GITting started with reproducibility: An introduction to git and knitr

Biostatistics Student Association Computing Workshop

Nick Seewald

Department of Statistics
University of Michigan

January 29, 2016

Why do I care about reproducibility?

Reproducible research is a hallmark of the scientific method, but we're pretty bad at it.

> *In 2012, a researcher then at the biotechnology company Amgen wrote in Nature that when his team tried to reproduce 53 landmark cancer studies, they could replicate just six. And according to a news report in Nature, a project aiming to reproduce the findings of 100 psychology papers has managed to replicate results for only 39 of them (the project's findings are still under peer review).*

"What Science Can Tell Us About Bad Science", *The Atlantic*, September 2015. `http://www.theatlantic.com/magazine/archive/2015/09/a-scientific-look-at-bad-science/399371/`

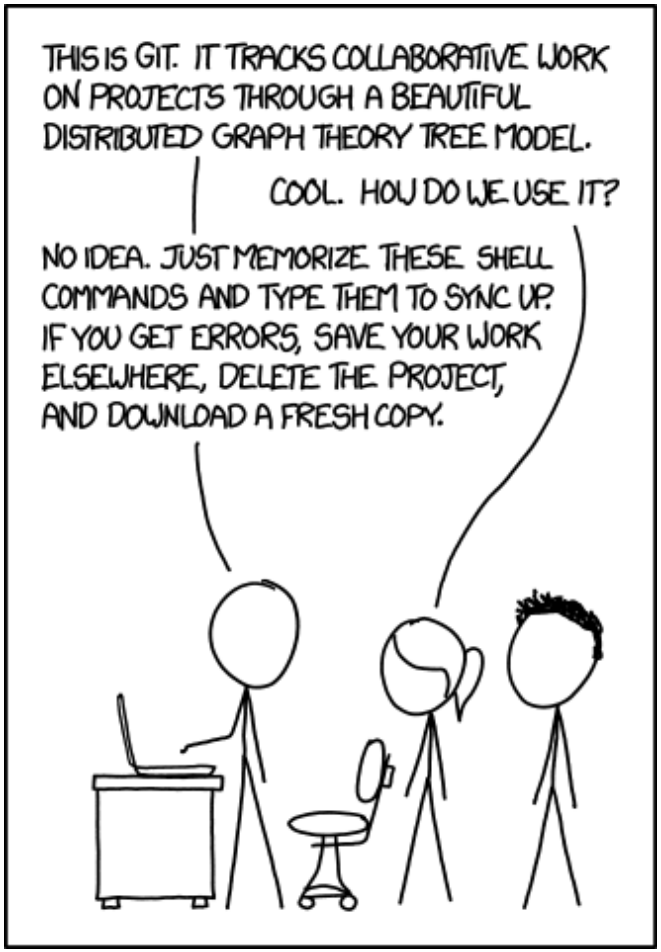
But I'm a Biostatistician!

- Reproducibility is important in both science AND statistics!
- As statisticians, we need to be able to reproduce our results on the same data set
  - This means we have to write reports in a way that minimizes error and write code so that we can get the same results years later.

Agenda

1. `Git`: A "version control" tool used for collaborating and maintaining different versions of a file, typically for code.
   - Great for collaborating, or just saving your own ass.
   - Often used in conjunction with *GitHub*, an online repository storage service.
2. `knitr`: An R package that lets you create documents containing R code and output.
   - Keep everything you need to generate a report (e.g., for research, homework, or 699) in one place!
   - My favorite part: Update code without having to re-create tables! (This is where errors creep in!)

A Brief Warning



Source: `https://xkcd.com/1597/`

Setup

Create a GitHub account, then download either Git or GitHub Desktop.

- Pure Git (i.e., just command-line tools): `https://git-scm.com/downloads`
- GitHub Desktop (GUI & command-line tools): `https://desktop.github.com/`

Forking a Repository

A *fork* is a copy of a repository. Forks let you

- make changes to an existing repo without affecting the original project
- use the existing project as a starting point for your own

One of the benefits of open source!

Exercise: Fork my `bsa-computing` repository.

Cloning a Repository

- To create a local copy of an existing git repository, use `git clone [url] [directory-name]`.
- In a terminal (Mac, Linux) or the Git Shell (Windows), navigate to the folder you want to clone the repository into.
- **Exercise:** Clone your forked `bsa-computing` repository onto your computer. The URL will be of the form `https://github.com/[username]/bsa-computing.git`

Make Changes!

- You now have a copy of both the current version of `bsa-computing` *and* access to every previous version.
- This clone is NOT automatically synced, à la Dropbox
  - Anything you break is completely isolated from the pristine copy on GitHub and your previous "commits".
- **Exercise:** Add to `food-exercise.md`, and save your changes.

Making Commits and Pushing

- Once you've accomplished a relatively small, but still significant task, you'll want to "commit" your code to the repository.
- This creates a labeled snapshot of the directory at the time of the commit.
- Syncing your commits with the existing remote repository is called "pushing".

Shell Commands for Commits and Push/Pull

- To *pull* down the most recent version of the remote repository, use `git pull`
- To make and push commits
  - Check what you've changed with `git status` and `git diff`.
  - Add files to the commit using `git add [files]`.
  - Create a commit using `git commit -m [message]`
  - Push your commit to the remote repository using `git push`

Merging Your Fork with the Original

On GitHub, create a *pull request* to merge your changes into my repository. The owner (me) will be able to look at and accept/decline them.

What is knitr?

- knitr lets you embed code and output from R into LaTeX, HTML, RMarkdown, etc. I'll be using LaTeX.
- Might require a bit more thought with R code to get the correct output, but it's worth it.

The knitr Bible: http://yihui.name/knitr/

Chunks

`knitr` separates R code from LaTeX using structures called "chunks".

Chunk Options

When you define a chunk, you also can include a number of options that tell `knitr` how to interpret the R code inside. Some basic important options are:

- `echo (TRUE)`: Whether or not to include the R code itself in the document.
- `results`: How to display results. Default is in a LaTeX `verbatim` environment.
  - `asis`: Write raw results into the document (e.g., with `xtable`)
- `include (TRUE)`: Whether to include the chunk output in the document. The code is still run. Useful for data steps.
- `cache (FALSE)`: Stashes the output of a chunk to avoid running it every time you compile.

More options and more details at `http://yihui.name/knitr/options/`.