How to get started with GitHub for CTT-Archbold data management
Author: Young Ha Suh
Date: Oct 19, 2020

Prerequisites:
- Create an account at https://github.com
- Download Git at https://git-scm.com/downloads
- Download R studio https://rstudio.com/products/rstudio/

Useful readings prior:
- Ch 1. Jenny Bryan's Happy Git with R
- Excuse me, do you have a moment to talk about version control? by Jenny Bryan
- GitHub for project management

1. Introduction

This is a coarse documentation on the step-by-steps to setting up your local device for GitHub and getting the necessary code from Cellular Tracking Technologies (CTT). Ultimately, CTT's goal is to have an R package but who knows when that will happen. Also, by editing the code directly, this gives us leverage to manipulate the code and produce code that will be useful for our own purposes.

Github is like an interactive Dropbox where you can share files online but the crucial feature is that it tracks and syncs any changes made, making it particularly useful for your own version control or collaborations using code because you can revert to older versions or contribute to a common code. You can also build websites! Overall, it is a really neat tool that is rapidly gaining popularity and proving to be extremely useful for collaborations, especially when sharing data sets or R code. **Git** is the software that tracks versions of your files and GitHub stores this online, allowing collaborations. I will not go into too much detail here especially since there are much better documentations and articles out there but there are a lot of technical jargon which I will try to explain as we go. For a better explanation for some of these terms, take a note at the readings above.

2. Ensuring Git & R studio are communicating

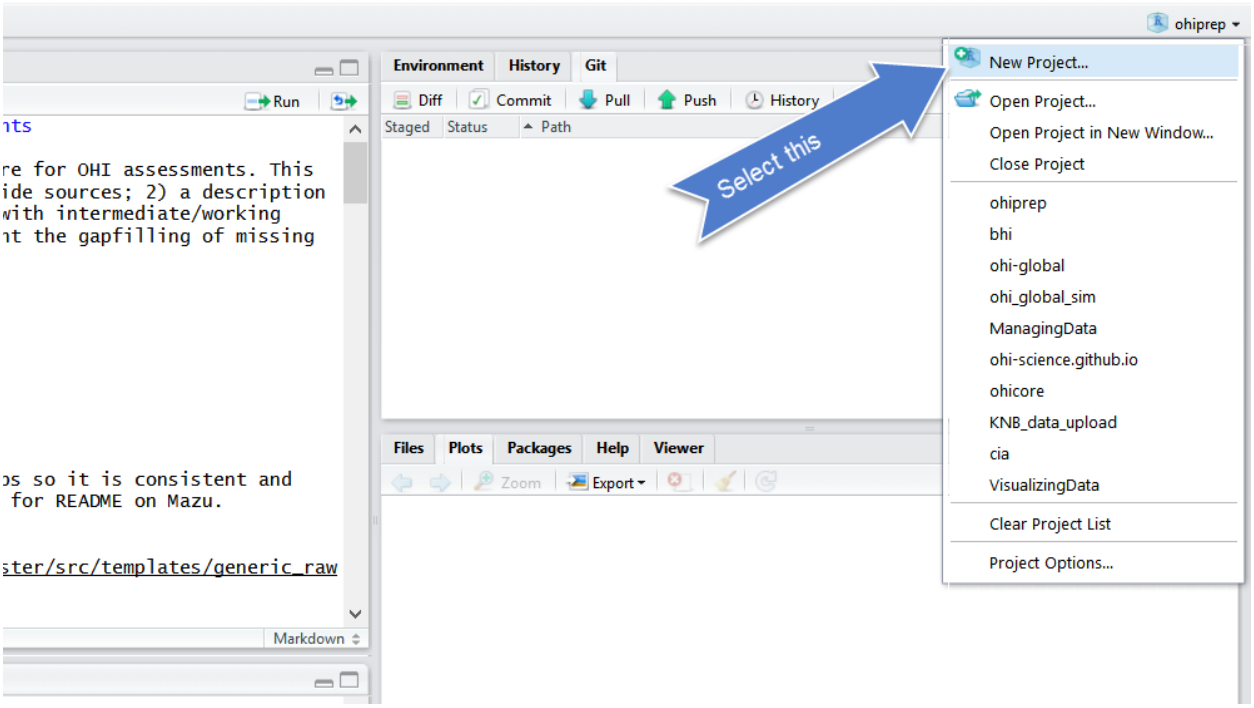
Before we begin, we need to make sure to set up the configuration so that GitHub and R can communicate. This is a one-time thing but you will need to remember your GitHub username, the email address you created your GitHub account with, and your GitHub password.

Start by opening R studio and a new R file, and typing the following:

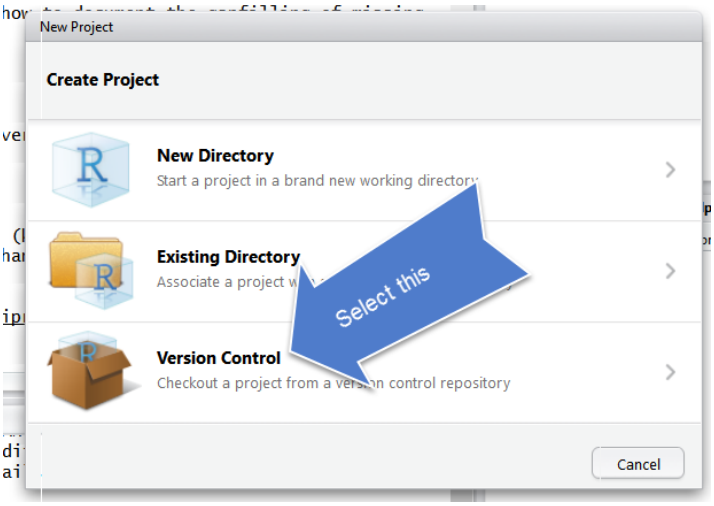
```
install.packages("usethis")
library("usethis")

## use_git_config function with my username and email as arguments
use_git_config(user.name = "jules32", user.email = "jules32@example.org")
```

Hopefully this worked but let's check quickly if it actually did.

