1 Introduction

git is a version control system, which means it facilitates tracking changes made to your documents. It is important to do this because it allows one to backtrack to happy times when code ran and data were not jacked up.

git can be run locally, meaning you keep the history of changes to your documents on your personal computer. git can also be used to sync changes/history to a remote computer. This is highly recommended because it allows you to sync your work among your computers (e.g. laptop and work desktop, etc.) and because it provides an external backup.

1.1 Purpose of this document

A simple exposition of how to use git. Much of this has been duplicated umpteen times on the internet, but repetition is good. Examples will include using github and Rstudio together. For how to use Overleaf with git and R see Jessi Rick's tutorial:
github.com/jessicarick/resources/blob/ master/tutorials/R_Overleaf_Integration.pdf

1.2 Terminology

repo - A git repository, or repo for short, is a directory that contains all the stuff for a particular project that you are tracking. One will probably want to have separate git repos for each project. One will likely have local repos that are associated with remote repos. Just think of a repo as a folder or directory. Just like you may have the same folder on your home computer and google drive or dropbox, so you can have repos that are associated via git. git is powerful because it allows convenient syncing of these folders.

commit - a term used to describe the act of saving a change to your document.

push - sending changes to your local documents to a remote repo. Say you are working on your laptop and you commit your changes and push those to your remote repo. Now your changes are backed up remotely and ready for download by anyone that has access to the remote repo.

pull - downloading changes to your documents that are saved to a remote repo to your local computer. Say you head home for the day, but want to download to your laptop the changes you made to your documents while using your work computer. Assuming that before you left work you pushed your changes to a remote repo, then you can pull those changes to your laptop.

clone - this means to copy a remote repo to your own system. Say you like an R package and want to download its code and start hacking it. You would clone the repo for the package to do this. You might also start a new project on github or somewhere then clone it to your local computers.

branch- say you are working on a project and have an idea for an analysis that takes your work in a very new direction. But you still want to keep this new direction under the same project, and perhaps you might want to work on the new idea and the old idea at the same time to figure out which method works best. You can do this by making a new branch for the new idea. Once you figure out which method you like, then you could try to merge the branches, if desired.

fork - this is more of a github construct by my understanding. It is a cloned repo that you are working on. Once you do something worthwhile you can send a pull request to whomever is managing the main repo to have them incorporate your changes.

pull request - say you are hacking away on your forked version of some project and come up with something cool. You can then ask the person who runs the main project repo to pull your changes and incorporate them. This is a pull request, because you are requesting someone else pull your changes.

Other terms exist and will be defined in place in the examples.

2 Make a repo on github and link it to R

1. Go to github (set up account if you don't have one) 2. Make a new repo (at time of writing a plus symbol in upper right of window) 3. Copy the URL shown 4. Go to R studio and make a new project using checkout from version control 5. paste the URL in and that is it!

Now make a test file. Go to the git tab and click the new file. A window will pop up and ask if you want to commit and prompt you for a commit message.

Use commit messages to make notes regarding that version's unique use (if there is one). For instance, you could say "important addition of function x" that way you can find that version in the future if you need it.

push the new commit. Now it shows up in github.

Now, say you were on a different computer, you could clone the repo you just made. Just repeat the process described above, but use the URL for the repo you just made. Easy!

Of course, all this can be done on the command line, and this is a skill that is worth learning so that you can use git for things outside of R studio. To do that, use these commands from the command line after navigating to the directory that you want to be your repo:

git init
git add ProjectFolderName
git commit -m "first commit"
git remote add origin https://github.com/YourGithubUsername/RepositoryName.git
git push -u origin master
The u is for upstream

3 Make a repo locally and push it to github

Open an existing R project. Go to Tools¿Version Control¿Project Set up. Click on Git/SVN in the window that pops up. Click git from the drop down that says "Version control system" You will need to restart after clicking ok. Now you have a local repo. You will want to link this to a remote repo. First copy the link for the github or bitbucket repo you want to be the remote (make a new repo on github if needed).

Next open up the shell and navigate to your directory. Type: git remote add origin https://github.com/YourGithubUsername/RepositoryName.git

git add whateverfilesyouwant
git commit -m "first commit"
git push -u origin master

Note that if you are trying to push to a git repo that already has stuff in it and therefore a history that diverges from the repo you are trying to link, you will have problems. It is better to clone that repo and work on it, if that is an option. If you have two different repos that you want to merge, then you can google around to figure out how to do this, but it can be somewhat cumbersome.

4 Problems

Sometimes you will edit stuff locally and forget to push it to a remote repo. When you try to pull stuff from the remote repo you will get an error saying your local repo/branch is ahead of the remote one and you wont be able to pull until you stash your changes.

You can stash changes (look up stash), or just delete the offending file and try pulling again. It is best to get into the habit of pulling at the beginning of a session and pushing at the end to avoid these annoying discrepancies.