Assignment 5: Data Visualization

Student Name

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)
```

```
## -- 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 error
```

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

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

```r
library(cowplot)
```

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

```r
getwd()
```

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

```r
Peter.Paul.nutrient <- read.csv(
  file = here(
    "Data/Processed_KEY/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv"),
  stringsAsFactors = T
)
Litter <- read.csv(
  file = here(
    "Data/Processed_KEY/NEON_NIWO_Litter_mass_trap_Processed.csv"), stringsAsFactors = T
)

#2
class(Peter.Paul.nutrient$sampledate)
```

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

```r
class(Litter$collectDate) # both are read in as factors. Change to date format
```

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

```r
Peter.Paul.nutrient$sampledate <- ymd(Peter.Paul.nutrient$sampledate)
class(Peter.Paul.nutrient$sampledate)
```

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

```r
Litter$collectDate <- ymd(Litter$collectDate)
class(Litter$collectDate) # now both are in date format
```

```
## [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
# define my theme:
mytheme <- theme_classic(base_size = 10) +
  theme(axis.text = element_text(color = "dark grey"),
        panel.background = element_rect(fill = "light grey", color = NA),
        legend.position = "top",
        plot.title = element_text(size = 13, hjust = 0.5))

#set my theme:
theme_set(mytheme)
```

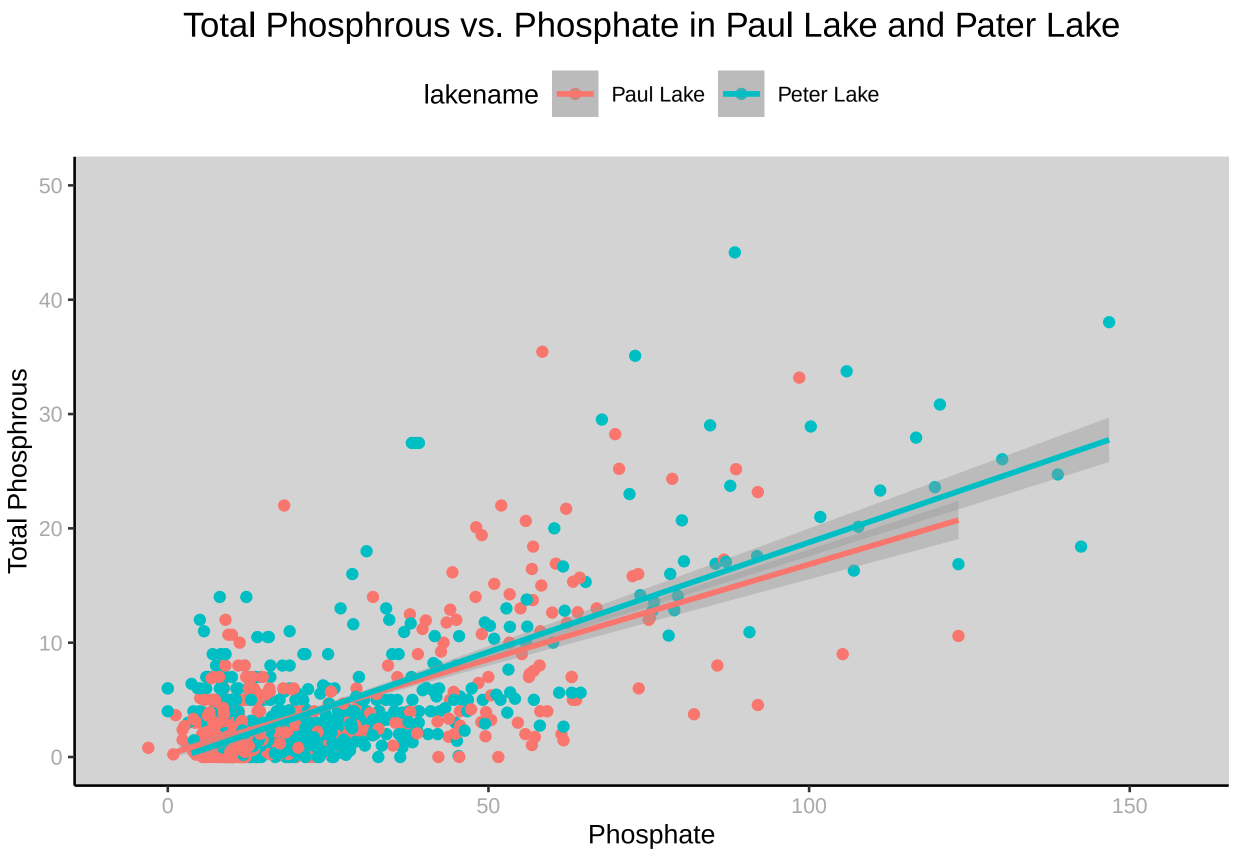
Create graphs

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 generate plot
phosphorus.phosphate.plot <-
  ggplot(Peter.Paul.nutrient, aes(x = tp_ug, y = po4, color = lakename)) +
  geom_point() +
  ylim(0, 50) +
  geom_smooth(method = lm) +
  labs(title = "Total Phosphrous vs. Phosphate in Paul Lake and Pater Lake",
       x = "Phosphate",
       y = "Total Phosphrous")
print(phosphorus.phosphate.plot)
```

```
## ‘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()‘).
```

