Assignment 5: Data Visualization

Jake Whisler

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.

```r
#1 Loading the necessary packages, checking the working directory, and grabbing needed files
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 /home/guest/EDE_Fall2023
```

```
library(cowplot)
```

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

```
getwd()
```

```
## [1] "/home/guest/EDE_Fall2023"
```

```
PeterPaul <- read.csv(
  "./Data/Processed_KEY/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv",
  stringsAsFactors = TRUE)

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

#2 Checking date column data formats and changing if needed
class(Litter$collectDate)
```

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

```
class(PeterPaul$sampledate)
```

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

```
Litter$collectDate <- as.Date(Litter$collectDate, format = "%Y-%m-%d")
PeterPaul$sampledate <- as.Date(PeterPaul$sampledate, format = "%Y-%m-%d")
```

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

```r
#3 Creating a theme and setting it as the default
mytheme <- theme_classic(base_size = 13) +
  theme(legend.position = "right", axis.text = element_text(color = "darkolivegreen"))

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 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()`).

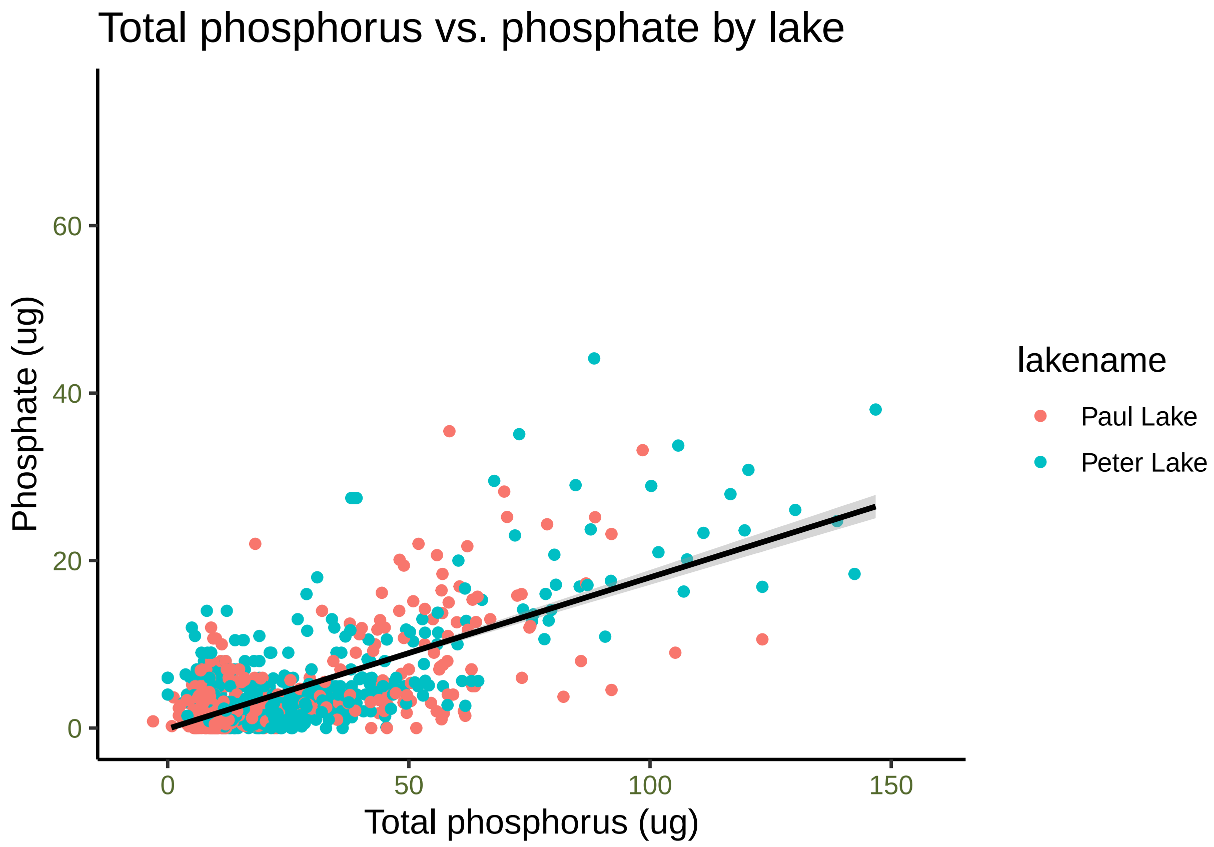
```r
#4 Creating a scatterplot with a line of best fit for both Peter and Paul lakes
PvsPO4 <- ggplot(PeterPaul) +
  geom_point(aes(x = tp_ug, y = po4, color = lakename)) +
  geom_smooth(aes(x = tp_ug, y = po4), method = lm, color = "black") +
  ylim(0,75) +
  xlab("Total phosphorus (ug)") +
  ylab("Phosphate (ug)") +
  ggtitle("Total phosphorus vs. phosphate by lake")
print(PvsPO4)
```

