Assignment 5: Data Visualization

Jaleesia Amos

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

#----Load packages into session & download if not already installed----#
pacman::p_load(tidyverse, lubridate, here, cowplot)

#----Check working directory----#
getwd()
```

```
## [1] "/Users/jaleesiad.amos/Documents/EDA-Spring2023"
```

```r
#----Read in dataset 1: Nutrients dataset-----#
PeterPaul_processed <- read.csv("./Data/Processed/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed

#----Read in dataset 2: Litter dataset------#
litter_processed <- read.csv("./Data/Processed/NEON_NIWO_Litter_mass_trap_Processed.csv", stringsAsFact

#2

#----Checking structure of Nutrients for date format------#
str(PeterPaul_processed)
```

```
## '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
#----Checking structure of Litter for date format------#
str(litter_processed)
```

```
## '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 ...
```

```r
#_____Date Conversion: Nutrients dataset______#
PeterPaul_processed$sampledate <- ymd(PeterPaul_processed$sampledate)

#_____Date Conversion: Litter dataset______#
litter_processed$collectDate <- ymd(litter_processed$collectDate)
```

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
#----Custom theme: change title, axis title, and legend----#
jatheme <- theme_bw(base_size = 12) +
  theme(plot.title = element_text(face ="bold", hjust = 0.5),
        axis.title = element_text(face = "bold"),
        legend.position = "right")

#-----Set theme for environment-----#
theme_set(jatheme)
```

