git: A powerful tool to facilitate greater reproducibility and transparency in science.

Karthik Ram, Ph.D.
Environmental Science, Policy, and Management.
University of California, Berkeley.
Berkeley, CA 94720. USA.
karthik.ram@berkeley.edu

Abstract

Reproducibility is the hallmark of good science (Vink et al., 2012). Maintaining a high degree of transparency in scientific publications is essential not just for gaining trust and credibility within the scientific community but also essential for facilitating novel science. Sharing data and computer code associated with publications is becoming increasingly common, motivated partly in response to data deposition requirements from journals and mandates from funders. Despite this increase in transparency, it is still difficult to reproduce or build upon the findings of most scientific publications without access to a complete workflow.

Version control systems (VCS), which have long been used to maintain code repositories in the software industry, are now finding new applications in science [@citation_for_vcs]. One such open-source VCS, **git**, provides a robust framework that allows scientists to track every component (data, code, figures, and text) of a research endeavor from start to finish, with the ability to revert any file or an entire project back to any stage in its development. Since the system is decentralized, every copy of a repository not only contains all the data and code but also the entire history of changes along with detailed notes documenting each of those decisions. The power of a git repository can be further extended by linking it to one or more git hosting services (e.g. GitHub, BitBucket, self-hosted) which makes it easy for multiple collaborators to work asynchronously and merge their changes as needed. A git based workflow is particularly suited for science because it is designed to protect against data loss, quickly retrace errors, and allow multiple ideas and methods to co-exist in parallel.

In this paper I review how git benefits science and why more scientists should make git-based workflows an integral part of their research. I also provide use-cases demonstrating how git repositories can foster collaborations, increase accountability, track contributions, in addition to lowering barriers to data reuse and supporting novel synthesis.

Introduction

One of the original motivations behind publishing scientific articles was to ensure that results could be reproduced, validated and extended. Articles with detailed methods sections were necessary not just for evaluating the soundness of the central findings but also to ensure that readers could validate such findings and improve upon them. Over time, the scientific process has become quite complex, requiring increasing amounts of data collected using complex methods. The volume of research has also grown considerably over the years, resulting in a shortening of the length of articles (first because space was limiting in publications but later continued even though e-papers don't have any limitations). Now most papers contain brief methods. To reproduce the central findings of any paper, a reader not only requires access to detailed methods, but also the underlying data and code used to arrive at them. While this practice has become more common, there are several problem. Many articles still do not share their data and or code. There is also the problem that they are not shared in adequate detail (derived data versus raw data) or all authors do not share all the decisions they used to arrive at their final conclusions. So, some decisions, such as removing outliers, may not be adequately in the steps.

Another problem is the rise in retractions in recent years. (Van Noorden, 2011). Talk about why retractions are costly, hurt science, and are a bad thing overall.

Open science is needed to accelerate resaearch. In the era of declining funding, we need to leverage existing data and code in new ways. Not only that we must also ensure that our data come with appropriate licenses that not only support fair use but also credit original authors (Neylon, 2013). Sharing git repositories with a full history of changes can be one way to lower barriers and accelerate open science. In this paper I describe why the distributed version control system **git** is ideal for supporting reproducibel research.

Science is supposed to be reproducible. We shared all the relevant details necessary for someone else to reproduce, validate, or build upon existing findings. As technology has improved, so has the software and the size of data. These days methods sections are fairly short and rarely provide sufficient detail to reproduce the central findings, or build upon them. Even when sufficient raw materials are shared, producing a manuscript requires a research team to make several small decisions that are often relegated to lab notebooks and rarely available for review. Some of these decisions could be vital, such as deciding to subset data, remove outliers, and or choosing one type of statistical model over another. Even if a researcher is open to sharing these types of information, there is often no venue to do so. Thus, we need a more robust way to share all the decisions that result in a manuscript, along with all the assets (code, data, and ...).

Version control systems

Version control systems have been used in software development for code review, and as a way to capture the entire provenance of a project. Each commit (or snapshot) also provides a way point that allows one to explore new methods and ideas and quickly revert if this results in a dead end. It also allows for new methods to be explored side by side, in different branches. Thus, when sharing a git-managed repository, all of these nuances are also shared alongside.

