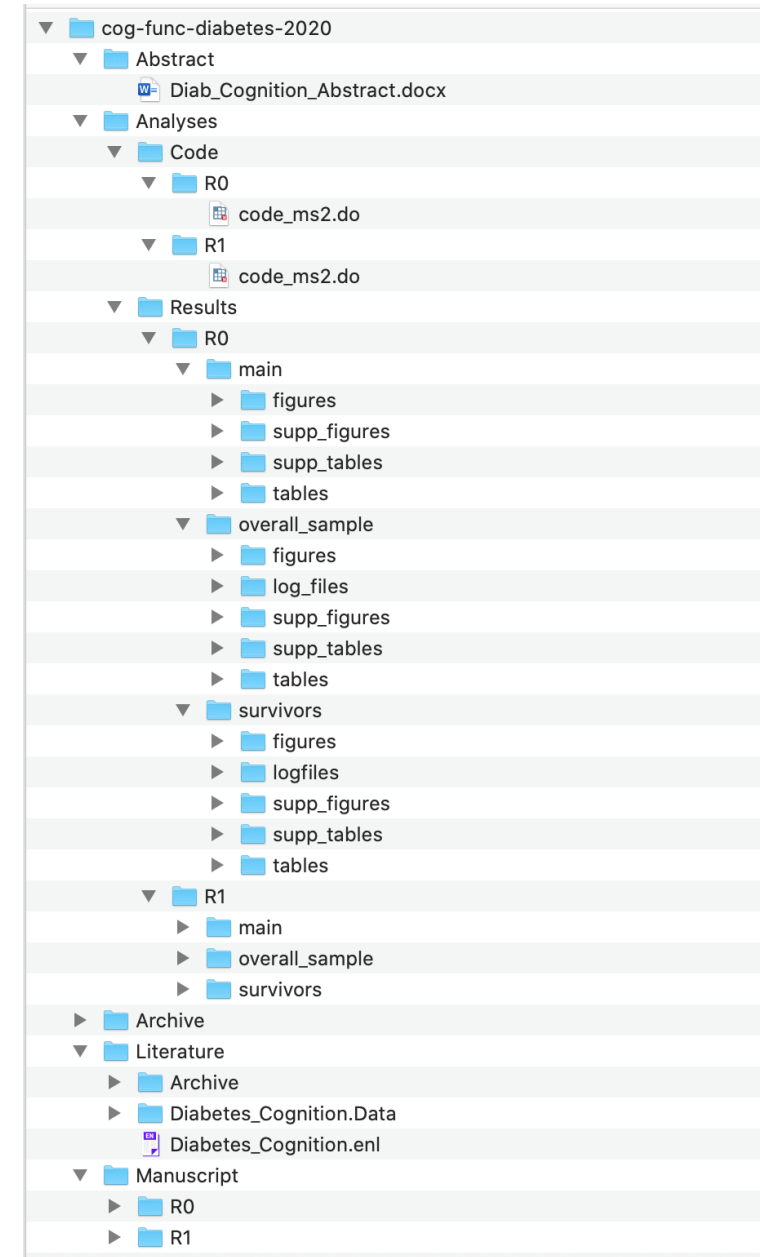


## GitHub Repo



# Be planful with your project infrastructure

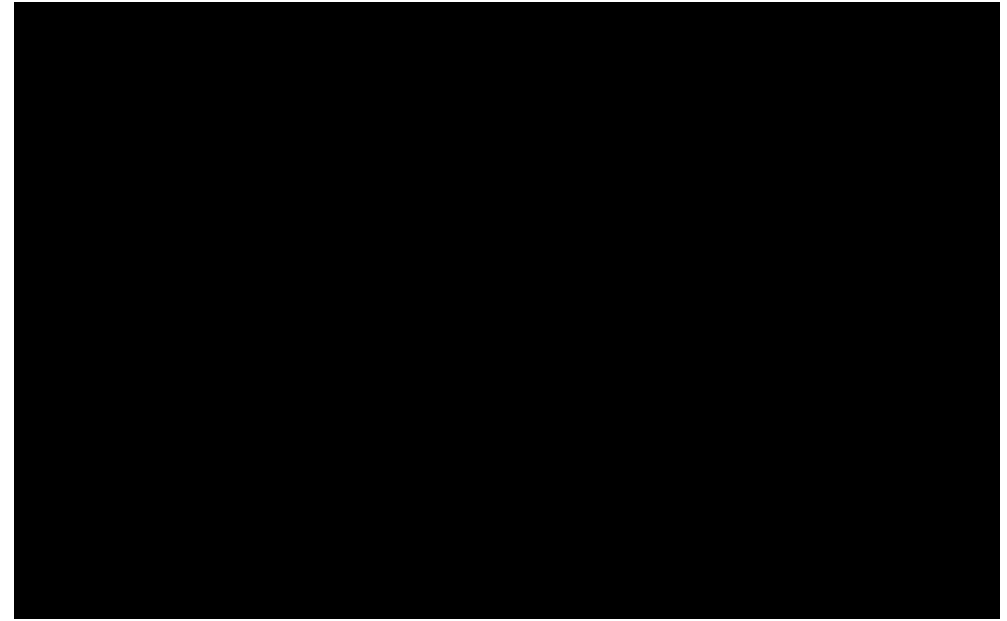
- Plan and incrementally improve the organizational structure
  - Strive for easy to follow structure that reflects the way you approach your research
  - Streamline (create near uniformity) to facilitate cloning of structure across projects
  - Make smart decisions about
    - What to name your folder, subfolders, and documents
    - Where, when, and what to save
    - How often and what to commit
- The more of it you do the better you get at it



# Work in RStudio

## Rmarkdown:

Notebook interface that weaves narrative, code, results, and visualization

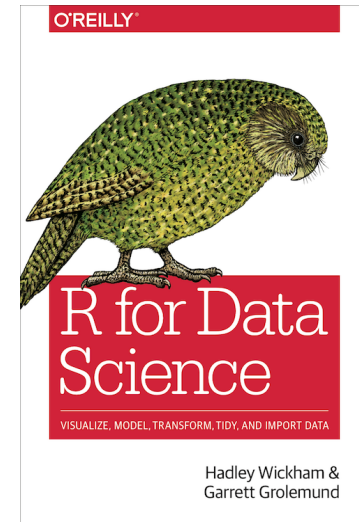
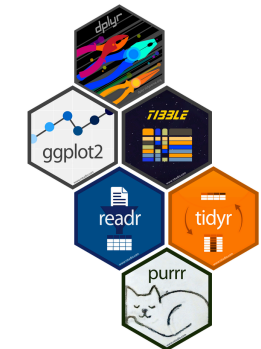