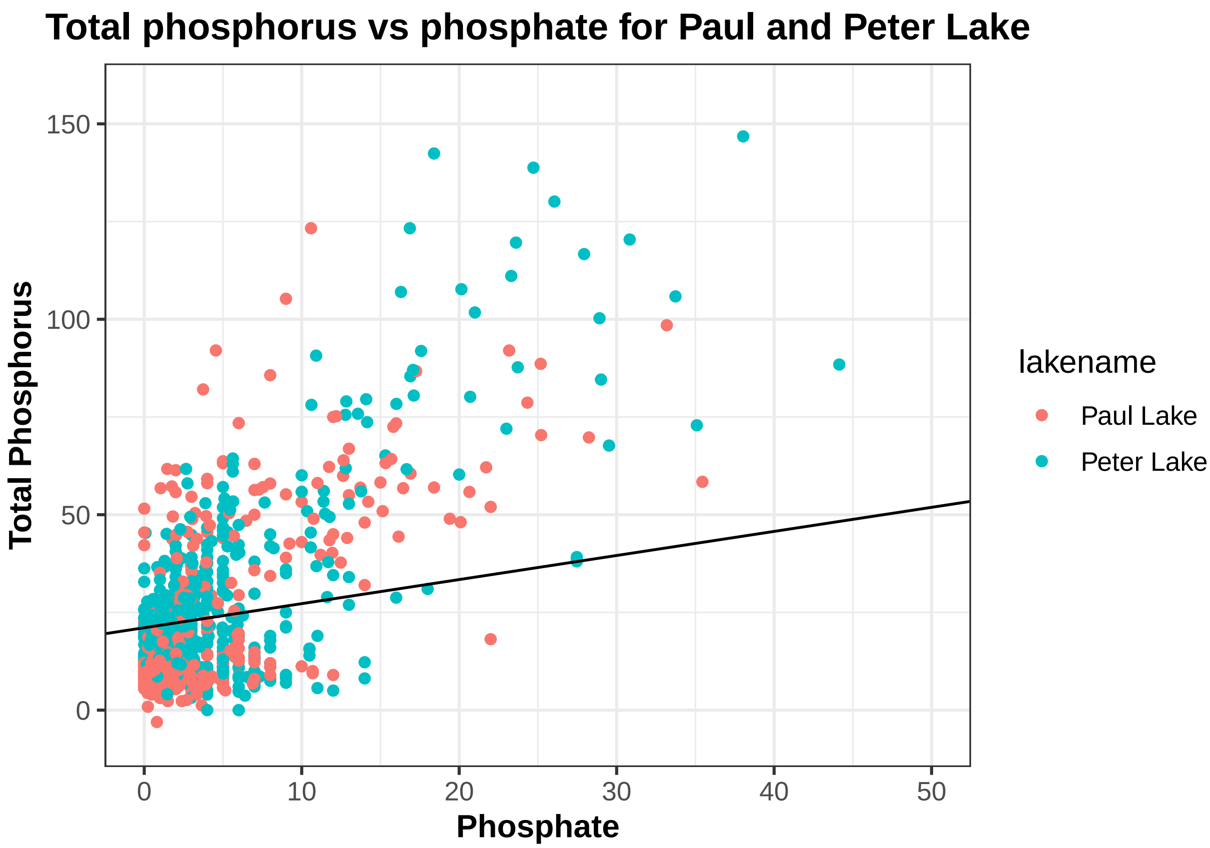
Create graphs

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()`).

```
#4
#----Couldn't create line of best of fit through geom_line or geom_abline----#
LinMod <- lm(tp_ug~po4, data = PeterPaul_processed)

#----get intercept and slope value
coeff <- coefficients(LinMod)
intercept <- coeff[1]
slope <- coeff[2]

#----Create plot---#
ggplot(data = PeterPaul_processed) +
  geom_point(aes(x = po4, y = tp_ug, color = lakename)) +
#----Change axis title---#
  xlab("Phosphate") +
  ylab("Total Phosphorus") +
#----Add plot title----#
  ggtitle("Total phosphorus vs phosphate for Paul and Peter Lake") +
#----Adjust axis scale----#
  xlim(0, 50) +
  geom_abline(intercept = intercept, slope = slope, color = "black")
```

```
## Warning: Removed 21947 rows containing missing values (‘geom_point()‘).
```

