Assignment 5: Data Visualization

Chunyi Xu

Fall 2024

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
#Noting down codes to install packages
#install.packages("tidyverse");
#install.packages("lubridate");
#install.packages("here")
#install.packages("cowplot")

#Loading necessary packages (tidyverse, lubridate, here)
library (tidyverse)
```

```
## -- 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   3.5.0     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
```

```
library (lubridate)
library (here)
```

```
## Warning: package 'here' was built under R version 4.3.3
```

```
## here() starts at /Users/xuchunyi/Desktop/EDA_Spring2025/EDA_Spring2025
```

```
library (cowplot)
```

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

```
#Checking working directory is the project folder
here()
```

```
## [1] "/Users/xuchunyi/Desktop/EDA_Spring2025/EDA_Spring2025"
```

```
#Reading in the NTL-LTER processed data files for nutrients and chemistry/physics
# for Peter and Paul Lakes
Lake.Chem.Nutrient <- read.csv(
  file = here("./Data/Processed_KEY/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv"),
  stringsAsFactors = TRUE)

#Reading in the processed data file for the Niwot Ridge litter dataset
Niwot.Litter <- read.csv(
  file = here("./Data/Processed_KEY/NEON_NIWO_Litter_mass_trap_Processed.csv"),
  stringsAsFactors = TRUE)

#2
#Checking if R is reading dates as date format
#For Lake.Chem.Nutrient
class (Lake.Chem.Nutrient$sampledate)
```

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

```
#The result is factor
# Changing the Date column to be date objects
Lake.Chem.Nutrient$sampledate <- as.Date(Lake.Chem.Nutrient$sampledate, format = "%Y-%m-%d")

#For Niwot.Litter
class (Niwot.Litter$collectDate)
```

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

```
#The result is factor
# Changing the Date column to be date objects
Niwot.Litter$collectDate <- as.Date(Niwot.Litter$collectDate, 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

```
#3
#Loading the necessary packages
library (ggplot2)
#Building my default theme
default.theme <- theme_classic(base_size = 14) +
  theme(plot.background = element_rect(color='white',fill='white'),
        plot.title = element_text(color='black', size = 14),
        axis.text = element_text(color = 'black',size = 10),
        panel.grid.minor = element_line(size = 0.5),
        panel.grid.major = element_line(size = 0.5),
        legend.background = element_rect(color='white', fill = 'white'),
        legend.position = "right",
        legend.title = element_text(color='black',size=12),
        legend.text = element_text(size = 8))
```

```
## Warning: The ‘size‘ argument of ‘element_line()‘ is deprecated as of ggplot2 3.4.0.
## i Please use the ‘linewidth‘ argument instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

