Assignment 5: Data Visualization

Emma Kaufman

Fall 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. 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
#Install packages
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## 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) #The here package allows for better control of relative paths
```

```
## here() starts at /home/guest/EDE_Fall2023
```

```r
library(cowplot)
```

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

```r
library(ggplot2)
library(ggthemes)
```

```
##
## Attaching package: ’ggthemes’
##
## The following object is masked from ’package:cowplot’:
##
##     theme_map
```

```r
#Ensure that "here" points to your project folder
here()
```

```
## [1] "/home/guest/EDE_Fall2023"
```

```r
#get working directory
getwd()
```

```
## [1] "/home/guest/EDE_Fall2023"
```

```r
#read in files
LakeChem <- read.csv(
  file=here("Data/Processed_KEY/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv"),
  stringsAsFactors = TRUE
)

Litter <- read.csv(
  file=here("Data/Processed_KEY/NEON_NIWO_Litter_mass_trap_Processed.csv"),
  stringsAsFactors = TRUE
)

#2
#changing to date format
Litter$collectDate <- ymd(Litter$collectDate)
LakeChem$sampledate <- ymd(LakeChem$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

```
#3
#building my theme
my_theme <- theme_base()  +
  theme(
    line = element_line(
      color = 'navyblue',
      linewidth = 0.5
      ),
    plot.title = element_text(
      color = 'navyblue',
      size = 15
    ),
    axis.text = element_text(
      color = 'navyblue'
    ),
    axis.title.x = element_text(
      color = "navyblue",
      #size = 8
      ),
  axis.title.y = element_text(
    color = "navyblue"
    ),
    plot.background = element_rect(
      fill = 'lightblue'
    ))

theme_set(my_theme) #setting as 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 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()`).

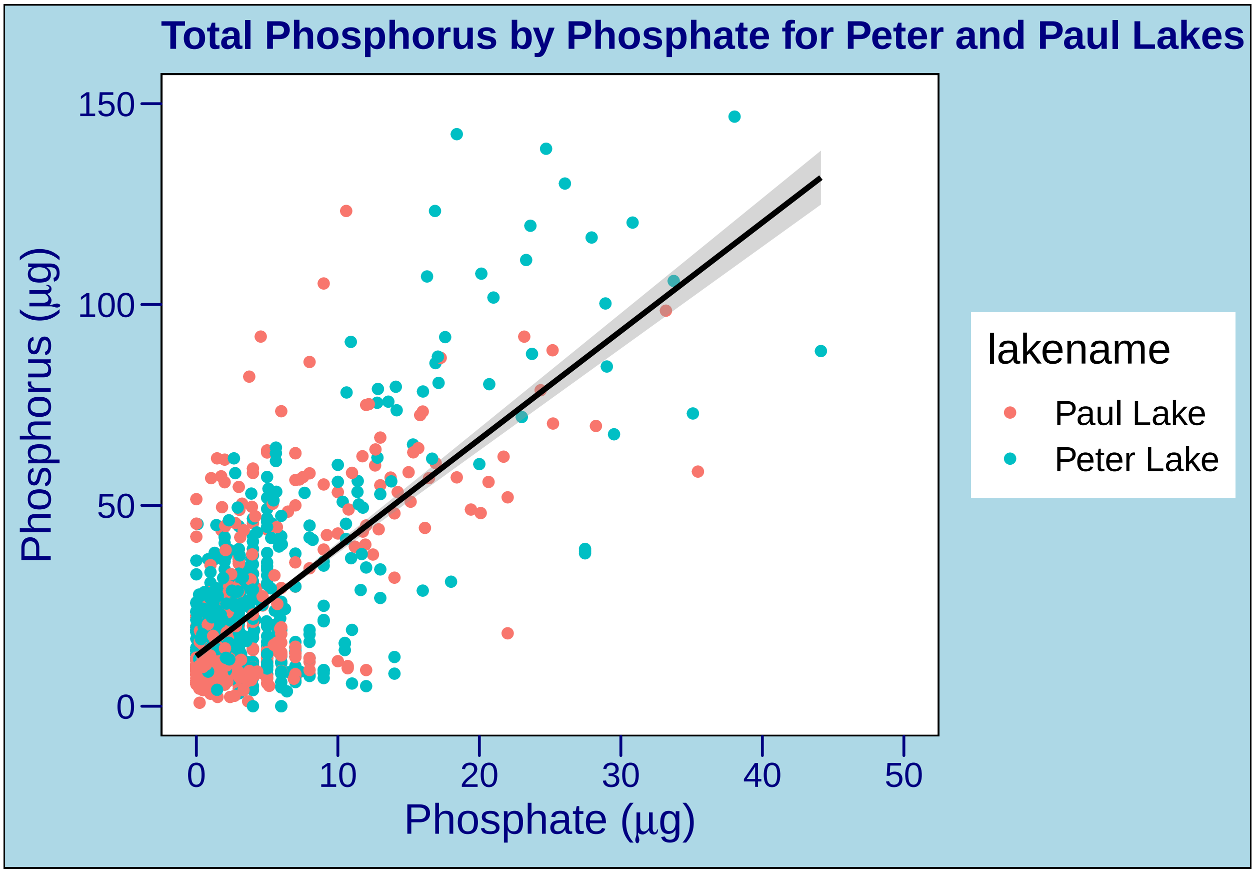
```
#4
phos_plot<-
  ggplot((LakeChem),
         #choosing variables
         aes(
            y = tp_ug,
            x = po4,
            #shape = lakename,
            color = lakename
          )) +
  geom_point()+
  #adding trendline
  geom_smooth(method = 'lm',
              color = 'black')+
  #removing outliers
  ylim(0,150)+
  xlim(0,50)+
  #adjusting aesthetics of the plot
  ggtitle("Total Phosphorus by Phosphate for Peter and Paul Lakes")+
  #my_theme +
  labs(y= expression(paste("Phosphorus (", mu, "g)")), x= expression(paste("Phosphate (", mu, "g)")))

