GitFun!

Ana Martinovici

20200206

Fun, but first plan

- News
- **Why** we should use version control
- **How** to use version control

News
source: Antonio Schettino (OSCR)

- Talk by Daniel Lakens – February 11 at 15.00 in room T13-67
  - https://www.openscience-rotterdam.com/2020/01/dpecs-lakens-feb2020/

- Open science festival! August 27 2020, Wageningen
  - https://opensciencefestival.nl/

- How to use R and the tidyverse to clean, plot, and analyze data:
  - https://github.com/aschetti/MPI2020_intro_tidyverse


WHAT DO YOU MEAN
THERE'S NO VERSION CONTROL?
makeameme.org

WHY

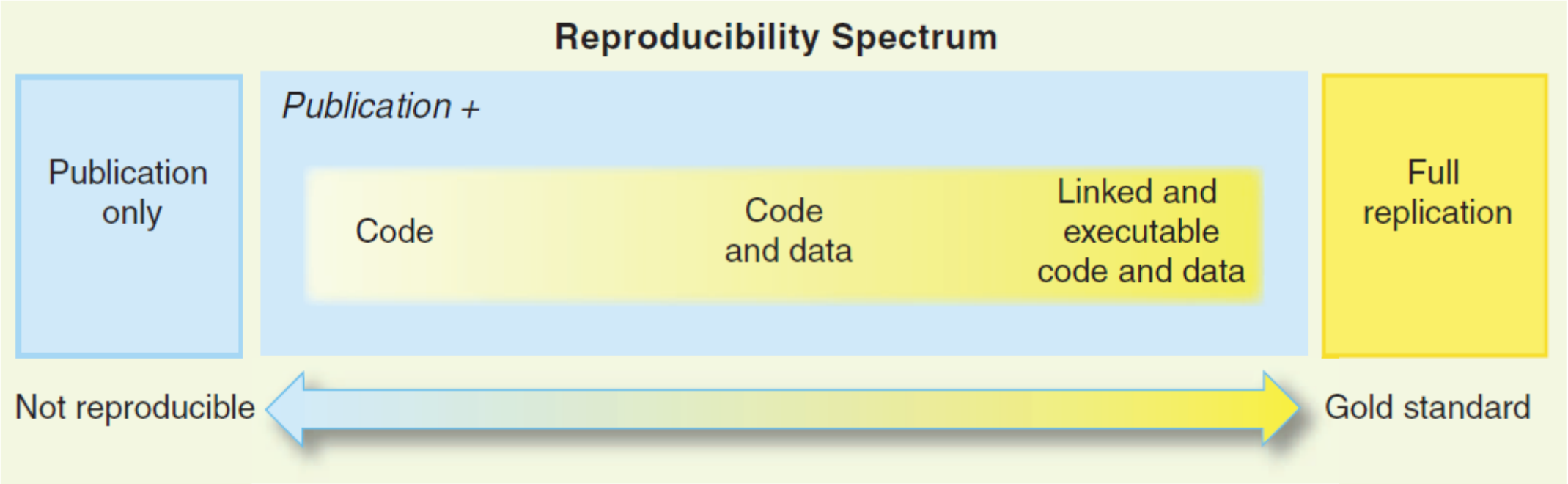
(some) Research goals

- Collaborate with co-authors
- Contribute to ongoing projects
- Share your work with the world
- Reproducible results

Reproducibility?

A study is reproducible if you can take the original data and the computer code used to analyze the data and reproduce all of the numerical findings from the study.

Reproducibility? Yes, No, Maybe



Source: Roger D. Peng (2011) “Reproducible Research in Computational Science”, Science

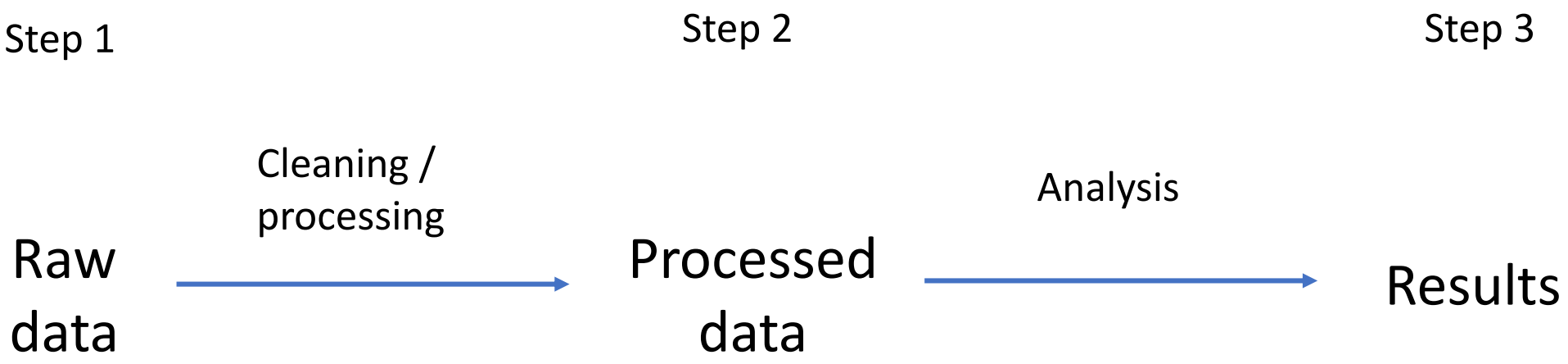
Objective

Reproduce ALL results and NOTHING BUT results included in:

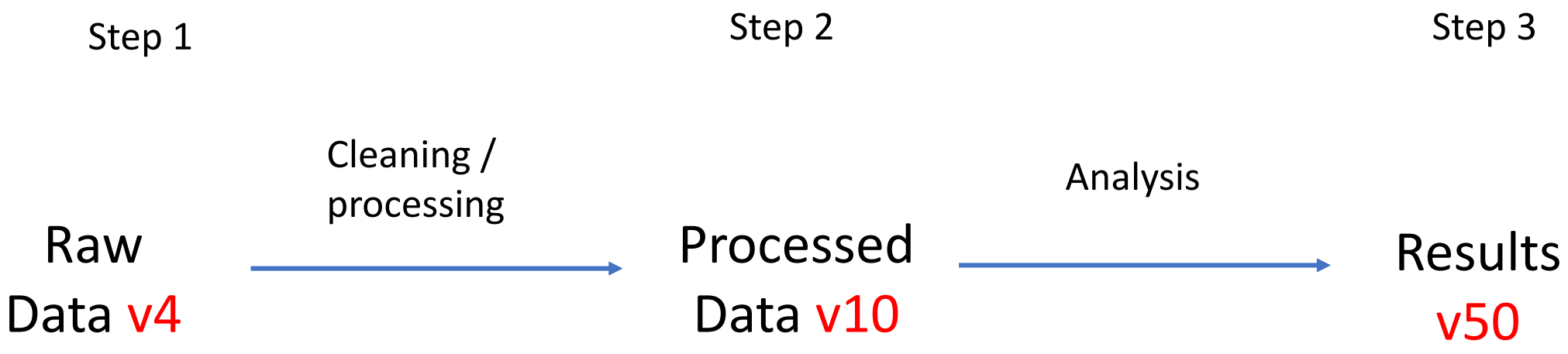
- Current version of the paper
- Previous analyses

as fast and easy as possible.

