Matching catalogued data

**Description:** This tutorial describes matching Institute of Fisheries Research lake observations and fish collection events with catalogued museum data from the Global Biodiversity Information Facility (GBIF).

# Part 1: Load libraries and data

This example uses lake summary data (summary\_card\_data.csv) that contains unique identifiers for a lake (new\_key), the lake name, county, and dates when the sampling was done. This information can be used to match lake data with specimen that were preserved from these lakes. This dataset also includes environmental variables for the lake, such as lake area and depth, temperature, and shoreline development.

#### load libraries ####   
library(dplyr) #library for data wrangling

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library(stringr) #library for extracting data   
library(tidyr)  
library(ggplot2) #library for graphs  
  
#### load summary card data ####   
summ\_dat<-read.csv("summary\_card\_data.csv")

As an example, we also downloaded bluegill specimen from GBIF, which includes all specimen of bluegill that were submitted to the University of Michigan Museum of Zoology database. We selected fields that we thought would help us to match including recorded by, dates, state, county, latitude/longitude, and locality, which includes the lake name. The preparations column contains the number of individuals in the collection, so we had to separate this number into its own column that we called “num\_individuals”. Then we filtered the museum records to select only observations in Michigan because this is where our data was collected.

#read in bluegill records in the UMMZ from GBIF and select only Michigan records recorded by IFR   
#preparations column contains # of individuals in a lot   
museum\_dat<-read.csv("blg\_UMMZ\_GBIF.csv") %>%   
 select(gbifID, identifier, basisOfRecord, occurrenceID, catalogNumber, preparations, fieldNumber,   
 eventDate, year, month, day, stateProvince, county, decimalLatitude, decimalLongitude, locality,   
 recordedBy ) %>%   
 mutate(num\_individuals = as.numeric(str\_extract(preparations, "(?<=EtOH - )\\d+")))%>% #extract number of individuals from the preparations column, where ?<= is a positive look-behind from the pattern "EtOH - " and \\d+ pulls out the digit  
 filter(stateProvince == "Michigan") %>%   
 mutate(year2 = str\_extract(fieldNumber, "[^-]+"),   
 year2 = as.numeric(str\_extract(year2, "[[:digit:]]+")) ) %>% #pull the year from the field number   
 mutate(year3=ifelse(year2 >= 19 & year2 <= 96, paste0("19",year2), NA)) %>% #add 19 infront of the year   
 mutate(year = as.integer(ifelse(is.na(year), year3, year))) %>% #if the year is missing then add the new year from fieldNumber  
 filter(year >= 1915 & year <= 1995) # filter to match the years for the summary cards

We get 485 observations from GBIF if we select only samples from Michigan between the year 1915-1995.

# Part 2: Match lake summary cards to museum records

In order to match these two datasets, we need to create a new column that includes only the lake name from the locality column and do some other standardizing. We make both the locality column and the county column uppercase to reduce differences in the way a lake name/county name is written as well as remove apostrophes. Then we extract the word before lake or pond into a new column, which is the lake name.

museum\_dat<-museum\_dat%>%   
mutate(locality = toupper(locality), #make locality uppercase so that names are standard   
 county = toupper(county), #make county uppercase so that names are standard   
 locality = gsub("'",'',locality), #strip apostrophes so that names are standard  
 locality = gsub(")" ,'',locality), #strip parenthesis so that names are standard  
 locality = gsub("L\\.",'LAKE',locality)) %>% #replace L. with Lake   
 mutate(lakename = str\_extract(locality, regex("\\b\\w+\\b\\s(?=LAKE)|\\b\\w+\\b\\s(?=POND)"))) %>% #extract the word right before lake or pond  
 mutate(lakename = trimws(lakename, which = c("both"))) %>% #remove white space around the lake name   
 mutate(lakename = ifelse(gbifID == 1889129462, "STCLAIR", lakename), #several that didn't pull the correct name   
 lakename = ifelse(gbifID == 1888996419, "FIVE", lakename),  
 lakename = ifelse(lakename == "BEESE", "BAWBEESE", lakename),  
 lakename = ifelse(lakename == "DEVIL", "DEVILS", lakename),  
 lakename = ifelse(lakename == "FIRST", "STHELENS", lakename),  
 lakename = ifelse(lakename == "INTERMEDIATE", "CENTRAL", lakename),  
 lakename = ifelse(gbifID == 1889017451, "CHAINOFS", lakename),  
 lakename = ifelse(gbifID ==1889003005, "DEVILSWASHBASIN", lakename),   
 lakename = ifelse(lakename == "PAW", "PAWPAW", lakename),   
 lakename = ifelse(lakename == "SELKIRK", "SELKIRKNORTH", lakename),   
 lakename = ifelse(gbifID ==1889079575, "FIRSTSISTER", lakename),  
 lakename = ifelse(gbifID ==1889007643, "THIRDSISTER", lakename),  
 lakename = ifelse(lakename == "STAR", "STARBIG", lakename),   
 lakename = ifelse(gbifID ==1888977161, "TWINLITTLE", lakename),   
 lakename = ifelse(gbifID ==1888977053, "TWINLITTLE", lakename)  
   
