03_Class_Activity

Bill Perry

In class activity 4:

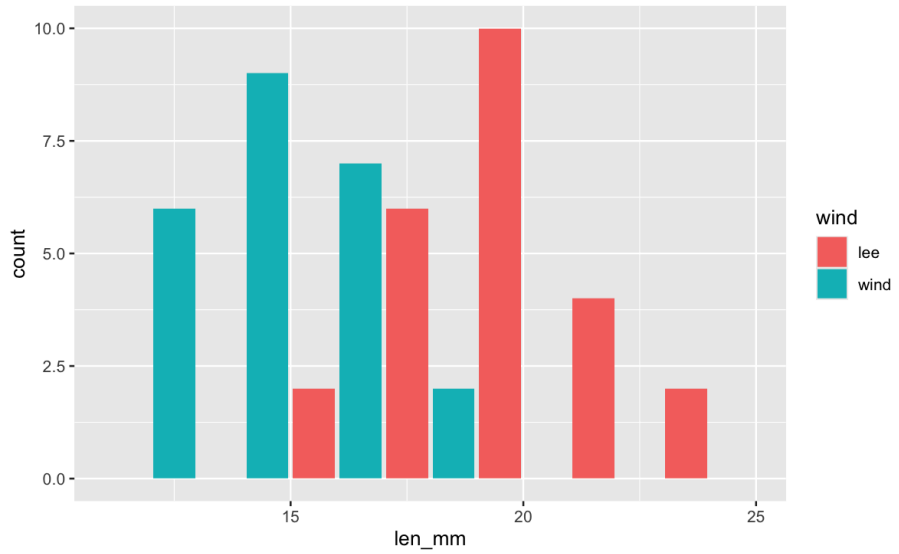
What did we do last time in activity 3?

- Setting up a project and variable names and code names
- How to use the pipe command %>%
- How to create descriptive statistics of a sample

```
p_df %>%
  summarize(
    mean_length = mean(length_mm, na.rm = TRUE),
    sd_length = sd(length_mm, na.rm = TRUE),
    n_length = sum(!is.na(length_mm)))
```

- More graphs...

```
ggplot(data = p_df, aes(x=length_mm, fill = wind)) +
  geom_histogram( binwidth = 2,
# sets the width in units of the bins - try different nubmers
   position = position_dodge2(width = 0.5))
```



- What questions do you have and what is unclear - what did not work so far when you started the homework?

Introduction

In this active learning module, we'll explore real data from fish populations in Alaska. We'll focus on understanding:

- How to create and interpret frequency distributions
- How sample size affects our view of a population
- How distributions differ among lakes

We'll use the `tidyverse` package for data manipulation and visualization.

Setup

First, let's load the packages we need and the dataset:

```
# # Install the patchwork package if needed

# install.packages("patchwork")
library(patchwork)
library(skimr)
library(tidyverse)


# Read in the data file
s_df <- read_csv("data/sculpin.csv")

# Look at the first few rows
head(s_df)
```

```
# A tibble: 6 × 5
   site lake  species       length_mm mass_g
  <dbl> <chr> <chr>             <dbl>  <dbl>
1   146 E 01  slimy sculpin        53   1.25
2   146 E 01  slimy sculpin        61   1.9
3   146 E 01  slimy sculpin        53   1.75
4   146 E 01  slimy sculpin        77   4.25
5   146 E 01  slimy sculpin        45   0.9
6   146 E 01  slimy sculpin        48   0.9
```

Basic Data Summary

Let's first check what lakes are in our dataframe:

```
# Get a list of unique lakes
unique(s_df$lake)
```

```
[1] "E 01"   "E 05"   "NE 12"  "NE 14"  "S 06"   "S 07"   "Toolik"
```

How many fish do we have from each lake?

```
# Count observations by lake
s_df %>%
  group_by(lake) %>%
  summarize(sculpin_n = n())
```

```
# A tibble: 7 × 2
  lake   sculpin_n
  <chr>      <int>
1 E 01         268
2 E 05          75
3 NE 12        180
4 NE 14         37
5 S 06         132
```

```
6 S 07          73
7 Toolik       287
```

```
# Count observations by lake
s_df %>%
  group_by(lake) %>%
  summarize(sculpin_n = sum(!is.na(length_mm)))
```

```
# A tibble: 7 × 2
  lake   sculpin_n
  <chr>      <int>
1 E 01          79
2 E 05          14
3 NE 12        180
4 NE 14         37
5 S 06         132
6 S 07          73
7 Toolik       208
```

