Assignment 5: Data Visualization

Brian Mulu Mutua

Fall 2023

OVERVIEW

This exercise accompanies the lessons in Environmental Data Analytics on Data Visualization

Directions

1. Rename this file `<FirstLast>_A05_DataVisualization.Rmd` (replacing `<FirstLast>` with your first and last name).
2. Change "Student Name" on line 3 (above) with your name.
3. Work through the steps, **creating code and output** that fulfill each instruction.
4. Be sure your code is tidy; use line breaks to ensure your code fits in the knitted output.
5. Be sure to **answer the questions** in this assignment document.
6. When you have completed the assignment, **Knit** the text and code into a single PDF file.

Set up your session

1. Set up your session. Load the tidyverse, lubridate, here & cowplot packages, and verify your home directory. Read in the NTL-LTER processed data files for nutrients and chemistry/physics for Peter and Paul Lakes (use the tidy `NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv` version in the Processed_KEY folder) and the processed data file for the Niwot Ridge litter dataset (use the `NEON_NIWO_Litter_mass_trap_Processed.csv` version, again from the Processed_KEY folder).

2. Make sure R is reading dates as date format; if not change the format to date.

```
#1 Initial setup
#Checking working directory
getwd()
```

```
## [1] "C:/Users/bmm100/Documents/EDE_Fall2023"
```

```
#Loading necessary libraries
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
```

```
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(lubridate)
library(here)
```

```
## here() starts at C:/Users/bmm100/Documents/EDE_Fall2023
```

```
library(cowplot)
```

```
##
## Attaching package: 'cowplot'
##
## The following object is masked from 'package:lubridate':
##
##     stamp
```

```
library(ggthemes)
```

```
##
## Attaching package: 'ggthemes'
##
## The following object is masked from 'package:cowplot':
##
##     theme_map
```

```
#Reading processed data files
processedNTL.LTER.data <-
↪  read.csv(here("Data/Processed_KEY/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv"))
processedNiwot.Ridge.data <-
↪  read.csv(here("Data/Processed_KEY/NEON_NIWO_Litter_mass_trap_Processed.csv"))

#2 Making sure R is reading dates as date format
#Checking date format of the loaded data for the North Temperate Lakes LTER Data
glimpse(processedNTL.LTER.data$sampledate)
```

```
##  chr [1:23008] "1984-05-27" "1984-05-27" "1984-05-27" "1984-05-27" ...
```

```
#Adjusting date format using lubridate
processedNTL.LTER.data$sampledate <- ymd(processedNTL.LTER.data$sampledate)
#Using class function to show that date format has been updated successfully
glimpse(processedNTL.LTER.data$sampledate)
```

```
##  Date[1:23008], format: "1984-05-27" "1984-05-27" "1984-05-27" "1984-05-27" "1984-05-27" ...
```

```
#Checking date format of the loaded data for the Niwot Ridge Long-Term Ecological
↪  Research (LTER) station
glimpse(processedNiwot.Ridge.data$collectDate)
```

```
##  chr [1:1692] "2016-06-16" "2016-06-16" "2016-06-16" "2016-06-16" ...
```

```
processedNiwot.Ridge.data$collectDate <- ymd(processedNiwot.Ridge.data$collectDate)
#Using glimpse function to show that date format has been updated successfully
glimpse(processedNiwot.Ridge.data$collectDate)
```

```
##  Date[1:1692], format: "2016-06-16" "2016-06-16" "2016-06-16" "2016-06-16" "2016-06-16" ...
```

Define your theme

3. Build a theme and set it as your default theme. Customize the look of at least two of the following:

- Plot background
- Plot title
- Axis labels
- Axis ticks/gridlines
- Legend

```
#3 Building personal theme
brian_theme_A05 <- theme_base()  +
  theme(
    #Modifying colour and size of elements
    line =  element_line(colour = "grey10"),
    rect =  element_rect(colour = "grey10",
                         fill = "Honeydew"),
    text =  element_text(colour = "grey10",size = 11),

    #Modifying plot and axis titles
    plot.title = element_text(family = "sans",
                              face = "bold",
                              size = 16,colour = "grey10"),
    axis.title.x = element_text(family="sans",
                                size = 11,
                                colour = "grey10",
                                face = "bold"),
    axis.title.y = element_text(family="sans",
                                size = 11,
                                colour = "grey10",
                                face = "bold"),
    axis.text = element_text(family="sans",
                             size = 11,
                             colour = "grey10"),

    #Modifying grid line, axis ticks and plot margin
    axis.ticks =element_blank(),
    panel.grid.major =  element_line(colour = "grey80"),
```

```
    panel.grid.minor =  element_blank(),
    panel.border = element_rect(colour = "grey80"),

    #Modifying bacjground colour
    plot.background = element_rect(fill = "Honeydew",
                                   colour = NA),
    panel.background =  element_rect(fill = "Honeydew",
                                     colour = "grey80"),
    legend.key =  element_rect(fill="Honeydew"),

    #Modifying legend
    legend.position = "bottom",
    legend.background = element_rect(colour = "grey10"),
    legend.text = element_text(colour = "grey10",
                               size = 11)
    )

#Setting personal theme as default theme
theme_set(brian_theme_A05)
```

