Assignment 5: Data Visualization

Becca Cox

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

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   3.5.1     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
```

```r
library(lubridate)

library(here)
```

```
## here() starts at /home/guest/EDE_Fall2024/EDE_Fall2024
```

```r
library(cowplot)
```

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

```r
getwd()
```

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

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

NiwotRidge.litter <-
  read.csv(here("Data/Processed_KEY/NEON_NIWO_Litter_mass_trap_Processed.csv"),
           stringsAsFactors = T)

#2

class(PeterPaul.chem.nutrients$sampledate)
```

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

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

class(PeterPaul.chem.nutrients$sampledate)
```

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

```r
class(NiwotRidge.litter$collectDate)
```

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

```r
NiwotRidge.litter$collectDate <- ymd(NiwotRidge.litter$collectDate)

class(NiwotRidge.litter$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

library(ggplot2)

#install.packages("wesanderson")

library(wesanderson)


my_theme <- theme(panel.background =
          element_rect(fill = 'lavenderblush1'),
          plot.background = element_rect(fill = "lavenderblush2"),
          axis.text = element_text(color = "darkslateblue"),
          axis.title = element_text(color = "darkslateblue"),
          plot.title = element_text(color = "darkslateblue"),
          legend.title = element_text(color = "darkslateblue"),
          scale_color_manual(values = wes_palette("GrandBudapest2")),
          legend.key = element_rect(fill = 'lavenderblush1'),
          legend.background = element_rect(fill = 'lavenderblush2'),
          legend.position = 'top')
```

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

totalpbypo4 <- PeterPaul.chem.nutrients %>%
  ggplot(aes(x = po4,
             y = tp_ug,
             color = lakename,
             )) +
  geom_point(alpha = 0.6) + my_theme +
  scale_color_manual(values = wes_palette("GrandBudapest2"))+
  ylab("Phosphorus (ug)")+
  xlab("Phosphate")+
  xlim(0,50)+
```

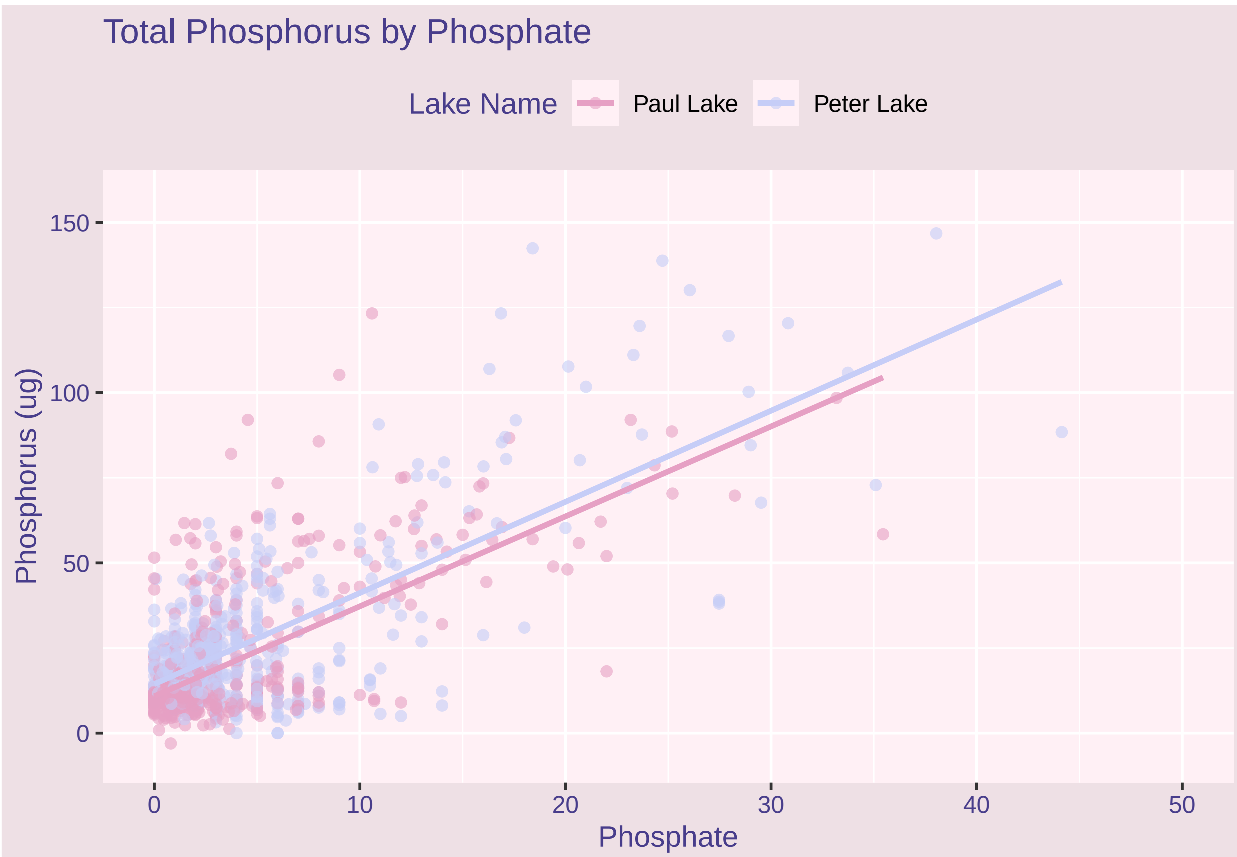
```
  labs(col = "Lake Name", title = "Total Phosphorus by Phosphate") +
  geom_smooth(method = "lm",
              se = FALSE)

print(totalpbypo4)
```

