Collaborative Data Analysis

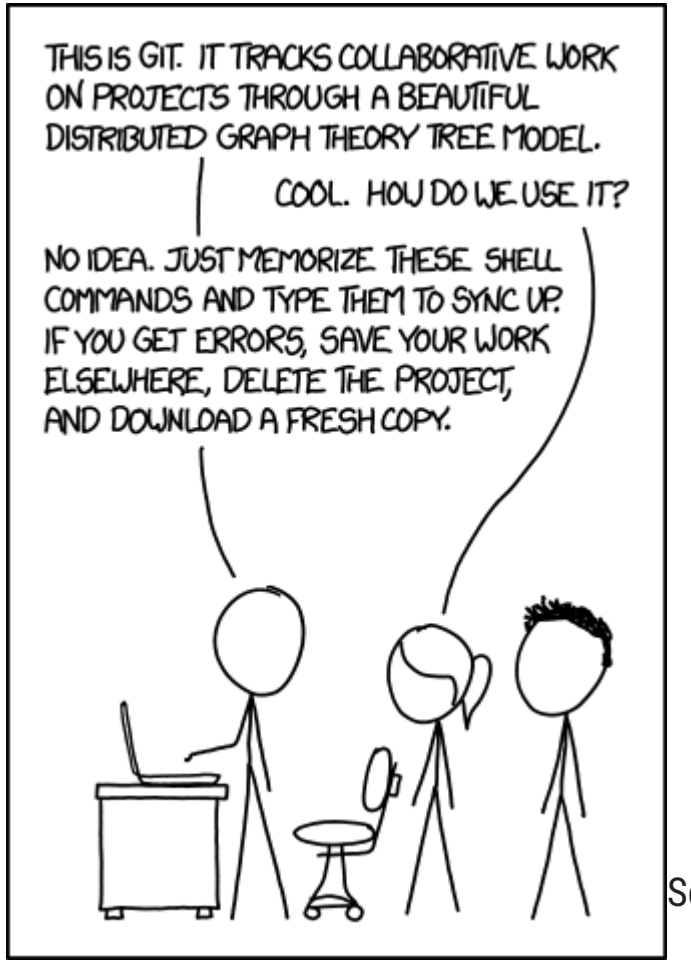
using Git & GitHub

Cristóbal Moya

July 27 2020

Contents

- Collaborative data analysis
- Git for data analysis
  - Version control systems
  - Git vocabulary
  - Basic Git functions
- GitHub for collaborative work
  - Isn't Git enough?
  - Basic collaborating functions
- Hands-on example



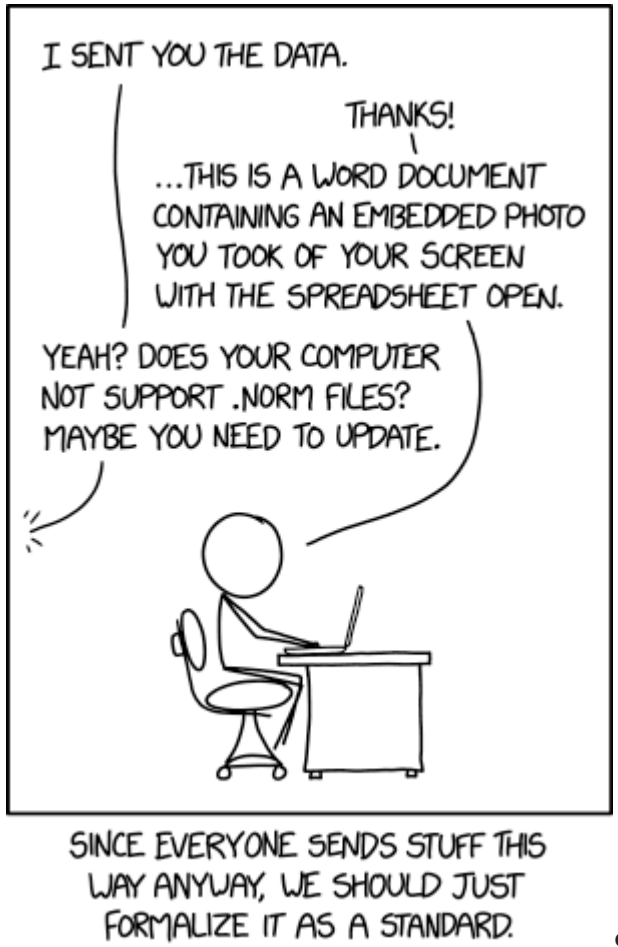
Source: xkcd

Collaborative data analysis

Collaborative data analysis

Implies solving at least the following issues:

- Centralize all operations and files
- Track all relevant procedures behind the analysis
- Make available the same copy of the project to any collaborator
- Create a workflow where collaborators can work at the same time handling conflicts
- Allow upscaling of the project
- Foster reproducibility of the project



Source: xkcd

Git for data analysis

Version control systems

Git

> Git is a distributed version-control system for tracking changes in source code during software development

Wikipedia

Oh wait, I'm not a software developer 🤨

> Git has been re-purposed by the data science community. In addition to using it for source code, we use it to manage the motley collection of files that make up typical data analytical projects, which often consist of data, figures, reports, and, yes, source code.

Happy Git with R

Who will your collaborators be?

- The first problem to solve is *how to collaborate with **yourself***
- Even working in pairs, things can get ugly without an **ordered process**
- Projects may upscale in unexpected ways:
  - New collaborators
  - Reproducibility of results
  - Crowdsourced replications
- Downside: your collaborators also need to know Git & GitHub 🙄
  - Many tools for making the collaboration easier (e.g., GitHub desktop, GitKraken)
  - Future standard?

What do you want to version control?

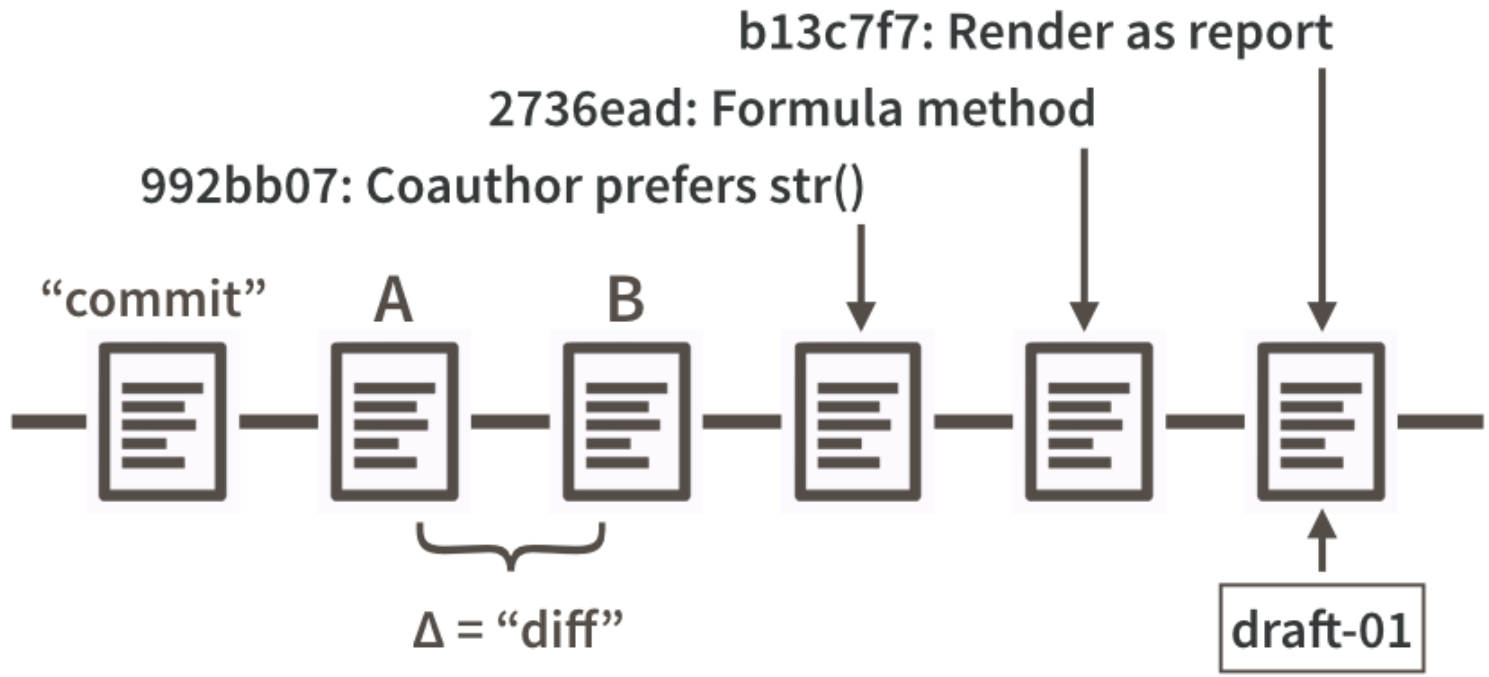
- It's important to assess what should your data analysis project track
  - Code, documentation, prose: ✔
  - Figures & tables: ✔
  - Data: ⚠
  - Software: ⚠

