2021-06-18

INTERMOUNTAIN WEST LICHEN DNA REFERENCE LIBRARY

Brian Colgrove

Honors Thesis

INTERMOUNTAIN WEST LICHEN DNA REFERENCE LIBRARY

by
Brian Allan Colgrove

Submitted to Brigham Young University in partial fulfillment of graduation requirements for University Honors.

Department of Statistics
Brigham Young University
June 2021

Advisor: Steve Leavitt

Honors Coordinator: Steven Peck

ABSTRACT

INTERMOUNTAIN WEST LICHEN DNA REFERENCE LIBRARY

Brian Allan Colgrove

Statistics Department

Bachelor of Science

Accurate estimates of biodiversity can play crucial roles in monitoring ecological health. BYU's Lichen Air Quality Biomonitoring program (Wright) represents one of the largest biomonitoring programs in the nation. Recently, DNA metabarcoding approaches have shown promise in streamlining lichen biodiversity inventories. However, to date, lichen diversity of western North America is poorly represented in available DNA reference libraries, preventing biologists and land managers from using DNA barcoding to identify unknown specimens. To solve this problem, I have developed a DNA reference library for over 500 species occurring in the Intermountain West region. Using bioinformatic and statistical tools, I created a user-friendly database that integrates with broad, world-wide initiatives. My research will be pivotal in moving BYU's Lichen Air Quality Biomonitoring program forward and help secure ongoing support from the US Forest Service.

ACKNOWLEDGMENTS

I want to acknowledge the work that Dr. Leavitt has put in helping me with this project. His knowledge of lichens and patience with me as a non-biology major has greatly moved this project along.

TABLE OF CONTENTS

LIST OF TABLES AND FIGURES

Introduction

Background

DNA barcoding can facilitate rapid and effective identification of specimens, providing crucial insight into biodiversity and ecological processes. Comprehensive DNA reference libraries are fundamental to successful specimen identification through DNA barcoding. In the last decade, impetus has been placed on developing large, worldwide DNA barcoding initiatives, e.g., BOLD (Barcode of Life Database) and the fungal-specific Unite database, which have important roles in a wide range of biological research. Despite the impressive growth of DNA reference libraries and other publicly available reference sequences, significant portions of species diversity are not represented by DNA barcodes. DNA barcoding is a method of identifying species based on a short DNA sequence. DNA barcode sequences are produced by isolating DNA from a sample and using PCR to amplify the barcode region. The DNA barcodes are then used along with reference libraries to identify specimen. BYU's Lichen Air-Quality Biomonitoring Program is one of the largest biomonitoring programs in North America and has documented impressive diversity in the Intermountain West region. While these species are represented by vouched collections housed in the Herbarium of Non-Vascular Cryptogams, a significant proportion are not yet represented by DNA sequence data.

Objective

The objective of this research project is to compile a dynamic DNA reference library for the lichens of the Intermountain West region. Specifically, I aim to incorporate over 5,000 DNA sequences representing over 500 species of lichen-forming fungi into a region-specific DNA barcoding reference library. This project will fill an important gap in effectively characterizing biodiversity and will be the foundation for the continued growth and use of BYU's Lichen Air Quality Program for assessing the ecological health of the Intermountain West lichens being held in the Bean Museum.

Review of Critical Literature

OLICH

The literature that was most critical to this research was one done by Dr. Marthinsen and her team in Norway. In her research paper *OLICH: A reference library of DNA barcodes for Nordic lichens*, she details how she created a DNA reference library for the lichens collected in Norway. The lichens used in her DNA reference library came from freshly collected samples and curated specimens in the University of Oslo herbarium (Marthinsen). Her DNA reference library contains over 1324 sequences and a total of 507 differing lichen species (Marthinsen). Her DNA reference library column names set the basis for our DNA reference library, which includes the voucher number, unique DNA ID, taxonomy DNA barcode sequence, and collection data. OLICH's paper produced the lichen DNA reference library, a map of collection sites, and observations made from the lichens' phylogenesis.

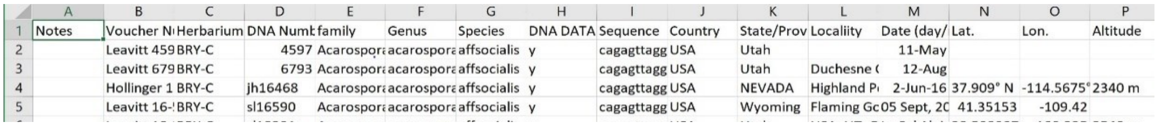
R Shiny

RStudio has created a way to create an app by coding in R called "R shiny". Their website has tutorials and sample code to start learning how to write in R shiny (Learn Shiny). I taught myself R shiny by using their tutorials and examples in order to create my own app that can host our DNA reference library. Furthermore, I used two GitHub repositories to learn how to use the package “DT”, which creates a R interface to JavaScript library “DataTables” (McLellan). Both repositories give examples on how to create an editable table that is user-friendly. By utilizing the tutorials and GitHub repositories, I was able to create a new and interactive way of accessing the DNA reference library that non-programmers can view and edit (Rstudio). The literature on R shiny and the GitHub repositories were critical to being able to create a dynamic DNA reference library.