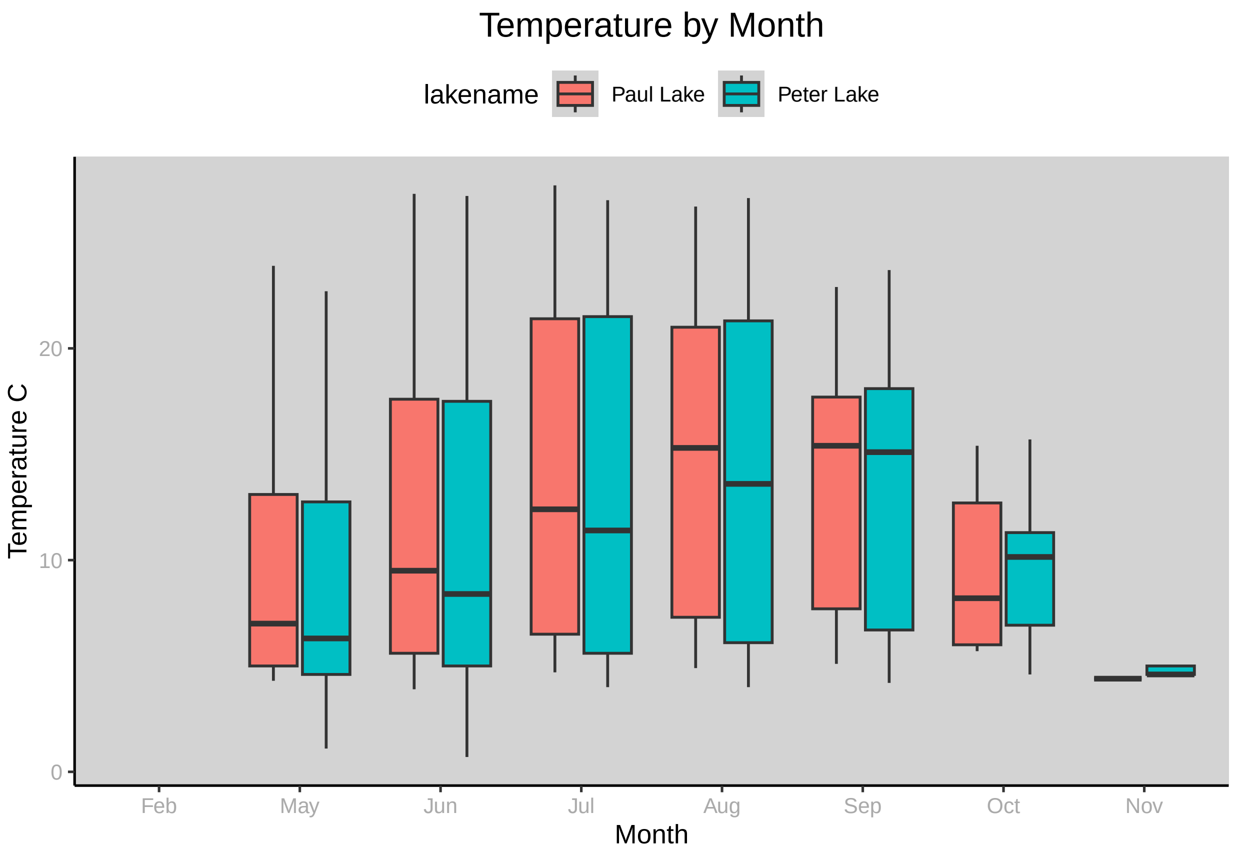


5. [NTL-LTER] Make three separate boxplots of (a) temperature, (b) TP, and (c) TN, with month as the x axis and lake as a color aesthetic. Then, create a cowplot that combines the three graphs. Make sure that only one legend is present and that graph axes are aligned.

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`.

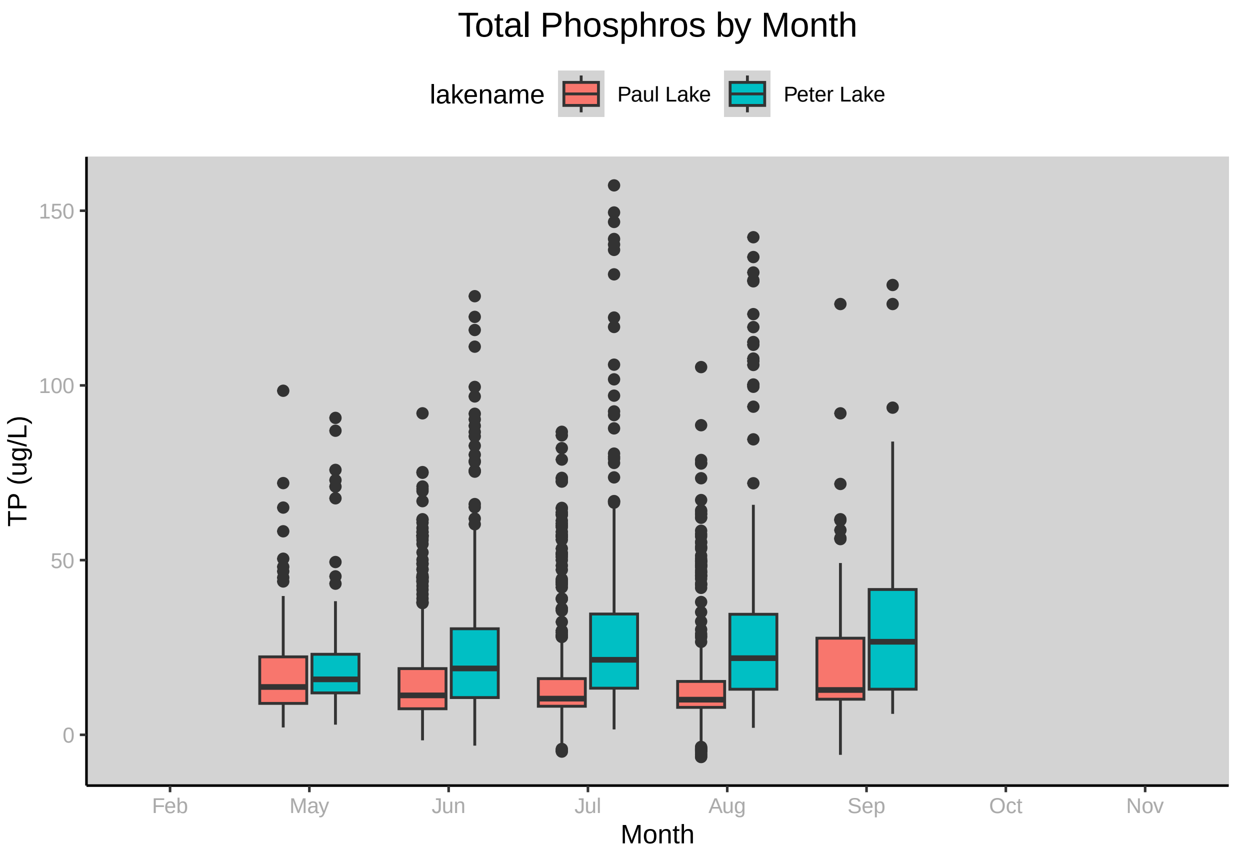
```
#5
# separate plot 1 temperature
Peter.Paul.nutrient$month <- as.numeric(Peter.Paul.nutrient$month)

Peter.Paul.nutrient$month <- factor(Peter.Paul.nutrient$month,
                                    levels = 1:12,
                                    labels = month.abb)
temperature.plot <-
  ggplot(Peter.Paul.nutrient, aes(x = month, y = temperature_C, fill = lakename)) +
  labs(title = "Temperature by Month",
       x = "Month",
       y = "Temperature C") +
  geom_boxplot()
print(temperature.plot)
```

