# Assignment 5: Data Visualization

Fiona Price

Fall 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 in required packages and verify home directory.
library(tidyverse)
```

```
## -- 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   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## 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
```

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

```
## here() starts at /home/guest/ede_fall2024
```

```r
library(cowplot)
```

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

```r
getwd()
```

```
## [1] "/home/guest/ede_fall2024"
```

```r
here()
```

```
## [1] "/home/guest/ede_fall2024"
```

```r
#Read in the NTL-LTER processed data files for nutrients and chemistry/physics
#for Peter and Paul Lakes

PeterPaul_processed <- read.csv(
  here("Data/Processed/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv"),
  stringsAsFactors = TRUE)

NeonNiwo_processed <- read.csv(
  here("Data/Processed/NEON_NIWO_Litter_mass_trap_Processed.csv"),
  stringsAsFactors = TRUE)


#2 Make syre R is reading the dates as date format.
#Start with PeterPaul data.
class(PeterPaul_processed$sampledate)
```

```
## [1] "factor"
```

```r
#Reading it as a factor; change to data
PeterPaul_processed$sampledate <- as.Date(PeterPaul_processed$sampledate,
                                          format = "%Y-%m-%d")
class(PeterPaul_processed$collectDate)
```

```
## [1] "NULL"
```

```
#Now for Neonics data.
class(NeonNiwo_processed$collectDate)
```

```
## [1] "factor"
```

```
#Reading it as a factor; change to data
NeonNiwo_processed$collectDate <- as.Date(NeonNiwo_processed$collectDate,
                                          format = "%Y-%m-%d")
class(NeonNiwo_processed$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. Create my own theme
library(ggthemes)
```

```
##
## Attaching package: 'ggthemes'
```

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

```
library(hrbrthemes)
library(viridis)
```

```
## Loading required package: viridisLite
```

```
library(RColorBrewer)
my_theme <- theme_bw() +
  theme(
    legend.position = "right",
    legend.text = element_text(size = 10),
    legend.title = element_text(size = 10),
    legend.direction = "vertical",
    plot.title = element_text(color = "#008B8B"),
    axis.text.x = element_text(angle = 45, hjust = 1, size = 8)
  )
```

### 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.

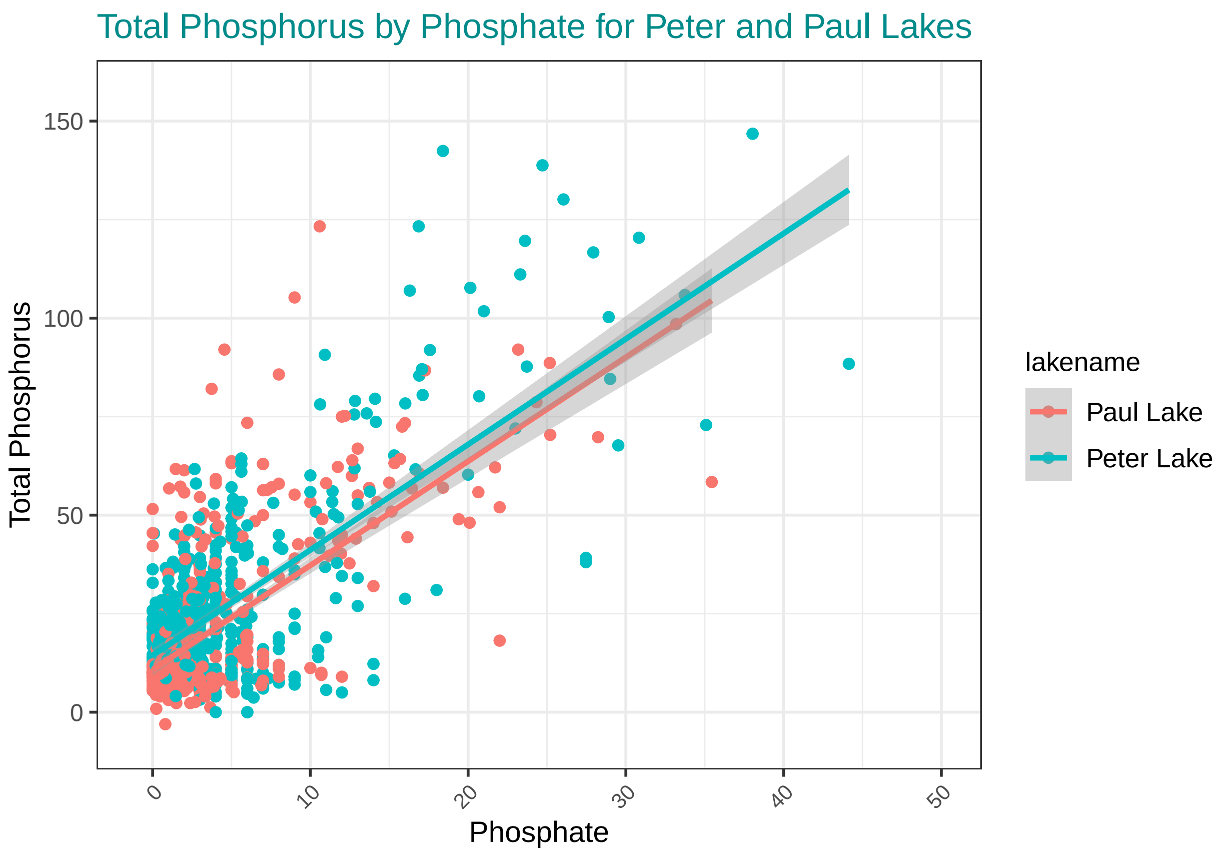
4. [NTL-LTER] Plot total phosphorus (`tp_ug`) by phosphate (`po4`), with separate aesthetics for Peter and Paul lakes. Add line(s) of best fit using the `lm` method. Adjust your axes to hide extreme values (hint: change the limits using `xlim()` and/or `ylim()`).

