Assignment 5: Data Visualization

Lily Zhang

Spring 2024

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.

```r
#1 Load necessary package
library(tidyverse) #For data manipulation and visualization
```

```
## Warning: package 'ggplot2' was built under R version 4.3.1

## Warning: package 'readr' was built under R version 4.3.1

## Warning: package 'dplyr' was built under R version 4.3.1

## Warning: package 'lubridate' was built under R version 4.3.1

## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.1
## -- 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 error
```

```r
library(lubridate) #For working with dates
library(here) #For managing file paths
```

```
## here() starts at /Users/lilyzhang/Desktop/EDA_Spring2024
```

```r
library(cowplot) #For creating complex plots
```

```
## Warning: package 'cowplot' was built under R version 4.3.1

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

```r
#For enhancing visualization
library(ggridges)
```

```
## Warning: package 'ggridges' was built under R version 4.3.1
```

```r
library(viridis)
```

```
## Warning: package 'viridis' was built under R version 4.3.1

## Loading required package: viridisLite
```

```r
library(RColorBrewer)
library(colormap)
library(ggthemes)
```

```
## Warning: package 'ggthemes' was built under R version 4.3.1

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

```r
library(ggplot2)
library(dplyr)
library(formatR)
#Print the current working directory to the console
#Read the csv data into data frame
here()
```

```
## [1] "/Users/lilyzhang/Desktop/EDA_Spring2024"
```

```r
PeterPaul.chem.nutrients <-
  read.csv(here("Data/Processed_KEY/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv"),
           stringsAsFactors = T)
Niwot.litter <-
  read.csv(here("Data/Processed_KEY/NEON_NIWO_Litter_mass_trap_Processed.csv"),
           stringsAsFactors = T)
#2 Convert date columns to date format and check the class
PeterPaul.chem.nutrients$sampledate <-
  ymd(PeterPaul.chem.nutrients$sampledate)
class(PeterPaul.chem.nutrients$sampledate)
```

```
## [1] "Date"
```

```
Niwot.litter$collectDate <-
  ymd(Niwot.litter$collectDate)
class(Niwot.litter$collectDate)
```

```
## [1] "Date"
```

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 Build customized theme
my_theme <- theme_base()  +
  theme(
    line = element_line(color = "#23679e"),
    rect = element_rect(color = "#23679e"),
    text = element_text(color = "#23679e"),
    # Modify inheritance structure of text element
    plot.title =        element_text(color = "#23679e",
                                     size = 12, face = "bold",
                                     margin = margin(5, 0, 5, 0)),
    axis.title.x =      element_text(color = "#23679e",
                                     size = 12),
    axis.title.y =      element_text(color = "#23679e",
                                     size = 12, angle = 90),
    axis.text =         element_text(color ="#23679e", size = 10),
    # Modify inheritance structure of line element
    axis.ticks =        element_line(color = "#8dc1ff"),
    panel.grid.major =  element_line(color = "#8dc1ff", linetype = "dashed"),
    panel.grid.minor =  element_blank(),
    # Modify inheritance structure of rect element
    plot.background =   element_rect(fill = "#deebf7"),
    panel.background =  element_rect(fill = "#deebf7"),
    legend.key =        element_rect(fill = "#deebf7"),
    legend.background = element_rect(fill = "#deebf7"),
    # Modify strip structure
    strip.background = element_rect(fill = "#deebf7"),
    strip.text = element_text(color = "#23679e",
                              margin = margin(5, 0, 5, 0)),
    # Modify legend position
    legend.position = 'right',