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



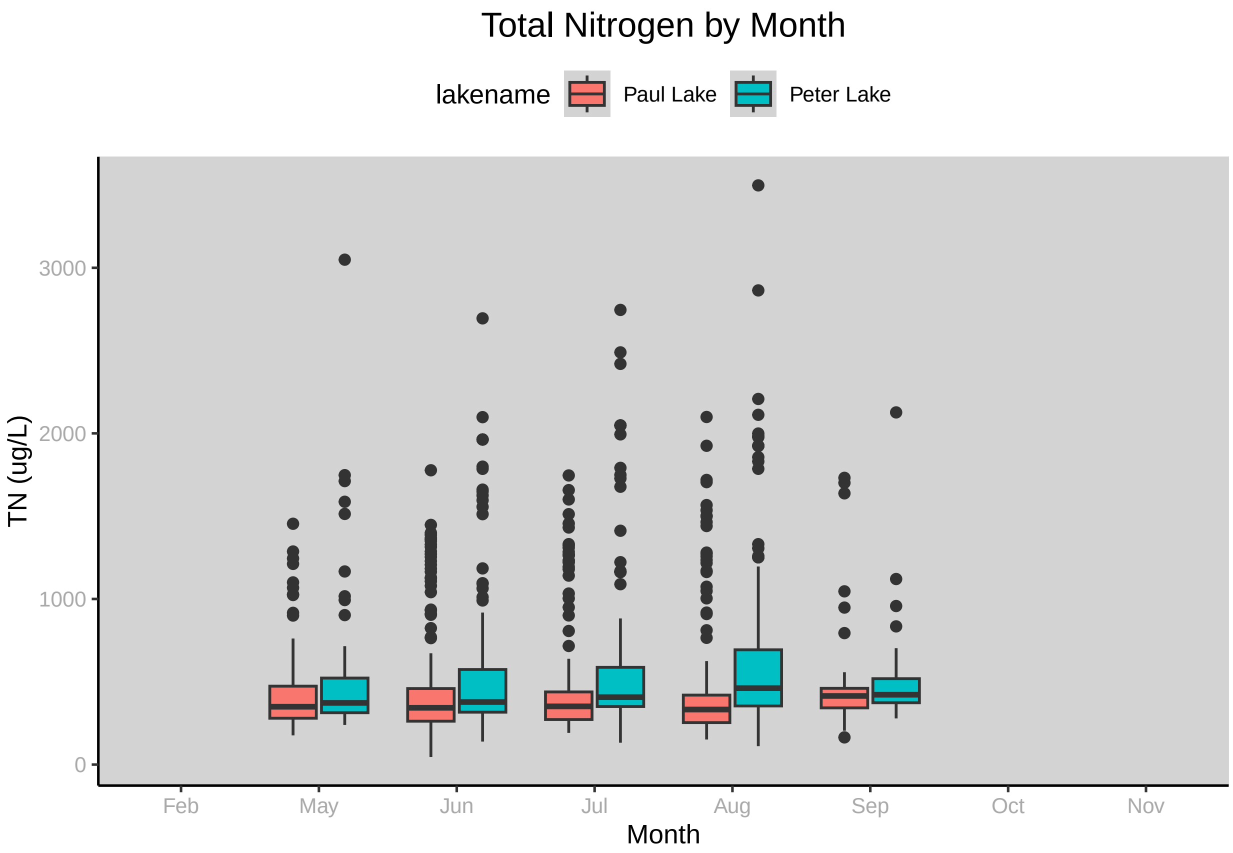
```
# separate plot 2 TP
TP.plot <-
  ggplot(Peter.Paul.nutrient, aes(x = month, y = tp_ug, fill = lakename)) +
  labs(title = "Total Phosphros by Month",
       x = "Month",
       y = "TP (ug/L)") +
  geom_boxplot()
print(TP.plot)
```