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



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`.

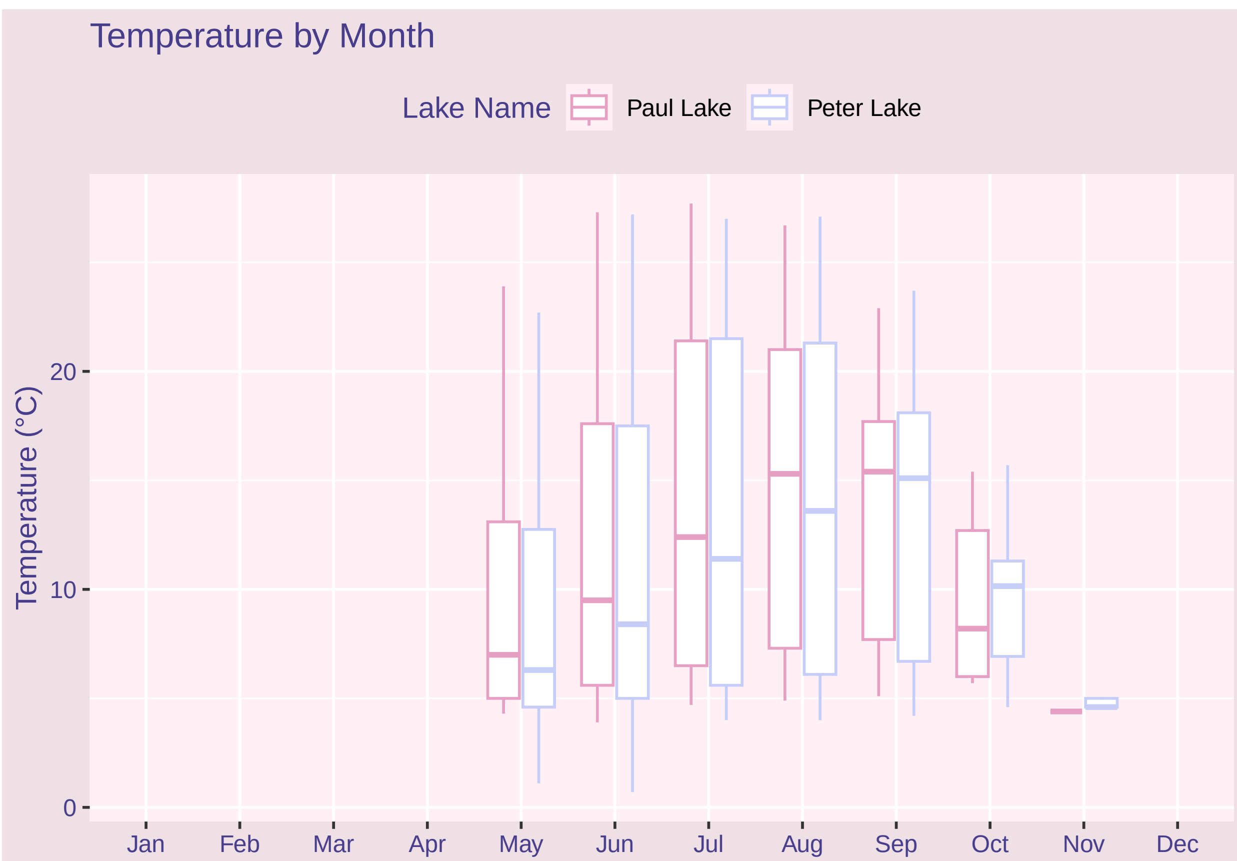
```r
#5

PeterPaul.chem.nutrients$month <- factor(PeterPaul.chem.nutrients$month,
                                         levels = 1:12,
                                         labels = month.abb)

tempbymonth <- PeterPaul.chem.nutrients %>%
  ggplot(aes(x = month,
             y = temperature_C,
             color = lakename,
             )) +
  geom_boxplot() +
  my_theme +
  theme(axis.title.x = element_blank())+
  scale_color_manual(values = wes_palette("GrandBudapest2"))+
  ylab("Temperature (°C)")+
  #xlab("Month") removed to clarify cowplot
  labs(col = "Lake Name", title = "Temperature by Month")+
  scale_x_discrete(drop = FALSE)

print(tempbymonth)
```

