**Tailoring GitHub for ecology**

**Introduction**

Traditional field ecology is currently experiencing a data revolution. It is becoming more apparent that ecologists need to write code as part of their field, lab and modeling research (Mislan et al., 2016). The need to write efficient code, store the code, and allow for other researchers to reproduce the analyses for the project is becoming a growing concern. Environmental crises have also propagated ecologists to a rapid shift into “big science” ecology (Hampton et al., 2013). At this time, there is a lack of training in basic data management, curation, and workflow design among scientists (Lowndes et al., 2017). A recent NSF (National Science Foundation) survey (Lowndes et al., 2017) demonstrated that of the 704 scientists who participated in the survey, “data skills” (e.g. multi-step workflows, ability to store, share and publish data) was identified as the largest unmet need (Barone, Williams, and Micklos, 2017; Lowndes et al., 2017).

The LCR (Lone Cabbage Reef) project is a large ecological effort to restore an oyster reef to historic elevation levels so that it may endure sea level rise and river discharge. Our project generates data from multiple sources including observations of oyster populations and measurements by field biologists and continuous autonomous water quality data via sensors. These data are updated at different frequencies and require specific attention to be processed (Moreno et al., 2020 hopefully). However, once they are processed there is a need to store these data so that they made be used among project team members and collaborators. We use GitHub version control software to keep track of regularly updated data, and to keep the multiple working projects using these data organized.

This paper describes our approach at standardizing file naming conventions, GitHub repository structures, and our repository workflow to streamline the availability of data to differing LCR projects. Our main concerns were for different team members to have access to the most up-to-date data relating their projects, without having to constantly e-mail or ask for the data.

#### Box. 1 Terminology

big science- originally termed by Alvin Wienberg in 1961, as a sizeable investment, involving international and collaborative efforts among scientists (Wienberg, 1961)

version control- a system which allows the users to track iterative changes to code and text (Blischak et al., 2016)

project repository- term used to identify one type of analysis that is being conducted on a LCR project dataset

#### End of Box 1

**GitHub and version and control**

**GitHub permissions and branch workflow**

**Challenges working in one repository**

As the LCR project started to generate data, it became apparent that its GitHub structure was becoming increasing more difficult to maintain. One of the main complaints was that it was difficult to find files and locate their data source. Collaborators working in a single GitHub repository were not always following repository guidelines, however the guidelines at the time did not address many of our newfound needs such as how to account for multiple working project repositories. Without having proper guidelines, our GitHub repository quickly began to grow and expand. Our main active repository started to grow projects inside of itself, leading to a confusion of what was in the repository was which data and scripts were used for the different projects. We soon came to realize that the GitHub repository structure we had employed was not effective in keeping our files or projects organized.

**Creating a new GitHub repository structure and workflow**

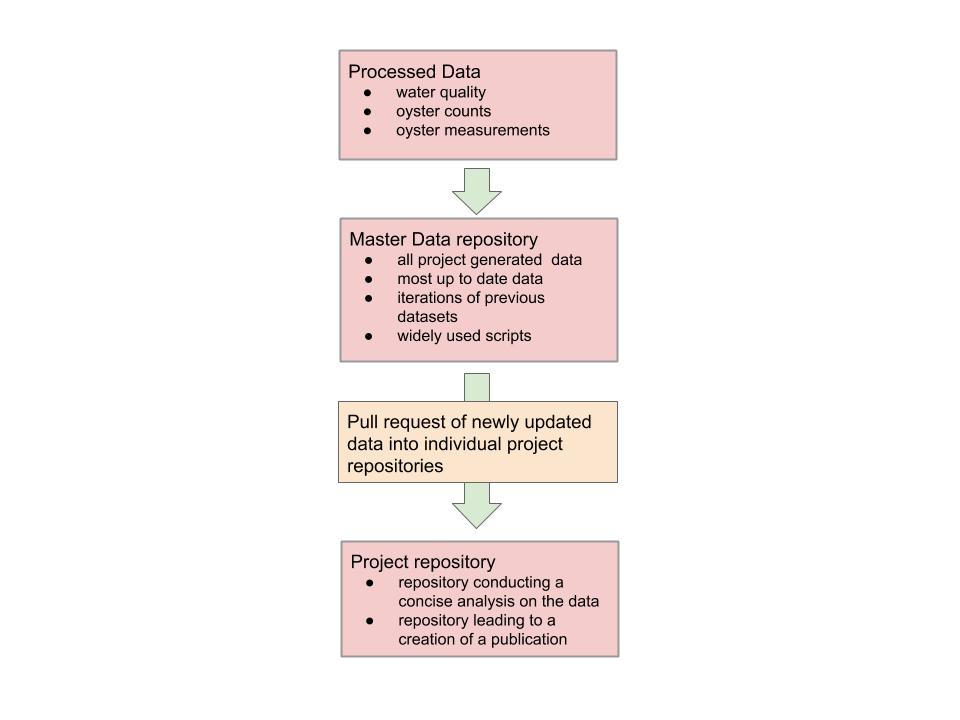


Figure 1- new workflow

**Naming conventions for repository, files, and folders**

Proper file naming conventions helps the user understand the contents inside the file. For coding scripts naming conventions exists if the file creates a function or a certain output the file should also be named that way (<https://style.tidyverse.org/package-files.html#names-1>). For our project we created a set of consistent set of guidelines to name files.

|  |  |  |  |
| --- | --- | --- | --- |
| Type | Naming Convention | Example | Definition |
| Project Repository | study\_location\_projectsummary | bird\_bb\_monitoring | Big Bend camera and survey bird monitoring project |
| Scripts | lowercase, no uppercase (camelcase) nor all caps, all names with separate words need to include a underscore ( \_ ) and no spaces, no dates in the names unless it helps with the descriptions of the conten script file names should be descriptive and concise. Scripts that have a single output should be named in a similar fashion to its filetype output. | discharge\_1941\_2018\_quantile.R | R script which reports quantiles from river discharge from 1941 to 2018 |
| Figures | study\_location\_type\_summary.filetype | oys\_lco8a\_map\_transect.tiff | oyster transect on reef element LCO8A map in a tiff image |
| Tables | study\_location\_summary.filetype | wq\_lcr\_inshore\_vs\_offshore.csv | LCR water quality inshore and offshore comparison |
| Data | every dataset file is required to be in lowercase, no uppercase (camelcase) nor all caps, all names with separate words need to include a underscore ( \_ ) with **no spaces**, no dates in the names unless it helps with the descriptions of the content | discharge\_1941\_2018.csv | River discharge data from 1941 to 2018 in a text file |

Figure 2- Table of naming conventions