Part 1: Creating Frequency Distributions

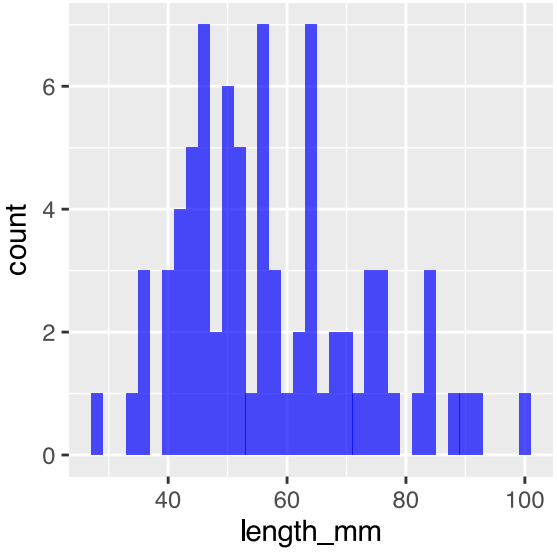
Basic Histograms

A histogram shows how many observations fall into certain ranges (or "bins").

Let's create a simple histogram of fish lengths from Lake E 01 :

```
# Filter for Toolik Lake and create a histogram
s_df %>%
  filter(lake == "E 01") %>%
  ggplot(aes(x = length_mm)) +
  geom_histogram(binwidth = 2, fill = "blue", alpha = 0.7)
```

```
Warning: Removed 189 rows containing non-finite outside the scale range
(`stat_bin()`).
```



Activity 1

Try changing the `binwidth` parameter to 5 and then to 1. How does the appearance of the histogram change?

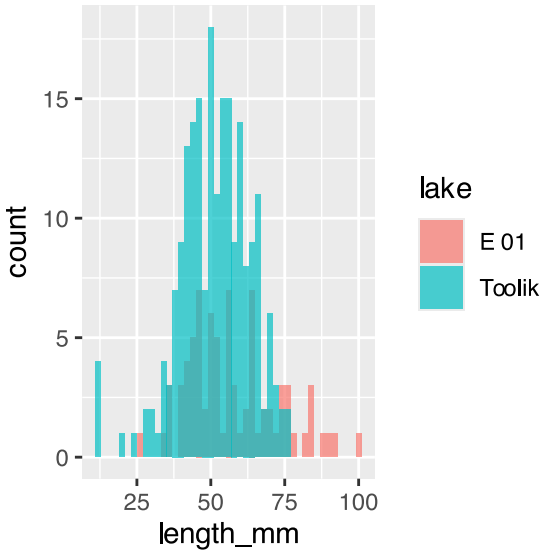
```
# Try it here
```

Comparing Lakes

Now let's compare two lakes

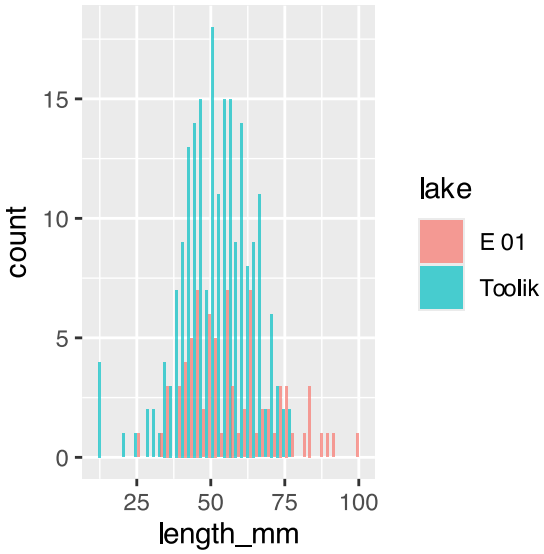
```
# Compare histograms from Toolik and E 01 lakes
s_df %>%
  filter(lake %in% c("Toolik", "E 01")) %>%
  ggplot(aes(x = length_mm, fill = lake)) +
  geom_histogram(binwidth = 2, alpha = 0.7,
                 position = "identity")
```

```
Warning: Removed 268 rows containing non-finite outside the scale range
(`stat_bin()`).
```




```
# Compare histograms from Toolik and E 01 lakes
s_df %>%
  filter(lake %in% c("Toolik", "E 01")) %>%
  ggplot(aes(x = length_mm, fill = lake)) +
  geom_histogram(binwidth = 2, alpha = 0.7,
                 position = position_dodge2(width=1))
```

