Checklist for getting started with `git`

Aaditya Ramdas (`aramdas@cmu.edu`), Carnegie Mellon University

Some personal suggestions (via accumulated wisdom) for consistency within my own research group.

- **Why git?** If multiple authors are editing a paper on Dropbox/Box/Drive, it will result in conflicted files (unless authors create a token system for editing and follow it dutifully). If conflicts occur, it is often unclear how to merge the copies. Version control using git (say GitHub/GitLab) helps track changes, permitting simultaneous editing of code/papers. Browser-based editors (like Overleaf) can also be synced with git.
- **Folder structure.** Folder structure helps minimize #clicks to find the right file. The style I prefer is that there is a single tex file for the paper and this is not in a subfolder. Use separate folders for `refs` (related work), `code`, `figures`, `notes` (brainstorming, rough ideas, etc.). Last, use a subfolder called `attic` to keep old versions of anything (figures, tex, etc) that probably will never need a revisit.
- **Cloning an existing repository.** To create a local copy of an existing repository, use

  ```
  git clone https://github.com/[repo-owner]/[repo-name].git.
  ```

  The following commands must be issued in the terminal from within the created folder.
- **Pulling and merging.** The command

  ```
  git pull
  ```

  will merge the latest version on the repo (changes made by others) with your local version. This resolves conflicts automatically if possible, otherwise search for `HEAD` to find the conflicts and resolve them manually.
- **Committing and pushing your changes.** Use `git status` to see the locally modified files. Then,

  ```
  git add [filename].tex
  ```

  will mark a file that you want to commit. To actually perform the commit, use

  ```
  git commit -m "[updates made]".
  ```

  Finally, to push changes to the repo, say

  ```
  git push.
  ```
- **What files should be committed?** Only commit `.tex` and `.bib` files, and figures needed for compilation. Never commit auxiliary files that are generated by compiling the tex file locally or generated by running code, including the final `.pdf` of the paper. `https://github.com/github/gitignore/blob/master/TeX.gitignore` has a good `.gitignore` file, allows you to say `git add` ∗ if you have a lot of new figures to commit.

The repository should be private, and only visible to project collaborators. More advanced users may wish to split the master branch into other branches, these can be accessed with commands like `git checkout [branch-name]`.

Before submitting the paper to a journal/conference, please do the following for transparency/reproducibility.

- For figures/tables in the paper relating to simulations or real-data, make the associated code (only) public.
- Add instructions, perhaps as `.html` or a `readme` file, or a Python notebook, that clearly specifies how a user needs to run the code in order to reproduce the results.
- We may wish to upload the paper to ArXiv at the time of, or sometime just before or after, submission to a conference/journal (if allowed). The paper should link to the aforementioned code, perhaps as a footnote.
- The ArXiv paper should always be in a neutral template if possible: mostly involves `\documentclass{article}`, `\usepackage[margins=1.5in]{geometry}`. Email IDs and affiliations below the title enable easier feedback.
- Finally, link to the ArXiv paper, and separately to the code, from your personal website.