```
## `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()`).

## Warning: Removed 2 rows containing missing values (`geom_smooth()`).
```

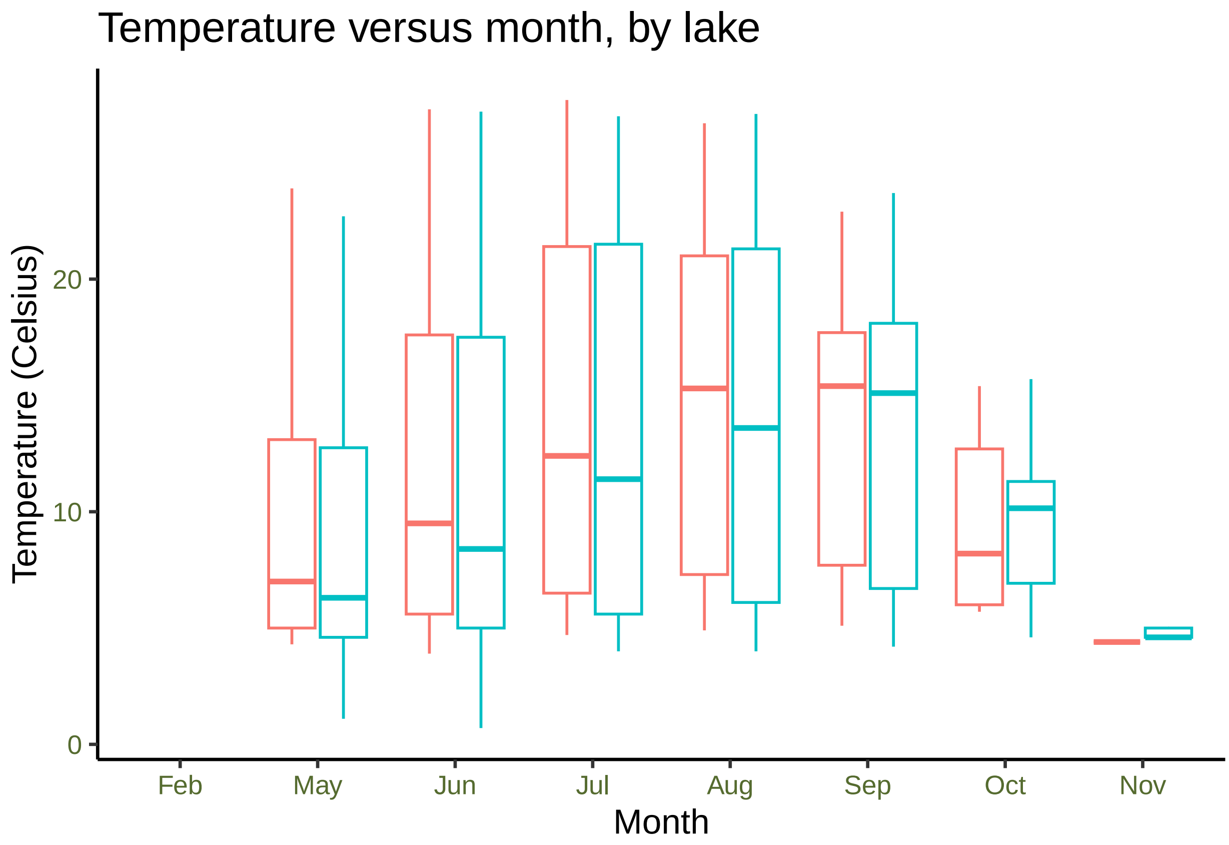


5. [NTL-LTER] Make three separate boxplots of (a) temperature, (b) TP, and (c) TN, with month as the x axis and lake as a color aesthetic. Then, create a cowplot that combines the three graphs. Make sure that only one legend is present and that graph axes are aligned.

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

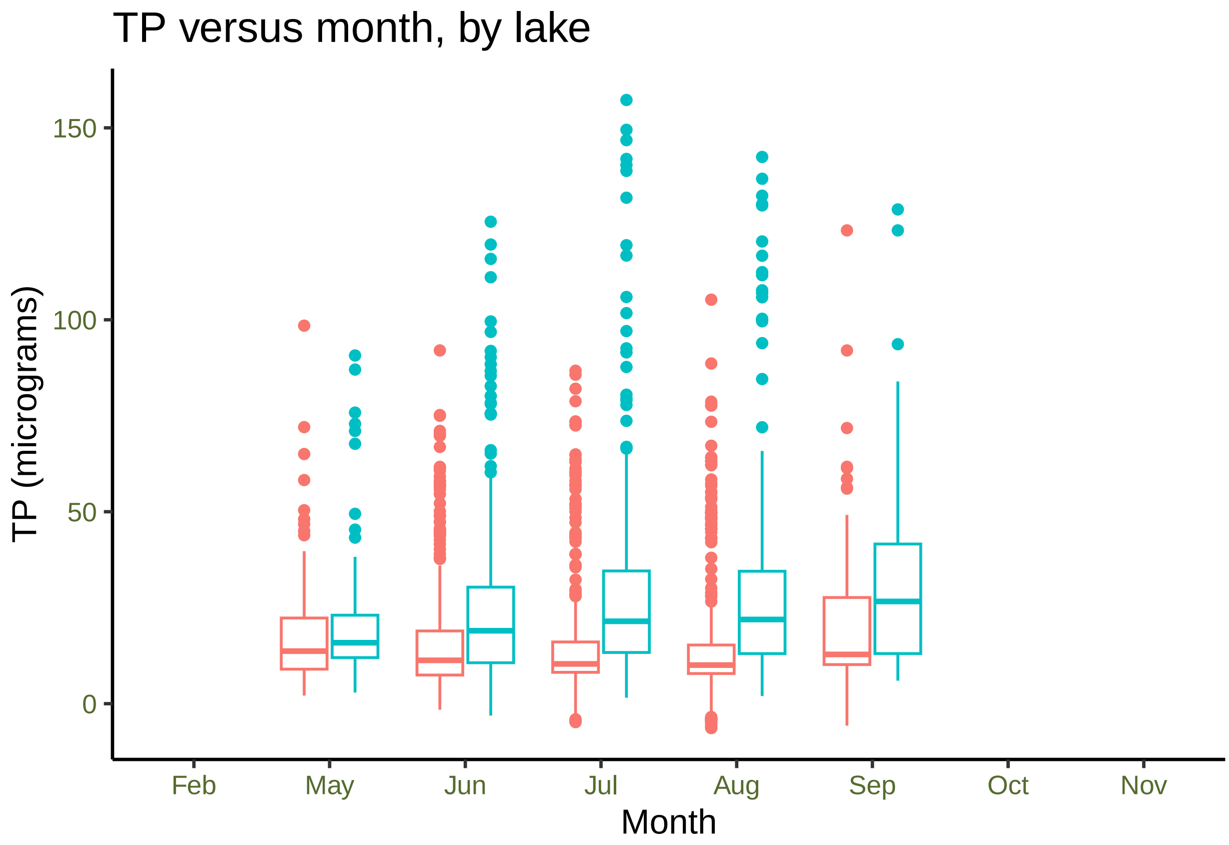
```r
#5 Creating the three boxplots and combining using a cowplot
# Month vs. Temperature
MonthvsTemperature <-
  ggplot(PeterPaul, aes(x = factor(month), y = temperature_C, color = factor(lakename))) +