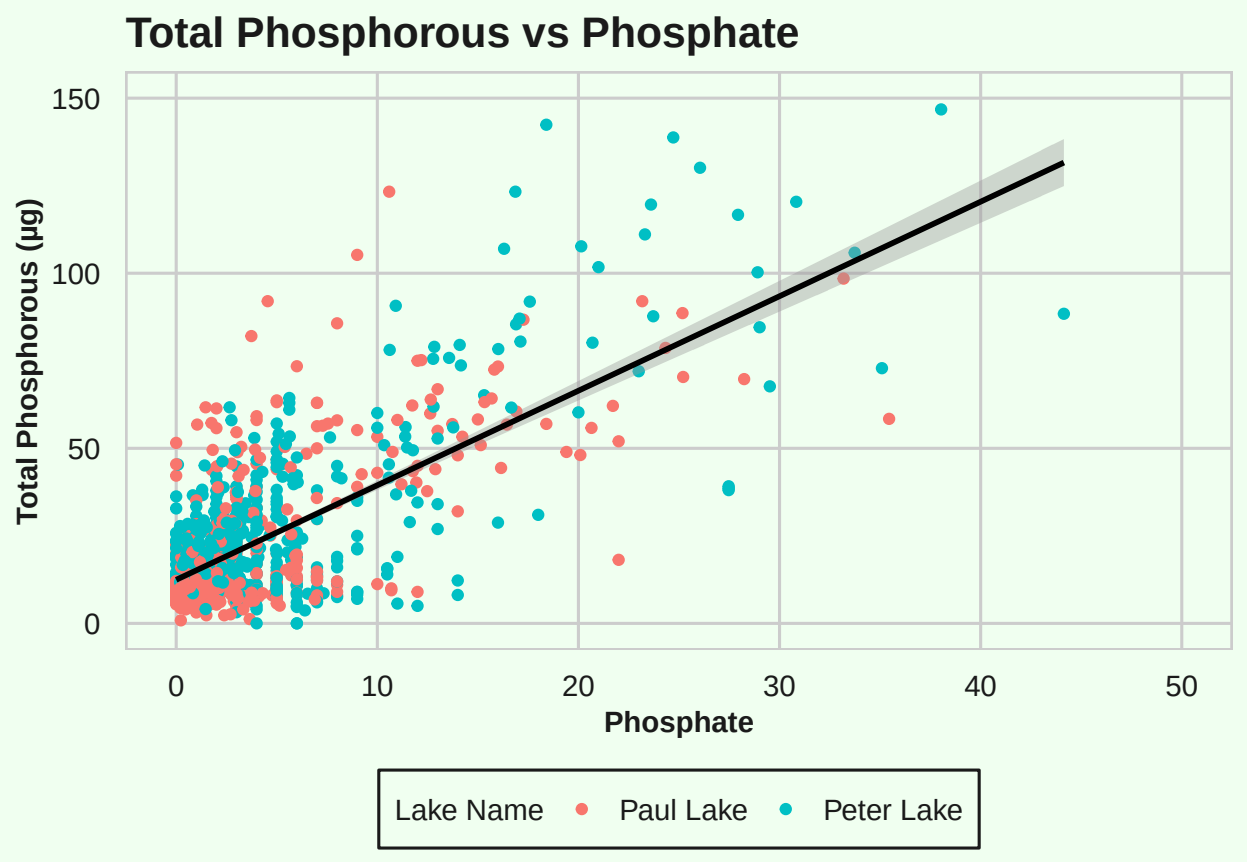
Create graphs

For numbers 4-7, create ggplot graphs and adjust aesthetics to follow best practices for data visualization. Ensure your theme, color palettes, axes, and additional aesthetics are edited accordingly.

