Assignment 5: Data Visualization

Danlei Zou

OVERVIEW

This exercise accompanies the lessons in Environmental Data Analytics on Data Visualization

Directions

1. Rename this file `<FirstLast>_A02_CodingBasics.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 to **answer the questions** in this assignment document.
5. When you have completed the assignment, **Knit** the text and code into a single PDF file.

The completed exercise is due on Friday, Oct 14th @ 5:00pm.

Set up your session

1. Set up your session. Verify your working directory and load the tidyverse, lubridate, & cowplot packages. Upload the NTL-LTER processed data files for
nutrients and chemistry/physics for Peter and Paul Lakes (use the tidy [`NTL-LTER_Lake_Chemistry_Nutrients_PeterP`
version) and the processed data file for the Niwot Ridge litter dataset (use the [`NEON_NIWO_Litter_mass_trap_Processe`
version).

2. Make sure R is reading dates as date format; if not change the format to date.

```
# 1

# loading packages
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.4
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.2      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
##
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(cowplot)
```

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

```
# loading relevant datasets
PeterPaul.Lake <- read.csv("./Data/Processed/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv"
    stringsAsFactors = TRUE)
Litter <- read.csv("./Data/Processed/NEON_NIWO_Litter_mass_trap_Processed.csv", stringsAsFactors = TRUE

# 2

# Set date to date format
PeterPaul.Lake$sampledate <- as.Date(PeterPaul.Lake$sampledate, format = "%Y-%m-%d")
Litter$collectDate <- as.Date(Litter$collectDate, format = "%Y-%m-%d")

class(PeterPaul.Lake$sampledate)
```

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

```
class(Litter$collectDate)
```

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

Define your theme

3. Build a theme and set it as your default theme.

```
# 3

# Set theme
mytheme <- theme_classic(base_size = 14) + theme(axis.text = element_text(color = "black"),
    legend.position = "top")
theme_set(mytheme)
```