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



```
#separate plot 3 TN
TN.plot <-
  ggplot(Peter.Paul.nutrient, aes(x = month, y = tn_ug, fill = lakename)) +
  labs(title = "Total Nitrogen by Month",
       x = "Month",
       y = "TN (ug/L)") +
  geom_boxplot()
print(TN.plot)
```

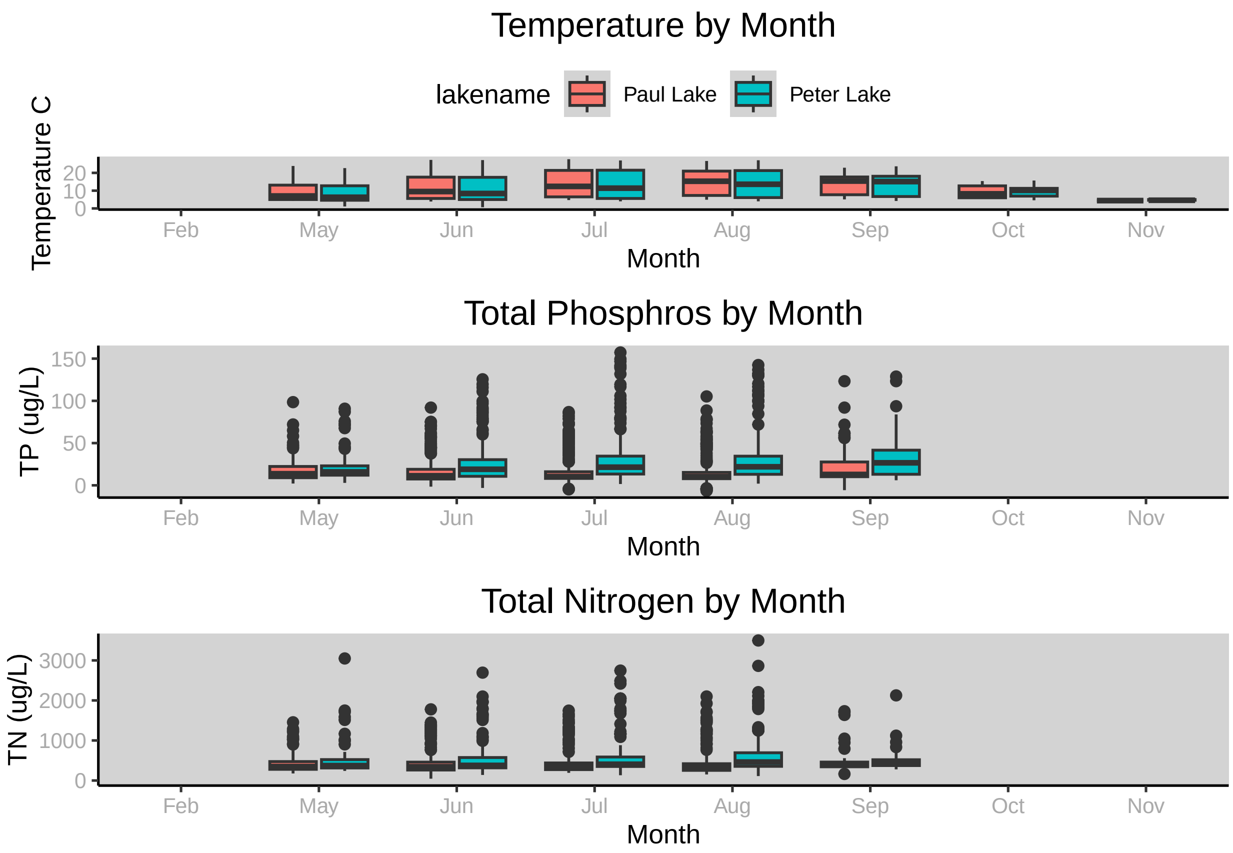
```
## Warning: Removed 21583 rows containing non-finite outside the scale range
## (‘stat_boxplot()‘).
```



```r
# combining 3 plots
temp.TP.TN.plot <- plot_grid(temperature.plot,
                             TP.plot + theme(legend.position = "none"),
                             TN.plot + theme(legend.position = "none"),
                             ncol = 1,
                             align = 'v',
                             axis = 'lr',
                             label_size = 8
                             )
```

```
## 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()‘).
```

```r
print(temp.TP.TN.plot)
```