Methods

Structure

A lichen DNA reference library is a dataset that contains curated lichen DNA sequences. The dataset contains columns of the needed information, and the rows contain a specific sequence and unique ID. A DNA reference library should contain all information that would be needed in future projects such as those that deal with taxonomy or phylogeny. Therefore, the foremost step in creating a DNA reference library is to have a solid structure that contains all the important information. As previously stated, I based the Intermountain West Lichen DNA reference library structure on the OLICH DNA reference library structure. The OLICH DNA reference library based their library off of Barcode of Life Database also known as BOLD. Our DNA reference library is similarly based off BOLD but ours does not contain specimen images. Table 1 below contains the structure of our lichen DNA reference library.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	Notes	Voucher N	Herbarium	DNA Numb	family	Genus	Species	DNA DATA	Sequence	Country	State/Prov	Localiity	Date (day/	Lat.	Lon.	Altitude
2		Leavitt 459	BRY-C	4597	Acarospora	acarospora	affsocialis	y	cagagttagg	USA	Utah		11-May			
3		Leavitt 679	BRY-C	6793	Acarospora	acarospora	affsocialis	y	cagagttagg	USA	Utah	Duchesne (	12-Aug			
4		Hollinger 1	BRY-C	jh16468	Acarospora	acarospora	affsocialis	y	cagagttagg	USA	NEVADA	Highland P	2-Jun-16	37.909° N	-114.5675°	2340 m
5		Leavitt 16-!	BRY-C	sl16590	Acarospora	acarospora	affsocialis	y	cagagttagg	USA	Wyoming	Flaming Gc	05 Sept, 2C	41.35153	-109.42	

Table 1: Header of Intermountain West Lichen DNA Reference Library

In addition to the categories contained within the OLICH DNA reference library had because we had some differing information for our collections. For example, I had to include a "Country" column because we had collections over a variety of countries (see Figure 5 below).

Assembling

Once the structure has been set for the DNA reference library, the next step was to assemble the library. DNA barcode sequences (ITS) have already been created by the BYU Lichen Lab. For each lichen family, there was a phylogenetic tree with an DNA ID along with an excel sheet of containing information regarding the DNA ID and sequence. For each family, I would find the matching DNA ID on the phylogenetic tree with the matching DNA ID on the excel sheet. I would assign the lichen a taxonomy based on the phylogenetic tree. Over 5,000 sequences were matched using this assembling method. Once all of the lichen families had been assembled in separate excel sheets, we combined them into our final and comprehensive DNA reference library.

Cleaning

After the lichens had been assembled, there was a need to clean up the data. There were some missing values in locality, date, and collector. NCBI or National Center for Biotechnology Information contains the BLAST algorithm that allows us to identify a published paper related to a sequence. To best try to fill these missing values, I would use the BLAST algorithm to blast the sequences in NCBI to find the related published paper. From that published paper, I would find the data table containing some of the missing information. I would fill in the missing values as best as I could, but some of the missing information could not be found. In addition to filling in missing values, I would delete duplicated sequences, move information to the correct column, and replace the Psora short ITS sequences with longer and more accurate sequences.

Summary Statistics

We needed up with three different datasets. We call our "working dataset contains the sequences collected by the BYU Lichen Herbarium. The "TLE" dataset contains the sequences collected T. L. Esslinger Herbarium. Lastly, we have the "genbank" dataset which contains the sequences collected by Genbank. To summarize the three datasets, I have created graphs for the collection density for both throughout the world and the states. It makes sense that the United States is the country with most collections within our DNA reference library, and that Utah is the state with the most collection sites. For our working data set, we have 1290 differing species for over 5,000 sequences. Figures 3-8 below show the states and world collection density map for each data set.

