Assignment 5: Data Visualization

Jack Carpenter

OVERVIEW

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

Directions

1. Change "Student Name" on line 3 (above) with your name.
2. Work through the steps, **creating code and output** that fulfill each instruction.
3. Be sure to **answer the questions** in this assignment document.
4. When you have completed the assignment, **Knit** the text and code into a single PDF file.
5. After Knitting, submit the completed exercise (PDF file) to the dropbox in Sakai. Add your last name into the file name (e.g., "Fay_A05_DataVisualization.Rmd") prior to submission.

The completed exercise is due on Monday, February 14 at 7:00 pm.

Set up your session

1. Set up your session. Verify your working directory and load the tidyverse and cowplot packages. Upload 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) and the processed data file for the Niwot Ridge litter dataset (use the [`NEON_NIWO_Litter_mass_trap_Processed.csv`] version).

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

```
#1 session setup
getwd()
```

```
## [1] "/Users/Jack/Documents/Duke/Spring 2022/Environmental Data Analytics/Environmental_Data_Analytics
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.4     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(cowplot)

PeterPaul.nutrients.chem.phys.processed <- read.csv(
  "./Data/Processed/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv",
  stringsAsFactors = TRUE)
```

```
Niwot.litter.processed <- read.csv(
  "./Data/Processed/NEON_NIWO_Litter_mass_trap_Processed.csv",
  stringsAsFactors = TRUE)

#2 Make sure dates are dates
class(PeterPaul.nutrients.chem.phys.processed$sampledate)
```

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

```
PeterPaul.nutrients.chem.phys.processed$sampledate <-
  as.Date(PeterPaul.nutrients.chem.phys.processed$sampledate,
        format = "%Y-%m-%d")
class(PeterPaul.nutrients.chem.phys.processed$sampledate)
```

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

```
class(Niwot.litter.processed$collectDate)
```

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

```
Niwot.litter.processed$collectDate <-
  as.Date(Niwot.litter.processed$collectDate, format = "%Y-%m-%d")
class(Niwot.litter.processed$collectDate)
```

