RStudio + Git (+ GitHub) for Collaboration and Reproducibility

...

Let's Git R-ganized!

Dan Kerchner
GW Libraries & Academic Innovation
November 15, 2019

What this workshop is *not*

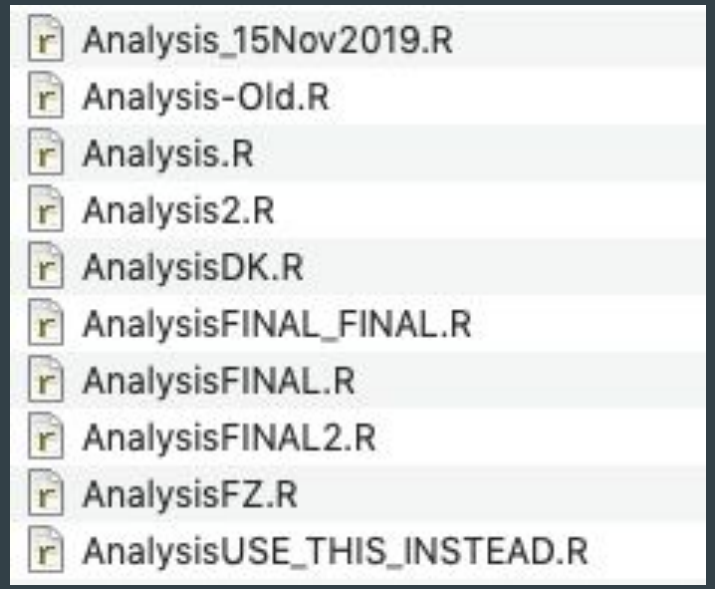
- R or RStudio for beginners
- R for Public Health Applications

