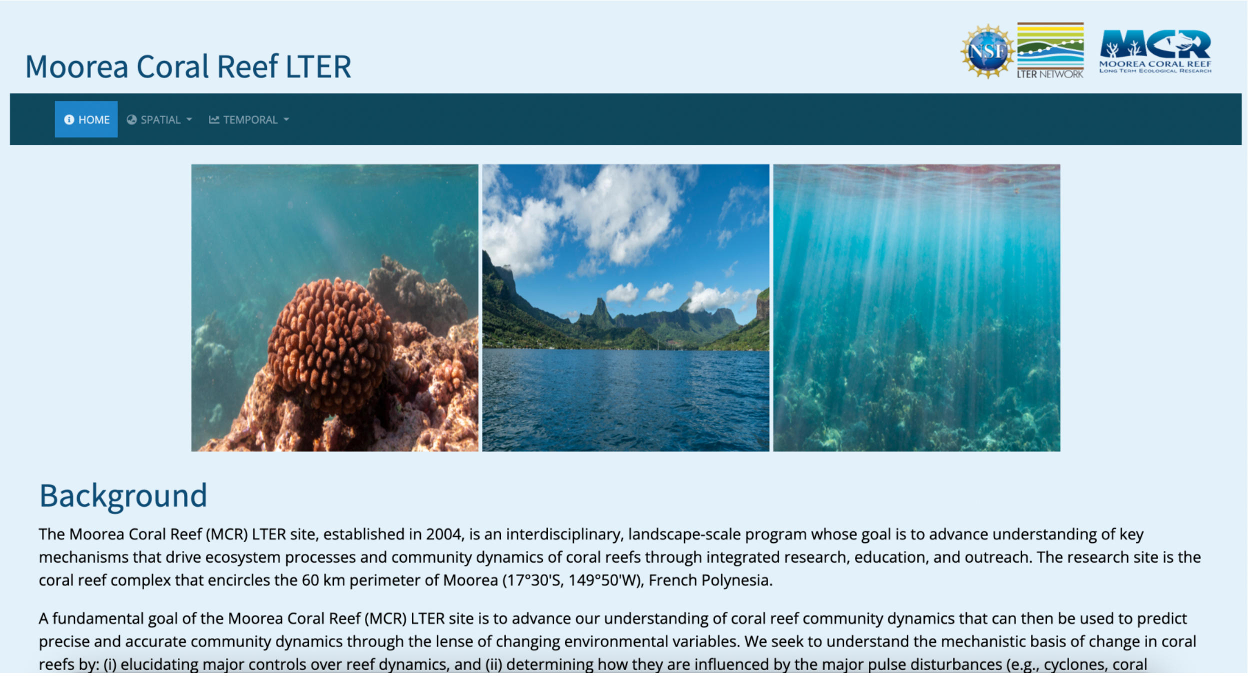
Shiny App User Guide

Visualizing Spatial and Temporal Patterns of Coral Reef Stressors Surrounding Moorea, French Polynesia



Background

The Moorea Coral Reef (MCR) LTER site, established in 2004, is an interdisciplinary, landscape-scale program whose goal is to advance understanding of key mechanisms that drive ecosystem processes and community dynamics of coral reefs through integrated research, education, and outreach. The research site is the coral reef complex that encircles the 60 km perimeter of Moorea (17°30'S, 149°50'W), French Polynesia.

A fundamental goal of the Moorea Coral Reef (MCR) LTER site is to advance our understanding of coral reef community dynamics that can then be used to predict precise and accurate community dynamics through the lense of changing environmental variables. We seek to understand the mechanistic basis of change in coral reefs by: (i) elucidating major controls over reef dynamics, and (ii) determining how they are influenced by the major pulse disturbances (e.g., cyclones, coral