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

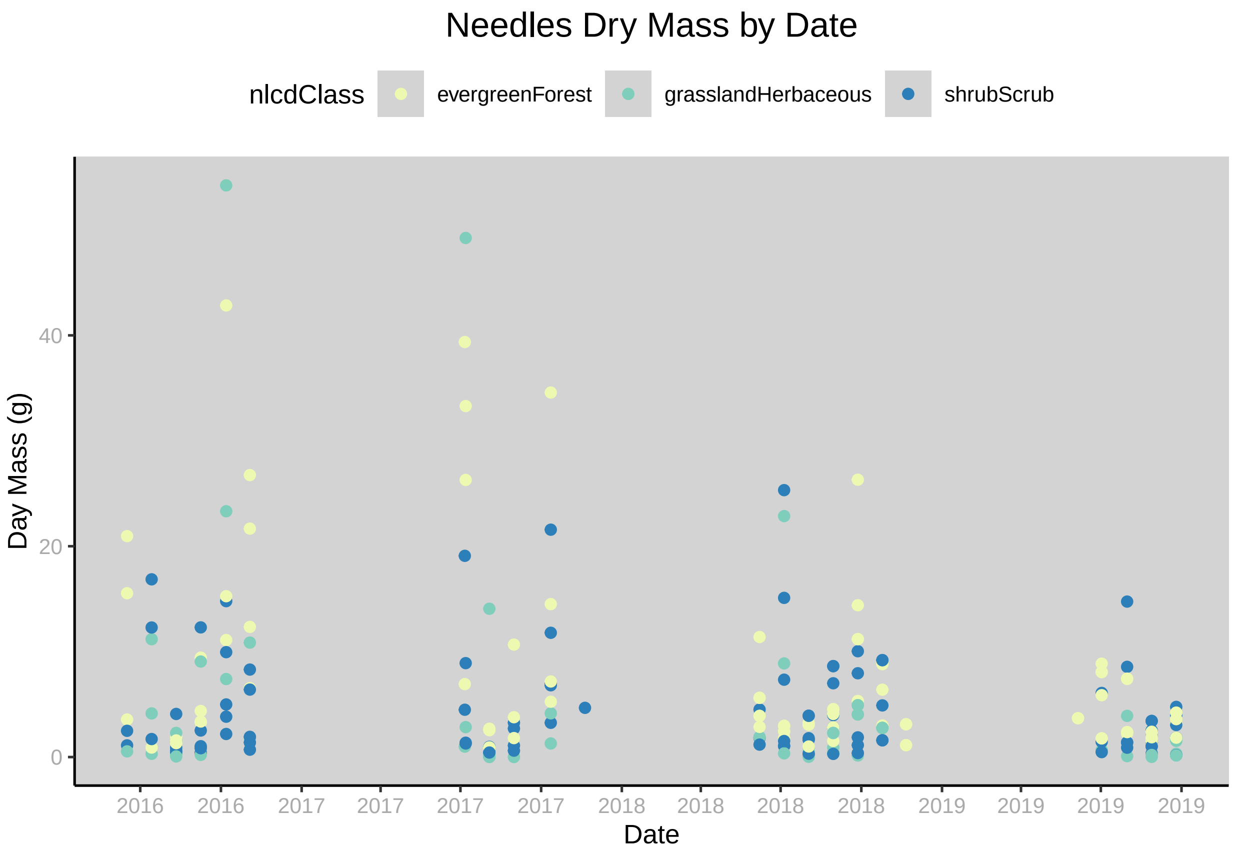
> Answer: Temperature is the highest from July and August for both lakes. Peter Lake has both higher total phosphorus and total nitrogen compared to Peter Lake, especially in June.