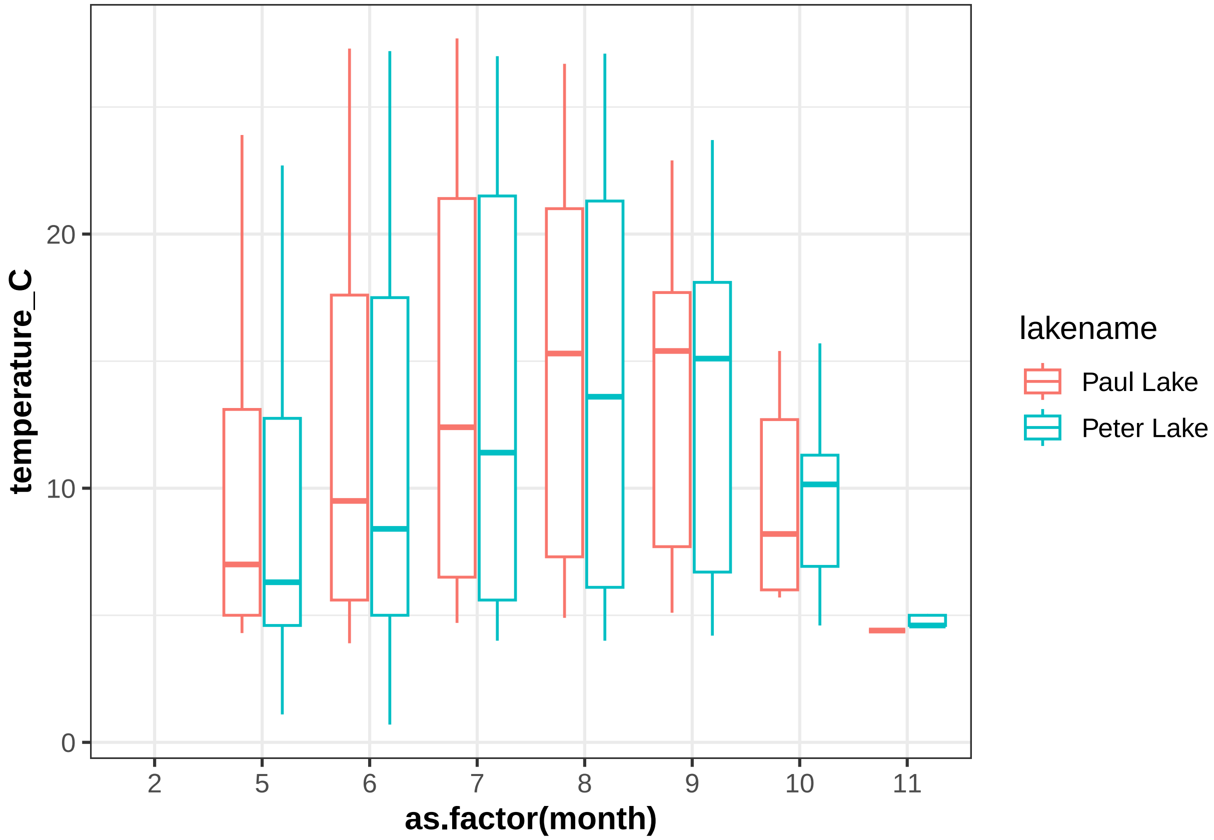


5. [NTL-LTER] Make three separate boxplots of (a) temperature, (b) TP, and (c) TN, with month as the x axis and lake as a color aesthetic. Then, create a cowplot that combines the three graphs. Make sure that only one legend is present and that graph axes are aligned.

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

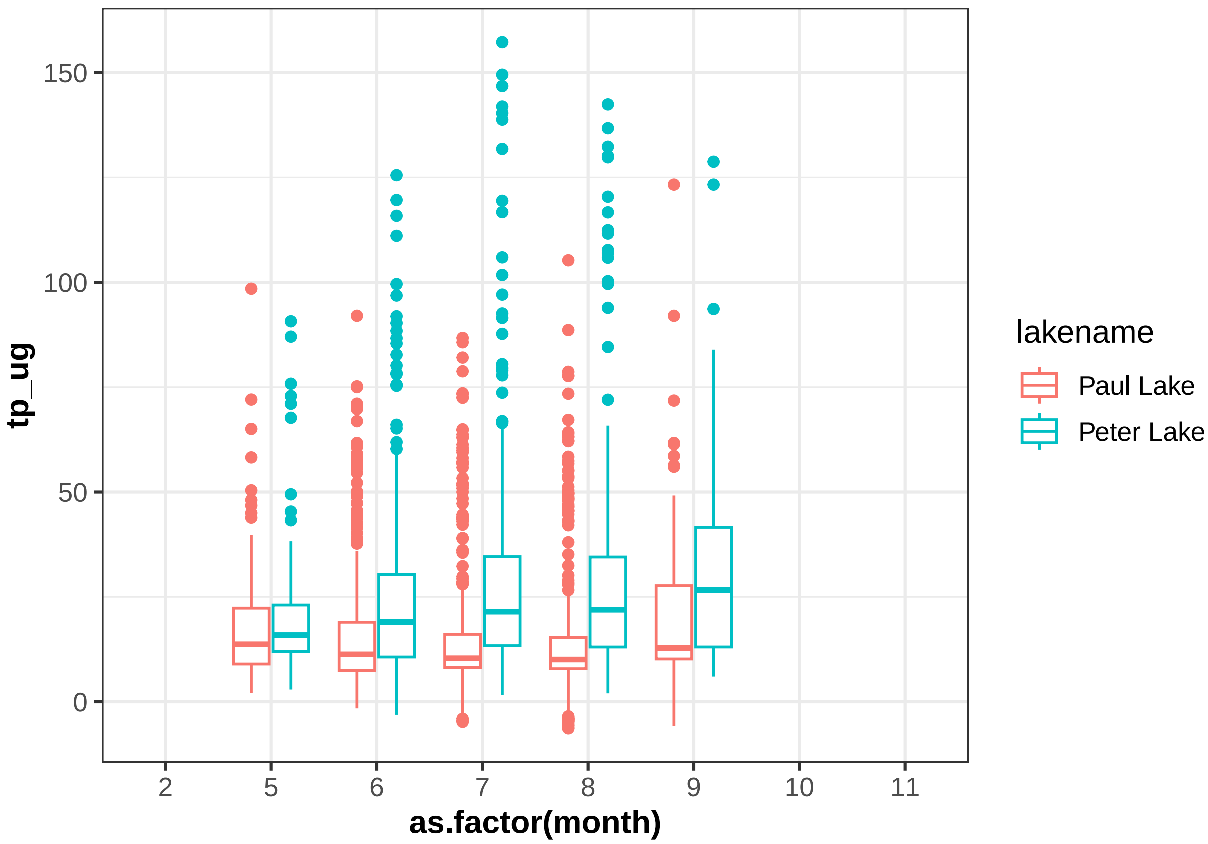
```
#5
#-----5a. boxplot of temperature------#
tempPlot <- ggplot(data = PeterPaul_processed)  +
  geom_boxplot(aes(x = as.factor(month), y = temperature_C,
                   color = lakename))
tempPlot
```