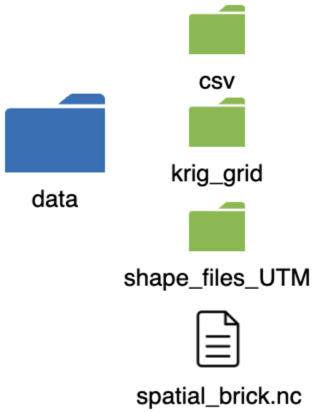
Table of Contents

User Documentation

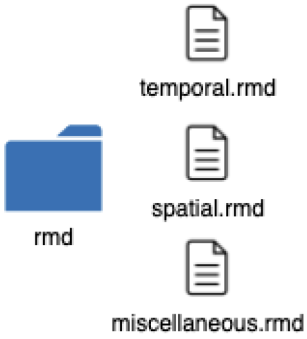
Data Management

Infrastructure/file structure

- data
    - csv: contains all raw datasets needed to produce the spatial layers, leaflet map, and temporal visualizations
    - krig_grid: contains the 5x5 meter gridded spatial pixel polygon of the reef boundary used for the kriging process
    - shape_files_UTM:
        - clipped_reef_bound: contains the shapefile for the reef boundary with the island clipped from the center
        - moorea: contains the shapefile for the island boundary
    - spatial_brick.nc: contains the 9 spatial layers as rasters

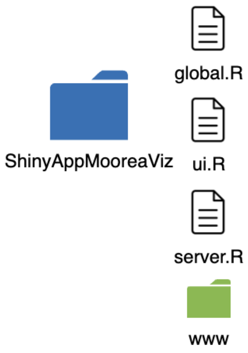


- rmd
    - temporal.rmd: includes all code to clean, tidy, subset, and join various temporal datasets in order to produce the plots for coral cover, macroalgae, fish biomass, and Crown-of-Thorns for the Shiny app. The final joined dataset is written to a csv file to be called into the Shiny app within the global.R file. This Rmd file is used to make sure the data are wrangled correctly and also includes code for exploratory plots that can be used in the app.rmd file to build the visualizations on the Shiny app.
    - spatial.rmd: Includes all code to clean, wrangle, and krige the spatial data to produce rasters. There is code to create the six LTER sites' bounding boxes and apply all visualizations to an exploratory leaflet map.
    - miscellaneous.rmd: Includes miscellaneous code useful for initial data exploration, plots, and visualization attempts. Includes code to make and clip the gridded box used for the kriging process, write and save shape files, and make explanatory ggplots with the island outline. Everything is in UTM.



- figs
    - leaf.html: contains the leaflet map
- ShinyAppMooreaViz
    - global.R: loads all packages and datasets used within the app

    - ui.R: contains code for the user interface, including app layout and user inputs
    - server.R: contains code for making reactive dataframes where necessary, plot outputs, and anything pertaining to producing the visualizations based on user inputs
    - www: contains all photos used within the app and a .css script to do custom theming for the app. The www directory is the directory expected by a Shiny application to locally store the elements that will be rendered in the web browser and are not the output of the scripts. It must always have this name.
- island: contains the shape files for all of the surrounding islands in the French Polynesia island complex
- island_shape: contains the shape file for just the island of Moorea



Datasets and Metadata

The spatial data sets present in the Shiny application are derived from multiple survey methods. The bathymetry data for the lagoon was collected using airborne operated LiDAR and compiled into an XYZ data file. The predicted sewage data was calculated through modeling conducted by fellow researchers and stored as a spatial csv file. The percent nitrogen and isotopic data was collected through *Turbinaria ornata* samples approximately evenly distributed throughout the surrounding lagoon and stored as a spatial csv file. The coral bleaching data was collected through visual surveys conducted by divers. Divers would enter the water at anchor point locations where *Turbinaria ornata* was collected and swim in a compass heading visually estimating the percent of each coral bleached along the compass heading and stored as a spatial csv file. All spatial csv files were then krigged using ordinary kriging from the evenly distributed data points across the 4 spatial data sets.

The temporal data sets present in the Shiny application are surveyed on MCR LTER long-term survey sites. Across the six long-term sites there are permanent transects that are surveyed annually. The four temporal variables, percent coral cover, Crown-of-Thorns density, herbivore fish biomass density, and percent macroalgae cover have variable survey methods at the permanent transects. The four survey methods are conducted on or around the permanent transects in order to gather an accurate representation of the biota present on that portion of the reef during that survey period.