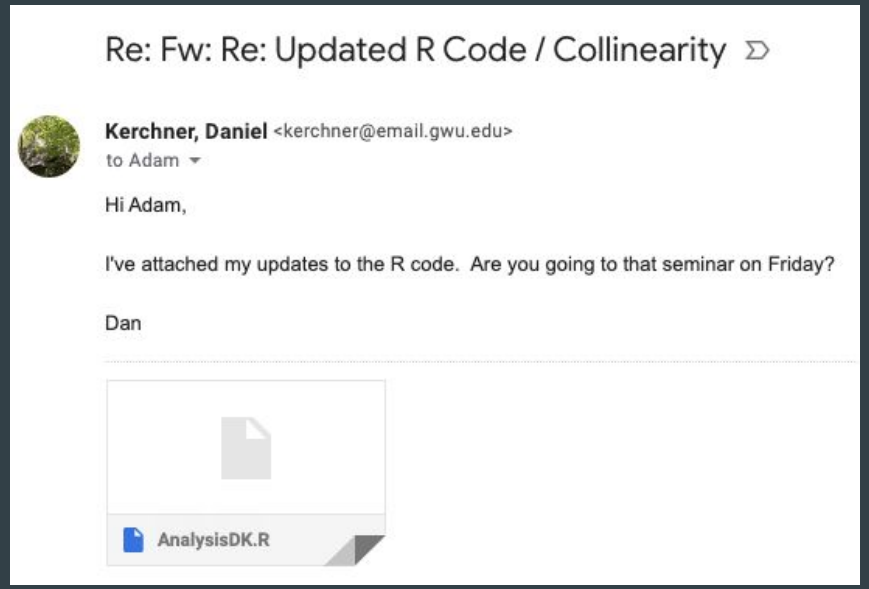
Agenda

- Motivation
- Hands-on Workshop
- Additional topics

Thoughts on current workflows

DIY Version Control and Collaboration





Hands-On Workshop

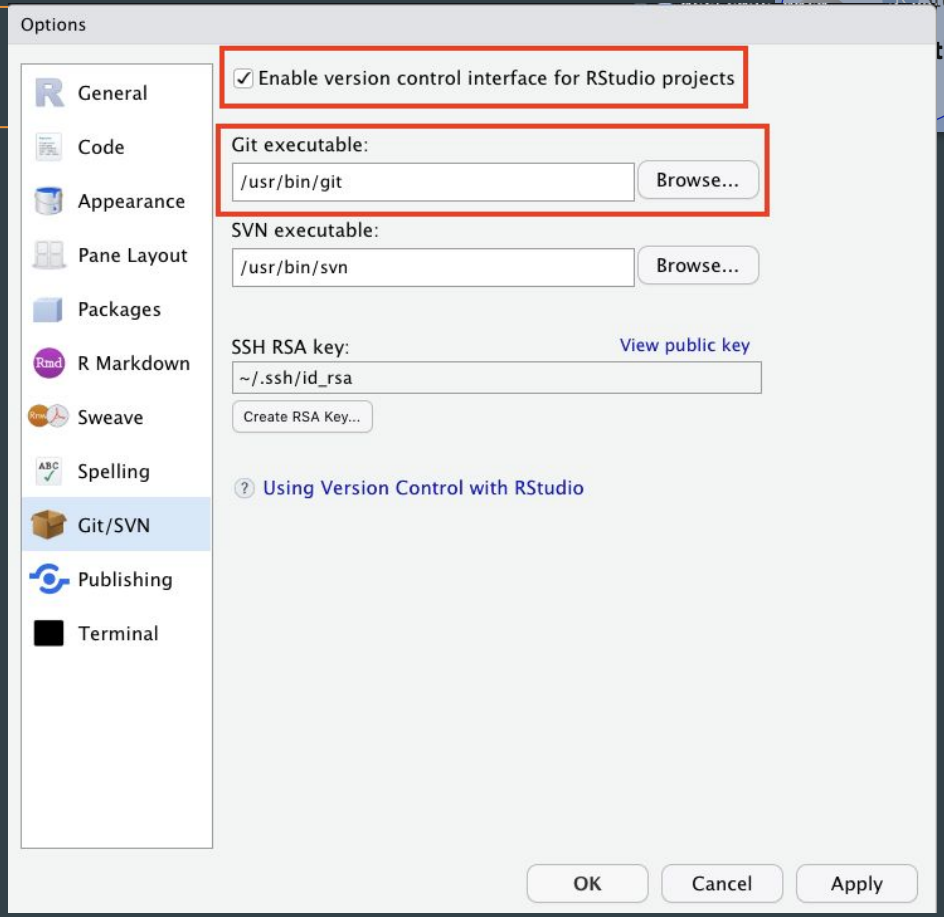
Prerequisite: Create Github account

To do later:

Go to **education.github.com** and sign up for extra free benefits!

Step 0. Verify Setup

Tools → Options → Git/SVN



Set git configurations:

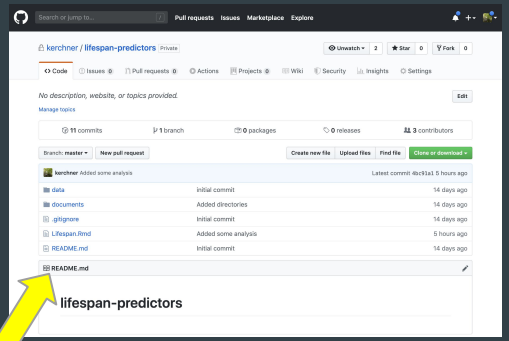
In the RStudio "Terminal" tab (next to Console)

(These should match your Github account)

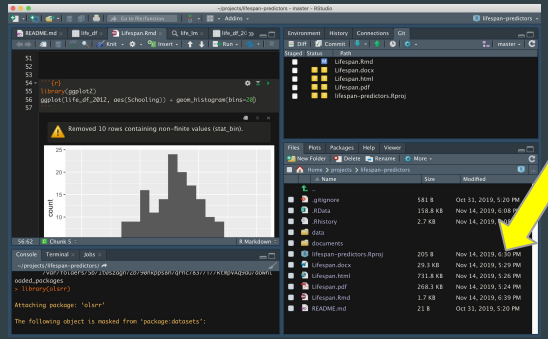
```
git config --global user.email "you@youremail.com"
git config --global user.name "Your Name"
```

