Assignment 5: Data Visualization

Vicky Fong

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.

```r
#1
library(tidyverse); library(lubridate); library(here); library(cowplot)
```

```
## -- 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   3.5.1     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
## here() starts at /home/guest/EDE_Fall2024
##
##
## Attaching package: ’cowplot’
##
##
## The following object is masked from ’package:lubridate’:
##
##     stamp
```

```
getwd()
```

```
## [1] "/home/guest/EDE_Fall2024"
```

```
ntl <- read.csv(
  here('Data','Processed_KEY','NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv')
  )

neon <- read.csv(
  here('Data','Processed_KEY','NEON_NIWO_Litter_mass_trap_Processed.csv')
  )

#2
class(ntl$sampledate)
```

```
## [1] "character"
```

```
ntl$sampledate <- ymd(ntl$sampledate)
class(ntl$sampledate)
```

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

```
class(neon$collectDate)
```

```
## [1] "character"
```

```
neon$collectDate <- ymd(neon$collectDate)
class(neon$collectDate)
```

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

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
my_theme <-  theme(
    plot.title = element_text(face = "bold", size = 12),
    panel.background = element_rect(fill = "white", colour = NA),
    panel.border = element_rect(fill = NA, colour="grey50"),
    panel.grid.major = element_line(colour = "black", size = 0.01),
    panel.grid.minor =  element_line(colour = "black", size = 0.01),
    axis.text = element_text(size = 10),
    axis.ticks = element_blank(),
  )
```

```
## 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.
```

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()`).

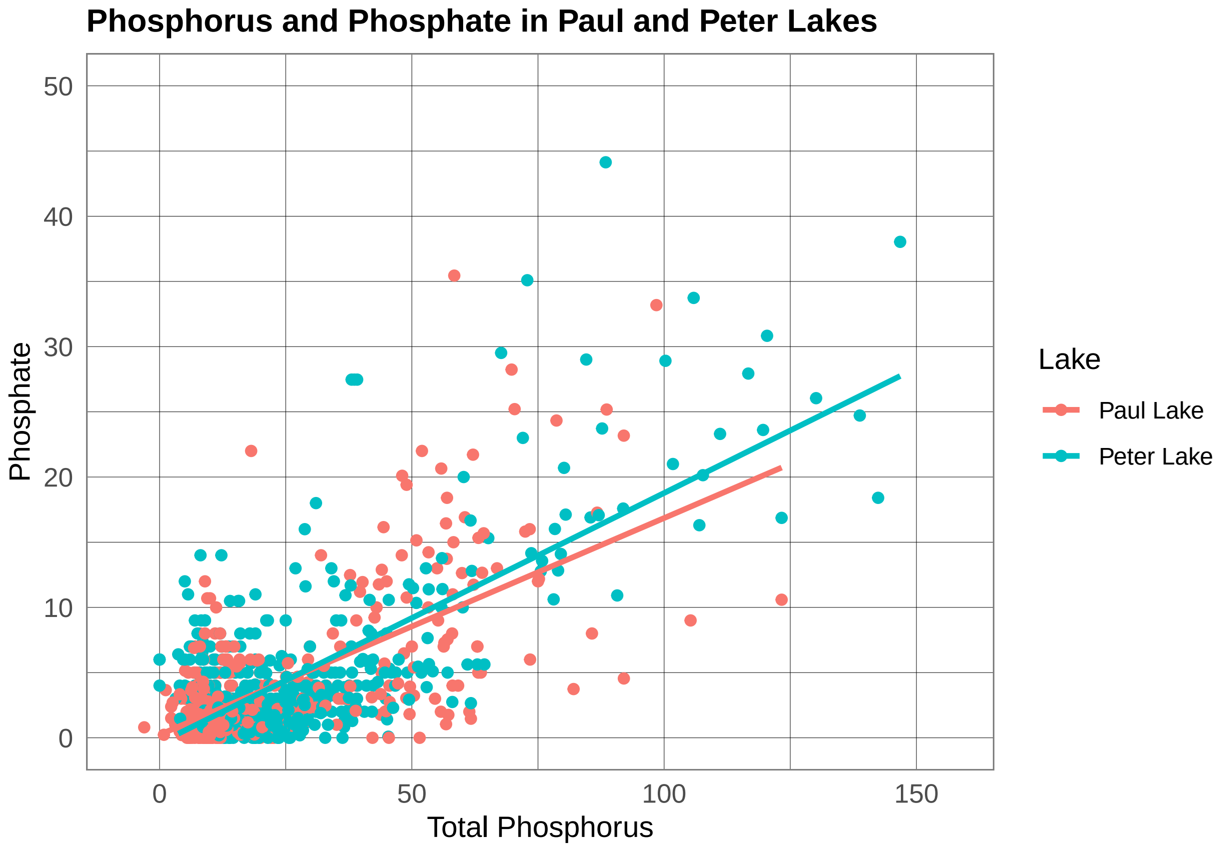
```
#4
ntl %>%
  ggplot(aes(x = tp_ug, y = po4, color = lakename)) +
  geom_point() +
  geom_smooth(method=lm, se=FALSE) +
  labs(x="Total Phosphorus", y="Phosphate", color="Lake",
       title="Phosphorus and Phosphate in Paul and Peter Lakes") +
  ylim(0,50) +
  my_theme
```

