Assignment 5: Data Visualization

Christopher Starr

Spring 2025

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

#loading in libraries my code will use
library(tidyverse);library(lubridate);library(here)
```

```
## -- 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
## here() starts at /home/guest/New Git Spring 2025
```

```
library(ggridges); library(cowplot)
```

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

```
#checking where my code is looking around in the directory
here()
```

```
## [1] "/home/guest/New Git Spring 2025"
```

```
#Reading in the datasets I will be using
LakeData <-
read.csv(here("Data/Processed_KEY/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv"), stringsAs

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

#2

#checking date formats
class(LakeData$sampledate)
```

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

```
class(ForestLitterData$collectDate)
```

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

```
#dates are showing as factors, using lubridate to make them date format
LakeData$sampledate <- ymd(LakeData$sampledate)
ForestLitterData$collectDate <- ymd(ForestLitterData$collectDate)

#checking date formats
class(LakeData$sampledate)
```

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

```
class(ForestLitterData$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 custom theme with a Minnesota Vikings color scheme - purple and gold elements
mytheme <- theme(
    # Plot background
    panel.background = element_rect(fill = "#686868", color = "#9370DB"),
    plot.background = element_rect(fill = "#787878"),

    # Axis text and titles - changed axis titles to black
    axis.text = element_text(color = "gold", family = "mono", size = 11),
    axis.title = element_text(color = "black", family = "mono", face = "bold", size = 12),  # Changed t

    panel.grid.major = element_line(color = "#9370DB", linetype = "dashed", linewidth = 0.2),
    panel.grid.minor = element_line(color = "#9370DB", linetype = "dotted", linewidth = 0.1),

    legend.background = element_rect(fill = "#787878"),
    legend.text = element_text(color = "gold", family = "mono"),
    legend.title = element_text(color = "#9370DB", family = "mono", face = "bold"),
    legend.position = "top"
)

# Set as default theme for future plots
theme_set(mytheme)
```

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()`).

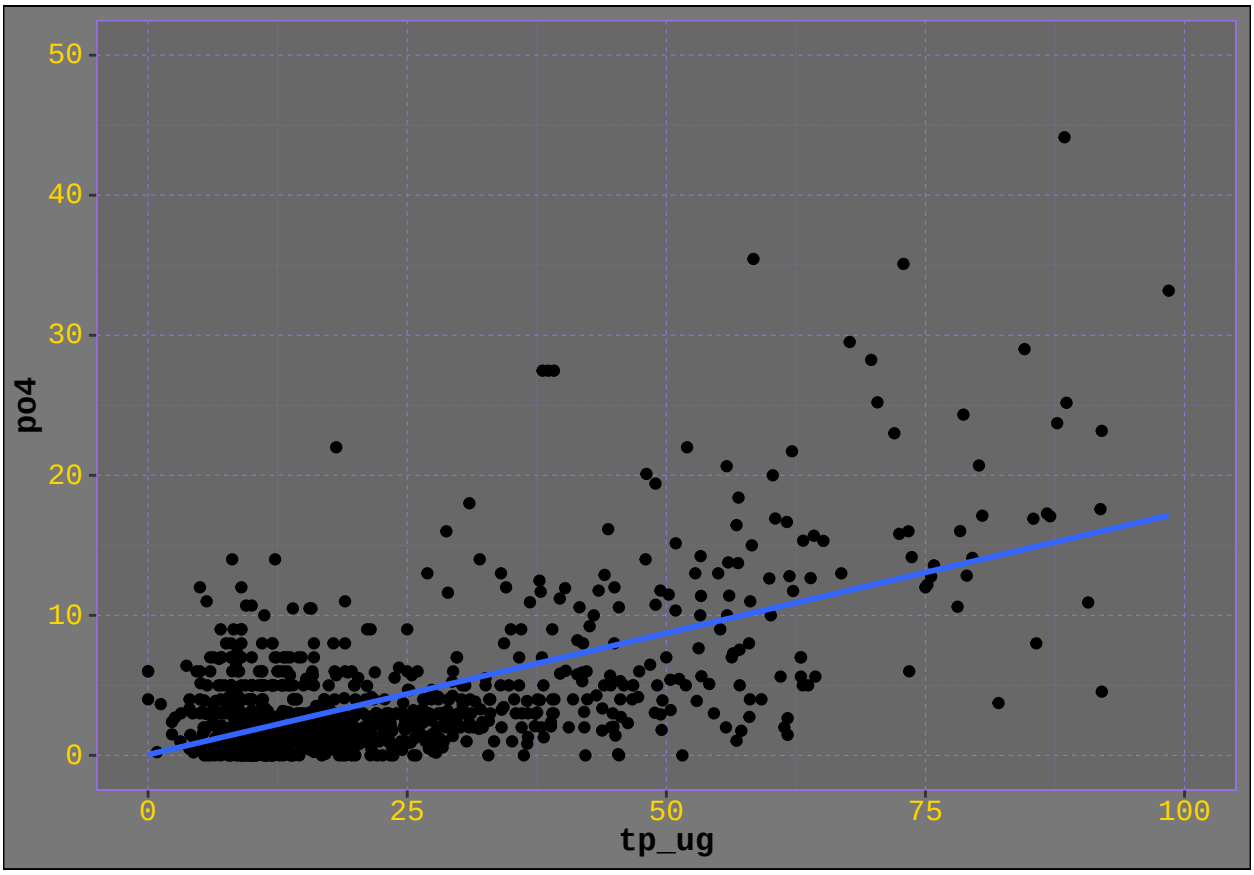
```
#4
phos_phorusbyphate <-
  ggplot(LakeData, aes(x = tp_ug, y = po4)) +
  geom_point() +
  xlim(0, 100) +
  ylim(0, 50)+
  geom_smooth(
    method = lm,
    se=FALSE
```