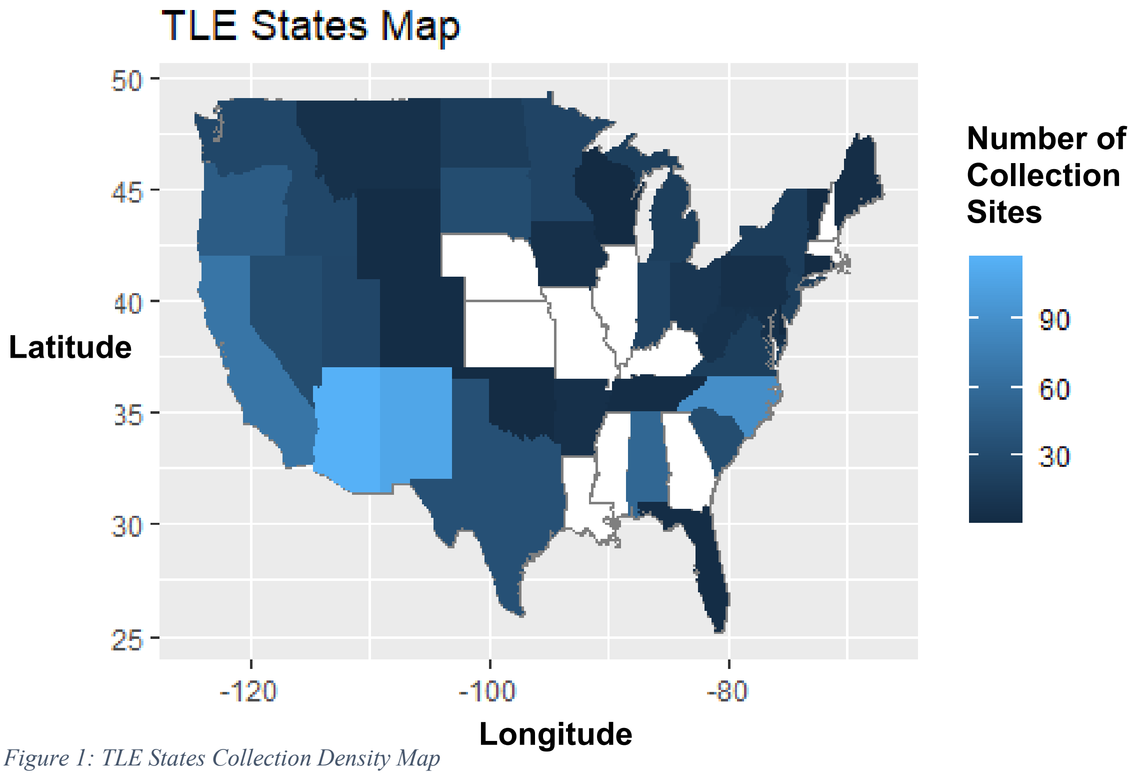


Figure 1: TLE States Collection Density Map

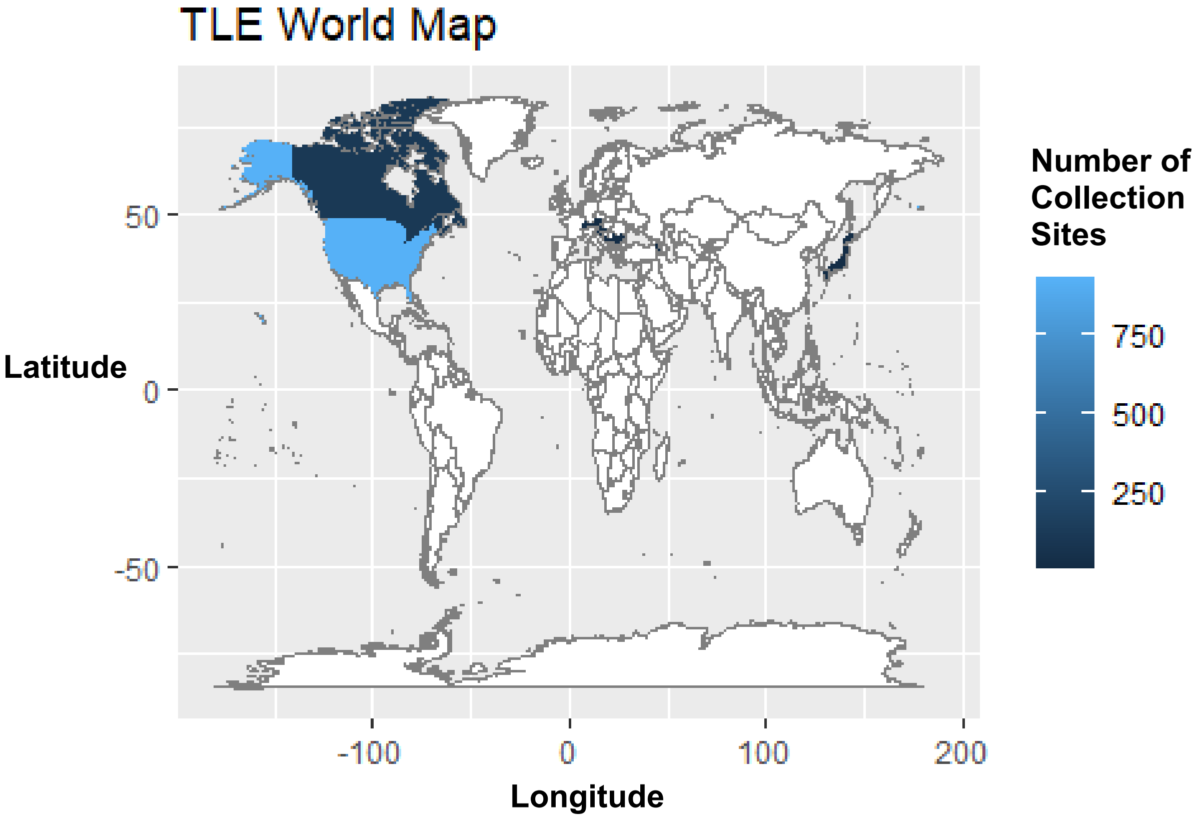


Figure 2: TLE World Collection Density Map

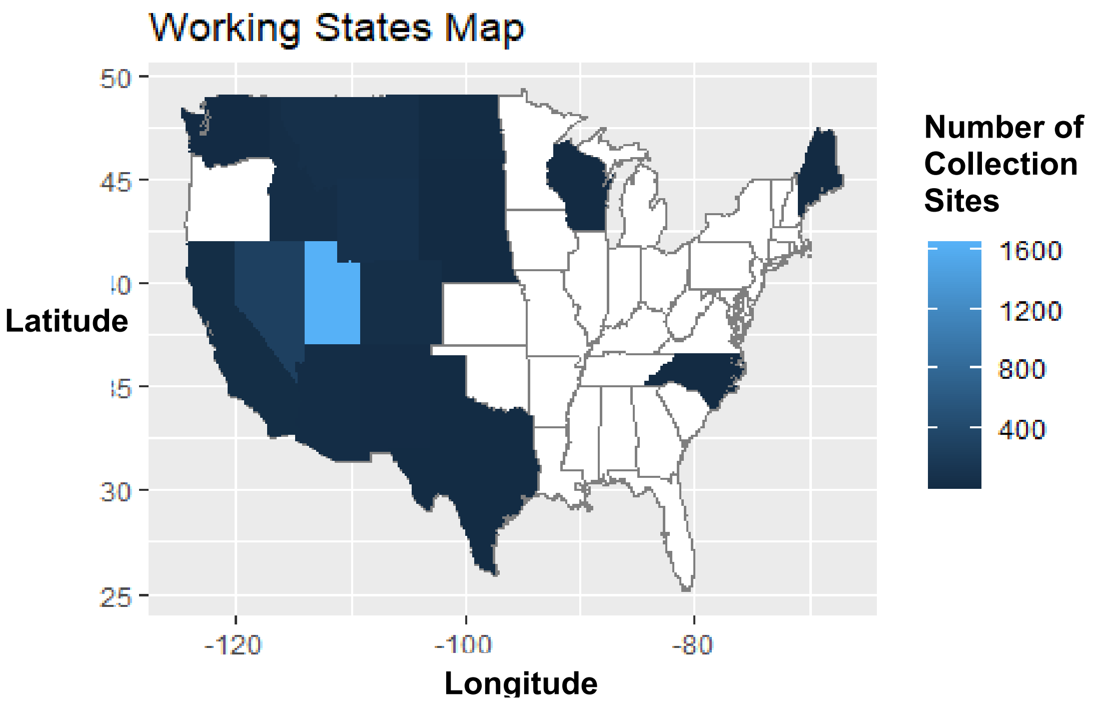


Figure 3: Working States Collection Density Map

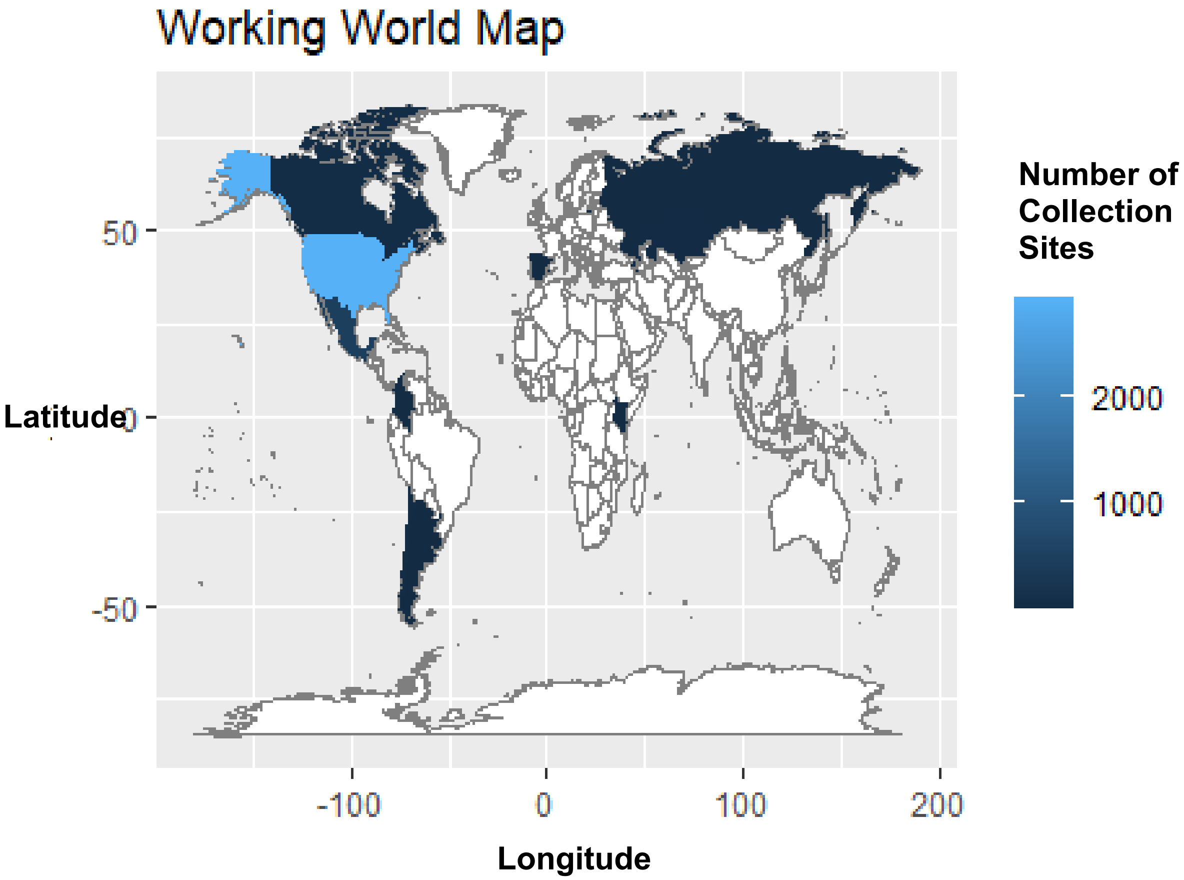


Figure 4: Working World Collection Density Map

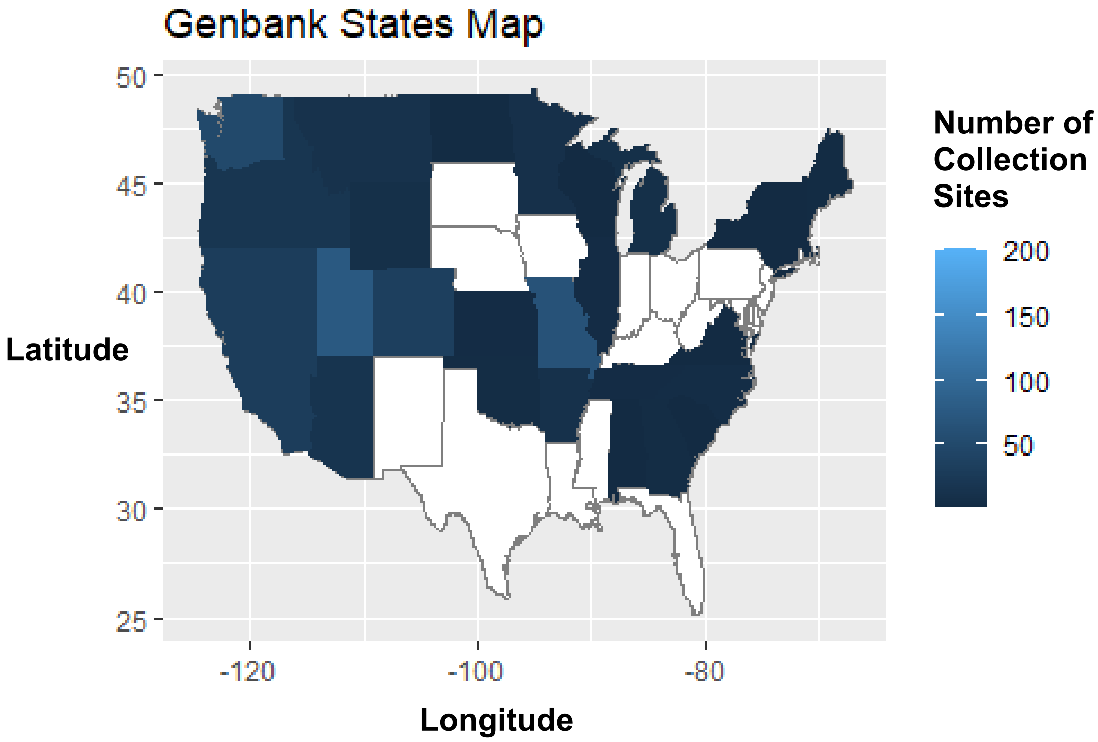


Figure 5: Genbank States Collection Density Map

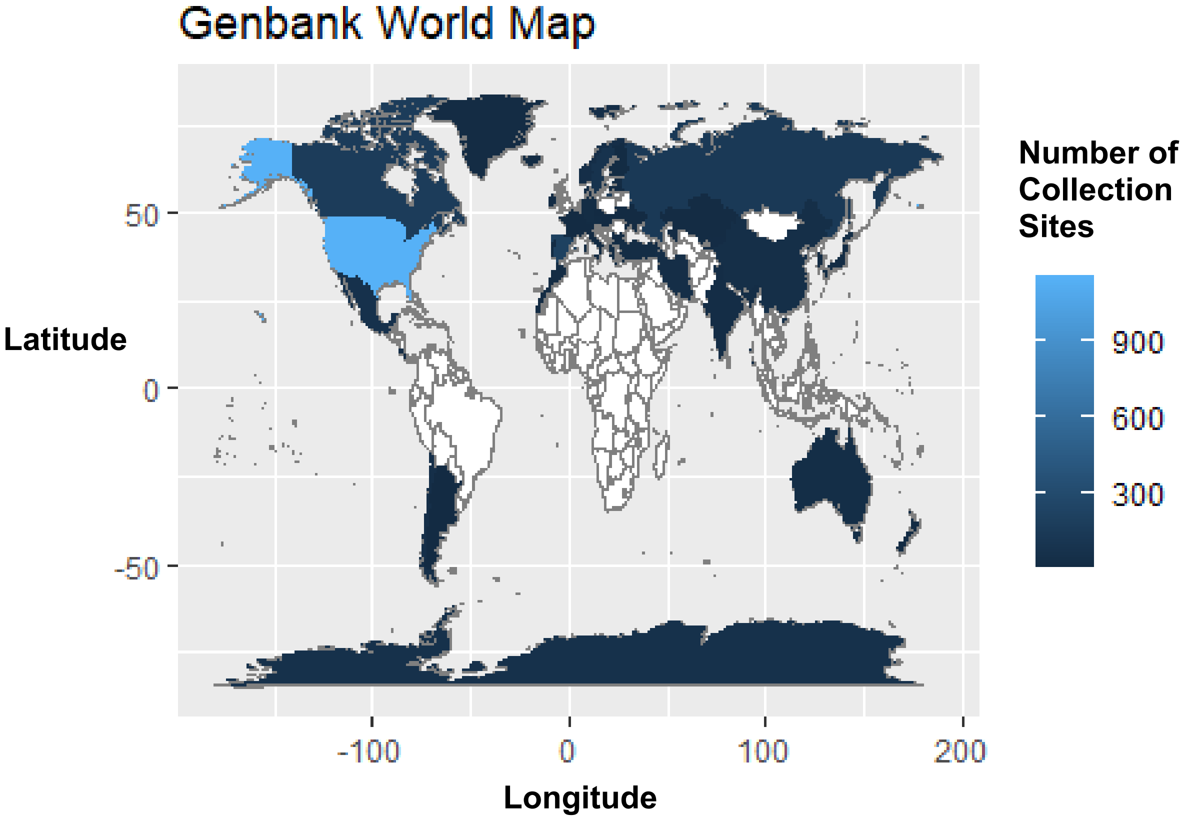
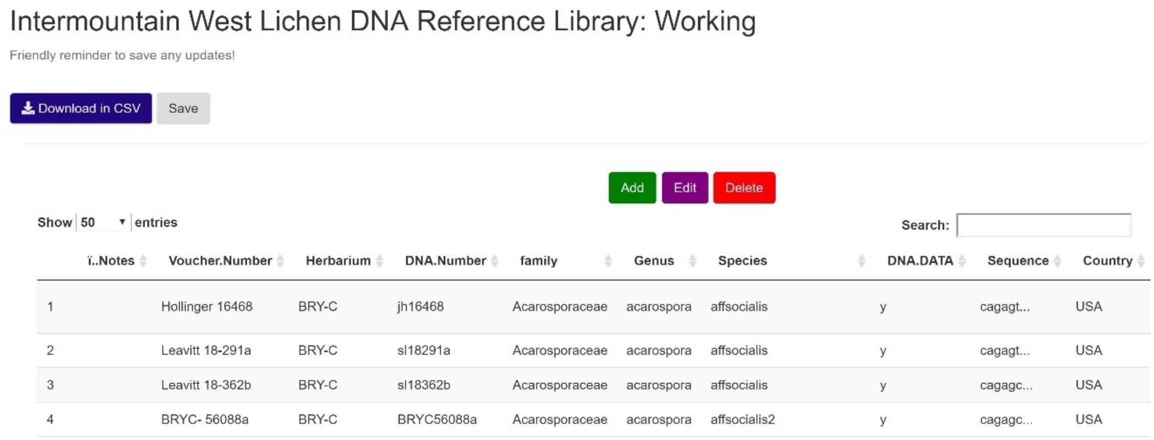


Figure 6: Genbank World Collection Density Map

R Shiny

The purpose of my shiny app is to allow the Intermountain West Lichen DNA Reference Library to be dynamic and to be continuously updated at one sole location. Before my app, the Lichen Lab had to send the DNA reference library through email and have a local copy that becomes quickly outdated. My app fixes this problem by having a reference library that is always up to date. As previously stated, I used the "shiny" and "DT" packages within R to create my user-friendly Intermountain West Lichen DNA Reference Library. The "shiny" package allows me to create an R Shiny app that can be hosted on shinyapps.io. The "DT" package allows me to edit JavaScript library "DataTables" but in R. The combination of these two packages allows me to create an editable dataset that can be hosted without running any code from a user. Within R shiny, there are three main components when coding an app. Their three components are executable script, an user interface script, and a server script. The executable script imports the UI and server scripts and summarily executes them. The user interface script contains all the code that produces what the user sees when he or she accesses the app. The code here is in charge of creating buttons, tabs, colors, number of entries, and anything that the user may interact with. The user interface code is also creating the structure of the app regarding the title, panels, and displays. Finally, the server script is the functionality behind all of the UI. If the user clicks the save button, the server reads that input and saves the new changes to the DNA reference library.