```
## ‘geom_smooth()‘ using formula = ’y ~ x’
```

```
## Warning: Removed 21947 rows containing non-finite outside the scale range
## (‘stat_smooth()‘).
```

```
## Warning: Removed 21947 rows containing missing values or values outside the scale range
## (‘geom_point()‘).
```

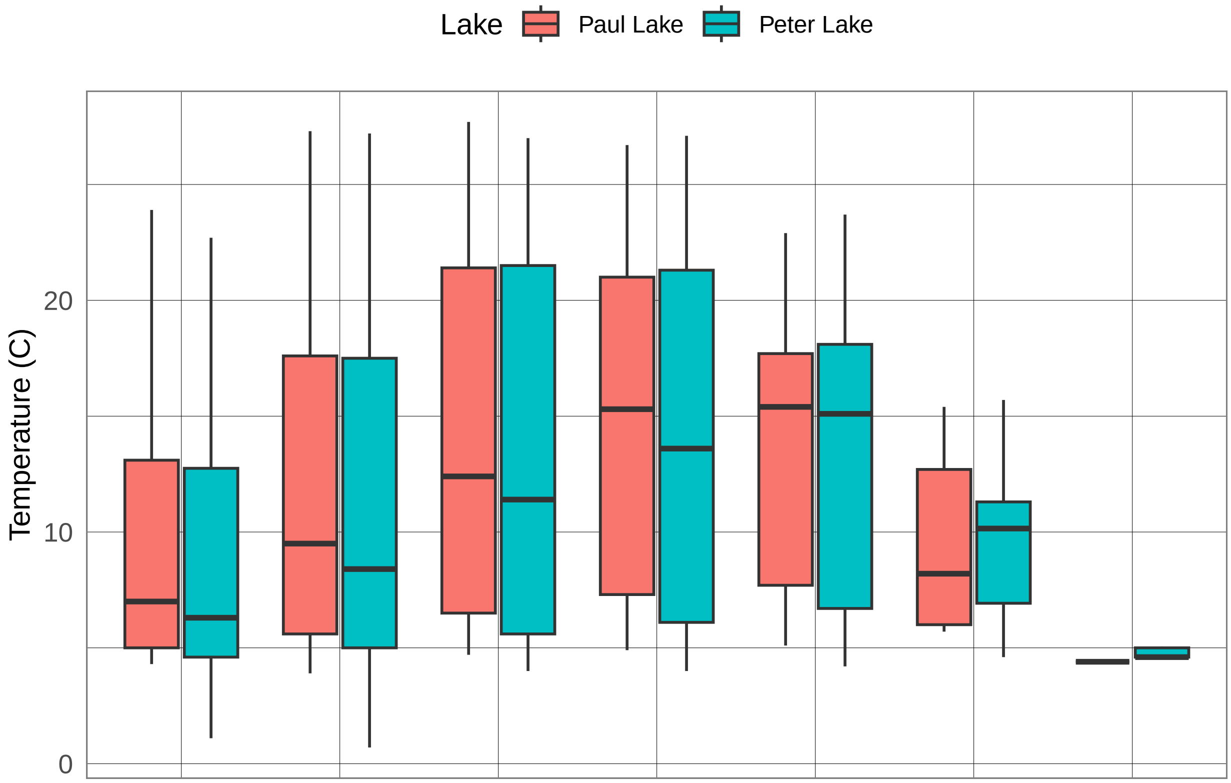
```
## Warning: Removed 4 rows containing missing values or values outside the scale range
## (‘geom_smooth()‘).
```



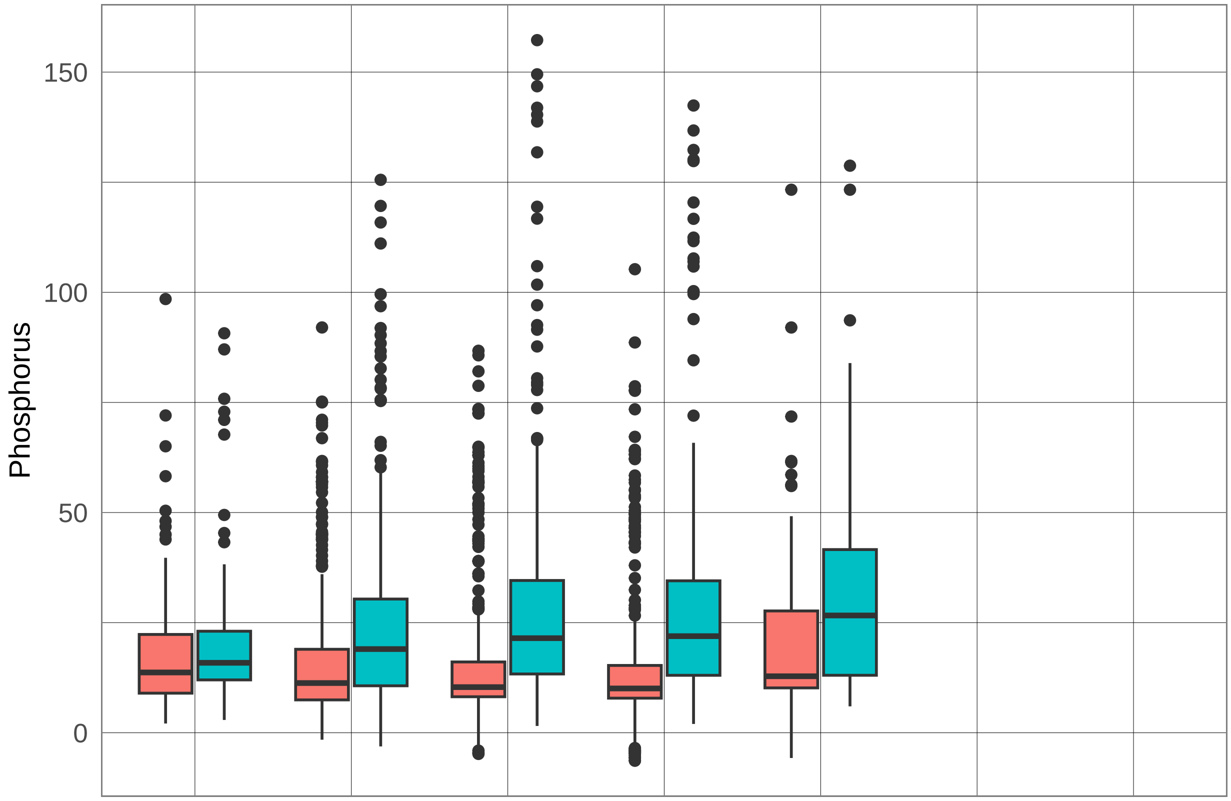
5. [NTL-LTER] Make three separate boxplots of (a) temperature, (b) TP, and (c) TN, with month as the x axis and