```
#Setting the theme as the default theme
theme_set(default.theme)
```

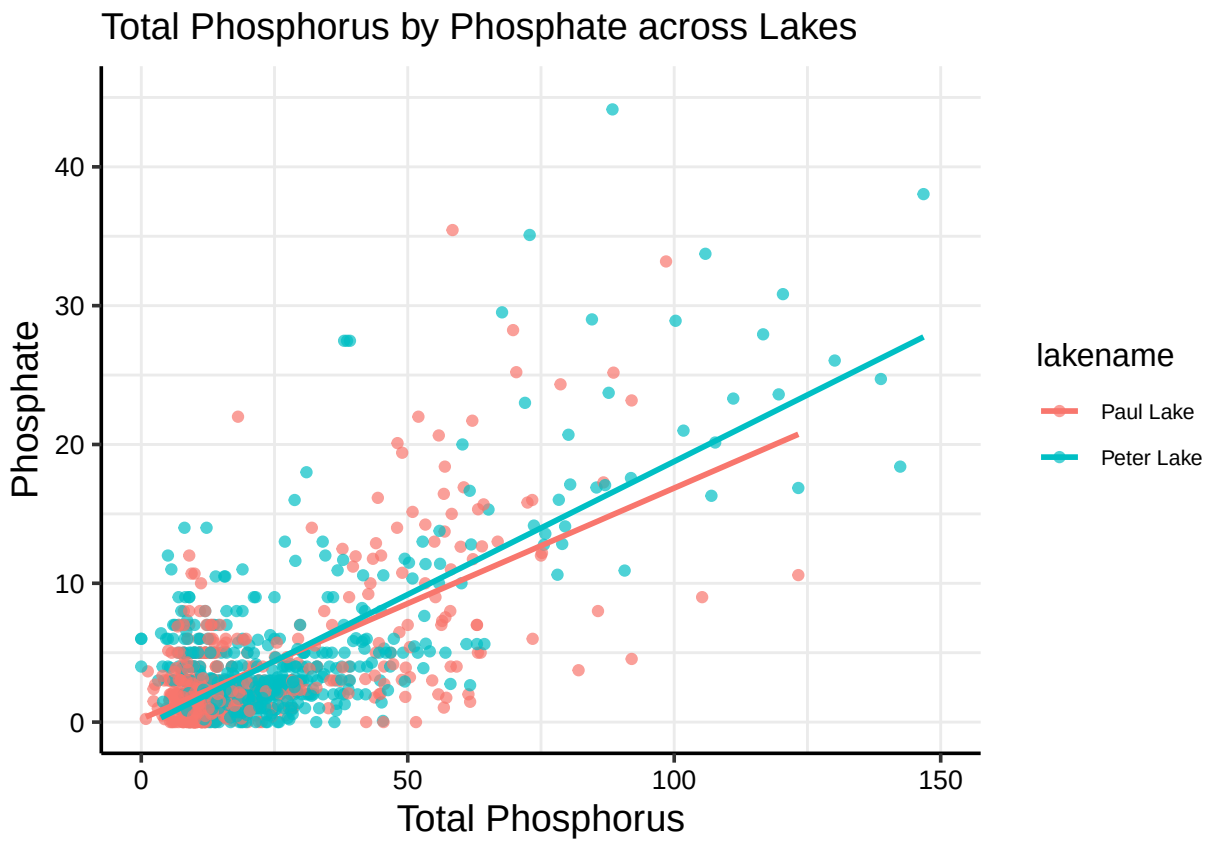
Create graphs

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()`).

