Assignment 5: Data Visualization

Amanda Sajewski

Spring 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. 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
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.4.0     v purrr   1.0.1
## v tibble  3.1.8     v dplyr   1.1.0
## v tidyr   1.3.0     v stringr 1.5.0
## v readr   2.1.3     v forcats 1.0.0
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(lubridate)
```

```
##
## Attaching package: ’lubridate’
##
## The following objects are masked from ’package:base’:
##
##     date, intersect, setdiff, union
```

```
library(here)
```

```
## here() starts at C:/Users/asaje/EDA-Spring2023/EDA-Spring2023
```

```
library(cowplot)
```

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

```
PeterPaul.chem.nutrients <-
  read.csv(here("Data/Processed_KEY/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv"),
           stringsAsFactors = T)
Litter <- read.csv(here("Data/Processed_KEY/NEON_NIWO_Litter_mass_trap_Processed.csv"),
                   stringsAsFactors = T)

#2
str(Litter)
```

```
## ’data.frame':    1692 obs. of  13 variables:
##  $ plotID          : Factor w/ 12 levels "NIWO_040","NIWO_041",..: 9 8 9 11 7 7 4 4 4 4 ...
##  $ trapID          : Factor w/ 15 levels "NIWO_040_139",..: 11 10 11 13 9 9 5 5 5 5 ...
##  $ collectDate     : Factor w/ 24 levels "2016-06-16","2016-07-14",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ functionalGroup : Factor w/ 8 levels "Flowers","Leaves",..: 6 5 8 6 4 2 2 6 7 8 ...
##  $ dryMass         : num  0 0.27 0.12 0 1.11 0 0 0 0.07 0.02 ...
##  $ qaDryMass       : Factor w/ 2 levels "N","Y": 1 1 1 1 2 1 1 1 1 1 ...
##  $ subplotID       : int  31 41 31 32 32 32 40 40 40 40 ...
##  $ decimalLatitude : num  40.1 40 40.1 40 40 ...
##  $ decimalLongitude: num  -106 -106 -106 -106 -106 ...
##  $ elevation       : num  3477 3413 3477 3373 3446 ...
##  $ nlcdClass       : Factor w/ 3 levels "evergreenForest",..: 3 1 3 1 3 3 2 2 2 2 ...
##  $ plotType        : Factor w/ 1 level "tower": 1 1 1 1 1 1 1 1 1 1 ...
##  $ geodeticDatum   : Factor w/ 1 level "WGS84": 1 1 1 1 1 1 1 1 1 1 ...
```

```
str(PeterPaul.chem.nutrients)
```

```
## ’data.frame’:    23008 obs. of  15 variables:
##  $ lakename       : Factor w/ 2 levels "Paul Lake","Peter Lake": 1 1 1 1 1 1 1 1 1 1 ...
##  $ year4          : int  1984 1984 1984 1984 1984 1984 1984 1984 1984 1984 ...
##  $ daynum         : int  148 148 148 148 148 148 148 148 148 148 ...
##  $ month          : int  5 5 5 5 5 5 5 5 5 5 ...
##  $ sampledate     : Factor w/ 1103 levels "1984-05-27","1984-05-28",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ depth          : num  0 0.25 0.5 0.75 1 1.5 2 3 4 5 ...
##  $ temperature_C  : num  14.5 NA NA NA 14.5 NA 14.2 11 7 6.1 ...
##  $ dissolvedOxygen: num  9.5 NA NA NA 8.8 NA 8.6 11.5 11.9 2.5 ...
##  $ irradianceWater: num  1750 1550 1150 975 870 610 420 220 100 34 ...
##  $ irradianceDeck : num  1620 1620 1620 1620 1620 1620 1620 1620 1620 1620 ...
##  $ tn_ug          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ tp_ug          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ nh34           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ no23           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ po4            : num  NA NA NA NA NA NA NA NA NA NA ...
```