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



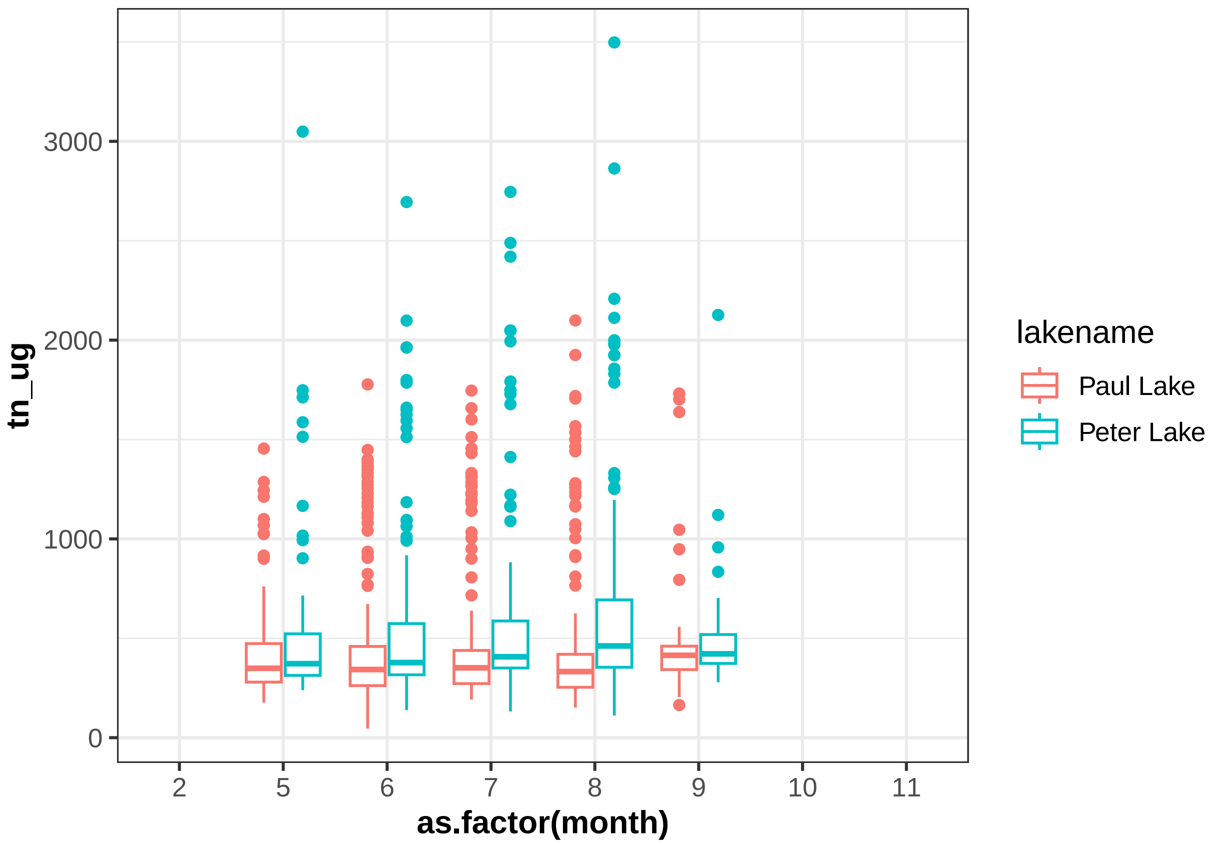

```
#-----5b. boxplot of TP------#
tpPlot <- ggplot(data = PeterPaul_processed) +
  geom_boxplot(aes(x = as.factor(month), y = tp_ug,
                   color = lakename))
tpPlot
```