4. [NTL-LTER] Plot total phosphorus (`tp_ug`) by phosphate (`po4`), with separate aesthetics for Peter and Paul lakes. Add a line of best fit and color it black. Adjust your axes to hide extreme values (hint: change the limits using `xlim()` and/or `ylim()`).

```
#4 Setting up plot
NTL.LTER.Question4Plot <- processedNTL.LTER.data %>%
  ggplot(aes(x=po4,y=tp_ug,color=lakename)) +
  geom_point() + xlim(0,50) + ylim(0,150) +
  geom_smooth(method=lm,
              color="black")+
  labs(y = "Total Phosphorous (\u00B5g)",
       x = "Phosphate",
       colour = "Lake Name",
       title = "Total Phosphorous vs Phosphate")

#Printing out plot
NTL.LTER.Question4Plot
```

```
## `geom_smooth()` using formula = 'y ~ x'

## Warning: Removed 21948 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21948 rows containing missing values (`geom_point()`).
```



5. [NTL-LTER] Make three separate boxplots of (a) temperature, (b) TP, and (c) TN, with month as the x axis and lake as a color aesthetic. Then, create a cowplot that combines the three graphs. Make sure that only one legend is present and that graph axes are aligned.

Tip: * Recall the discussion on factors in the previous section as it may be helpful here. * R has a built-in variable called `month.abb` that returns a list of months;see https://r-lang.com/month-abb-in-r-with-example

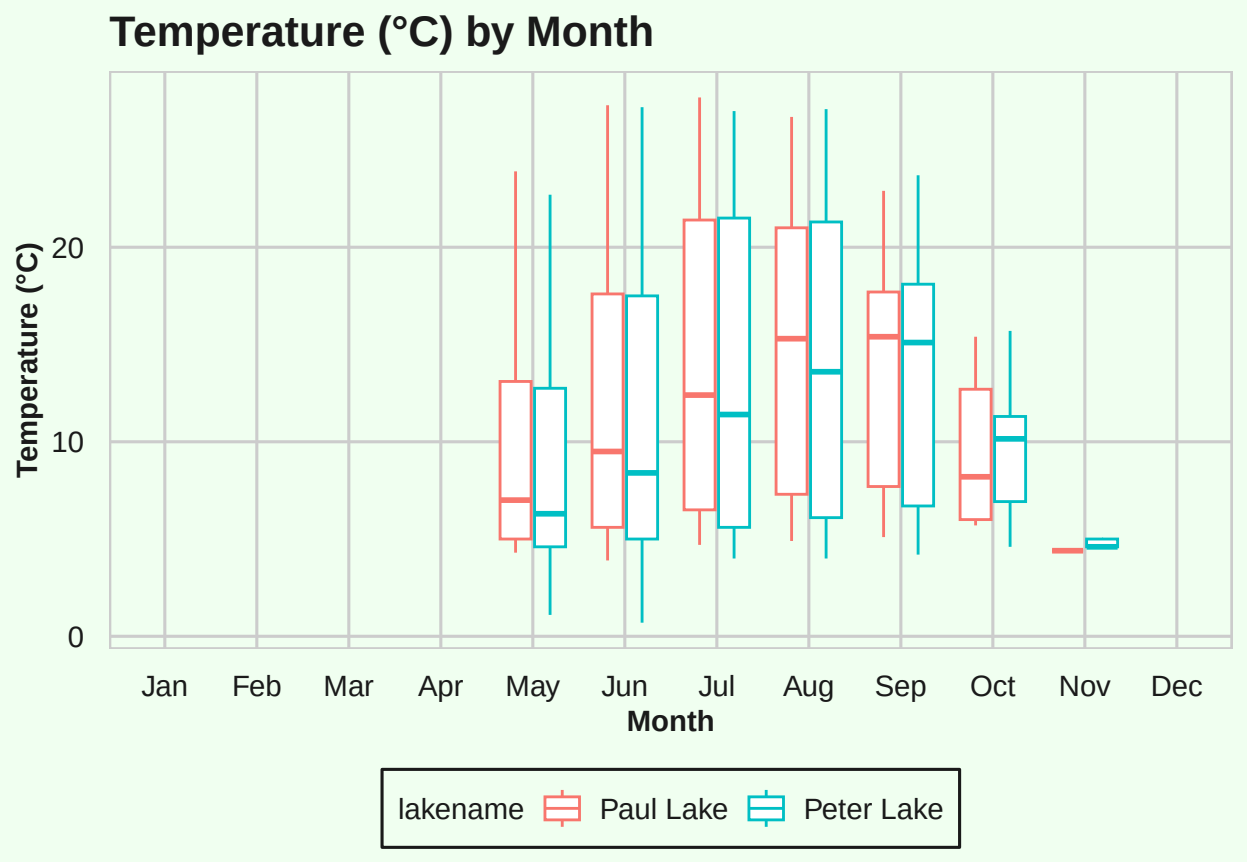
```
#5 Setting up the boxplots
#Checking how month variables are stored
glimpse(processedNTL.LTER.data$month)
```

```
##  int [1:23008] 5 5 5 5 5 5 5 5 5 5 ...
```

```
#Since month is stored as an integer, it needs to be converted to a factor to probably
↪  plot the months on the x axis. This is done in the respective pipes for the different
↪  visualizations outlined below.

#Setting up boxplots of temperatures in the different months of the year
temperatureBoxplot.NTL.LTER <- processedNTL.LTER.data %>%
  ggplot(aes(x=factor(month, levels=1:12, labels=month.abb),
             y=temperature_C,
             color=lakename))+
  geom_boxplot()+
  scale_x_discrete(name="Month",
```