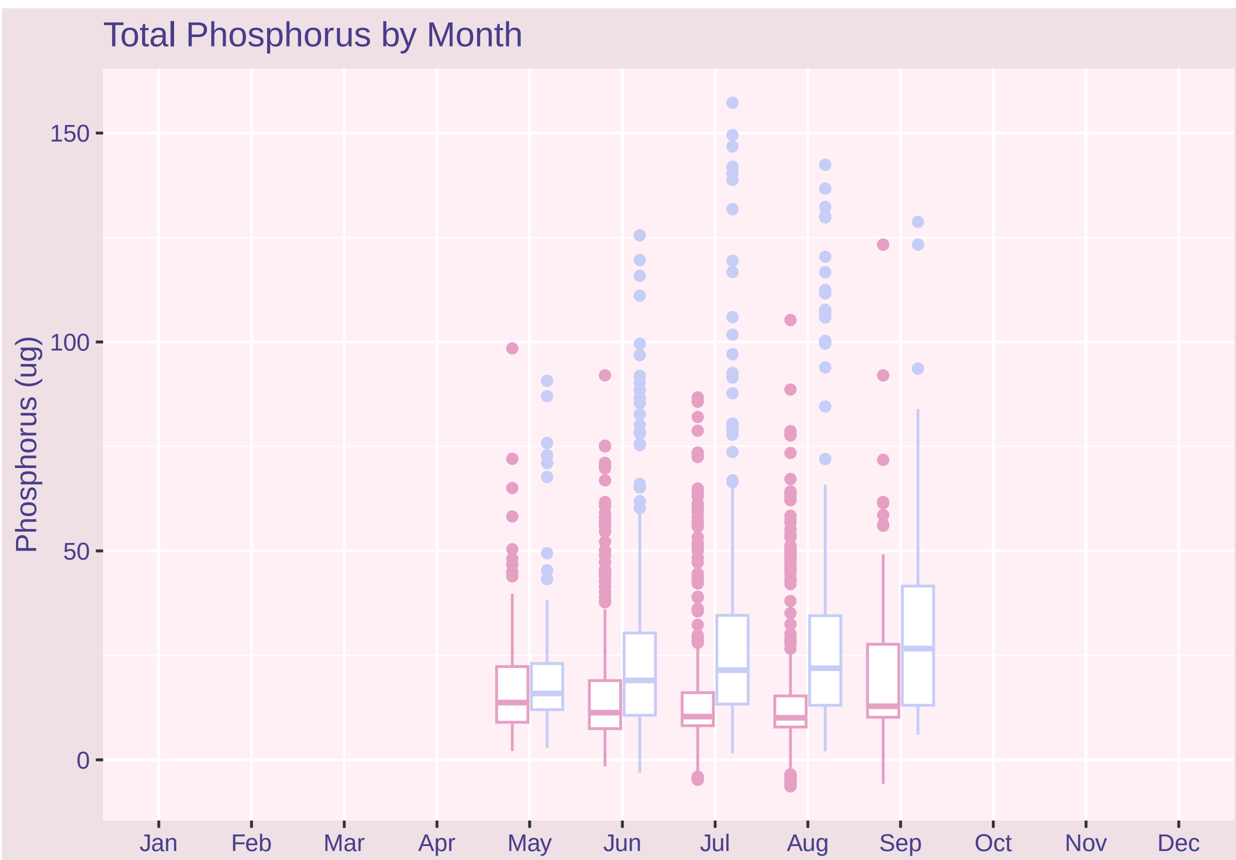
```
## Warning: Removed 3566 rows containing non-finite outside the scale range
## (‘stat_boxplot()‘).
```



```r
tpbymonth <- PeterPaul.chem.nutrients %>%
  ggplot(aes(x = month,
             y = tp_ug,
             color = lakename,
             )) +
  geom_boxplot() +
  my_theme +
  theme(axis.title.x = element_blank(), legend.position = 'none')+ #removed legend to clarify cowplot
  scale_color_manual(values = wes_palette("GrandBudapest2"))+
  ylab("Phosphorus (ug)")+
  #xlab("Month") removed to clarify cowplot
  labs(col = "Lake Name", title = "Total Phosphorus by Month")+
  scale_x_discrete(drop = FALSE)

print(tpbymonth)
```