```r
Litter$collectDate <- ymd(Litter$collectDate)
PeterPaul.chem.nutrients$sampledate <- ymd(PeterPaul.chem.nutrients$sampledate)
```

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

mytheme <- theme_bw(base_size = 12) +
    theme(axis.text = element_text(color = "black"),
        legend.position = "right")
```

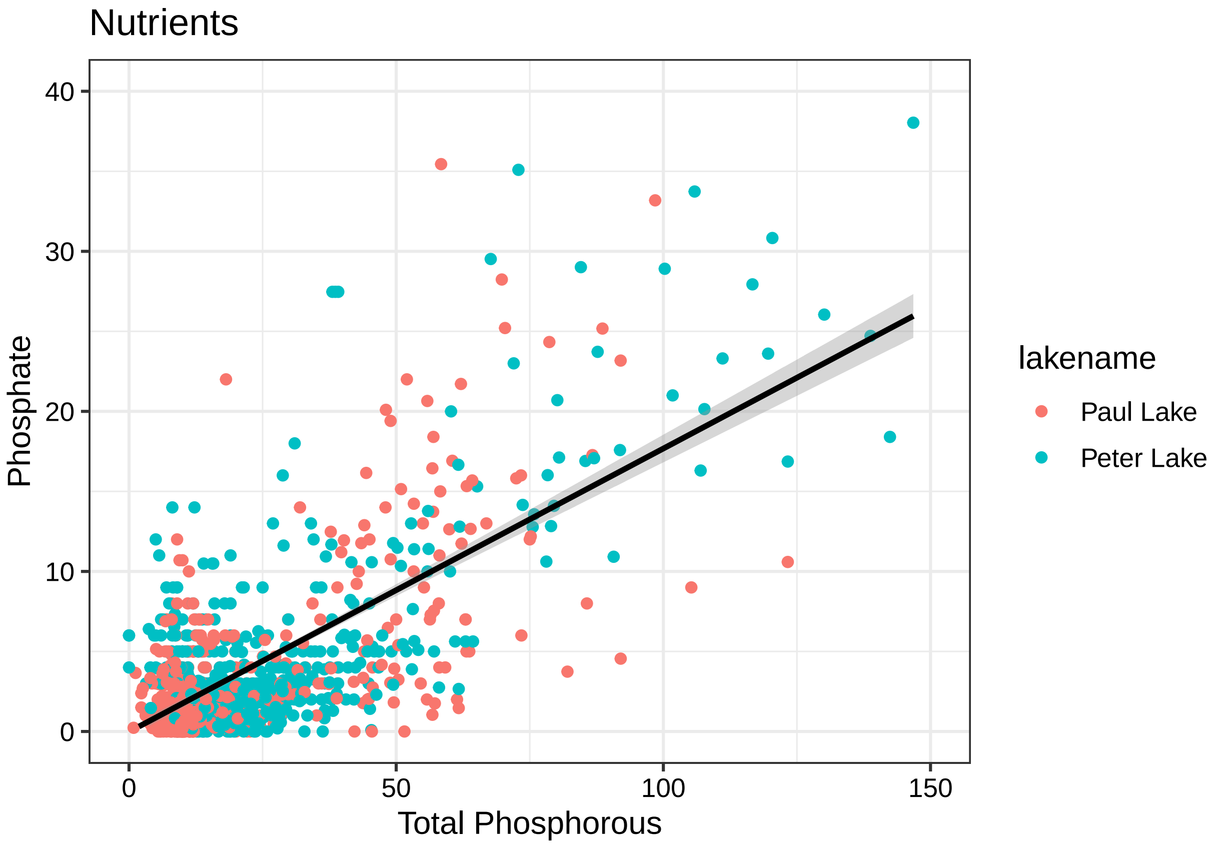
Create graphs

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
theme_set(mytheme)

phosvphos <-
  ggplot(PeterPaul.chem.nutrients, aes(x = tp_ug, y = po4, color=lakename)) +
  geom_point()+
  ylim(0,40) +
  xlim (0,150) +
  geom_smooth(method=lm, colour="black")+
  labs( title = "Nutrients",
        x= "Total Phosphorous",
        y="Phosphate")