lake as a color aesthetic. Then, create a cowplot that combines the three graphs. Make sure that only one legend is present and that graph axes are aligned.

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`.

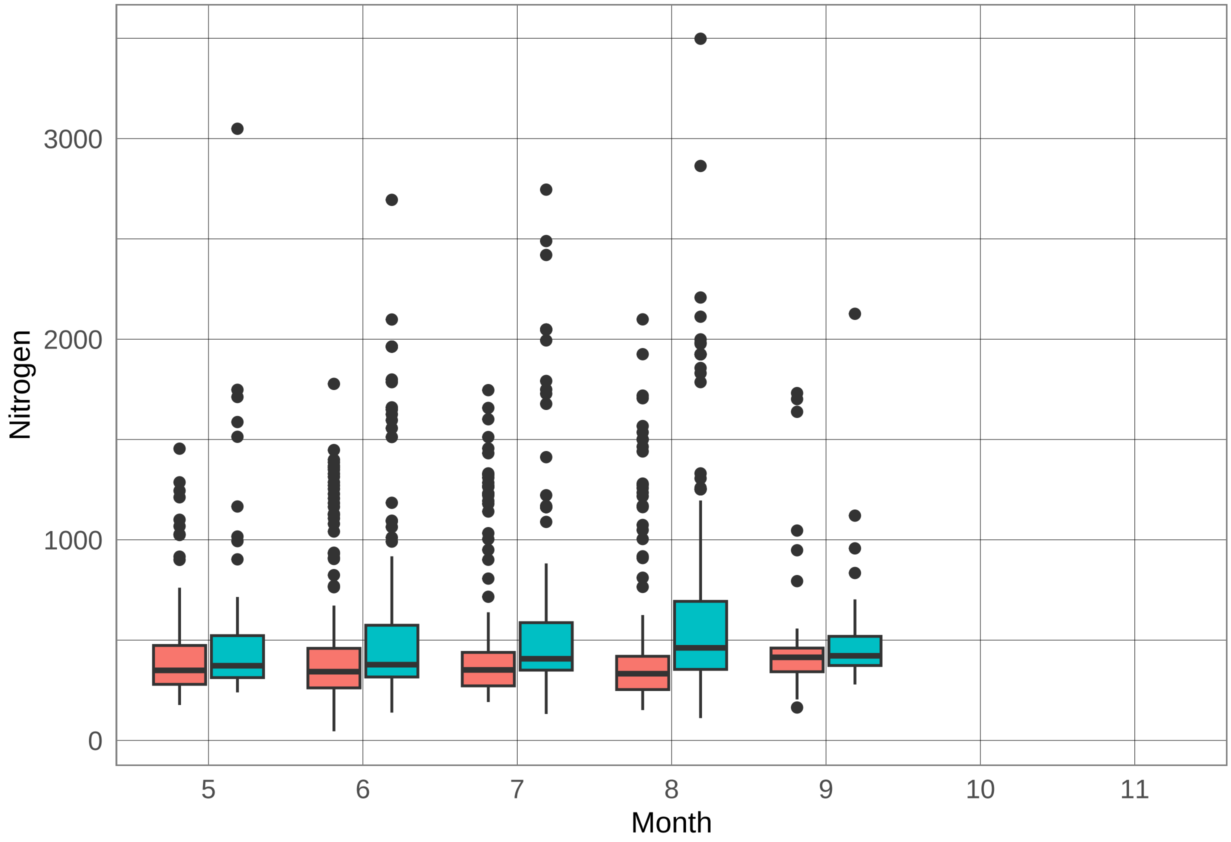
```r
#5
ntl$month <- as.factor(ntl$month)