print(phos_plot) #displaying the graph
```

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

## Warning: Removed 21948 rows containing non-finite values (`stat_smooth()`).

## Warning: Removed 21948 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: * Recall the discussion on factors in the previous section as it may be helpful here. * R has a built-in variable called `month.abb` that returns a list of months;see https://r-lang.com/month-abb-in-r-with-example

```
#5
unique(LakeChem$month) #seeing which months should plot data
```

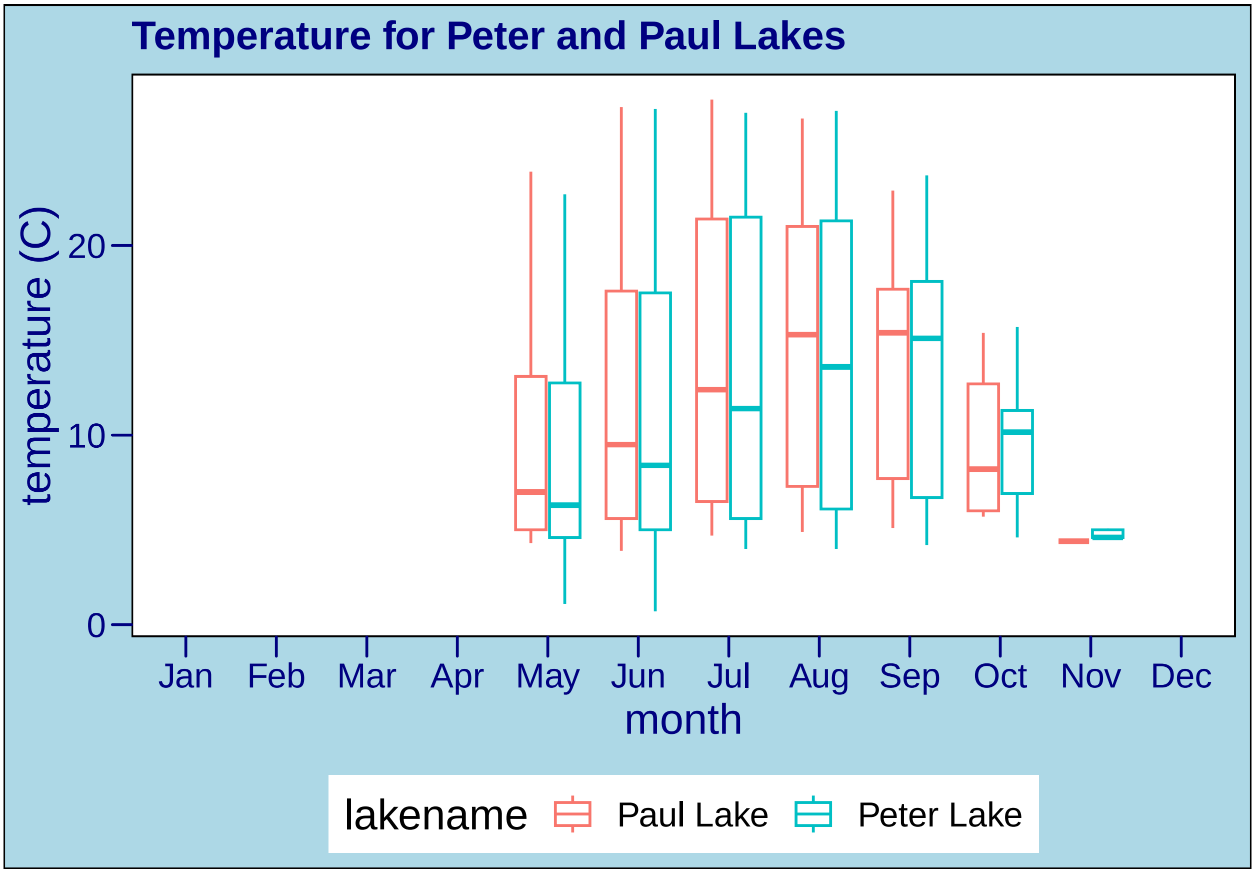
```
## [1]  5  6  7  8  9 10 11  2
```