The 'tidyverse' - a collection of packages  
3 packages to begin

haven – to import data

dplyr – to wrangle the data

ggplot2 – to plot the data



# Working with data – an Rmarkdown example

(1) Import data

(2) Prepare the data

Data mergers

Determine a set of  
observations and variables of  
interest

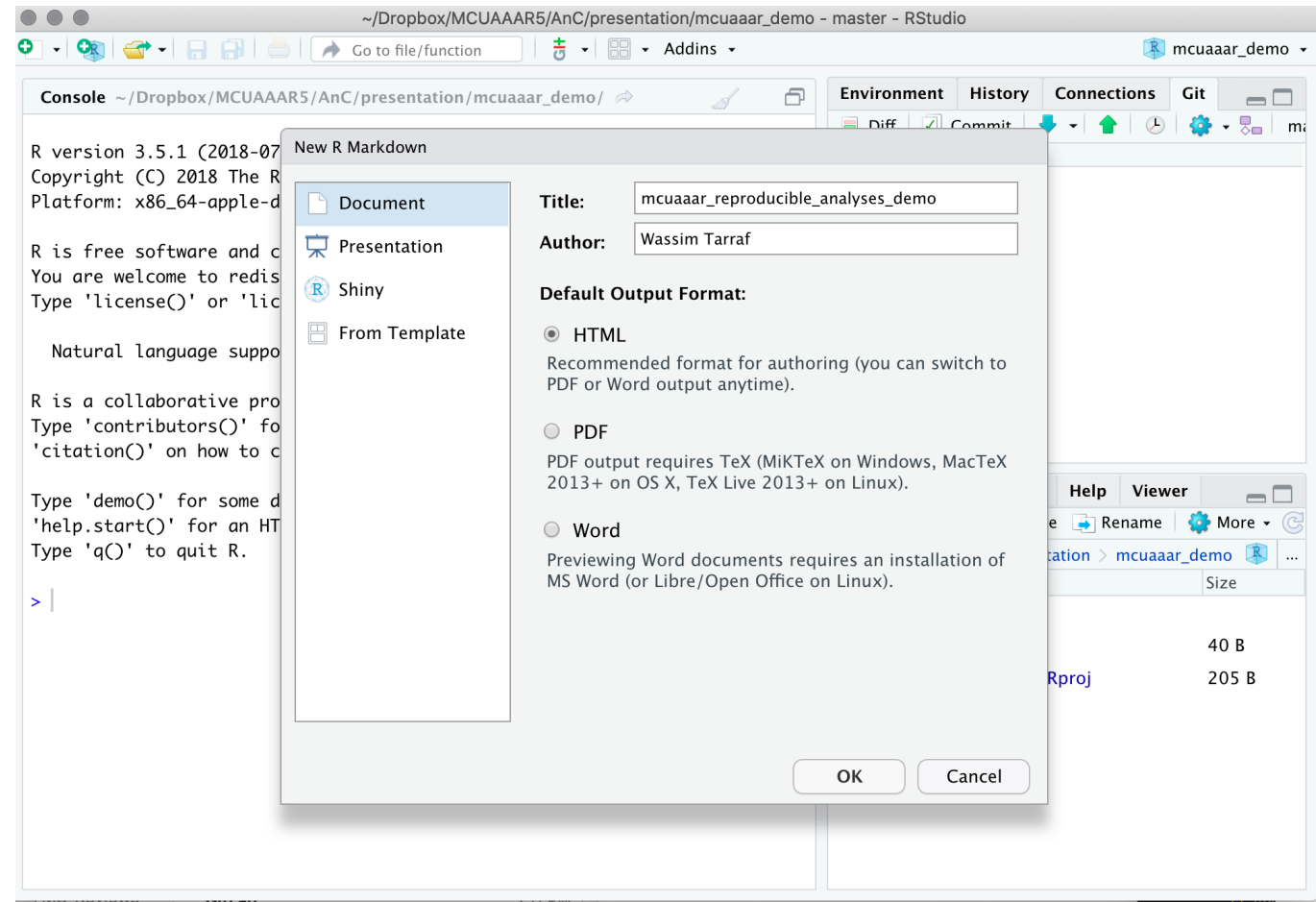
-> Filters (data split)

-> Selections (data  
reduction)

Consider transformation of  
the variables (mutate)