 )

We then match the museum and lake summary information by the county, lake name, and the year that the sample was taken. This ensures that the samples were taken from the correct lake in the same year. If the year is left blank in the GBIF data or cards or if the lake names don’t match up exactly then we wont get a match.

#### matching ####   
#\* match by lake name and county and year ####  
matches<-inner\_join(museum\_dat, summ\_dat, by=c("county", "lakename", "year"= "begin\_date\_year")) %>%   
 filter(subject\_id != 58643135) #remove duplicate card   
  
n\_distinct(matches$lakename, matches$county) # how many distinct lakes ?

## [1] 27

We get 60 museum records for 27 different lakes that match historical cards with data.

# Part 3: Summary of lakes and museum records

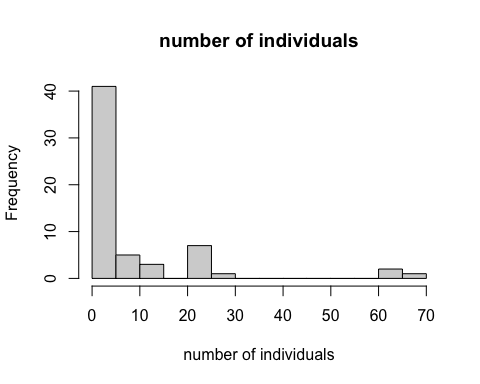
We can summarize the data and make some simple figures to investigate the information that we have for these lakes.

We can summarize the number of individuals from the museum records.

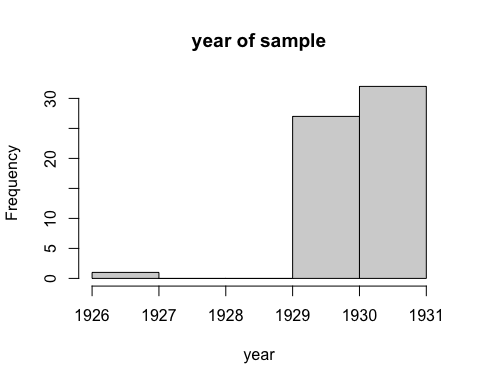
#summary  
matches %>%  
 summarise(  
 mean=mean(num\_individuals),  
 sd=sd(num\_individuals)  
 )

## mean sd  
## 1 8.666667 14.95606

#histogram   
hist(matches$num\_individuals, breaks = 10,   
 main = paste("number of individuals"),  
 xlab = 'number of individuals')



hist(matches$year,   
 main = paste("year of sample"),  
 xlab = 'year')

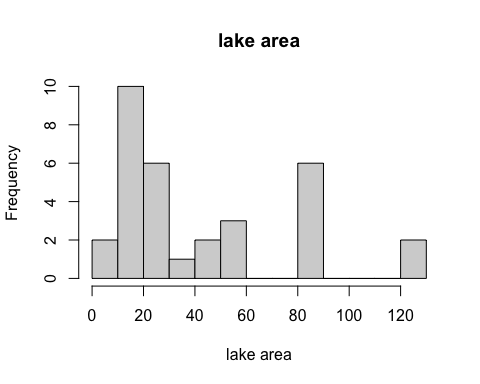


We can look at the lake attributes of the lakes that we have specimen for.