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.

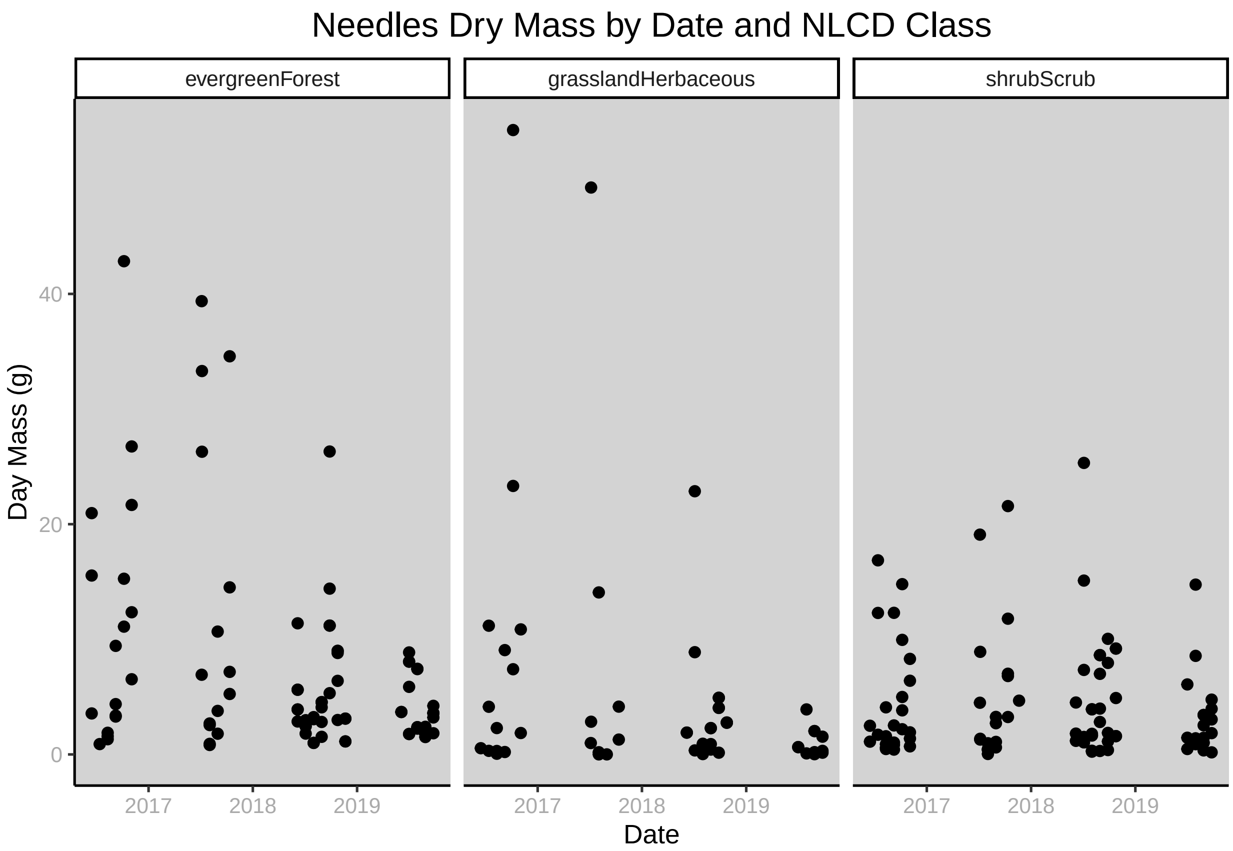
```
#6
# filter out needle data
needles.data <- Litter %>%
  filter(functionalGroup == "Needles")  %>%
  mutate(collectDate = as.Date(collectDate))

head(needles.data)
```

```
##     plotID       trapID collectDate functionalGroup dryMass qaDryMass subplotID
## 1 NIWO_058 NIWO_058_101  2016-06-16         Needles    1.11         Y        32
## 2 NIWO_047 NIWO_047_197  2016-06-16         Needles    0.54         N        40
## 3 NIWO_057 NIWO_057_081  2016-06-16         Needles   20.96         N        31
## 4 NIWO_064 NIWO_064_103  2016-06-16         Needles    3.56         N        32
## 5 NIWO_061 NIWO_061_169  2016-06-16         Needles   15.54         Y        41
## 6 NIWO_062 NIWO_062_050  2016-06-16         Needles    2.49         N        31
##   decimalLatitude decimalLongitude elevation           nlcdClass plotType
## 1        40.04872        -105.5872    3446.4          shrubScrub    tower
## 2        40.05466        -105.5844    3509.8 grasslandHerbaceous    tower
## 3        40.04708        -105.5851    3382.5     evergreenForest    tower
## 4        40.04737        -105.5840    3373.2     evergreenForest    