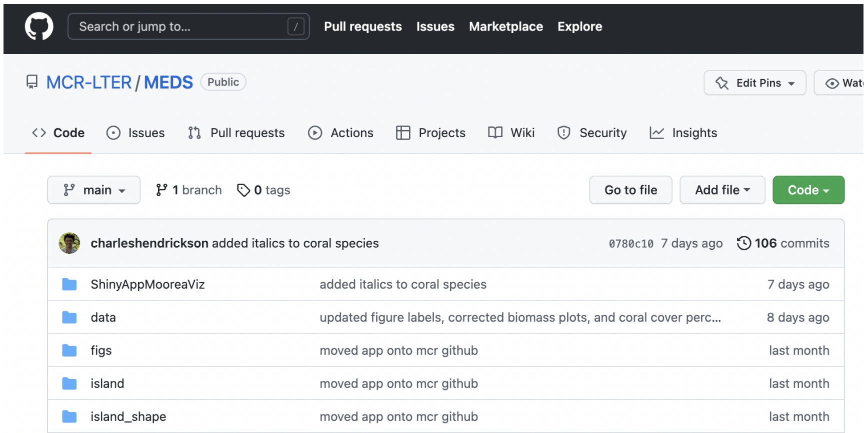
All associated metadata for the public data sets can be found on the MCR LTER website data repository. There is also metadata included within the Shiny app itself.

Code Management

App maintenance

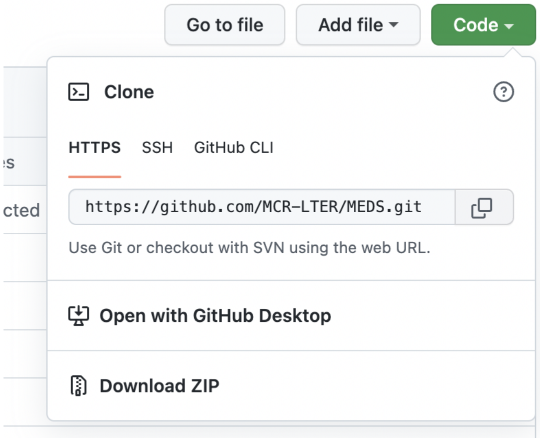
- All changes to the app should be pushed to GitHub. Instructions on how to push and pull to git and how to create a GitHub account can be found below.
    - How to create a Github account
        - Great instructions can be found on the GitHub website

- Connecting your Github to your R and cloning the repository
    - To get started working in the repository you need to clone the MCR-LTER/MEDS repository and add it to your local machine. This is done by clicking the green code button on GitHub and copying the repository.



charleshendrickson added italics to coral species 0780c10 7 days ago 106 commits

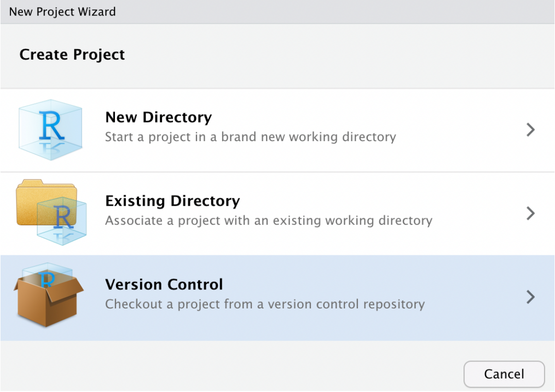
ShinyAppMooreaViz	added italics to coral species	7 days ago
data	updated figure labels, corrected biomass plots, and coral cover perc...	8 days ago
figs	moved app onto mcr github	last month
island	moved app onto mcr github	last month
island_shape	moved app onto mcr github	last month

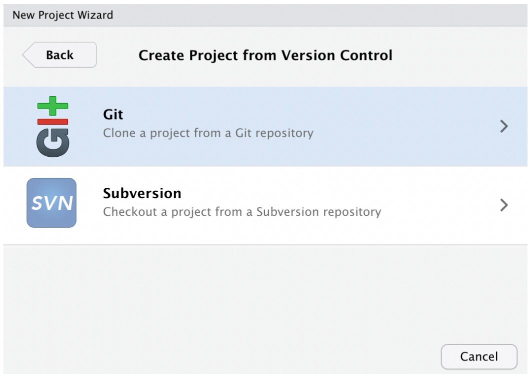


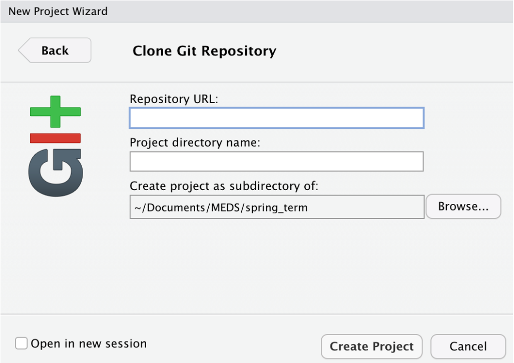
https://github.com/MCR-LTER/MEDS.git

Use Git or checkout with SVN using the web URL.