print(phosvphos)
```

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

## Warning: Removed 21949 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21949 rows containing missing values (`geom_point()`).

## Warning: Removed 1 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.

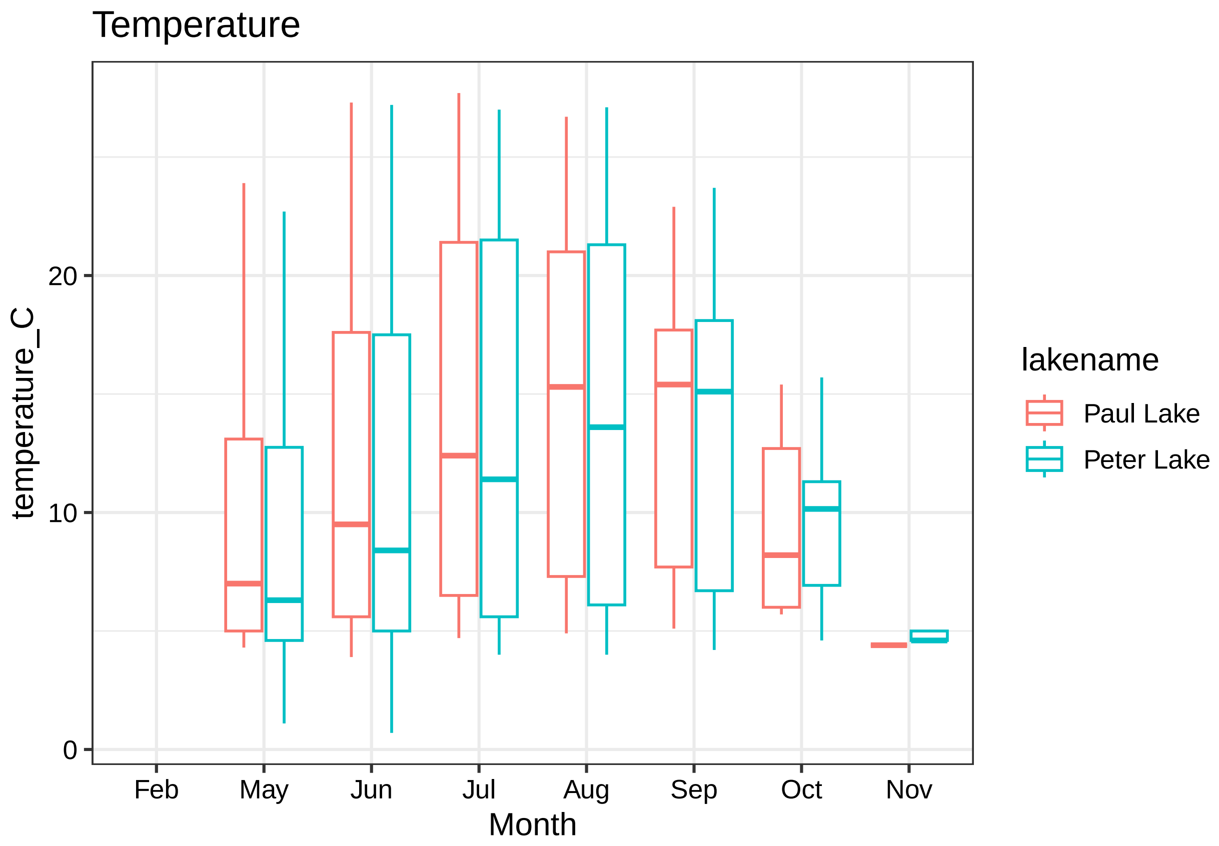
Tip: R has a build in variable called `month.abb` that returns a list of months;see https://r-lang.com/month-abb-in-r-with-example