Mental Model: One-person* project

github.com/you/yourproject



Your Computer



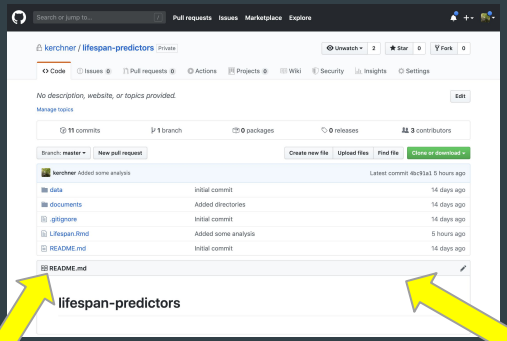
Workflow: One person

1. Github - create empty repository
2. RStudio - clone. Stop and note new controls in RStudio
3. RStudio - Make changes to R files (e.g. create an R script)
4. RStudio - Stage, Commit, Push to Github

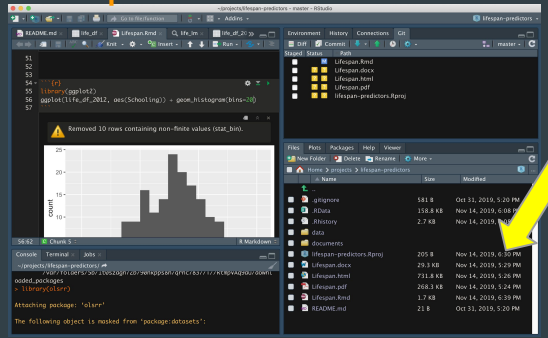
Collaboration

Mental model: Collaboration

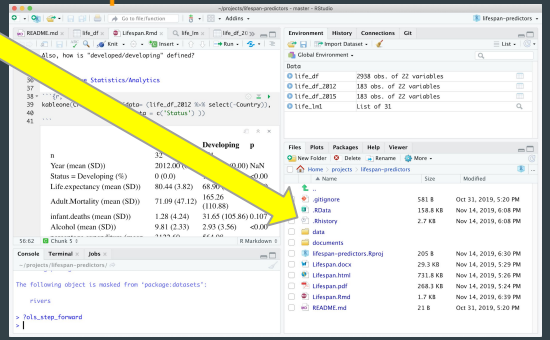
github.com/you/yourproject



Your Computer



Your Collaborator's Computer



The problem

"master copy"

```
# some code
alpha <- 0.05
# some more code
```

```
# some code
alpha <- 0.01
# some more code
```

??

copy

```
# some code
alpha <- 0.05
# some more code
```

change

```
# some code
alpha <- 0.01
# some more code
```

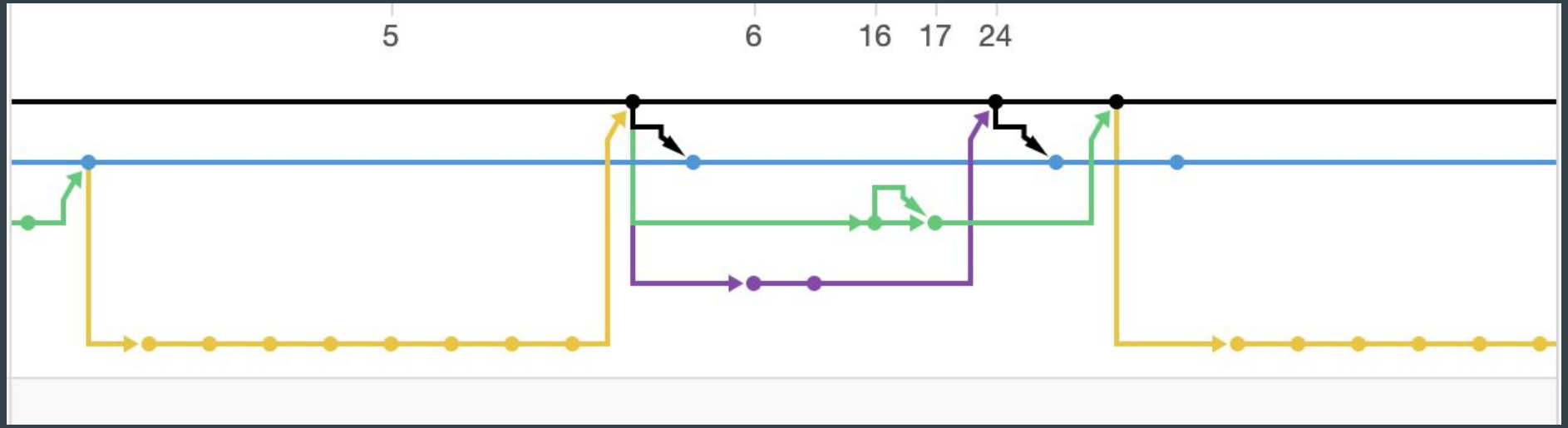
copy

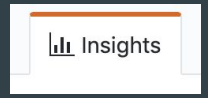
```
# some code
alpha <- 0.05
# some more code
```

```
# some code
alpha <- 0.15
beta <- 1.2
# some more code
```