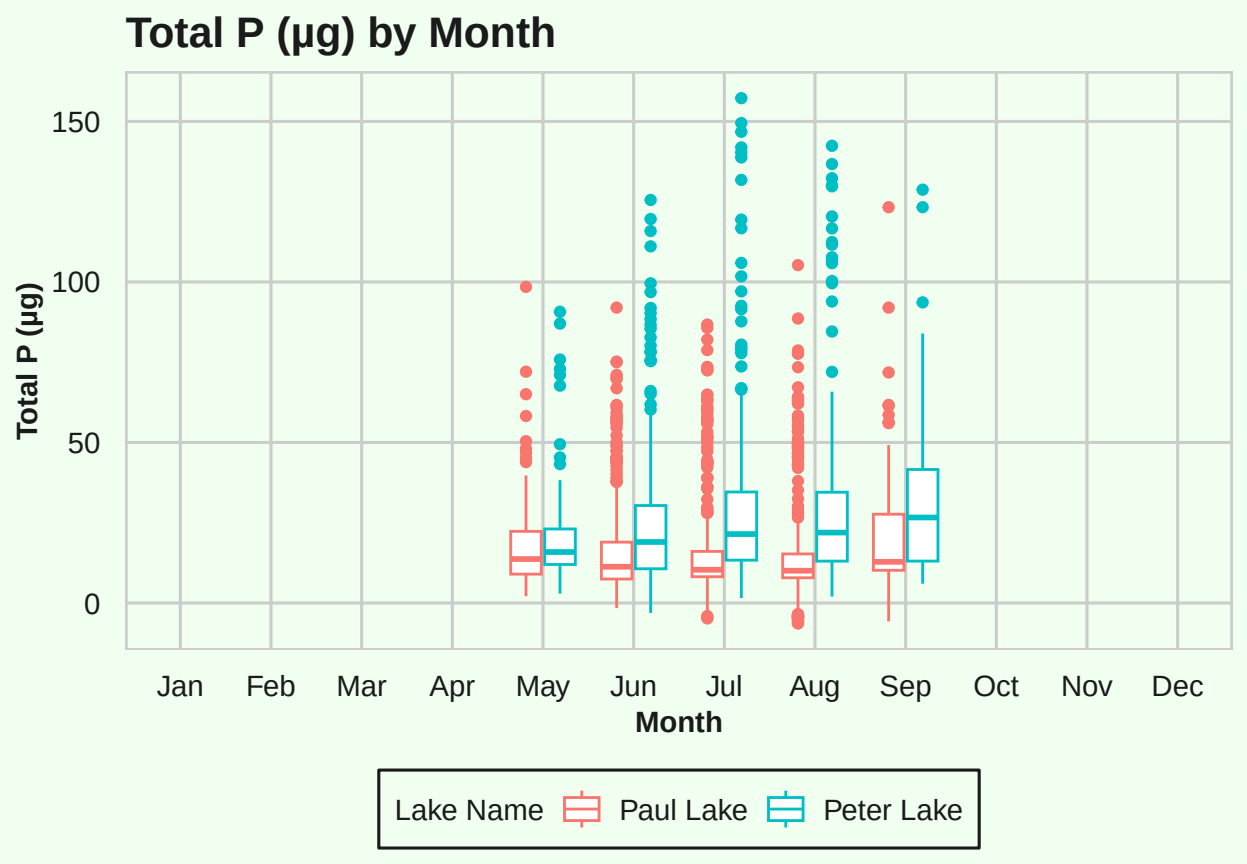
```
                  drop=FALSE)+
  labs(y="Temperature (\u00B0C)",
       title = "Temperature (\u00B0C) by Month")

#Displaying temperature box plot
temperatureBoxplot.NTL.LTER
```

```
## Warning: Removed 3566 rows containing non-finite values (`stat_boxplot()`).
```