```
#5
PeterPaul.chem.nutrients$month <- factor(PeterPaul.chem.nutrients$month, levels = 1:12, labels = month.a
class(PeterPaul.chem.nutrients$month)
```

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

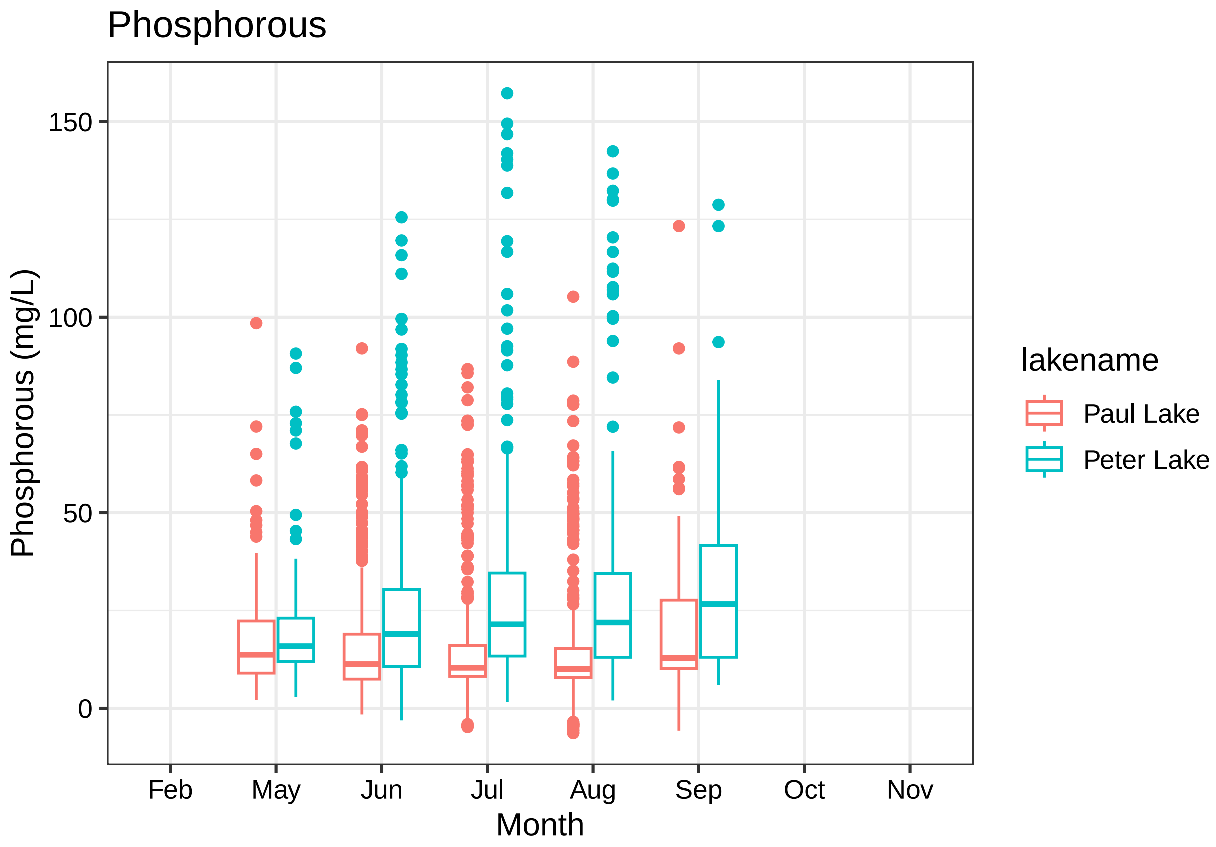
```
NutrientTemp <- ggplot(PeterPaul.chem.nutrients,
       aes(x=factor(month), y = temperature_C))+
  geom_boxplot(aes(color=lakename)) +
  labs(title= "Temperature",
       x= "Month")