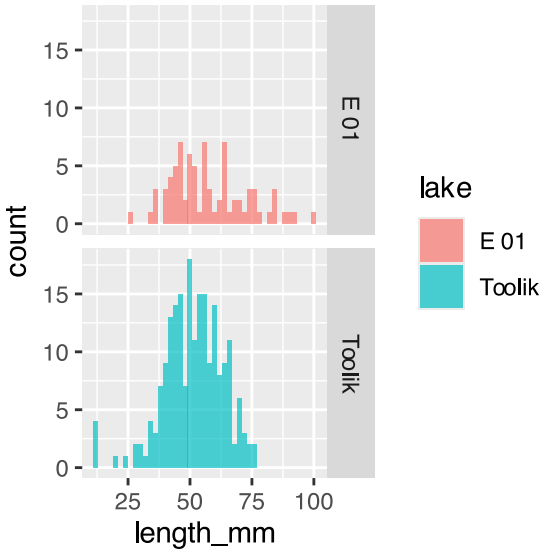
```
Warning: Removed 268 rows containing non-finite outside the scale range
(`stat_bin()`).
```



Now let's compare two lakes side by side:

```
# Compare histograms from Toolik and E 01 lakes
s_df %>%
  filter(lake %in% c("Toolik", "E 01")) %>%
  ggplot(aes(x = length_mm, fill = lake)) +
  geom_histogram(binwidth = 2, alpha = 0.7, position = "identity") +
  # facet_wrap(~lake, ncol = 1) +
  facet_grid(lake~.)
```

```
Warning: Removed 268 rows containing non-finite outside the scale range
(`stat_bin()`).
```



Activity 2

Choose two new lakes to compare. What differences do you notice in their distributions?

Add notes here

Part 2: Sample Size Effects

Let's explore how the sample size affects what we see.

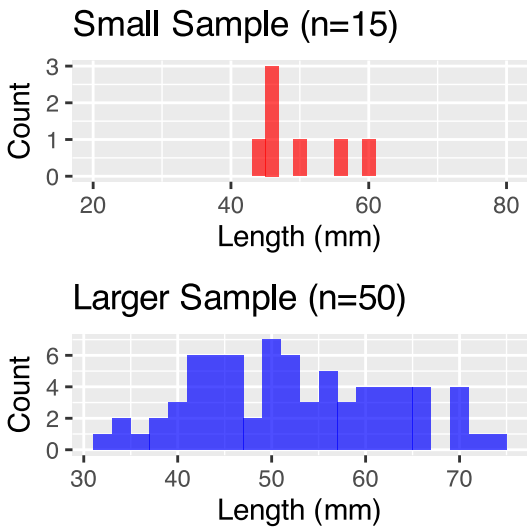
Small vs. Large Samples

We'll randomly select different sample sizes from Toolik Lake:

```
# Set a seed for reproducibility
set.seed(123)

# Create small sample (15 fish)
small_sample <- s_df %>%
  filter(lake == "Toolik") %>%
  sample_n(10)

# Create larger sample (50 fish)
larger_sample <- s_df %>%
  filter(lake == "Toolik") %>%
  sample_n(100)

# Plot both samples
p1 <- small_sample %>%
  ggplot(aes(x = length_mm)) +
  geom_histogram(binwidth = 2, fill = "red", alpha = 0.7) +
  # coord_cartesian(xlim = c(20,80)) +
  labs(title = "Small Sample (n=15)",
       x = "Length (mm)",
       y = "Count") +
  coord_cartesian(xlim = c(20,80))

p2 <- larger_sample %>%
  ggplot(aes(x = length_mm)) +
  geom_histogram(binwidth = 2, fill = "blue", alpha = 0.7) +
  # coord_cartesian(xlim = c(20,80)) +
  labs(title = "Larger Sample (n=50)",
       x = "Length (mm)",
       y = "Count")


# Display the plots side by side
p1 + p2 +
  plot_layout(ncol = 1)
```

```
Warning: Removed 3 rows containing non-finite outside the scale range
(`stat_bin()`).
```

```
Warning: Removed 25 rows containing non-finite outside the scale range
(`stat_bin()`).
```


Small Sample (n=15)
Count
3
2
1
0
20
40
60
80
Length (mm)

Larger Sample (n=50)
Count
6
4
2
0
30
40
50
60
70
Length (mm)

Activity 3

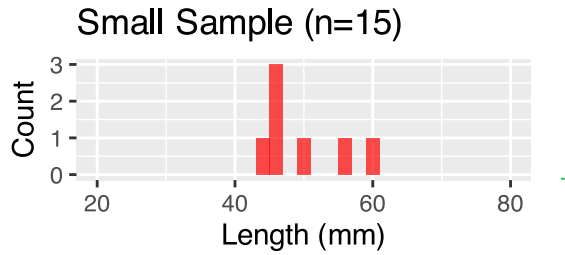
Try changing the sample sizes. What happens when you use very small samples (n=5)? What about larger samples (n=150)?