```
  )

#Print the plot
print(phos_phorusbyphate)
```

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

```
## Warning: Removed 21964 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

```
## Warning: Removed 21964 rows containing missing values or values outside the scale range
## (`geom_point()`).
```

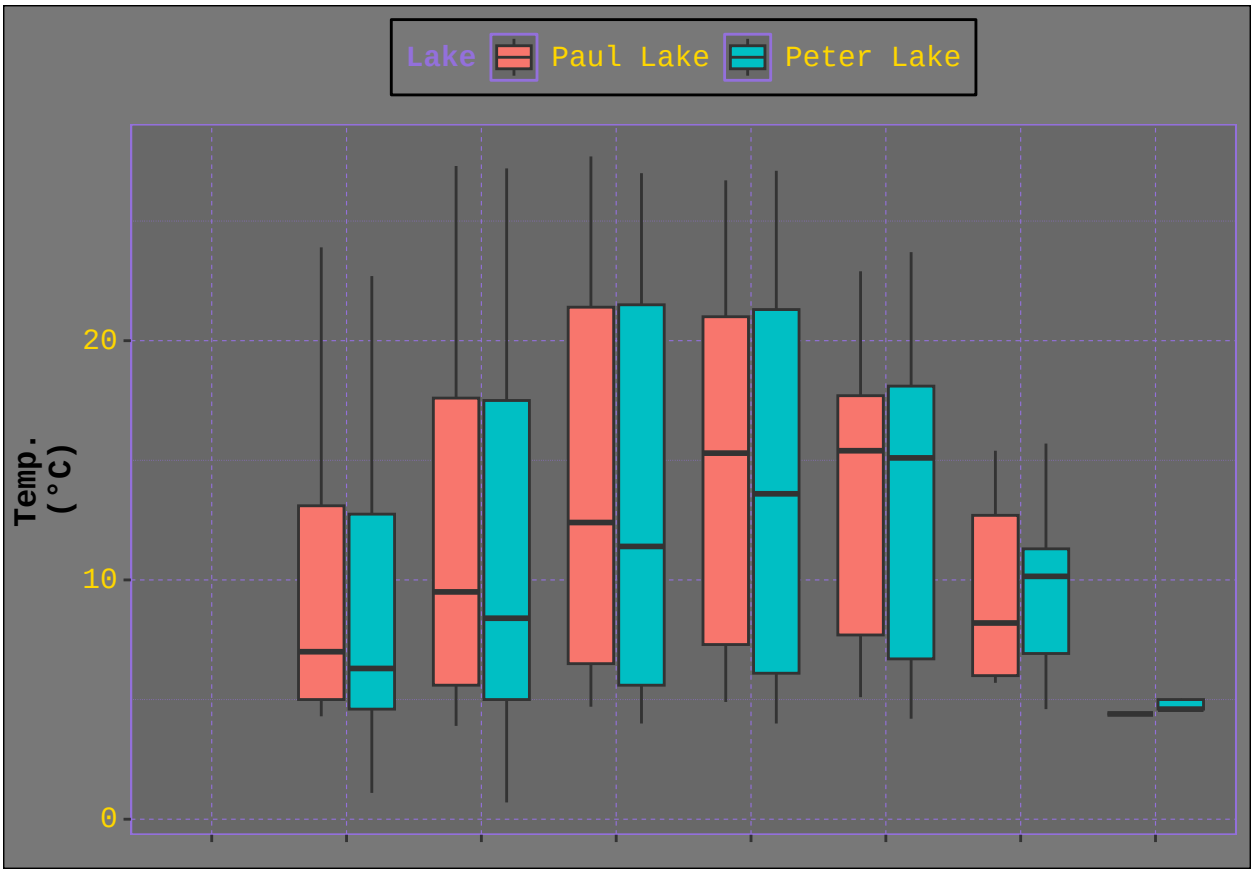


5. [NTL-LTER] Make three separate boxplots of (a) temperature, (b) TP, and (c) TN, with month as the x axis and lake as a color aesthetic. Then, create a cowplot that combines the three graphs. Make sure that only one legend is present and that graph axes are aligned.

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 factor for months in the correct order
month_labels <- c("Jan", "Feb", "Mar", "Apr", "May", "Jun",
                  "Jul", "Aug", "Sep", "Oct", "Nov", "Dec")
LakeData$month <- factor(LakeData$month,
                         levels = 1:12,
                         labels = month_labels)

# Temperature boxplot (with legend)
temp_plot <- ggplot(LakeData, aes(x = month, y = temperature_C, fill = lakename)) +
  geom_boxplot() +
  labs(x = element_blank(),
       y = "Temp.\n(°C)",
       fill = "Lake") +
  theme(axis.text.x = element_blank())