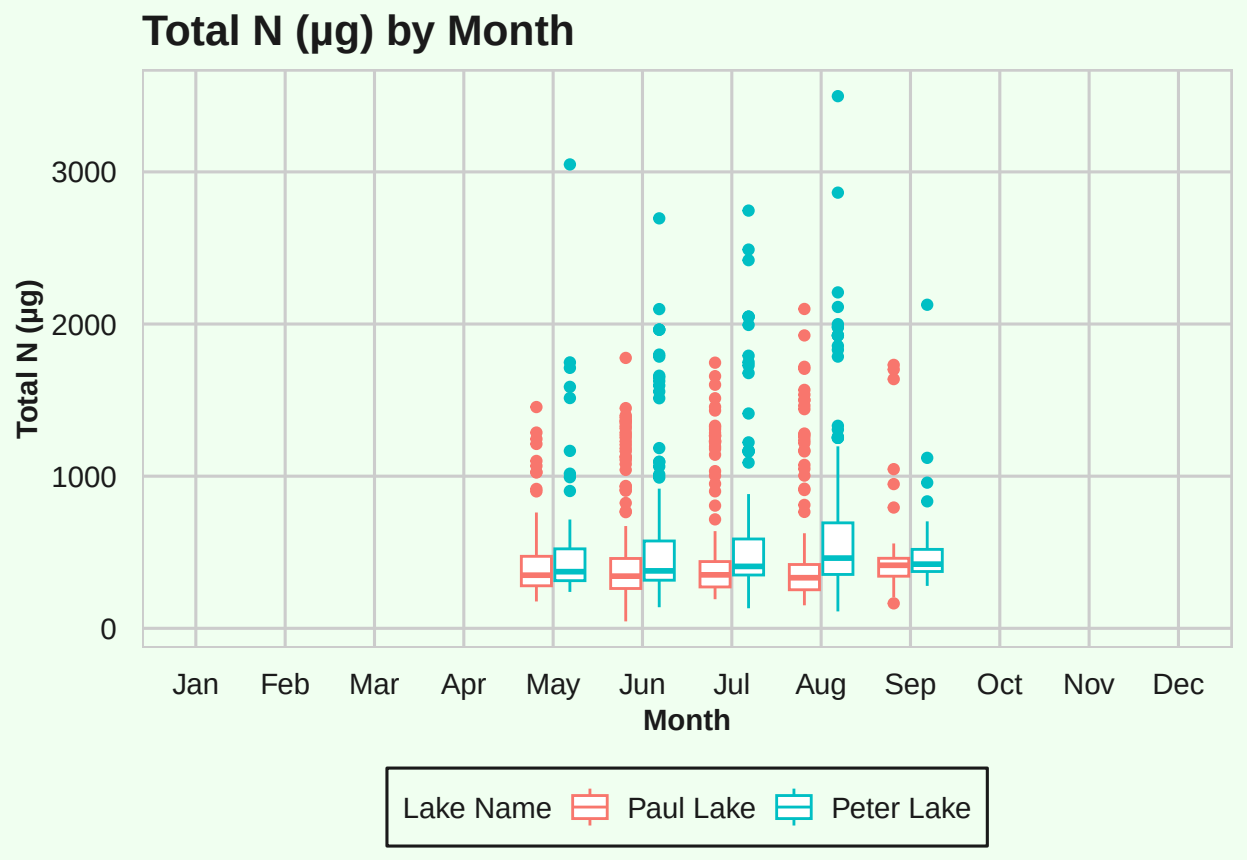

```
#Setting up box plots of TP in the different months of the year
TotalPBoxplot.NTL.LTER <- processedNTL.LTER.data %>%
  ggplot(aes(x=factor(month,levels=1:12,labels=month.abb)
             ,y=tp_ug,color=lakename))+
  geom_boxplot()+
  scale_x_discrete(name="Month",drop=FALSE)+
  labs(y="Total P (\u00B5g)",
       colour = "Lake Name",
       title = "Total P (\u00B5g) by Month")

#Displaying total P box plot
TotalPBoxplot.NTL.LTER
```

```
## Warning: Removed 20729 rows containing non-finite values (`stat_boxplot()`).
```




```
#Setting up box plots of TN in the different months of the year
TotalNBoxplot.NTL.LTER <- processedNTL.LTER.data %>%
  ggplot(aes(x=factor(month,levels=1:12,labels=month.abb),
             y=tn_ug,color=lakename))+
  geom_boxplot()+
  scale_x_discrete(name="Month",drop=FALSE)+
  labs(y="Total N (\u00B5g)",
       colour = "Lake Name",
       title = "Total N (\u00B5g) by Month")

#Displaying total N box plot
TotalNBoxplot.NTL.LTER
```

```
## Warning: Removed 21583 rows containing non-finite values (`stat_boxplot()`).
```