```
#4. Plot total phosphorus by phosphate with separate aesthetics for Peter and
#Paul lakes.

P_vs_P04 <-
  ggplot(PeterPaul_processed, aes(x = po4, y = tp_ug, color = lakename)) +
  geom_point() + #indicate scatter plot
  geom_smooth(method = lm) + #add line of best fit
  my_theme + #add my theme created in step 3
  xlim(-1, 50) + #hide extreme values
  labs(title = "Total Phosphorus by Phosphate for Peter and Paul Lakes",
       x = "Phosphate",
       y = "Total Phosphorus") #change titles

show(P_vs_P04)
```

```
## ‘geom_smooth()‘ using formula = ’y ~ x’
```

```
## Warning: Removed 21947 rows containing non-finite outside the scale range
## (‘stat_smooth()‘).
```

```
## Warning: Removed 21947 rows containing missing values or values outside the scale range
## (‘geom_point()‘).
```



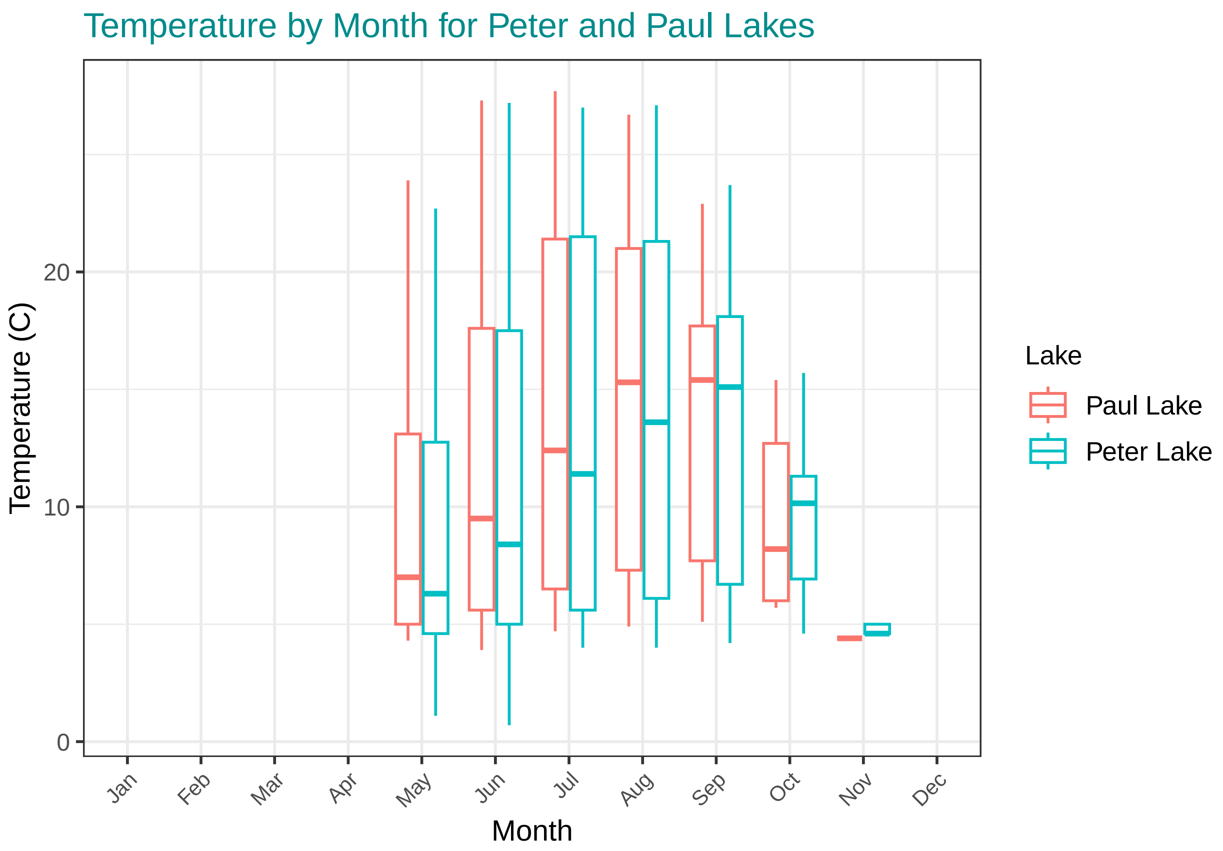
5. [NTL-LTER] Make three separate boxplots of (a) temperature, (b) TP, and (c) TN, with month as the x axis and lake as a color aesthetic. Then, create a cowplot that combines the three graphs. Make sure that only one legend is present and that graph axes are aligned.

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 boxplot of temperature with month as the x-axis and lake as a
#color aesthetic.
#Convert month to a factor -- with 12 levels, labelled with month names
PeterPaul_processed$month <- factor(PeterPaul_processed$month,
       levels = 1:12,
       labels = month.abb)

temp_box <- ggplot(PeterPaul_processed, aes(x = month, y = temperature_C,
                                             color = lakename)) +
  geom_boxplot() + #specify box plot
  scale_x_discrete(drop = F) + #include all months
  my_theme + #add my theme
  labs(title = "Temperature by Month for Peter and Paul Lakes",
       x = "Month",
       y = "Temperature (C)",
       color = "Lake") #add titles
```