  geom_boxplot(show.legend = FALSE) +
  xlab("Month") +
  ylab("Temperature (Celsius)") +
  ggtitle("Temperature versus month, by lake") +
  scale_x_discrete(label = c("Feb","May","Jun","Jul","Aug","Sep","Oct","Nov"))
print(MonthvsTemperature)
```

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



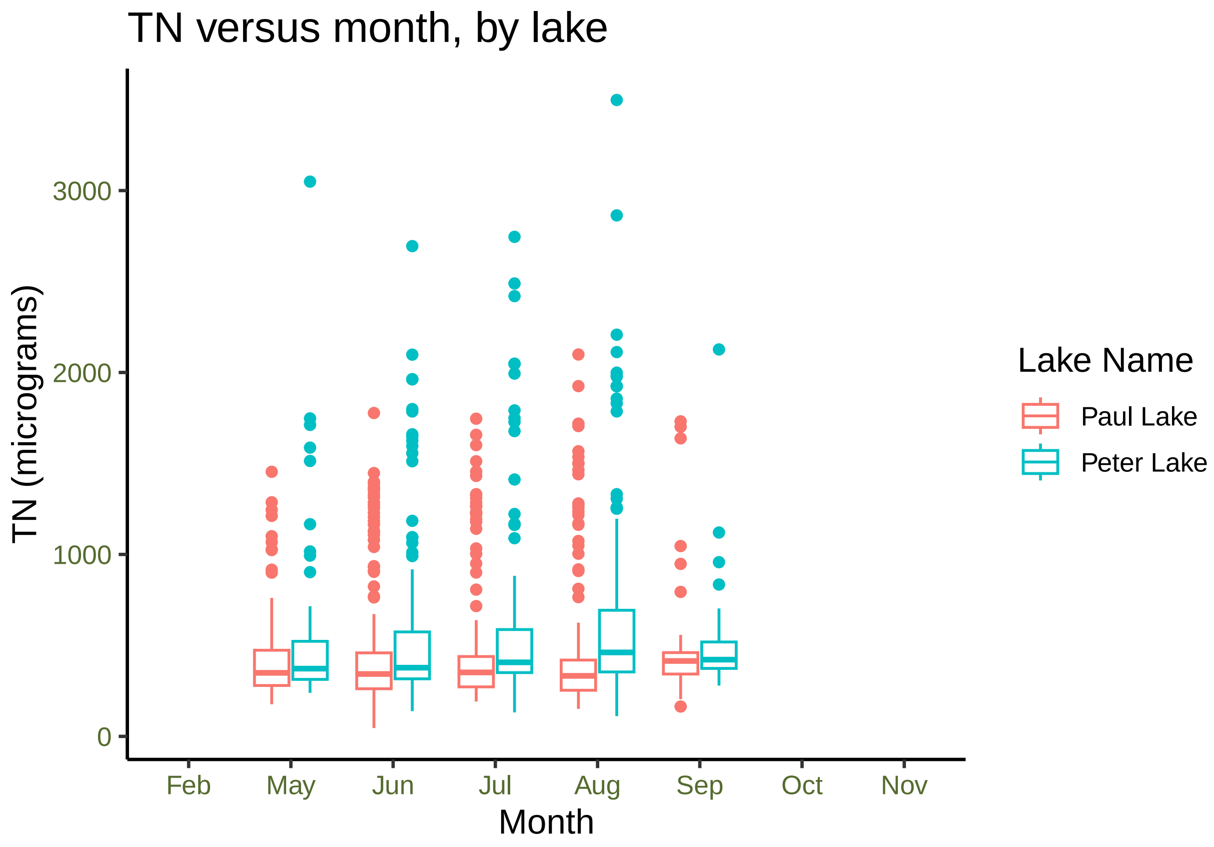
```r
# TP vs. Temperature
TPvsTemperature <-
  ggplot(PeterPaul, aes(x = factor(month), y = tp_ug, color = factor(lakename))) +
  geom_boxplot(show.legend = FALSE) +
  xlab("Month") +
  ylab("TP (micrograms)") +
  ggtitle("TP versus month, by lake") +
  scale_x_discrete(label = c("Feb","May","Jun","Jul","Aug","Sep","Oct","Nov"))
print(TPvsTemperature)
```

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