```
#a) plotting temperature
Temperatureplot <-
  ggplot((LakeChem),
         aes(
           x = factor(LakeChem$month, levels=1:12, labels = month.abb),
           y = temperature_C,
           color = lakename)) +
  geom_boxplot() +
  ggtitle("Temperature for Peter and Paul Lakes") +
  scale_x_discrete(name= "month", drop = FALSE) +
  labs(y = "temperature (C)") +
  my_theme +
  theme(legend.position = "bottom")

print(Temperatureplot)
```

```
## Warning: Use of ‘LakeChem$month‘ is discouraged.
## i Use ‘month‘ instead.
```

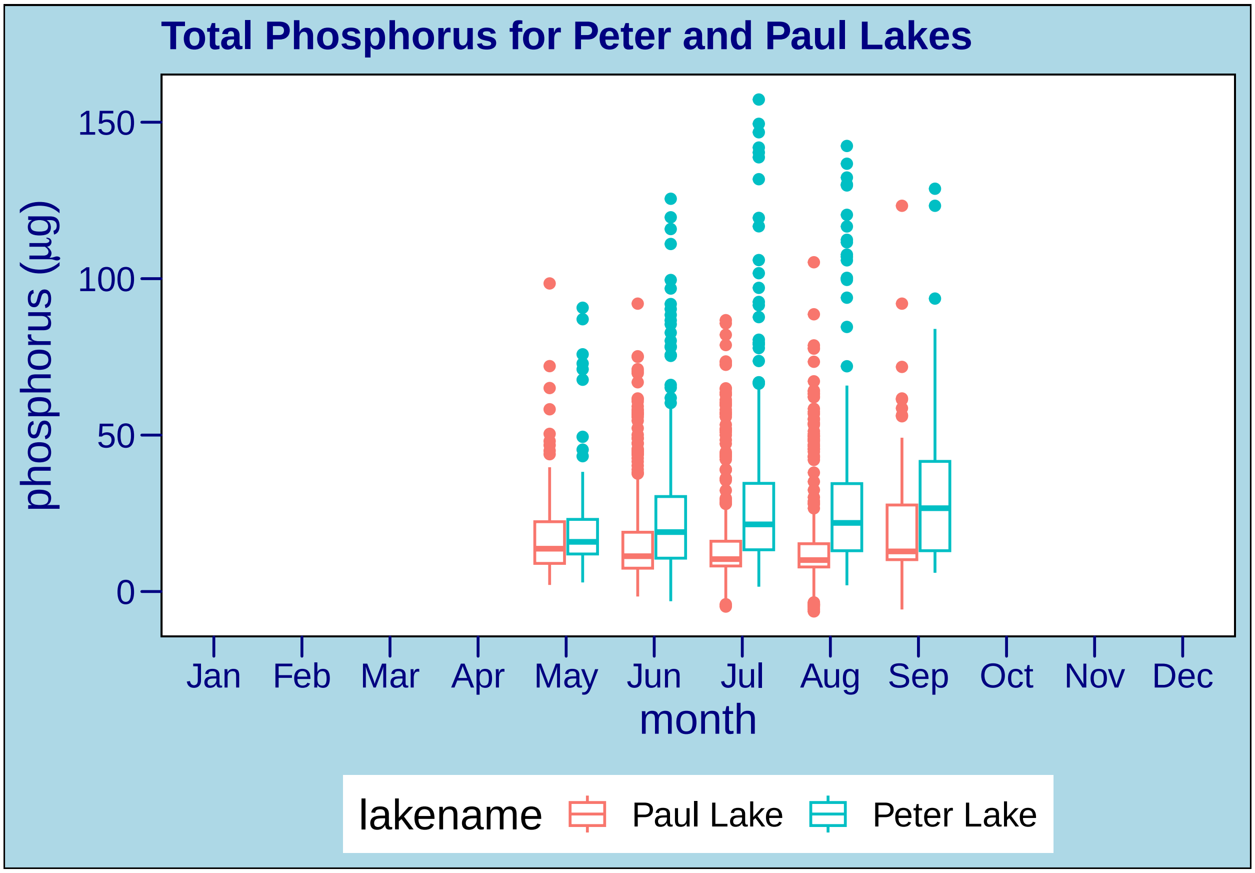
```
## Warning: Removed 3566 rows containing non-finite values (‘stat_boxplot()‘).
```



```
#b) plotting total phosphorus
TPplot <-
  ggplot((LakeChem),
         aes(
           x = factor(month, levels=1:12, labels = month.abb),
           y = tp_ug,
           color = lakename)) +
  geom_boxplot() +
  ggtitle("Total Phosphorus for Peter and Paul Lakes") +
  scale_x_discrete(name= "month", drop = FALSE) +
  labs(y = expression(paste("phosphorus (", mu, "g)")))+
  my_theme +
  theme(legend.position = "bottom")