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

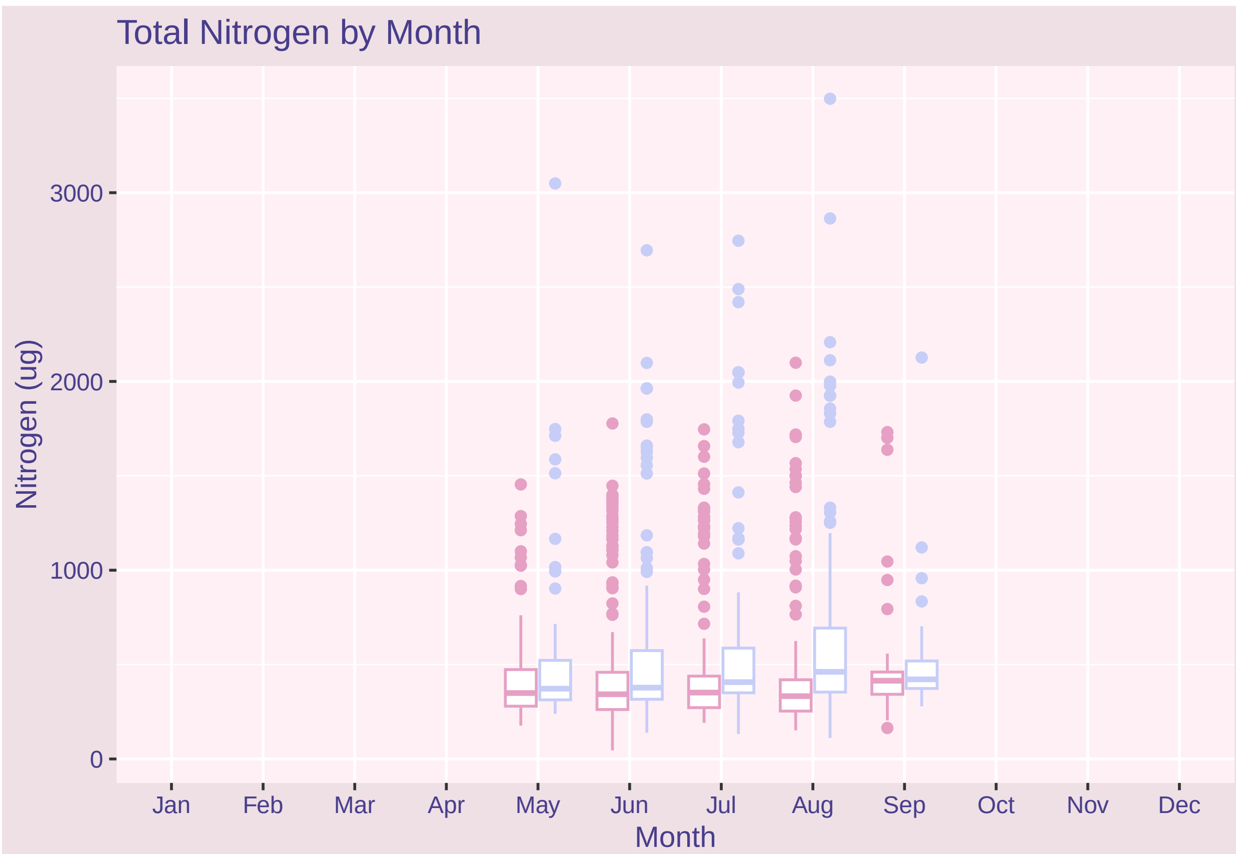



```r
tnbymonth <- PeterPaul.chem.nutrients %>%
  ggplot(aes(x = month,
             y = tn_ug,
             color = lakename,
             )) +
```

```
  geom_boxplot() +
  my_theme +
  theme(legend.position = 'none')+ #removed to clarify cowplot
  scale_color_manual(values = wes_palette("GrandBudapest2"))+
  ylab("Nitrogen (ug)")+
  xlab("Month")+
  labs(col = "Lake Name", title = "Total Nitrogen by Month")+
  scale_x_discrete(drop = FALSE)

print(tnbymonth)
```