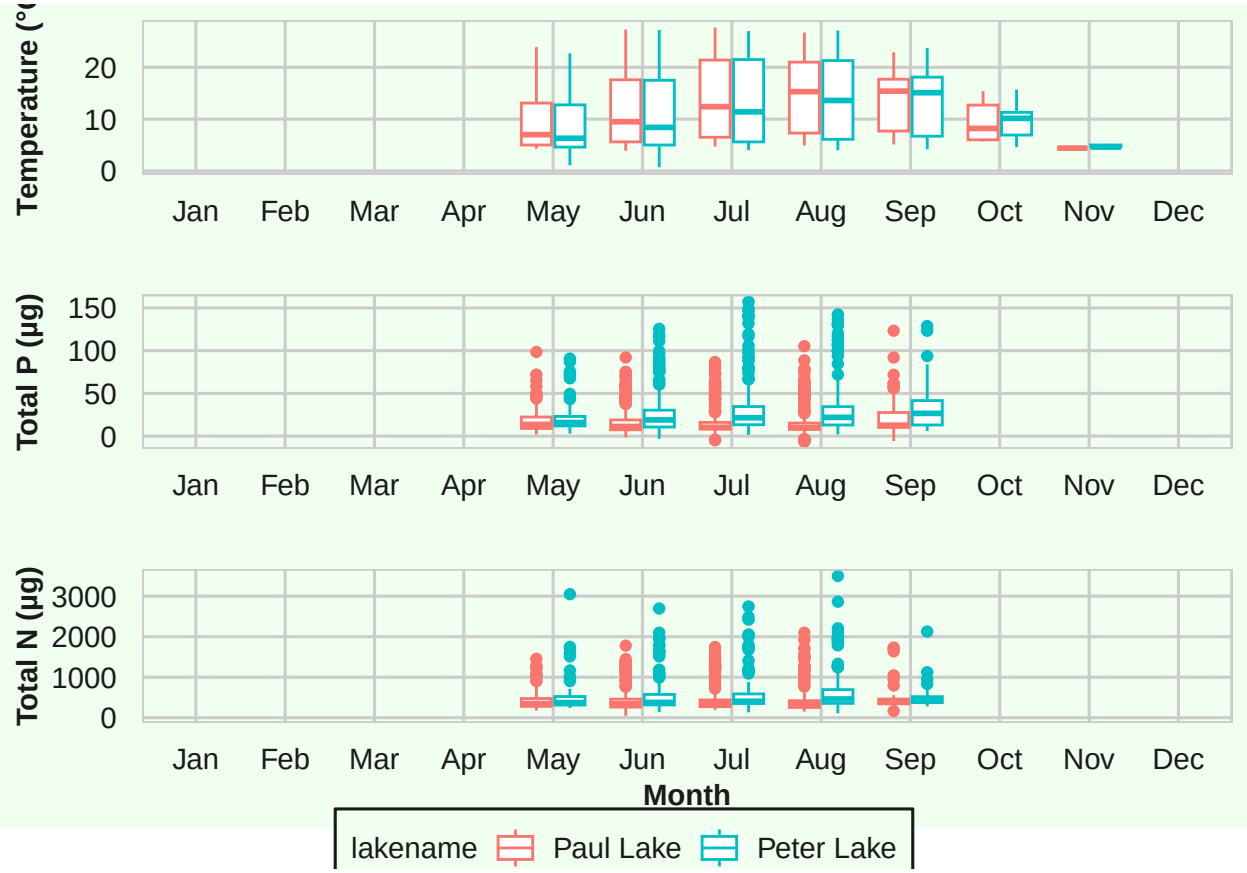

```
#Consolidating the 3 boxplots created above in one cowplot
cowplotWithoutLegend <- plot_grid(temperatureBoxplot.NTL.LTER+
            theme(legend.position ="none",
                  axis.title.x = element_blank(),
                  plot.title = element_blank()),
          TotalPBoxplot.NTL.LTER+
            theme(legend.position ="none",
                  axis.title.x = element_blank(),
                  plot.title = element_blank()),
          TotalNBoxplot.NTL.LTER+
            theme(legend.position ="none",
                  plot.title = element_blank()),
          nrow = 3,
          align = "hv")
```

```
## Warning: Removed 3566 rows containing non-finite values (`stat_boxplot()`).

## Warning: Removed 20729 rows containing non-finite values (`stat_boxplot()`).

## Warning: Removed 21583 rows containing non-finite values (`stat_boxplot()`).
```

```
#Storing legend separately to be added to consolidated cowplot
cowplotlegend <- get_legend(temperatureBoxplot.NTL.LTER+
                              guides(colour=guide_legend(nrow=1))+
                              theme(legend.position = "bottom"))
```

```
## Warning: Removed 3566 rows containing non-finite values (`stat_boxplot()`).
```

```
#Plotting consolidated boxplots with legend
plot_grid(cowplotWithoutLegend, cowplotlegend, ncol = 1, rel_heights = c(1,.05))
```



Question: What do you observe about the variables of interest over seasons and between lakes?