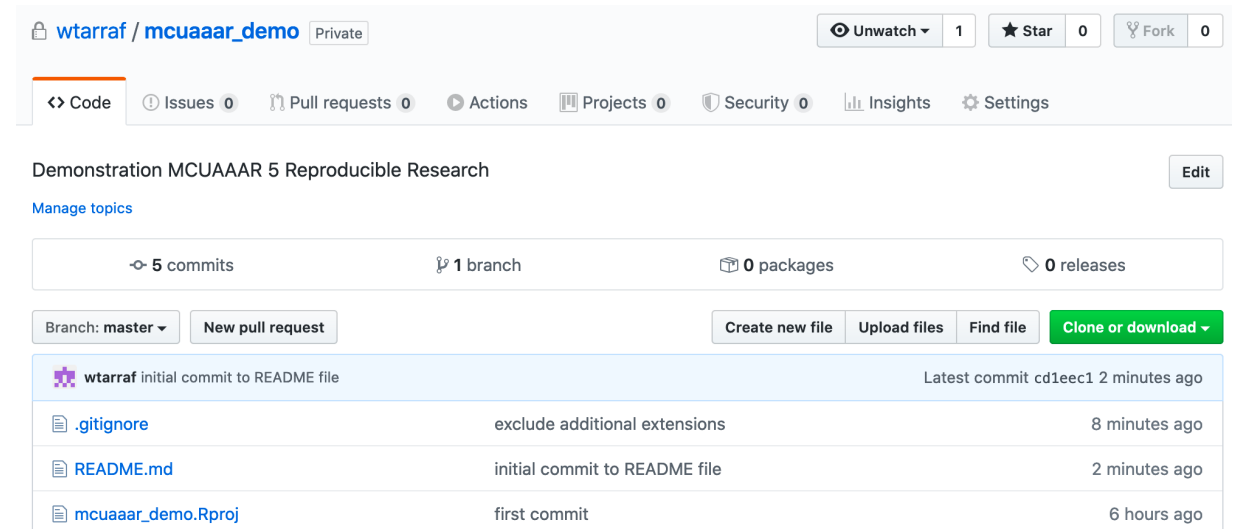
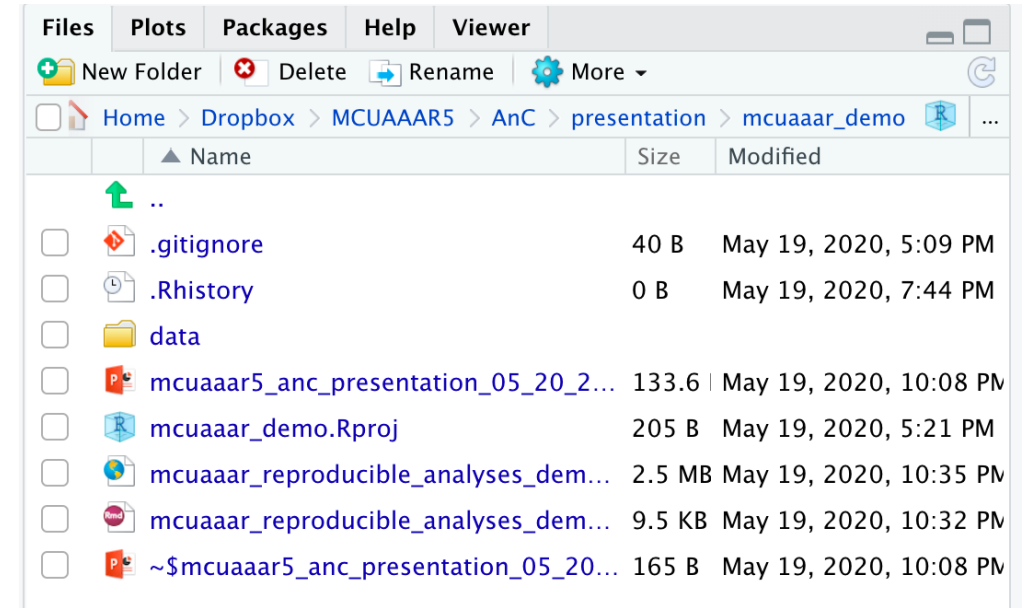
(3) Model your data

(4) Visualize it



# Back to git

- Do this as often as needed
  - add*
  - commit*
  - push*
- When collaborating
  - Learn how to *branch*
  - pull*
  - and if necessary *merge*
- Make use of what others have to offer
  - clone*
  - fork*