But there are several challenges with traditional VCS. For one, they tend to rely on a central server. Without a connection to the server, it would be impossibel to get much done. If this was a 3rd party service, then there is no guarauntee that the service would persist. More recently, a relatively new DVCS, git has taken over as the widely used one. Judging from the number of public repositories hosted by GitHub, a popular git hosting service, there is a poplar trend. While the tool was build with the idea of supporting software development, many of its characteristics are ideally suited for science. In this article I argue that git is ideal to manage day to day scentific activities such as gathering data, writing manuscripts, analyzing data and visualizing the results. Used correctly, git could make more research easily reprodicible and accessible, improve transparency because one could verify results and retrace steps that might have led to errors, and also build upon from the right point (e.g. where one might require data in a raw, untransformed state).

Why git?

Various version control systems have existed for a long time. A common feature in most VCS is that they are centralized, which means that one has to checkout a copy from a central repository, make changes, and check those back into the central repository. Version control systems have existed in various forms over the years. A common feature with most is the existance of a central repository from which people checkout copies, contribute changes, then commit. git differs in that the process is decentralized. when the main hub is unavailable, a user can still get work done.

1. The process is distributed. So any of the collaborators are free to work asynchronously and merge their changes as needed.
2. By linking a git repo to a remote git hosting service, it serves both as a backup but also as a central point where multiple contributors can synchronize their changes. Unlike using track changes in Microsoft Word where the history of changes and the authors who proposed them are lost once accepted or rejected, in this case git preserves authorship and can even be used to track contributions over time.
3. git supports a very flexible branching model, which means that it supports efforts to explore various ideas and methods in a managable way without an unmanagable proliferation or files or no way to keep track of them.

Advantages of git as a tool to manage scientific workflows

The features that make `git` the popular choice among software developers also make it ideal for managing scientific products. Most importantly, git retains a complete provenance with every single copy of a repository. Since there is no need for a central server (any copy can act as either the client or server), there is no single point of failure. Perhaps the biggest reason that makes git so well suited for tracking scientific activities is that it maintains a history of authorship not just in the original repository but also in every single copy cloned by anyone.

1. git logs as a Lab Notebook

Since git requires a description (referred to as commit message) to accompany each commit, over time these messages can serve as a human-friendly log of changes that occured over the course of a project. While lab notesbooks are a wealth of information, they are rarely available alongside a publication or to members outside of that lab. Since the git history is always part of the repository, anyone with a copy can search, query, and browse this history of changes without access to original lab notebook entries.
A common method for ad-hoc versioning in research settings is to duplicate files with semi-informative names (like `data-old`) but over time these become impossible to track or revisit without a detailed record or history. With git on the other hand, with each addition, such as new data, new lab entries, new text for a manuscript, git captures those changes making it easier to compare versions and see notes. And since each copy of a repo carries the entire history of notes, someone trying to build upon this work can easily see what decisions were made at each stage of a projects' evolution and decide where to branch from.

4. Tracks Collaboration

Collaborators can share write-access to a git repository and push changes asynchronously. In other cases, a single user can retain ownership and allow others to fork a copy, and submit pull-requests.
A git hosting service allows multiple collaborators to work asynchronously without having to wait on anyone else. As they finish meaningful contributions, local changes can be merged with a remote copy. While most merges will be automatic, edits to the same lines will require manual intervention. In papers where I have collaborates with other scientists, the lead author creates a repository on github, and every contributor `forks` a copy to edit. As they incorporate their individual contribtions, authors can send `pull requests`, which are requests to merge both copies at the discretion of the lead author. Authors can update their copies from the original branch anytime to keep up with changes merged from other pull requests. This workflow allows lead authors to manage how and when individual contributions get merged into the master copy of a manuscript. In a recent paper led by Philippe Desjardins https://github.com/PhDP/article_preprint/network, we followed this exact protocol.

4. Backup/failsafe

Storing local copies without adequate backups, or hosting data and code on lab websites are often unreliable. Data can lost due to accidents, computer failures, and changes in web pages can result in 404 over time. Using a git repository provides a remote backup during the course of the project. Although none of the existing git remotes are members of CLOCKSS, where membership gaurantees permanent archiving, upon compeltion of a project one could easily zip up a git repository and store in a repository such as figshare, or other. These can be imported and built upon.