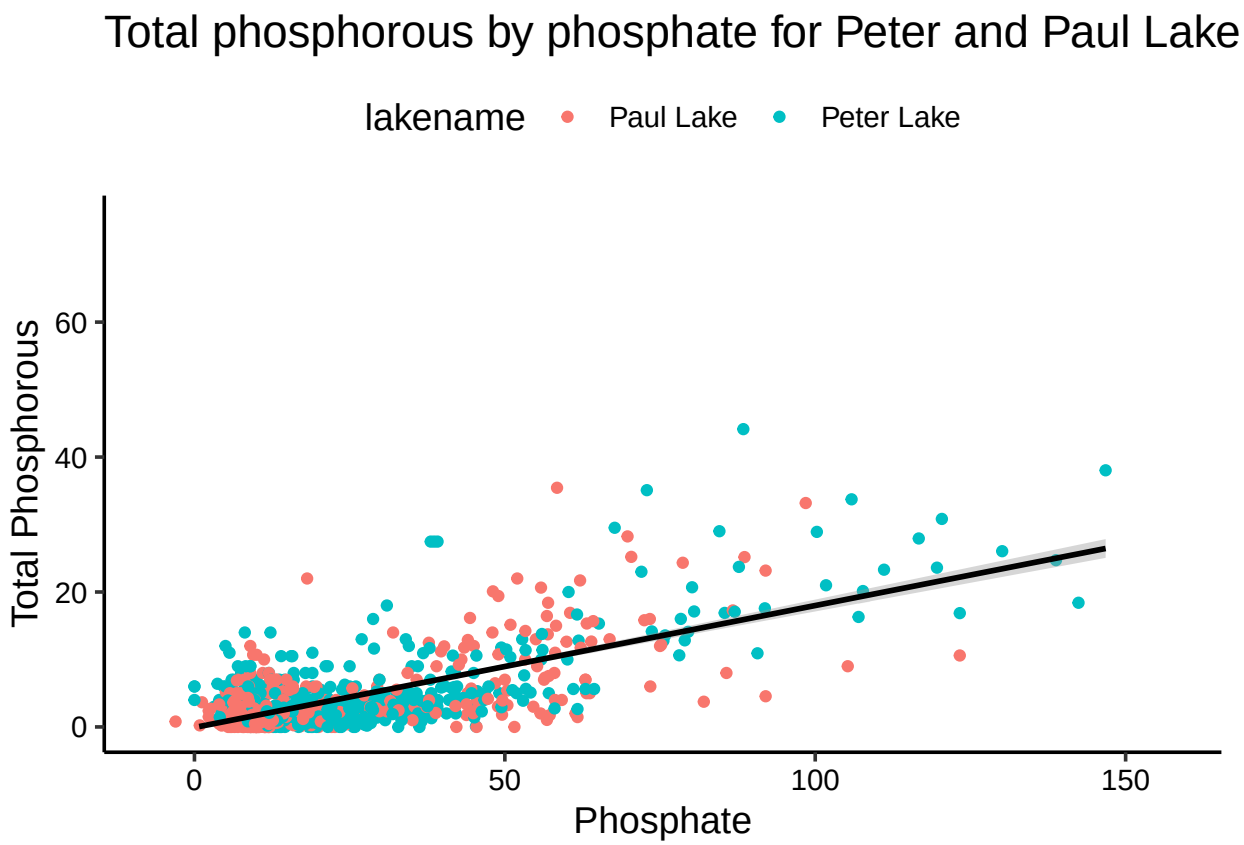
Create graphs

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 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

# creating plot of tp_ug by po4

PeterPaul.plot <- ggplot(PeterPaul.Lake, aes(x = tp_ug, y = po4, color = lakename)) +
    geom_point() + geom_smooth(method = lm, color = "black") + xlab("Phosphate") +
    ylab("Total Phosphorous") + ylim(0, 75) + ggtitle("Total phosphorous by phosphate for Peter and Paul
print(PeterPaul.plot)
```

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

## Warning: Removed 21947 rows containing non-finite values (stat_smooth).

## Warning: Removed 21947 rows containing missing values (geom_point).

## Warning: Removed 2 rows containing missing values (geom_smooth).
```



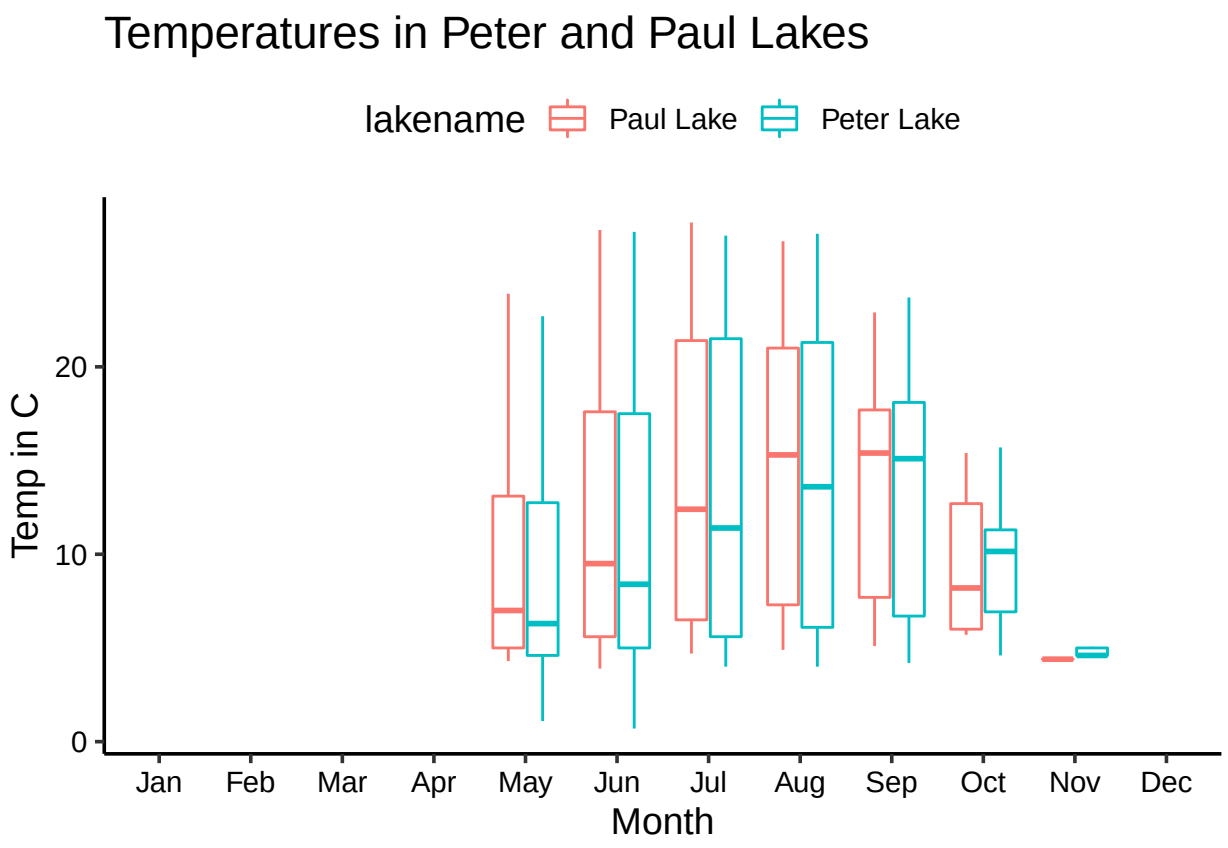
5. [NTL-LTER] Make three separate boxplots of (a) temperature, (b) TP, and