add code here

```
# Set a seed for reproducibility
set.seed(123)

# Create small sample (15 fish)
small_sample <- s_df %>%
  filter(lake == "Toolik") %>%
  sample_n(10) # CHANGE NUMBERS HERE --------------------------------

# Create larger sample (50 fish)
larger_sample <- s_df %>%
  filter(lake == "Toolik") %>%
  sample_n(100) # CHANGE NUMBERS HERE --------------------------------

# Plot both samples
p1 <- small_sample %>%
  ggplot(aes(x = length_mm)) +
  geom_histogram(binwidth = 2, fill = "red", alpha = 0.7) +
  # coord_cartesian(xlim = c(20,80)) +
  labs(title = "Small Sample (n=15)",
       x = "Length (mm)",
       y = "Count") +
  coord_cartesian(xlim = c(20,80))

p2 <- larger_sample %>%
  ggplot(aes(x = length_mm)) +
  geom_histogram(binwidth = 2, fill = "blue", alpha = 0.7) +
  # coord_cartesian(xlim = c(20,80)) +
  labs(title = "Larger Sample (n=50)",
       x = "Length (mm)",
       y = "Count")


# Display the plots side by side
p1 + p2 +
  plot_layout(ncol = 1)
```

```
Warning: Removed 3 rows containing non-finite outside the scale range
(`stat_bin()`).
```

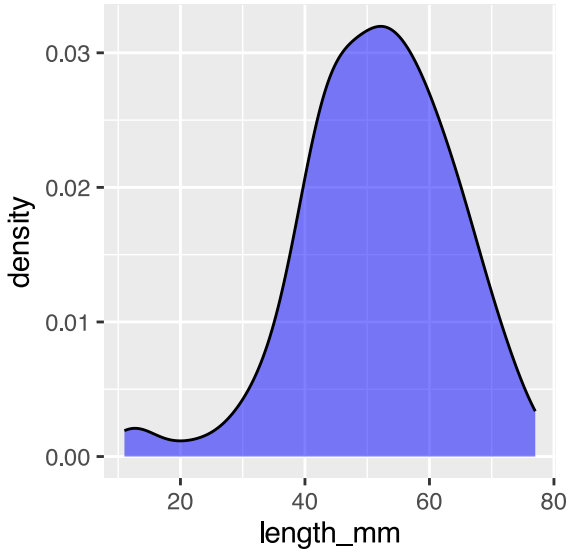
```
Warning: Removed 25 rows containing non-finite outside the scale range
(`stat_bin()`).
```



Part 3: From Histograms to Density Plots

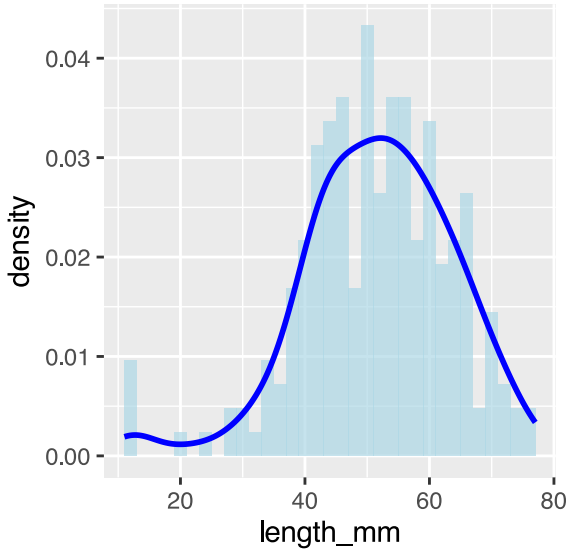
Density plots give us a smoothed version of the histogram:

```
# Create a density plot
s_df %>%
  filter(lake == "Toolik") %>%
  ggplot(aes(x = length_mm)) +
  geom_density(fill = "blue", alpha = 0.5)
```



We can overlay the histogram and the density plot:

```
# Combine histogram and density plot
s_df %>%
  filter(lake == "Toolik") %>%
  ggplot(aes(x = length_mm)) +
  geom_histogram(aes(y = after_stat(density)), binwidth = 2,
                 fill = "lightblue", alpha = 0.7) +
  geom_density(color = "blue", linewidth = 1)
```