print(NutrientTemp)
```

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



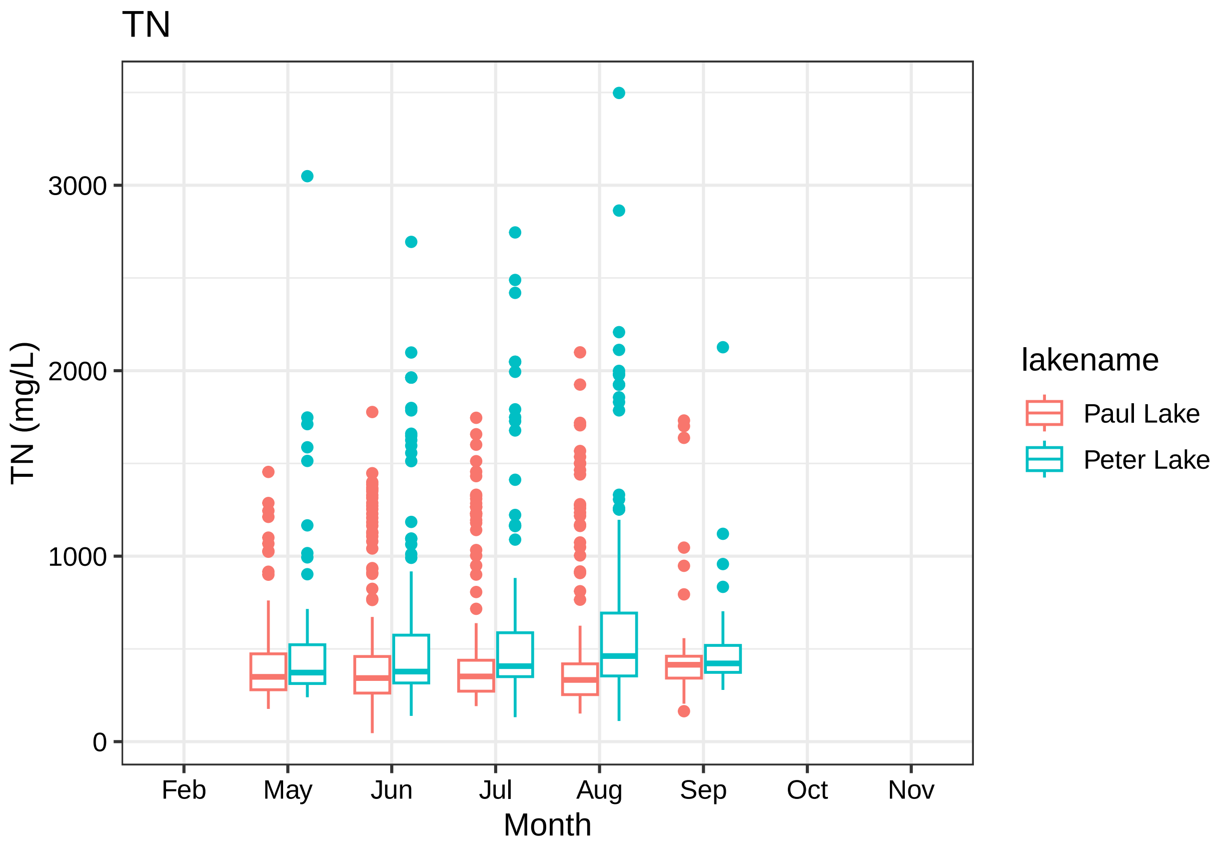
```
NutrientTP <- ggplot(PeterPaul.chem.nutrients,
       aes(x=factor(month), y = tp_ug))+
  geom_boxplot(aes(color=lakename)) +
  labs(title= "Phosphorous",
       x= "Month",
       y="Phosphorous (mg/L)")

print(NutrientTP)
```

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



```
NutrientTN <- ggplot(PeterPaul.chem.nutrients,
       aes(x=factor(month), y = tn_ug))+
  geom_boxplot(aes(color=lakename)) +
  labs(title= "TN",
       x= "Month",
       y="TN (mg/L)")