    complete = TRUE
    )
#Set the custom theme as the default theme
theme_set(my_theme)
```

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.

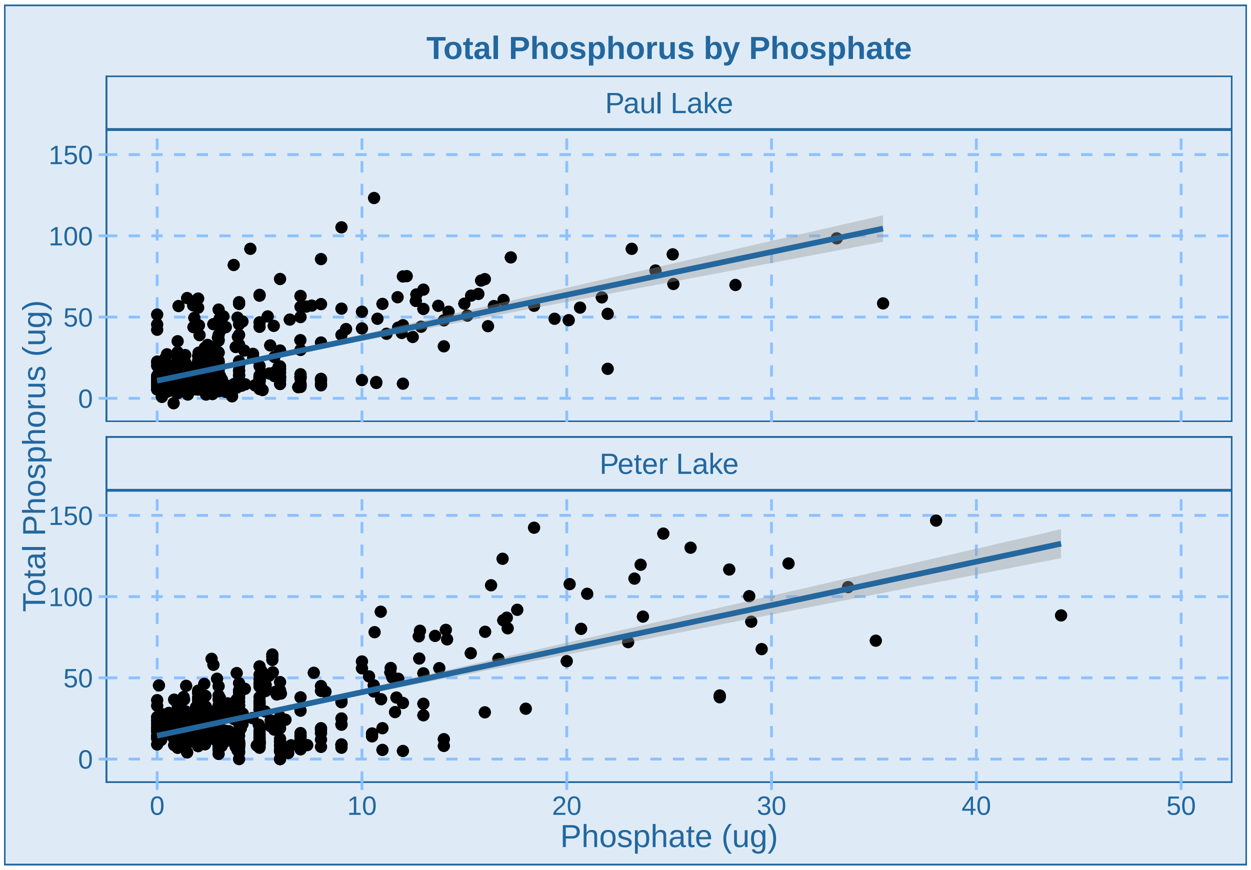
4. [NTL-LTER] Plot total phosphorus (`tp_ug`) by phosphate (`po4`), with separate aesthetics for Peter and Paul lakes. Add line(s) of best fit using the `lm` method. Adjust your axes to hide extreme values (hint: change the limits using `xlim()` and/or `ylim()`).

```r
#4 Create scatter plot of Total Phosphorus by Phosphate
PeterPaul.phosphate <-
  ggplot(PeterPaul.chem.nutrients, aes(x = po4, y = tp_ug)) +
  geom_point() +
  labs(
    x = "Phosphate (ug)",
    y = "Total Phosphorus (ug)",
    title = "Total Phosphorus by Phosphate",
  ) +
  geom_smooth(method = "lm", color = "#23679e") +
  xlim(0, 50) +
  facet_wrap(vars(lakename), nrow = 2)
print(PeterPaul.phosphate)
```

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

```
## Warning: Removed 21947 rows containing non-finite values (`stat_smooth()`).
```