(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.

Tip: R has a build in variable called `month.abb` that returns a list of months; see https://r-lang.com/month-abb-in-r-with-example

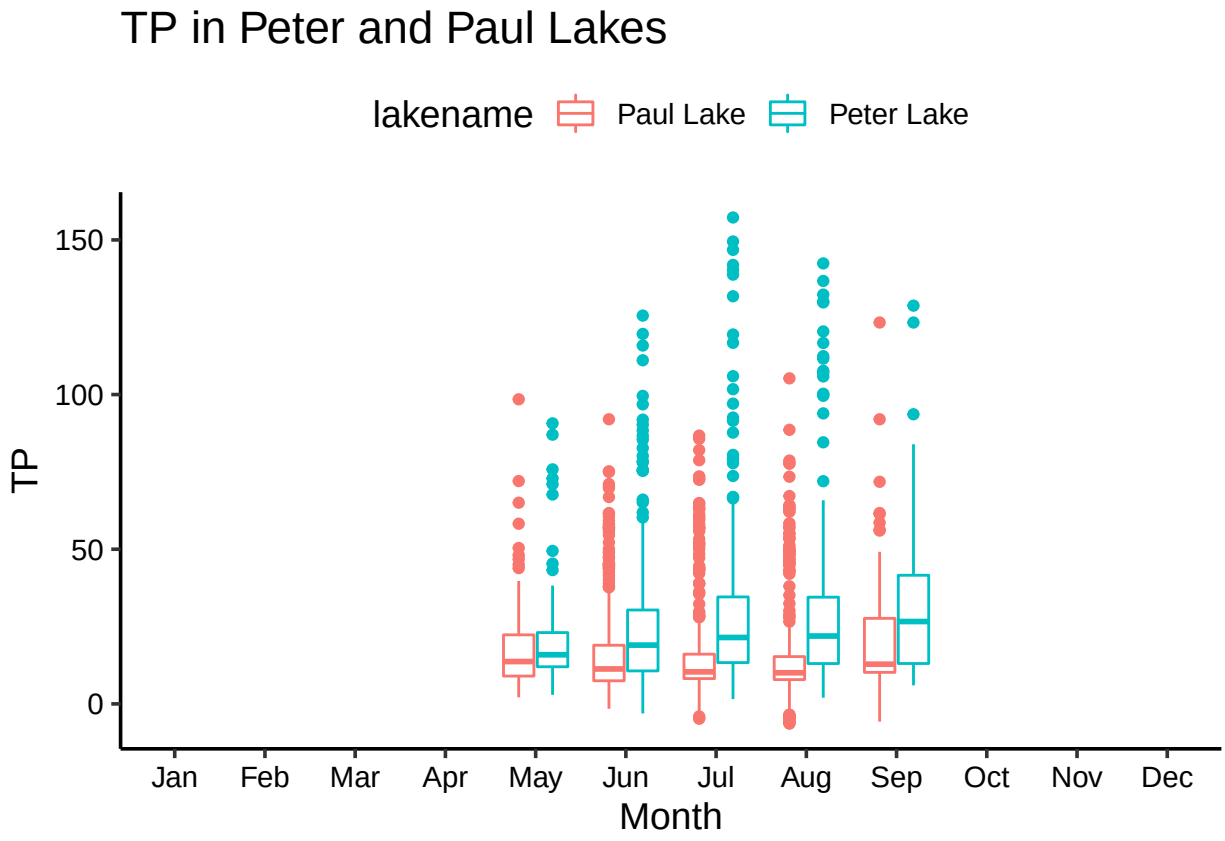
```
# 5

# boxplot of temperature
PeterPaul.temperature.plot <- ggplot(PeterPaul.Lake, aes(x = factor(month, levels = c(1:12),
    labels = month.abb), y = temperature_C)) + geom_boxplot(aes(color = lakename)) +
    ggtitle("Temperatures in Peter and Paul Lakes") + xlab("Month") + ylab("Temp in C") +
    scale_x_discrete(drop = FALSE)
print(PeterPaul.temperature.plot)
```