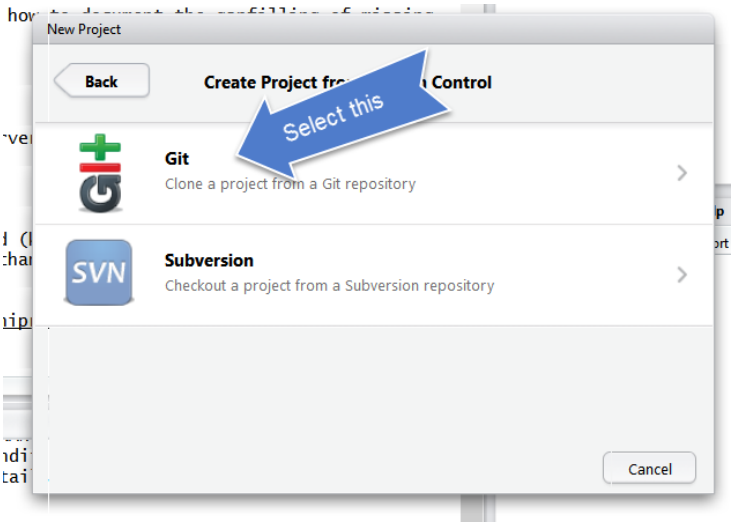
Now, click on New Project.



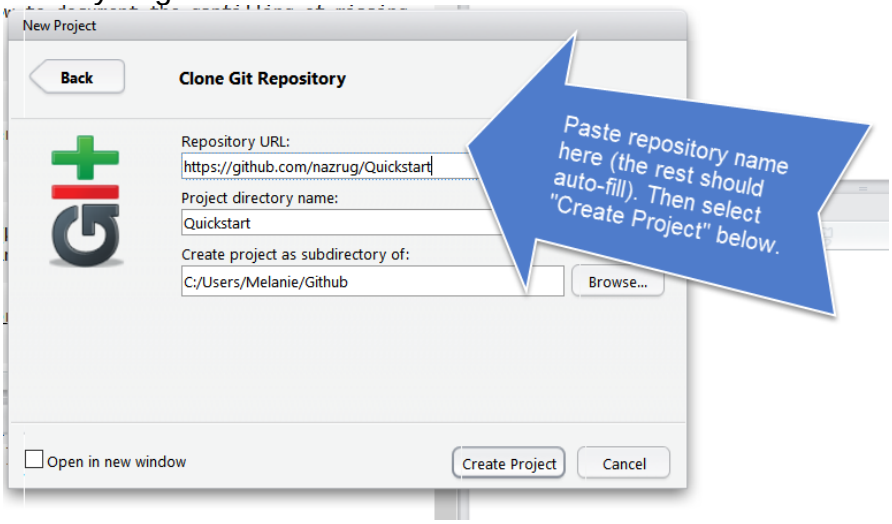
Select Version Control.



Select Git.