    - Once you have that copied you can go to RStudio, create a new repository, and choose the Version Control option. The page that pops up is where you can paste the repo link. Here you can also decide where on your local machine you want this repository folder to live.

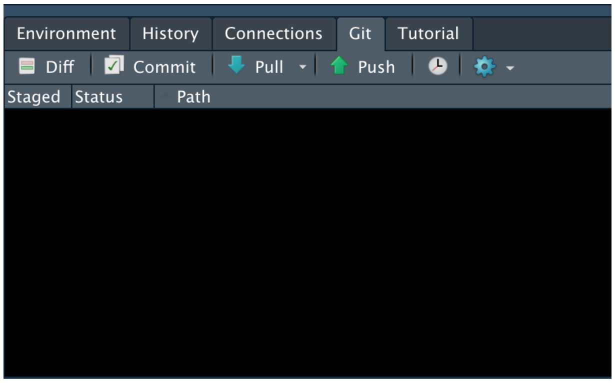

New Project Wizard
Create Project
New Directory
Start a project in a brand new working directory
Existing Directory
Associate a project with an existing working directory
Version Control
Checkout a project from a version control repository
Cancel


New Project Wizard
Back
Create Project from Version Control
Git
Clone a project from a Git repository
Subversion
Checkout a project from a Subversion repository
Cancel

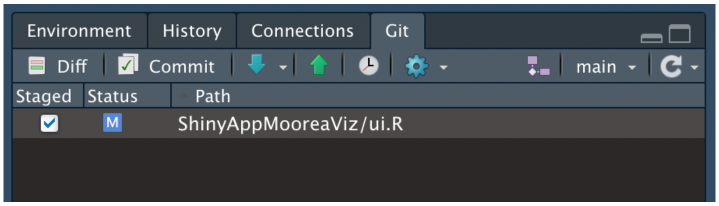


Create project as subdirectory of: ~/Documents/MEDS/spring_term

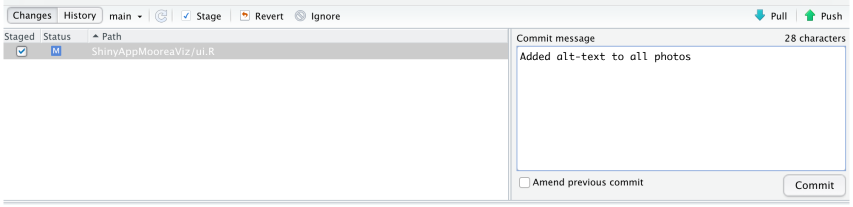
        - Once everything is named and in the desired location, create the repository and you will now be connected to Git.
    - How to push and pull from Git
        - From R
            - Once you have cloned the repo and created a version controlled repository in R (steps above) you should see a Git panel in the top right corner of your screen.



            - Any files that have changes you have made (save first!) will show up under that Git panel. Select the boxes of all the ones you want to push to GitHub.



- Then hit commit and type out a message to go along with your commit. This message will show up on Github explaining the changes you made.



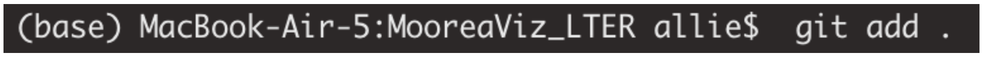
Commit message — 28 characters

```
Added alt-text to all photos
```

- Once you have committed, pull changes, and then push your changes, using the blue and green arrows in the photo above

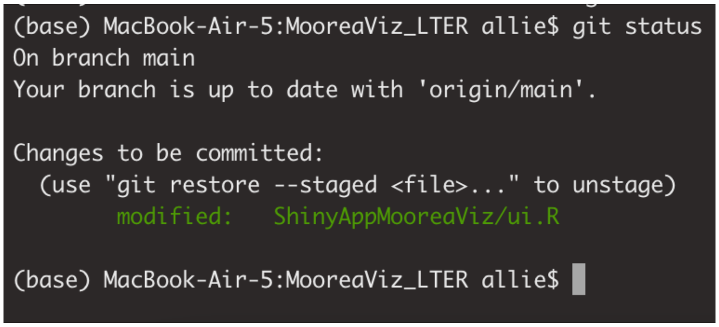
■ [From terminal]

