Reproducibility with hpc

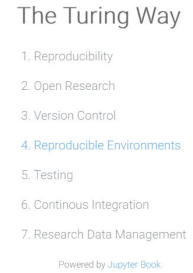
Becky Arnold & Anna Krystalli

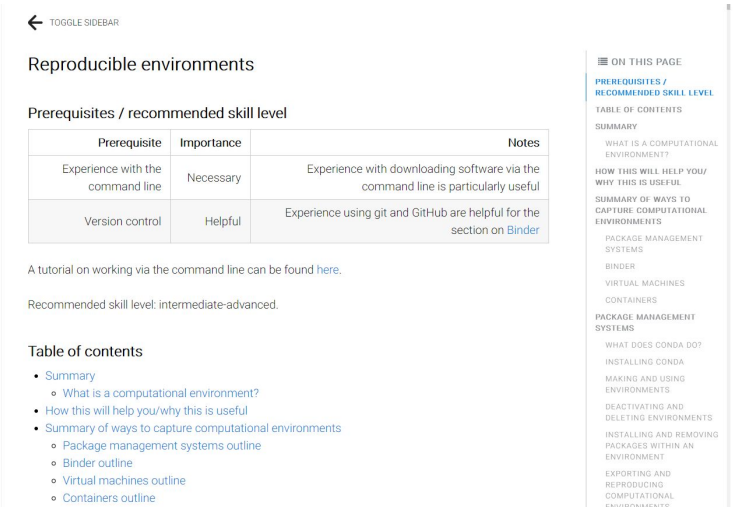
Our bit of the project

Open source textbook (Becky)

- What is reproducibility?
- Version control
- Open research
- Testing
- Reproducible computational environments
- Continuous integration

Checklists (Becky and Anna)





Make use of Git

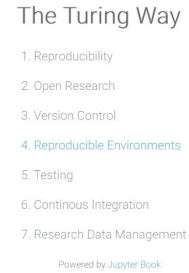
- [ ] Make your project version controlled by initialising a Git repository in its directory using `git init`
- [ ] Add and commit all your files to the repository using `git add .` then `git commit`
- [ ] Continue to add and commit changes as your project progresses. Stage the changes in specific files to be committed with `git add filename`, and add messages to your commits
  - [ ] Each commit should make one simple change
  - [ ] No generated files committed
  - [ ] Commit messages are meaningful, with a ~50 character summary at the top
  - [ ] Commit messages are in the present tense and imperative
- [ ] Develop new features on their own branches, which you can create via `git checkout -b branch_name` and switch between via `git checkout branch_name`
  - [ ] Branches have informative names
  - [ ] Master branch is kept clean
  - [ ] Each branch has a single purpose and only changes related to that purpose are made on it
- [ ] Once features are complete merge their branches into the master branch by switching to the feature branch and running `git merge master`
  - [ ] Merge other's changes into your work frequently
  - [ ] When dealing with merge conflicts make sure you fully understand both versions before trying to resolve them

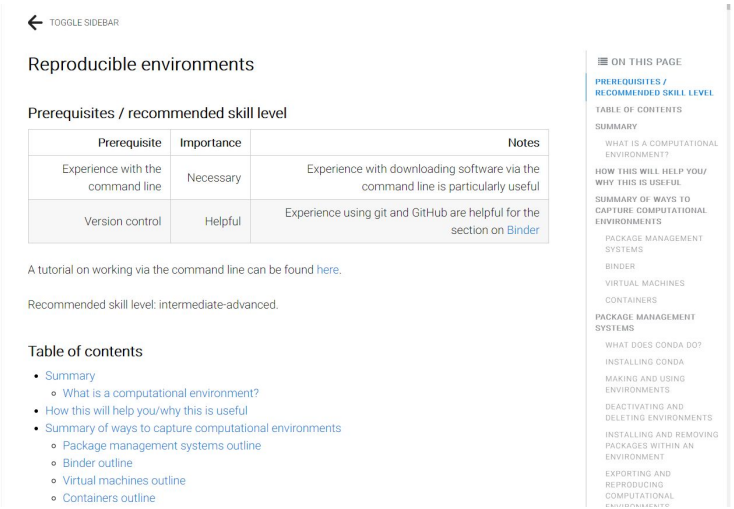
Our bit of the project

Open source textbook (Becky)