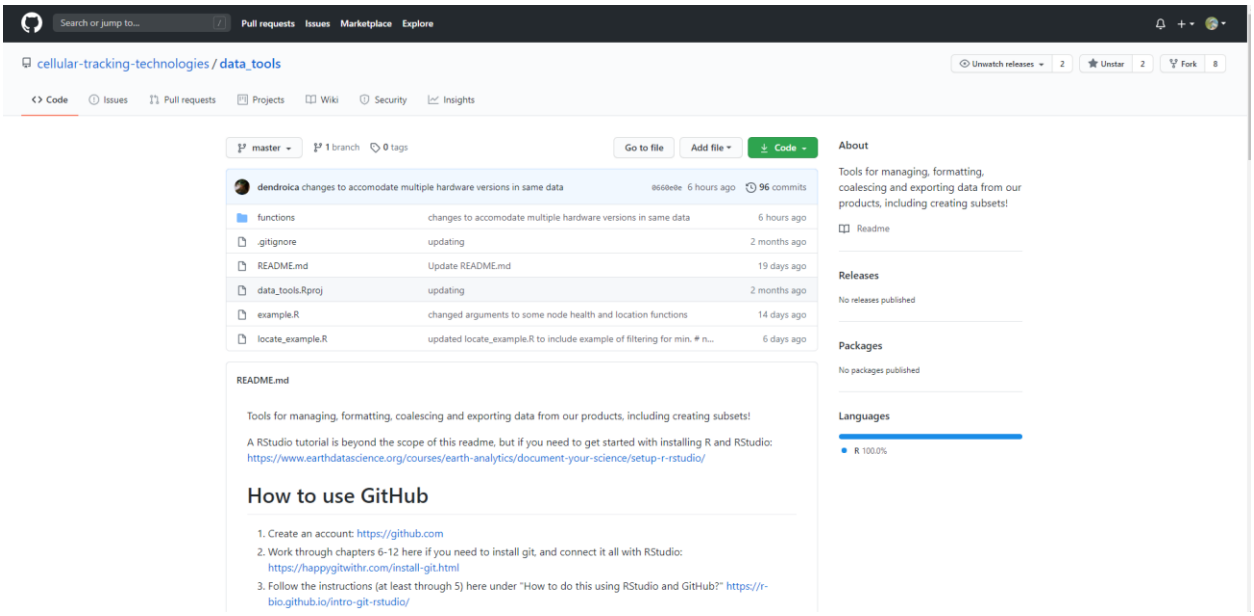


See if you get



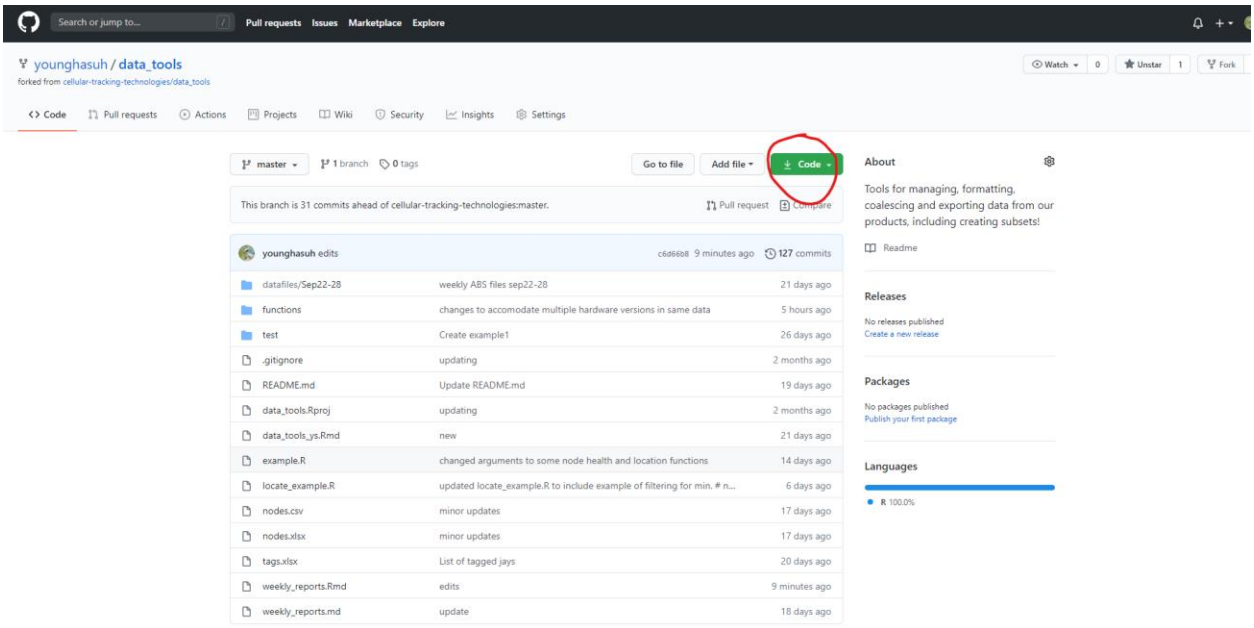
3. Getting started with CTT-Archbold repository

As a quick starter with how this works with the CTT repository (often called "repo" for short), their main functions and data tools are stored in a public repo: https://github.com/cellular-tracking-technologies/data_tools

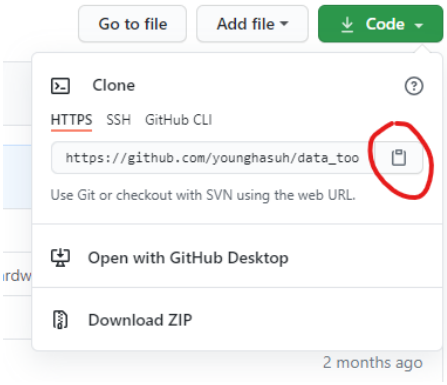


There is a README file that contains some basics but what is happening is that they have a source code (in the “functions” folder) and the users are able to "branch" from this "master" repository to get updates and such. What branching is doing is creating a sort of copy, that is linked to the master but available for us to edit, without directly affecting anything from the source (master). So, like in a tree, if something goes wrong in the branch (some fatal error) you could like chop off the branch without affecting the tree (master).

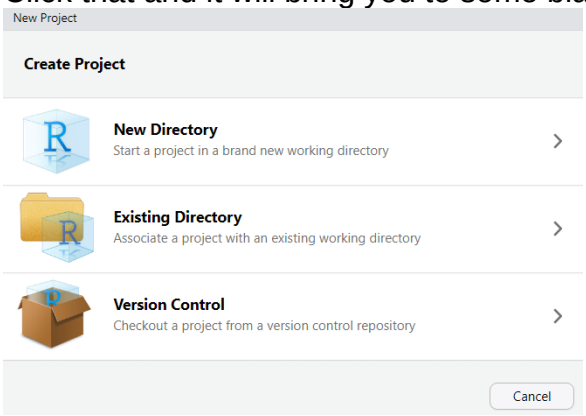
I created a branch from the CTT master which goes under my own repo and is available here: https://github.com/younghasuh/data_tools. Since we will be using the same code and possibly same edits, I figured it would be best for us to be collaborators on this branch and make edits on the same one instead of you creating a branch separately from the CTT master repo. An invite should be sent by email. Once you accept, you can make a clone of this on your local device.