Git for data analysis

Git vocabulary

Vocabulary

- `commit`
  - Author
  - Message
  - Timestamp
- `repo`
- `diff`
- `tag`

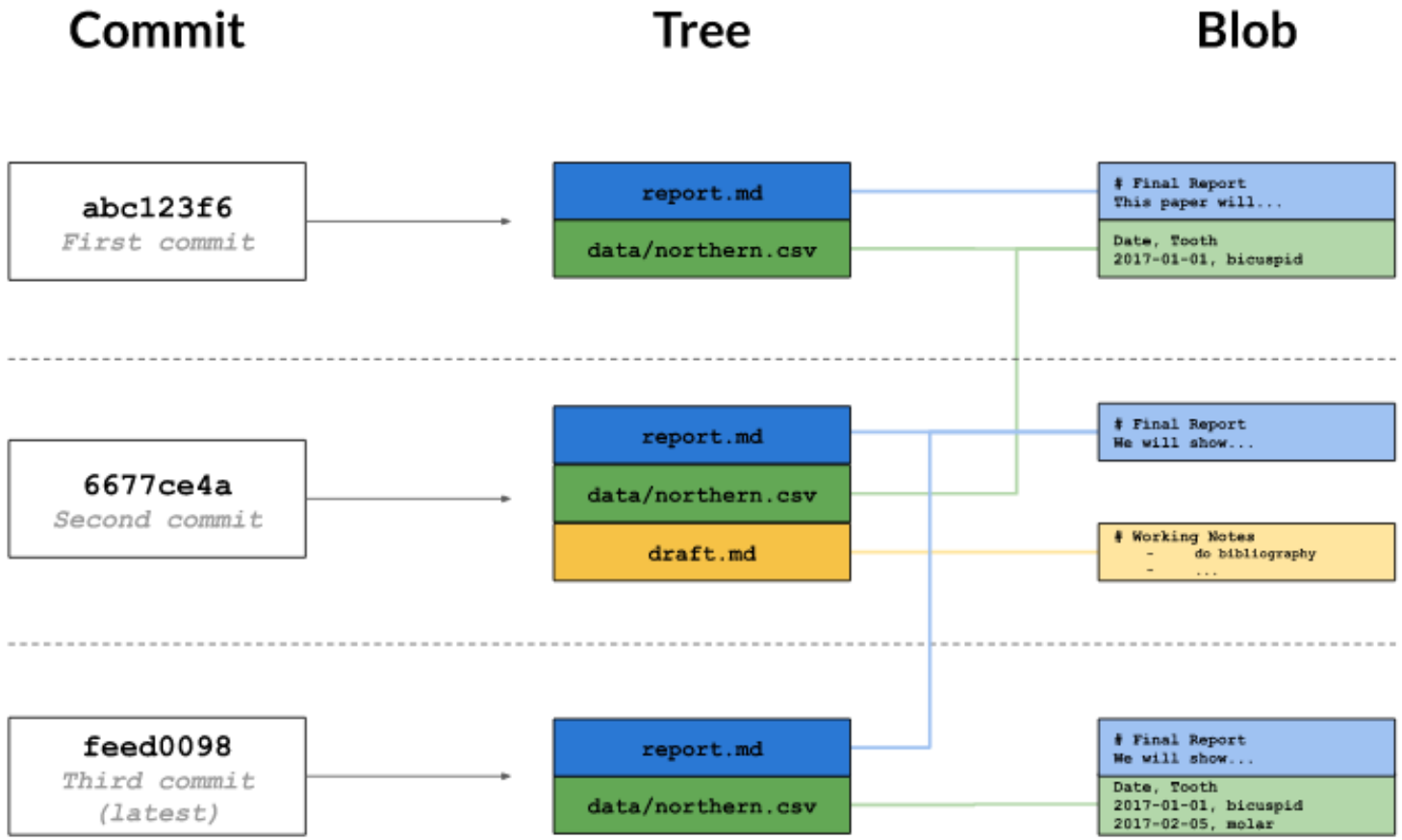


Source: Happy Git with R

System rationale

Under the hood

- Efficient register
- Change-focused



Source: Datacamp

Git for data analysis

Basic Git functions

Who am I?