print(temp_plot)
```

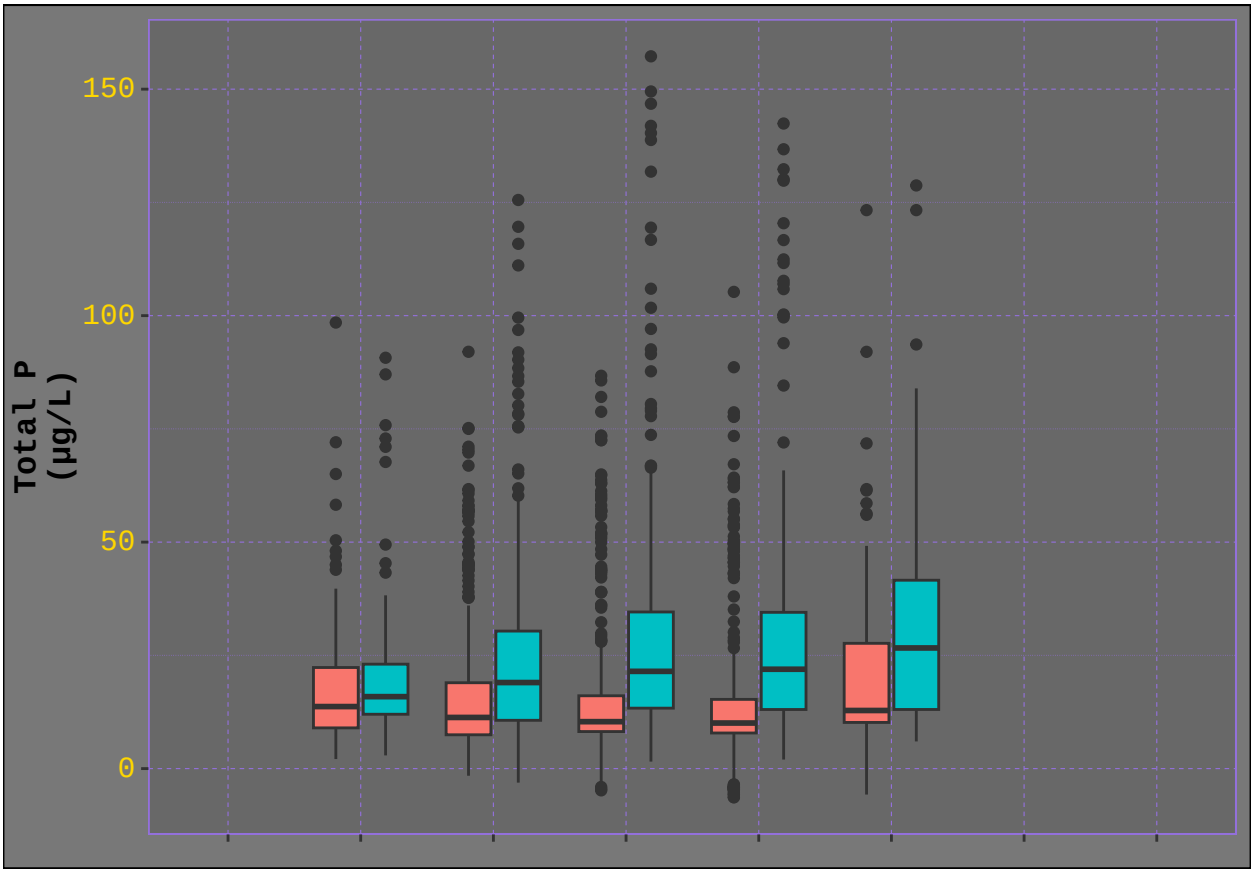
```
## Warning: Removed 3566 rows containing non-finite outside the scale range
## (`stat_boxplot()`).
```