```
# TN vs. Temperature (includes the legend for later use)
TNvsTemperature <-
  ggplot(PeterPaul, aes(x = factor(month), y = tn_ug, color = factor(lakename))) +
  geom_boxplot() +
  xlab("Month") +
  ylab("TN (micrograms)") +
  ggtitle("TN versus month, by lake") +
  scale_x_discrete(label = c("Feb","May","Jun","Jul","Aug","Sep","Oct","Nov")) +
  labs(color = "Lake Name")
print(TNvsTemperature)
```

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

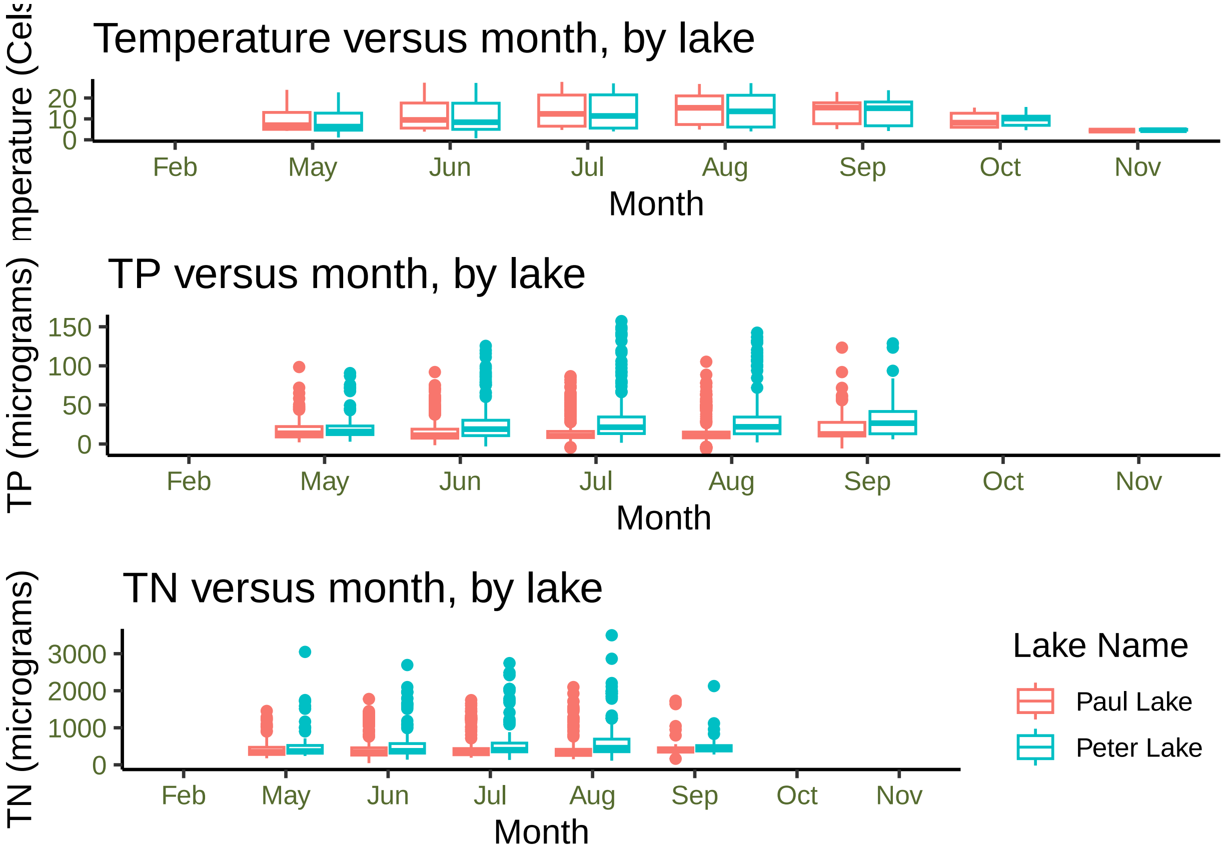