```
## Warning: Removed 21947 rows containing missing values (`geom_point()`).
```

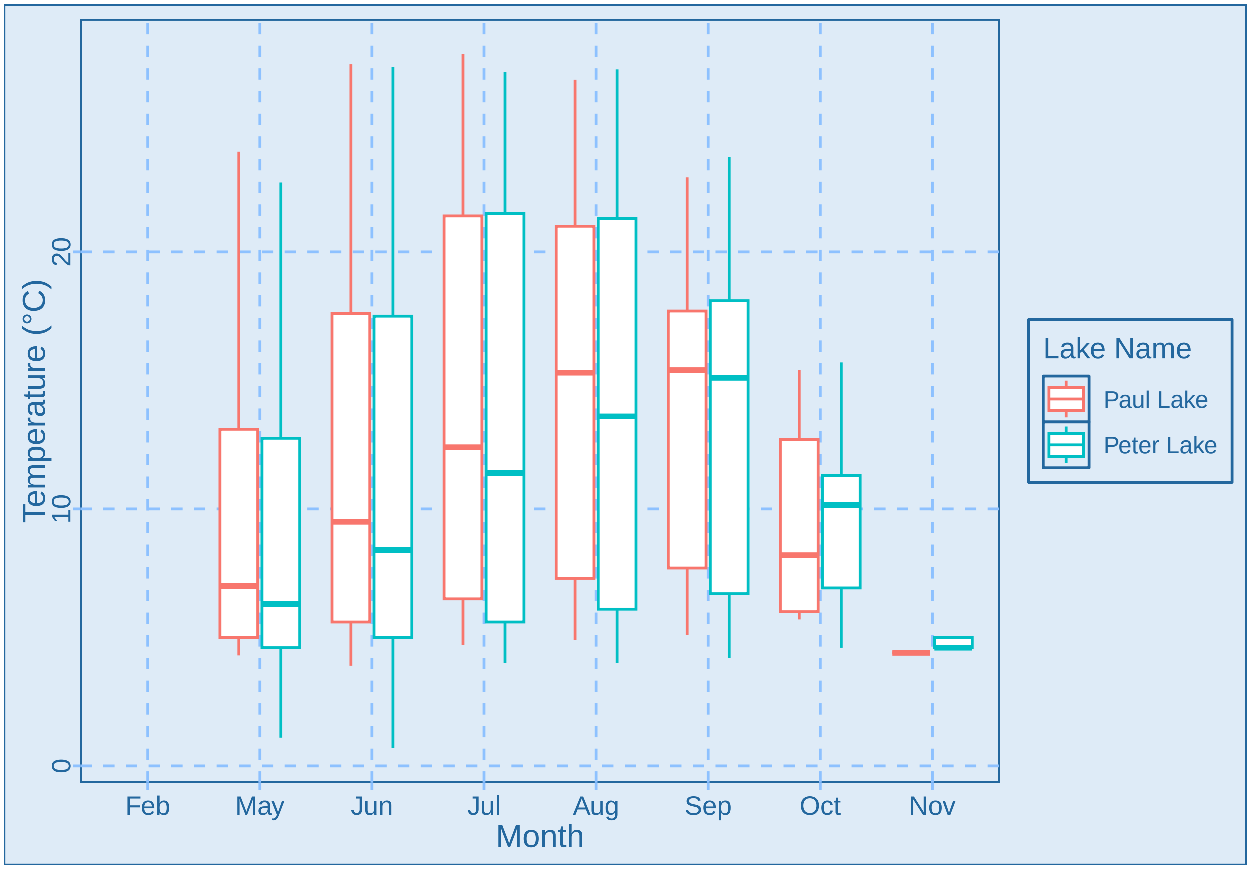


5. [NTL-LTER] Make three separate boxplots of (a) temperature, (b) TP, and (c) TN, with month as the x axis and lake as a color aesthetic. Then, create a cowplot that combines the three graphs. Make sure that only one legend is present and that graph axes are aligned.

Tips: * Recall the discussion on factors in the lab section as it may be helpful here. * Setting