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

Define your theme

3. Build a theme and set it as your default theme.

```
#3 set theme
theme_set(theme_classic())
```

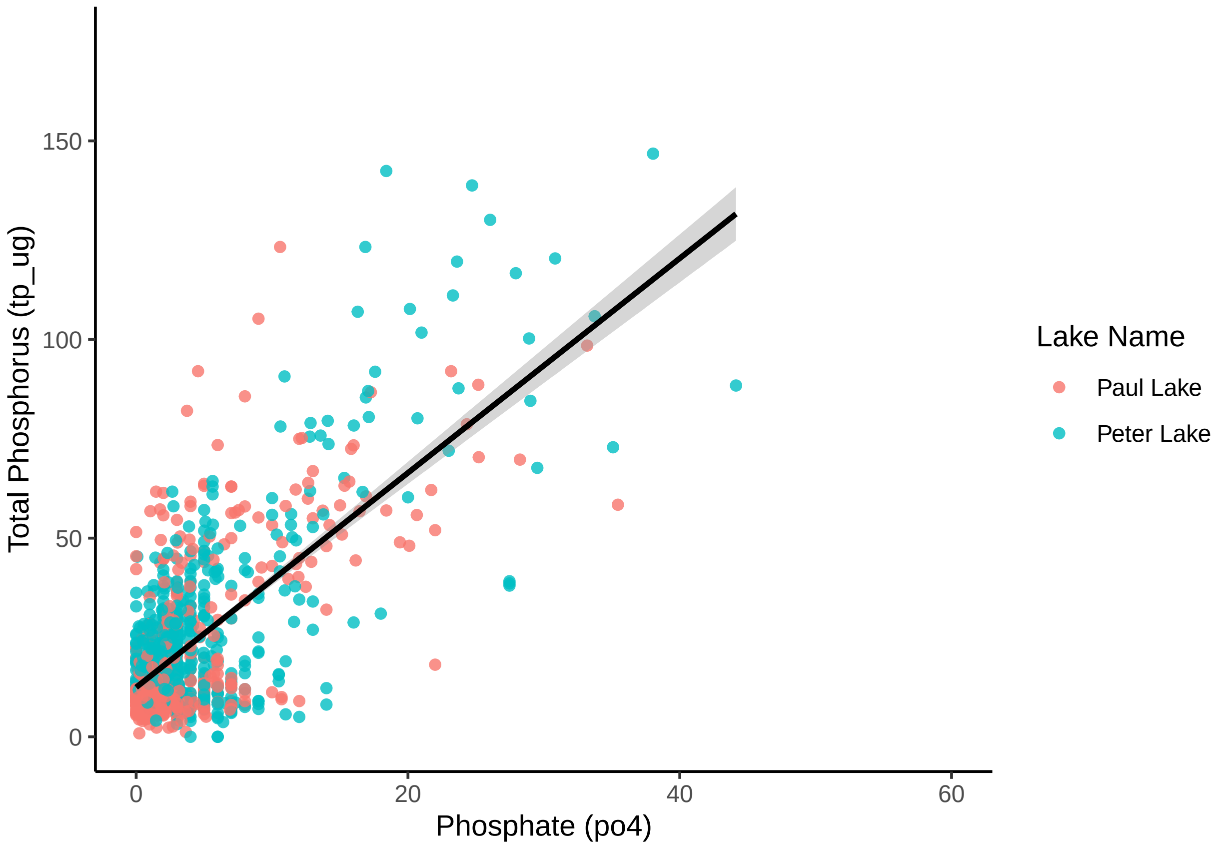
Create graphs

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 `ylim()`).

```
#4 Phosphorus by lake
phosphorus.plot <- ggplot(PeterPaul.nutrients.chem.phys.processed,
                          aes(x = po4, y = tp_ug, color = lakename)) +
  geom_point(alpha = 0.8) +
  geom_smooth(method = lm, color = "black") +
  xlim(0,60) +
  ylim(0,175) +
  labs(x = "Phosphate (po4)", y = "Total Phosphorus (tp_ug)",
       color = "Lake Name")
print(phosphorus.plot)
```