# Back to git – reconciling local and repo

- Do this as often as needed

*add*  
*commit*  
*push*

- When collaborating

Learn how to *branch*  
*pull*

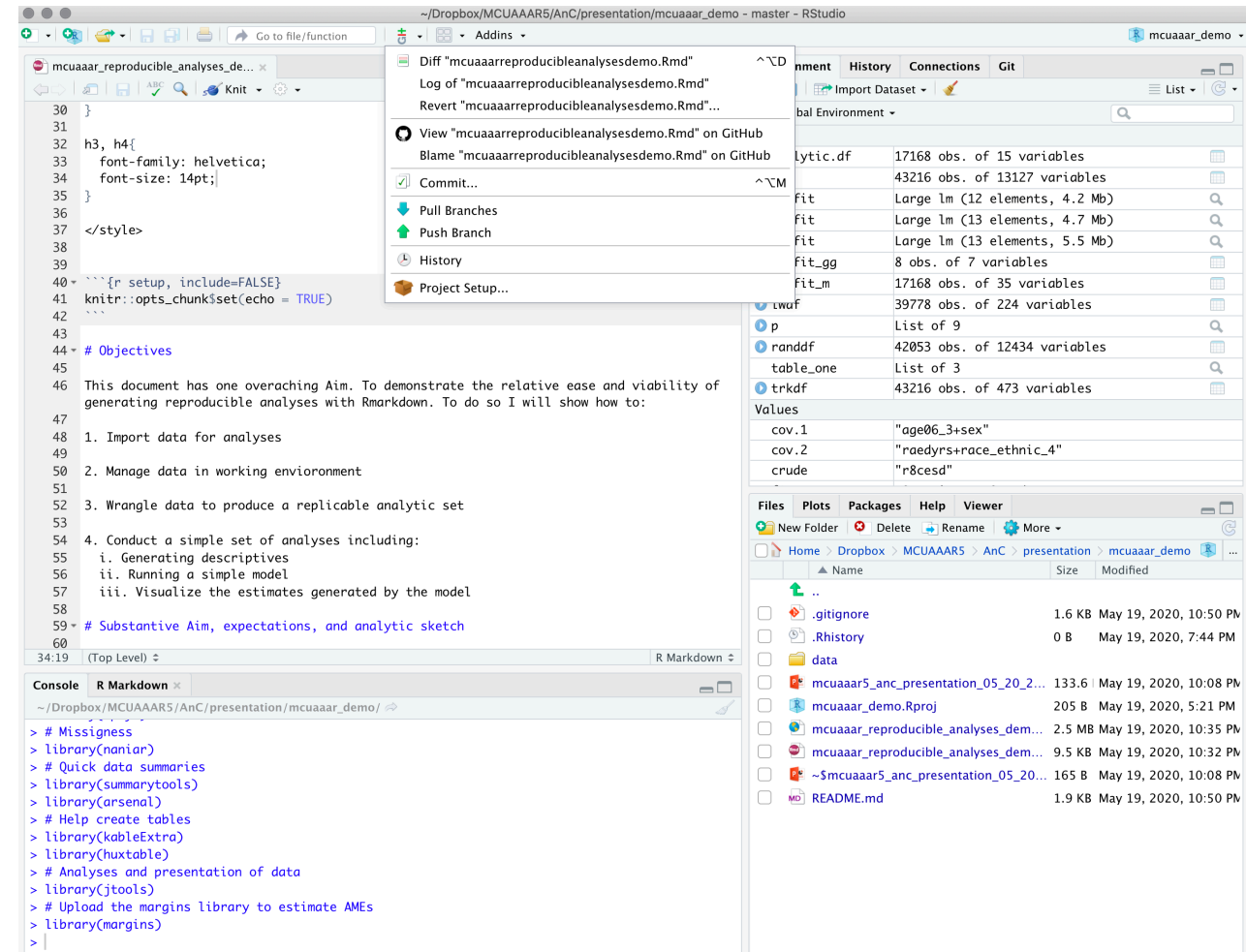
and if necessary *merge*

- Make use of what others have  
to offer

*clone*  
*fork*

Pull changes that I added to my repo locally:

- (1) I updated my .gitignore to restrict types of files that I can push
- (2) I created a README file to describe the project