print(TPplot)
```

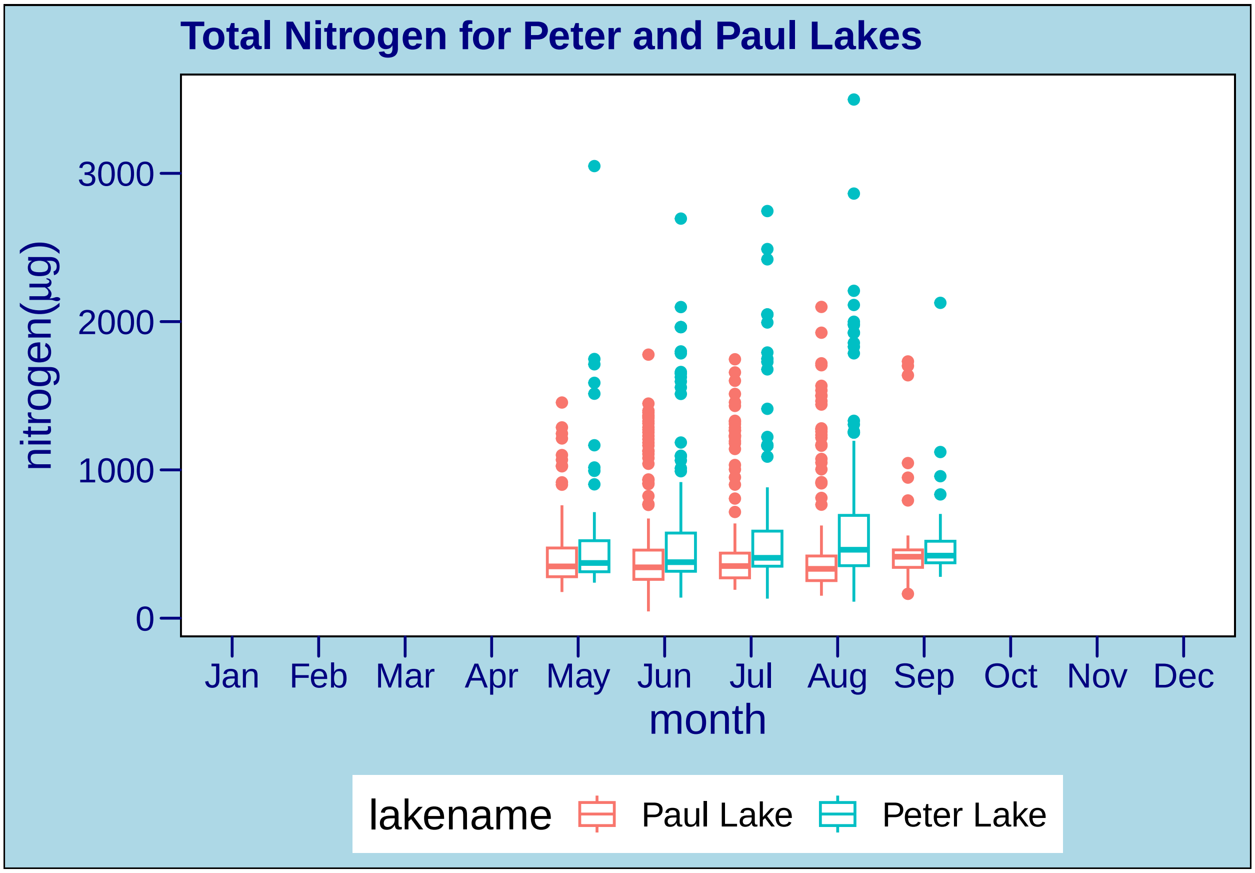
```
## Warning: Removed 20729 rows containing non-finite values (‘stat_boxplot()‘).
```



```
#c) plotting total nitrogen
TNplot <-
  ggplot((LakeChem),
         aes(
           x = factor(month, levels=1:12, labels = month.abb),
           y = tn_ug,
           color = lakename)) +
  geom_boxplot() +
  ggtitle("Total Nitrogen for Peter and Paul Lakes")+
  scale_x_discrete(name= "month", drop = FALSE) +
  labs(y = expression(paste("nitrogen(", mu, "g)"))) +
  my_theme +
  theme(legend.position = "bottom")