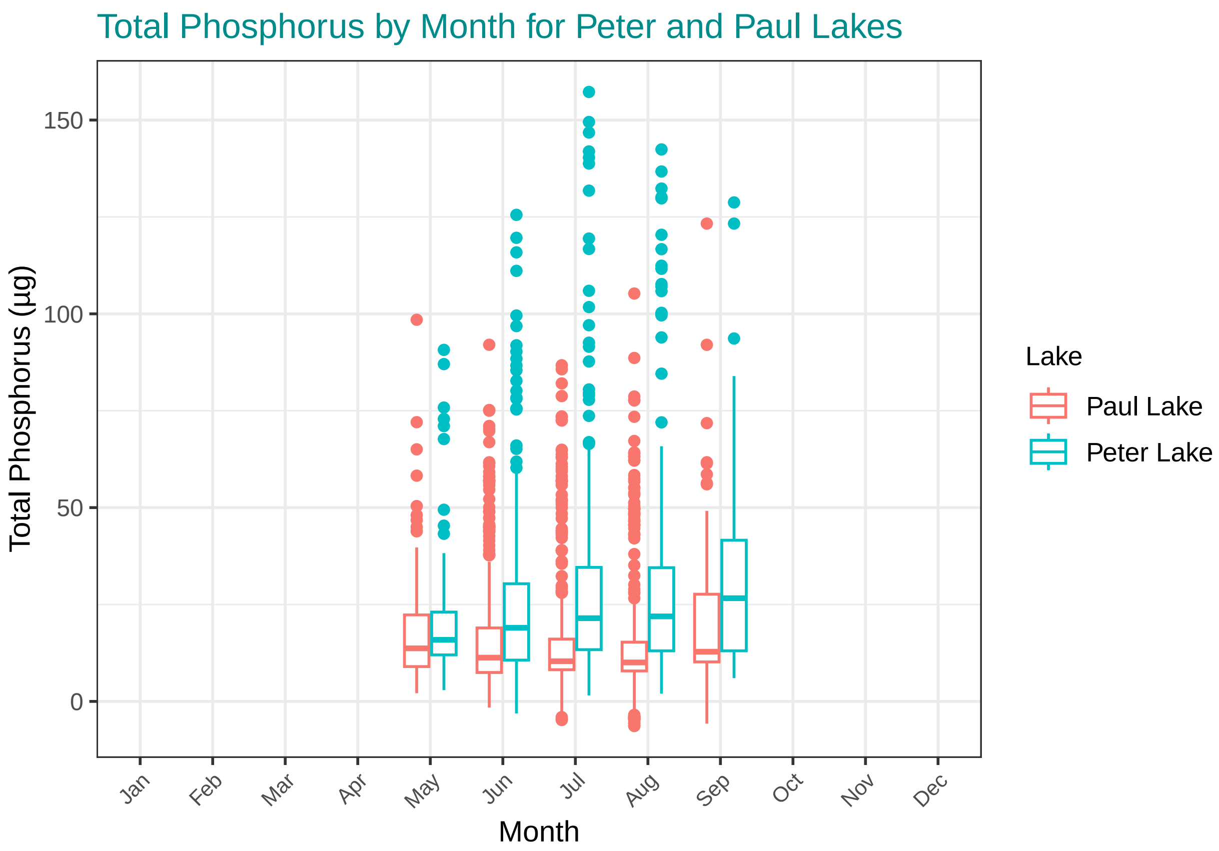
```
show(temp_box)
```

```
## Warning: Removed 3566 rows containing non-finite outside the scale range
## (‘stat_boxplot()‘).
```