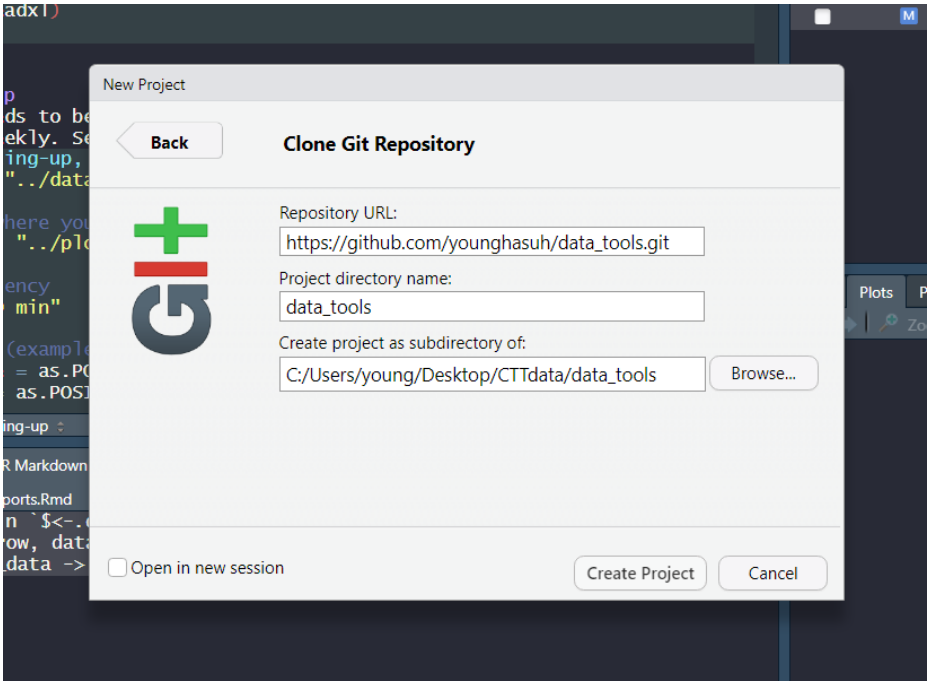


From my branch, click on the green code button. From there, click on the small clipboard button to get a copy of the HTTPS which you will use in R.



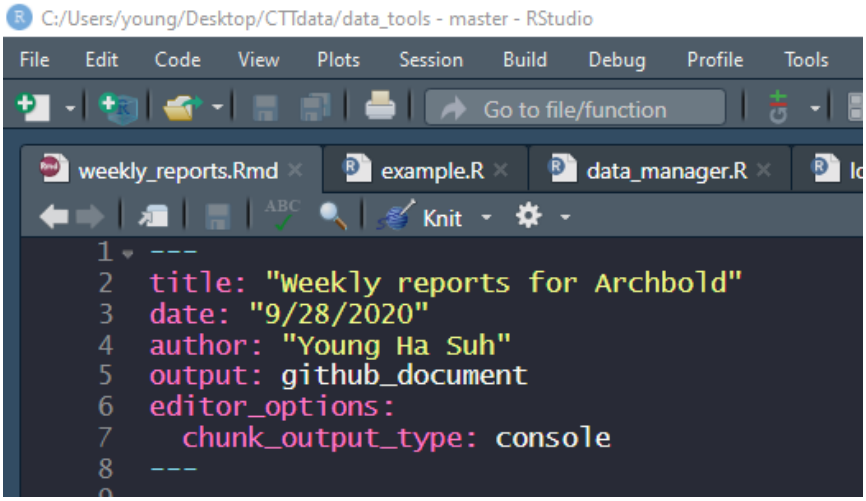
Now boot up R markdown, and click File > New Project. You will get three options, click "Version Control". Once you have Git downloaded, you should have an option "Git". Click that and it will bring you to some blank lines you can fill in.



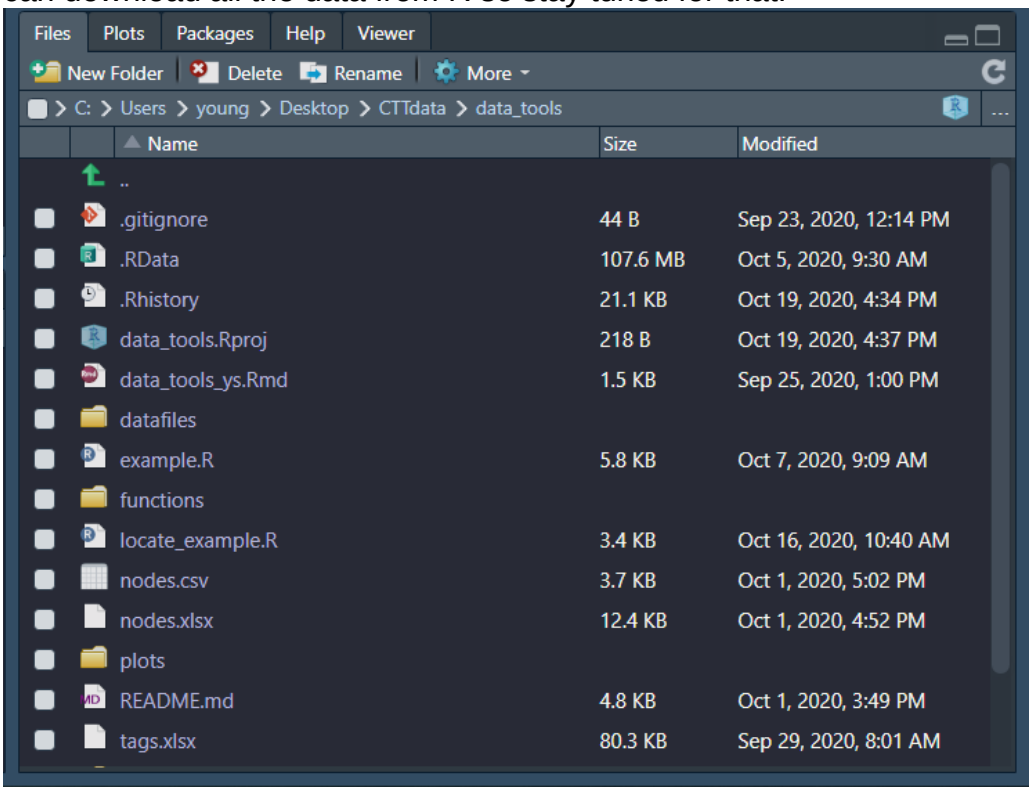


In the Repository URL, paste the line you just copied from above. Give it a name (you can also use "data_tools" like in the example above) and then set its directory wherever you would like. What this is doing is cloning my repo onto your laptop so you have the same files and folders. The main script I am working on is **"weekly_reports.Rmd"** which is a markdown file based on the “examples.R” file made by Jessica. Click on it in the Files tab on the bottom right to open this R markdown document.