print(TNplot)
```

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



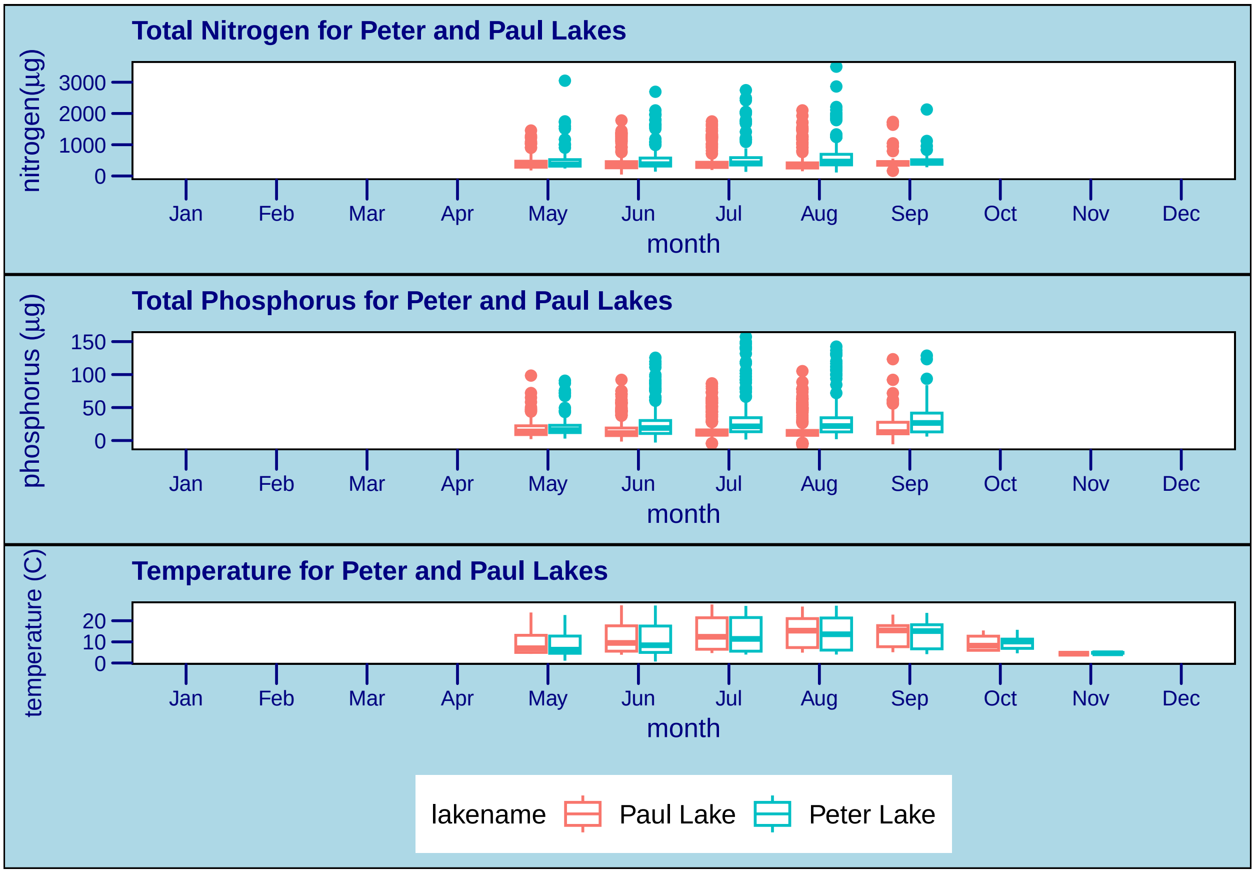
```
#plotting using cowplot to display all figures as one image
plot_grid(TNplot+theme(
  legend.position = "none",
  axis.title.x = element_text(size= 10),
  axis.title.y = element_text(size = 10),
  plot.title = element_text(size = 10),
  axis.text = element_text(size = 8)),
  #no legend for two of the plots
  TPplot+ theme(legend.position = "none",
              axis.title.x = element_text(size= 10),
              axis.title.y = element_text(size = 10),
              plot.title = element_text(size = 10),
              axis.text = element_text(size = 8)),
  Temperatureplot + theme(axis.title.x = element_text(size= 10),
              axis.title.y = element_text(size = 9),
              plot.title = element_text(size = 10),
              legend.text = element_text(size = 10),
              legend.title = element_text(size = 10),
              axis.text = element_text(size = 8)),
  ncol = 1,
  rel_heights = c(1.25,1.25,1.5),
  #ensuring alignment
  align = "vh")
```

```
## Warning: Removed 21583 rows containing non-finite values (‘stat_boxplot()‘).
## Warning: Removed 20729 rows containing non-finite values (`stat_boxplot()`).