# Back to git – reconciling repo and local

- Do this as often as needed

*add*  
*commit*  
*push*

- When collaborating

Learn how to *branch*  
*pull*

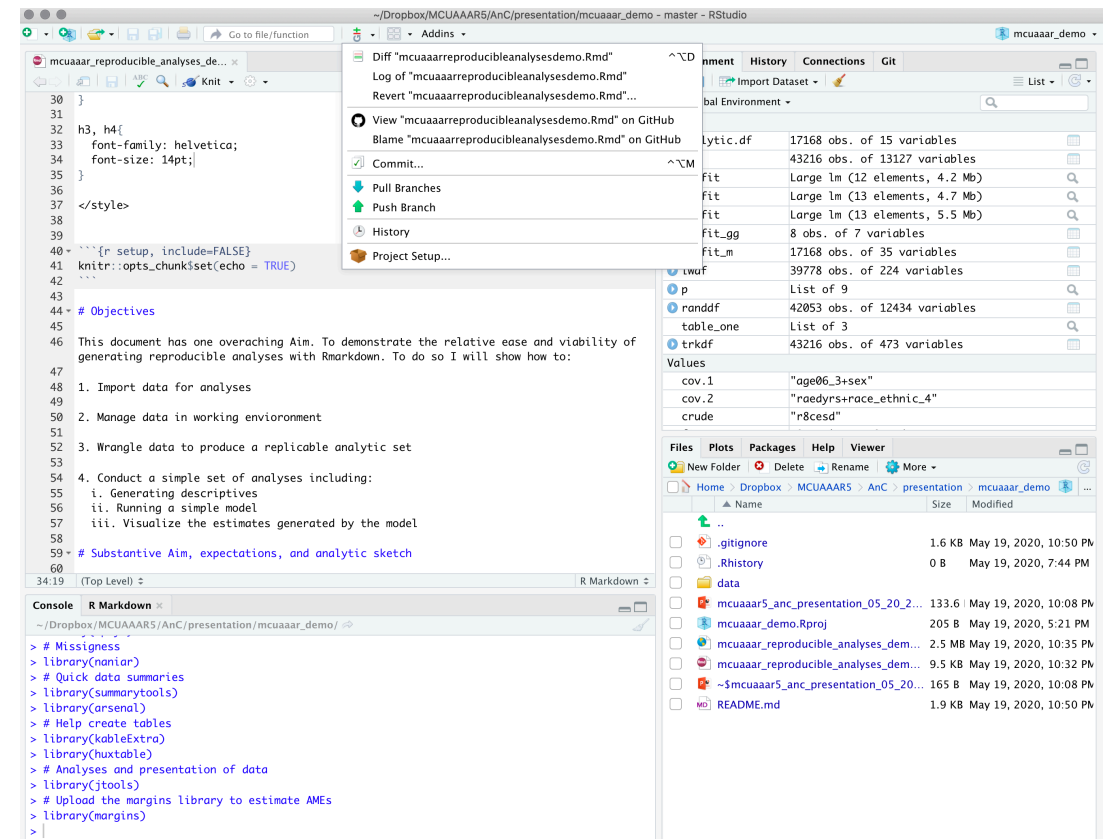
and if necessary *merge*

- Make use of what others have  
to offer

*clone*  
*fork*

Pull changes that I added to my repo locally:

- (1) I saved the powerpoint presentation – I will choose not to push it (i.e. keep it locally)
- (2) I saved my Rmarkdown file
- (3) I saved an HTML version of my knitted Rmarkdown file
- (4) although I added a lot of data – the push for these is restricted (see gitignore)





# Back to git – reconciling repo and local

- Do this as often as needed
  - add*
  - commit*
  - push*
- When collaborating
  - Learn how to *branch*
  - pull*
  - and if necessary *merge*
- Make use of what others have to offer
  - clone*
  - fork*

Pull changes that I added to my repo locally:

- (1) I saved the powerpoint presentation – I will choose not to push it (i.e. keep it locally)
- (2) I saved my Rmarkdown file
- (3) I saved an HTML version of my knitted Rmarkdown file
- (4) although I added a lot of data – the push for these is restricted

The screenshot shows a GitHub repository page for 'wtarraf / mcuaaar\_demo'. The repository is private and has 1 star and 0 forks. The main content area shows the repository name 'Demonstration MCUAAAR 5 Reproducible Research' and a list of commits. The commits list includes files like .gitignore, README.md, mcuaaar\_demo.Rproj, mcuaaar\_reproducible\_analyses\_demo.Rmd, and mcuaaar\_reproducible\_analyses\_demo.html, with their commit messages and timestamps.

File	Commit Message	Time Ago
.gitignore	exclude additional extensions	21 minutes ago
README.md	initial commit to README file	15 minutes ago
mcuaaar_demo.Rproj	first commit	6 hours ago
mcuaaar_reproducible_analyses_demo.Rmd	commit Rmd, and HTML files	2 minutes ago
mcuaaar_reproducible_analyses_demo.html	commit Rmd, and HTML files	2 minutes ago

# Thanks!

follow-up and questions:  
wassim.Tarraf@wayne.edu