copy

Branches



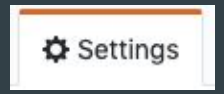


(see Insights → Network)

Workflow: Collaboration

Repository Owner:

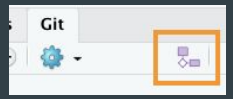
1. Github - Add collaborator(s)



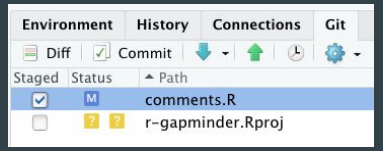
→

Collaborator(s):

2. RStudio - Clone repository
3. Create a new branch



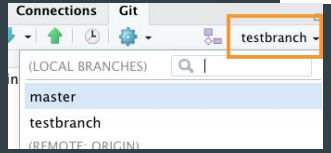
4. Make changes
5. Stage+commit changes



6. 

Push branch to Github
7. Github - Create pull request

8. RStudio - Pull branches
9. RStudio - Switch to the new branch; Test the code



10. Github - Merge pull request

11. RStudio - Switch to master branch. Pull branches.

Git terminology

Stage - put changes in a staging area (to be committed)

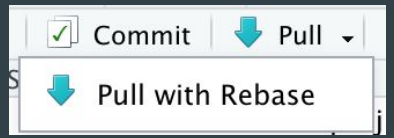
Commit - commit staged changes to the [local] repository

Push - push local repository changes to the remote repository

Pull - pull remote repository changes to the local repository

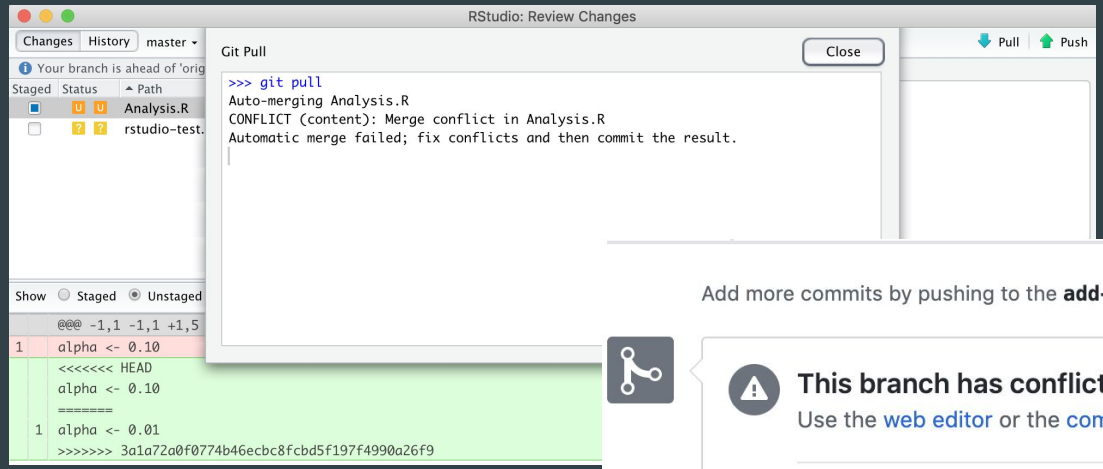
More

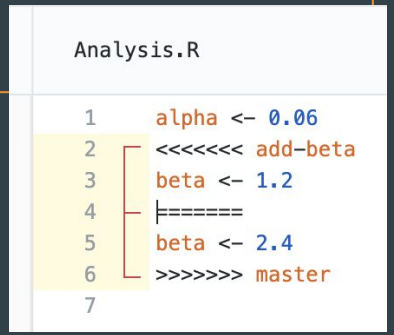
- History
- Diff
- `.gitignore`
- Reverting
- Code releases ← needed for Zenodo
- Rebasing/Merging