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



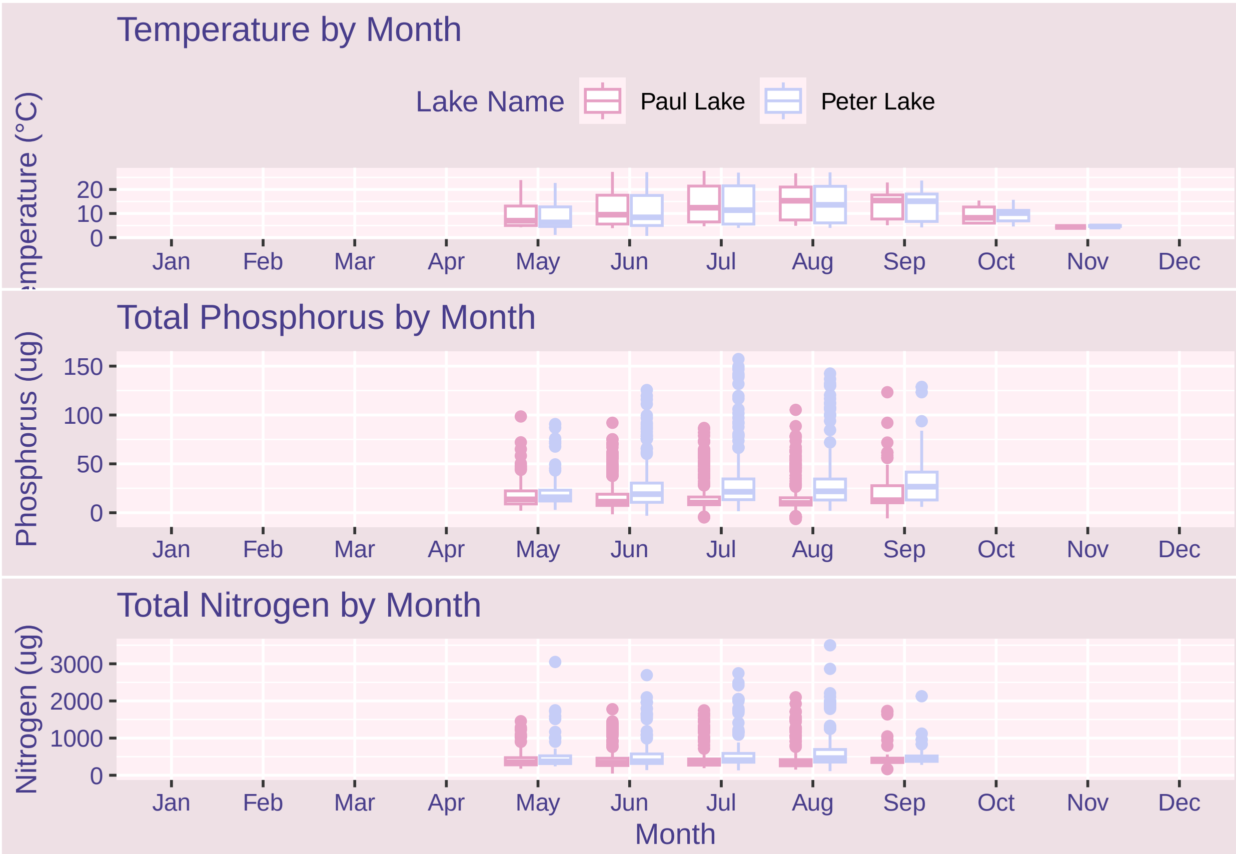

```
#install.packages("cowplot")
library(cowplot)

plot_grid(tempbymonth, tpbymonth, tnbymonth, nrow = 3, align = 'v')
```

```
## 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?

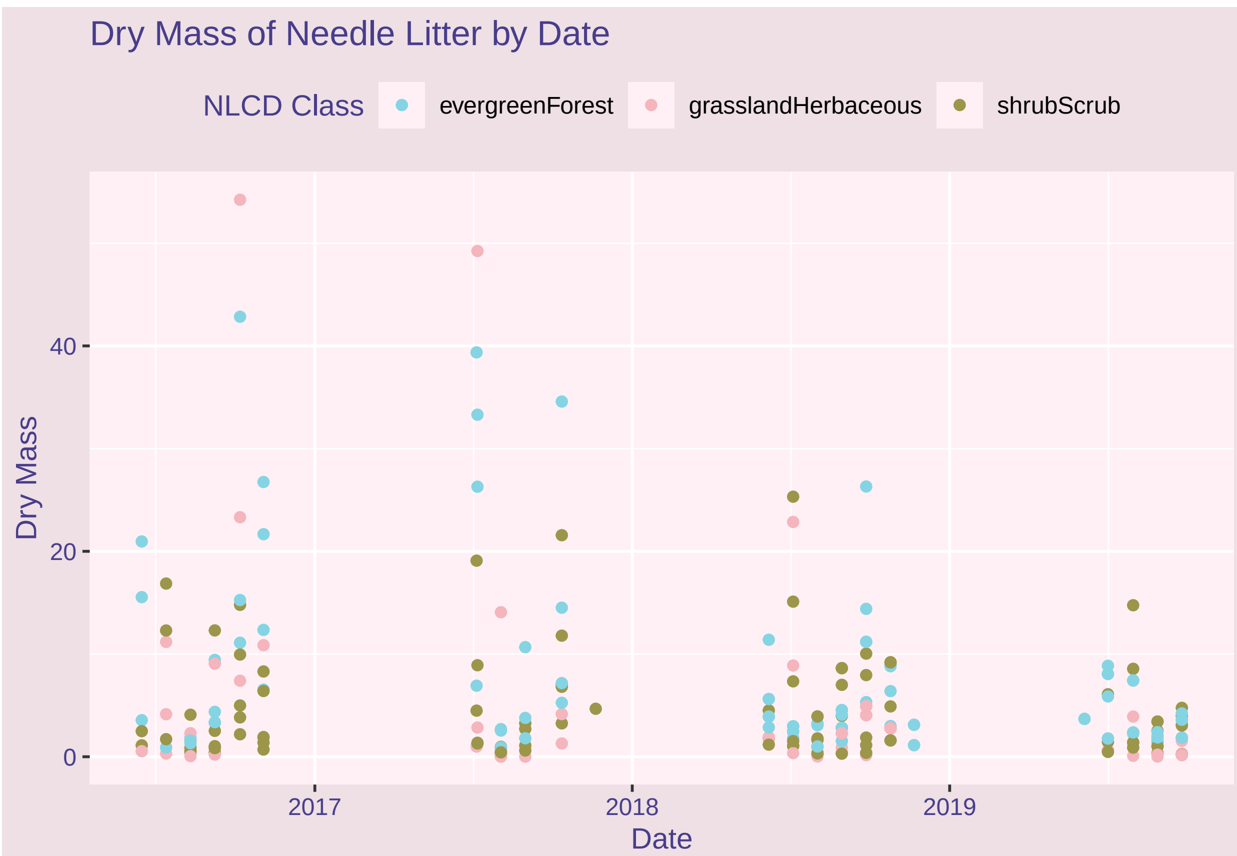
> Answer: All of these variables have the highest median values later in the year, the medians peaking in August or September. The peak values for Phosphorus and Nitrogen in Peter Lake, however, occur in July and August respectively; in Paul Lake, they occur in September and August respectively. The medians of both Phosphorus and Nitrogen are higher in Peter Lake between May and September. The median of Phosphorus in Paul Lake appears to decrease slightly between May and August, which differs from what happens in Peter Lake, though the boxes within the plot still overlap and the maximum median is still in September, similar to the other variables. In both lakes, Phosphorus and Nitrogen have smaller boxes within the box plot and therefore more outliers, meaning the values for these nutrients do not vary greatly, whereas there is more variability in - and therefore bigger boxes for - Temperature values. Temperature noticeably decreases in both lakes in October and November. Overall, the values are comparable between lakes for Temperature, Phosphorus, and Nitrogen.

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.