I have been trying to be good at documenting this but let me know if things are not clear and feel free to add your own notes as well. Note that the output is set to be a github_document, meaning that we can open this file directly on GitHub pretty seamlessly. But this needs to be first **knitted**, which will produce a .md file that can be opened in GitHub. Note, knitting takes a while so I would do that step last if possible.

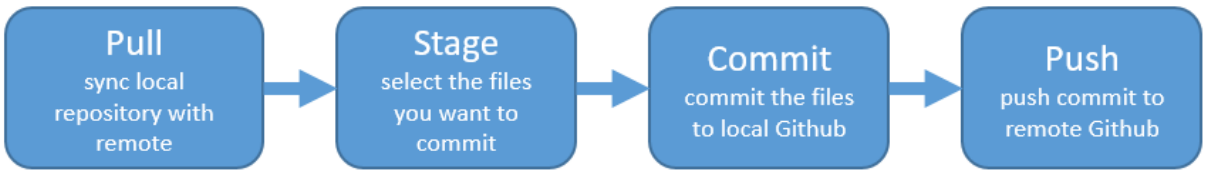


If you take a look at the console, you can see all the files in this folder. The "functions" folder is where all the CTT-generated code is, specifically functions for calculating node health or localization estimates. CTT is currently working to create an API key so we can download all the data from R so stay tuned for that.

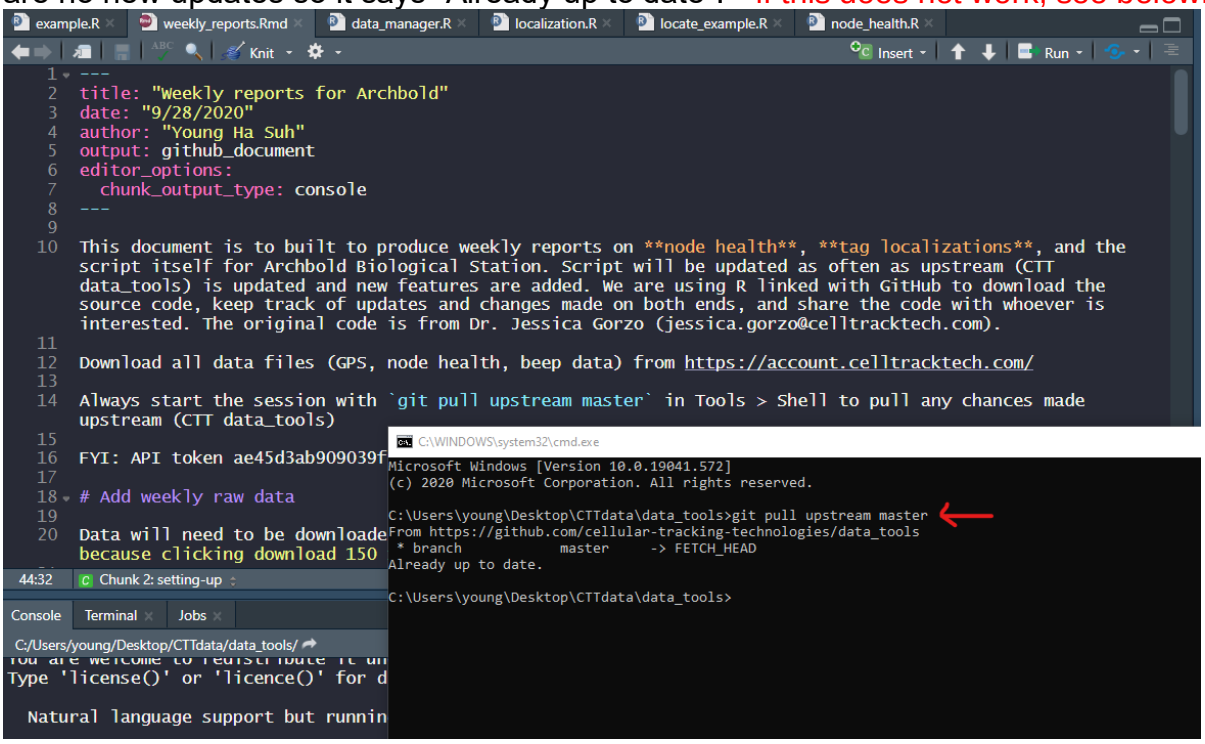


Now with multiple collaborators, the main issue that may arise is when multiple people work on this simultaneously and save different changes, which will result in a conflict. There are several ways to troubleshoot this but I think the easiest way is prevention -- the two ways that this can be done is through communication and making sure you start the session by pulling from the master and local, in case there are any changes others have made.

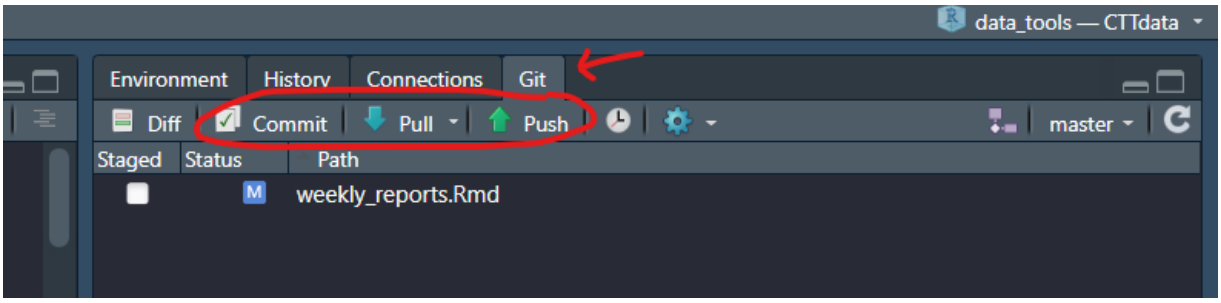
When I say pull, this is the first step in syncing your local RStudio to the remote GitHub. Essentially you are "pulling" or syncing any new changes to your local device from GitHub. As a quick note, the next step is "staging" which is selecting which files we want to sync to GitHub, then "commiting" in which you commit to sending these to GitHub, and finally "pushing" where you send the changes or commits to GitHub so that it is synced. Again, I understand this is super technical and jargon-heavy, I was super overwhelmed by this at first!