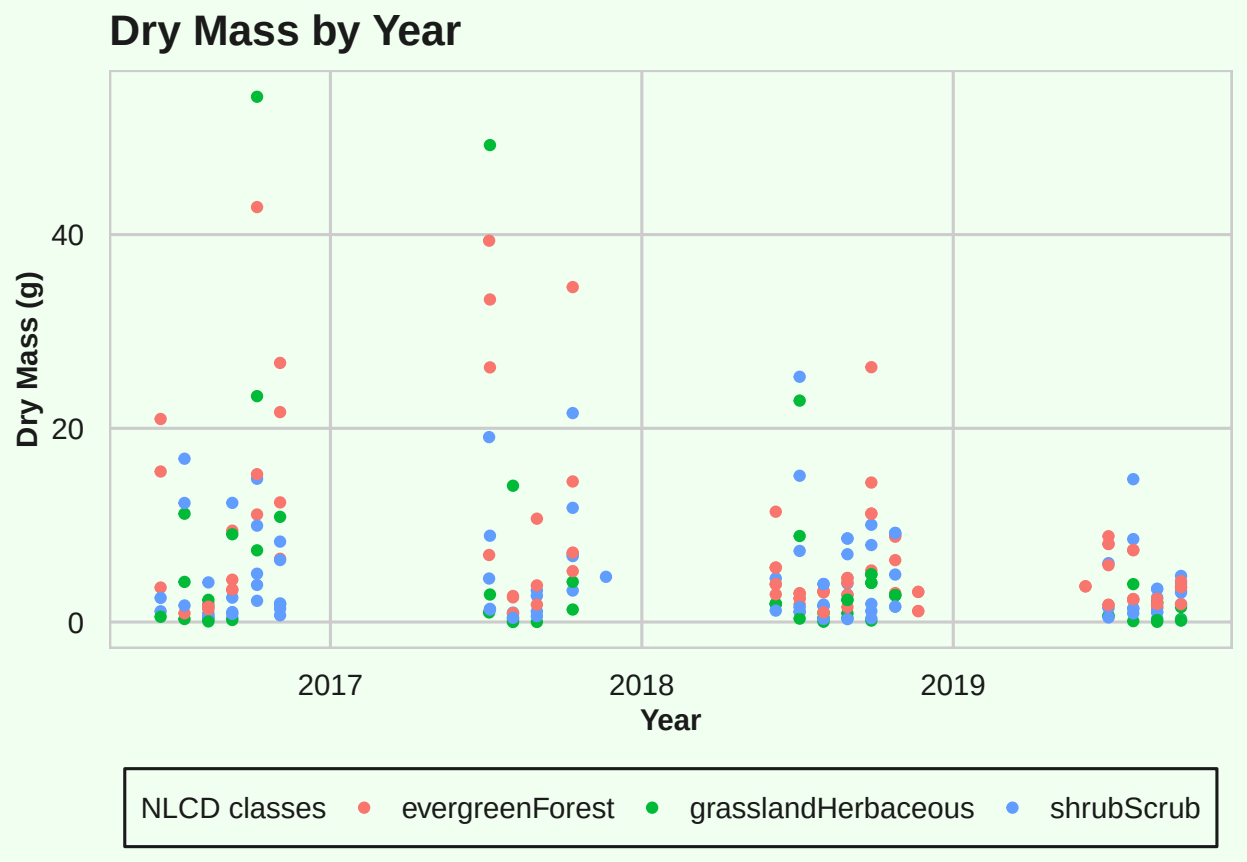
> Answer: From the boxplots, we observe the following concerning the variables: 1. The median measured temperature in both lakes varies seasonally and is highest in the months of August and September. The median temperature rises in the summer months between May and August then decereases in the months of October and November. The range of temperatures measured is also lower in the months of October and November compared to the preceding months. The temperature distributions in both lakes are similarly skewed and fall within a similar range. 2. Total P and N values are fairly concentrated with **several outliers**.The Total P and Total N values for Peter Lake are more widely distributed than those of Paul Lake.