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




```
#-----5c. boxplot of TN------#
tnPlot <- ggplot(data = PeterPaul_processed) +
  geom_boxplot(aes(x = as.factor(month), y = tn_ug,
                   color = lakename))
tnPlot
```

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



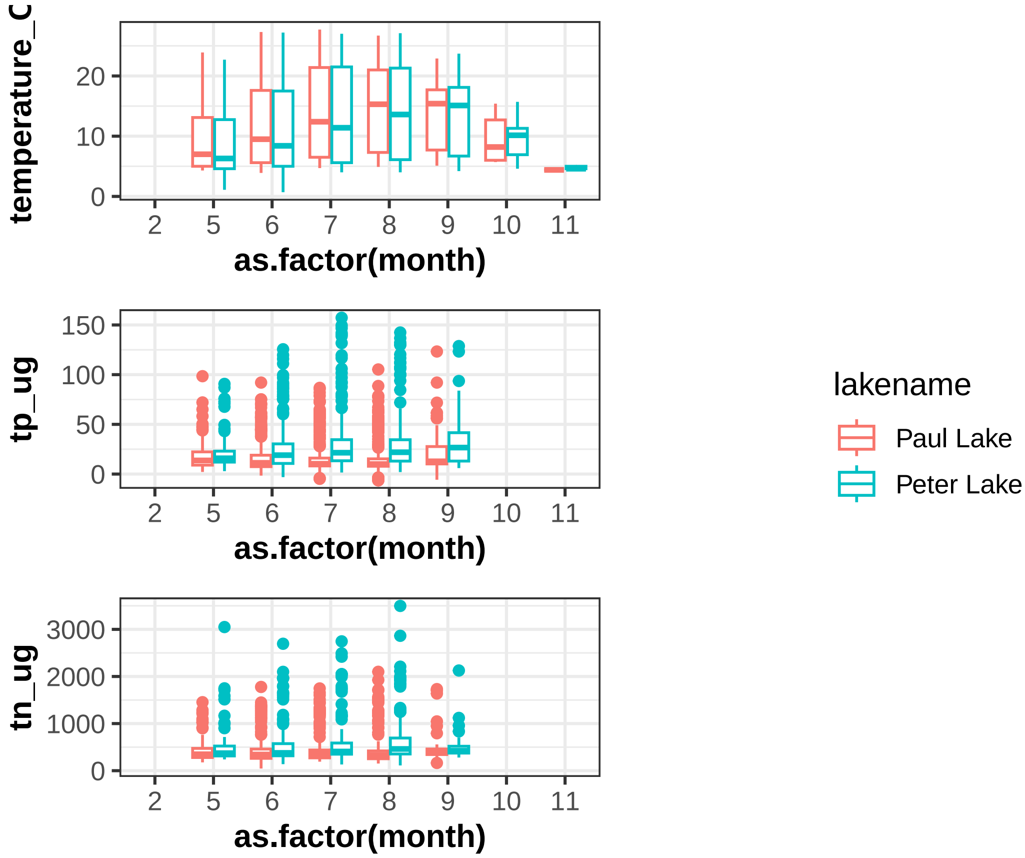

```
#------Remove legends from individual plots-----#
prow <- plot_grid(tempPlot + theme(legend.position = "none"),
                  tpPlot + theme(legend.position = "none"),
                  tnPlot + theme(legend.position = "none"),
                  nrow = 3,
                  axis = "b",
                  align = "v")
```