- If you are more comfortable working in the terminal you can push and pull from there. This can be done from the Terminal in R or the terminal on your computer, the commands are the same
- The photos here are using the R studio terminal.
- Start by using ` git add .` this will select all the files that have changed (similar to selecting all the boxes in the workflow above) or you can type the name of the specific file you would like to add. IF done correctly it will look like nothing has happened.



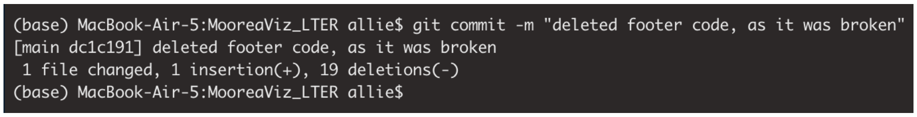
```
(base) MacBook-Air-5:MooreaViz_LTER allie$  git add .
```

- Next, verify that you have the files you need by typing `git status` If the files you want are ready you can move on.



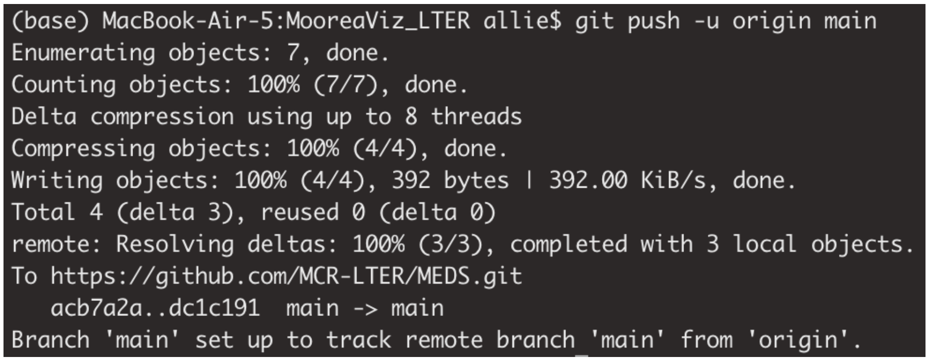
```
(base) MacBook-Air-5:MooreaViz_LTER allie$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   ShinyAppMooreaViz/ui.R

(base) MacBook-Air-5:MooreaViz_LTER allie$ 
```

- Now you are ready to commit. `git commit -m "comment"` where the “comment” is the message explaining your commit



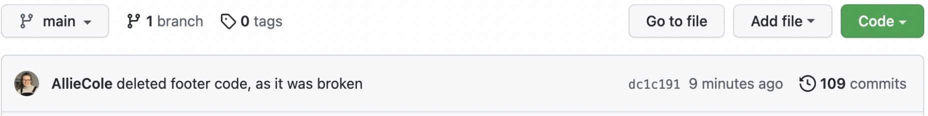
```
(base) MacBook-Air-5:MooreaViz_LTER allie$ git commit -m "deleted footer code, as it was broken"
[main dc1c191] deleted footer code, as it was broken
 1 file changed, 1 insertion(+), 19 deletions(-)
(base) MacBook-Air-5:MooreaViz_LTER allie$
```

    - Now that everything is committed you can push by using `git push -u origin branch_name`



```
(base) MacBook-Air-5:MooreaViz_LTER allie$ git push -u origin main
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 392 bytes | 392.00 KiB/s, done.
Total 4 (delta 3), reused 0 (delta 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To https://github.com/MCR-LTER/MEDS.git
   acb7a2a..dc1c191  main -> main
Branch 'main' set up to track remote branch 'main' from 'origin'.
```

- Versioning and History with Git
    - If you need to access an older version of your code you can access all the previous commits through Github. Click the clock icon that tell you the number of commits.



AllieCole deleted footer code, as it was broken dc1c191 9 minutes ago 109 commits

    - Here you will see all the commits that have been made and you can click on one to access the code as it was at the time of that specific commit.