- You have to be someone
  - Name
  - Email
- Everything in Git is a command

```
git config --list
```

```
## user.email=cristobalmoya@gmail.com
## user.name=Cristóbal Moya
## core.autocrlf=input
## core.repositoryformatversion=0
## core.filemode=true
## core.bare=false
## core.logallrefupdates=true
## remote.origin.url=https://github.com/Crismoc/collaborative_analysis.git
## remote.origin.fetch=+refs/heads/*:refs/remotes/origin/*
## branch.master.remote=origin
## branch.master.merge=refs/heads/master
```

Where am I?

- You're always in some repo
  - Local
  - Remote
  - Both
- There is a history
  - **Actual state**
  - Log

```
git status
```

```
## On branch master
## Your branch is up to date with 'origin/master'.
##
## nothing to commit, working tree clean
```

Where am I?

- You're always in some repo
  - Local
  - Remote
  - Both
- There is a history
  - Actual state
  - **Log**

```
git log
```

```
## commit 7131d4d015c344aba6791f2d6e5e5f598d8d30de
## Author: Cristóbal Moya <cristobalmoya@gmail.com>
## Date:   Mon Jul 27 03:04:58 2020 -0400
##
##     Add activity URL & present git log example
##
## commit 3f39e05ee16e926fed0511c451946f7344bf6b3c
## Author: Cristóbal Moya <cristobalmoya@gmail.com>
## Date:   Mon Jul 27 02:58:34 2020 -0400
##
##     Initial commit
```

Interfaces

- Originally, Git was designed for being used through the Command Line Interface
- Thankfully, there are many interfaces that can make Git tasks much easier, e.g:
  - GitHub Desktop
  - **GitKraken**
  - RStudio
  - Atom

GitHub for collaborative work

Isn't Git enough?

Isn't Git enough?

- In principle, Git just tracks your local machine
- Any collaboration or publication beyond your local machine will need to incorporate **remotes**, e.g:
  - GitHub
  - Gitlab
  - Bitbucket

GitHub for collaborative work

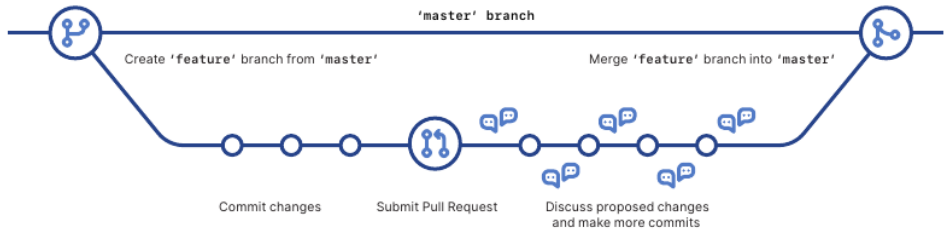
Basic collaborating functions

Fork & Pull model

Most common model where

- There is an *owner* or *project leader* with and **upstream repo**
- She/he assigns rights to *collaborators*
  - Every *collaborator* has a **fork** of the project
- Collaborators do not have **push** access to main (upstream) repo
- *Owner* accepts **Pull Requests** from collaborators, reviews them, then **merges** them into main repo

- Every *collaborator* develops major changes in **branches** (parallel universes), e.g:
  - Processing data
  - Specific analysis
  - Report or paper draft
- When finished, the branch is merged into the `master` branch



Source: GitHub cheatsheet

Does it worth it?

- Critique to an *American Economic Review* paper

Does High Public Debt Consistently Stifle Economic Growth? A Critique of Reinhart and Rogoff

Thomas Herndon* Michael Ash Robert Pollin

April 15, 2013

JEL CODES: E60, E62, E65

Abstract

We replicate Reinhart and Rogoff (2010a and 2010b) and find that coding errors, selective exclusion of available data, and unconventional weighting of summary statistics lead to serious errors that inaccurately represent the relationship between public debt and GDP growth among 20 advanced economies in the post-war period. Our finding is that when properly calculated, the average real GDP growth rate for countries carrying a public-debt-to-GDP ratio of over 90 percent is actually 2.2 percent, not −0.1 percent as published in Reinhart and Rogoff. That is, contrary to RR, average GDP growth at public debt/GDP ratios over 90 percent is not dramatically different than when debt/GDP ratios are lower.

Source: Herndon et al., 2013

Hands-on example

Example

Let's conduct an example considering the general workflow for collaborating through a Git repo

- Check the step-by-step activity here