```r
#6

needlesbydate <- NiwotRidge.litter %>%
  filter(functionalGroup=="Needles") %>%
  ggplot(aes(x = collectDate,
             y = dryMass,
             color = nlcdClass,
             )) +
  geom_point() +
  my_theme +
  scale_color_manual(values = wes_palette("Moonrise3"))+
  ylab("Dry Mass")+
  xlab("Date")+
  labs(col = "NLCD Class", title = "Dry Mass of Needle Litter by Date")

print(needlesbydate)
```

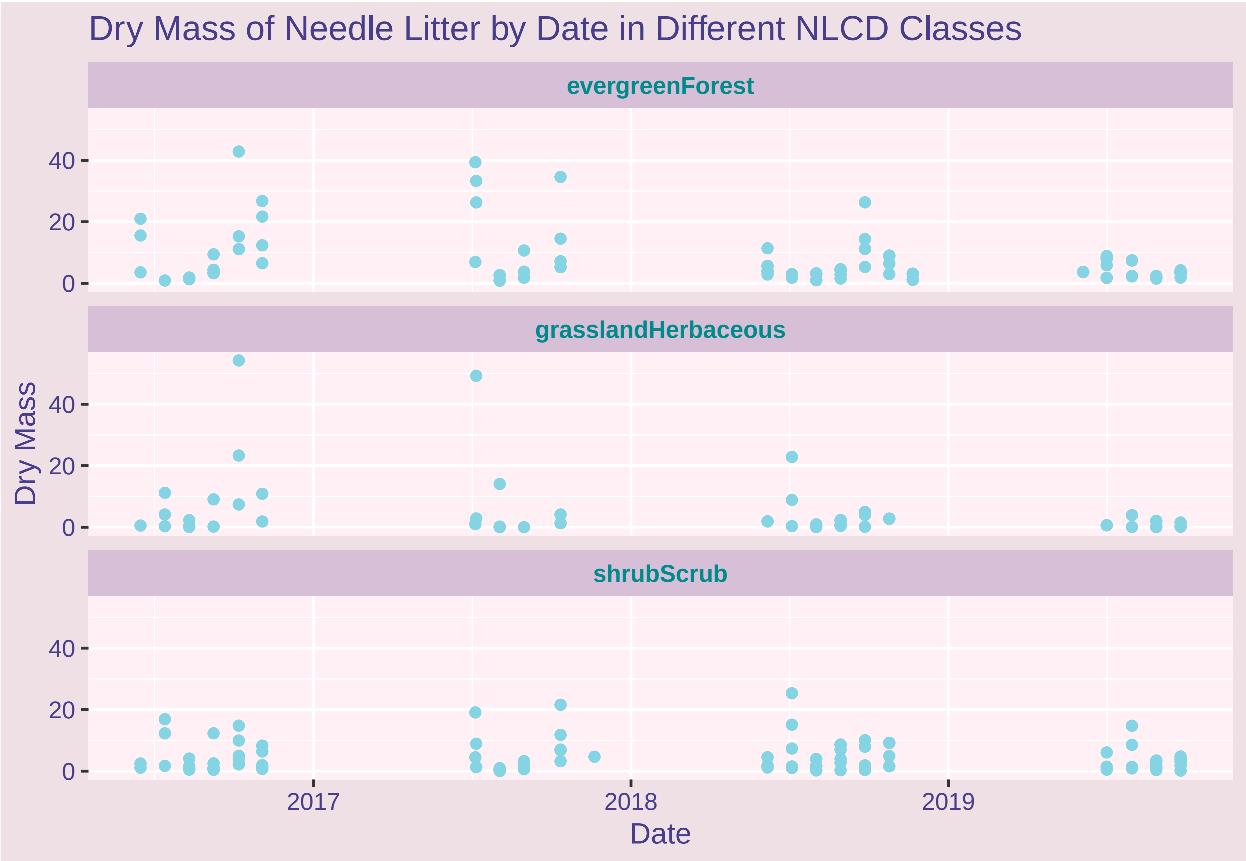



```r
#7

needlesfacet <-
  NiwotRidge.litter %>%
  filter(functionalGroup=="Needles") %>%
  ggplot(aes(x = collectDate,
             y = dryMass,
             color = functionalGroup)) +
```

```
  geom_point() +
  my_theme +
  theme(legend.position = 'none')+
  scale_color_manual(values = wes_palette("Moonrise3"))+
  ylab("Dry Mass")+
  xlab("Date")+
  labs(title = "Dry Mass of Needle Litter by Date in Different NLCD Classes") +
  facet_wrap(facets=vars(nlcdClass),nrow=3,)+
  theme(strip.background =element_rect(fill="thistle"))+
  theme(strip.text = element_text(colour = 'cyan4', face = (face ="bold")))

print(needlesfacet)
```



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

> Answer: I think that plot 7, in which the NLCD classes are divided into facets, is more effective. The data points from different classes overlap in plot 6; plot 7 is clearer. Because they are separated but share the same x-axis, it is easier to compare between different classes within the same time frame.