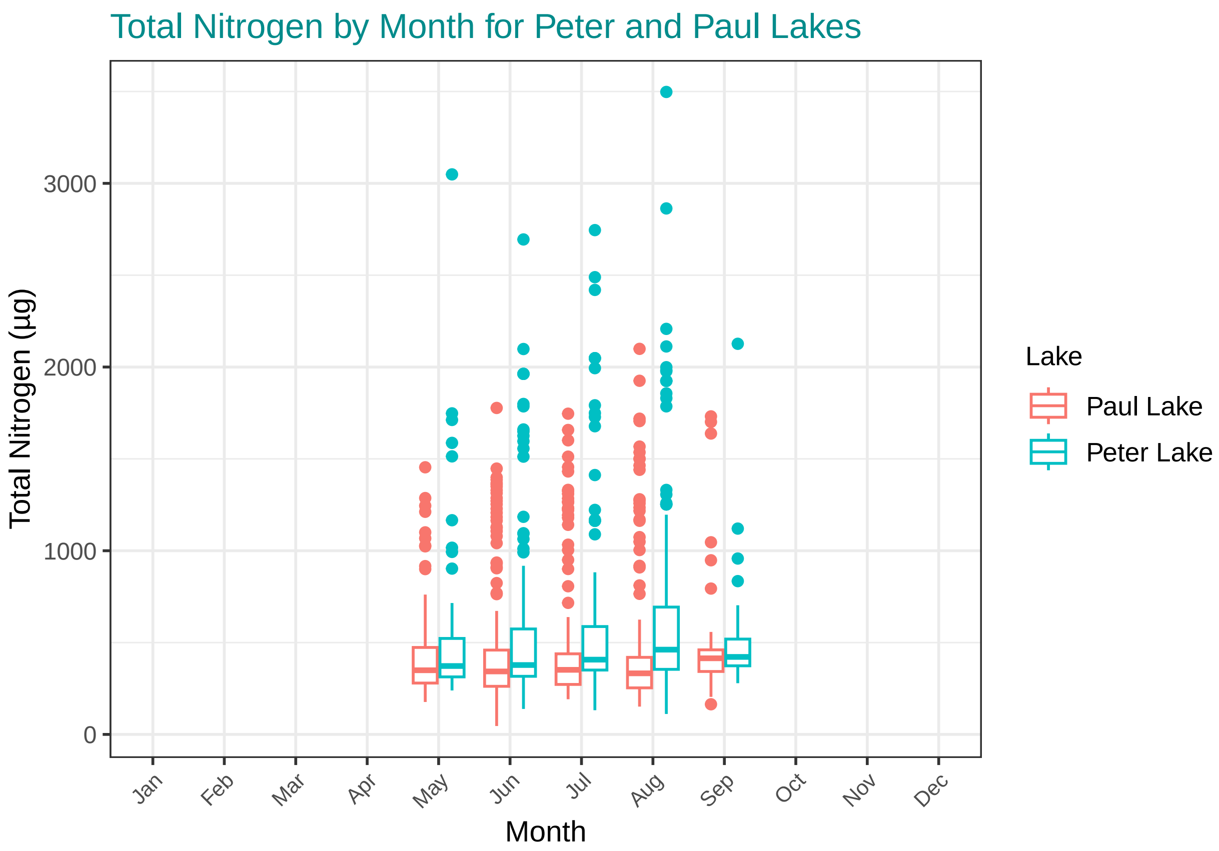
```
#Create a boxplot of TP with month as the x-axis and lake as a
#color aesthetic.
TP_box <- ggplot(PeterPaul_processed, aes(x = month, y = tp_ug,
                                         color = lakename)) +
  geom_boxplot() + #specify box plot
  scale_x_discrete(drop = F) + #include all months
  my_theme + #add my theme
  labs(title = "Total Phosphorus by Month for Peter and Paul Lakes",
       x = "Month",
       y = "Total Phosphorus (µg)",
       color = "Lake") #add titles

show(TP_box)
```

```
## Warning: Removed 20729 rows containing non-finite outside the scale range
## (‘stat_boxplot()‘).
```



```
#Create a boxplot of TN with month as the x-axis and lake as a
#color aesthetic.
TN_box <- ggplot(PeterPaul_processed, aes(x = month, y = tn_ug,
                                          color = lakename)) +
  geom_boxplot() + #specify boxplot
  scale_x_discrete(drop = F) + #include all months
  my_theme + #add my theme
  labs(title = "Total Nitrogen by Month for Peter and Paul Lakes",
       x = "Month",
       y = "Total Nitrogen (µg)",
       color = "Lake") #add titles
show(TN_box)
```

```
## Warning: Removed 21583 rows containing non-finite outside the scale range
## (‘stat_boxplot()‘).
```



```
#Create a cowplot

#Create a temperature box plot without a title, x-axis, or legend. Change the
#size of the labels to fit in a cowplot.
temp_box_clean <- temp_box +
  labs(title = NULL, x = NULL) +
  theme(legend.position = "none") +
  theme(plot.margin = unit(c(1, 0, 1, 0), "cm")) +
  theme(axis.text.y = element_text(size = 6),
        axis.text.x = element_text(size = 6),
        axis.title.y = element_text(size = 8))

#Create a TP box plot without a title, x-axis, or legend. Change the