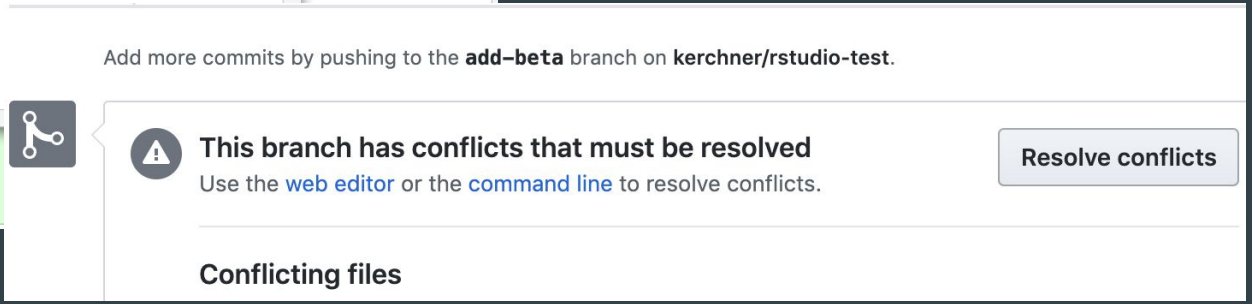


- Forking
- Merge conflicts...

Merge conflicts



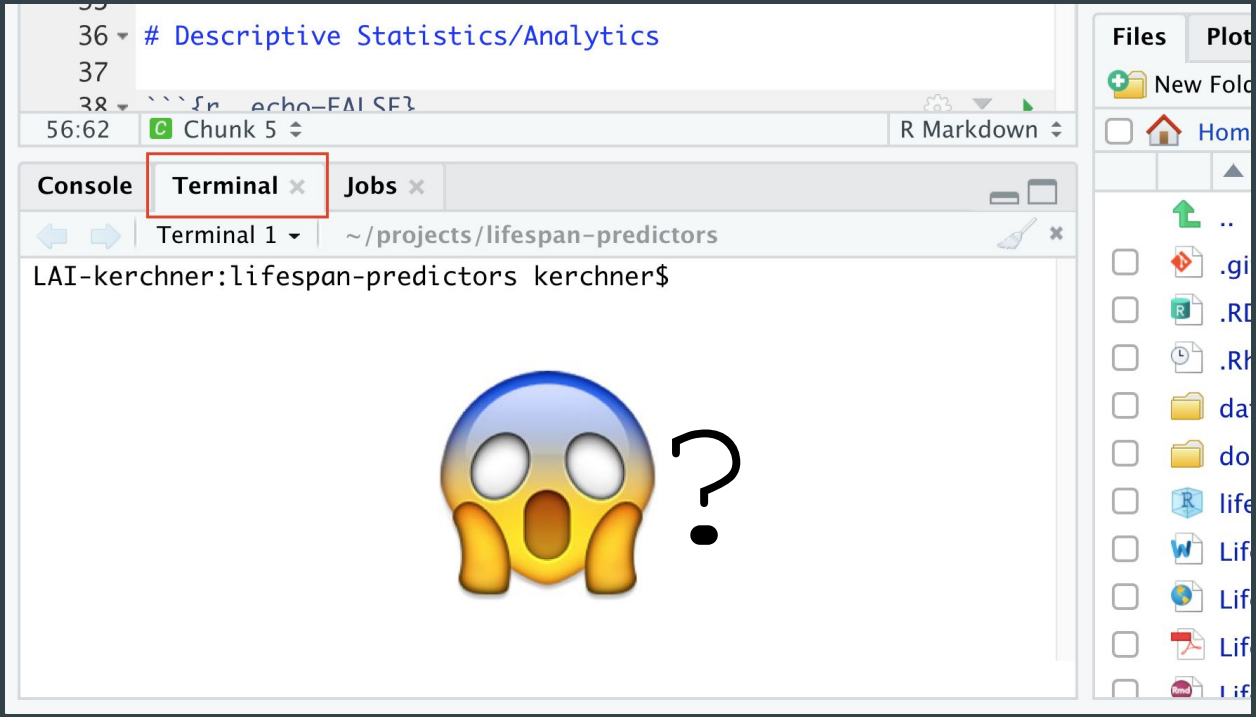




Breathe :-) Follow the instructions in the message.