- What is reproducibility?
- **Version control**
- Open research
- **Testing**
- **Reproducible computational environments**
- Continuous integration

Checklists (Becky and Anna)

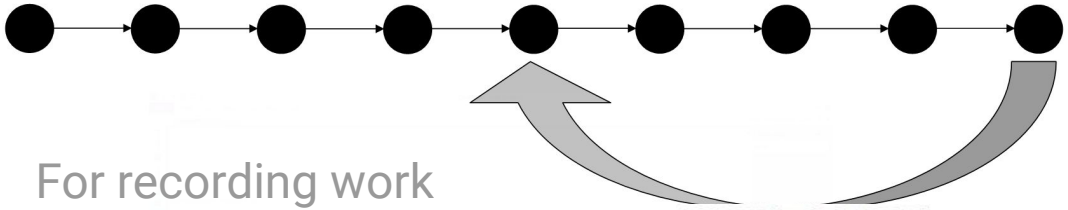




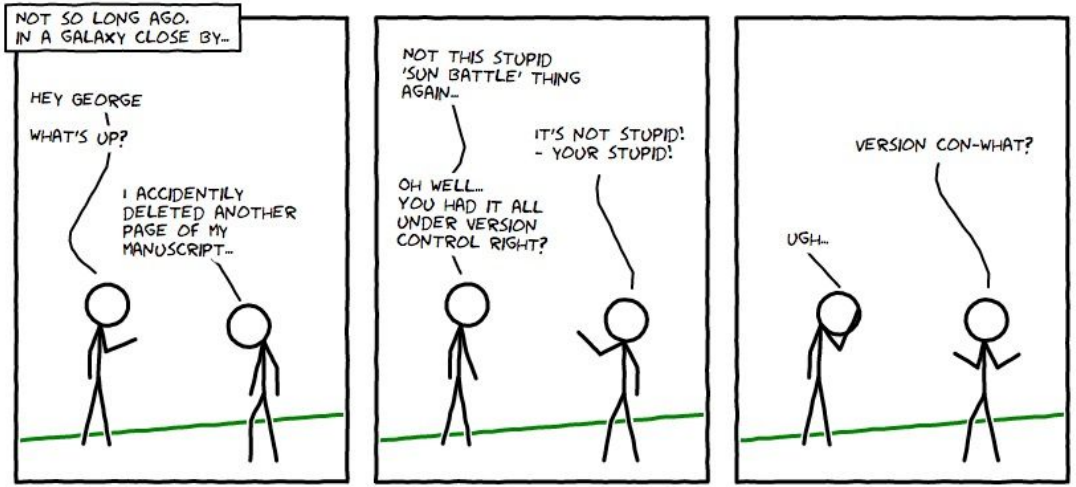
Make use of Git

- [ ] Make your project version controlled by initialising a Git repository in its directory using `git init`
- [ ] Add and commit all your files to the repository using `git add .` then `git commit`
- [ ] Continue to add and commit changes as your project progresses. Stage the changes in specific files to be committed with `git add filename`, and add messages to your commits
  - [ ] Each commit should make one simple change
  - [ ] No generated files committed
  - [ ] Commit messages are meaningful, with a ~50 character summary at the top
  - [ ] Commit messages are in the present tense and imperative
- [ ] Develop new features on their own branches, which you can create via `git checkout -b branch_name` and switch between via `git checkout branch_name`
  - [ ] Branches have informative names
  - [ ] Master branch is kept clean
  - [ ] Each branch has a single purpose and only changes related to that purpose are made on it
- [ ] Once features are complete merge their branches into the master branch by switching to the feature branch and running `git merge master`
  - [ ] Merge other's changes into your work frequently
  - [ ] When dealing with merge conflicts make sure you fully understand both versions before trying to resolve them