#size of the labels to fit in a cowplot.
TP_box_clean <- TP_box +
  labs(title = NULL, x = NULL) +
  theme(legend.position = "none") +
  theme(plot.margin = unit(c(1, 0, 1, 0), "cm")) +
  theme(axis.text.y = element_text(size = 6),
        axis.text.x = element_text(size = 6),
        axis.title.y = element_text(size = 8))

#Create a TN box plot without a title, x-axis, or legend. Change the
#size of the labels to fit in a cowplot.
TN_box_clean <- TN_box +
  labs(title = NULL, x = NULL) +
  theme(legend.position = "none") +
  theme(plot.margin = unit(c(1, 0, 1, 0), "cm")) +
  theme(axis.text.y = element_text(size = 6),
        axis.text.x = element_text(size = 6),
        axis.title.y = element_text(size = 8))

#Create a legend for the cowplot and adjust sizes.
legend <- get_legend(TN_box_clean + theme(legend.position = "right",
                                          legend.title = element_text(size = 8),
                                          legend.text = element_text(size = 6),))
```

```
## Warning: Removed 21583 rows containing non-finite outside the scale range
## (`stat_boxplot()`).
```

```
## Warning in get_plot_component(plot, "guide-box"): Multiple components found;
## returning the first one. To return all, use `return_all = TRUE`.
```

```
#Create a cowplot.
library(cowplot)
combined_plot <- plot_grid(temp_box_clean,
          TP_box_clean, TN_box_clean, legend,
          nrow = 1, align = 'h', rel_widths = c(1,1,1,0.5)) +
  draw_label("Temperature, TP, and TN by Month for Peter and Paul Lakes",
             size = 12,
             x = 0.5, y = 0.95, hjust = 0.5, color = "#008B8B") +
  draw_label ("Month",
              size = 10,
              x = 0.5, y = 0.05, hjust = 0.5)
```