```r
#4
# Plotting total phosphorus by phosphate across two lakes
Lake.tp_ug.po4 <-
  ggplot(Lake.Chem.Nutrient, aes(x = tp_ug, y = po4, color = lakename)) +
  geom_point(alpha = 0.7, size = 1.5) +
#Adding lines of best fit
  geom_smooth(method = lm, se=FALSE)+
#Adjusting the axes to hide extreme values
  xlim(0, 150) +
  ylim(0, 45)+
  xlab("Total Phosphorus")+
  ylab("Phosphate")+
  labs (title = "Total Phosphorus by Phosphate across Lakes")
print(Lake.tp_ug.po4)
```

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

## Warning: Removed 21948 rows containing non-finite outside the scale range
## (`stat_smooth()`).

## Warning: Removed 21948 rows containing missing values or values outside the scale range
## (`geom_point()`).

## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`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.

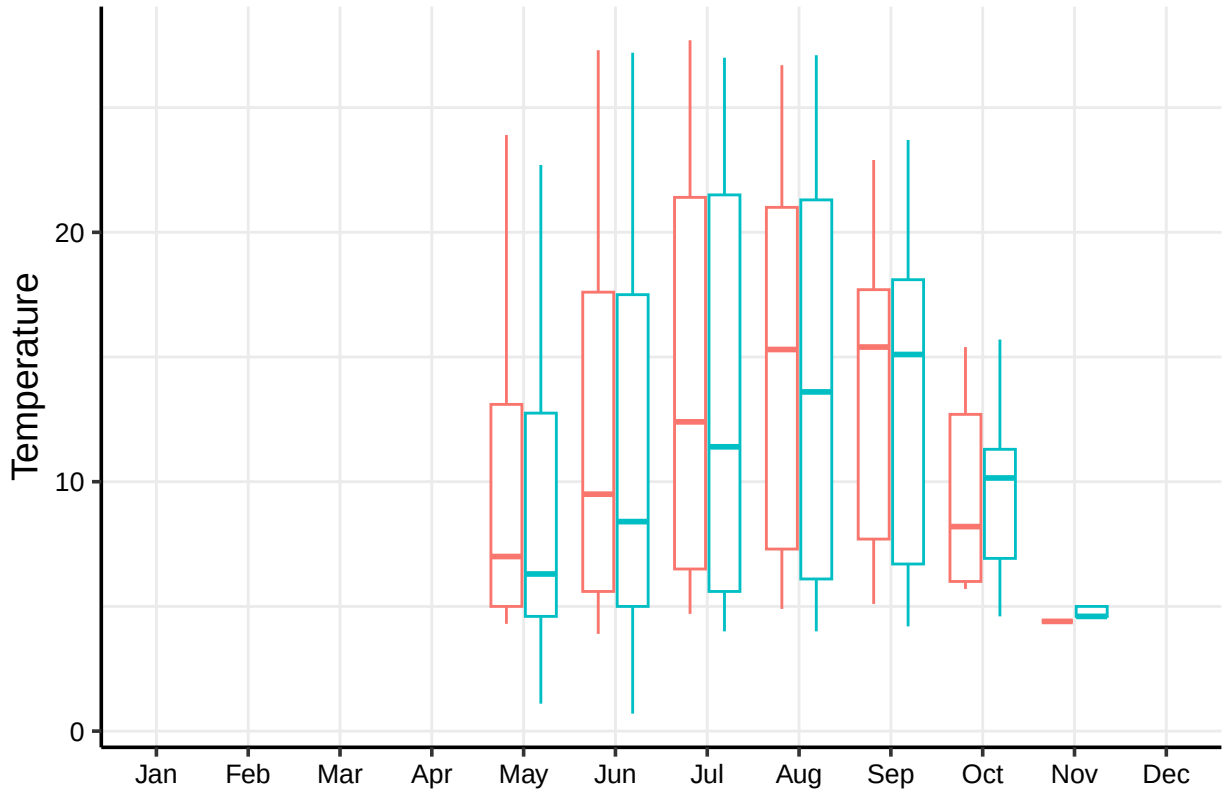
Tips: * Recall the discussion on factors in the lab section as it may be helpful here. * Setting an axis title in your theme to `element_blank()` removes the axis title (useful when multiple, aligned plots use the same axis values) * Setting a legend's position to "none" will remove the legend from a plot. * Individual plots can have different sizes when combined using `cowplot`.

```
#5

#Checking the class of the month, and the result is integer
class(Lake.Chem.Nutrient$month)
```

