Bluegill Growth Card Data Cleaning and Aggregation

**Description:** This tutorial describes the process of cleaning, aggregating, and standardizing output from Zooniverse transcribed data. The initial data extraction and reduction process from Zooniverse JSON data output was first run in Python using the panoptes-aggregation package, a package specifically made for Zooniverse projects, and is not included in this tutorial (Figure 1). The resulting csv files from this process come from two Zooniverse workflows 1) Basic tasks - where volunteers transcribe geographic location, dates of sampling, and collectors which are fields common to all data card types and 2) Growth tasks - where volunteers transcribe the number of fish, range and mean length of the fish, and fish age from growth cards. We combine several types of tasks from each of these workflows to produce a complete set of bluegill data from growth cards (Figure 1).

We first clean each of these tasks separately, then aggregate the datasets, and join them with the comments from the volunteer transcribers and the lake names that were matched to a local authority file (Figure 1). Finally, we standardize and quality check units.

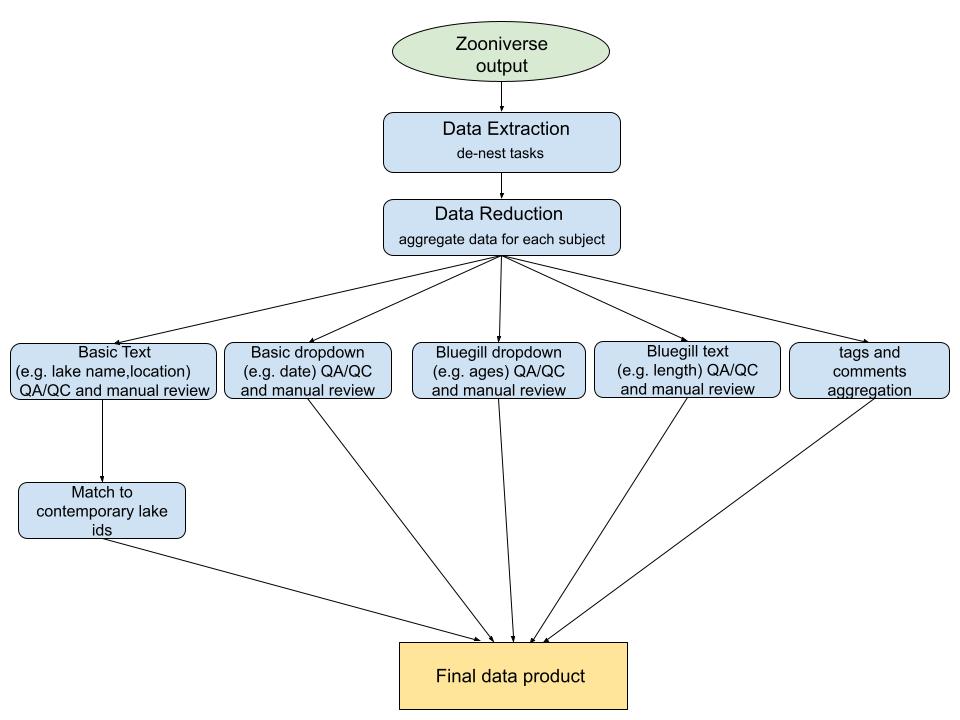
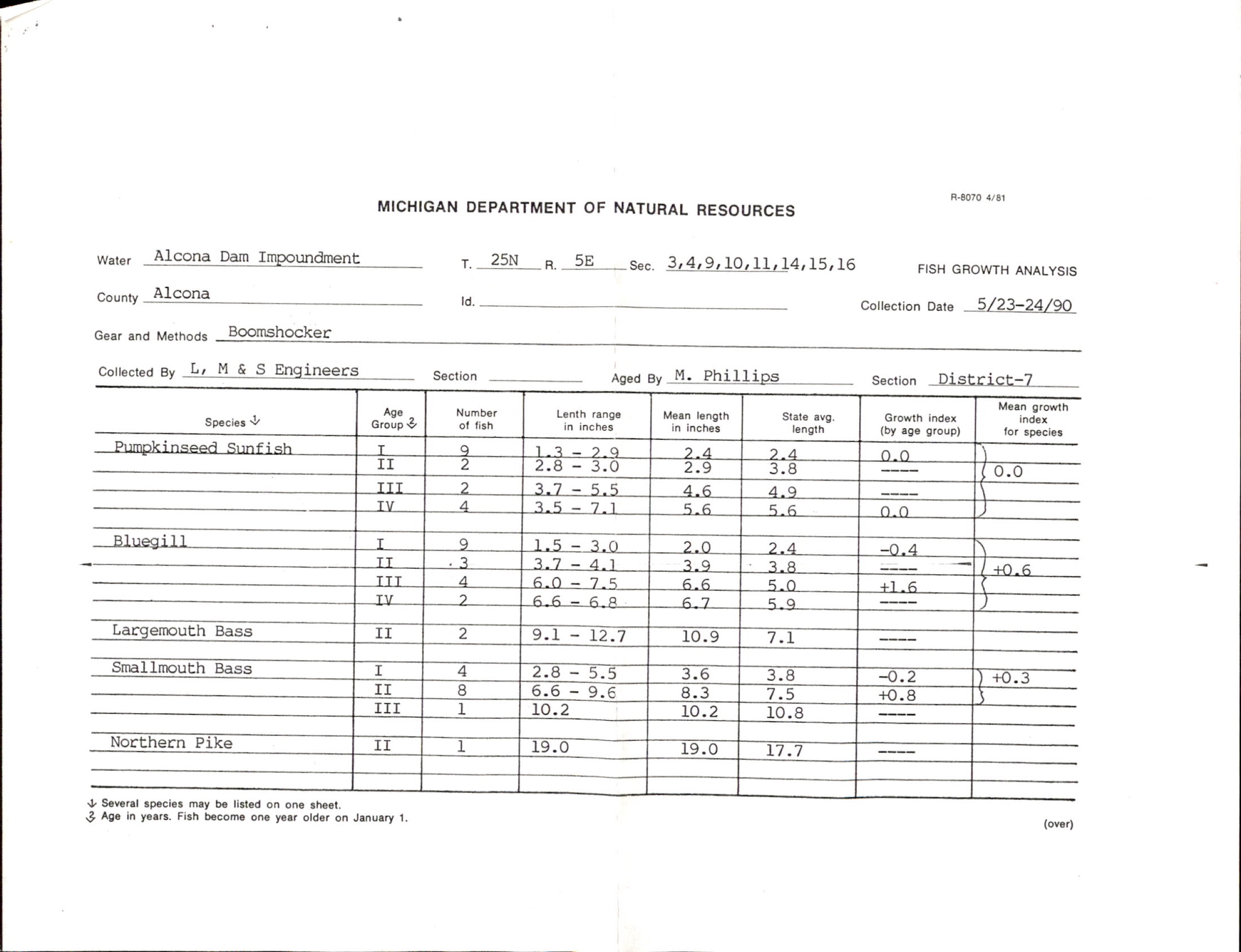