To make sure the version you are working on is the most up to date and to prevent any conflicts, it is best to sync your local Rstudio file to the one available on Github through two steps. First, in the **Tools > Shell** option, type `git pull upstream master` then press enter in the to make sure we are synced up with the master. The shell will open up a new window. If Jessica from CTT updates the code on the master (which is what we designate as upstream), you will see any updates here. In the example below, there are no new updates so it says “Already up to date”. * if this does not work, see below.

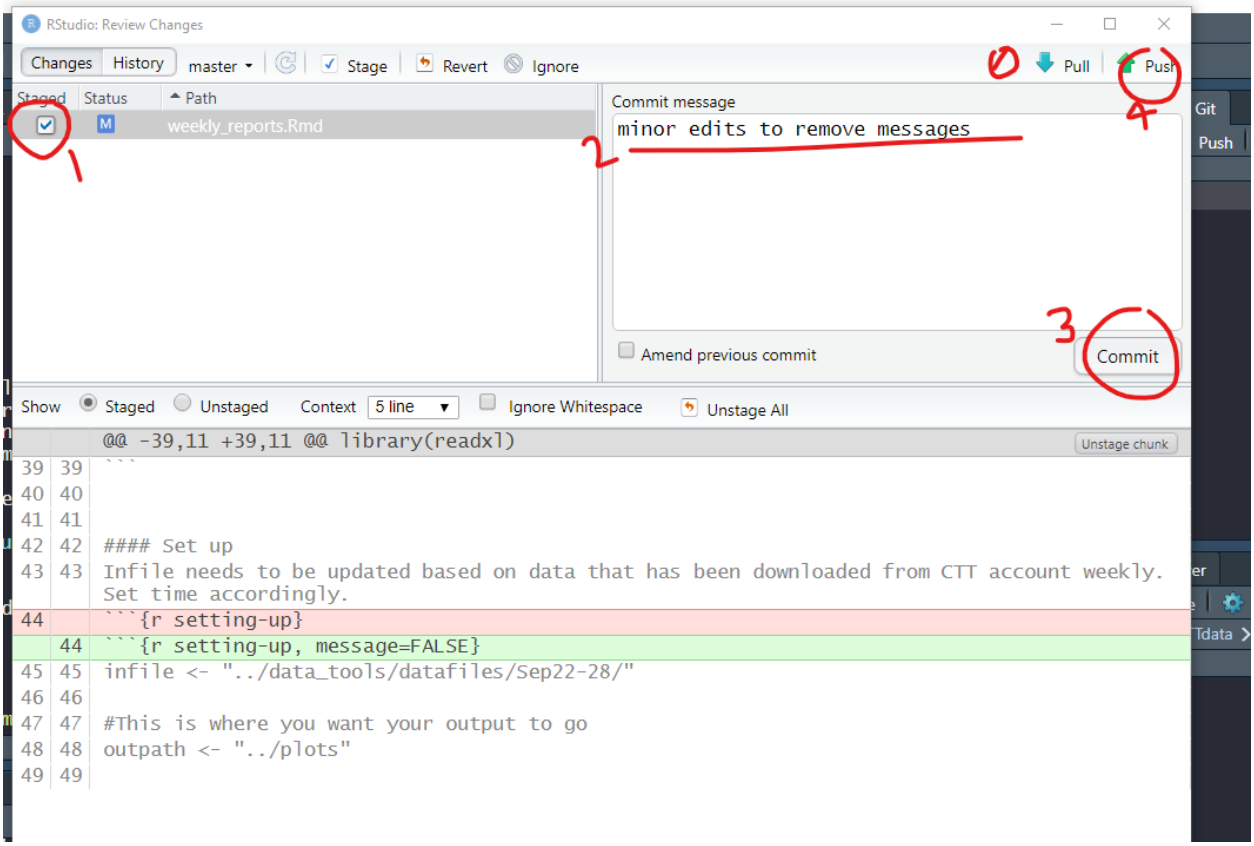


The second step is pulling in from our collaborative branch. Go to the Git tab on the top right corner. You will see several options to do this by clicking buttons. Again, the 4 steps of Git are Pull > Stage > Commit > Push and you can do all of that here. Click the blue down arrow to pull any changes made on our branch and make sure you are working with the most up to date synced version.



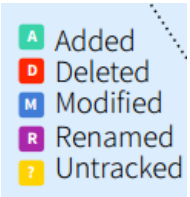
This is a crucial step! Make sure to develop a habit of always **pulling before you start** working on your project.

If you make edits to any documents in our data_tools folder, and **save** it, you will see the file pop up on that corner as well (in example above, "weekly_reports.Rmd" is available because I made an edit and saved it). If you want to send this to GitHub, click on the "Commit" button with the checked boxes. A new pop-up will appear and let you make the commits. Here you can see the edits made in the box below, with red lines indicating how it used to be and green showing the new edits – how cool is that!!!