```
## [1] "integer"
```

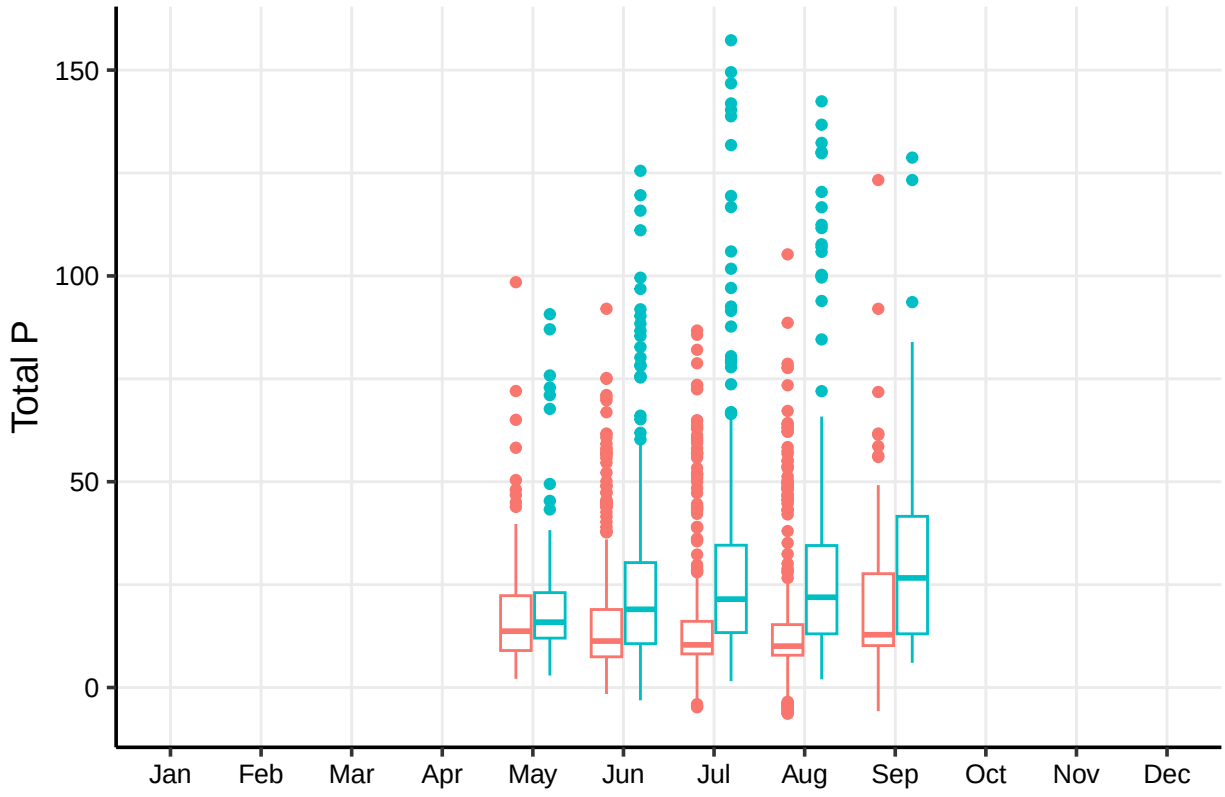
```
# Changing the month to be a factor
Lake.Chem.Nutrient$month <- factor(Lake.Chem.Nutrient$month,levels=1:12,
                                   labels=month.abb)