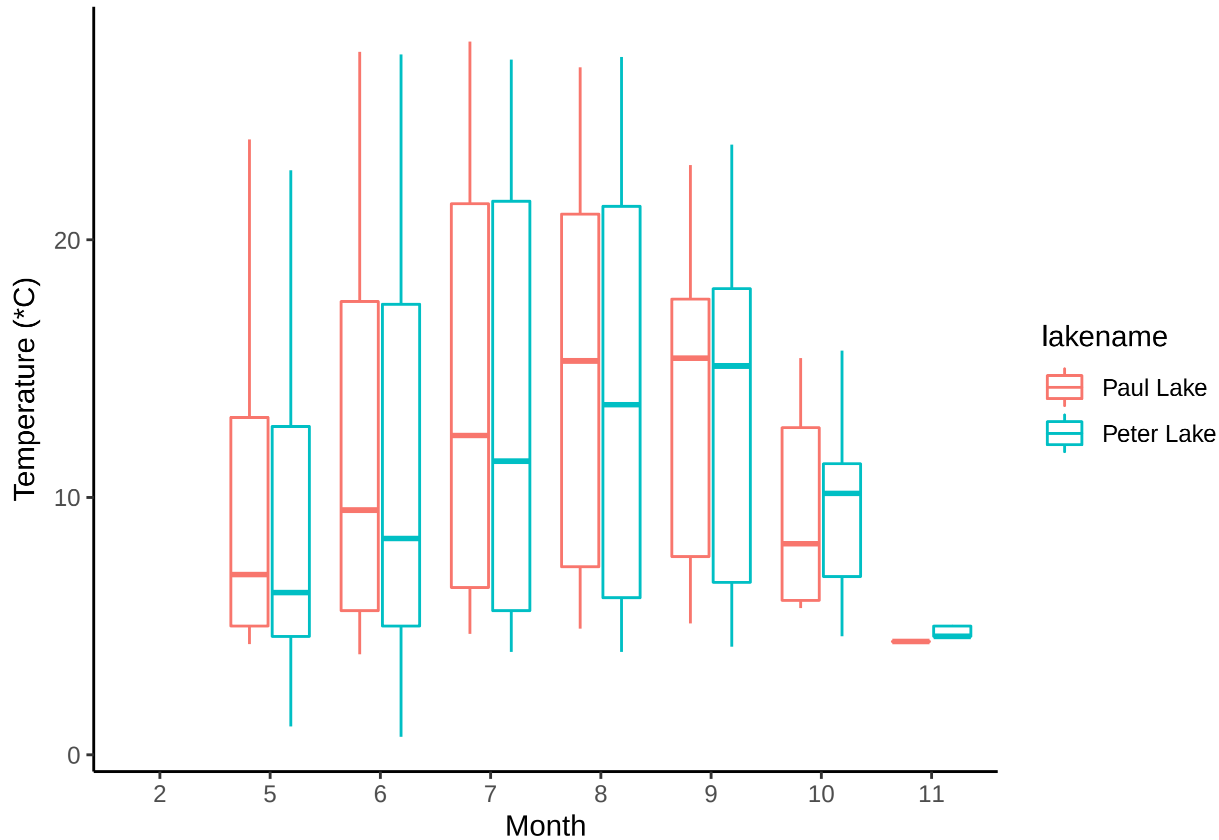
```
## `geom_smooth()` using formula 'y ~ x'

## 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.

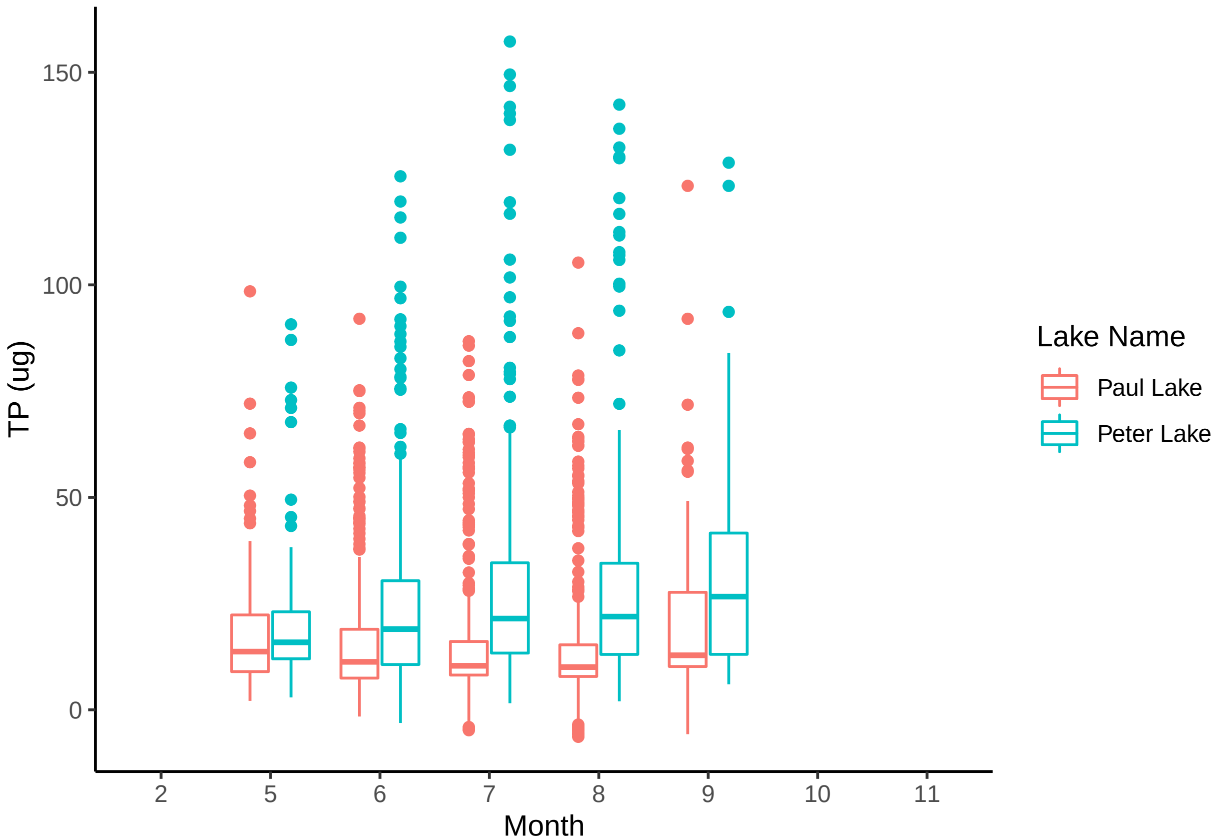
```
#5 Boxplots
# a) Temperature
temp_plot <- ggplot(PeterPaul.nutrients.chem.phys.processed,