an axis title in your theme to `element_blank()` removes the axis title (useful when multiple, aligned plots use the same axis values) * Setting a legend's position to "none" will remove the legend from a plot. * Individual plots can have different sizes when combined using `cowplot`.

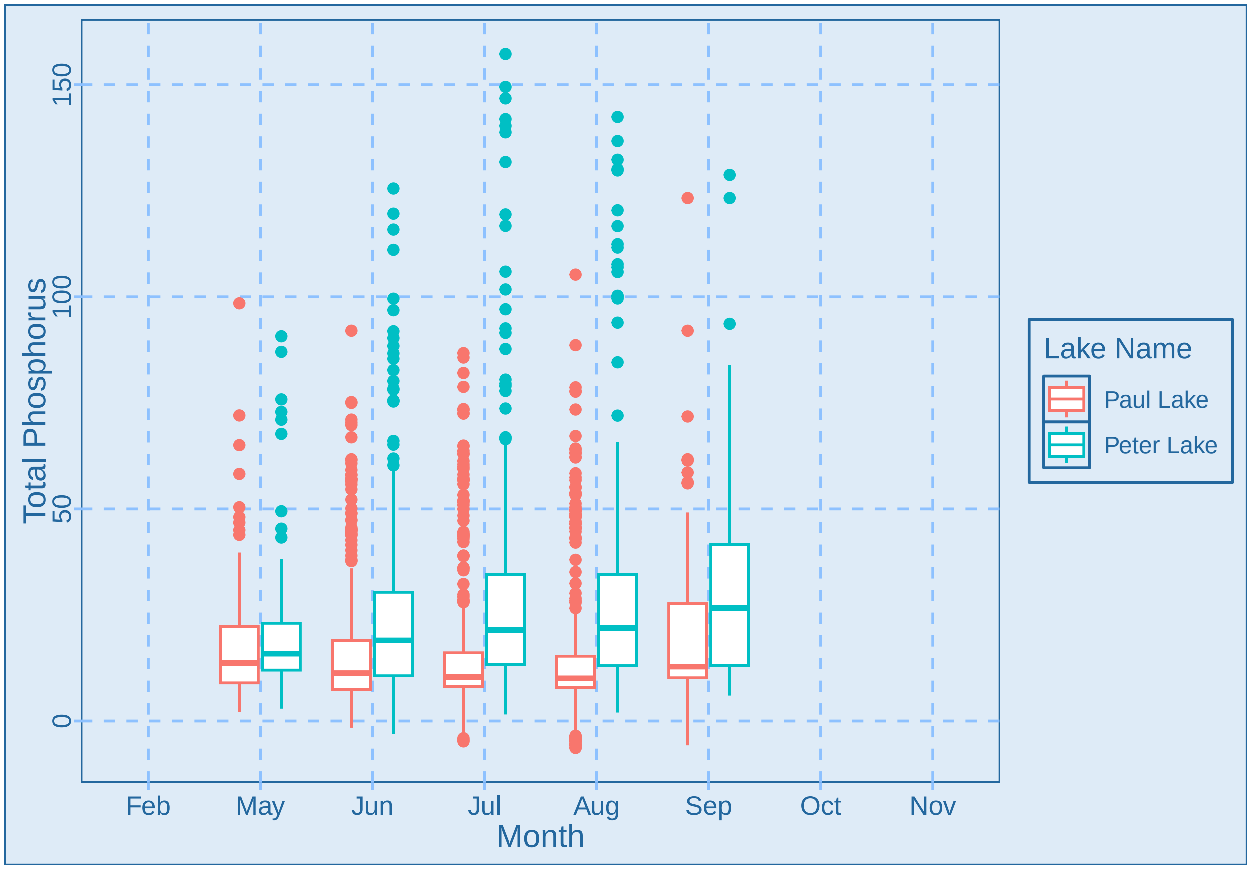
```r
#5 Create a factor variable for months with abbreviated labels
Month.factor <- factor(
  PeterPaul.chem.nutrients$month,
  levels = 1:12,
  labels = month.abb
)