- Info from GRIT as to how to deploy an update
    - Message Arron Martin via GRIT to deploy an update, grit-zammad@ucsb.edu

Adding new data

Integrating new or updated data into the Shiny app requires downloading the new data into the data folder, reading in the data into the RStudio environment, and formatting and tidying the data before it can be visualized on the Shiny app.

New Spatial Data

- First, the new data must be downloaded into the R project.
    - Download the new data into the `csv` folder, which is within the `data` folder.
    - Spatial data should be .csv files, while LiDAR data should be .xyz files.
    - The new data should be labeled with a short, descriptive name and the year of collection (Ex: name_year.csv)
- Second, the new data must be read into the RStudio environment from the csv folder.
    - Navigate to the `ShinyAppMooreaViz` folder, which contains the `global.R` script.
    - Data will always be read in at the top of the `global.R` script.
    - The script's documentation will show which lines of code are used to read in specific data types, such as nitrogen, bleaching, or sewage data.

Example:

    - To read in nitrogen data, navigate to the documentation called `nitrogen data`.
      - #nitrogen data
    - Then, input the complete file name into the file path after `data` and `csv`.

      -
        ```
        nitrogen_data <- read_csv(here("data", "csv", "name_year.csv")) %>%
        clean_names()
        ```
    - This code will create a data frame named `nitrogen_data` that will be called upon later in the Shiny app script.
  - Once all the files are set up to be read in, it is time to format and tidy the data.
- Third, the data must be formatted with the correct column names and row values. Additionally, the data frame must be formatted into Tidy format before it can be used in the Shiny app.
  - The columns within the new data frames must have the same names as the previous data.
  - The rows within the new data frames must have the same units as the previous data.
    Example:
    - Column names must be kept the same (percent_n_jan, percent_n_may, percent_n_july) . However, we use the `mutate()` function to modify the rows in these three columns so that they contain values in percentages, not decimals.

      -
        ```
        n_data <- nitrogen_data %>%
        mutate(percent_n_jan = percent_n_jan *100,
                 percent_n_may = percent_n_may *100,
                 percent_n_july = percent_n_july *100)
        ```
  - Untidy data must be wrangled into Tidy format, where each observation is a row and each column is a variable.
    Example:
    - The “nitrogen_data” data frame was not in tidy format so two columns named `type` and `percent_n` were created to contain the different nitrogen samples (percent_n_jan, percent_n_may, percent_n_july) and the percentages for those data types respectively.

      -
        ```
        pivot_longer(!1:5, names_to = "type", values_to = "percent_n") %>%
        separate(type, into = c("method","random", "date"), sep = "_") %>%
        dplyr::select(-random)
        ```
- Once the data frames are formatted correctly, the Shiny app can be run with the new data.

New Temporal Data

- To add new temporal data, the existing individual raw data csv files should be appended in a way that preserves the current format (i.e., all columns should have the same names and order).
  - The new files will then need to be placed in the `csv` folder within the `data` folder, replacing the outdated raw dataset using the *exact same name*.
- Rerun the entire Rmarkdown file (temporal.Rmd) to produce and export the updated combined temporal csv (temporal_data_joined.csv).

    - Note: Delete the existing temporal_data_joined.csv before running the entire Rmarkdown file to avoid issues with overwriting the pre-existing file.
- Since the temporal_data_joined.csv is already sourced in the global.R file within the `ShinyAppMooreaViz` folder, you do not need to edit anything within the Shiny app code files, aside from updating plot titles, labels, etc. that come with a new year's data (i.e., updating the plot subtitles to say Moorea French Polynesia, 2006-2022).
- Re-run the Shiny app and all temporal visualizations should be updated.
- Special notes
    - All reactive dataframes need to be followed by a set of parentheses when they are called into the renderPlot functions (i.e., `temporal_reactive_df()`)
    - All inputIds and plotOutputs used within the Shiny app need to have unique names

Archive Access

The Shiny application script, spatial script, temporal script, and this User Guide will all be archived on the Moorea LTER organizational GitHub repository. All data used for this project can be found on the public GitHub repository MCR-LTER/MEDS. The Shiny app will be hosted on the Bren School server for up to six months until it can be moved over to the UCSB GRIT server. Here is the link to the currently deployed version of the Shiny app: https://shinyapps.bren.ucsb.edu/ShinyAppMooreaViz/