              aes(x = as.factor(month), y = temperature_C, color = lakename)) +
  geom_boxplot() +
  labs(x = "Month", y = expression("Temperature (*C)", color = "Lake Name"))
print(temp_plot)
```

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



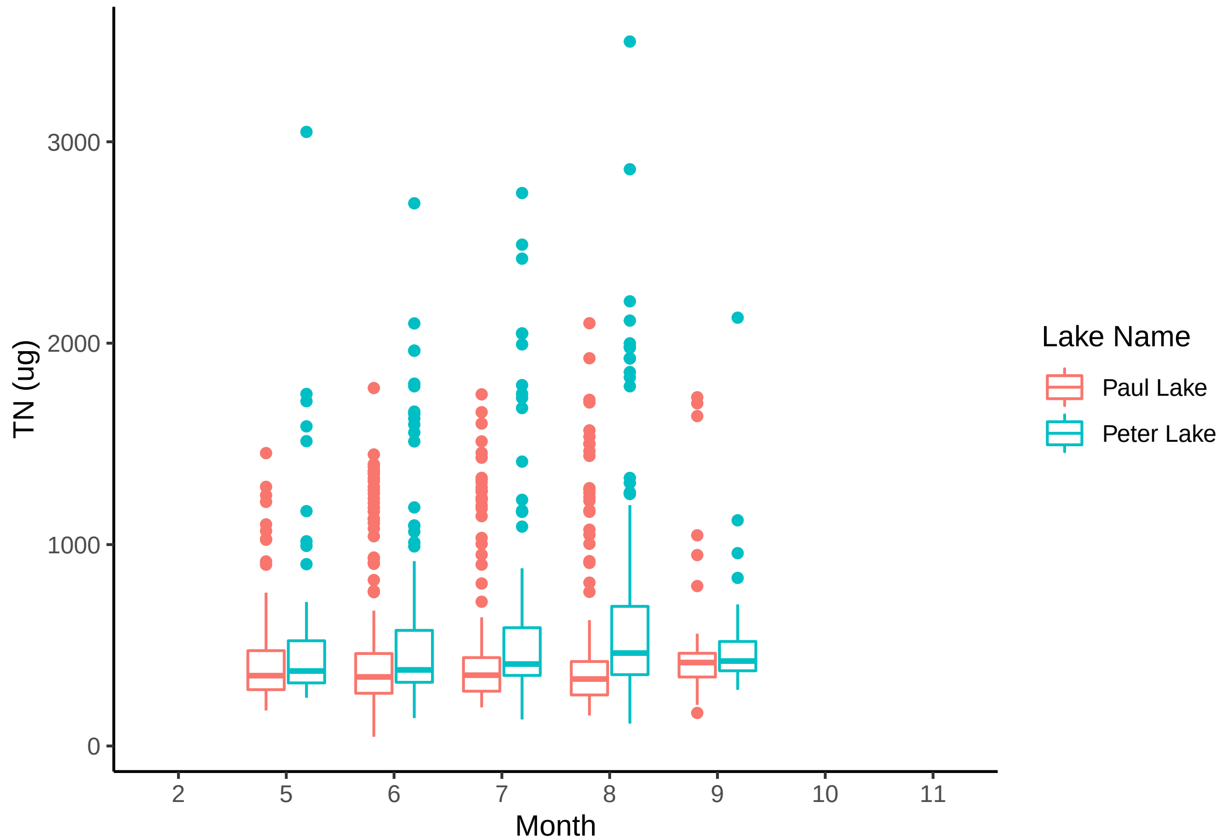