Figure 1: An overview of data curation steps for the Bluegill growth cards.

# Part 1: Load libraries and data

We have several datasets to load that will help us clean and match the output from Zooniverse. All data sets and code for this tutorial can be downloaded from github, <https://github.com/CHANGES-UM/data_curation_examples>.

1. “grow\_card\_lake\_match\_13Sep2021.csv” is a file that links historical lakes to the current “new\_key”, a lake identifier given to each lake by the Michigan Department of Natural Resources.
2. “basic\_GROW\_URLs.csv” is a file that has all of the links to the data card images. Take a look at an example image of a data card to become familiar with the original data format. 
3. “Basic\_task\_value.csv” and “GROW\_task\_values.csv” are files that have the task code from Zooniverse that can be linked to the actual task name
4. “Basic\_date\_values.csv” and “GROW\_age\_values” are files that have codes from Zooniverse drop down options that link to the actual date or the actual age value.

#### load libraries ####   
library(dplyr) # library for data munging   
library(tidyr) # library for data munging   
library(stringr) #library for regex functions   
library(rqdatatable) #join and replace NAs from one table with data from another   
  
#### load data ####   
#read in lake match data where lakes were matched to MDNR authority file   
lake\_match <-read.csv("blg\_grow\_data/lake\_match.csv", header=TRUE, na.strings = c("", "NA")) %>%  
 select(subject\_id, county, lakename, 'new.key') %>%  
 rename(new\_key = 'new.key') %>%  
 drop\_na('new\_key')  
  
#read in urls for card images   
urls<-read.csv("blg\_grow\_data/basic\_GROW\_URLs.csv", na.strings = c("", "NA")) %>%   
 distinct(subject\_id, .keep\_all = TRUE)   
  