# Create boxplot for temperature
PeterPaul.Temp <-
  ggplot(PeterPaul.chem.nutrients,
         aes(x = Month.factor, y = temperature_C, color = lakename)) +
  geom_boxplot() +
  labs(
    x = "Month",
    y = "Temperature (°C)",
    color = "Lake Name"
    ) +
  theme(axis.text.y = element_text(angle = 90))
PeterPaul.Temp
```

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



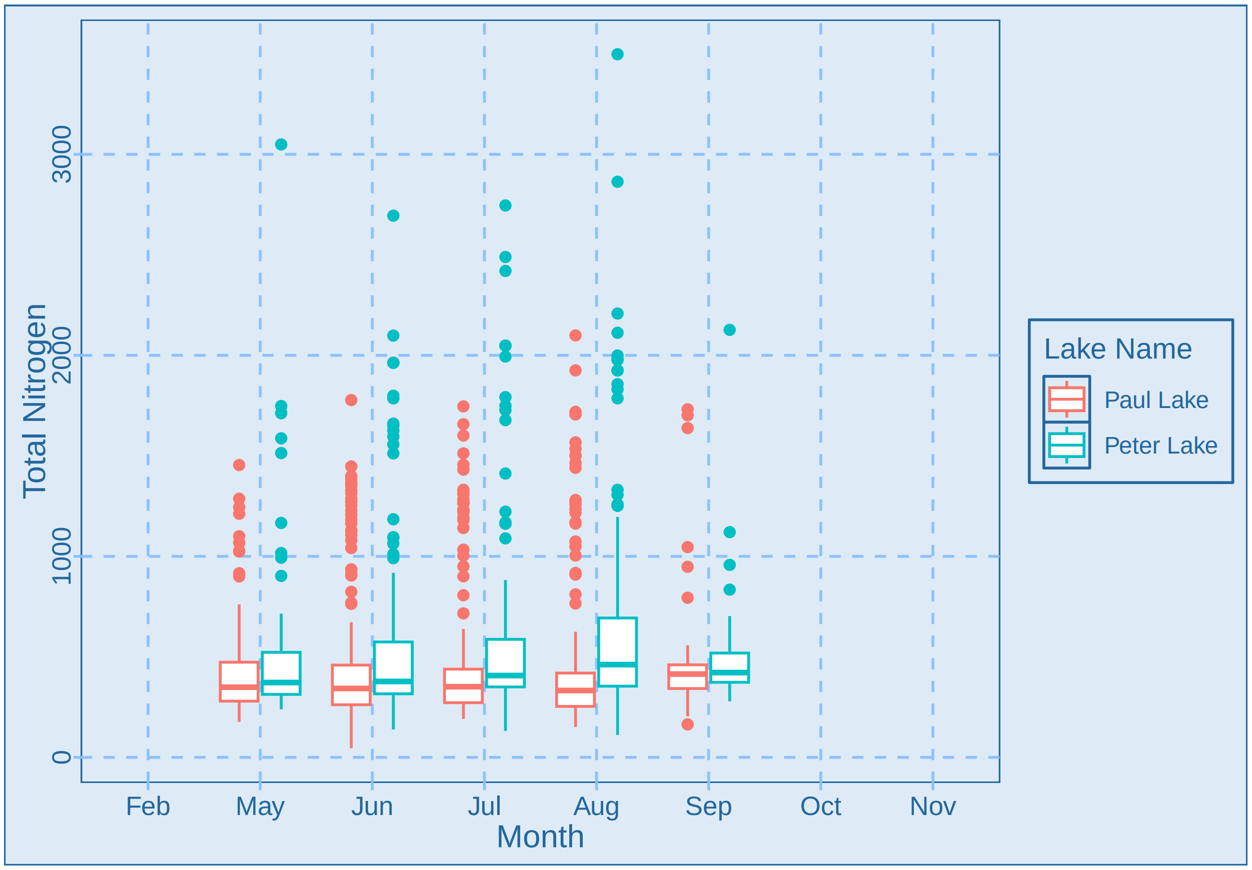
```
# Create boxplot for Total Phosphorus
PeterPaul.TP <-
  ggplot(PeterPaul.chem.nutrients,
         aes(x = Month.factor, y = tp_ug, color = lakename)) +
  geom_boxplot() +
  labs(
    x = "Month",
    y = "Total Phosphorus",
    color = "Lake Name"
    ) +
  theme(axis.text.y = element_text(angle = 90))
PeterPaul.TP
```

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