```
#Merging the three plots into one cowplot
plot_grid(MonthvsTemperature, TPvsTemperature, TNvsTemperature, nrow = 3, align = 'h',
          rel_heights = c(1.5,2,2))
```

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



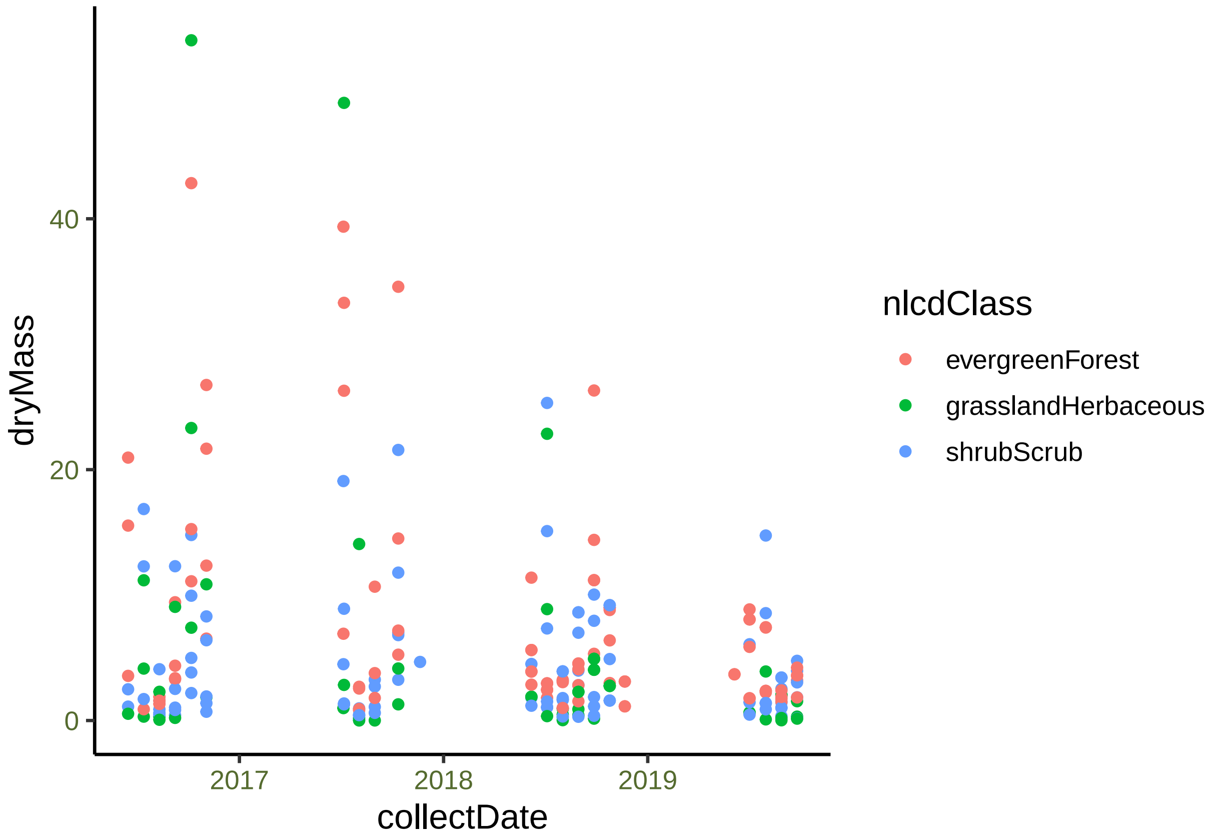
Question: What do you observe about the variables of interest over seasons and between lakes?

> Answer: Generally, it seems as though Peter Lake fluctuates in temperature less than Paul Lake, as Paul Lake becomes hotter in the summer