Repositories can be made private during development (but released publicly upon completion). This has to be decided by the lab group and the privacy implications surrounding the data itself. Although open science is a good practice, not every lab can do this. Those labs can still leverage the power of git but keep the remote copies private and only share them with select groups of people. 4. Freedom to explore new ideas and methods With branches, it becomes easy to explore new ideas. For example, reviewers often ask authors why

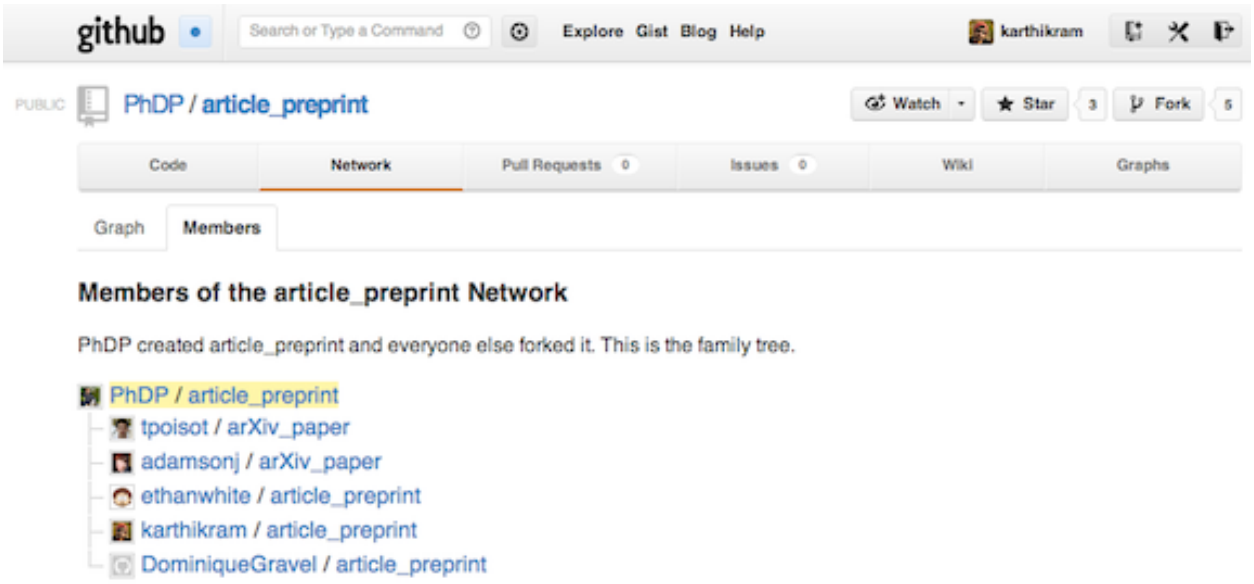

Figure 1: A network graph showing list of collaborators and their contributions to a project

they used one model over another. In a git framework, authors can test various approaches in a managable way by creating branches. In each branch they explore a different method. Ones that do not yeild productive or useful results can be left as a record without affecting the master branch which would contain the final approach chosen for the paper. Reviews who have some familiarity with how git works can easily see these other efforts. Even if reiewvers themeselves as not git savvy, given how long review process takes, this would allow authors themselves to dig up these alternate methods quickly becuause those files are also part of the same original repo. In a non-git framework, these files would likely be deleted.

4. As a mechanism to solicit feedback and reviews

While it is possible to leverage most of core functionality in git at the local level, git hosting services offer additional advantages that can accelerate collaboration and review. Issue trackers can provide a mechanism for both feedback and review, especially since they can easily be linked to particular lines or blocks of code.

5. Transparency, verifiability

Methods sections in papers are often brief and succinct to adhere to strict word limits imposed by journal guidelines. This practice is particularly common when describing well-known methods where authors assume a certain degree of familiarity among informed readers. One unfortunate consequence of this practice is that any modifications to the standard protocol implemented in a study are not available to the reviewers. However, seemingly small decisions, such as choosing an appropriate distribution to use in a statistical method, can have a disproportionately strong influence on the central conclusion of a paper. Without access to a detailed history, a reviewer compotent in statistical methods has to give the benefit of the doubt to the authors and assume that assumptions of methods use where clearly met. Sharing a git repository can remove these kinds of ambiguity and allow authors to point out commits where certain key decisions were made before using particular analytic approaches.