```r
# Create boxplot for Total Nitrogen
PeterPaul.TN <-
  ggplot(PeterPaul.chem.nutrients,
         aes(x = Month.factor, y = tn_ug, color = lakename)) +
  geom_boxplot() +
  labs(
    x = "Month",
    y = "Total Nitrogen",
    color = "Lake Name"
    ) +
  theme(axis.text.y = element_text(angle = 90))
PeterPaul.TN
```

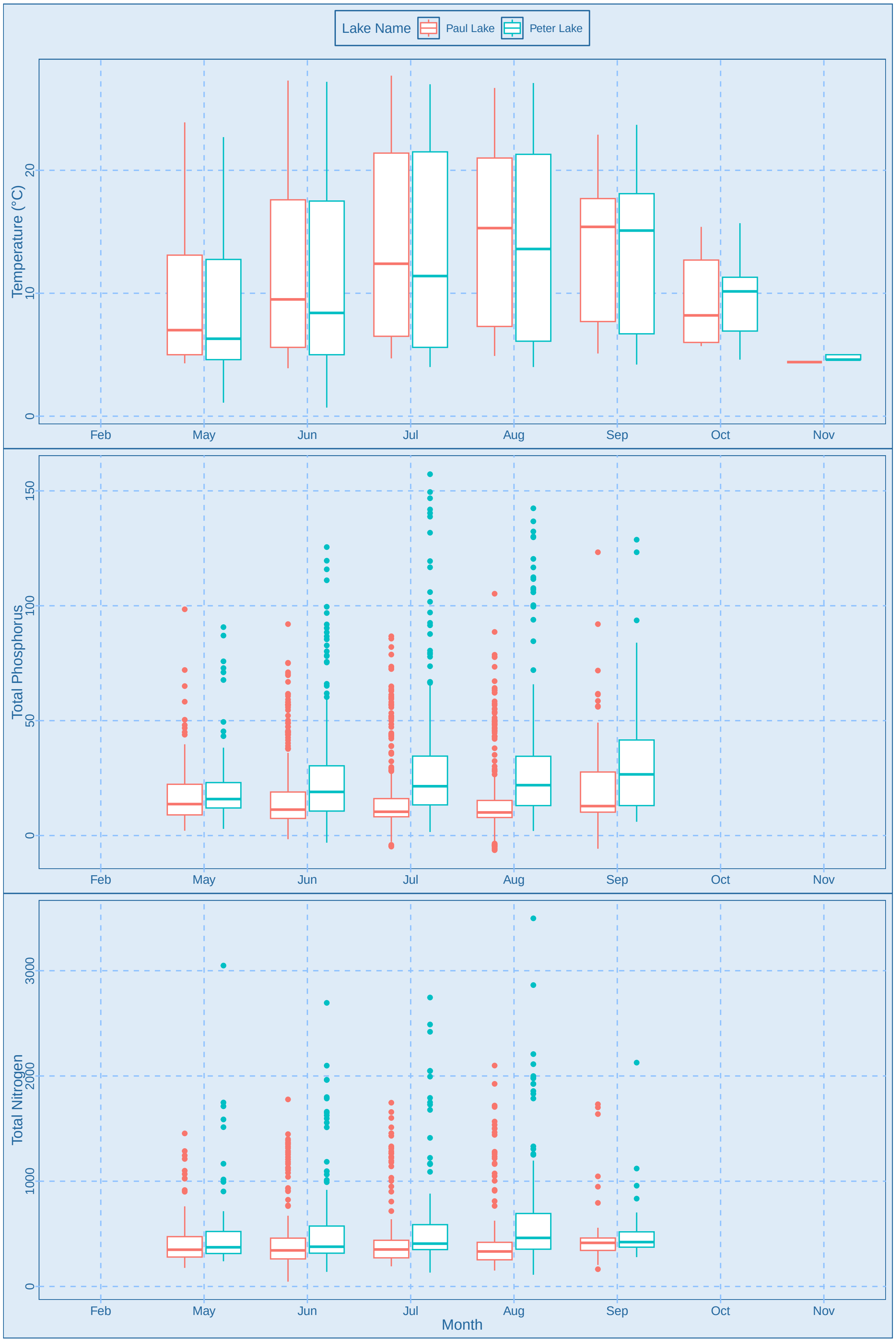
```
## Warning: Removed 21583 rows containing non-finite values (`stat_boxplot()`).
```



