Using git

⚔

Tales of Peril, Pain and Protection

Julia Piaskowski

2022/11/08

https://jpiaskowski.gitlab.io/talks/git-asa-cssa-sssa-2022/

What Do I Use git & GitHub For?

1. Share and access data

2. Collaborate

3. Build things

GitHub Octocat

Share & Access Data/Software/Demos

Tidy Tuesday datasets

Share & Access Data/Software/Demos

Hugging Face Deep Learning Tools

Fork (Copy) a Repository

Course Template

Collaborate: File Issues

- Find out if someone else has experience a problem similar to yours
- Notify developers of a software bug
- Request a new feature

Issue tab for r-lib/actions

Collaborate With Others

Collaboration is the most compelling reason to manage a project with Git and GitHub. My definition of collaboration includes hands-on participation by multiple people, including your past and future self, as well as an asymmetric model, in which some people are active makers and others only read or review.

J. Bryan, *Peer J* (2017)

git vs. GitHub
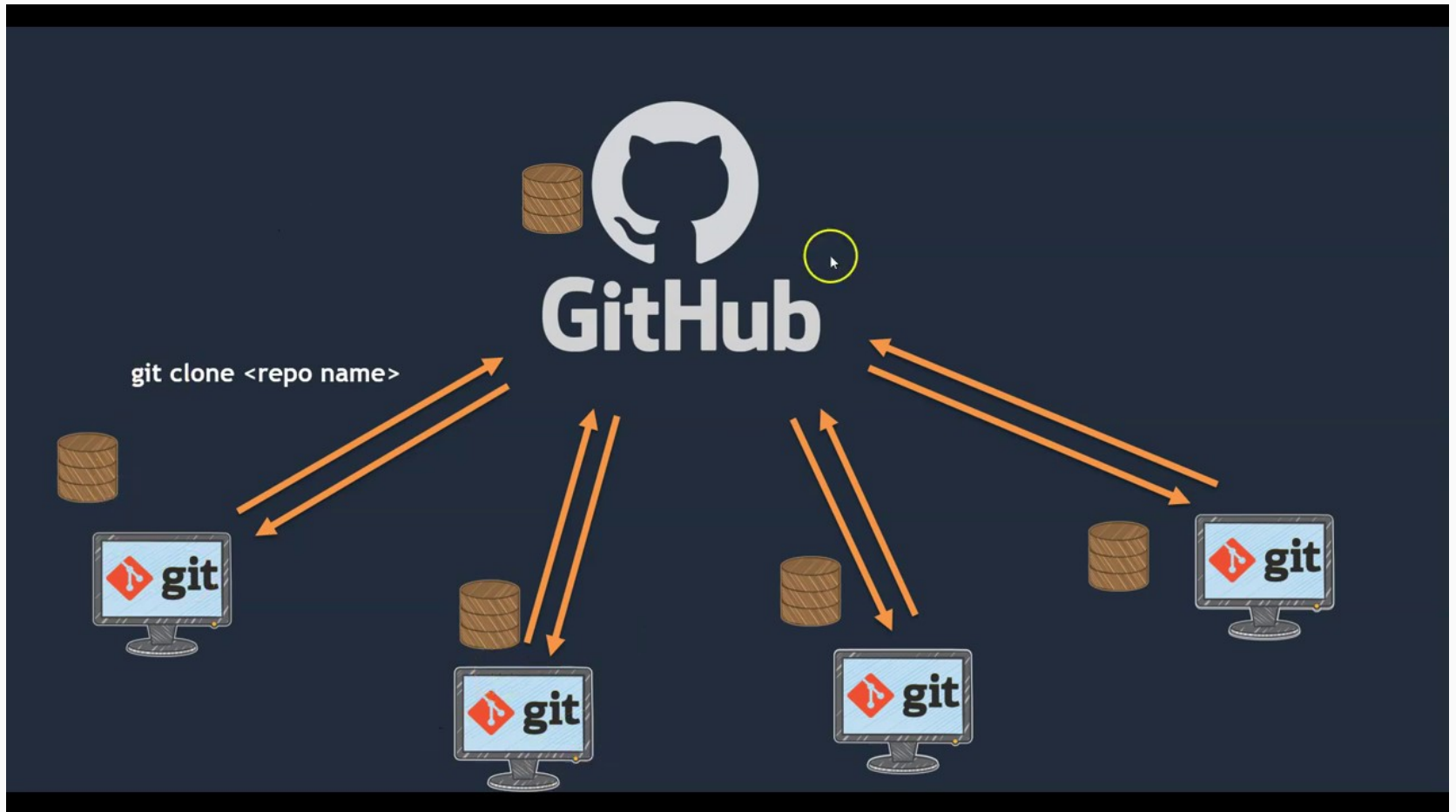
Version Control

Version control captures changes across files, describing the difference between each and maintaining a linear history that can be rewound if needed.