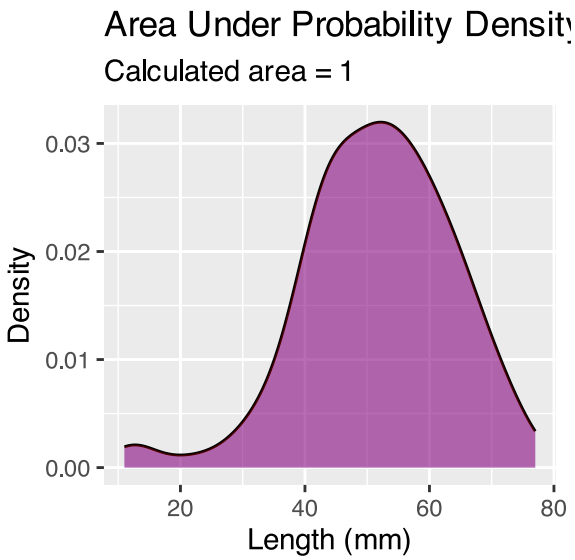
Activity 4

Create a density plot comparing multiple lakes. Which lakes have similar distributions? Which ones are different?

Try code here using patchwork or facet_grid

```
#Enter code here#
```

```
# Function to calculate area under density curve
calculate_density_area <- function(data_vector) {
  # Remove NA values
  data_vector <- data_vector[!is.na(data_vector)]

  # Calculate density
  dens <- density(data_vector)

  # Calculate area using numeric integration (trapezoidal rule)
  # Area should be approximately 1
  dx <- diff(dens$x)
  y_avg <- (dens$y[-1] + dens$y[-length(dens$y)]) / 2
  area <- sum(dx * y_avg)
  return(area)
}

# Apply to Toolik lake data
toolik_data <- s_df %>%
  filter(lake == "Toolik") %>%
  pull(length_mm)

area_value <- calculate_density_area(toolik_data)

# Create plot with calculated area
s_df %>%
  filter(lake == "Toolik") %>%
  ggplot(aes(x = length_mm)) +
  geom_density(fill = "blue", alpha = 0.4) +
  geom_area(stat = "density", fill = "red", alpha = 0.3) +
  labs(title = "Area Under Probability Density Function = 1",
       subtitle = paste("Calculated area =", round(area_value, 4)),
       x = "Length (mm)",
       y = "Density")
```



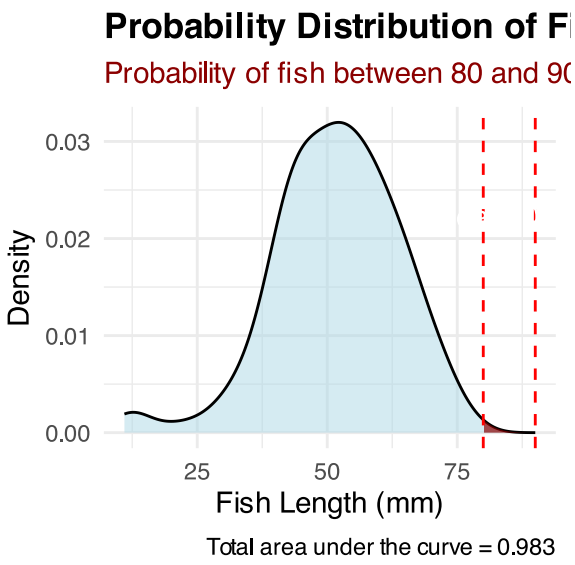
This can be adapted to calculate the area of a subset of the plot