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



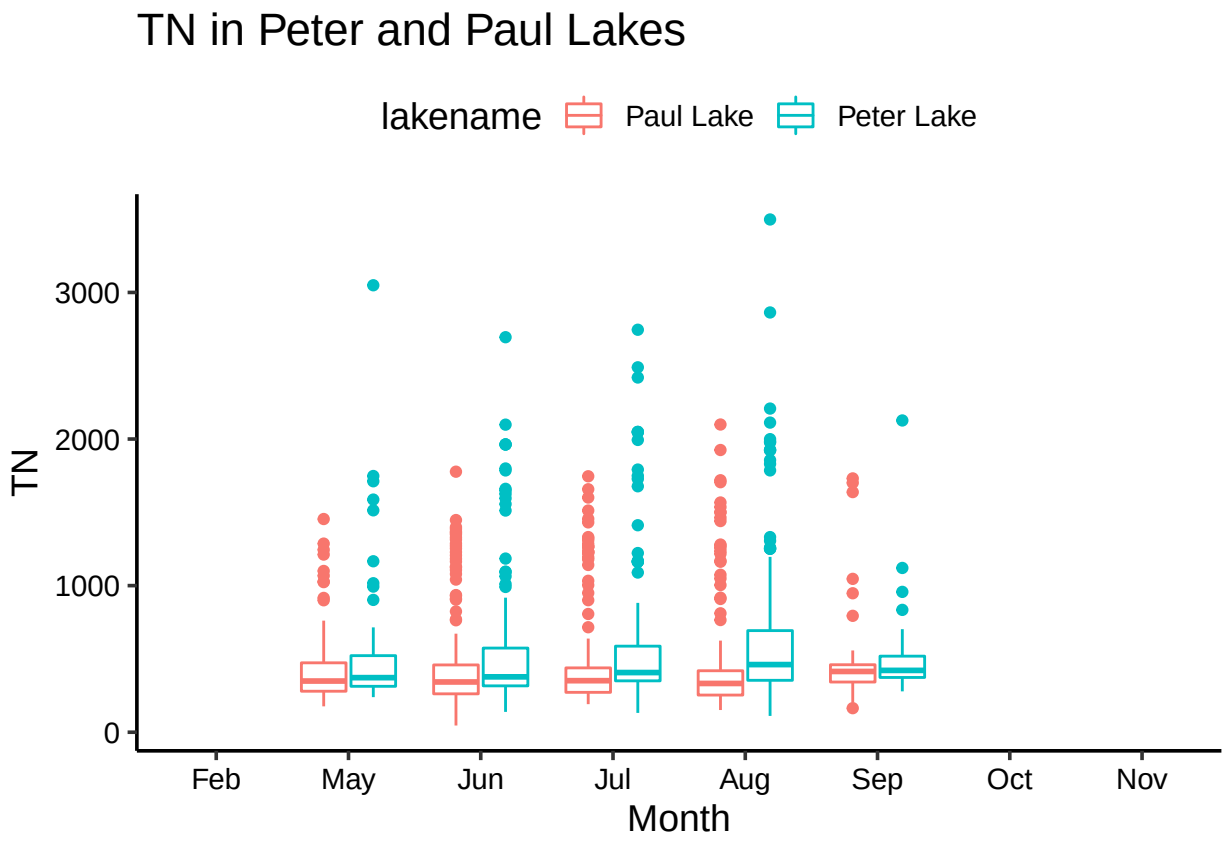

```
# boxplot of TP
PeterPaul.TP.plot <- ggplot(PeterPaul.Lake, aes(x = factor(month, levels = c(1:12),
    labels = month.abb), y = tp_ug)) + geom_boxplot(aes(color = lakename)) + ggtitle("TP in Peter and Pa
    xlab("Month") + ylab("TP") + scale_x_discrete(drop = FALSE)
print(PeterPaul.TP.plot)
```