```
# Combine three plots, keep only the top legend, align vertically
combined.boxplot <- plot_grid(
  PeterPaul.Temp + theme(legend.position = "top", axis.title.x = element_blank()),
  PeterPaul.TP + theme(legend.position = "none", axis.title.x = element_blank()),
  PeterPaul.TN + theme(legend.position = "none"),
  nrow = 3, ncol = 1, align = "v", aix = "tb"
  )
```

```
## 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()`).

## Warning in as_grob.default(plot): Cannot convert object of class character into
## a grob.

## Warning: Graphs cannot be vertically aligned unless the axis parameter is set.
## Placing graphs unaligned.
```

```
combined.boxplot
```



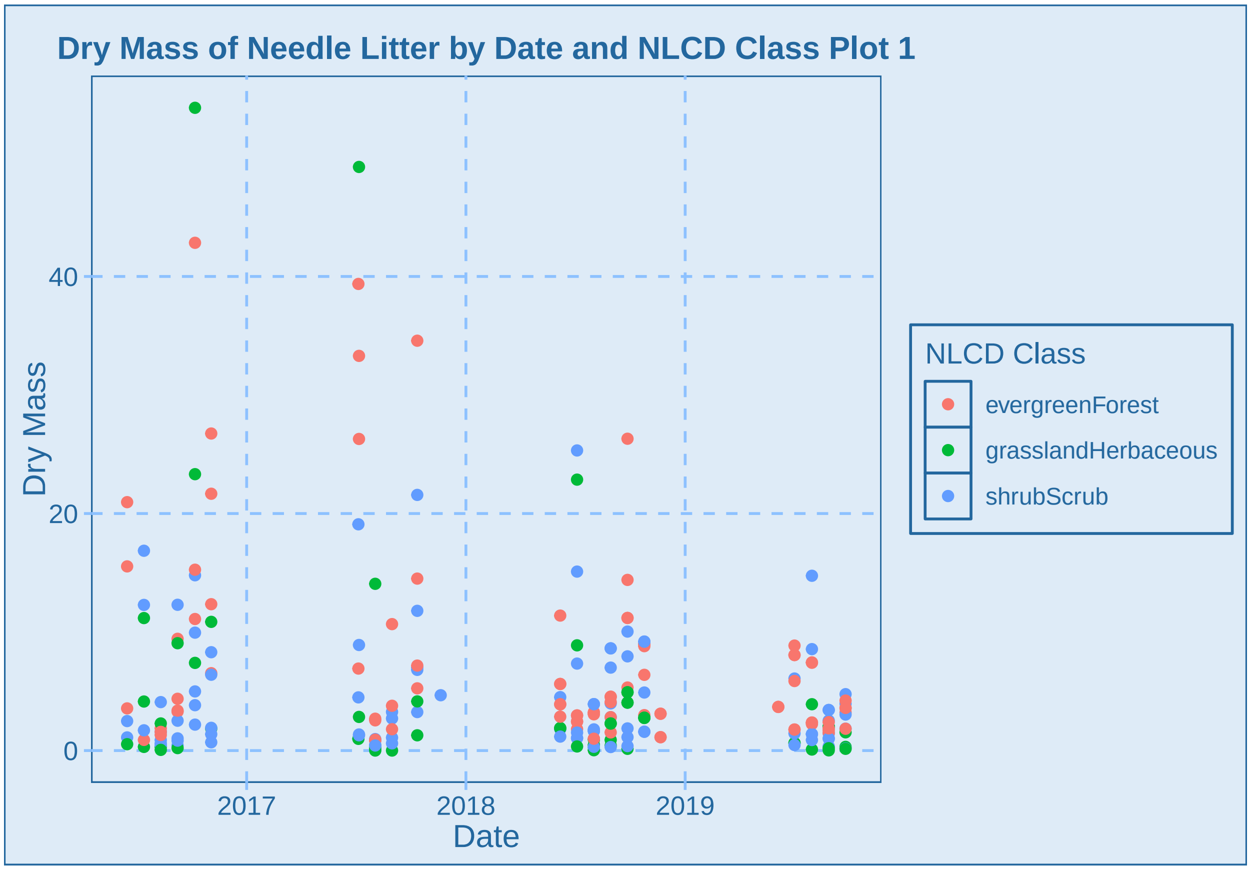
Lake Name
Paul Lake
Peter Lake
Temperature (°C)
Total Phosphorus
Total Nitrogen
0
10
20
50
100
150
1000
2000
3000
Feb
May
Jun
Jul
Aug
Sep
Oct
Nov
Month

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

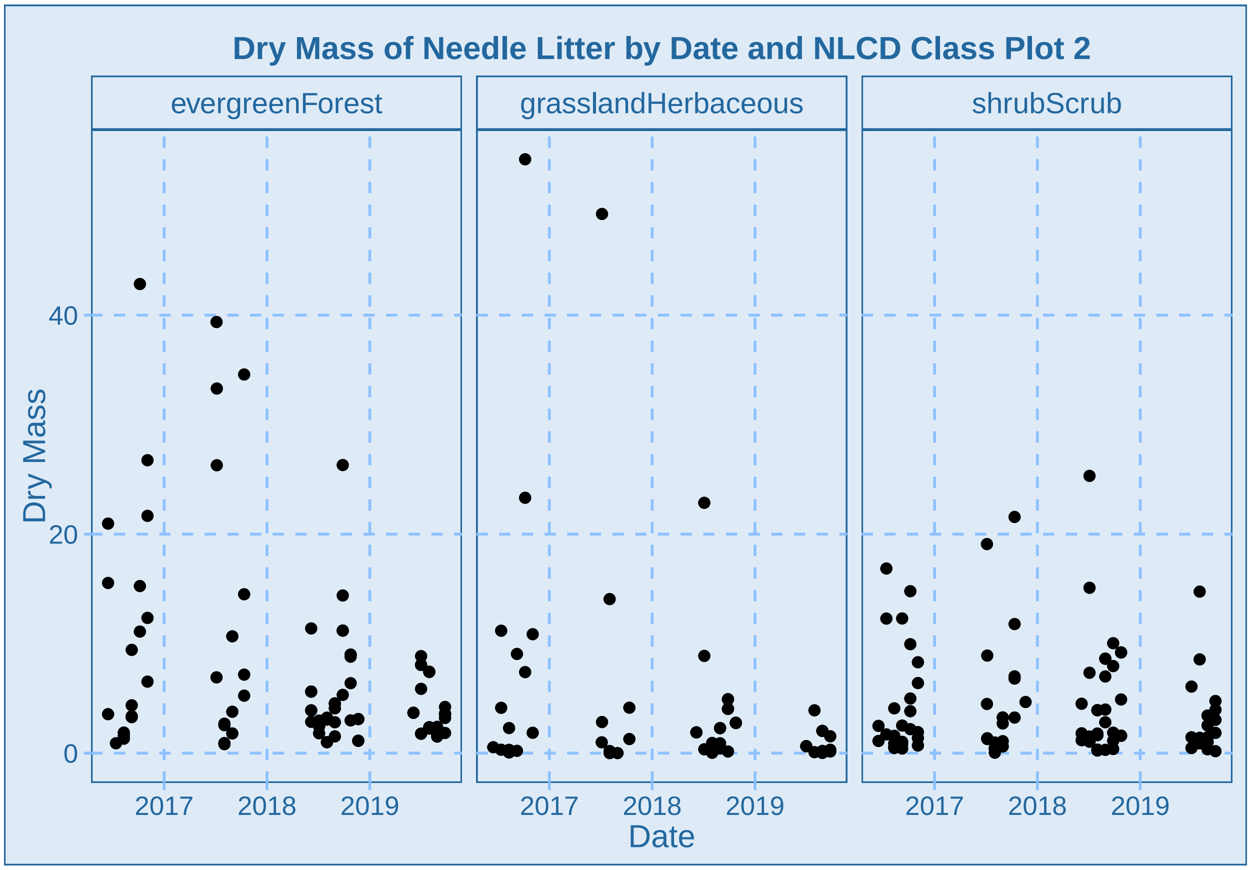
> Answer: Both lakes demonstrate relatively lower temperature during May, June, October, and November, typically averaging below 10 degrees Celcius. Both lakes have relatively higher temperature in July, August, and September, with August being the warmest month. The levels of total phosphorous and total nitrogen vary in accordance with temperature fluctuations; when temperature are higher, these levels tend to increase as well. The total phosphorus and total nitrogen levels in Peter Lake tend to be higher than those in 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 Filter data for only "Needles" functional group and plot