I don't expect you to know or be able to do all of this but is here to play with the code

```
# ------- PART 3: SET  INPUT VALUES -------
# change these values to calculate different probabilities
# For this example, let's calculate the probability of fish between 40mm and 60mm
lower_bound <- 80  # change this value
upper_bound <- 90  # change this value

# ------- PART 1: PREPARE THE DATA -------
# Filter data for just one lake to keep it simple for students
toolik_fish <- s_df %>%
  filter(lake == "Toolik") %>%
  filter(!is.na(length_mm))  # Remove any missing values

# ------- PART 2: CREATE A FUNCTION TO CALCULATE PROBABILITY -------
# This function calculates the probability of finding a fish with length between
# lower_bound and upper_bound using the empirical distribution of our data
calculate_probability <- function(data_vector, lower_bound, upper_bound) {
  # First, we create a density object from our data
  dens <- density(data_vector)

  # Find indices of x-values that fall within our bounds
  indices <- which(dens$x >= lower_bound & dens$x <= upper_bound)

  # If we have no points in the range, return 0
  if(length(indices) <= 1) {
    return(0)
  }

  # Get x and y values within our bounds
  x_values <- dens$x[indices]
  y_values <- dens$y[indices]

  # Calculate the area using the trapezoidal rule
  # (average height × width) for each segment, then sum all segments
  widths <- diff(x_values)
  avg_heights <- (y_values[-1] + y_values[-length(y_values)]) / 2
```

```r
  area_in_range <- sum(widths * avg_heights)

  # Return the calculated probability
  return(area_in_range)
}

# ------- PART 4: CALCULATE THE PROBABILITY -------
# Calculate the probability for the specified range
probability <- calculate_probability(toolik_fish$length_mm, lower_bound, upper_bound)

# Calculate the total area to show that the complete distribution sums to approximately 1
total_area <- calculate_probability(toolik_fish$length_mm,
                                    min(toolik_fish$length_mm),
                                    max(toolik_fish$length_mm))

# ------- PART 5: CREATE THE VISUALIZATION -------
# Create density data for the highlighting
density_data <- density(toolik_fish$length_mm)
density_df <- data.frame(x = density_data$x, y = density_data$y)

# Create a subset for the area of interest
highlight_df <- density_df %>%
  filter(x >= lower_bound & x <= upper_bound)

# Create the plot
ggplot(toolik_fish, aes(x = length_mm)) +
  # First, plot the overall density curve in light blue
  geom_density(fill = "lightblue", alpha = 0.5) +

  # Then highlight our region of interest in dark red
  geom_area(data = highlight_df, aes(x = x, y = y),
            fill = "darkred", alpha = 0.7) +

  # Add vertical lines to clearly mark the boundaries
  geom_vline(xintercept = lower_bound, linetype = "dashed", color = "red") +
  geom_vline(xintercept = upper_bound, linetype = "dashed", color = "red") +

  # Add informative labels
  labs(
    title = "Probability Distribution of Fish Lengths",
    subtitle = paste0("Probability of fish between ", lower_bound,
                      " and ", upper_bound, " mm = ",
                      round(probability * 100, 1), "%"),
    caption = paste("Total area under the curve =", round(total_area, 3)),
    x = "Fish Length (mm)",
    y = "Density"
  ) +

  # Add text annotations to explain the areas
  annotate("text", x = (lower_bound + upper_bound)/2,
           y = max(density(toolik_fish$length_mm)$y) * 0.7,
           label = paste0("Area = ", round(probability, 3)),
           color = "white", size = 4) +

  # Make the plot look nicer
  theme_minimal() +
  theme(
```

```
    plot.title = element_text(face = "bold"),
    plot.subtitle = element_text(color = "darkred")
  )
```



Part 4: Summary Statistics - descriptive statistics

Let's calculate basic summary statistics for each lake:

```
# Calculate mean, standard deviation, and sample size by lake
s_df %>%
  group_by(lake) %>%
  summarize(
    mean_length = mean(length_mm),
    sd_length = sd(length_mm),
    count = n(),
    .groups = "drop"
  ) %>%
  arrange(desc(count))
```

```
# A tibble: 7 × 4
  lake   mean_length sd_length count
  <chr>        <dbl>     <dbl> <int>
1 Toolik        NA        NA     287
2 E 01          NA        NA     268
3 NE 12         49.8      15.2   180
4 S 06          54.0      10.9   132
5 E 05          NA        NA      75
6 S 07          55.6      12.7    73
7 NE 14         47.3      10.5    37
```

WOAH - what happened there - there are NA values in the data

you need to either remove missing values or you can do that in the formulas

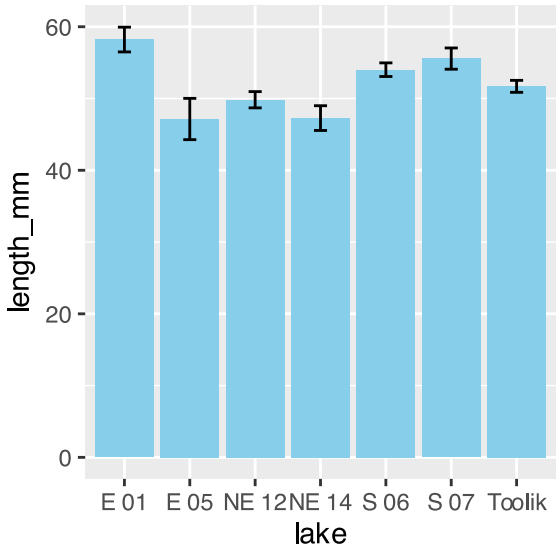
What is the advantage to manually removing or doing it in formulas?

```r
# Calculate mean, standard deviation, and sample size by lake
sculpin_stats_df <- s_df %>%
  group_by(lake) %>%
  summarize(
    mean_length = mean(length_mm, na.rm = TRUE),
    sd_length = sd(length_mm, na.rm = TRUE),
    se_length = sd(length_mm, na.rm = TRUE)/ sum(!is.na(length_mm))^.5,
    count = sum(!is.na(length_mm)),
    .groups = "drop"
  ) %>%
  arrange(desc(count))
sculpin_stats_df
```

```
# A tibble: 7 × 5
  lake   mean_length sd_length se_length count
  <chr>        <dbl>     <dbl>     <dbl> <int>
1 Toolik        51.7      12.0     0.834   208
2 NE 12         49.8      15.2     1.13    180
3 S 06          54.0      10.9     0.949   132
4 E 01          58.2      15.3     1.72     79
5 S 07          55.6      12.7     1.48     73
6 NE 14         47.3      10.5     1.72     37
7 E 05          47.1      10.8     2.88     14
```