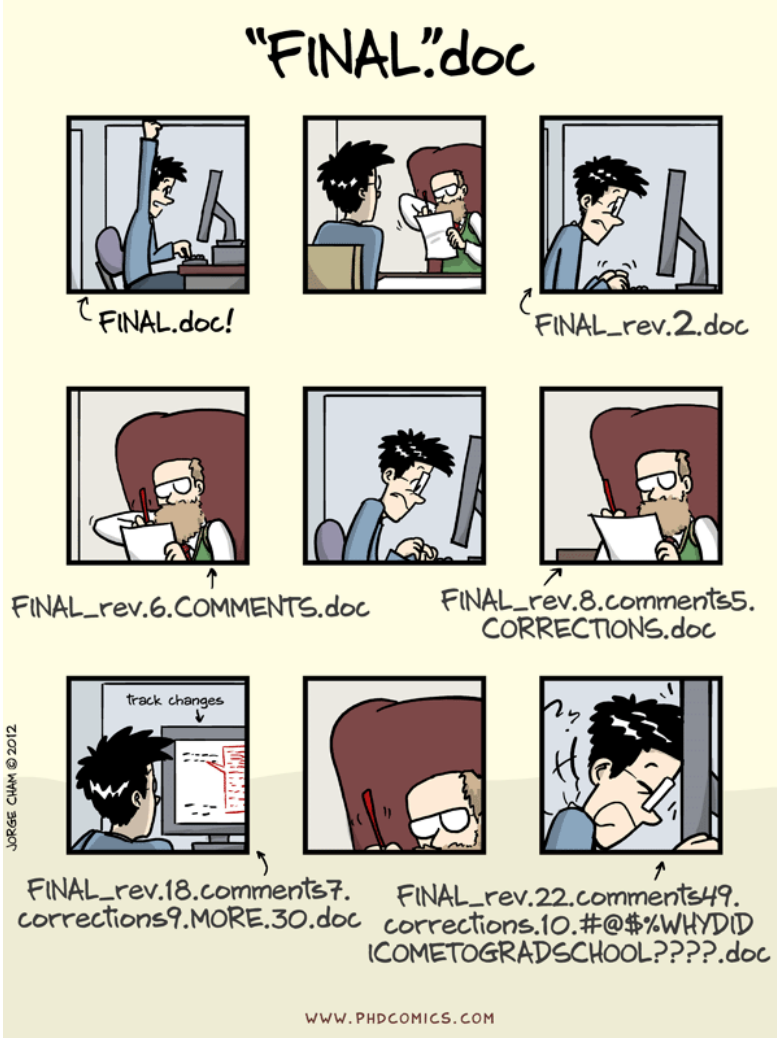
Best case scenario

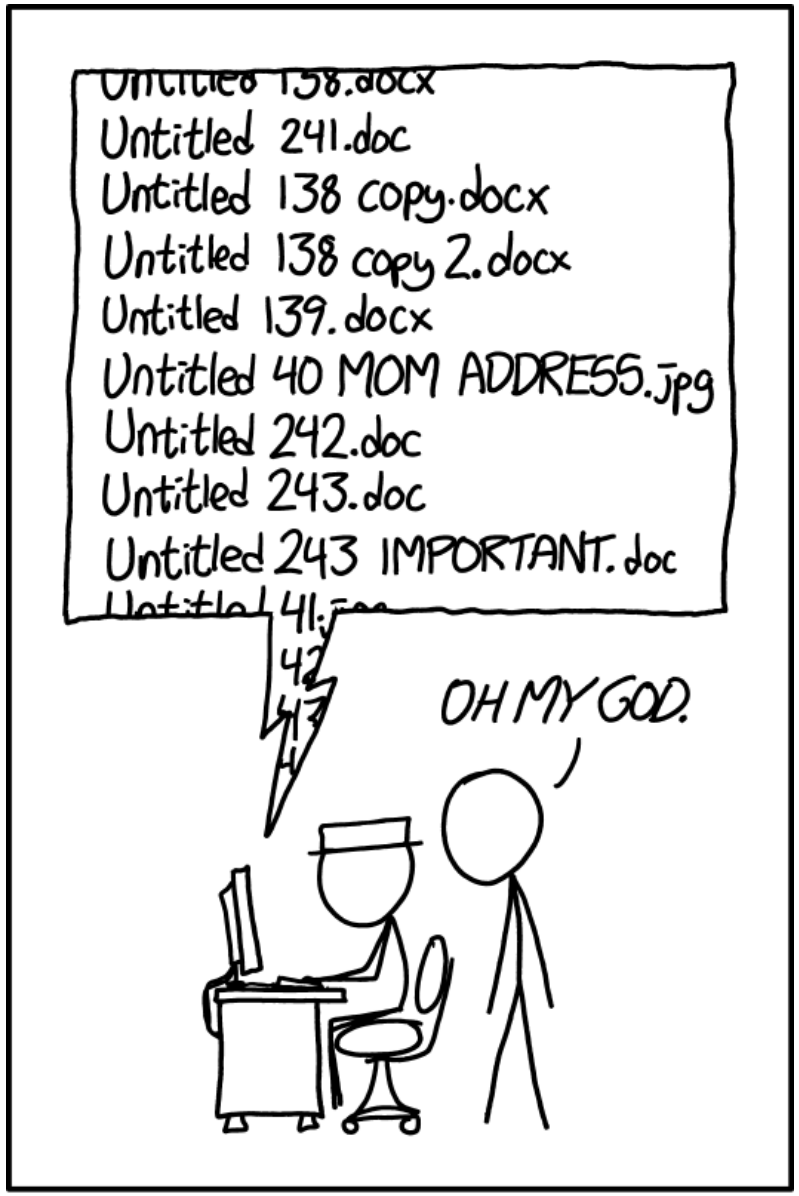


Most likely scenario



Final is never final





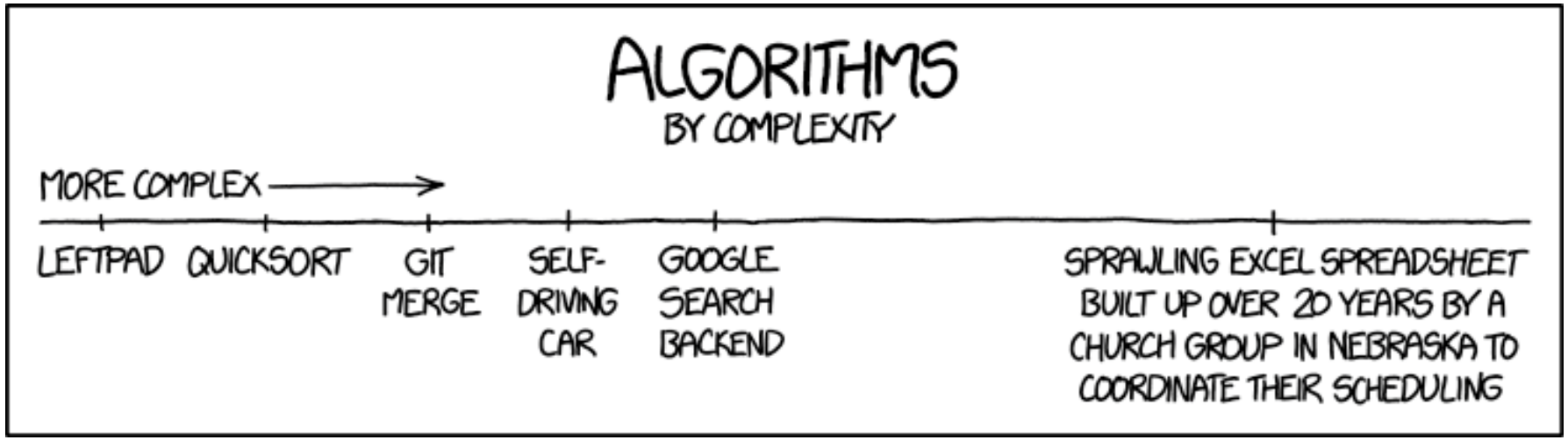
PROTIP: NEVER LOOK IN SOMEONE ELSE'S DOCUMENTS FOLDER.

Things that can go wrong

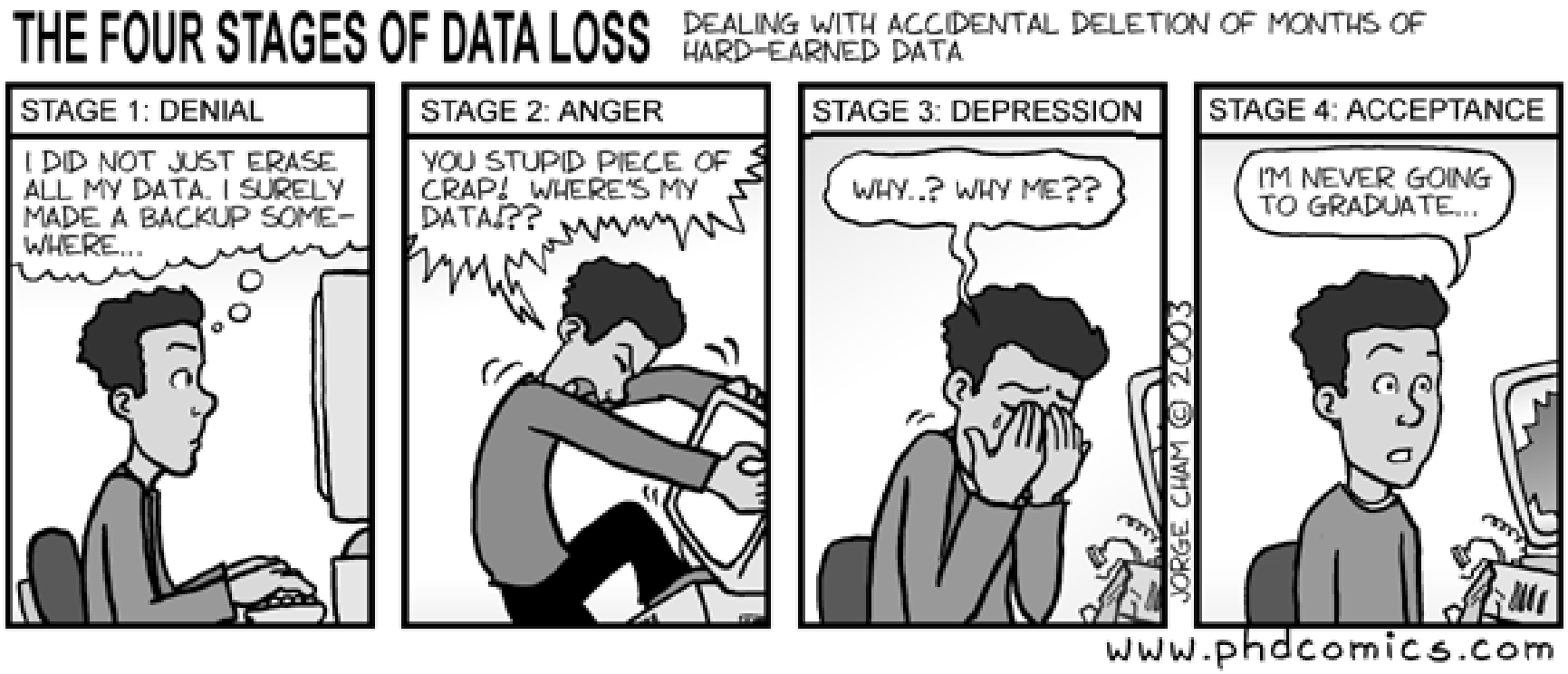
- Your PC/laptop/external HD explode
- Overwriting files, but also not overwriting files
- Forgetting which of the “final” files is really final
- Change file X, but forget to update all the other files/results that depend on it
- Software changes

Things that can go wrong

- Excel (in so many ways)



Things that can go wrong



Version control:

- Time travel
- Safe place for your data and code
- Notes to your future self

What is git?

- Formal version control system
- Developed by Linus Torvalds
  - Used to manage the course code for Linux
- Tracks content:
  - Source code
  - Data analysis projects
  - Websites
  - Presentations
  - Manuscripts

What is GitHub?

- A home for git repositories
- Interface for exploring public git repositories
- A ‘safe place’ to keep code and data
- Additional benefits: issues, projects, wiki, insights

Why use GitHub?