#read in basic task names  
basic\_task\_values<-read.csv("blg\_grow\_data/basic\_task\_value.csv", header=TRUE) %>%  
 select(task, column\_name)  
  
#read in grow task names   
grow\_task\_values<-read.csv("blg\_grow\_data/GROW\_task\_values.csv", header=TRUE)  
  
#read in dropdown values   
basic\_date\_values<-read.csv("blg\_grow\_data/basic\_date\_values.csv", header=TRUE)  
  
grow\_age\_values<-read.csv("blg\_grow\_data/GROW\_age\_values.csv", header=TRUE) %>%   
 mutate(code= as.factor(code))

# Part 2: Clean basic text data

First, let’s look at how we clean basic text data, in this case sampling dates. On our data cards, begin and end dates were reported for sampling periods. We read in data for the sampling dates that were transcribed in Zooniverse. Each task was answered by 4 or more volunteers, so in order to separate the answers from each of the volunteers into separate columns, we remove punctuation, add an \_ in place of spaces, and then separate the answers and number of reviews. We also join to the tasks and date code tables so that we get the actual names of tasks and the dates.

#basic\_dropdown (dates) cleaning #   
  
basic\_dropdown<-read.csv("blg\_grow\_data/basic\_GROW\_dropdowns.csv", na.strings = c("", "NA")) %>%  
 select(subject\_id, task, 'data.value') %>%  
 drop\_na('data.value') %>%   
 rename(data\_value = 'data.value') %>%   
 mutate(sep = gsub("[[:punct:]]", "", data\_value), #remove punctuation   
 sep = gsub("\\s", "\_", sep), #replace space with \_  
 ) %>%   
 separate(col=sep, c("date\_hash", "review", "date\_hash2", "review2", "date\_hash3", "review3", "date\_hash4", "review4" ), sep = "\_") %>%  
 left\_join(basic\_task\_values) %>% #get actual column name   
 left\_join(basic\_date\_values, by=c('date\_hash' = 'code')) %>%   
 left\_join(basic\_date\_values, by=c('date\_hash2' = 'code')) %>%   
 left\_join(basic\_date\_values, by=c('date\_hash3' = 'code')) %>%   
 left\_join(basic\_date\_values, by=c('date\_hash4' = 'code')) %>%   
 rename(date1=month.x, date2=month.y, date3=month.x.x, date4=month.y.y) %>%  
 mutate(date1 = ifelse(is.na(date1), date\_hash, date1), #if date is NA then replace with the day/year, else keep month   
 date2 = ifelse(is.na(date2), date\_hash2, date2),  
 date3 = ifelse(is.na(date3), date\_hash3, date3),  
 date4 = ifelse(is.na(date4), date\_hash4, date4)  
 )

Next, we keep answers that have consensus (>50%) and those without consensus are reviewed manually (about 4% of observations). You can review the data that did not have consensus as the first “dates\_still\_na” output below and compare it with the corresponding image found at the URL in the url\_front column. This output was saved as a .csv and the research team corrected data in a spreadsheet. This .csv file of manually reviewed data by the research team is read in (and will override the original “dates\_still\_na” output) and joined with the consensus data. Because volunteers had the option to either put in the actual end date or choose “same as the begin date”, we replace the “same as” answers with the actual date. We also pivot the table to wide format so that each subject (card) is a row and each task becomes its own column.

#filter out observations where the review is split 50:50 or less  
#but if there is a split because choosing 'same as' then don't need to review these   
#observations for manual review (n=1027 of 24809)  
dates\_still\_na<-filter(basic\_dropdown, review < 3 & (!is.na(review2) | review2 < 3)) %>%  
 select(-c(task, data\_value, date\_hash, date\_hash2, date\_hash3, date\_hash4)) %>%  
 mutate(date= case\_when(   
 (date2 == 'Same as Begin Date (Month)' | date2 == 'Same as Begin Date (Day)' | date2 == 'Same as Begin Date (Year)') & is.na(date3) ~ date1, # if ~ then   
 (date1 == 'Same as Begin Date (Month)' | date1 == 'Same as Begin Date (Day)' | date1 == 'Same as Begin Date (Year)') & is.na(date3) ~ date2)  
 ) %>%   
 filter(is.na(date)) %>%   
 left\_join(urls )  
  