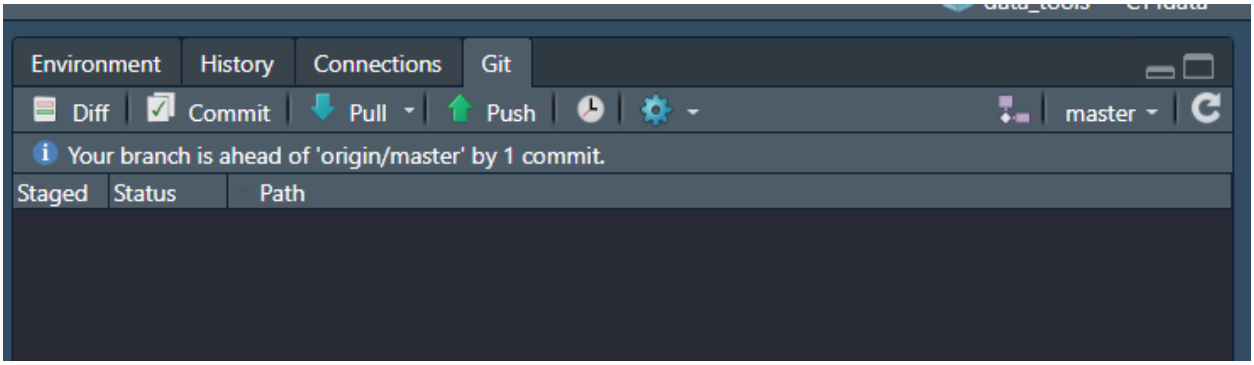


Here, you are given the option to pull again if needed (step 0). Check the boxes for the files you are commiting (step 1), then write a brief message to indicate what edits you are doing (step 2). Next you will "commit" (step 3), and then push (step 4). Note that once you hit "Commit", this page will be blank. That's normal -- just click "Push". Once that is done, you can close that pop-up window.

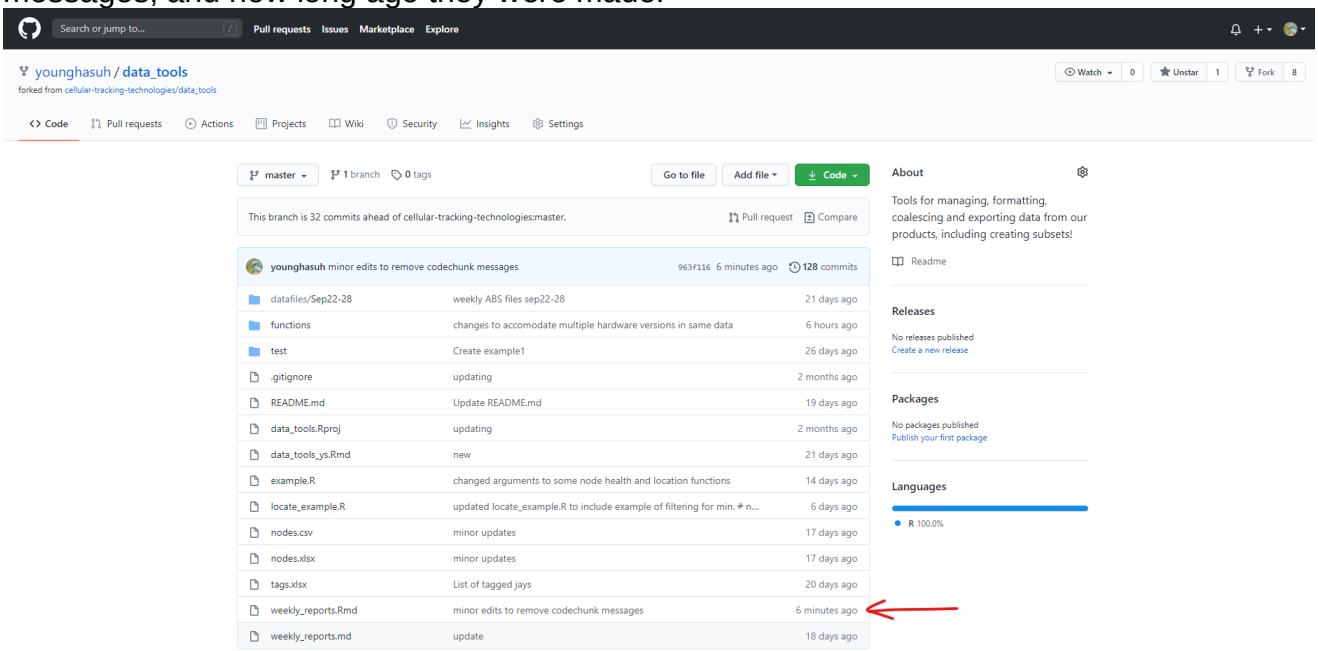
For staging, there are codes used to describe how the files are changed. “Staging a file” means that you are indicating that you want GitHub to track this file, and that you will be syncing it shortly. These are the codes used to describe how the files are changed:



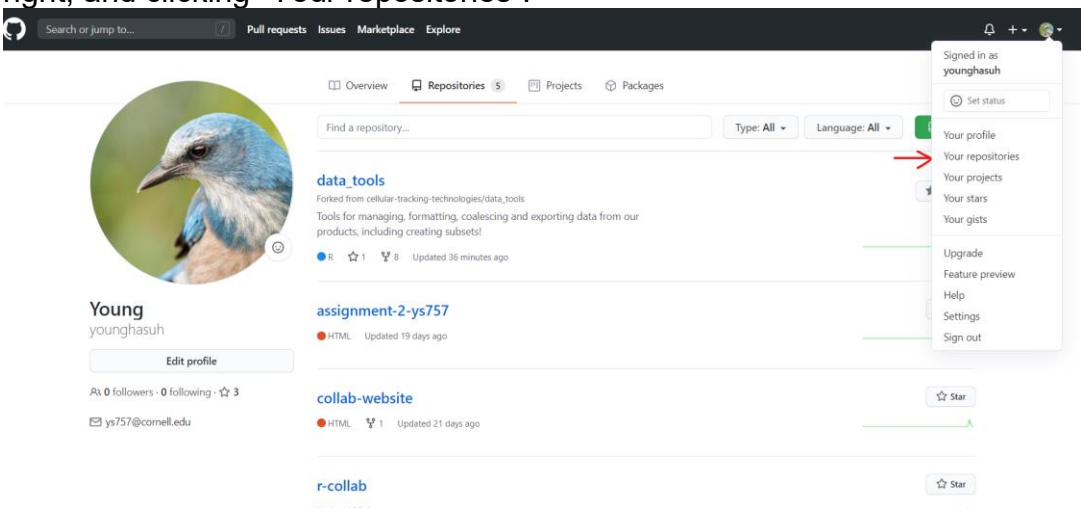
In the top right corner, you will now see that the object is gone and just a message saying that your local branch is ahead of origin/master by 1 commit, meaning that you made 1 change compared to the original.