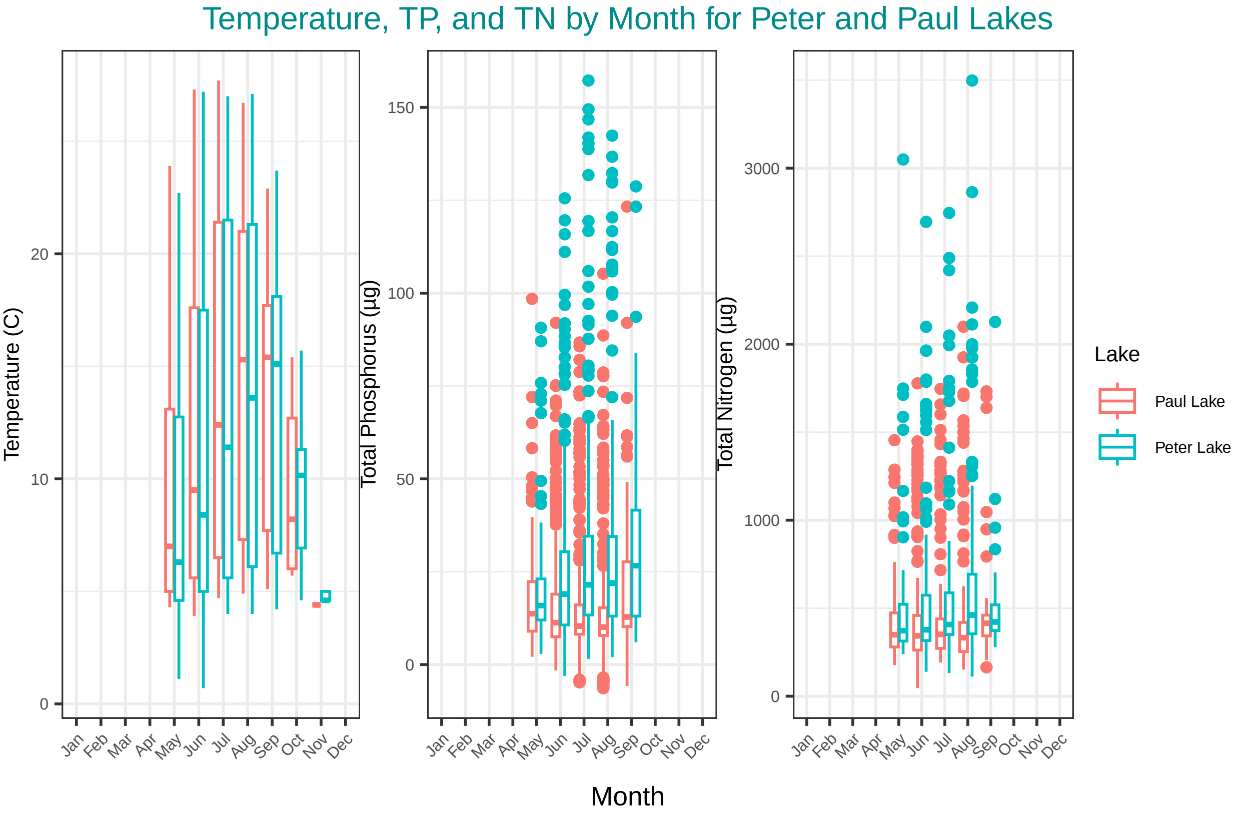
```
## Warning: Removed 3566 rows containing non-finite outside the scale range
## (`stat_boxplot()`).
```

```
## Warning: Removed 20729 rows containing non-finite outside the scale range
## (`stat_boxplot()`).
```

```
## Warning: Removed 21583 rows containing non-finite outside the scale range
## (`stat_boxplot()`).
```

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

```
show(combined_plot)
```