6. [Niwot Ridge] Plot a subset of the litter dataset by displaying only the "Needles" functional group. Plot the dry mass of needle litter by date and separate by NLCD class with a color aesthetic. (no need to adjust the name of each land use)

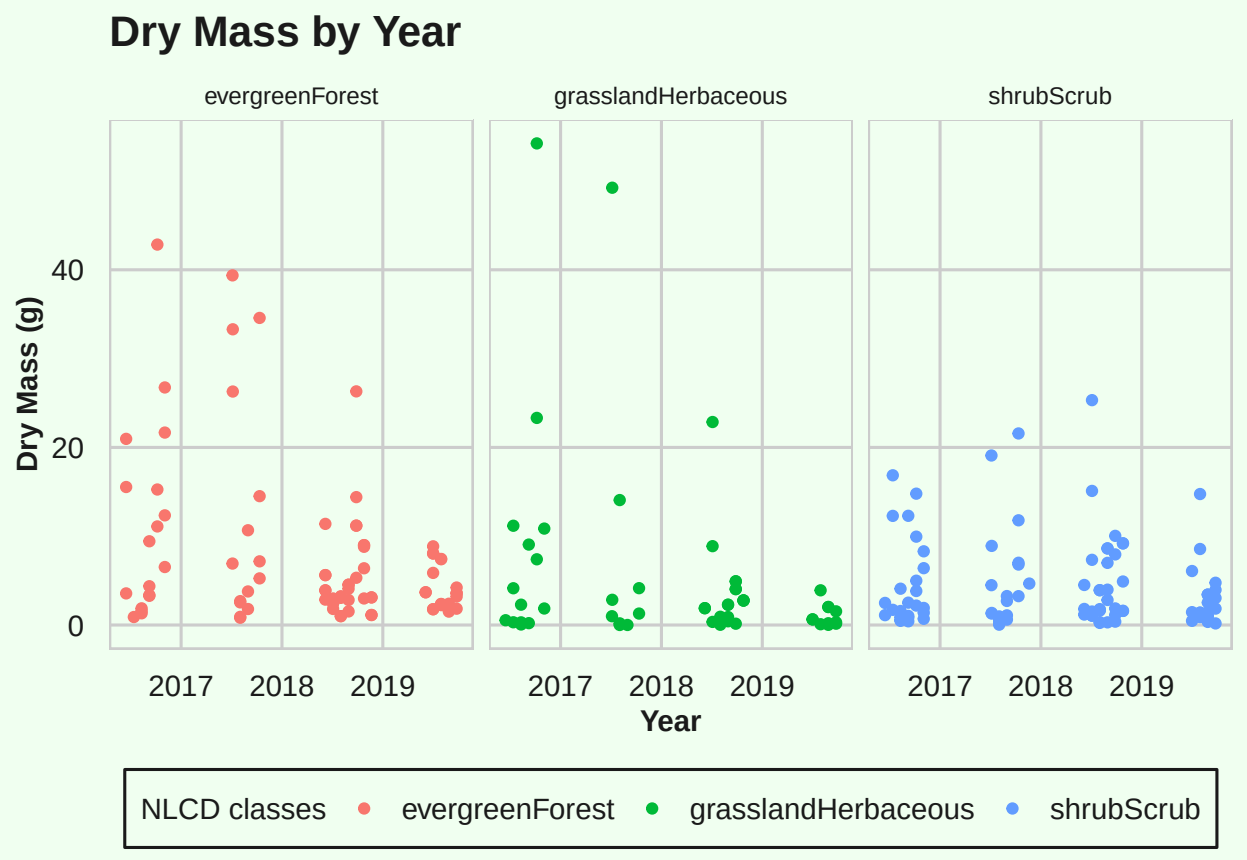
7. [Niwot Ridge] Now, plot the same plot but with NLCD classes separated into three facets rather than separated by color.

```
#6 Plotting a subset of the litter dataset displaying only the "Needles" functional group
NiwotRideDryMassbyDate.plot <- processedNiwot.Ridge.data %>%
```

```
  filter(functionalGroup=="Needles") %>%
  ggplot(aes(x=collectDate,
             y=dryMass,
             color=nlcdClass)) +
  geom_point() +
  labs(y = "Dry Mass (g)",
       x = "Year",
       colour = "NLCD classes",
       title = "Dry Mass by Year")
NiwotRideDryMassbyDate.plot
```




```
#7 Plotting the same data separated into facets
NiwotRideDryMassbyDate.facetedplot <- processedNiwot.Ridge.data %>%
  filter(functionalGroup=="Needles") %>%
  ggplot(aes(x=collectDate,
             y=dryMass,
             color=nlcdClass)) +
  geom_point() +
  facet_wrap(vars(nlcdClass),nrow = 1) +
  theme(legend.position = "bottom") +
  labs(y = "Dry Mass (g)",
       x = "Year",
       colour = "NLCD classes",
       title = "Dry Mass by Year")
NiwotRideDryMassbyDate.facetedplot
```



Question: Which of these plots (6 vs. 7) do you think is more effective, and why?

Answer: Plot 7 is more effective than Plot 6. This is because it is easier to observe the patterns of dryMass data for the different nlcd classes when the data is separated into facets than when all points are consolidated in one view. It is also easier to visually compare the differences in dryMass between the NLCD classes in different years.