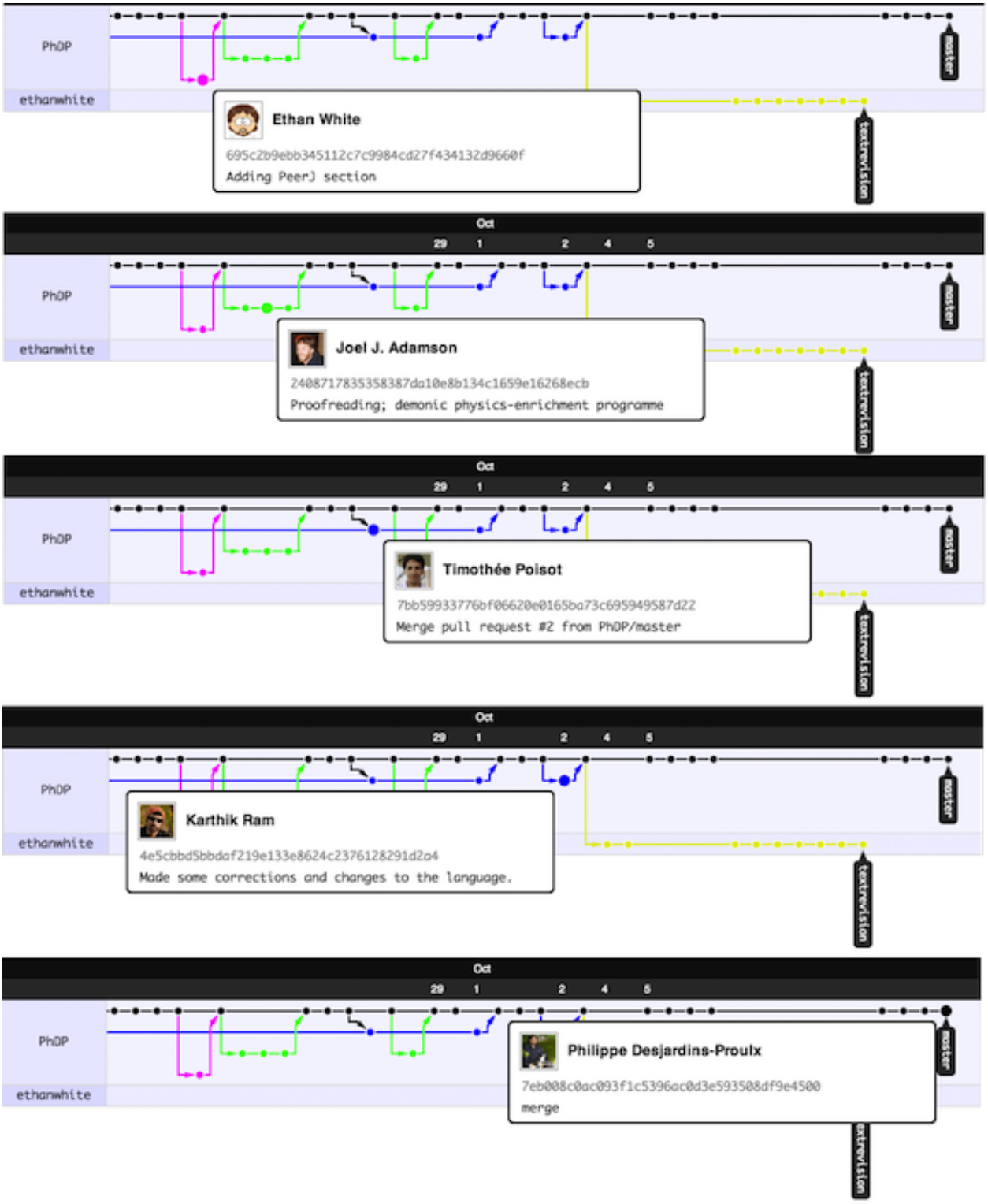

Figure 2: git makes it easy to track individual contribtions through time which helps track effort and maintain accountability