```
# b) TP
TP_plot <- ggplot(PeterPaul.nutrients.chem.phys.processed,
                  aes(x = as.factor(month), y = tp_ug, color = lakename)) +
  geom_boxplot() +
  labs(x = "Month", y = "TP (ug)", color = "Lake Name")
print(TP_plot)
```

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




```r
# c) TN
TN_plot <- ggplot(PeterPaul.nutrients.chem.phys.processed,
                  aes(x = as.factor(month), y = tn_ug, color = lakename)) +
  geom_boxplot() +
  labs(x = "Month", y = "TN (ug)", color = "Lake Name")
print(TN_plot)
```

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



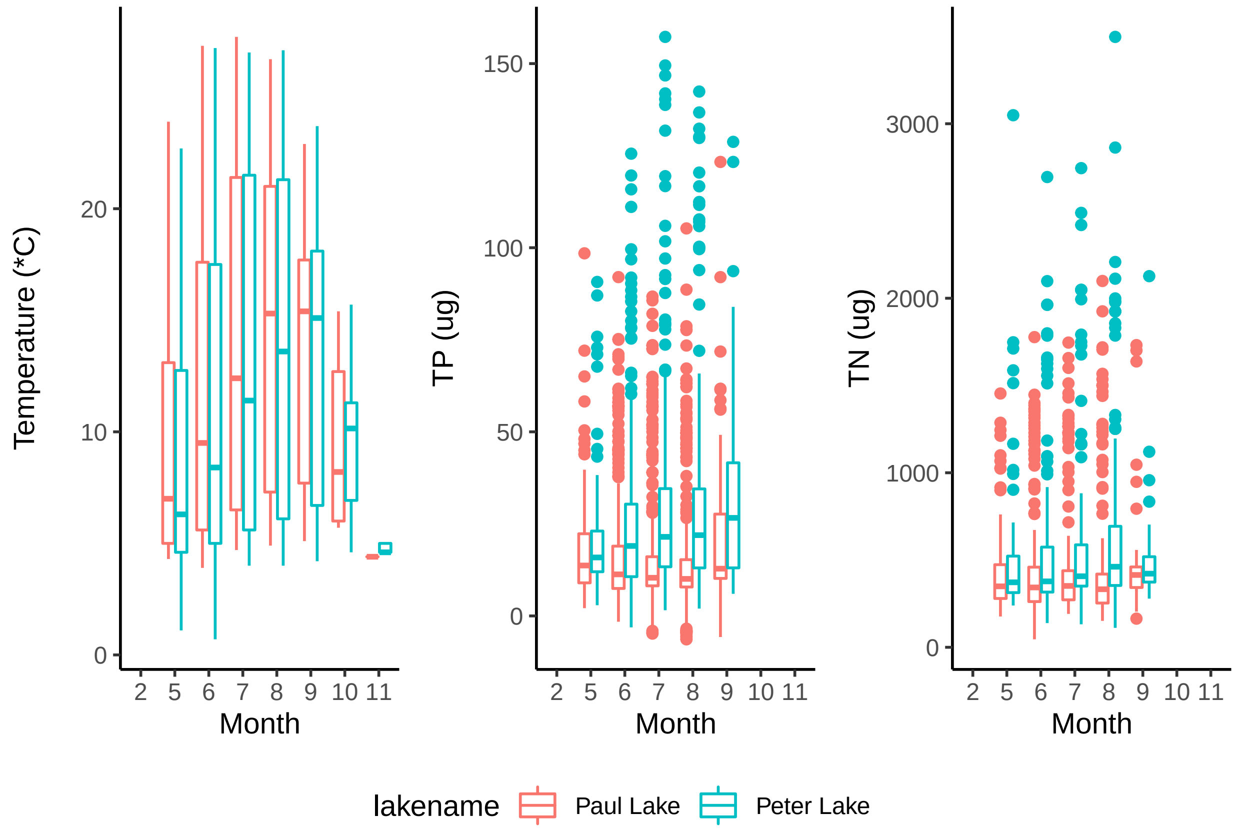

```
# cowplot combining all three
combined_plot <- plot_grid(
  temp_plot + theme(legend.position = "none"),
  TP_plot + theme(legend.position = "none"),
  TN_plot + theme(legend.position = "none"),
  align = "vh",
  nrow  = 1,
  hjust = -1)
```

```
## 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).
```

```
legend <- get_legend(temp_plot +
                     guides(color = guide_legend(nrow = 1)) +
                     theme(legend.position = "bottom"))
```

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