- Facilitates
  - Exploring code
  - Tracking issues
  - Learning from others
- Lowers the barriers to collaboration
  - Email “there’s a typo in your code in file X, line 30” vs
  - Pull request “here’s a correction to your code”
- Free for researchers and students

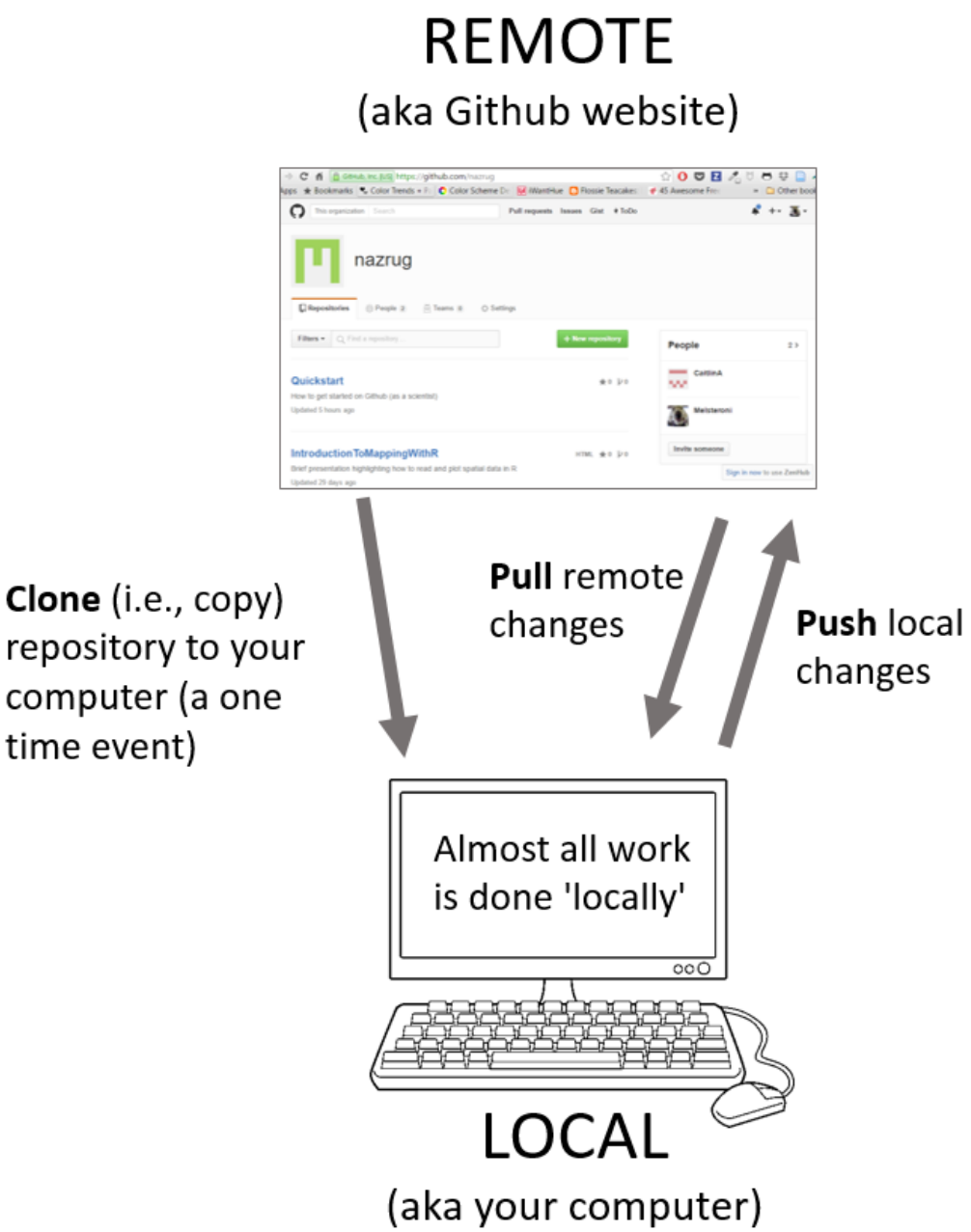

I SHOULD USE GITHUB
imgflip.com

HOW

$195 for 3h Intro in NYC

- https://www.nobledesktop.com/classes/git-classes-nyc

General idea



https://jules32.github.io/2016-07-12-Oxford/git/

Vocabulary

- **repository (repo)**: *(noun)* folder containing all tracked files as well as the version control history
- **commit**: *(noun)* a snapshot of changes made to the staged file(s); *(verb)* to save a snapshot of changes made to the staged file(s)
- **stage**: *(noun)* the staging area holds the files to be included in the next commit; *(verb)* to mark a file to be included in the next commit
- **track**: *(noun)* a tracked file is one that is recognized by the Git repository
- **branch**: *(noun)* a parallel version of the files in a repository

Vocabulary

- **local**: *(noun)* the version of your repository that is stored on your personal computer
- **remote**: *(noun)* the version of your repository that is stored on a remote server; for instance, on GitHub
- **clone**: *(verb)* to create a local copy of a remote repository on your personal computer
- **fork**: *(noun)* a copy of another user's repository on GitHub; *(verb)* to copy a repository; for instance, from one user's GitHub account to your own

Vocabulary

- **merge**: *(verb)* to update files by incorporating the changes introduced in new commits
- **pull**: *(verb)* to retrieve commits from a remote repository and merge them into a local repository
- **push**: *(verb)* to send commits from a local repository to a remote repository
- **pull request**: *(noun)* a message sent by one GitHub user to merge the commits in their remote repository into another user's remote repository

First use of git

```
$ git config --global user.name "Vlad Dracula"
$ git config --global user.email "vlad@tran.sylvan.ia"
```

https://swcarpentry.github.io/git-novice/02-setup/index.html

Challenges

- Data: big, small, too easy to get, too hard to get
- Changes during the review process
- Dependencies (e.g. R packages)
- System specifications
- Lisa/clusters

Checklist

- Have I done anything by hand?
  - If so, are those parts precisely documented?
  - Is that documentation saved in a ‘safe’ place?
  - Is the documentation a complete, correct, and specific description of what was done by hand?
- Have I coded as many of the steps as I could?
- Am I using version control?
- Have I documented the software environment?
- Have I saved any output that I cannot reconstruct from the original data?

Not a good idea: Doing things by hand

- Editing spreadsheets of data to “clean it up”
  - Removing outliers
  - Rescaling (reverse coding)
  - Create new variables (dummy variables, intervals, categories)
- Edit tables or figures (e.g. rounding, formatting)
- Move/split/rename data files on your computer
- “I’m only doing this once…”

Things done by hand need to be precisely documented (harder than it sounds).