The functionality of my DNA reference library allows the user to continuously update the reference library by editing the cells, adding new rows, and deleting undesired rows. After revising the desired cells and saving it, the user can also download the reference library as a CSV (comma-separated values) file onto his or her local computer. A CSV file is a file that separates data by commas. This allows them to have a local copy which can be modified through other programs into a fasta file. A search bar allows the DNA reference library to be adaptable to the needs of the user. Additionally, the user can change the number of entries, and sort based on columns. The user can also easily switch access one of the three apps: working, TLE, and genbank in the dashboard of shinyapps.io. Each one contains different information for differing sequences and collections. My app allows the Intermountain West Lichen DNA Reference Library to be robust for the user while at the same time being user-friendly and dynamic. See figure 3 below to observe the UI of the Intermountain West Lichen DNA Reference Library App.



Intermountain West Lichen DNA Reference Library: Working

Friendly reminder to save any updates!

	i..Notes	Voucher.Number	Herbarium	DNA.Number	family	Genus	Species	DNA.DATA	Sequence	Country
1		Hollinger 16468	BRY-C	jh16468	Acarosporaceae	acarospora	affsocialis	y	cagagt...	USA
2		Leavitt 18-291a	BRY-C	sl18291a	Acarosporaceae	acarospora	affsocialis	y	cagagt...	USA
3		Leavitt 18-362b	BRY-C	sl18362b	Acarosporaceae	acarospora	affsocialis	y	cagagc...	USA
4		BRYC- 56088a	BRY-C	BRYC56088a	Acarosporaceae	acarospora	affsocialis2	y	cagagc...	USA

Figure 9: Intermountain West Lichen DNA Reference Library App

Shinyapps.io

After I got my app running, I needed a place to host it where a non-programmer could easily access it without having to run my code each time. I talked to the Statistics department and to the IT office at the LSB. They both said that shinyapps.io would be the best place to host my app for our scale. shinyapps.io is a cloud-based server produced by RStudio that is easy to use and secure. I have set up an account for Dr. Leavitt, so he has full access to the code and R shiny app. shinyapps.io allows the user to host R shiny app on RStudio's server that can be easily accessed by an URL. The URL privacy setting can be set to private, so only the owner can access the R shiny app. Since we do not want just anyone to access and change the DNA reference library, Dr. Leavitt has restricted the privacy, so he will be the only person who has access to the app.

Dr. Leavitt is on the free version of shinyapps.io, which allows him to host and use the app for 25 hours per month. The 25 hours per month fits the scale of this project, assuming only Dr. Leavitt is accessing the URL. However, there is always an option to upgrade the plan from the free version to a paid plan that allows for additional hours. Given the small scale of people accessing the app, shinyapps.io allows Dr. Leavitt ease of use and control over the app's privacy all the while being free.

GitHub

In addition to hosting the app on shinyapps.io, I have pushed all of my code to GitHub on a public repository. Nobody can make changes to the data and code found there; however, anyone can download the data and code and run them on their local machines. This is a backup accounting for the unlikely event that shinyapps.io crashes or runs into serious errors. If Dr. Leavitt needs future functions in my app, he will have full access to the data and code. He can assign another statistics or biology student who has knowledge of R and R shiny to easily implement new code to add the desired functionality to my app. For those who are not familiar with GitHub, I have written instructions on how to easily download the code into one's local computer. For those who are not familiar with running R scripts, I have written instructions on how to run my code along with downloading RStudio onto their computer.

Limitations

Even though I tried to account for all scenarios of bugs, server crashes, and the needs of the user, there are some limitations to my app. Some of the most obvious are regarding it being hosted by shinyapps.io. There is a monthly cap of 25 hours for hosting the app. Additionally, reliance on a company for continuing support and updates could be a potential limitation. The biggest limitation to this project is that its format is fixed once its finalized. Unless another person uses my code to make changes, the Intermountain West Lichen DNA Reference Library will be difficult to modify. Despite all of its limitations, this app and the DNA reference library will greatly reduce time spent organizing by Dr. Leavitt and the BYU Lichen Lab.

Future Projects

The lichens of the Intermountain West DNA reference library are sorted by a novel ranking system compatible with other fungal DNA reference library initiatives—the BOLD and UNITE databases. Data from our reference library will eventually be available in public databases (NCBI, BOLD, and UNITE), in addition to having a dynamically updated version available on the BYU webpage. This DNA reference library will facilitate more objective inventories and allow trained technicians to complete field surveys that were previously only accessible to professional specialists (Wright, St. Clair, & Leavitt, 2019).

The DNA reference library contains over 5,000 selected specimens from the BYU Bean Museum herbarium. There are tens of thousands of specimen samples in the herbarium that need to be sequenced. The DNA reference library will be continually updated as more specimens are sequenced and added to the library. Species delimitation will continue as the right taxonomy will eventually be identified, and duplicated species will be deleted. The Lichen Lab will also identify the remaining unidentified specimens and update the correct taxonomy for those. Future projects will utilize BLAST comparisons, sequence-based species delimitation analysis, and phylogenetic approaches to develop robust DNA barcodes for a truly integrative perspective into lichen biodiversity. The Intermountain West Lichen DNA Reference library will be continually updated and used by professors and students alike in near future projects.

Conclusion

In conclusion, the Intermountain West Lichen DNA Reference Library fulfills the need within the BYU Lichen Lab of having a dynamic DNA reference library all while joining the world-wide initiative of having accessible and public DNA barcode sequences. The dynamic functionality of the DNA reference library will allow the BYU Lichen Lab to continually update the library to further support projects in the near future. Researchers will be able to use the data to identify lichen species in the field without being an expert. Lichen diversity of western North America will now be represented, which will allow biologists and land managers to use DNA barcoding to identify previously unknown specimens. In addition, the DNA reference library will be used to move BYU's Lichen Air Quality Biomonitoring program forward and help ensure continued collaboration with the US Forest Service.