temp <- ntl %>%
  select(lakename,month,temperature_C) %>%
  na.omit() %>%
  ggplot(aes(x=month, y=temperature_C, fill=lakename))+
  geom_boxplot() +
  labs(x='', y='Temperature (C)', fill='Lake')+
  scale_x_discrete(limits=c('5','6','7','8','9','10','11'), labels=NULL) +
  theme(legend.position = 'top') +
  my_theme
temp
```



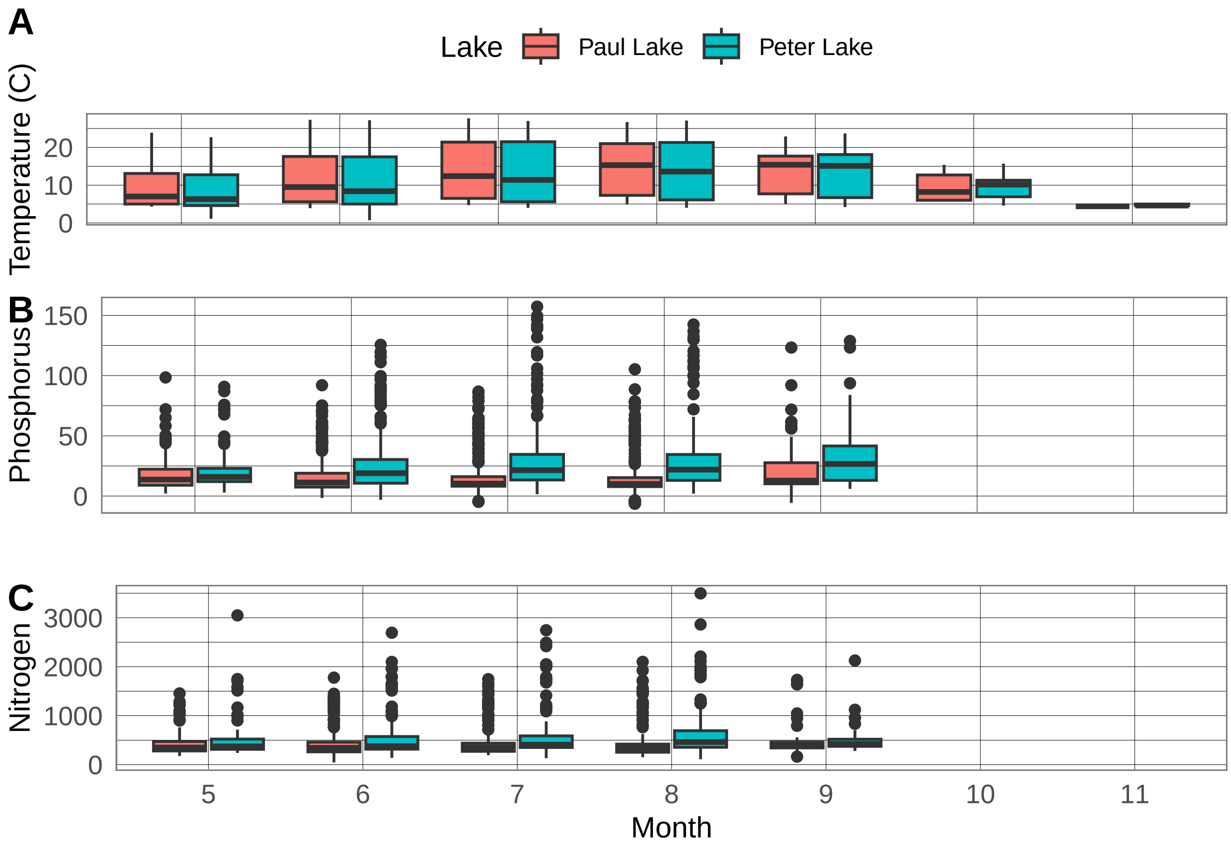
```
tp <- ntl %>%
  select(lakename,month,tp_ug) %>%
  na.omit() %>%
  ggplot(aes(x=month, y=tp_ug, fill=lakename))+
  geom_boxplot() +
  labs(x='', y='Phosphorus', fill='Lake')+
  scale_x_discrete(limits=c('5','6','7','8','9','10','11'), labels=NULL) +
  theme(legend.position = 'none') +
  my_theme
tp
```