Niwot.Needles.Data <- subset(Niwot.litter, functionalGroup == "Needles")
Niwot.Needles <-
  ggplot(Niwot.Needles.Data,
         aes(x = collectDate, y = dryMass, color = nlcdClass)) +
  geom_point() +
  labs (title = "Dry Mass of Needle Litter by Date and NLCD Class Plot 1",
        x = "Date",
        y = "Dry Mass",
        color = "NLCD Class")
Niwot.Needles
```



```r
#7 Separate NLCD Classes into three facets
Niwot.Needles.Facet <-
  ggplot(Niwot.Needles.Data,
         aes(x = collectDate, y = dryMass)) +
  geom_point() +
  labs (title = "Dry Mass of Needle Litter by Date and NLCD Class Plot 2",
        x = "Date",
        y = "Dry Mass") +
  #Separate NLCD classes into facets
  facet_wrap( ~nlcdClass)
Niwot.Needles.Facet
```



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

> Answer: I think this depends on the purpose of the study. If we want to directly compare dry mass across NLCD classes over time, plot 1 (6) would be more effective because NLCD classes are represtend by different colors on the same plot, which help identify trends and patterns in dry mass within each class. However, if we are looking for a more detailed examination of dry mass within each NLCD class, plot 2 (7) could provide a clearer view. In general, I would say plot 1 (6) is more effective.