#Creating a boxplot for temperature
Box.Temp <-
  ggplot(Lake.Chem.Nutrient, aes(x = month, y = temperature_C, color = lakename))+
  geom_boxplot() +
  theme (legend.position = "none")+
  scale_x_discrete(drop=F,name = "")+
  ylab("Temperature")
print (Box.Temp)
```

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



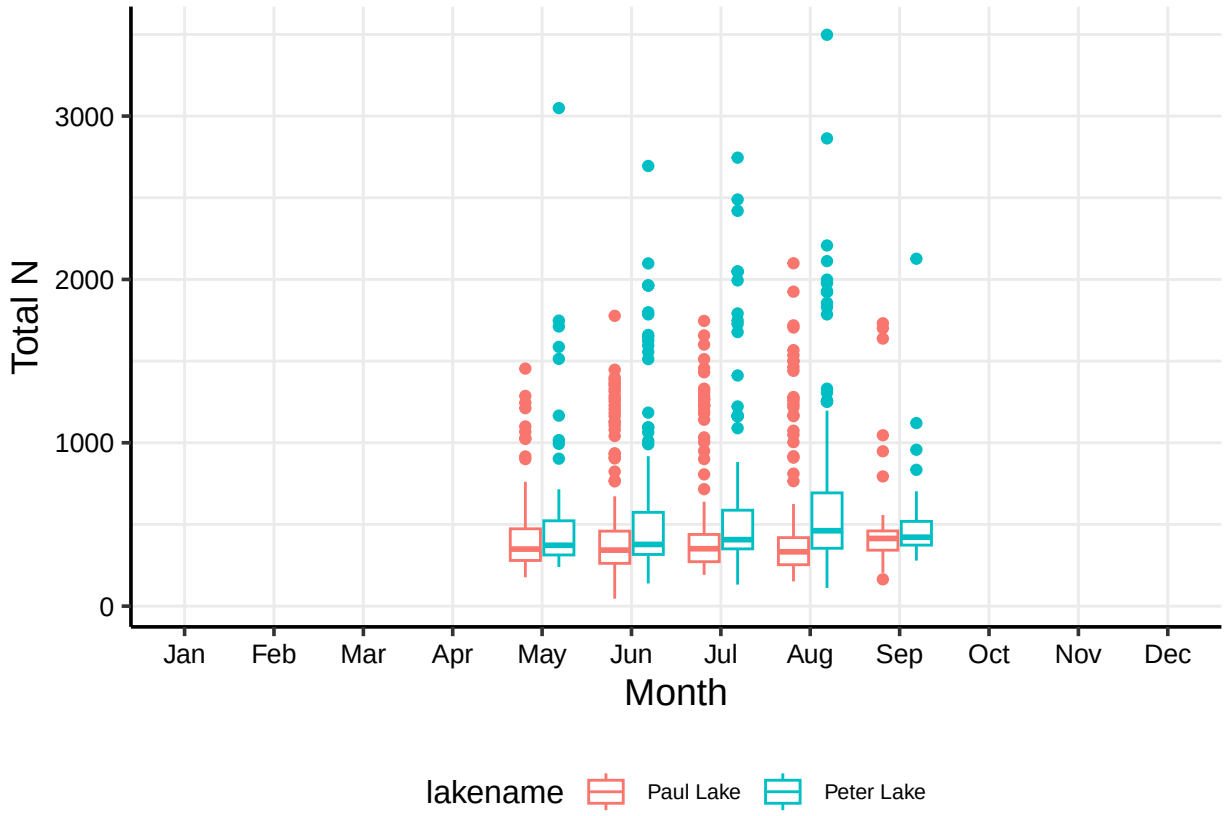

```
#Creating a boxplot for TP
Box.TP <-
  ggplot(Lake.Chem.Nutrient, aes(x = month, y = tp_ug, color = lakename))+
  geom_boxplot() +
  theme (legend.position = "none")+
  scale_x_discrete(drop=F,name = "")+
  ylab("Total P")
print (Box.TP)
```

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