Personal Experience

Alongside the material results produced, I developed critical and analytical skills that will greatly enhance my present and future educational experience. The research project allowed me to acquire new bioinformatics and statistical skills while fine-tuning my programming and computational abilities. I had not been previously exposed to R shiny or using the package "DT". It was quite the uphill battle to learn and implement both at the same time. I did learn some great insights on how to create an app through R shiny and how hosting a server works. I also had the opportunity to earn first-hand knowledge of lichens, taxonomy, and phylogeny by working with Dr. Leavitt. These experiences and skill sets will be instrumental in meeting my educational and professional goals.

Works Cited

Learn Shiny. (n.d.). Retrieved from https://shiny.rstudio.com/tutorial/

Marthinsen, G. Rui, S, & Timdal, E. (2019). OLICH: A reference library of DNA barcodes for Nordic lichens. *Biodiversity Data Journal*, *7,* 1-146. doi:10.3897/BDJ.7.e3625

McLellan, Jiena. (n.d.). DT_editor_shiny_module. GitHub repository, https://github.com/jienagu/DT_editor_shiny_module

Wright, B., St. Clair, L. L. S., & Leavitt, S. D. (2019). Is targeted community DNA metabarcoding suitable for biodiversity inventories of lichen-forming fungi? *Ecological Indicators*, *98*, 812–820. doi:10.1016/j.ecolind.2018.11.061

RStudio. (n.d.) DT: An R interface to the DataTables library. GitHub repository, https://rstudio.github.io/DT/

Notes

The code to the app is listed below in Appendices 1-3. Each Appendix is a different R file with the name of the script and its code. Appendix 4 contains the license to use by GitHub user jienagu.

Appendix 1:
Executable Code

```
# Shiny executable
# UI
moduleUI <- function(id, temp_df, add_tag ){

 ns <- NS(id)
 AA=tagList()
 # return a list of tags
 for (i in 1:length(colnames(temp_df) ) )
 {
  if (is.numeric(temp_df[[i]]))
  {
   AA[[i]]<-numericInput(ns(paste0(colnames(temp_df)[i],"_add",add_tag) ),
paste0(colnames(temp_df)[i],":"),0)
  }
  # else if( is.Date(temp_df[[i]])){
  #   AA[[i]]<-
dateInput(paste0(colnames(temp_df)[i],"_add",add_tag),paste0(colnames(te
mp_df)[i],":"), Sys.Date())
  # }
  else{
   AA[[i]]<-textInput(ns(paste0(colnames(temp_df)[i],"_add",add_tag)),
paste0(colnames(temp_df)[i],":") )
  }
 }
 return(AA)
}

# Server part
module_server<- function(input, output, session,temp_df, add_tag) {


 dataframe=reactive({

  inputlist=list()
  for (j in 1:length(colnames(temp_df) ) ){
   inputlist[j]=input[[paste0(colnames(temp_df)[j],"_add",add_tag )]]
  }
  df_matrix=do.call(cbind.data.frame,inputlist)

  df_temp=data.frame(df_matrix)
  colnames(df_temp)=colnames(temp_df)
  df_temp
 })
```

```
  # Return the reactive that yields the data frame
  return(dataframe)
}
```

Appendix 2:
UI Code

```
# Read in libraries
library(shiny)
library(shinyjs)
library(shinysky)
library(DT)
library(data.table)
library(lubridate)
library(shinyalert)
useShinyalert()

# Define UI
shinyUI(fluidPage(

 # Application title
 titlePanel("Intermountain West Lichen DNA Reference Library: Working"),
 tags$head(tags$style(HTML('

              .modal-lg {
              width: 1200px;
              }
              '))),
 helpText("Friendly reminder to save any updates!"),
 br(),
 tags$head(tags$style(".butt{background-color:#230682;} .butt{color:
#e6ebef;}")),
 downloadButton("Trich_csv", "Download in CSV", class="butt"),
 useShinyalert(), # Set up shinyalert
 actionButton(inputId = "Updated_trich",label = "Save"),
 uiOutput("MainBody_trich")
))
```