  
sameas\_dates<-filter(basic\_dropdown, review < 3 & (!is.na(review2) | review2 < 3)) %>%  
 select(-c(task, data\_value, date\_hash, date\_hash2, date\_hash3, date\_hash4)) %>%  
 mutate(date= case\_when(   
 (date2 == 'Same as Begin Date (Month)' | date2 == 'Same as Begin Date (Day)' | date2 == 'Same as Begin Date (Year)') & is.na(date3) ~ date1, # if ~ then   
 (date1 == 'Same as Begin Date (Month)' | date1 == 'Same as Begin Date (Day)' | date1 == 'Same as Begin Date (Year)') & is.na(date3) ~ date2)  
 ) %>%   
 filter(!is.na(date)) %>% pivot\_wider(  
 id\_cols = c(subject\_id),  
 names\_from = column\_name,   
 values\_from = c(date),   
 values\_fill = list(date = NA) )  
  
# keep reviews of 3 or greater as the answer   
dates\_good<-filter(basic\_dropdown, review >= 3 | is.na(review2) | review2 >= 3) %>%  
 mutate(date= case\_when(  
 is.na(review2) ~ date1, # if ~ then   
 review2 > review ~ date2,   
 review > review2 ~ date1   
 )) %>%  
 select(subject\_id, column\_name, date) %>%   
 pivot\_wider(id\_cols = c(subject\_id),  
 names\_from = column\_name,   
 values\_from = c(date),   
 values\_fill = list(date = NA) ) %>%  
 natural\_join(sameas\_dates, by = 'subject\_id', jointype = "FULL") %>%   
 mutate(end\_date\_day = ifelse(end\_date\_day =='Same as Begin Date (Day)', begin\_date\_day, end\_date\_day),  
 end\_date\_month =ifelse(end\_date\_month =='Same as Begin Date (Month)', begin\_date\_month, end\_date\_month),  
 end\_date\_year = ifelse(end\_date\_year =='Same as Begin Date (Year)', begin\_date\_year, end\_date\_year)   
 )  
  
#merging back in manually reviewed data   
dates\_still\_na<-read.csv("blg\_grow\_data/dates\_manual.csv", na.strings = c("", "NA")) %>%  
 select(subject\_id, column\_name, date) %>%   
 pivot\_wider(id\_cols = c(subject\_id),  
 names\_from = column\_name,   
 values\_from = c(date),   
 values\_fill = list(date = NA) )  
  
dates\_clean\_plus\_manual<-natural\_join(dates\_good, dates\_still\_na, by = 'subject\_id', jointype = "FULL")

Finally, a quality check is done to see if begin and end years are the same. If not, these are also reviewed manually by someone from the research team. The manually reviewed data by the research team is read in and joined with the consensus data.

#QA/QC:check begin and end years that don't match n=351 of 4682 - review manually   
dates\_clean\_plus\_manual<-dates\_clean\_plus\_manual %>%   
 mutate(review = ifelse(begin\_date\_year != end\_date\_year, "TRUE", "FALSE")  
 )  
  
grow\_dates\_review<-read.csv("blg\_grow\_data/year\_manual.csv")%>%   
 select(-c("review"))  
  
dates\_final<-filter(dates\_clean\_plus\_manual, review == 'FALSE' | is.na(review)) %>%   
 left\_join(urls) %>%   
 drop\_na('URL\_front') %>%   
 select(-c("review")) %>%  
 mutate (comment = NA)%>%  
 rbind(grow\_dates\_review) %>%   
 rename(url\_front= URL\_front, url\_back= URL\_back)

# Part 3: Clean fish lengths text data

Data for the the number of fish, fish length ranges, and fish length means are read in and joined to the task values. Data is then cleaned by removing text and units.

#### blgrow\_text (length info) cleaning ####  
blgrow\_text<-read.csv("blg\_grow\_data/bluegill\_texts.csv") %>%  
 select(subject\_id, task, 'data.aligned\_text', 'data.number\_views', 'data.consensus\_score','data.consensus\_text') %>%  
 drop\_na('data.number\_views') %>%   
 left\_join(grow\_task\_values) %>%   
 mutate(data = str\_remove(data.consensus\_text, "//.\*"), #remove string after //   
 data = ifelse(data =='3 six year old samples', 0, data),   
 data = gsub("mm" , '', data), #remove units  
 data = gsub("cm" , '', data),   
 data = gsub("in" , '', data)  
 ) %>% #remove one weird answer '3 six year old samples' should say 'no six year old samples'   
 naniar::replace\_with\_na(replace = list(data = c('-', '--', "---", '...', '. . .', "-----", "———"))) %>% #function that replaces all of a value with NA   
 drop\_na('data') %>%  
 separate(data, c("data", "trash"), sep = " ") %>% #gets rid of extra stuff  
 select(-c(trash))

Answers that have consensus (>50%) are kept and those without consensus were reviewed manually and are then joined back with the consensus data similarly to the basic data.