but cooler into the fall than Peter Lake does. Peter Lake seems to contain more TP and TN across the months, and TP increases from May to September. It's hard to tell a pattern in TN across months aside from a peak in Peter Lake 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.

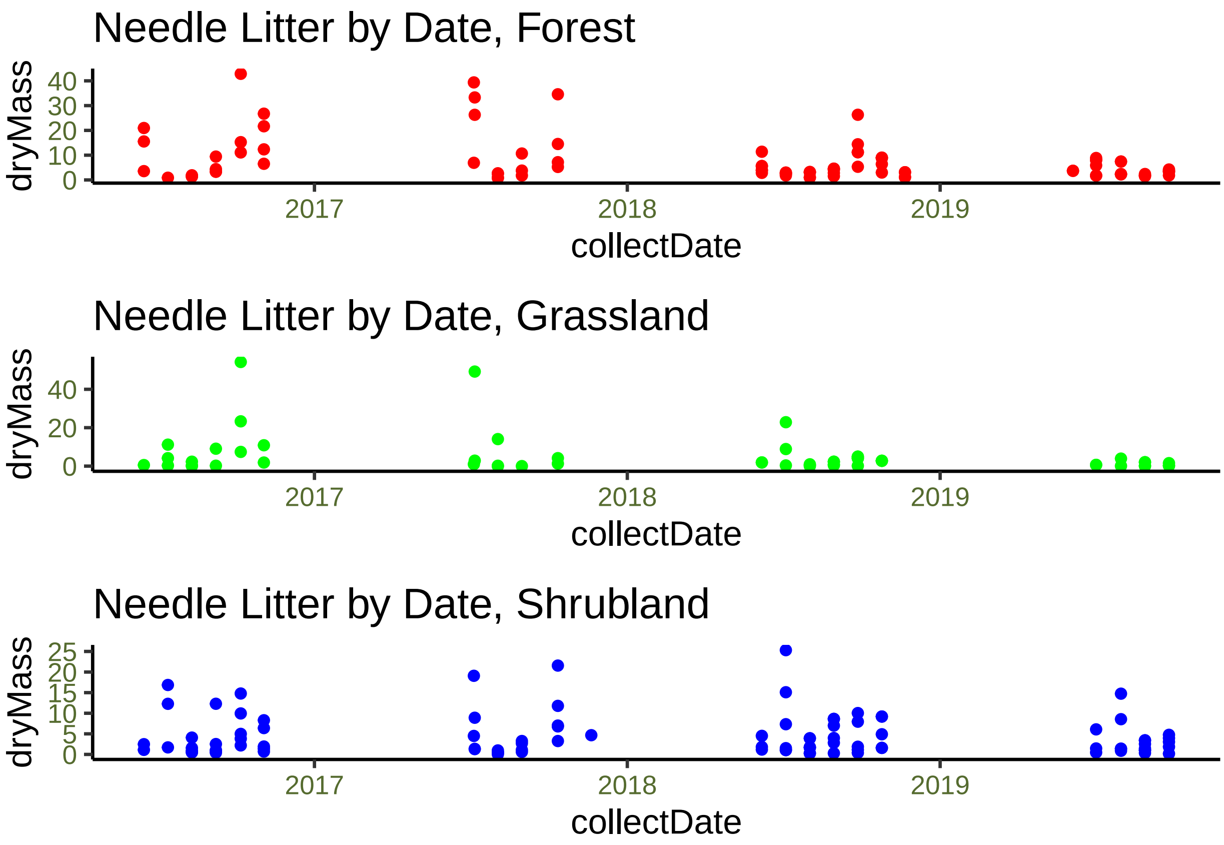
```
#6 Plotting needle litter by date and separated by NLCD class
#Creating a subset of litter data with only Needles
Needles <- subset(Litter, functionalGroup == "Needles")