```
tn <- ntl %>%
  select(lakename,month,tn_ug) %>%
  na.omit() %>%
  ggplot(aes(x=month, y=tn_ug, fill=lakename))+
  geom_boxplot() +
  labs(x='Month', y='Nitrogen', fill='Lake')+
  scale_x_discrete(limits=c('5','6','7','8','9','10','11')) +
  theme(legend.position = 'none') +
  my_theme
tn
```



```
plot_grid(temp,tp,tn, labels="AUTO", ncol = 1)
```

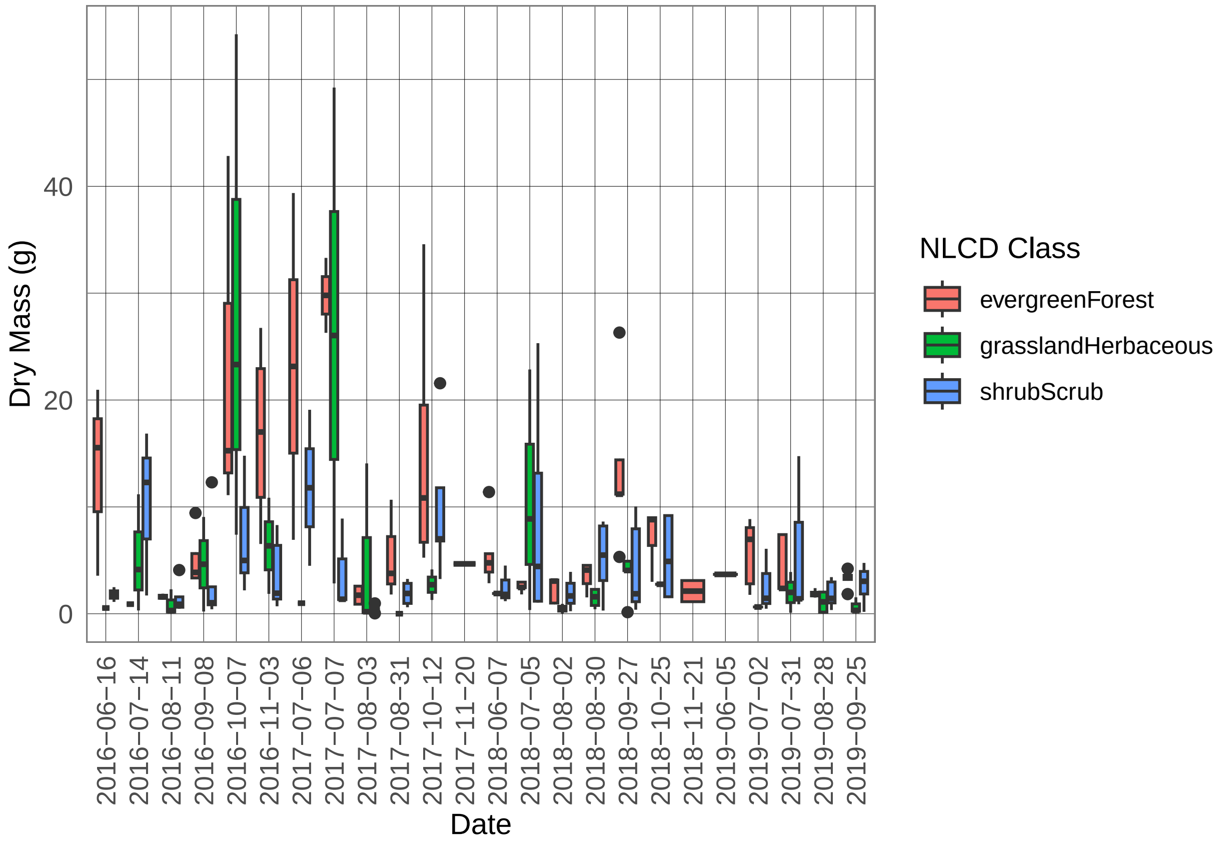


Question: What do you observe about the variables of interest over seasons and between lakes?

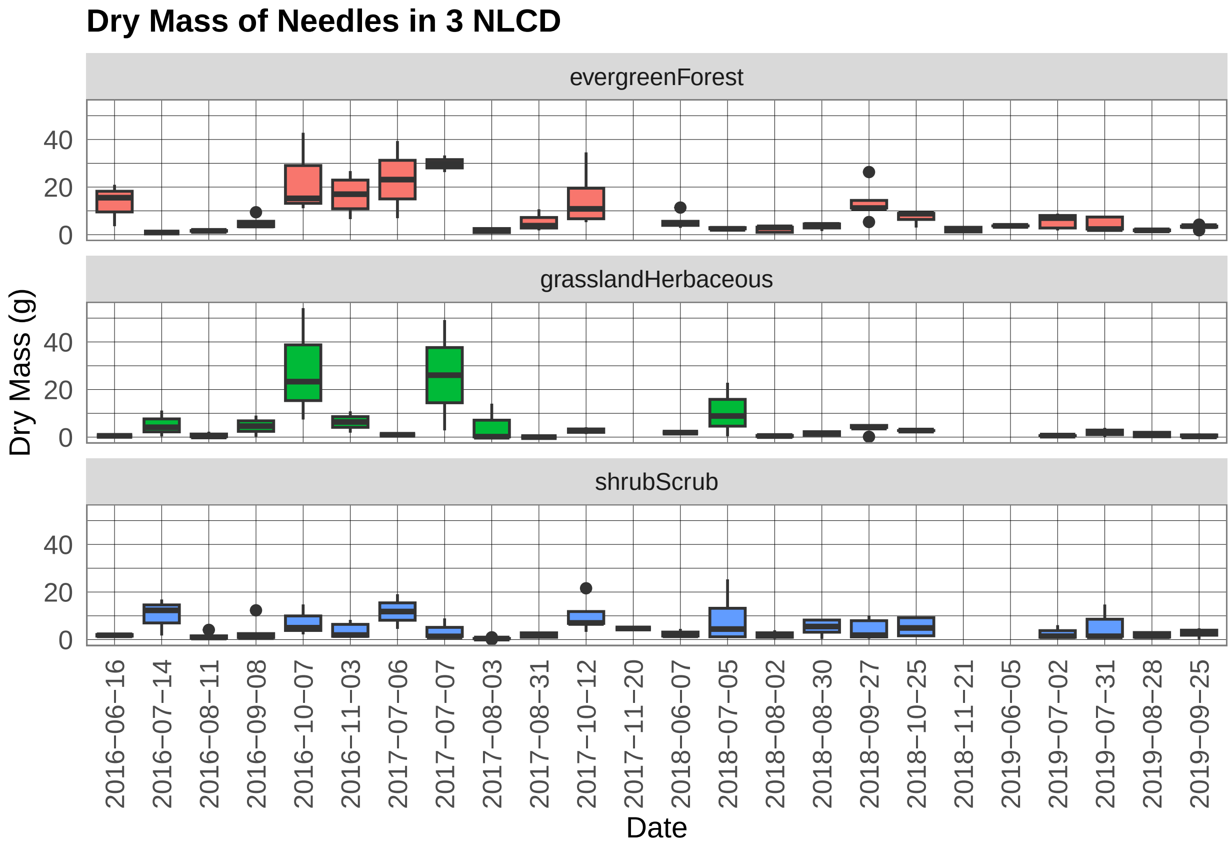
> Answer: Temperature in both lakes increase from May to June, peak in the summer months (July and August), and decrease over the fall months with the lowest temperatures in November. Overall mean phosphorus levels are lower in Paul Lake compared to Peter Lake with little seaonal variation. Mean phosphorus levels remain relatively stable in Peter Lake but a greater range of values were observed in July and August. Little seasonal variation was observed in nitrogen levels of both lakes, with Peter Lake showing a greater range of values in July and August.

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 <- neon %>%
  filter(functionalGroup == 'Needles') %>%
  select(collectDate,functionalGroup,dryMass,nlcdClass) %>%
  mutate(date = as.factor(collectDate))

needles %>%
  ggplot(aes(x=date, y=dryMass, fill=nlcdClass))+
  geom_boxplot() +
  labs(x='Date', y='Dry Mass (g)', fill='NLCD Class') +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
  my_theme
```



```
#7
needles %>%
  ggplot(aes(x=date, y=dryMass, fill=nlcdClass))+
  geom_boxplot() +
  facet_wrap(~nlcdClass, ncol = 1) +
  labs(x='Date', y='Dry Mass (g)', title='Dry Mass of Needles in 3 NLCD') +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1),
        legend.position = 'none') +
  my_theme
```



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

> Answer: The facet wrap plots in question 7 is more effective as the x-axis spans a large time frame and thus it was very difficult to see the data with three NLCD classes. Since the collection dates in the x-axis is the same across NLCD classes, separating the data into three facet wrap graphs allows us to better visualize the temporal trends of needle dry mass in each type of NLCD.