#1230   
## what was recorded by only 1 person  
blg\_text\_bad1<- filter(blgrow\_text, data.number\_views == 1)   
# what was viewed by >= 2 people but half did not agree   
blg\_text\_bad2<- filter(blgrow\_text, (data.number\_views >2 & data.consensus\_score < 2)| (data.number\_views ==2 & data.consensus\_score < 2) )   
  
#good data where consensus is >50%   
blg\_text\_good<-filter(blgrow\_text, (data.number\_views >2 & data.consensus\_score > 2)| (data.number\_views ==2 & data.consensus\_score >= 2) ) %>%  
select(subject\_id, column\_name, data)  
  
#merging back in manually reviewed data   
manual\_length1<-read.csv("blg\_grow\_data/manual\_text1.csv", na.strings = c("", "NA")) %>%  
 select(subject\_id, column\_name, data) %>%  
 mutate(column\_name = gsub(" row", "\_row", column\_name))  
manual\_length2<-read.csv("blg\_grow\_data/manual\_text2.csv", na.strings = c("", "NA")) %>%  
 select(subject\_id, column\_name, data)%>%  
 mutate(column\_name = gsub(" row", "\_row", column\_name))  
  
lengths\_clean<-gtools::smartbind(blg\_text\_good, manual\_length1, manual\_length2) %>%  
 pivot\_wider(   
 id\_cols = c(subject\_id),  
 names\_from = column\_name,   
 values\_from = c(data),   
 values\_fill = list(data = NA) )

# Part 4: Clean fish age dropdown tasks

The Zooniverse data for fish ages from a dropdown task is read in and cleaned similarly to the date dropdown task. Dropdown tasks are advantageous in that they give volunteers a fixed vocabulary and are therefore less prone to typos or inconsistent formatting. Punctuation is removed, the volunteers’ answers are moved into separate columns, the hash values are replaced with reported ages, and then joined with the task values.

#read in file   
blgrow\_dropdown<-read.csv("blg\_grow\_data/bluegill\_dropdowns.csv", na.strings = c("", "NA")) %>%  
 select(subject\_id, task, 'data.value') %>%  
 drop\_na('data.value') %>%   
 rename(data\_value = 'data.value') %>%   
 mutate(sep = gsub("[[:punct:]]", "", data\_value),   
 sep = gsub("\\s", "\_", sep)  
 ) %>%   
 separate(col=sep, c("age\_hash", "review", "age\_hash2", "review2", "age\_hash3", "review3", "age\_hash4", "review4" ), sep = "\_") %>%   
 mutate(age\_hash= as.factor(age\_hash),  
 age\_hash2=as.factor(age\_hash2),  
 age\_hash3=as.factor(age\_hash3),  
 age\_hash4=as.factor(age\_hash4) ) %>%   
 left\_join(grow\_task\_values) %>%   
 left\_join(grow\_age\_values, by=c('age\_hash' = 'code')) %>%   
 left\_join(grow\_age\_values, by=c('age\_hash2' = 'code')) %>%   
 left\_join(grow\_age\_values, by=c('age\_hash3' = 'code')) %>%   
 left\_join(grow\_age\_values, by=c('age\_hash4' = 'code')) %>%   
 rename(age1=age\_group.x, age2=age\_group.y, age3=age\_group.x.x, age4=age\_group.y.y) %>%   
 mutate(age1 = ifelse(age\_hash =='9f71969064233', 5, age1),   
 age2 = ifelse(age\_hash =='9f71969064233', 5, age2),  
 age3 = ifelse(age\_hash =='9f71969064233', 5, age3)) #for some reason this hash code did not match

Answers that have consensus (>50%) are kept and those without consensus are reviewed manually and then joined back with the consensus data.