Version control for Research I

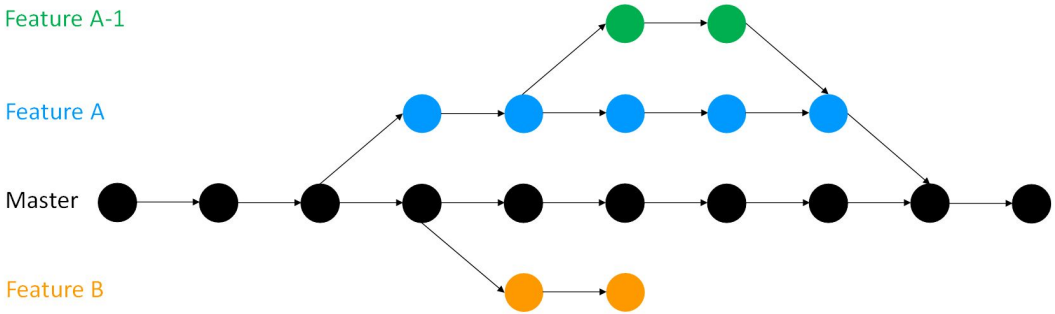


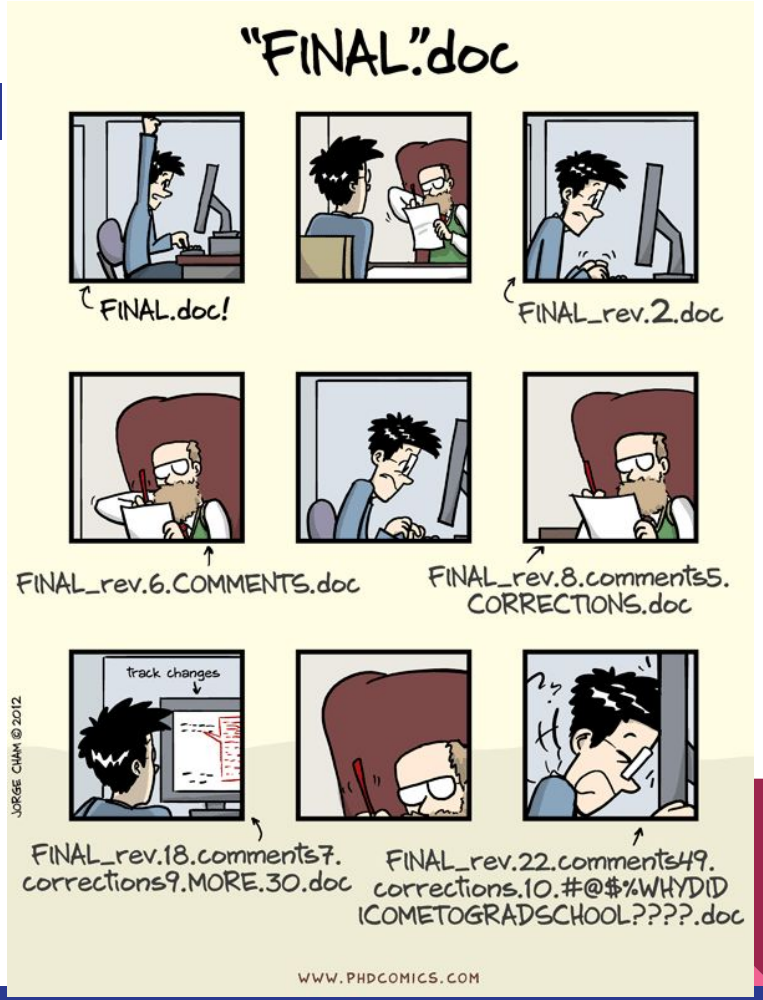
For recording work



Version control for Research II

For managing collaboration