#Creating the plot with all NLCD classes
NeedleLitterbyDate <-
  ggplot(Needles, aes(x = collectDate, y = dryMass, color = nlcdClass)) +
  geom_point()
print(NeedleLitterbyDate)
```



```
#7 Creating three new graphs and combining them for each NLCD class
NeedleLitterbyDate_Evergreen <-
  ggplot(subset(Needles, nlcdClass == "evergreenForest"),
         aes(x = collectDate, y = dryMass)) +
  geom_point(color = "red") +
  ggtitle("Needle Litter by Date, Forest")
NeedleLitterbyDate_Grassland <-
  ggplot(subset(Needles, nlcdClass == "grasslandHerbaceous"),
         aes(x = collectDate, y = dryMass)) +
  geom_point(color = "green") +
  ggtitle("Needle Litter by Date, Grassland")
NeedleLitterbyDate_Shrub <-
  ggplot(subset(Needles, nlcdClass == "shrubScrub"),
         aes(x = collectDate, y = dryMass)) +
  geom_point(color = "blue") +
  ggtitle("Needle Litter by Date, Shrubland")

NeedleLitterbyDate_separated <-
  plot_grid(NeedleLitterbyDate_Evergreen, NeedleLitterbyDate_Grassland, NeedleLitterbyDate_Shrub,
          nrow = 3)
print(NeedleLitterbyDate_separated)
```



Needle Litter by Date, Forest

Needle Litter by Date, Grassland

Needle Litter by Date, Shrubland

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

> Answer: I believe that the plot with the three different facets from question 7 is more effective, as there is less overlap between the different land use classes' data points, so it can be easier to compare trends in needle litter by date for the three classes. The color-separated plot can be used to draw overall conclusions about needle litter trends, but the land use class is harder to glean anything from.