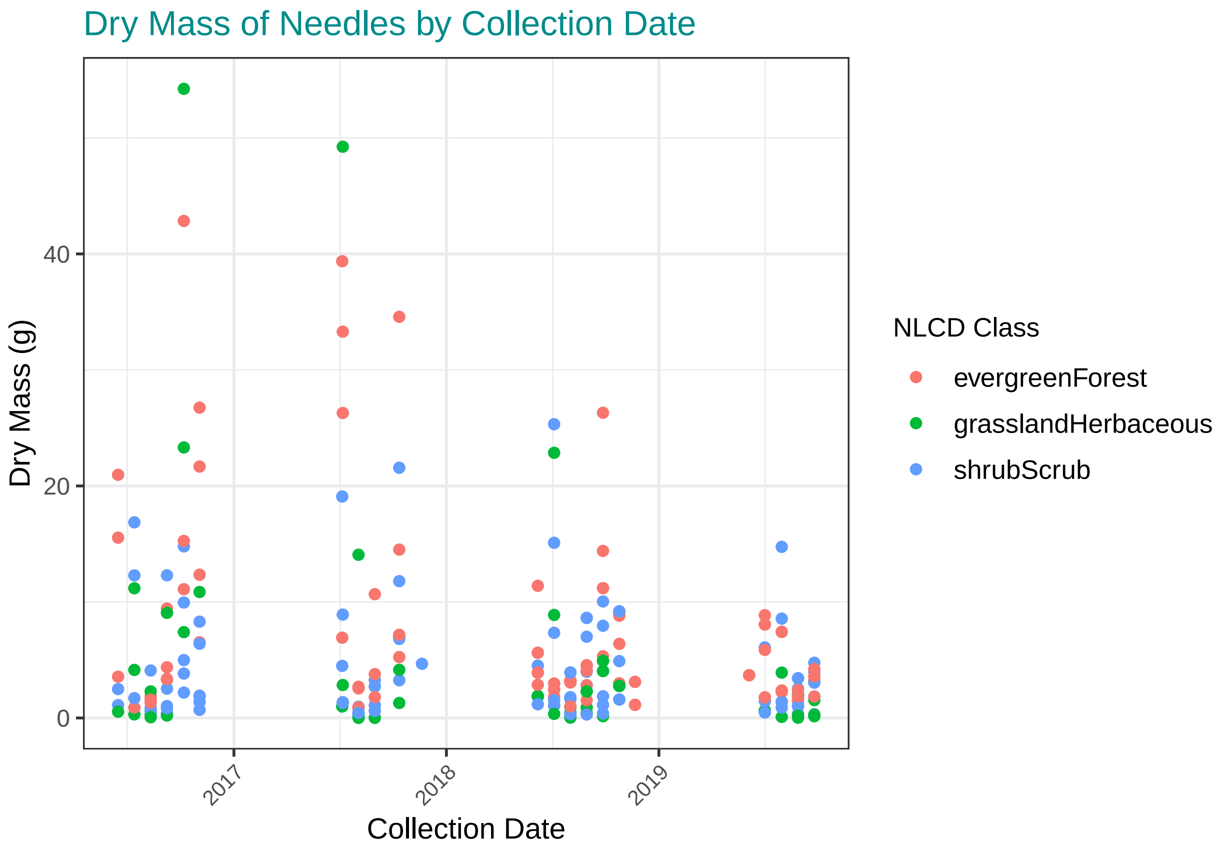


Question: What do you observe about the variables of interest over seasons and between lakes?