## Warning: Use of `LakeChem$month` is discouraged.
## i Use `month` instead.

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

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

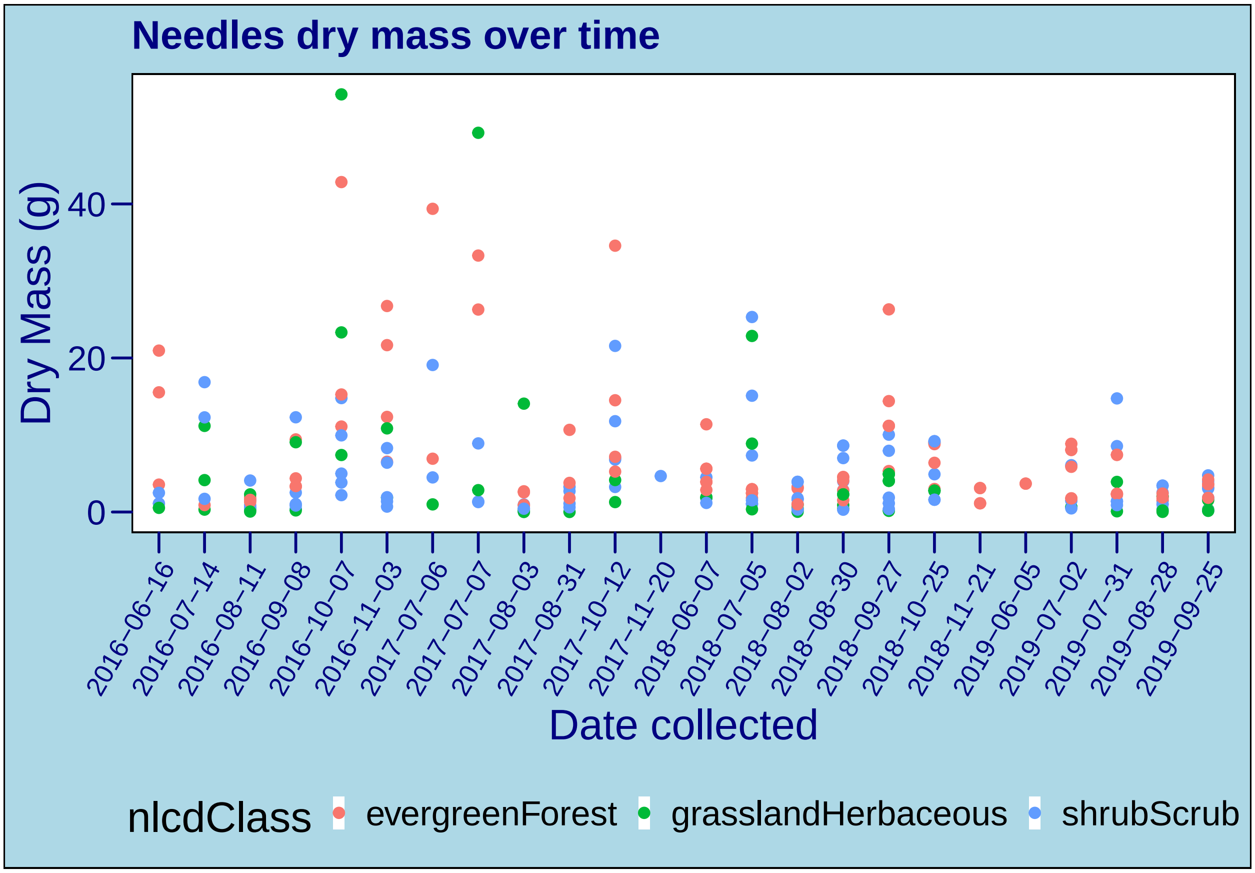


Question: What do you observe about the variables of interest over seasons and between lakes? > Answer: Temperature is similar in peter and paul lakes across the seasons displayed. Phosphorus and nitrogen are both higher in peter lake each month. The average of phosphorus and nitrogen stay relatively consistent for each lake regardless of month.

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.