print(NutrientTN)
```

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



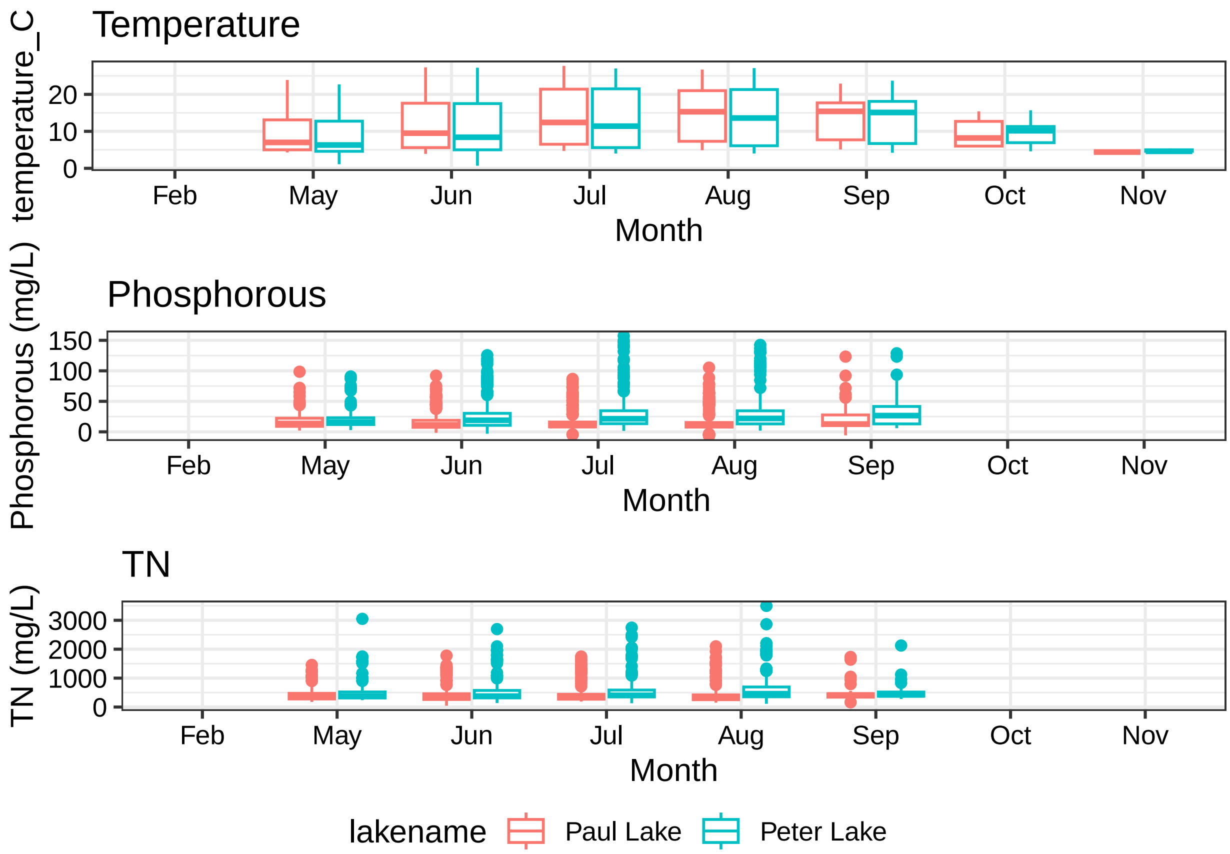
```
legend <- get_legend(NutrientTemp +
    guides(color = guide_legend(nrow = 1)) +
    theme(legend.position = "bottom"))
```

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

```
Nutrient3a <- plot_grid(
  NutrientTemp + theme(legend.position="none"),
  NutrientTP + theme(legend.position="none"),
  NutrientTN + theme(legend.position="none"),nrow = 4, ncol = 1, align = 'h',legend,
  rel_heights = c(1.25, 1.25, 1.25,.25))
```

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