```
combined_plot_legend <- plot_grid(
  combined_plot,
  legend,
  nrow = 2,
  rel_heights = c(3, 0.4))

print(combined_plot_legend)
```



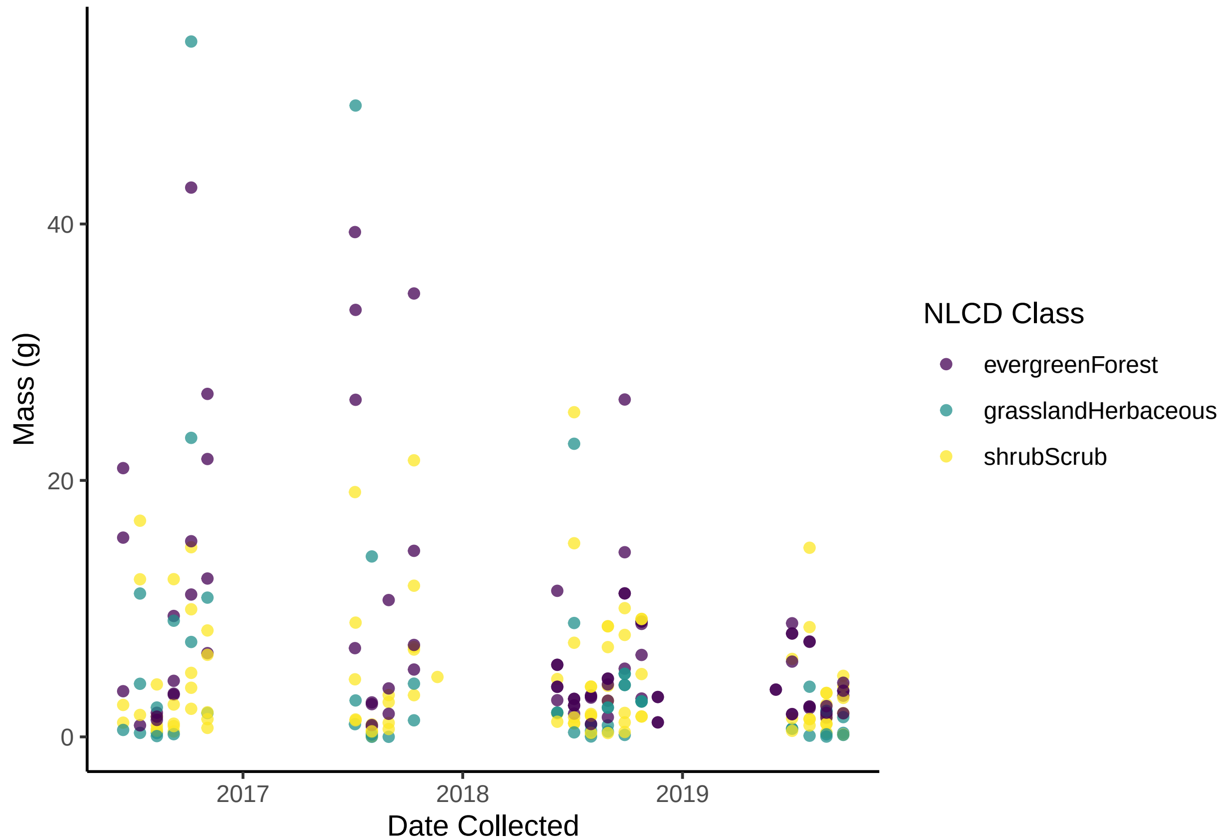

```
# zoom to see whole plot - too big for the little plot window
# how do I get the legend to be horizontal?
```

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

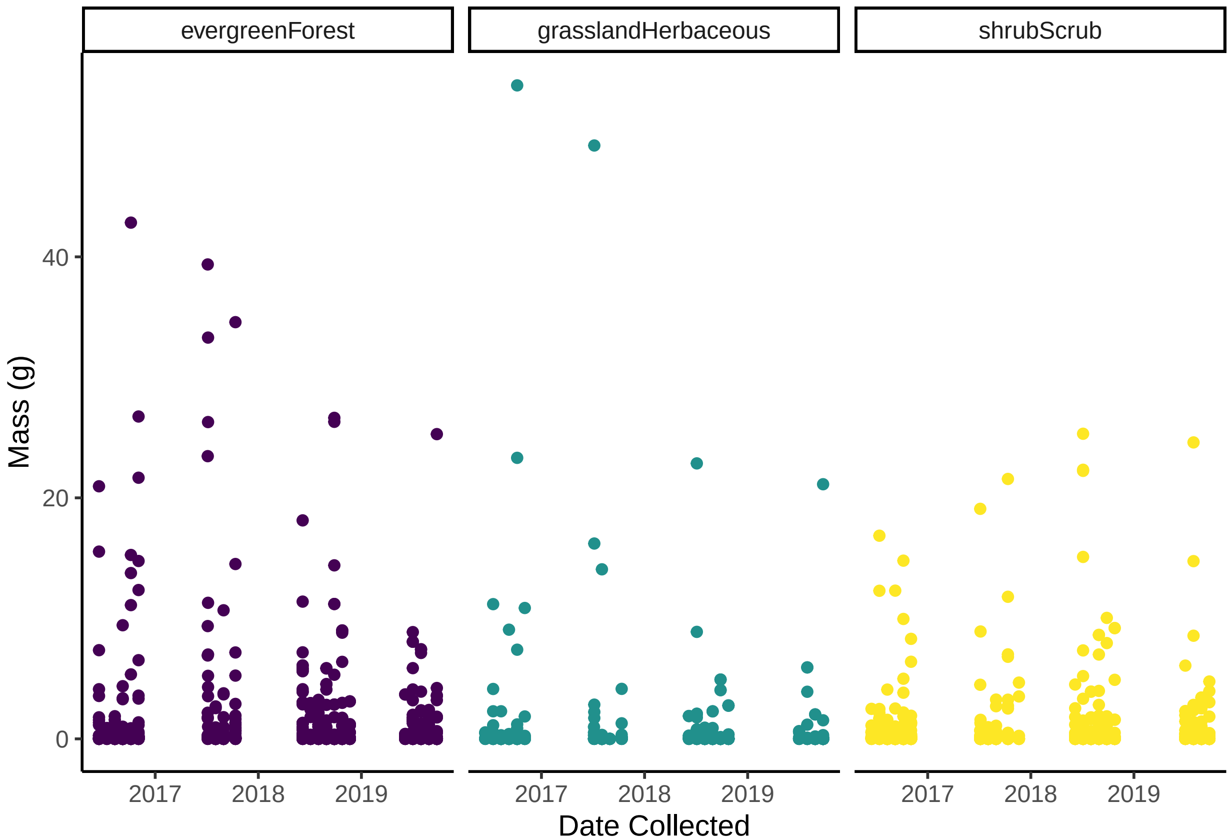
> Answer: Temperature follows exactly the pattern expected, increasing in summer, decreasing in fall. Interestingly, it seems as though Peter Lake is slightly behind Paul Lake, warming later then cooling later. TP in Paul Lake slightly decreases in the summer then increasing in Fall, with more variability in the spring and Fall. Peter Lake shows an increase in both TP and variability in TP through the course of the summer into the Fall. TN in Paul Lake holds steady, with limited variability through the summer, where in Peter Lake it increases in amount and variability through the summer befor decreasing again in September.

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 Needles
needle_plot <- ggplot(subset(Niwot.litter.processed,
                             functionalGroup == "Needles"),
                      aes(x = collectDate, y = dryMass, color = nlcdClass)) +
  geom_point(alpha = 0.75) +
  scale_color_viridis_d() +
  labs(x = "Date Collected", y = "Mass (g)", color = "NLCD Class")
print(needle_plot)
```




```
#7 Separate the NLCD classes into facets
needle_plot_faceted <- ggplot(subset(Niwot.litter.processed,
                                     functionalGroup = "Needles"),
                              aes(x = collectDate, y = dryMass, color = nlcdClass)) +
  geom_point() +
  scale_color_viridis_d() +
  facet_wrap(vars(nlcdClass)) +
  labs(x = "Date Collected", y = "Mass (g)")
needle_plot_faceted + guides(color = "none")
```



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

> Answer: It depends on the focus of the viewer. In the non-faceted plot, its easier to compare between classes over time because the points are all overlaid on one another and an overall pattern may be easier to discern. In the faceted plot, its easier to look at each class separately and look for patterns within each class. They are both effective, it just depends on what "level of zoom" the viewer is looking for.