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

```
#------Get legend from one plot------#
legend <- get_legend(tempPlot)
```

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

```
#------Combine plots into one plot-----#
plot_grid(prow, legend)
```

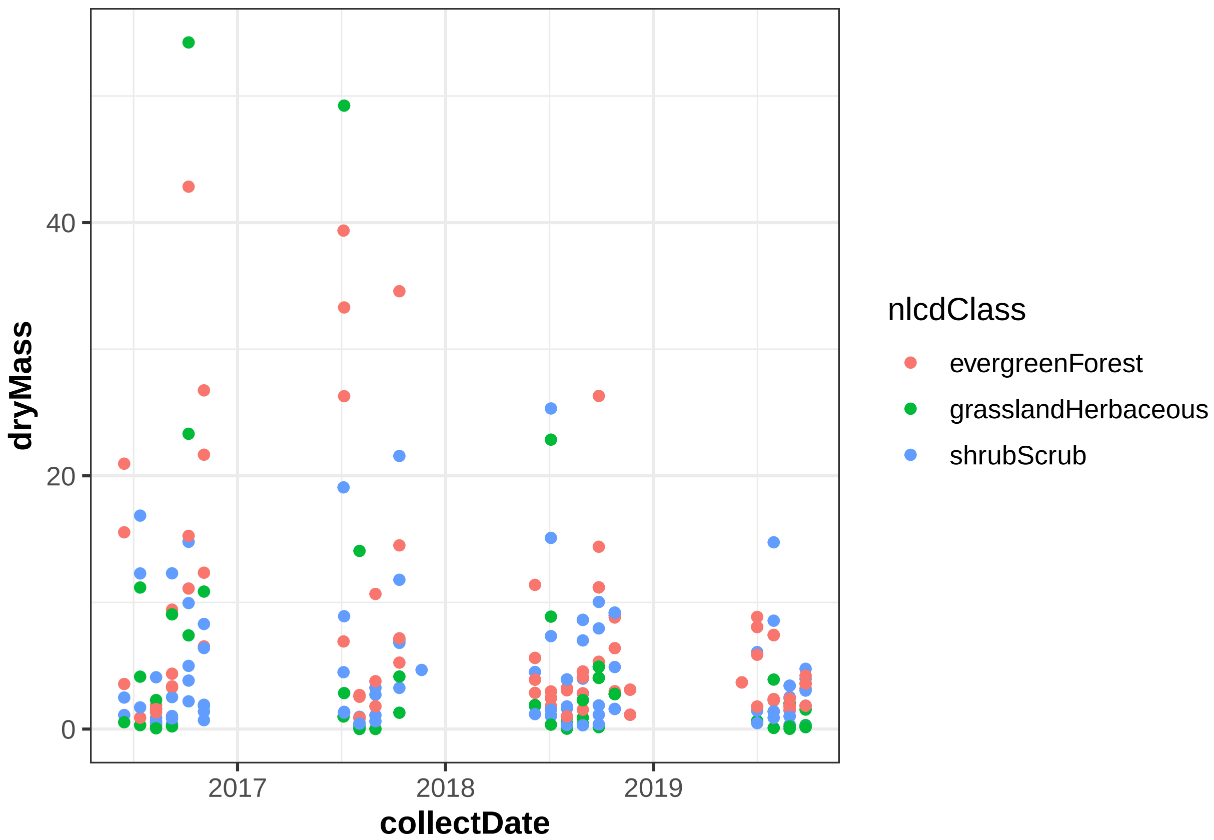


Question: What do you observe about the variables of interest over seasons and between lakes?