```r
# Total Phosphorus boxplot
tp_plot <- ggplot(LakeData, aes(x = month, y = tp_ug, fill = lakename)) +
  geom_boxplot() +
```

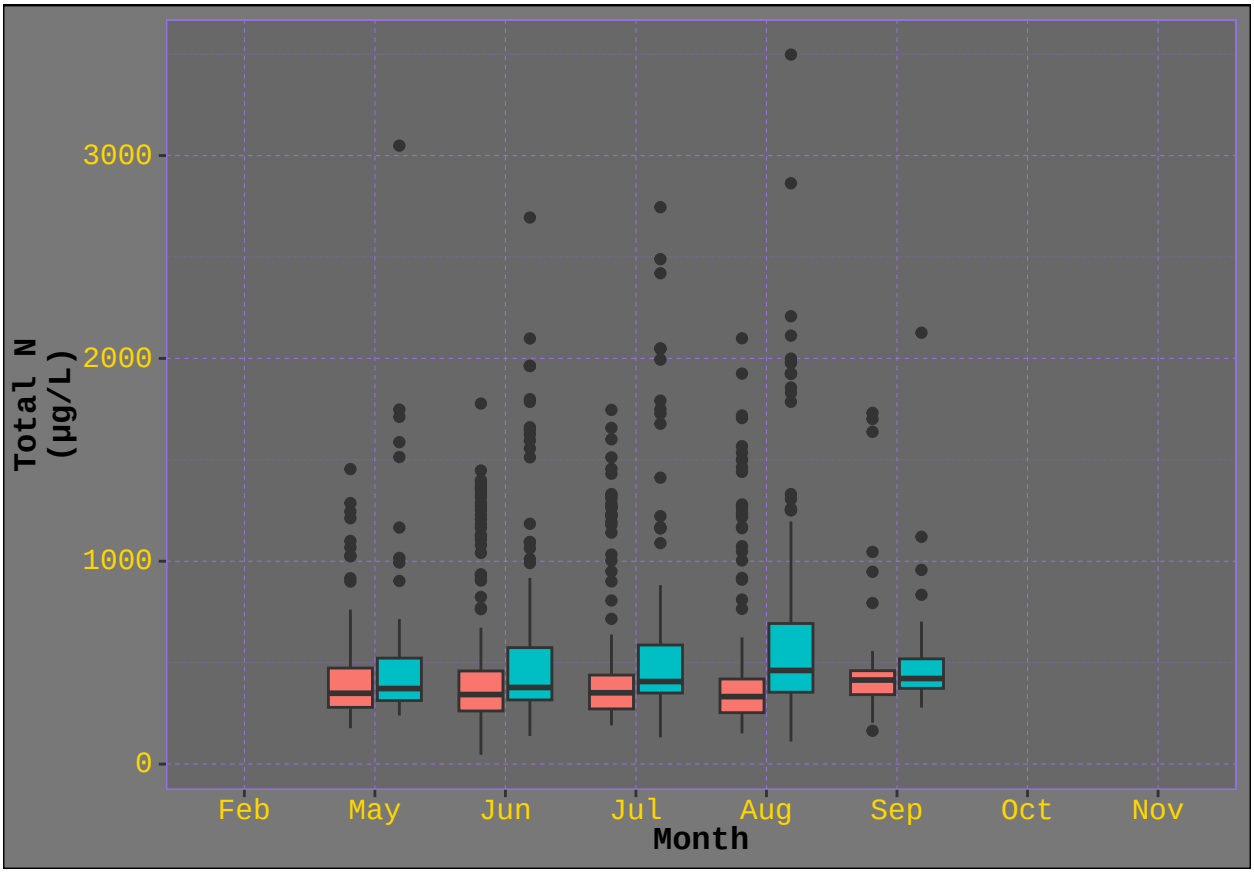
```
  labs(x = element_blank(),
       y = "Total P\n(µg/L)") +
  theme(legend.position = "none",
        axis.text.x = element_blank())

print(tp_plot)
```

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




```
# Total Nitrogen boxplot (now with month labels)
tn_plot <- ggplot(LakeData, aes(x = month, y = tn_ug, fill = lakename)) +
  geom_boxplot() +
  labs(x = "Month",
       y = "Total N\n(µg/L)") +
  theme(legend.position = "none")