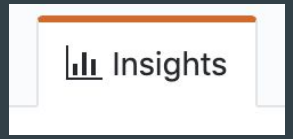
Manually merge/fix the file. Re-stage, commit, push.

Let's talk about the Terminal



Project organization

- Github Issues & Milestones
- Github visualizations



Tour of a Repository: About your code

- **README.md** ← showcase your work, this is your home page
- **LICENSE** file ← Important! choosealicense.com
- (Optional) Wiki pages

These use Markdown - lightweight formatting

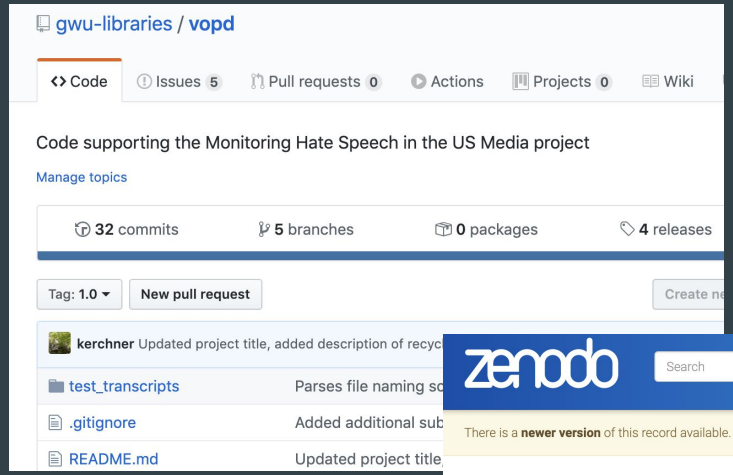
Tour of a Repository - The Code

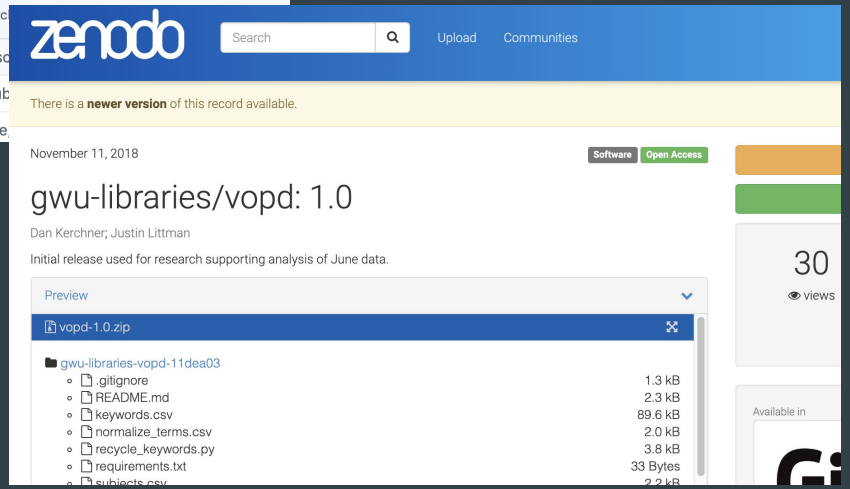
- Code
  - history of commits on a branch
  - history of commits on a file
- Branches
  - Master branch vs. others
  - Visually represented with the Insights → Network graph

Tour of a Repository - Code as a Project

- Issues - tags, assignment, releases, comments
- Milestones and Releases
- Pull requests
- Comment threads - on issues, commits, etc.
- Insights (graphs)

Your code, a "publication"