Not a good idea: point and click

- Many data processing / statistical analysis packages have graphical user interfaces (GUIs)
- GUIs are convenient but the actions you take can be difficult to reproduce
- Some GUIs produce a log file that can be saved for later examination

Not a good idea: saving output only

- Avoid saving data analysis (intermediary) output except perhaps temporarily for efficiency purposes

- Intermediate files can be ok as long as:
  - (1) there is clear documentation of how they were created and
  - (2) the links to and from these files are working

- Save the data + code that generated the output, rather than the output itself

Homework fun!

To ‘push’ you to practice the clone -> stage -> commit -> push steps ☺

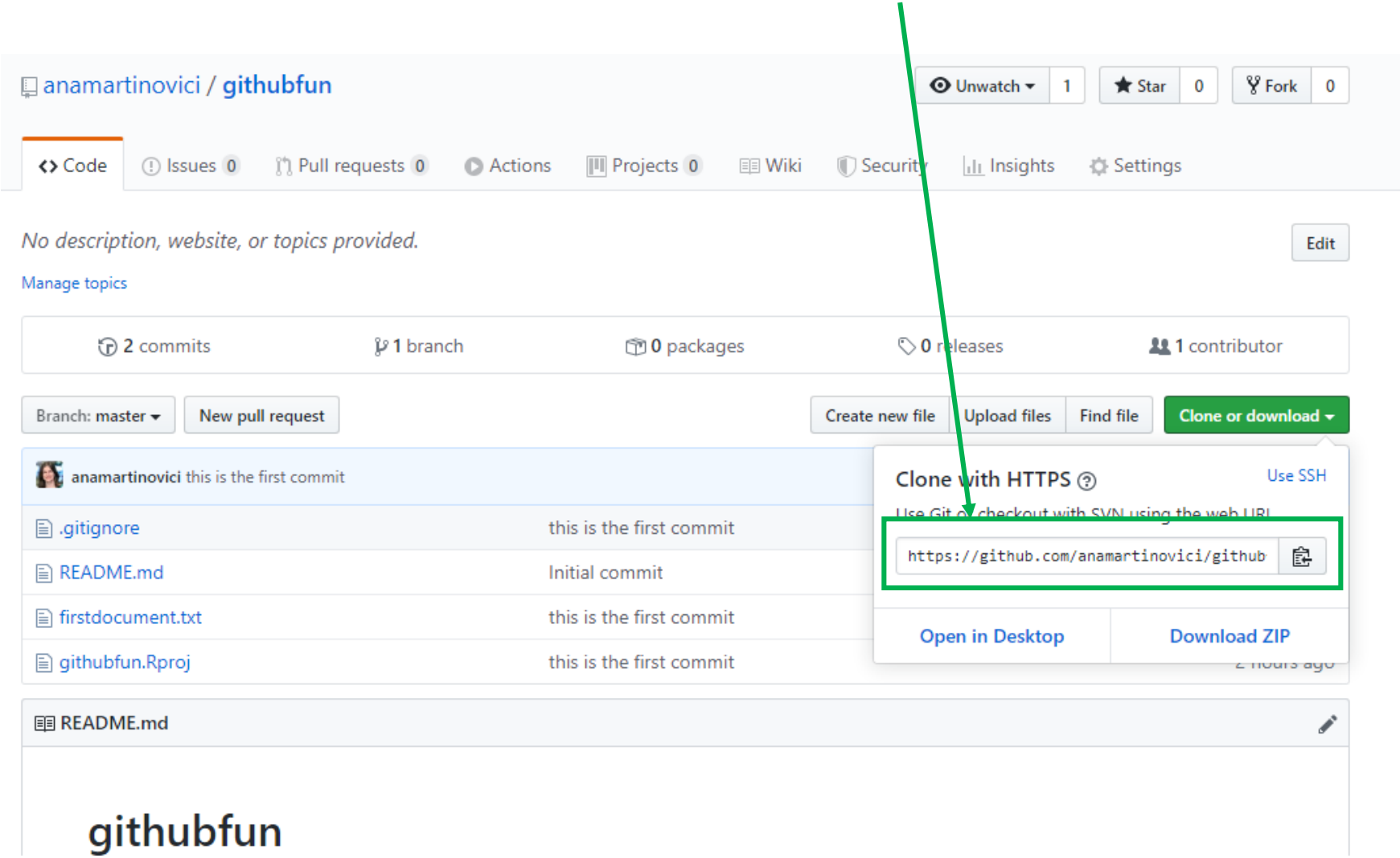
- Go to https://github.com/anamartinovici/githubfun
- Make sure you are a collaborator (check next slide)
- Clone the repository
- Add a txt file with your name (“name.txt”) and some text that you’re confortable sharing with the world
- Push your changes

How to make sure you are a collaborator

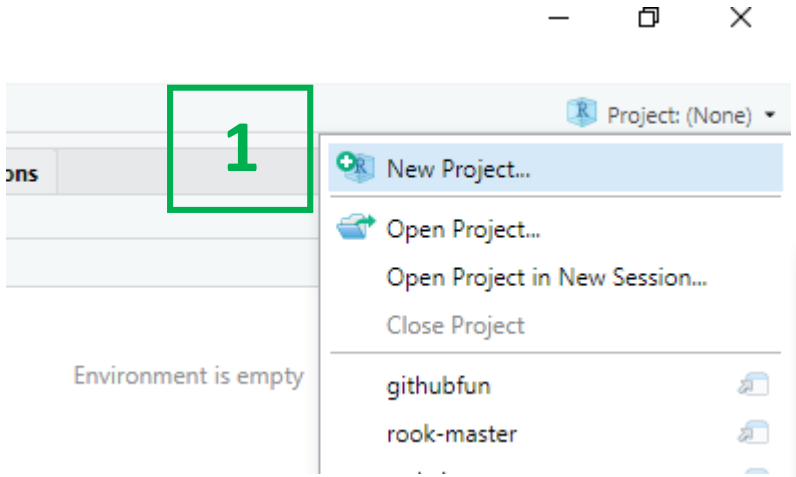
I don’t know all your GitHub user names, so I can’t add all of you immediately. So this is what you can do:

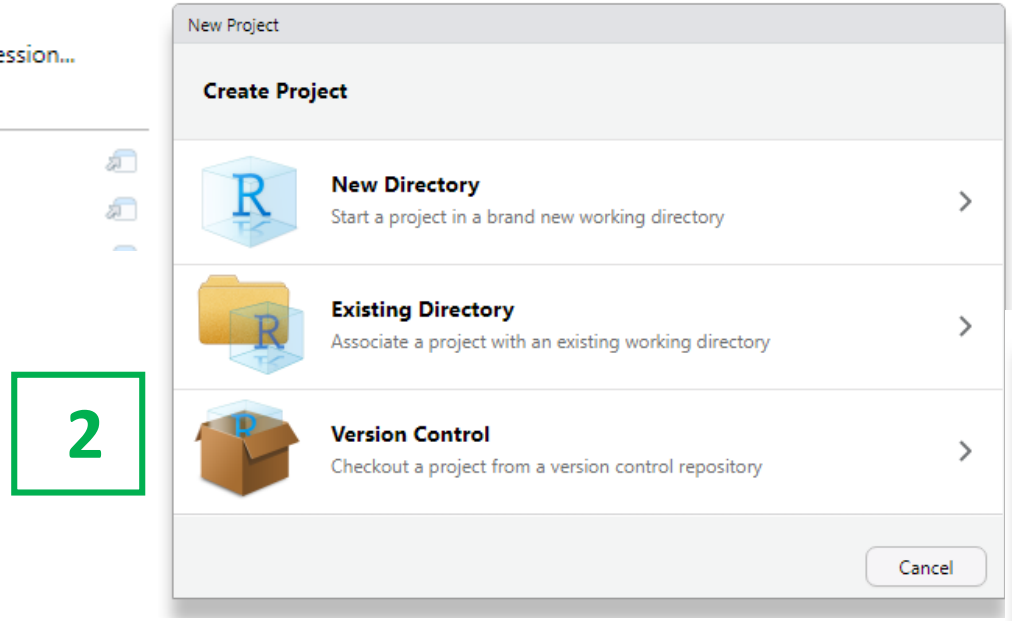
- Go to the repository page: https://github.com/anamartinovici/githubfun
- Click on Issues and create a new one. After all, it is an issue if you can’t contribute to this repository.
- Then, I will add you as a collaborator ☺

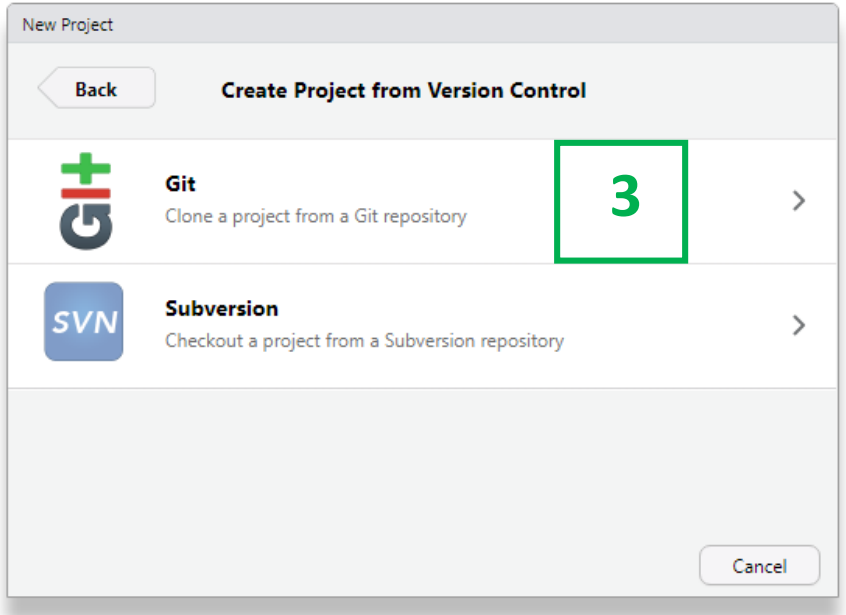
On GitHub.com: copy this link



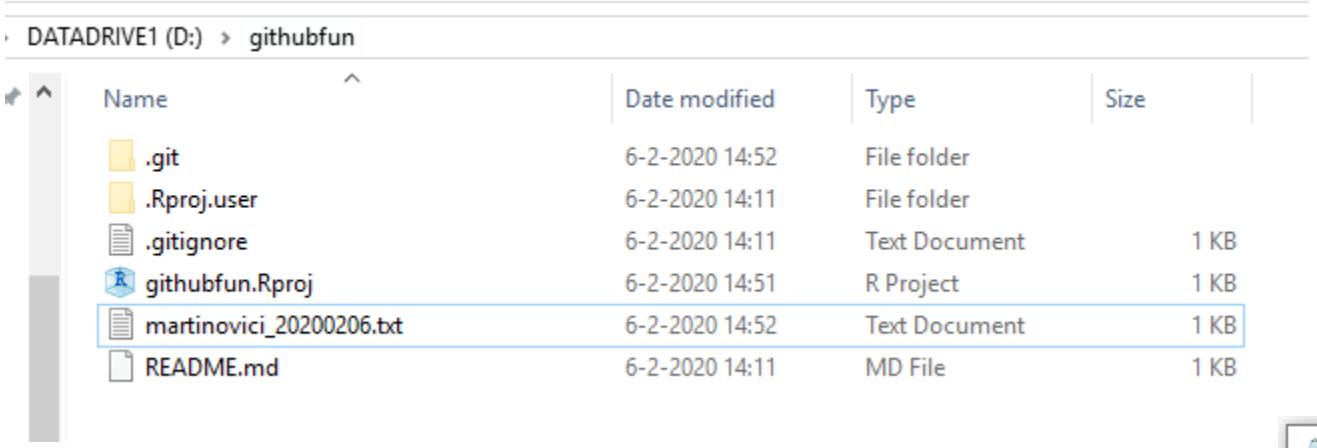
In Rstudio: new version control project (git)