print(tn_plot)
```

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



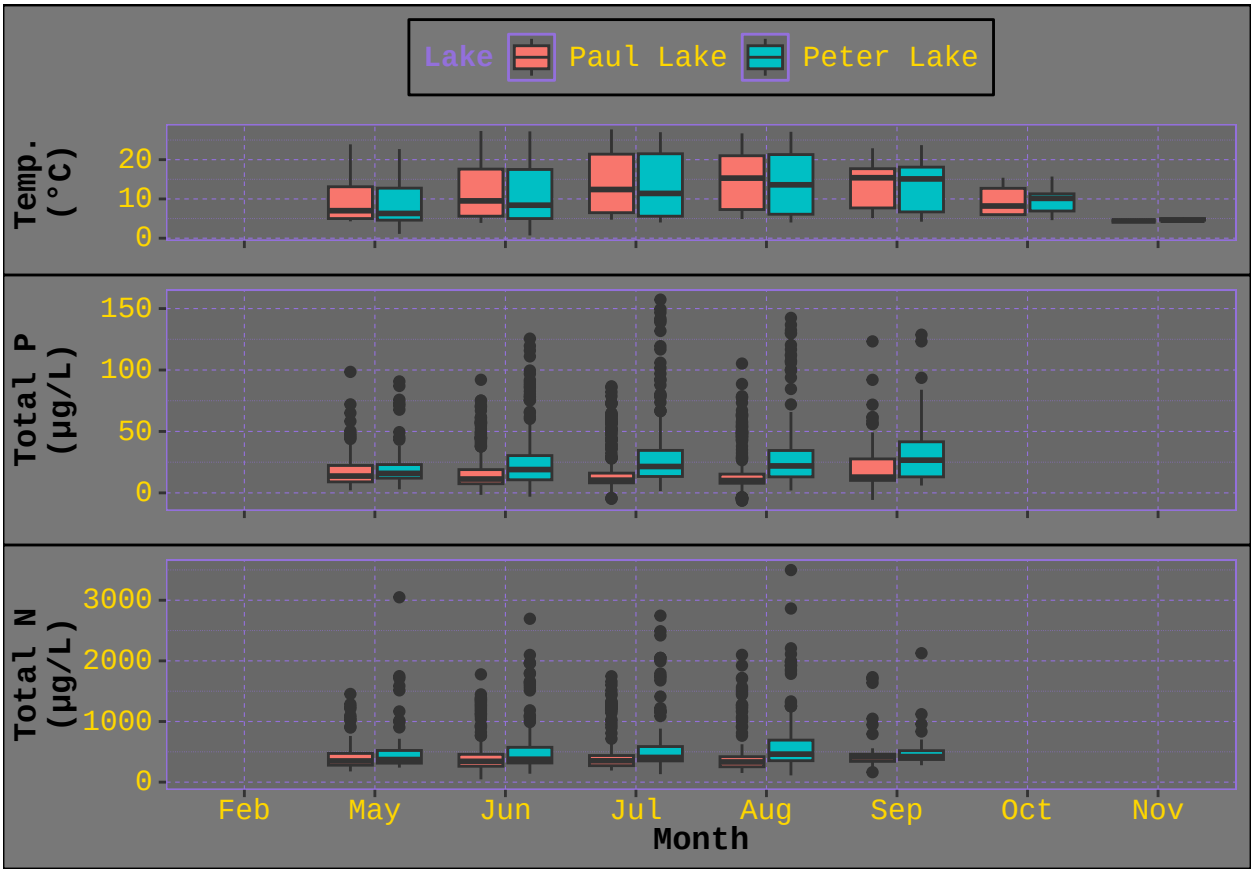

```
# Combine plots

combined_plot <- plot_grid(
  temp_plot,
  tp_plot,
  tn_plot,
  ncol = 1,
  align = "v",
  rel_heights = c(1, 1, 1.2)
)
```