Why Might You Need This?

To clarify your data curation and analysis process for *research reproducibility*

- Files change over time

- Results may be shared

- Many people are contributing to a project

  > ...reproducibility is obtaining consistent results using the same input data; computational steps, methods, and code; and conditions of analysis.

  ```
  - Reproducibility and Replicability in Science (2018)
      National Academies of Sciences
  ```

Reference

Why Might You Need This?

You may be already implementing ad hoc version control.

tidy_apple_phenotypic_data.csv

tidy_apple_phenotypic_data2.csv

tidy_apple_phenotypic_data3.csv

tidy_apple_phenotypic_data4.csv

tidy_apple_phenotypic_data5.csv

tidy_apple_phenotypice_data5_SV.CSV

name_conversion_2017Mar7.csv

tidy_apple_phenotypice_data6.CSV

tidy_apple_phenotypice_data7.CSV

What It Could Look Like
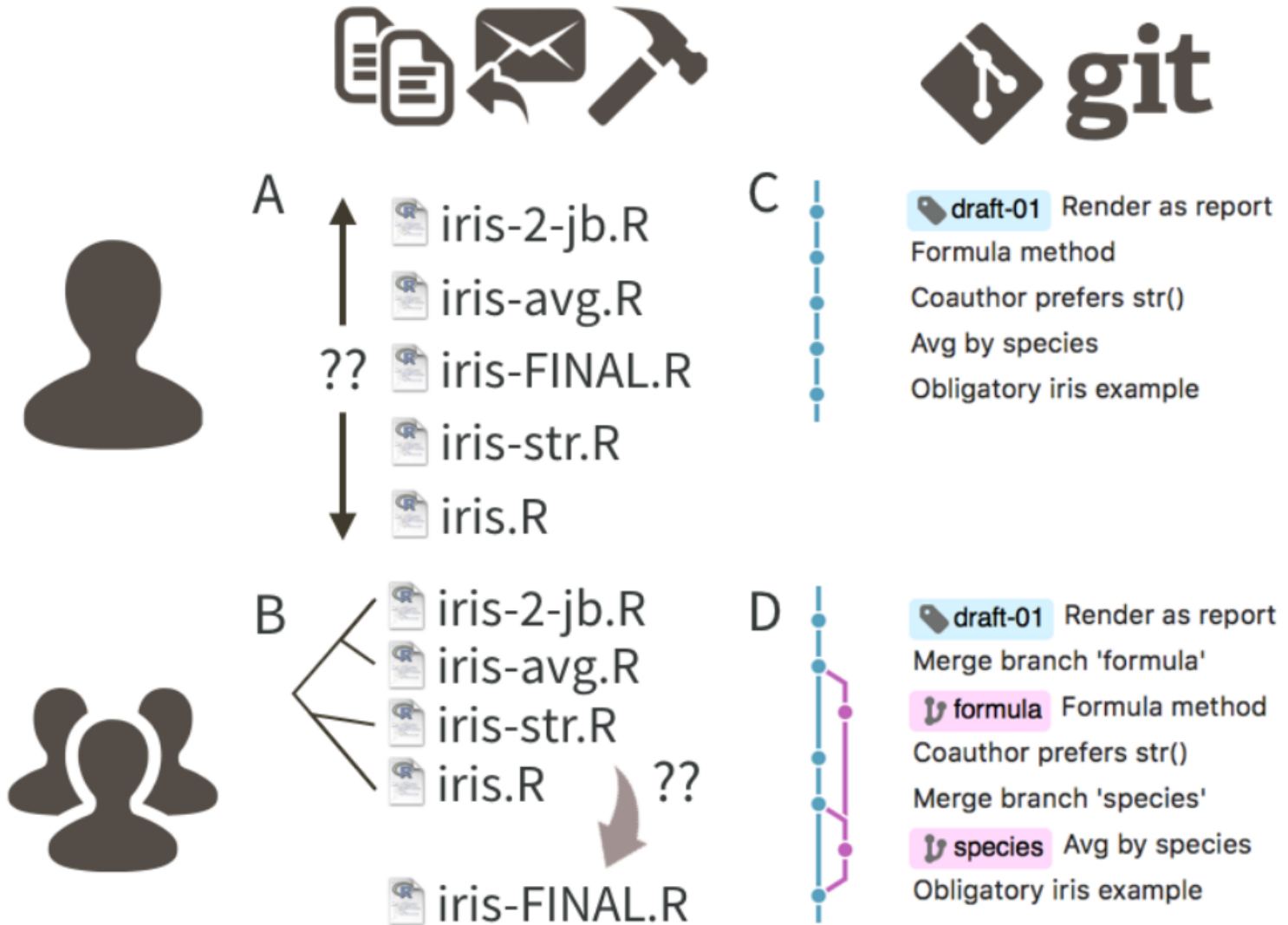

A

iris-2-jb.R
iris-avg.R
?? iris-FINAL.R
iris-str.R
iris.R

B

iris-2-jb.R
iris-avg.R
iris-str.R
iris.R   ??