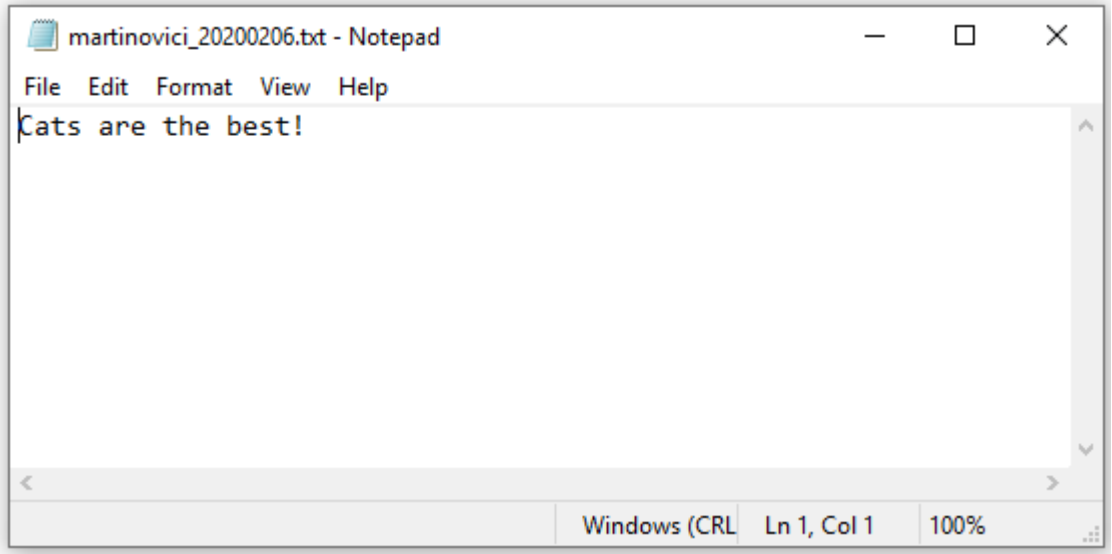




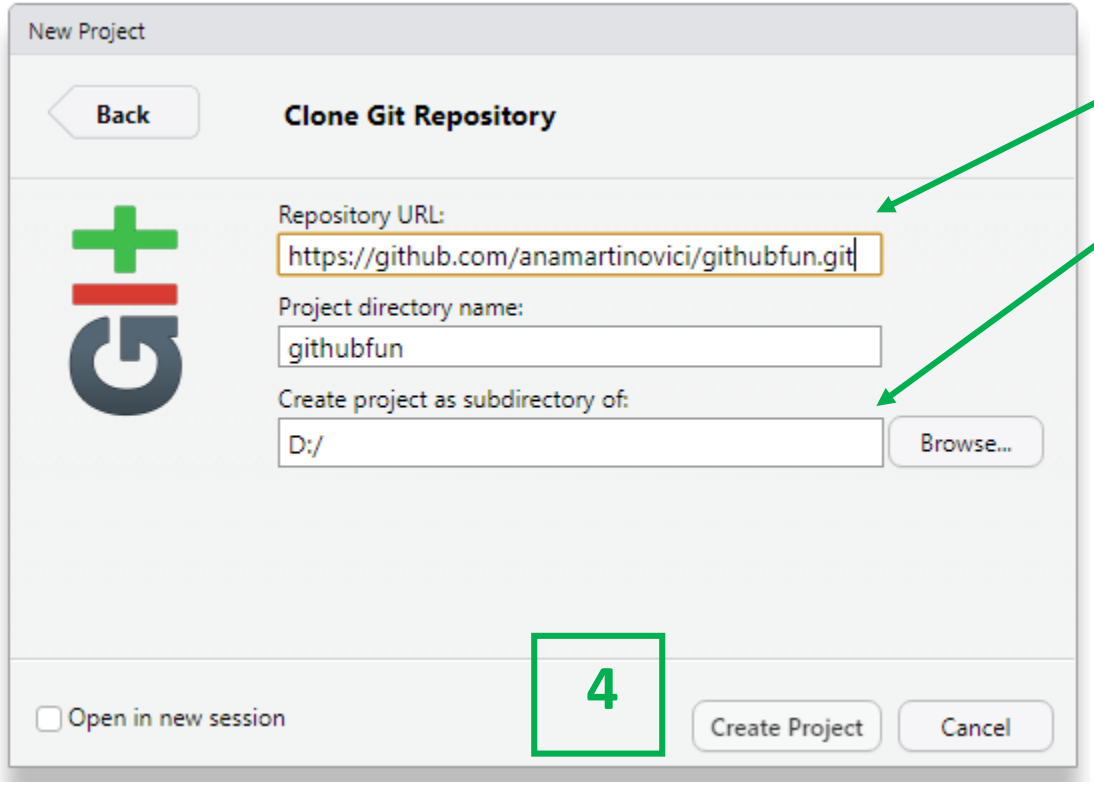


On your laptop/PC: create a file with your name





In Rstudio: new version control project (git)