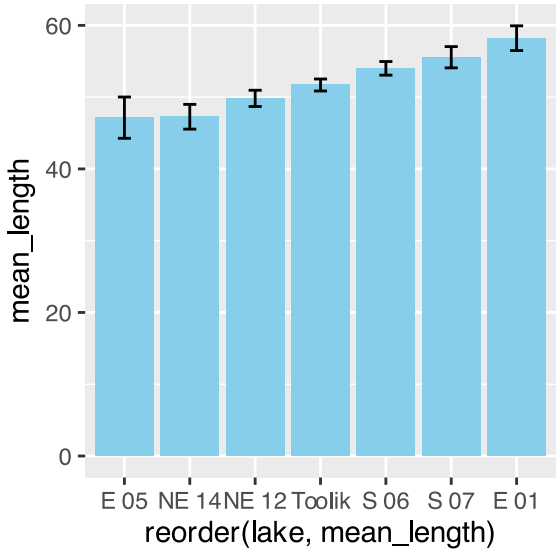
Now let's visualize these statistics:

```r
# Create a bar plot of mean lengths with error bars
s_df %>%
  ggplot(aes(lake, length_mm)) +
  stat_summary(
    fun = mean, na.rm = TRUE,
    geom = "bar",
    fill = "skyblue"
    ) +
  stat_summary(
    fun.data = mean_se, na.rm = TRUE,
    geom = "errorbar",
    width = 0.2)
```



We could also do this from the dataframe we just made

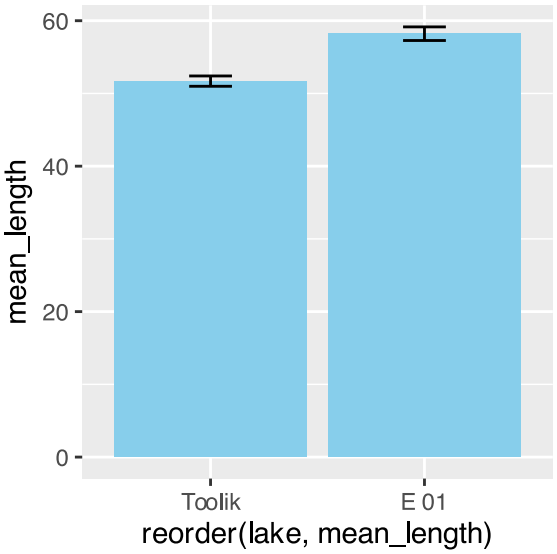
```
# Create a bar plot of mean lengths with error bars
sculpin_stats_df %>%
  ggplot(aes(x = reorder(lake, mean_length), y = mean_length)) +
  geom_bar(stat = "identity",
           fill = "skyblue") +
  geom_errorbar(aes(
    ymin = mean_length - se_length,
    ymax = mean_length + se_length),
    width = 0.2
    )
```



The power of the pipe command is you can do this without hving to make a new dataframe

```
# Create a bar plot of mean lengths with error bars
s_df %>%
  group_by(lake) %>%
  summarize(
    mean_length = mean(length_mm, na.rm = TRUE),
```

```
    sd_length = sd(length_mm, na.rm = TRUE),
    se_length = sd_length / sqrt(n()),
    count = n(),
    .groups = "drop"
  ) %>%
  filter(count >= 250) %>%  # Only include lakes with sufficient sample size
  ggplot(aes(x = reorder(lake, mean_length), y = mean_length)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  geom_errorbar(aes(ymin = mean_length - se_length,
                    ymax = mean_length + se_length),
                width = 0.2)
```



Activity 5

Based on the mean plot and what you've seen in the distributions, what can you say about fish sizes in different lakes? Are there lakes with particularly large or small fish?