iris-FINAL.R

C

draft-01 Render as report
Formula method
Coauthor prefers str()
Avg by species
Obligatory iris example

D

draft-01 Render as report
Merge branch 'formula'
formula Formula method
Coauthor prefers str()
Merge branch 'species'
species Avg by species
Obligatory iris example

Version Control Advantages

- Sharing for asynchronous work - no more "data_final_KC_EP_JLP_BP.csv" or wondering if you have the most recent version of files

- Abundantly clear what is the central or main version of files

- Very advanced version of "track changes"

- Commits create safety points in case of disaster - you can recover previous version

- Branching creates space for experimentation

Generate a Web Presence

Post a Tutorial as an Online Book
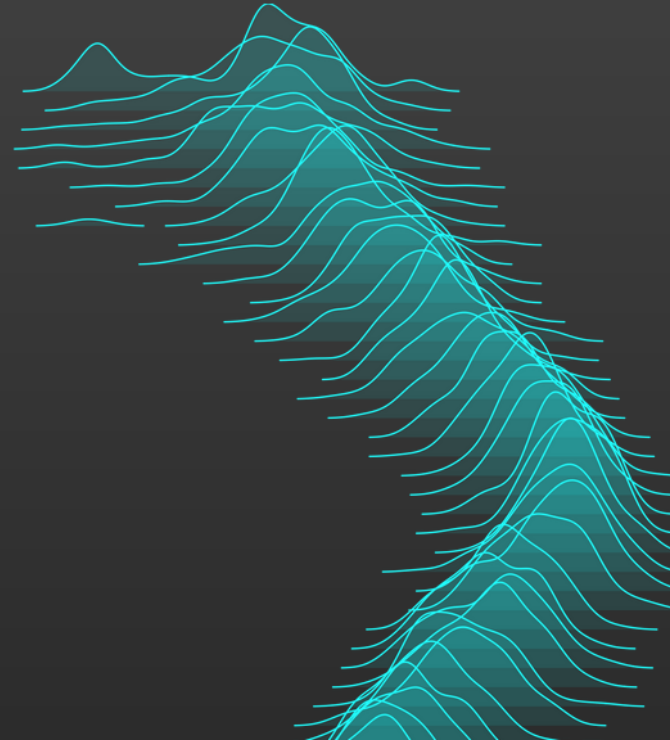
Happy Git and GitHub for the useR

Search

On this page

Let's Git started

License

View source

Edit this page

Still from Heaven King video

Happy Git provides opinionated instructions on how to:

- Install Git and get it working smoothly with GitHub, in the shell and in the RStudio IDE.

Happy Git with R | GitHub

Deploy Course Website

Documentation for Software

Automate Tasks

- Automate action: `R cmd check` or `blogdown::build_site()`
- Run a GitHub action to do CI/CD (continuous integration/continuous deployment)
- GitHUb actions can be challenging:

What Git and GitHub Are *Not* Suitable For

- Making publication data sets available - this is not an appropriate long-term repository (just ask the National Academy of Sciences!)

- Tracking changes in binary files (.doc, .docx, .xls, .xlsx, .pdf)

There are special tools to work with large files - make sure you use those if your files are large!

Things Can Go Wrong With Version Control

Sith Lord		Wipe out someone else's commits by force pushing to the main branch.
Procrastinator		Created a repository with a single "init" commit with just a README.md. Then never touch it for 5 years.
Secret Santa		Accidentally commit a secret API key in a public repository.

Snark GitHub Badges

Lesson 1: Take Learning git Seriously

THIS IS GIT. IT TRACKS COLLABORATIVE WORK ON PROJECTS THROUGH A BEAUTIFUL DISTRIBUTED GRAPH THEORY TREE MODEL.

COOL. HOW DO WE USE IT?

NO IDEA. JUST MEMORIZE THESE SHELL COMMANDS AND TYPE THEM TO SYNC UP. IF YOU GET ERRORS, SAVE YOUR WORK ELSEWHERE, DELETE THE PROJECT, AND DOWNLOAD A FRESH COPY.