Git & GitHub are a great way to sync with HPC

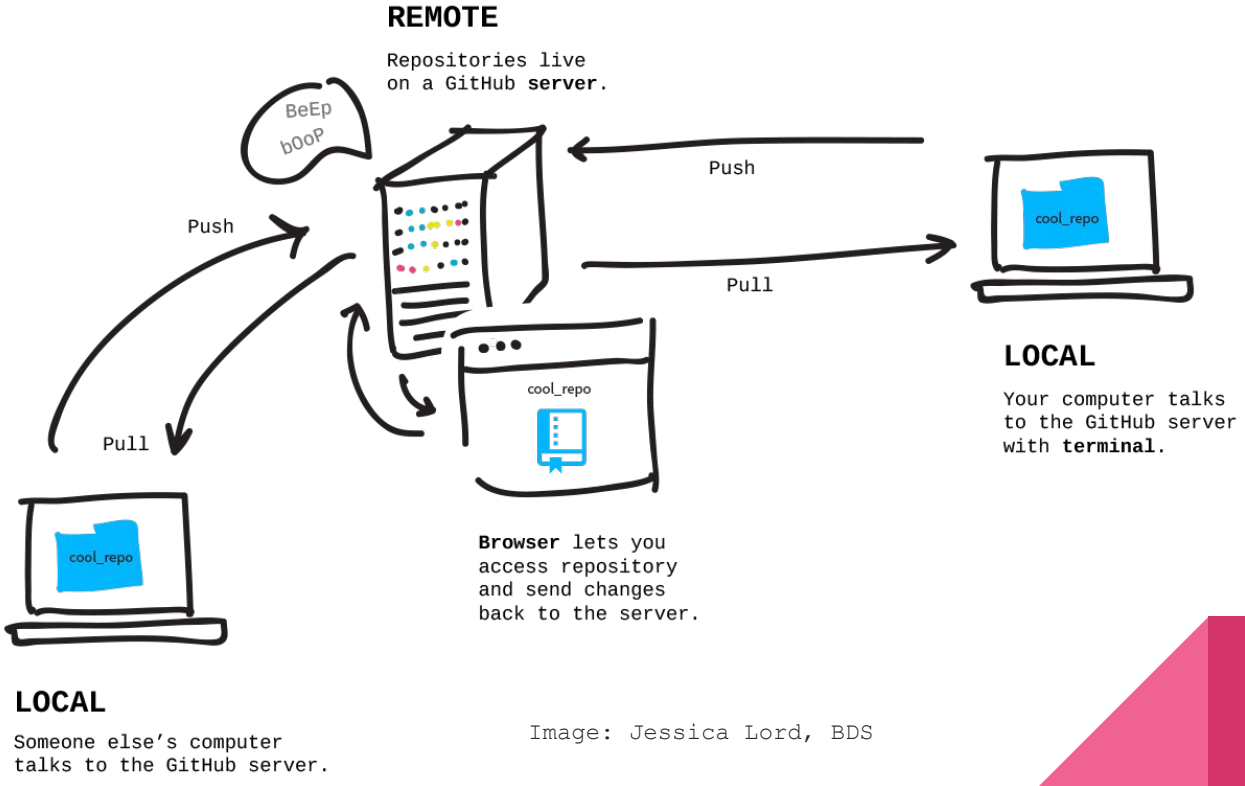


Image: Jessica Lord, BDS

Version control for tracking provenance

Lots of files to keep track of:

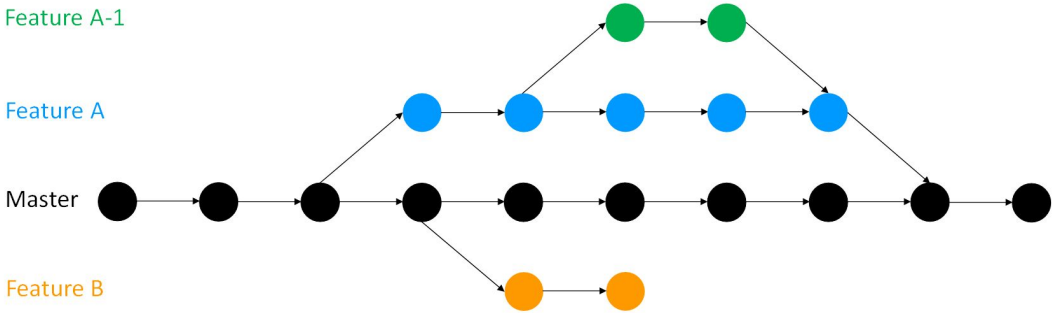
- Input files
- Code
- Make files
- Config files

Changes to these changes the results

- Overwritten files
- Can’t remember what they are = wasted hpc time

Easier to provide support

Easier viewing and access to code



Reproducible computational environments

Write on one computer, run on another

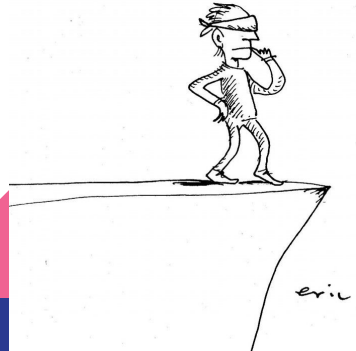
Transfer problems

- Installation issues (no admin privileges)
- Values/paths “baked in”

May not be able to run. If you can run it results may differ

E.g. different compilers/package versions

Defining environment important for reproducibility



Reproducing computational environments

Package management systems

- Conda
- Create and delete environments
- No admin privileges needed
- Export/import environments

Containers

- Package a computer
- Singularity (no admin)

Why testing matters

Reproducibility: “How do we know this code works?”

NEWS OF THE WEEK | SCIENTIFIC PUBLISHING

A Scientist's Nightmare: Software Problem Leads to Five Retractions

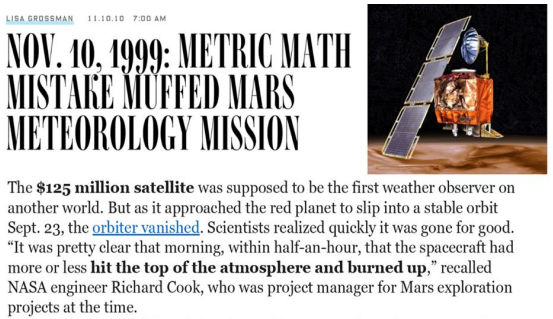
Greg Miller

+ See all authors and affiliations

Science 22 Dec 2006:
Vol. 314, Issue 5807, pp. 1856-1857
DOI: 10.1126/science.314.5807.1856

Everyone makes mistakes.