We will start to ask how different are they and is it by chance?

Where would you want to fish and why? What is the chance of catching a fish greater than X size?

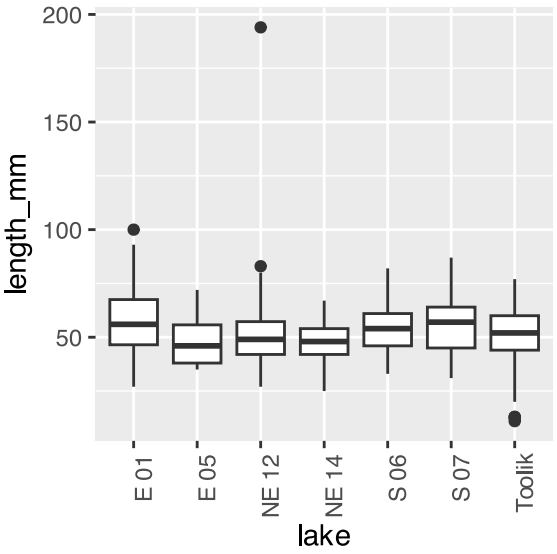
This is the inductive phase of doing research.

Part 5: Guided Challenges

Now it's your turn to explore the data! Work with your partner to complete these challenges:

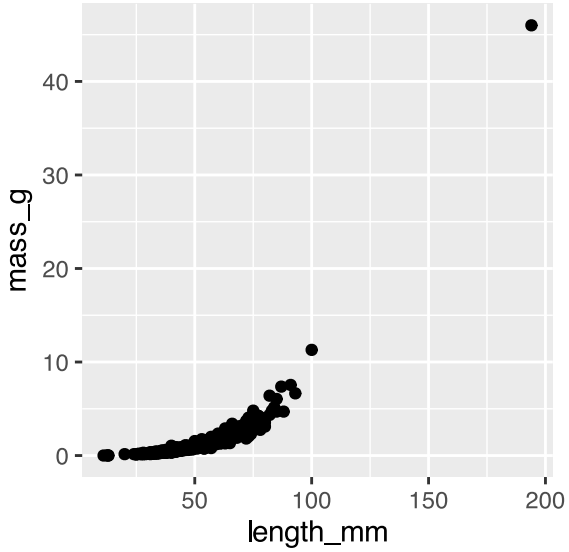
1. Find the lake with the widest range of fish lengths (hint: use the `range()` function)
2. Create box and whisker plots to compare fish lengths across lakes:

```
# Example boxplot code to get you started
s_df %>%
  filter(!is.na(length_mm)) %>%
  ggplot(aes(x = lake, y = length_mm)) +
  geom_boxplot() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



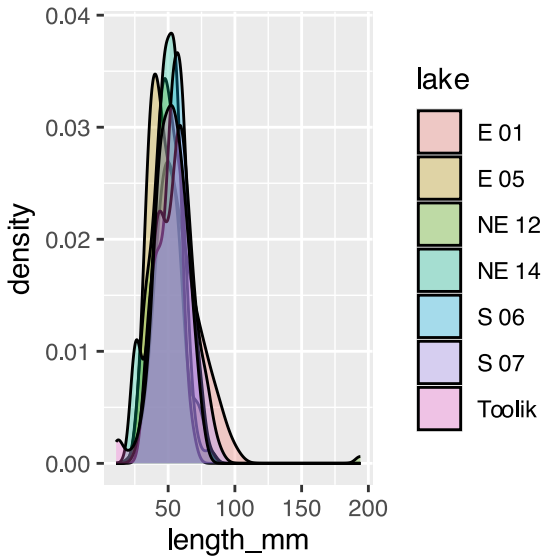
3. Explore if there's a relationship between fish length and mass:

```
# Starting code for length-mass relationship
s_df %>%
  filter(!is.na(length_mm), !is.na(mass_g)) %>%
  ggplot(aes(x = length_mm, y = mass_g)) +
  geom_point()
```



4. Try creating a density plot that shows all lakes in different colors:

```
# Starting code for multi-lake density plot
s_df %>%
  filter(!is.na(length_mm)) %>%
  ggplot(aes(x = length_mm, fill = lake)) +
  geom_density(alpha = 0.3)
```



Reflection Questions

After completing the activities, discuss these questions with your group:

1. How does sample size affect our view of a population's characteristics?
2. Why might fish lengths be different in different lakes?
3. What are the advantages and disadvantages of histograms versus density plots?
4. What additional data would help you better understand these fish populations?