```
#Creating a boxplot for TN
Box.TN <-
  ggplot(Lake.Chem.Nutrient, aes(x = month, y = tn_ug, color = lakename))+
  geom_boxplot() +
  theme (legend.position = "bottom")+
  scale_x_discrete(drop=F,name = "Month")+
  ylab("Total N")
print (Box.TN)
```

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

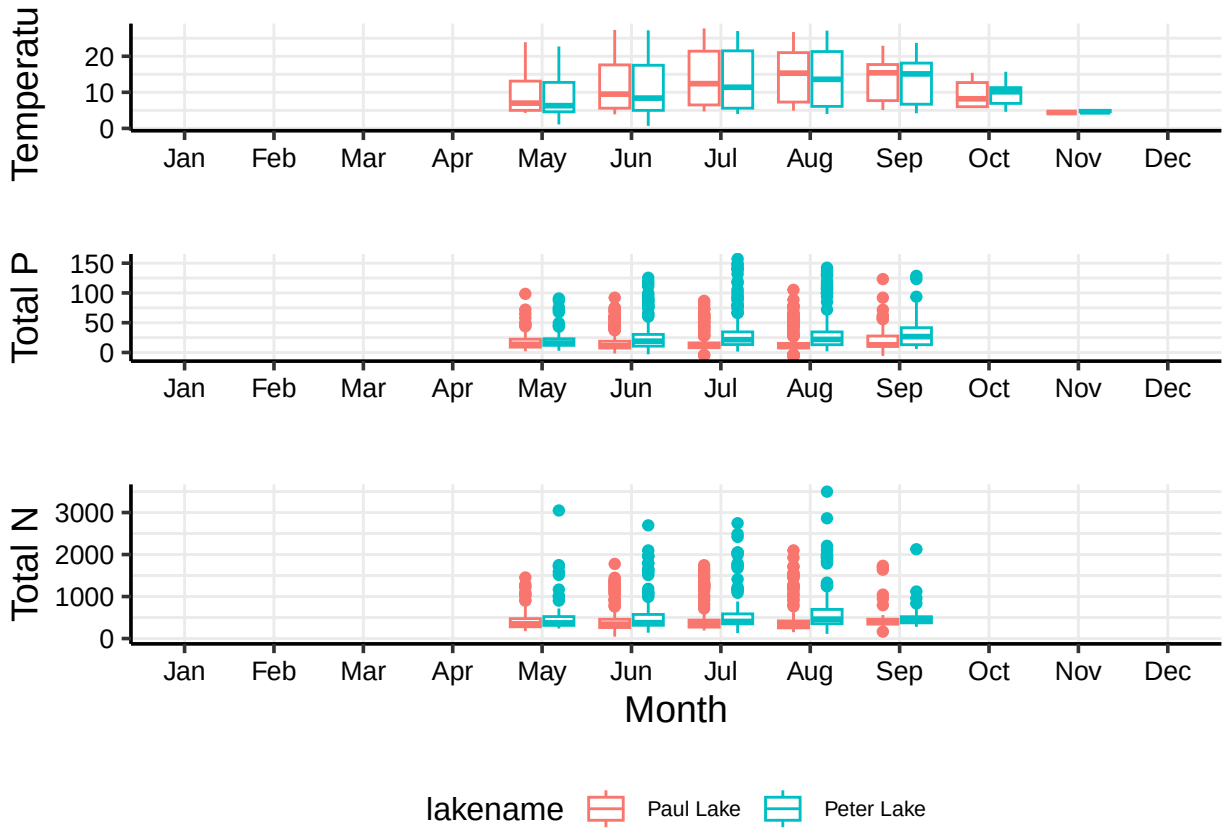