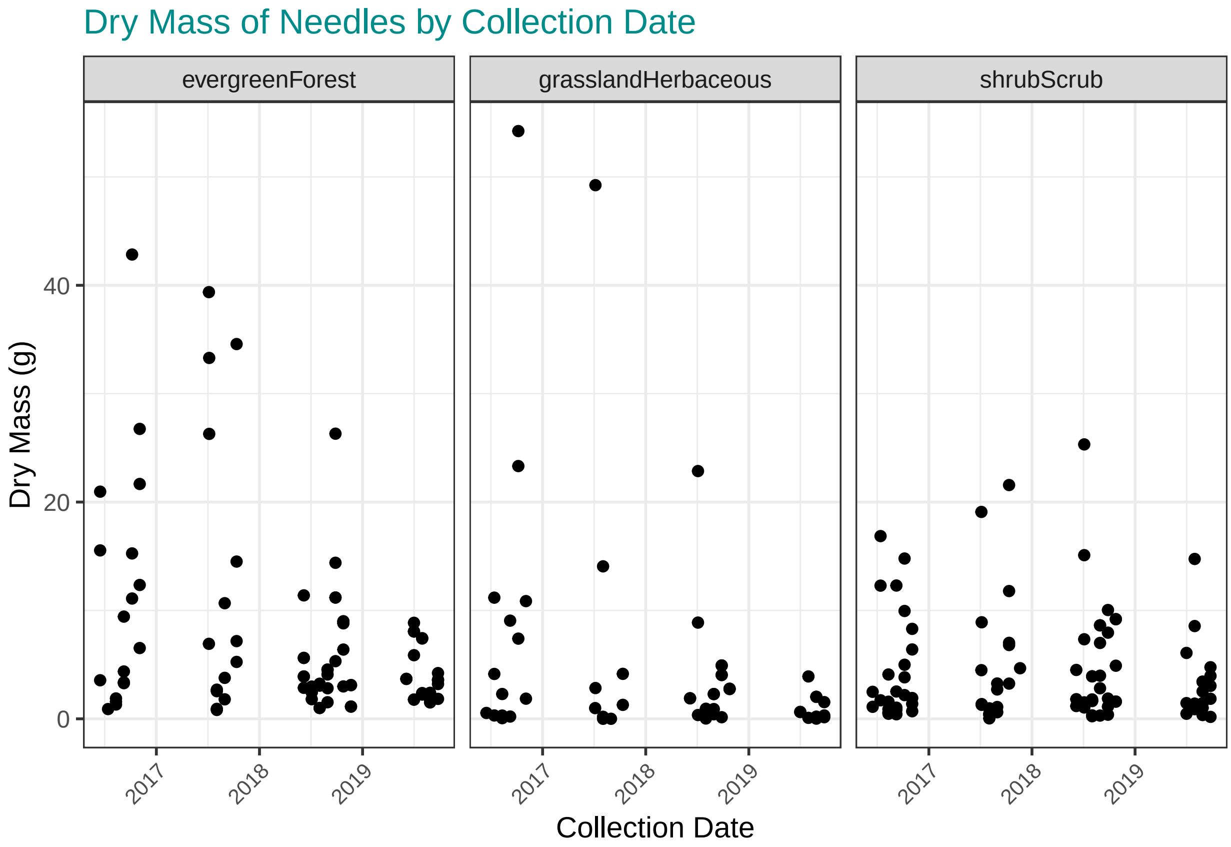
> Answer: Total phosphorus and total nitrogen increase in the late spring and hit their peak in late summer (July and August). Phosphorus and nitrogen data is not present past early fall. Temperature data is provided for late spring through early winter, but temperatures still peak in the summer. Phosophorus and nitrogen are higher for Peter lake than Paul lake. Tmperatures are faily consistent between the two.

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. 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.

ggplot(subset(NeonNiwo_processed,
              functionalGroup == "Needles")) + #subset Needles
  geom_point(aes(x = collectDate, y = dryMass, color = nlcdClass)) + #specify
  #scatter plot, data, and separate by NLCD class
  labs(title = "Dry Mass of Needles by Collection Date",
       x = "Collection Date",
       y = "Dry Mass (g)",
       color = "NLCD Class") + #add titles
  my_theme #add my theme
```



```
#7. Now, plot the same plot but with NLCD classes separated into three facets rather than separated by
ggplot(subset(NeonNiwo_processed,
              functionalGroup == "Needles")) + #subse Needles
  geom_point(aes(x = collectDate, y = dryMass)) + #specify
  #scatter plot and data used
  facet_wrap(vars(nlcdClass), nrow = 1) + #separate NLCD classes into 3
  #different facets
  labs(title = "Dry Mass of Needles by Collection Date",
       x = "Collection Date",
       y = "Dry Mass (g)") + #add titles
  my_theme #add my theme
```



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

Answer: I think plot 7 is more effective, because it's easier for me to see the data for each site. In plot 6, many of the data points are clustered together, making it difficult to identify which collection belongs to which site. While 7 has the drawback of having to look at 3 different graphs rather than just 1, I think it is ultimately easier to distinguish the data points and see each individual collection event. Additionally, it is easier for colorblind people (although color palettes can be selected to ensure ADA compliance).