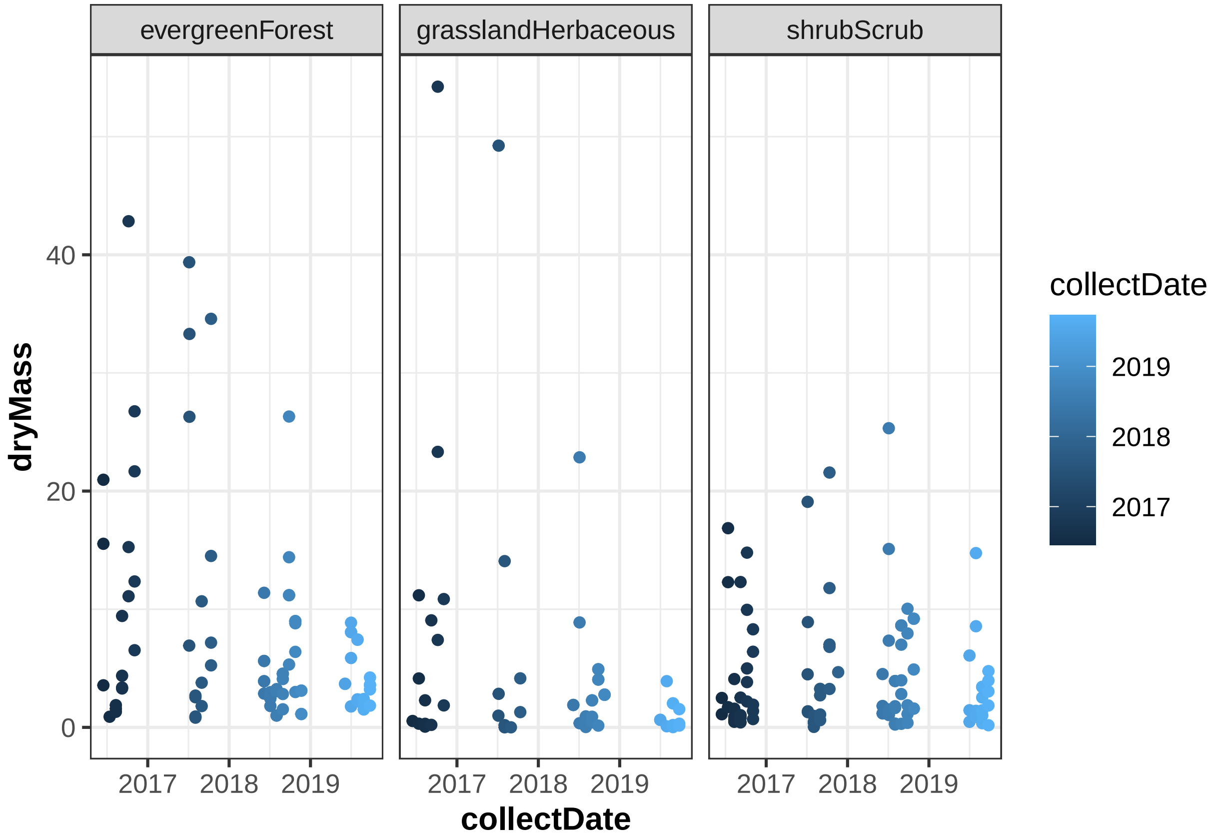
> Answer: There is a positive correlation between the temperature and the concentration levels of tp and tn. During the summer when temperatures are higher, tp and tn concentrations are also higher. Between Paul and Peter Lake, the tp and tn concentrations are higher in Peter lake. The temperatures between Paul and Peter Lake are nearly the same.

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 by functional group-----#
ggplot(data = litter_processed %>%
         filter(functionalGroup == "Needles")) +
         geom_point(aes(x = collectDate, y = dryMass, color = nlcdClass, fill = nlcdClass))
```




```
#7
#-----Create same plot with facets-----#
ggplot(data = litter_processed %>%
         filter(functionalGroup == "Needles")) +
         geom_point(aes(x = collectDate, y = dryMass,
                        color = collectDate)) +
                        facet_wrap(vars(nlcdClass))
```



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

> Answer: Plot 7 is more effective in this case, because the facet feature allows us to more clearly distinguish between the dry mass of needles by nlcdClass. For example, it is easier to determine that shrub scrub is produces the least amount needles across all years that data were collected.