```
#Creating a cowplot that combines the three graphs
plot_grid(Box.Temp, Box.TP, Box.TN, nrow = 3, align = 'v', rel_heights = c(0.8, 0.8, 1.4))
```

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



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

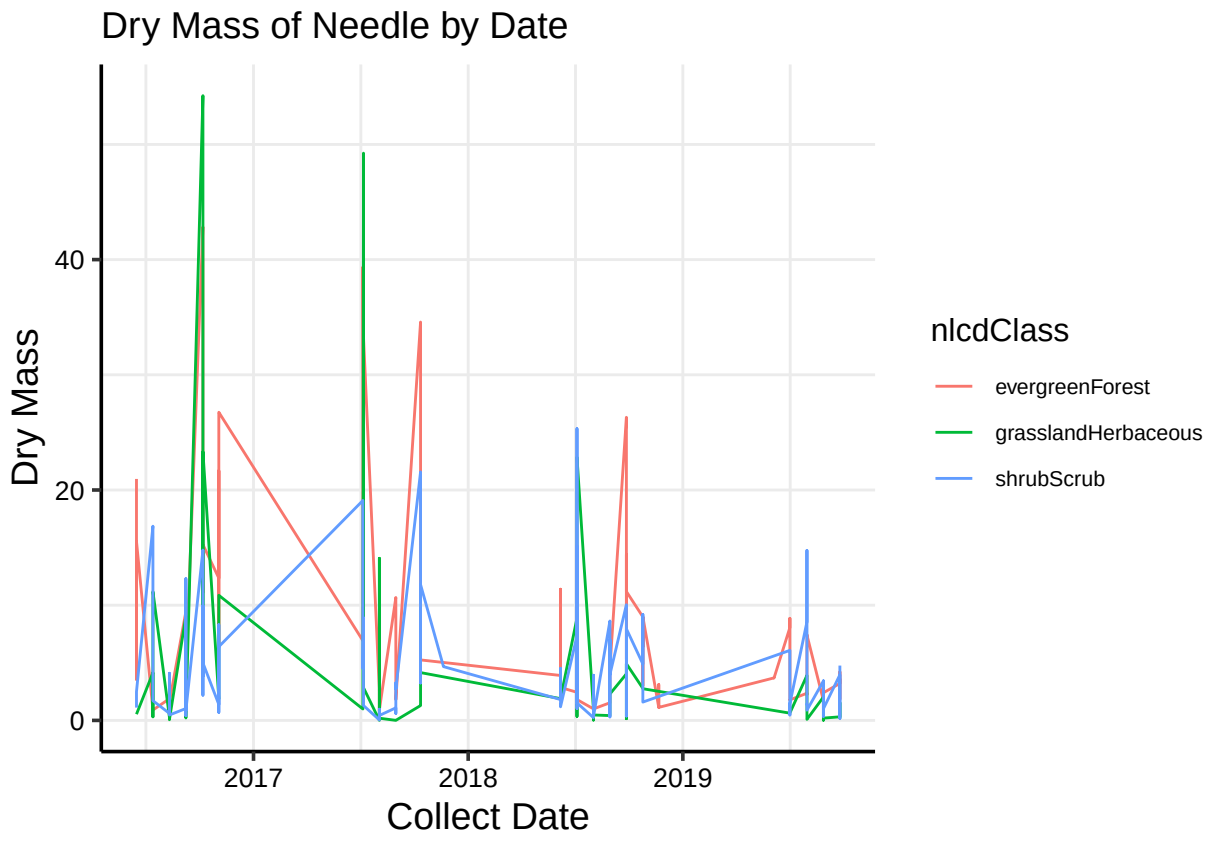
> Answer: For all variables of interest, our sample date mainly covers the summer and autumn. The distribution of temperature is more dispersed than total Phosphorus and total Nitrogen, while the distribution of total Phosphorus is more dispersed than total Nitrogen too. For temperature, both lakes' temperatures increase first and then decrease over seasons. During the summer and autumn, the temperature of Paul Lake is higher than that of Peter Lake. However, in late autumn and early winter (October and November), Paul Lake's temperature is higher than Peter Lake's Temperature. In terms of total Phosphorus, the mean total Phosphorus of Paul Lake decreases first but then increases in September, while the mean total Phosphorus of Peter Lake always increases and is higher than that of Paul Lake throughout the sampling months. For total Nitrogen, Paul Lake's total Nitrogen does not fluctuate a lot throughout the sampling months, although it increases a bit in September. Peter Lake's total Nitrogen is always higher than that of Paul Lake, and it gradually increases from May to August but decreases 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)

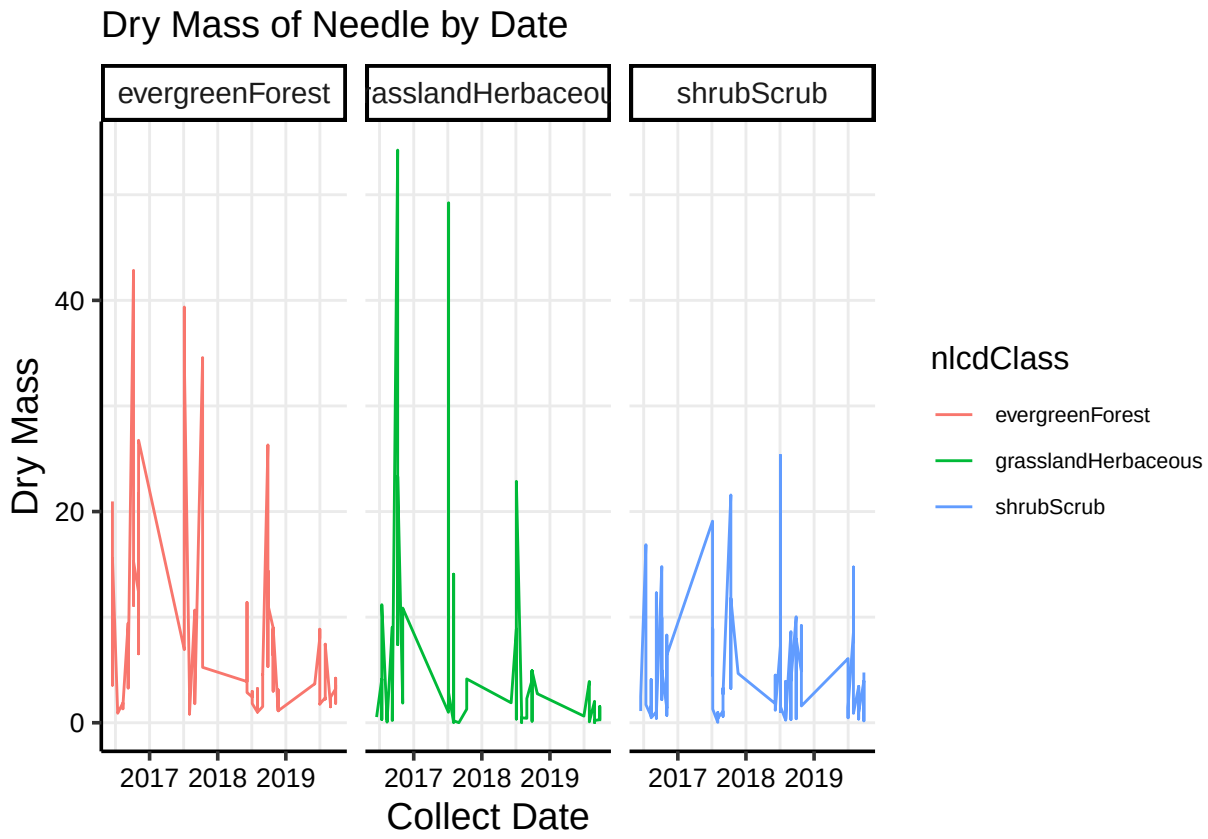
7. [Niwot Ridge] Now, plot the same plot but with NLCD classes separated into three facets rather than separated by color.