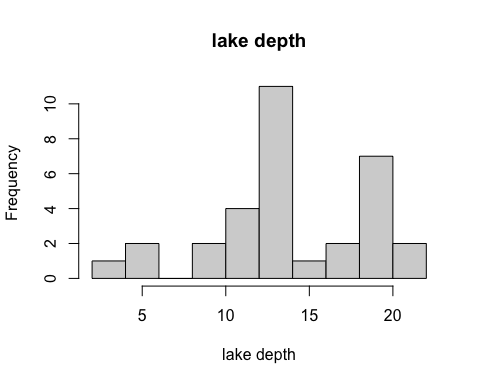
#summary table  
matches %>%  
 summarise(  
 area\_mean=mean(lake\_area\_ha, na.rm = TRUE),  
 area\_sd=sd(lake\_area\_ha, na.rm = TRUE),  
 depth\_mean=mean(max\_depth\_min\_m, na.rm = TRUE),  
 depth\_sd=sd(max\_depth\_min\_m, na.rm = TRUE),  
 temp\_mean=mean(temp\_surface\_min\_c, na.rm = TRUE),  
 temp\_sd=sd(temp\_surface\_min\_c, na.rm = TRUE),  
   
 )

## area\_mean area\_sd depth\_mean depth\_sd temp\_mean temp\_sd  
## 1 43.43875 35.33321 13.93437 4.861798 25.32988 1.261152

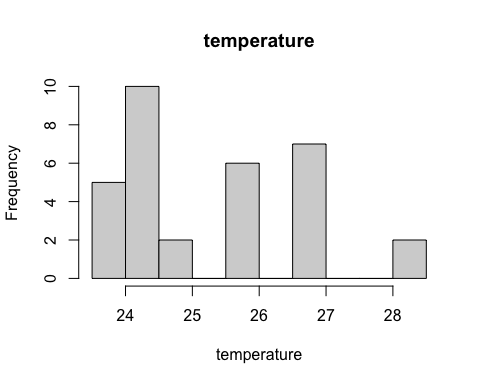
#histograms   
hist(matches$lake\_area\_ha, breaks = 10,   
 main = paste("lake area"),  
 xlab = 'lake area')



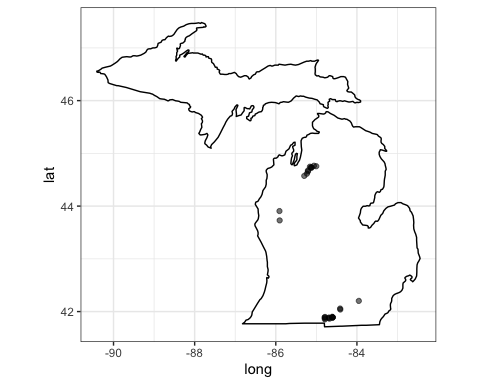
hist(matches$max\_depth\_min\_m, breaks = 10,   
 main = paste("lake depth"),  
 xlab = 'lake depth')



hist(matches$temp\_surface\_min\_c, breaks = 10,   
 main = paste("temperature"),  
 xlab = 'temperature')

 We can make maps of the lakes that matched

#select one point per lake   
lake\_matches<-matches%>%  
 distinct(lakename, county, .keep\_all = TRUE)  
#get the Michigan basemap  
MI\_basemap<-map\_data("state") %>%  
 subset(region %in% c("michigan")) # select michigan   
p<-ggplot(data = MI\_basemap) +   
 geom\_polygon(aes(x = long, y = lat, group = group), fill = "white", color = "black") + #this fill MI white and outline is black  
 coord\_fixed(1.3)   
  
# lake locations in MI  
p+ geom\_point(data=lake\_matches, aes(x = decimalLongitude, y = decimalLatitude, alpha = 0.1 ) ) +   
 theme\_bw() + # black and white theme for the background   
 theme(legend.position = "none") #remove legend



# we can color the points by an attribute, like the number of individuals in the museum collection   
p+ geom\_point(data=matches, aes(x = decimalLongitude, y = decimalLatitude, colour = c(num\_individuals))) +   
 labs(color="number individuals")+ #changes the labels on the legend  
 theme\_bw()