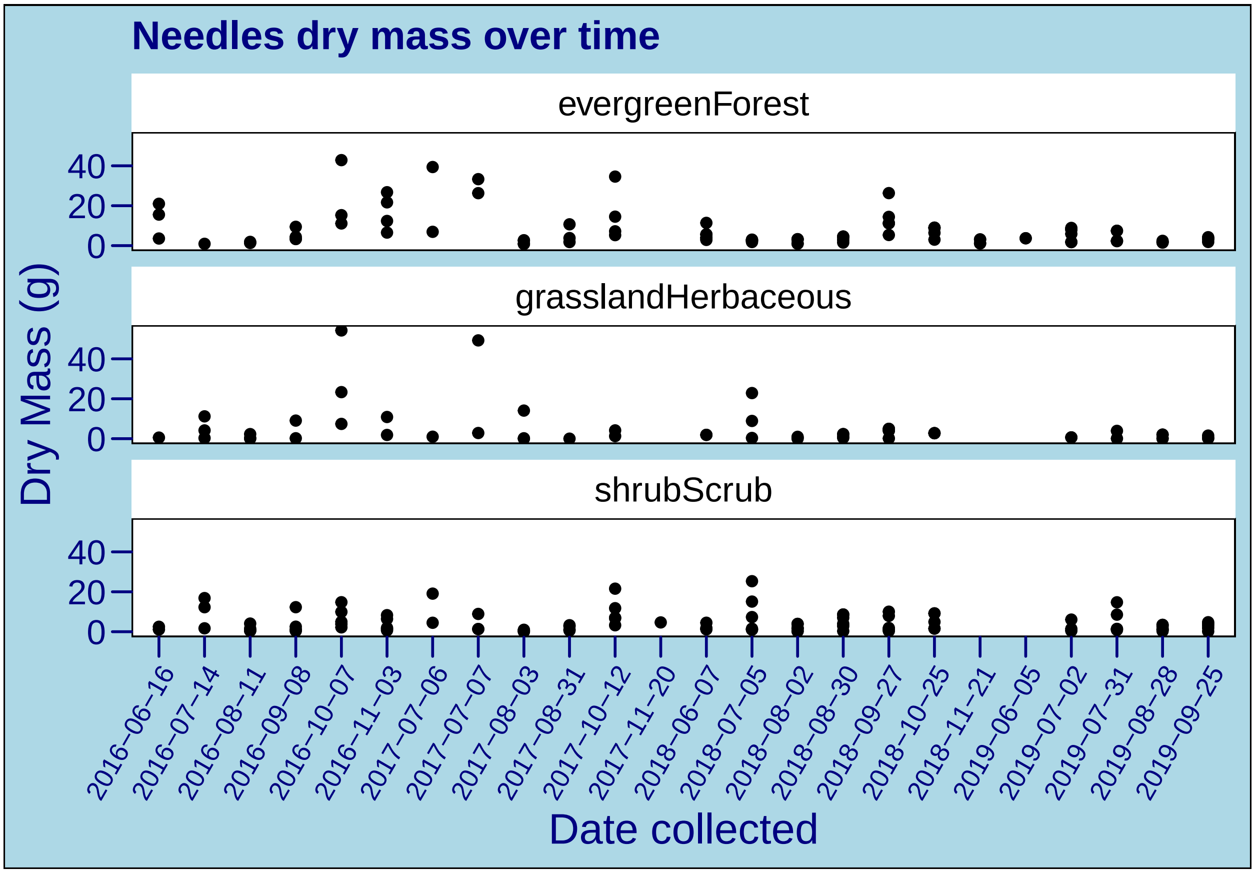
```
#6
Needles_mass <-
  Litter %>%
  filter(functionalGroup == "Needles")

Needlesplot <- ggplot((Needles_mass),
                      aes(
                        x= factor(collectDate), #what is the form the date should take?
                        #x = collectDate,
                        y= dryMass,
                        color = nlcdClass
                      )) +
  geom_point()+ #what is the type of plot I should make?
  ggtitle("Needles dry mass over time")+
  #scale_x_date(date_breaks = "4 months", date_labels = '%m/%y')+
  my_theme+
  theme(legend.position = "bottom",
        legend.key.size = unit(0.1, "cm"),
        legend.background = element_rect(fill = "transparent", color = NA),
        axis.text.x = element_text(angle = 60,
                                   vjust = 1,
                                   hjust = 1,
                                   size = 10)) +
        labs(x= "Date collected", y = "Dry Mass (g)")
print(Needlesplot)
```



```
#7
Needlesplot_faceted <- ggplot((Needles_mass),
                      aes(
                        x= factor(collectDate),
                        y= dryMass
                      )) +
  geom_point()+
  facet_wrap(vars(nlcdClass), nrow = 3) +
  ggtitle("Needles dry mass over time") +
 # scale_x_date(date_breaks = "4 months", date_labels = '%m/%y')+
  labs(x= "Date collected", y = "Dry Mass (g)") +
  my_theme +
  theme(axis.text.x = element_text(angle = 60,
                                   vjust = 1,
                                   hjust = 1,
                                   size = 10))

print(Needlesplot_faceted)
```



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

> Answer:The faceted plot is more effective. You can see differences in needle mass for different land cover classes must more effectively. It allows you to see patterns in the data broken down by land cover class. When all of the data are plotted together in different colors it is harder to see any pattern in the data. The faceted graphs allow for more direct comparison.