6. Managing large data

git is extremeley efficient with managing small data files such as ones routinely collected in experimental and observational studies. However, when the data are particularly large bioinformatics studies (in the order of tens of megabytes to gigabytes), managing them with git can degrade efficiency and slow down the performance of git operations. When large data files do not change often, it would be best to exclude such files from the repository by adding it to a ignore list, and tracking the metadata separately. In such cases, the best practice would be to exclude those data from the repository and only track the metadata. This protocol is especially ideal when such large datasets do not change often over the course of a study. In situations where the data are large and undergo frequent updates, one could leverage third party tools such as git-annex http://git-annex.branchable.com/ which makes this process of managing data with separate remotes without any need to check the data into the repository. The tool makes this process transparenyt to the user so it appears as though the data are being tracked by the same repository as the other project files.

7. Lower barriers to reuse

A common barrier that prevents someone from reproducing or building upon an existing method is lack of sufficient details about a method. Even in cases where methods are adequately described, the use of expensive proporietary software with restrictive licenses makes it difficult to do so. Sharing code with licenses that encourage fair use with appropriate attribution, removes such articifical barriers and encourages readers to modify methods to suit their research needs, improve upon them, or find new applications. Although this process of depositing code somewhere public with appropriate licenses create additional work for the authors, the benefits clearly outweight the costs. Making all research products publicly available not only increases citation rates (Piwowar et al., 2007) but can also increase opportunities for collaboration. For example, (Niedermeyer and Strohalm, 2012) struggled with finding appropriate software for comprehensive mass spectrum annotation, and eventually found an open source software to extend. In particular, the authors cite the open nature of the software as the catalyst for their choice. Examples of such collaboration and extension will only become more common by sharing fully versioned copies of projects.

A similar argument can be made for data as well. Even publications that deposit data in persistent repositories rarely share the original raw data. The versions submitted to persistent repositories are often cleaned datasets. In cases where no datasets are deposited, the only data accessible are mean values reported in the text. Raw data can be leveraged to answer questions not originally intended by the authors. For example, research questions that aim to address uncertainly often require messy raw data to test novel methods. Thus, versioned data provide opportunities to retrieve copies before they have been cleaned up for use in different contexts.

Conclusions

The power of the git version control system could be extremely beneficial to science. By managing projects with git right from the start, and regularly documenting additions and changes to the projects over time, researchers can ensure that all decisions made over the course of a project are securely stored in a format that can easily be viewed, and queried. Further, unlike lab notebooks which are not often accessible to everyone, a git repository carries the full history of changes with each copy. Because it works in decentralized format, loss or theft of any single computer will not result in catastrophic data loss. The use of GitHub as a remote, with regular syncs will ensure that a current copy can be retrieved from anywhere. If possible, including a link to a github repo in submitted manuscripts provides the transparency that reviewers often wish for. A git savvy reviewer could make much better decisions, rather than relying on incomplete information presented in methods sections. I wrote this manuscript as a git repository right from the beginning and a copy is available at: https://github.com/karthikram/smb_git.git

Acknowledgements

I was supported by NSF DEB-1021553 while preparing this manuscript. Comments from ... on earlier drafts greatly improved the final version of this article. This manuscript is available as a git repository (with a full history of changes) https://github.com/karthikram/smb_git.git along with a permanent archived copy on figshare (http://figshare.com/) (I'll add a link to figshare URL once a final version of the paper is accepted).

Literature Cited

Neylon,C. (2013) Open access must enable open use. *Nature*, **492**, 8–9.

Niedermeyer,T.H.J. and Strohalm,M. (2012) mMass as a software tool for the annotation of cyclic peptide tandem mass spectra. *PloS one*, **7**, 44913.

Piwowar,H.A. et al. (2007) Sharing Detailed Research Data Is Associated with Increased Citation Rate. *PLOS One*.

Van Noorden,R. (2011) The trouble with retractions. *Nature*, 6–8.

Vink,C.J. et al. (2012) Taxonomy and Irreproducible Biological Science. *BioScience*, **62**, 451–452.