tower
## 5        40.04762        -105.5861    3413.4     evergreenForest    tower
## 6        40.05114        -105.5858    3477.0          shrubScrub    tower
##   geodeticDatum
## 1         WGS84
## 2         WGS84
## 3         WGS84
## 4         WGS84
## 5         WGS84
## 6         WGS84
```

```
needles.plot <- ggplot(needles.data, aes(x = collectDate, y = dryMass, color = nlcdClass)) +
  geom_point() +
  labs(title = "Needles Dry Mass by Date",
       x = "Date",
       y = "Day Mass (g)") +
  scale_color_brewer(palette = "YlGnBu") +
  scale_x_date(
    date_breaks = "3 months",
    date_labels = "%Y"
  )

print(needles.plot)
```



```
#7
needles.plot.facets <- ggplot(needles.data, aes(x = collectDate, y = dryMass))+
  geom_point() +
  labs(title = "Needles Dry Mass by Date and NLCD Class",
       x = "Date",
       y = "Day Mass (g)") +
  facet_wrap(facets = vars(nlcdClass), nrow = 1, ncol = 3)

print(needles.plot.facets)
```



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

> Answer: Plot 7 is more efficient because it provides separate graphs for each land cover type. This is easier to read and provides more details than plot 6, where it combines all three types together in one plot.