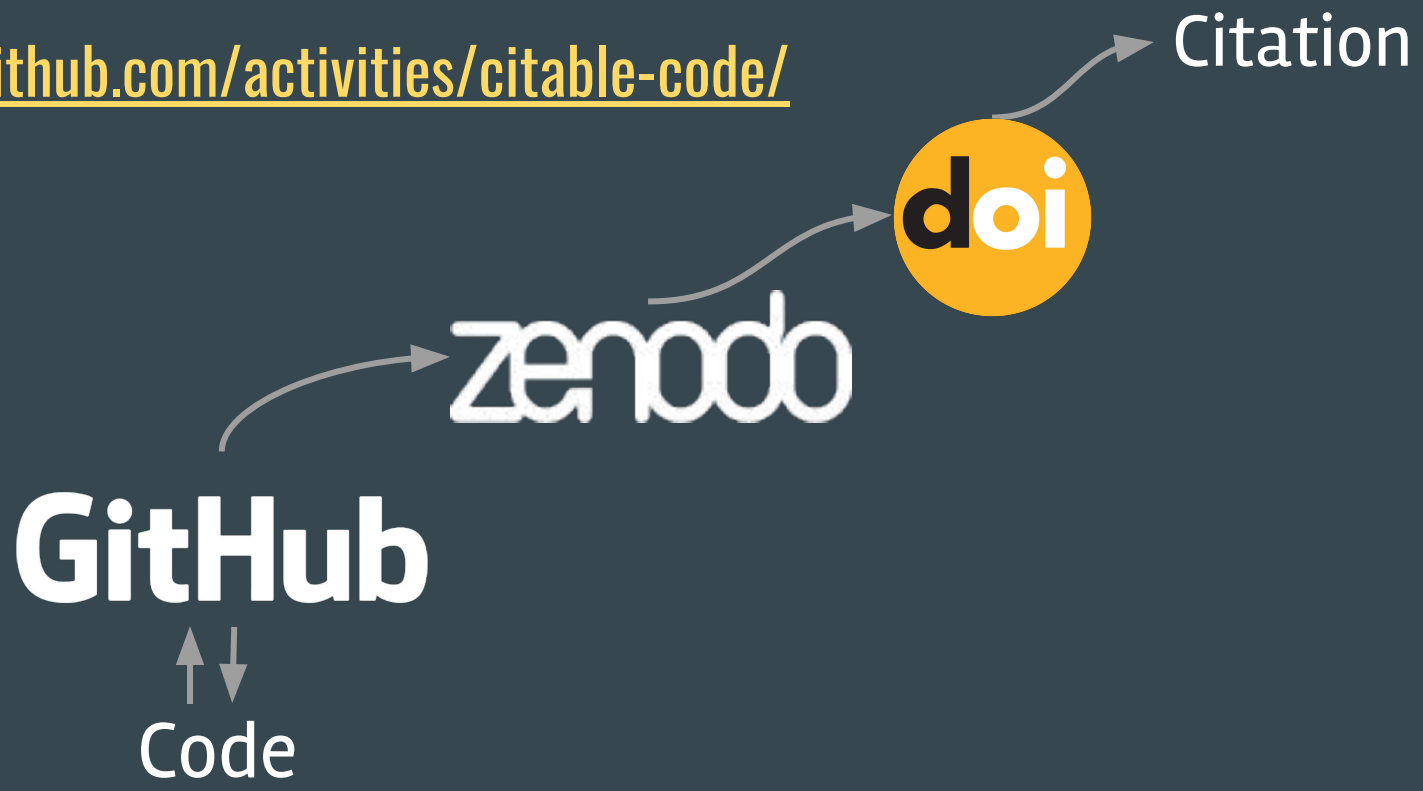




research shows that public sentiment towards
(Brewer, Joseph and Willnat 2003). As such, e
a country by mentioning them in a negative wa
building negative public sentiment. It is import
data when the country or its people (Russia, Ru
its leadership was mentioned (e.g. Russian gov
[2] While the data on daily viewers/listeners is mo
to know exact daily viewers for YouTube show
left (The Young Turks) and right (Alex Jones'
study, but removed from YouTube on August
[3] The authors wish to attribute credit to Justin Li
of the code, including much of the core logic u
[4] https://github.com/pdfminer/pdfminer.six
[5] DOI: 10.5281/zenodo.1482912

Your code as: citeable, durable, publishable

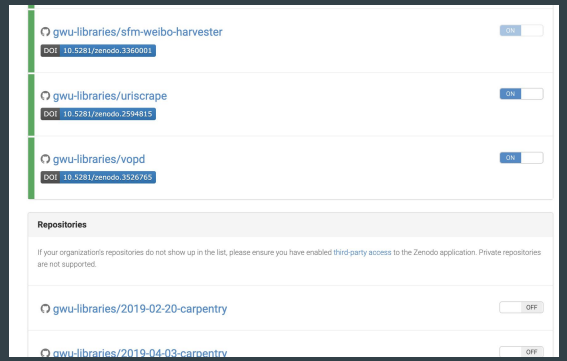
guides.github.com/activities/citable-code/