# keep >= 3s OR any that were 2 but don't have disagreements (NAs)   
age\_good<-filter(blgrow\_dropdown, review >= 3 | review2 >= 3 | (review == 2 & is.na(review2)) ) %>%  
 mutate(age= case\_when(  
 is.na(review2) ~ age1, # if ~ then (total agreement)  
 review2 > review ~ age2, # if ~ then (majority selected)  
 review > review2 ~ age1 # if ~ then (majority selected)  
 )) %>%  
 select(subject\_id, column\_name, age)  
  
#filter out the ones for manual review (n=429 of 12201)  
age\_manual<-filter(blgrow\_dropdown, review < 3 & review2 < 3 | (review == 1 & is.na(review2))) %>%  
 select(-c(task, data\_value, age\_hash, age\_hash2, age\_hash3, age\_hash4))  
  
age\_manual<-read.csv( "blg\_grow\_data/ages\_manual.csv") %>% #after manual review   
 select(subject\_id, column\_name, age)%>%  
 mutate(column\_name = gsub(" row", "\_row", column\_name))  
  
#join manually reviewed and good datasets   
age\_data<-rbind(age\_good, age\_manual)%>%  
 pivot\_wider(id\_cols = c(subject\_id),  
 names\_from = column\_name,   
 values\_from = c(age),   
 values\_fill = list(age = NA) )

# Part 5: Joining age and length data

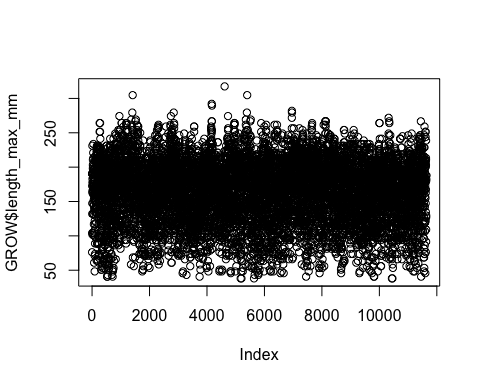
The age data are joined with the length data then we pivot the table so that we have a long table with a column for age group, fish count, length range, and length mean. We rename the columns and then remove rows that do not have data. We then split out the length range into min and max values. We also checked instances where the age group was listed as NA by looking at the original card to determine if this data was actually missing or were errors, these rows were removed due to errors in transcription.

#join age data: dropdown ages and length text ####  
all\_grow<-left\_join(age\_data, lengths\_clean) %>%   
 pivot\_longer(-subject\_id,  
 names\_to =c("row", ".value"),  
 names\_sep ="\_row") %>%   
 rename(age\_group = " age group ", fish\_count = " Number of Fish ", length\_range = " Length Range ", length\_mean = " Mean Length ") %>%   
 filter(if\_any(c("age\_group", "fish\_count", "length\_range", "length\_mean"), complete.cases)) %>%  
 separate(length\_range, c("length\_min", "length\_max"), sep = "-") %>%   
mutate(fish\_count = as.numeric(as.character(fish\_count)),   
 length\_min = as.numeric(as.character(length\_min)),   
 length\_max = as.numeric(as.character(length\_max)),  
 length\_mean = as.numeric(as.character(length\_mean))  
 )%>%   
 select(-c(row)) %>%   
 drop\_na('age\_group') #drop rows where age group is NA

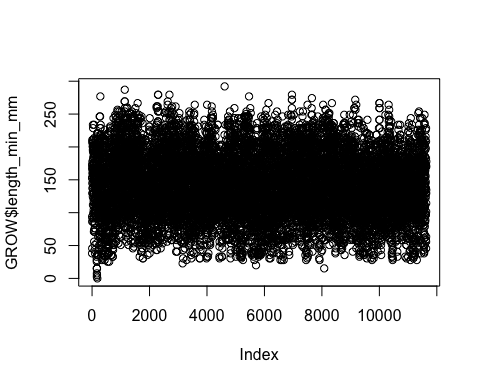
# Part 6: Combine all data and perform quality control

The final steps are combining all data and standardizing units. We read in and aggregate comments from volunteer transcribers. Volunteers could leave comments for individual images that they were classifying. These comments may mention units, interesting discoveries, or anomalies on the cards. These comments were helpful in discovering units and may include other information that researchers find useful when using the data. Original data were mostly recorded in inches, however, some cards specify when they were instead recorded in centimeters or millimeters, so we made a file that has these cases. We join the grow card information with the basic dates, the lake matches, the units, and comments.  
We standardize all units by converting all lengths to millimeters. A quality check is done to plot these data to look for outliers. A quality check is also done to ensure where the fish count is 0 there are no length data.