```
print(Nutrient3a)
```



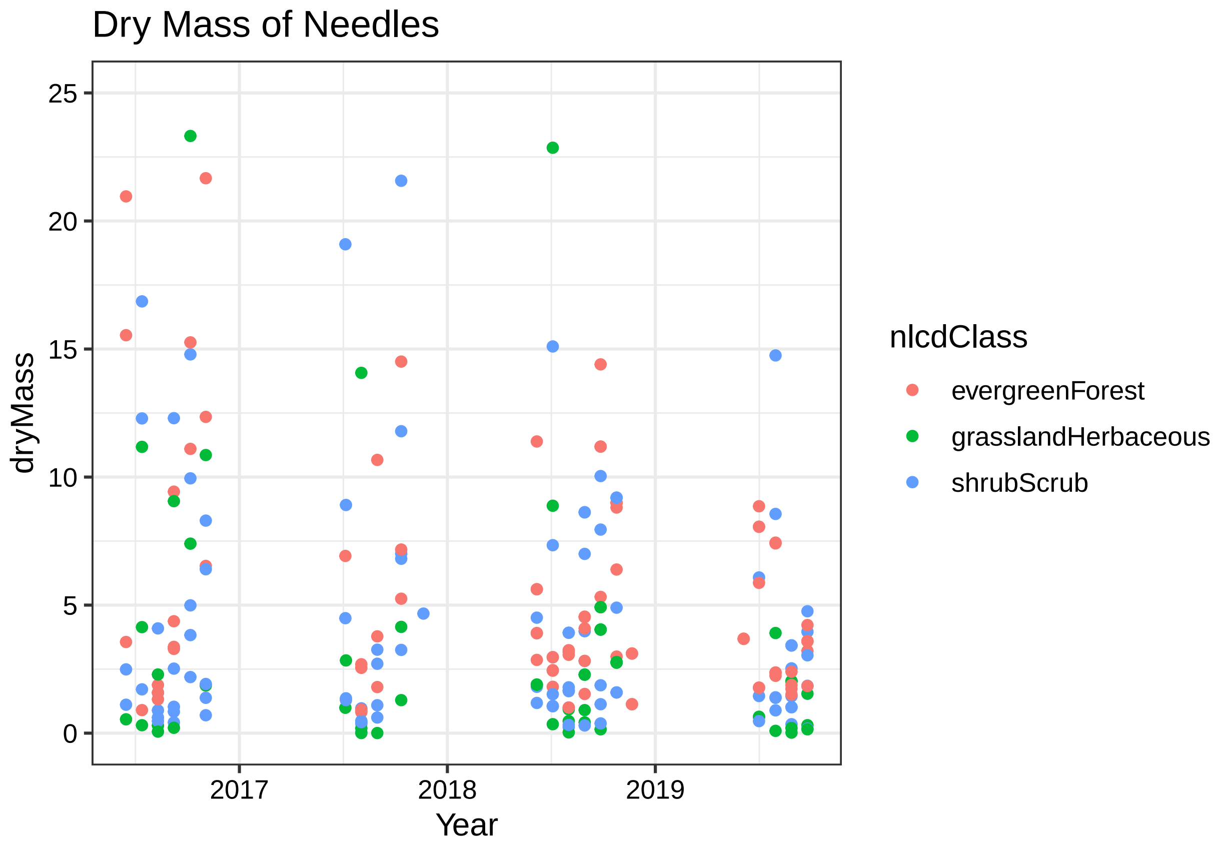
Question: What do you observe about the variables of interest over seasons and between lakes?

> Answer: Temperature is pretty similar across lakes and it rises and falls from spring to fall. Phosphorous rises across time, although it stays mostly steady in Paul Lake. TN stays mostly steady over time and across both lakes only rising marginally in September. Particularly for the nutrients warmer months had a wider distribution especially in Peter 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.

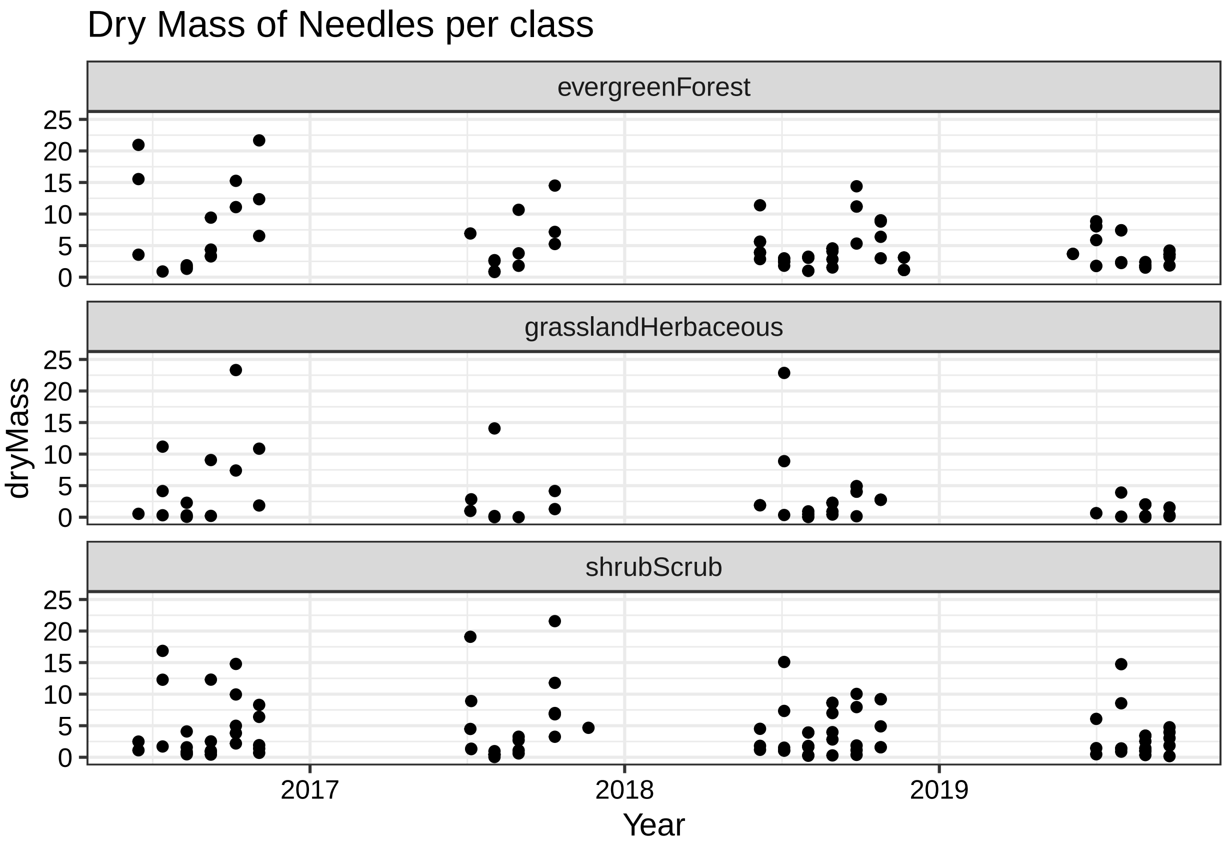
```
#6

Needles <- filter(Litter, functionalGroup == "Needles")

Needlesplot <- ggplot(Needles, aes(y=dryMass, x=collectDate))+
  geom_point(aes(color=nlcdClass)) + ylim(0,25) + labs(title = "Dry Mass of Needles", x= "Year")

#I restricted the y axis to get rid of outliers and visualize the bottom points better. I chose a scatt

print(Needlesplot)
```

```
## Warning: Removed 10 rows containing missing values (`geom_point()`).
```



```
#7

Needles.faceted <-
  ggplot(Needles, aes(y = dryMass, x = collectDate)) +
  geom_point() +
  facet_wrap(vars(nlcdClass), nrow = 3)+ ylim(0,25) + labs(title = "Dry Mass of Needles per class", x=

#I kept the same limit on the y axis so the plots would be directly comparable.

print(Needles.faceted)
```

```
## Warning: Removed 10 rows containing missing values (‘geom_point()‘).
```



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

Answer: I think plot 6 is more effective because all the points are directly comparable and you see which colors are most prevalent in each year and each weight. Whereas in plot 7 it takes more effort to see the differences in each class in each year. There are not so many points where the colors get muddied by being all together as in plot 6.