Can major consequences for projects and careers



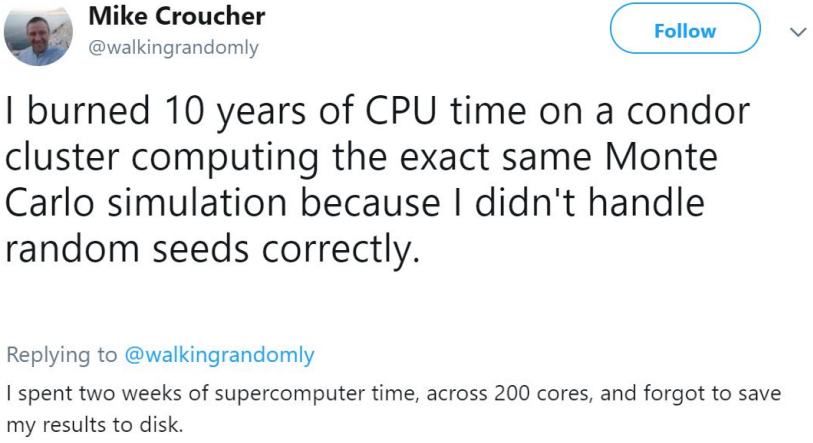
LISA GROSSMAN 11.10.10 7:00 AM

NOV. 10, 1999: METRIC MATH MISTAKE MUFFED MARS METEOROLOGY MISSION

The **$125 million satellite** was supposed to be the first weather observer on another world. But as it approached the red planet to slip into a stable orbit Sept. 23, the orbiter vanished. Scientists realized quickly it was gone for good. “It was pretty clear that morning, within half-an-hour, that the spacecraft had more or less **hit the top of the atmosphere and burned up**,” recalled NASA engineer Richard Cook, who was project manager for Mars exploration projects at the time.

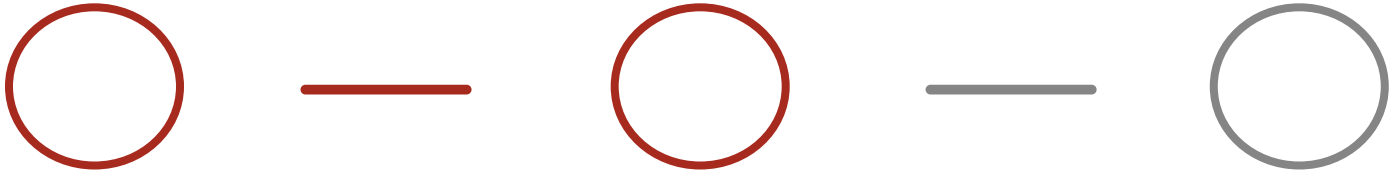
Mistakes on HPC

<u>Everyone</u> makes mistakes.



On hpc mistakes can be costly in both time & resources

Testing & HPC



Testing suite **Better local code** **Better code on the cluster!**

Running tests on the cluster:

+ Can flag problems early on
+ Can aid trouble-shooting

Testing in the Turing way

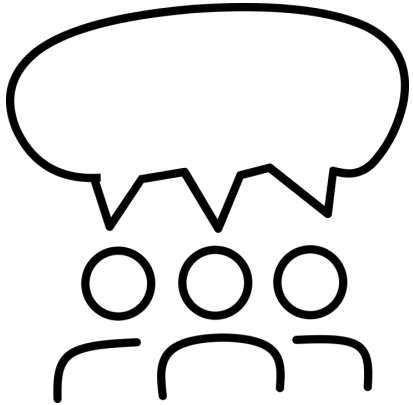
General good practice

Tips e.g. for testing stochastic code

Types of tests

- Uses
- How to implement them
- Type-specific good practice
- Type-specific tips

Come talk to us!



Booth over lunch- come chat!

Groundwork for chapter on reproducibility with HPC

HackMD: https://bit.ly/2CGhH1K

Want to hear experiences/horror stories/advice: https://bit.ly/2CL9CsD

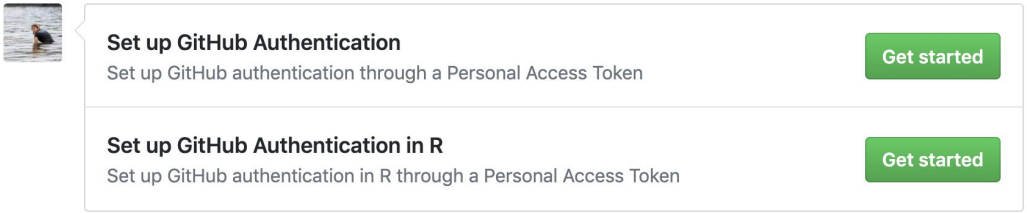
Project repository: https://github.com/alan-turing-institute/the-turing-way

Book: https://the-turing-way.netlify.com

Thank you for listening

check✅ lib https://github.com/checklib/checklib

#ColabW19 Project: a library of markdown checklists that can be incorporated as GitHub issue templates into your own projects.



https://github.com/annakrystalli/IUCNextractR/issues/1