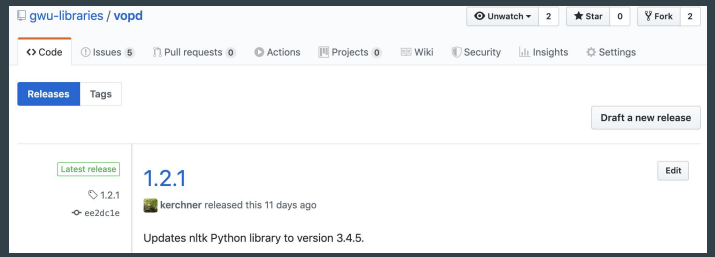
How to make your code citeable via Zenodo

1. Link your Github account to Zenodo

2. Enable your Github project in Zenodo



3. Create releases (tags) in Github (e.g. 1.0, 1.1..., 1.2.5, etc.)



What makes software a publication?

- Accessible
  - Available
- Citable
  - Title; Author(s); Version; Unique Persistent Identifier;
- Usable
  - Documented
  - Commented; Openly Licensed; Dependencies Outlined

Parting Thoughts

Advice

- Create your repository on Github first. Then clone it to RStudio. Then add files.
- Try RStudio+Git on your next coding project -- even if you're working alone. You might like it.
- Set up a Github organization for your team/department/school, to host many projects
- If you're stuck or just want feedback on your R/Git workflow:
  - Don't hesitate to reach out to me or my colleagues

Beyond RStudio

- Python user? ✅ Python + Git
- There are Git desktop apps
- There are other code versioning protocols: SVN
- There are other Git repositories: BitBucket, GitLab

- Consider learning to use the Terminal (bash shell)

Retrospective: Motivation

- Code versioning
- Collaboration
- Code Cite-ability
- Reproducibility*

* Same code + Same data = Same results
(prerequisites: code + data are accessible)

Resources & Recommended Reading

- git-scm.com - heavy on command-line
- Using version control with RStudio - rstudio.com
- Happy Git and Github for the useR - Jenny Bryan, UBC, author of Excuse me, do you have a moment to talk about version control? (2017)
- Good Enough Practices in Scientific Computing (2017)
  (sequel to Best Practices in Scientific Computing (2014))

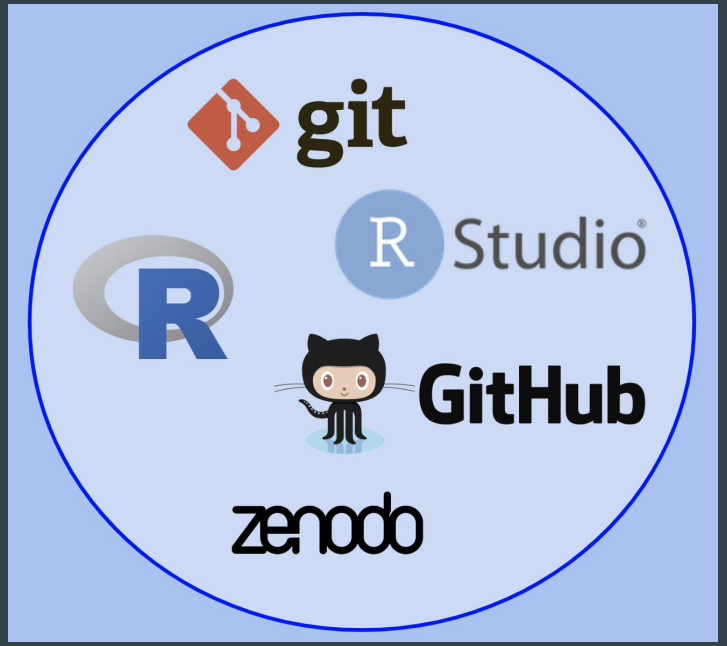
Questions / Discussion

Need help with git, coding, feedback... ?

Contact us!

kerchner@gwu.edu

STG team consultations → calendly.com/gwul-coding



Thanks!