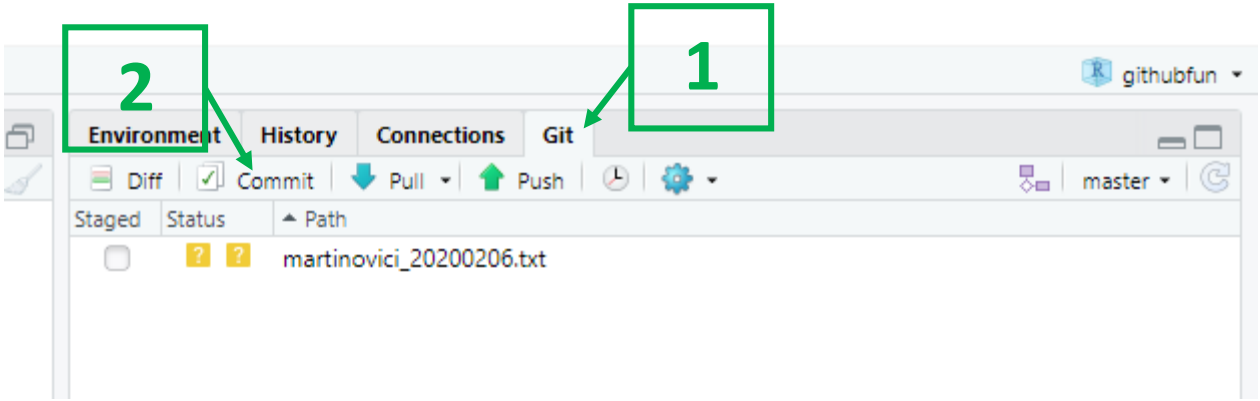


- Paste the link you've previously copied

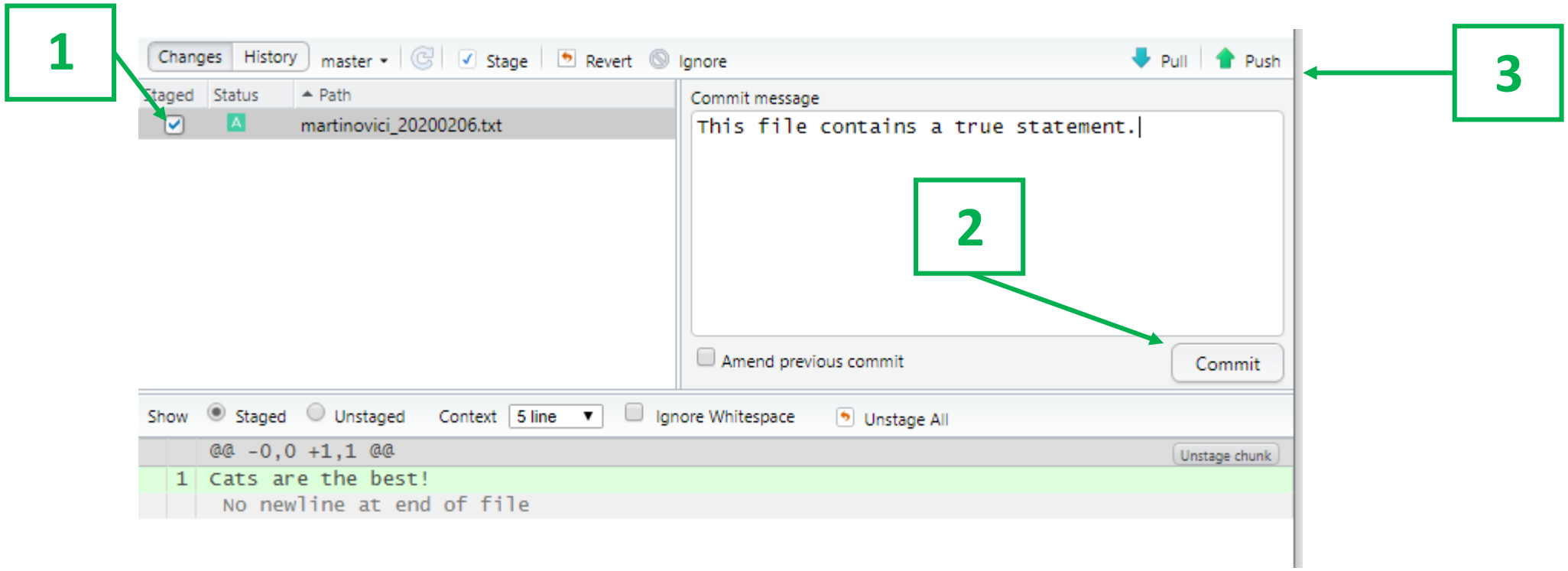
IMPORTANT:

- If you use a device from the university (laptop, PC), create the project on a local drive (C:/ or D:/). Don't create it on what appears to be the desktop ("CLsomenumbersandletters/Desktop"), as that's in fact a network location.
- If you use your own device, then you can save it on Desktop.
- Regardless of which device you use, do NOT save a repository in Dropbox. This will create sync problems that are better avoided. Dropbox and GitHub are useful, but for different purposes.

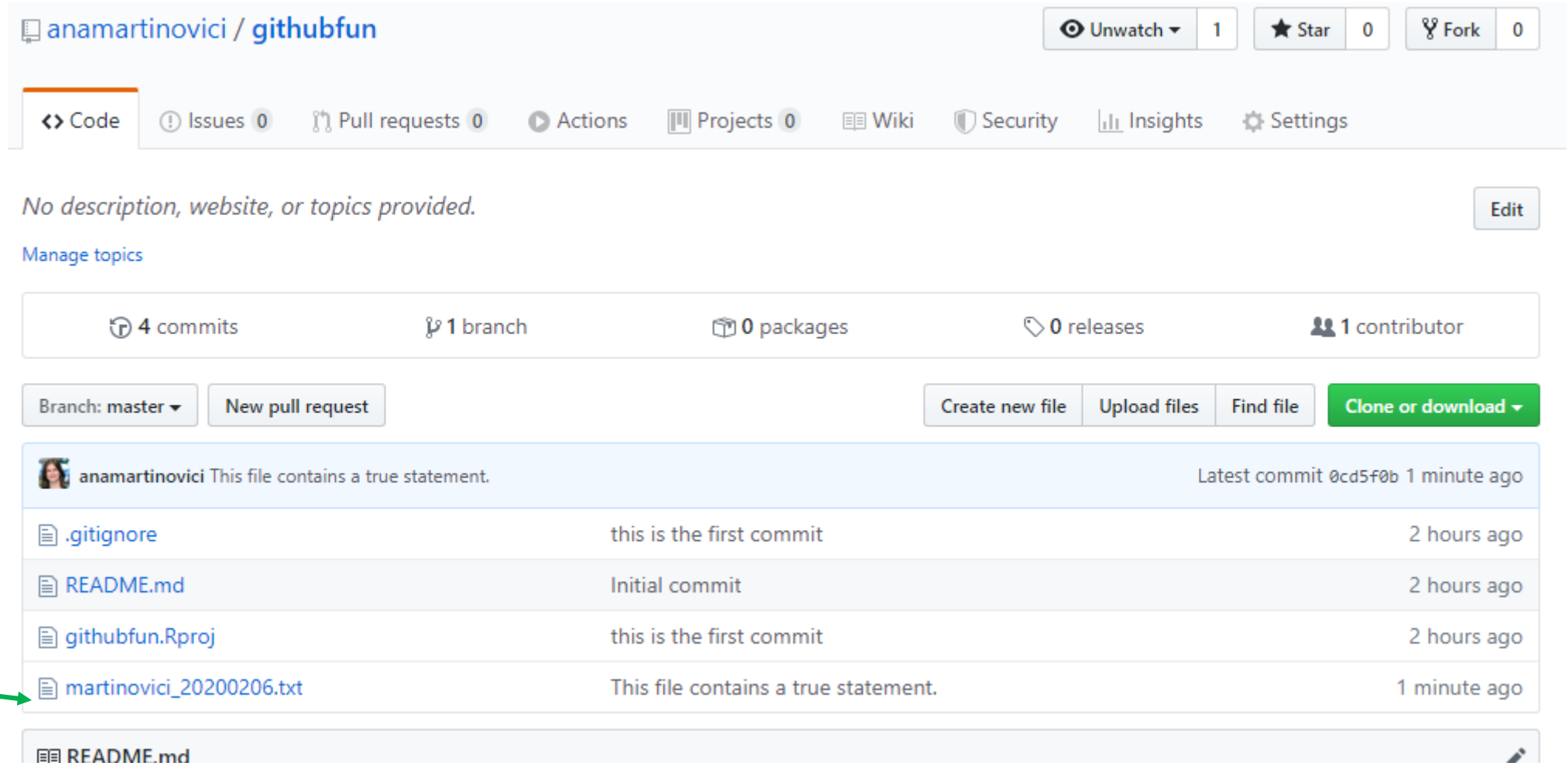
In Rstudio: open the commit window



In Rstudio: stage -> commit -> push



On GitHub: refresh the page for “githubfun”



New file

References and additional resources

- https://github.com/DataScienceSpecialization/courses/blob/master/05_ReproducibleResearch/
- https://github.com/jasonmtroos/rook
- http://kbroman.org/Tools4RR/
- http://blogs.nature.com/naturejobs/2018/06/11/git-the-reproducibility-tool-scientists-love-to-hate/
- https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1004668
- https://swcarpentry.github.io/git-novice/
- https://jules32.github.io/2016-07-12-Oxford/git/
- https://desktop.github.com/