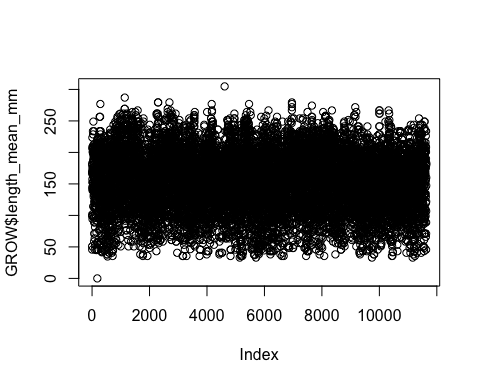
#### clean up comments and join all data ####   
#read in comments   
blgrow\_comments<-read.csv("blg\_grow\_data/blg\_comments.csv") %>%  
 select(subject\_id, "X\_...comment\_body") %>%  
 rename(comments = "X\_...comment\_body")  
  
comments <- aggregate(blgrow\_comments[2], blgrow\_comments[-2],   
 FUN = function(X) paste(unique(X), collapse=", "))  
  
grow\_units<-read.csv("blg\_grow\_data/grow\_card\_units.csv")   
  
GROW<-left\_join(all\_grow, dates\_final) %>%  
 left\_join(lake\_match) %>%  
 left\_join(grow\_units) %>% #units from manual review and tags  
 left\_join(comments) %>%  
 drop\_na('url\_front') %>% #drop cards without a url, these were removed from workflows   
 mutate(units= case\_when(  
 unit == "centimeters" ~ "centimeters",  
 unit == "millimeters" ~ "millimeters",  
 TRUE ~ 'inches' )) %>%  
 mutate(length\_min\_mm = case\_when( #convert everything to millimeters   
 units == 'inches' ~ length\_min\*25.4,   
 units == 'centimeters' ~ length\_min\*10,   
 TRUE ~ length\_min # else keep as mm   
 )) %>%  
 mutate(length\_max\_mm = case\_when(  
 units == 'inches' ~ length\_max\*25.4,   
 units == 'centimeters' ~ length\_max\*10,   
 TRUE ~ length\_max # else keep as mm   
 )) %>%  
 mutate(length\_mean\_mm = case\_when(  
 units == 'inches' ~ length\_mean\*25.4,   
 units == 'centimeters' ~ length\_mean\*10,   
 TRUE ~ length\_mean # else keep as mm   
 )) %>%  
 select(-c(length\_min, length\_max, length\_mean, units, unit)) %>%   
 mutate(length\_mean\_mm = ifelse(is.na(length\_mean\_mm) & fish\_count == 1, length\_min\_mm, length\_mean\_mm)) %>%   
 mutate(length\_min\_mm = ifelse(subject\_id =='58642811' & age\_group == 7, 180.34, length\_min\_mm) ) #one outlier that looks like a typo based on max and mean values   
  
#QAQC check for outliers   
plot(GROW$length\_max\_mm)



plot(GROW$length\_min\_mm)



plot(GROW$length\_mean\_mm)



#add some comments for cards with multiple bluegill records for different gear types   
GROW<-GROW %>%   
 mutate(comments=ifelse(subject\_id == "58642823", 'two gear types', comments),   
 comments=ifelse(subject\_id == "58642824", 'two gear types', comments)  
 )  
  
#QAQC if the fish count is 0 and not NA then there should not be length values, these were checked   
GROW<-GROW%>%   
 mutate(length\_min\_mm = ifelse(fish\_count == 0 & !is.na(fish\_count), NA, length\_min\_mm),  
 length\_max\_mm = ifelse(fish\_count == 0 & !is.na(fish\_count), NA,length\_max\_mm),  
 length\_mean\_mm = ifelse(fish\_count == 0 & !is.na(fish\_count), NA, length\_mean\_mm)  
) %>%   
 mutate(comment = ifelse(is.na(comment), comments, comment)  
 ) %>%   
 select(-c(comments))  
  
#reorder columns so that identifiers are at the beginning, followed by fish info, dates, and urls  
GROW <- GROW[, c(1, 15,14,13, 2, 3, 16,17,18, 4,5,6, 7, 8,9,12, 10,11)]