Now, if you return to the GitHub and hit refresh, you will see the edits, the commit messages, and how long ago they were made!



If you have other repositories, you can access them by clicking on your icon on the top right, and clicking “Your repositories”.



*** Troubleshooting upstream pulls**

I am not entirely sure if this step is needed if you are just collaborating on my repo. But if it is causing some issues, this may help.

In R studio, go to Tools > Shell and type the following:

```
git remote add upstream https://github.com/cellular-tracking-
technologies/data_tools.git
```

This is changing the setting so that the upstream we are pulling from is the CTT GitHub. Verify if this upstream remote by typing the following in the shell again:

```
git remote -v
```

You should get something like:

```
origin    https://github.com/YOU/REPO.git (fetch)
origin    https://github.com/YOU/REPO.git (push)
upstream  https://github.com/OWNER/REPO.git (fetch)
upstream  https://github.com/OWNER/REPO.git (push)
```

Where OWNER is referring to the CTT repo and YOU is our shared repo. Once this is set up, hopefully you can use the following to pull changes from the CTT repo to our copy.

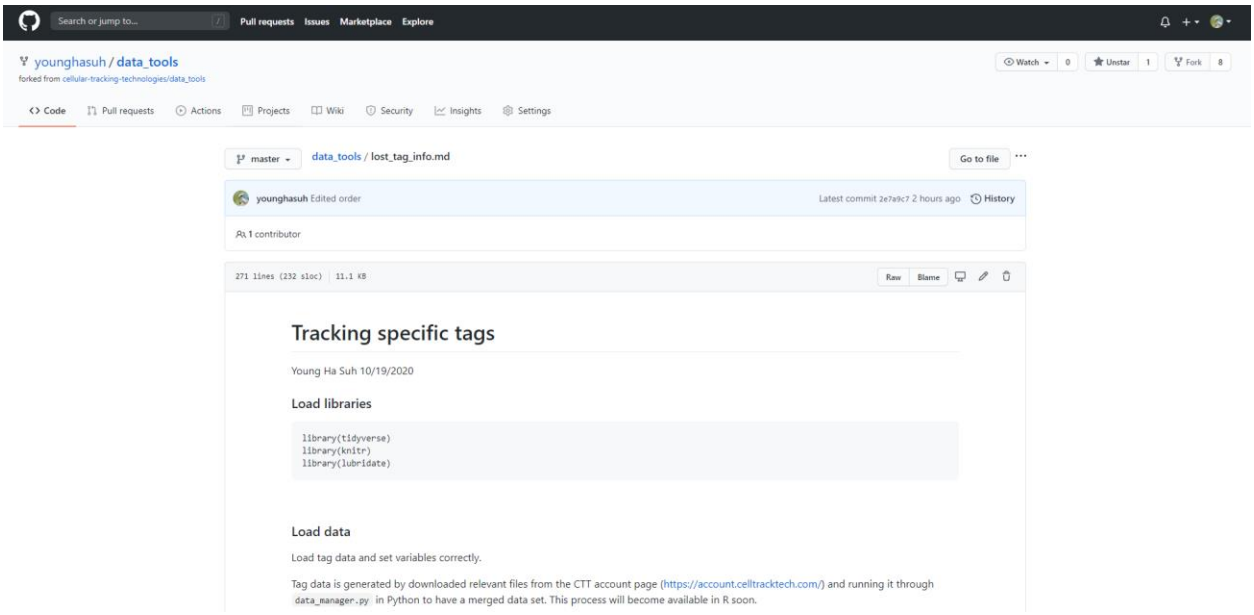
```
git pull upstream master
```

I will try to do this regularly so you won't need to do so.

4. File for locating tags: lost_tag_info.md

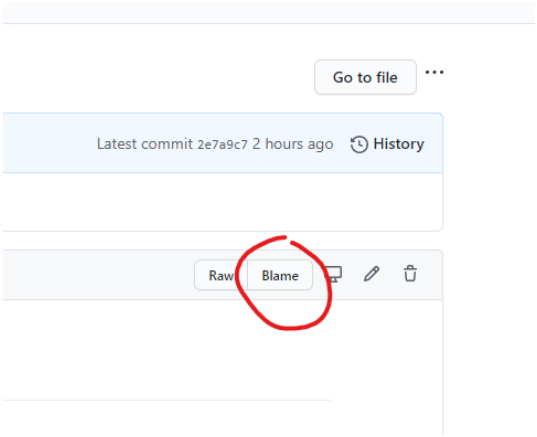
I created a short R markdown document that will give a summary of the node, RSSI, and time of day for specific tags. You can access it here: https://github.com/younghasuh/data_tools/blob/master/lost_tag_info.md

Once you pull the entire repo onto R, you will be able to edit this .Rmd document. As a reminder, markdown files can be opened on the GitHub page once you knit the document as a github_document, and push both .Rmd and .md files to GitHub. The page you see is a .md file. It also contains info on who contributed, whether it is the master branch, and the history of any edits (here, I edited 2 hours ago with the notes "Edited order").



I have edited so that it contains metadata for the files I load and a map of the combined nodes at Archbold.

The tag data is generated by first downloading the files from the CTT account page (https://account.celltracktech.com/) and running it through data_manager.py in Python to have a merged data set. This process will become available in R soon.



If you click “Blame” on the top right, it will show you the edits made along with the comments I left when I committed any changes on the document (effectively showing “who to blame” if there is an issue ha).