Decent resources for self study:

- Happy Git with R

- Software Carpentry git workshop

- Thee book of git

xkcd cartoon

Lesson 2: Use a git Client

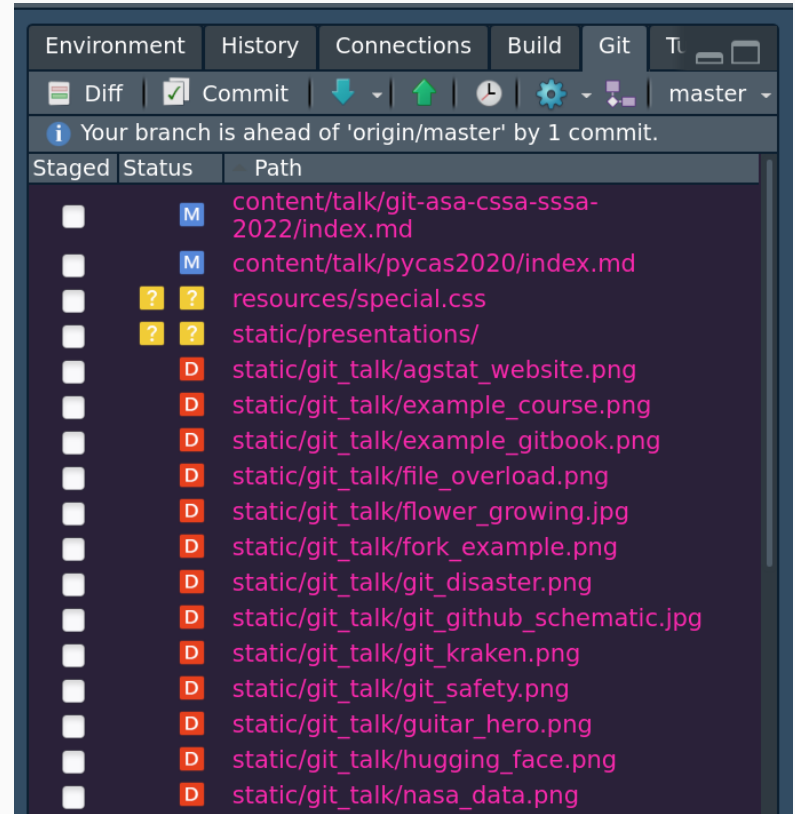(A graphical user interface)

Some Graphical User Interfaces

- GitHub GUI

- Git Kraken

- SourceTree

- ....tons more

The goal is to become accustomed to using git regularly - use the tools that help you reach that

Lesson 3: Be Patient When an Error Occurs

When the inevitable error happens:

- a merge error

- can't pull or push!

- a `git revert` gone horribly wrong

Proceed **cautiously** and diagnose the problem

Lesson 4: Don't Expose Secrets

- Learn about and use the `.gitignore` file

- Consider private repositories when appropriate

- There are some guidelines regarding legal compliance (e.g. HIPAA)
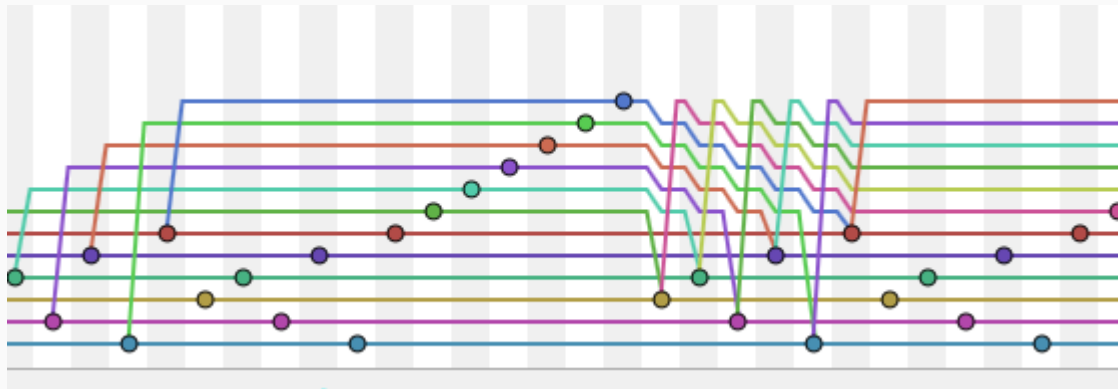
git is a Humbling Experience

ag stats stan
@SeedsAndBreeds

If you you have not experienced a git disaster, have you really used git?

(I still think git is worth the trade-offs)

original tweet