Appendix 3:
Server Code

```
# Read in libraries
library(shiny)
library(shinyjs)
library(shinysky)
library(DT)
library(data.table)
library(lubridate)
library(shinyalert)

# Import functions, clear workspace
rm(list = ls())
useShinyalert()
source("shiny_module.R")

# R Shiny server
shinyServer(function(input, output, session){

  # Interactive dataset
  vals_trich<-reactiveValues()
  vals_trich$Data<-readRDS("working.rds")


  # MainBody_trich is the id of DT table
  output$MainBody_trich<-renderUI({
    fluidPage(
        hr(),
        column(6,offset = 6,
            HTML('<div class="btn-group" role="group" aria-label="Basic
example" style = "padding:10px">'),
            ### tags$head() This is to change the color of "Add a new row"
button
            tags$head(tags$style(".butt2{background-color:green;}
.butt2{color: #e6ebef;}")),
            div(style="display:inline-block;width:30%;text-align:
center;",actionButton(inputId = "Add_row_head",label = "Add",
class="butt2") ),
            tags$head(tags$style(".butt4{background-color:purple;}
.butt4{color: #e6ebef;}")),
            div(style="display:inline-block;width:30%;text-align:
center;",actionButton(inputId = "mod_row_head",label = "Edit",
class="butt4") ),
            tags$head(tags$style(".butt3{background-color:red;} .butt3{color:
#e6ebef;}")),
            div(style="display:inline-block;width:30%;text-align:
center;",actionButton(inputId = "Del_row_head",label = "Delete",
class="butt3") ),
            ### Optional: a html button
            # HTML('<input type="submit" name="Add_row_head"
value="Add">'),
            HTML('</div>') ),

       column(12,dataTableOutput("Main_table_trich")),
         tags$script("$(document).on('click', '#Main_table_trich button',
function () {
              Shiny.onInputChange('lastClickId',this.id);
              Shiny.onInputChange('lastClick', Math.random()) });")

    )
   })

  # Render DataTable part
  output$Main_table_trich<-renderDataTable({
   DT=vals_trich$Data
   datatable(DT, selection = "single",
          escape=F,
          options = list(
            pageLength = 50,
            lengthMenu = c(50, 100, 500, 1000),
            columnDefs = list(list(
             targets = c(9,12),
             render = JS(
               "function(data, type, row, meta) {",
               "return type === 'display' && data.length > 6 ?",
               "'<span title=\"' + data + '\">' + data.substr(0, 6) + '...</span>' :
data;",
               "}")
            ))))
   })


  observeEvent(input$Add_row_head, {
   # This is the pop up board for input a new row
   showModal(modalDialog(title = "Add a new row",
                  moduleUI("module_input_id",temp_df=vals_trich$Data,
add_tag=input$Add_row_head),
                  actionButton("go", "Add item"),
                  easyClose = TRUE, footer = NULL ))

  })
```

```
 # Add a new row to DT
 observeEvent(input$go, {
   datafile <- callModule(module_server, "module_input_id",
temp_df=vals_trich$Data, add_tag=input$Add_row_head )
   new_row=data.frame( datafile() )

   vals_trich$Data<-data.table(rbind(vals_trich$Data, new_row, fill=TRUE) )
   removeModal()
 })




 # Save to RDS file
 observeEvent(input$Updated_trich,{
   saveRDS(vals_trich$Data, "working.rds")
   shinyalert(title = "Saved!", type = "success")
 })

 # Delete selected rows
 # Warning messge for deleting
 observeEvent(input$Del_row_head,{
   showModal(
     if(length(input$Main_table_trich_rows_selected)>=1 ){
       modalDialog(
       title = "Warning",
     paste("Are you sure
delete",length(input$Main_table_trich_rows_selected),"rows?" ),
     footer = tagList(
       modalButton("Cancel"),
       actionButton("ok", "Yes")
     ), easyClose = TRUE)
     }else{
       modalDialog(
         title = "Warning",
       paste("Please select row(s) that you want to delect!" ),easyClose = TRUE
       )
     }

   )
 })

 # If user say OK, then delete the selected rows
 observeEvent(input$ok, {
    vals_trich$Data=vals_trich$Data[-input$Main_table_trich_rows_selected]
    removeModal()
 })

 # Edit button
 observeEvent(input$mod_row_head,{
  showModal(
   if(length(input$Main_table_trich_rows_selected)>=1 ){
    modalDialog(
     fluidPage(
      h3(strong("Modification"),align="center"),
      hr(),
      dataTableOutput('row_modif'),
      actionButton("save_changes","Save changes"),
      tags$script(HTML("$(document).on('click', '#save_changes', function
() {
              var list_value=[]
              for (i = 0; i < $( '.new_input' ).length; i++)
              {
              list_value.push($( '.new_input' )[i].value)
              }
              Shiny.onInputChange('newValue', list_value) });")) ), size="l" )
   }else{
    modalDialog(
     title = "Warning",
     paste("Please select the row that you want to edit!" ),easyClose = TRUE
    )
   }

  )
 })




 # Modify part
 output$row_modif<-renderDataTable({
  selected_row=input$Main_table_trich_rows_selected
  old_row=vals_trich$Data[selected_row]
  row_change=list()
  for (i in colnames(old_row))
  {
   if (is.numeric(vals_trich$Data[[i]]))
   {
    row_change[[i]]<-paste0('<input class="new_input" value=
','"',old_row[[i]],'"','  type="number" id=new_',i,' ><br>')
   }
   # else if( is.Date(vals_trich$Data[[i]])){
```

```
   #   row_change[[i]]<-paste0('<input class="new_input" value=
','"',old_row[[i]],'"',' type="date" id=new_  ',i,'  ><br>')
   # }
   else{
    row_change[[i]]<-paste0('<input class="new_input" value=
','"',old_row[[i]],'"',' type="textarea"  id=new_',i,'><br>')
   }
  }
  row_change=as.data.table(row_change)
  setnames(row_change,colnames(old_row))
  DT=row_change
  DT
  },escape=F,options=list(dom='t',ordering=F,scrollX =
TRUE),selection="none" )



 # Replace the modified row to existing row
 observeEvent(input$newValue,
       {
        newValue=lapply(input$newValue, function(col) {
         if (suppressWarnings(all(!is.na(as.numeric(as.character(col)))))) {
          as.numeric(as.character(col))
         } else {
          col
         }
        })
        DF=data.frame(lapply(newValue, function(x) t(data.frame(x))))
        colnames(DF)=colnames(vals_trich$Data)
        vals_trich$Data[input$Main_table_trich_rows_selected]<-DF
       }
 )
 # Download csv
 output$Trich_csv<- downloadHandler(
  filename = function() {
   paste("working_DNA_reference_library", Sys.Date(), ".csv", sep="")
  },
  content = function(file) {
   write.csv(data.frame(vals_trich$Data), file, row.names = F)
  }
 )

})
```

Appendix 4:
License to Use from Jiena's GitHub Repository

```
MIT
License

          Copyright (c) 2019 Jiena

          Permission is hereby granted, free of charge, to any person obtaining a
          copy
          of this software and associated documentation files (the "Software"), to
          deal
          in the Software without restriction, including without limitation the
          rights
          to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
          copies of the Software, and to permit persons to whom the Software is
          furnished to do so, subject to the following conditions:

          The above copyright notice and this permission notice shall be included in
          all
          copies or substantial portions of the Software.

          THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
          IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
          FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL
          THE
          AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
          LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING
          FROM,
          OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
          THE
          SOFTWARE.
```