```
#6
#Plotting the dry mass of needle litter by date and separate by NLCD class with a color aesthetic
Needle.Sub <-
  Niwot.Litter %>%
```

```
#Filtering a subset of the litter dataset
  filter (functionalGroup == "Needles")%>%
  ggplot (aes (x=collectDate, y=dryMass,color=nlcdClass)) +
  geom_line()+
  labs (title = "Dry Mass of Needle by Date")+
  xlab("Collect Date")+
  ylab ("Dry Mass")
print (Needle.Sub)
```




```
#7
#Plotting the same plot but with NLCD classes separated into three facets
Needle.Sub.Facet <-
  Niwot.Litter %>%
#Filtering a subset of the litter dataset
  filter (functionalGroup == "Needles")%>%
  ggplot (aes (x=collectDate, y=dryMass,color=nlcdClass)) +
  geom_line()+
  facet_wrap(vars(nlcdClass), ncol = 3)+
  labs (title = "Dry Mass of Needle by Date")+
  xlab("Collect Date") +
  ylab ("Dry Mass")
print (Needle.Sub.Facet)
```



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

Answer:The plot in Q7 is more effective. Since we want to observe how the dry mass changes over time, a line graph will be more appropriate for visualization. In Q6, the lines for three NLCD classes overlap with each other, making it challenging to differentiate between them and observe any trends. The whole graph in Q6 becomes messy. However, by separating the line graphs of the three NLCD classes into three facets, each graph is clearer and easier to read trends.