```
## 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()`).
```

```
print(combined_plot)
```



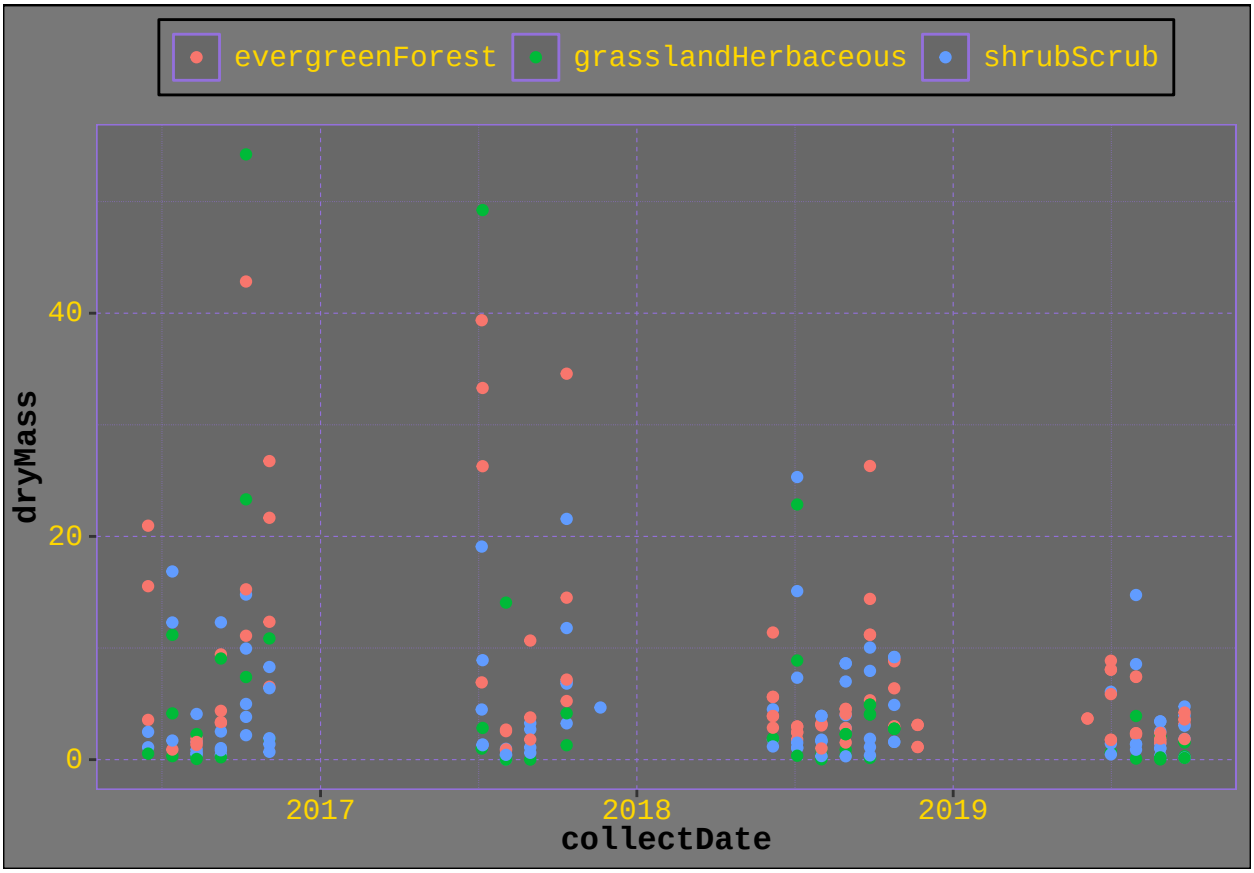
Question: What do you observe about the variables of interest over seasons and between lakes?

Answer: We don't see a lot of measures of Nitrogen or Phosphorus in Jan/Feb/Mar/Apr/Oct/Nov/Dec. The temperatures follow a similar range for both lakes but the phosphorus jumps quite a bit higher in Peter lake, as does the Nitrogen

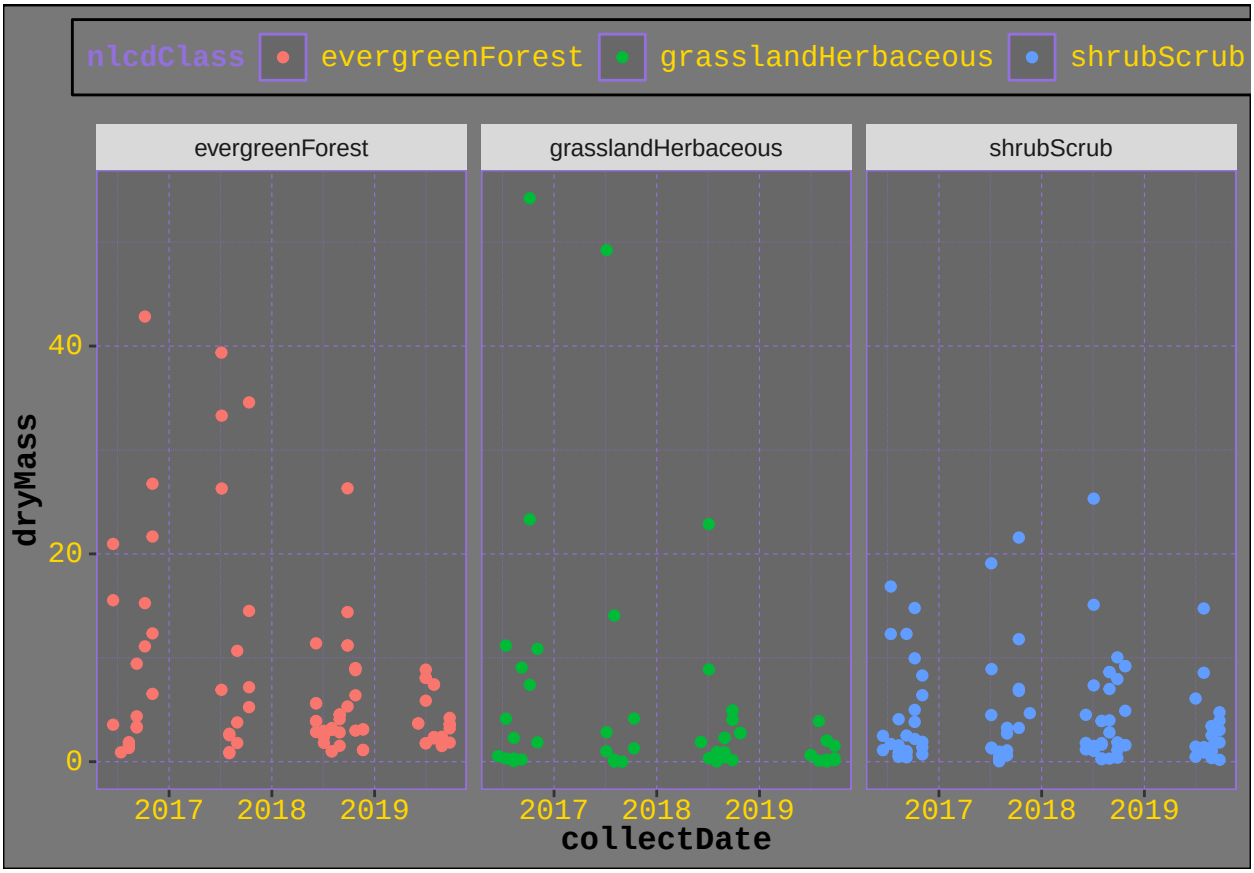
6. [Niwot Ridge] Plot a subset of the litter dataset by displaying only the "Needles" functional group. Plot the dry mass of needle litter by date and separate by NLCD class with a color aesthetic. (no need to adjust the name of each land use)

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

```
#6
#create a plot that looks at the needle weight by data
needles_plot <- ForestLitterData %>%
  filter(functionalGroup == "Needles") %>%
  ggplot(
    mapping = aes(
      x = collectDate,
      y = dryMass,
      color = nlcdClass)) + #color dots by nlcdClass
  geom_point()+
  theme(legend.title = element_blank())

#display the plot
print(needles_plot)
```




```
#7
#create a plot that looks at the needle weight by data
needles_plot_facets <- ForestLitterData %>%
  filter(functionalGroup == "Needles") %>%
  ggplot(
    mapping = aes(
      x = collectDate,
      y = dryMass,
      color = nlcdClass)) +
  geom_point()+
  facet_wrap(facet=vars(nlcdClass))

#display the plot
print(needles_plot_facets)
```



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

Answer: I think #7 is more effective. It more clearly allows for the viewer to see the weight and clustering across different areas.