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




```
# boxplot of TN
PeterPaul.TN.plot <- ggplot(PeterPaul.Lake, aes(x = factor(month, levels = c(1:12),
    labels = month.abb), y = tn_ug)) + geom_boxplot(aes(color = lakename)) + ggtitle("TN in Peter and Pa
    xlab("Month") + ylab("TN")
print(PeterPaul.TN.plot)
```

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



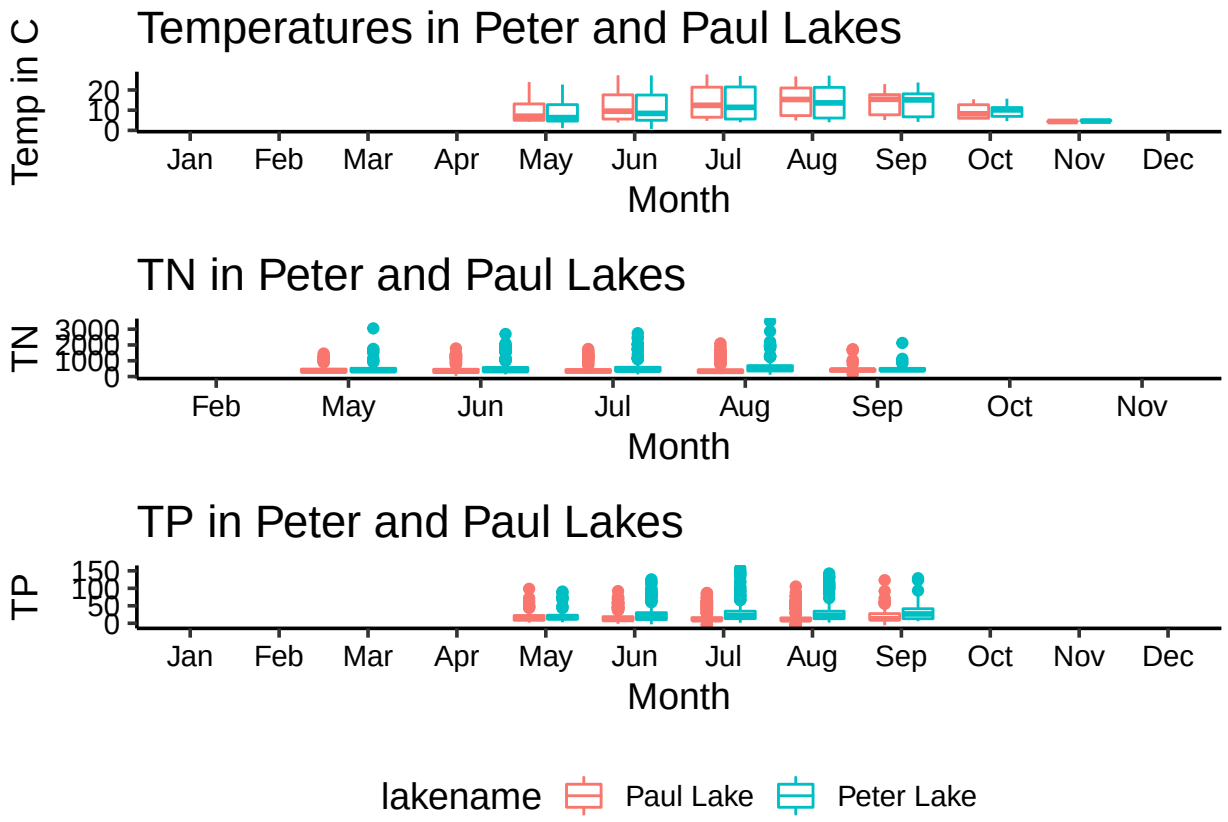

```
# creating cowplot combining the three graphs
plot_grid(PeterPaul.temperature.plot + theme(legend.position = "none"), PeterPaul.TN.plot +
    theme(legend.position = "none"), PeterPaul.TP.plot + theme(legend.position = "bottom"),
    nrow = 3, align = "hv", rel_heights = c(1, 1, 1.5), rel_widths = 1)
```

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

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

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

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



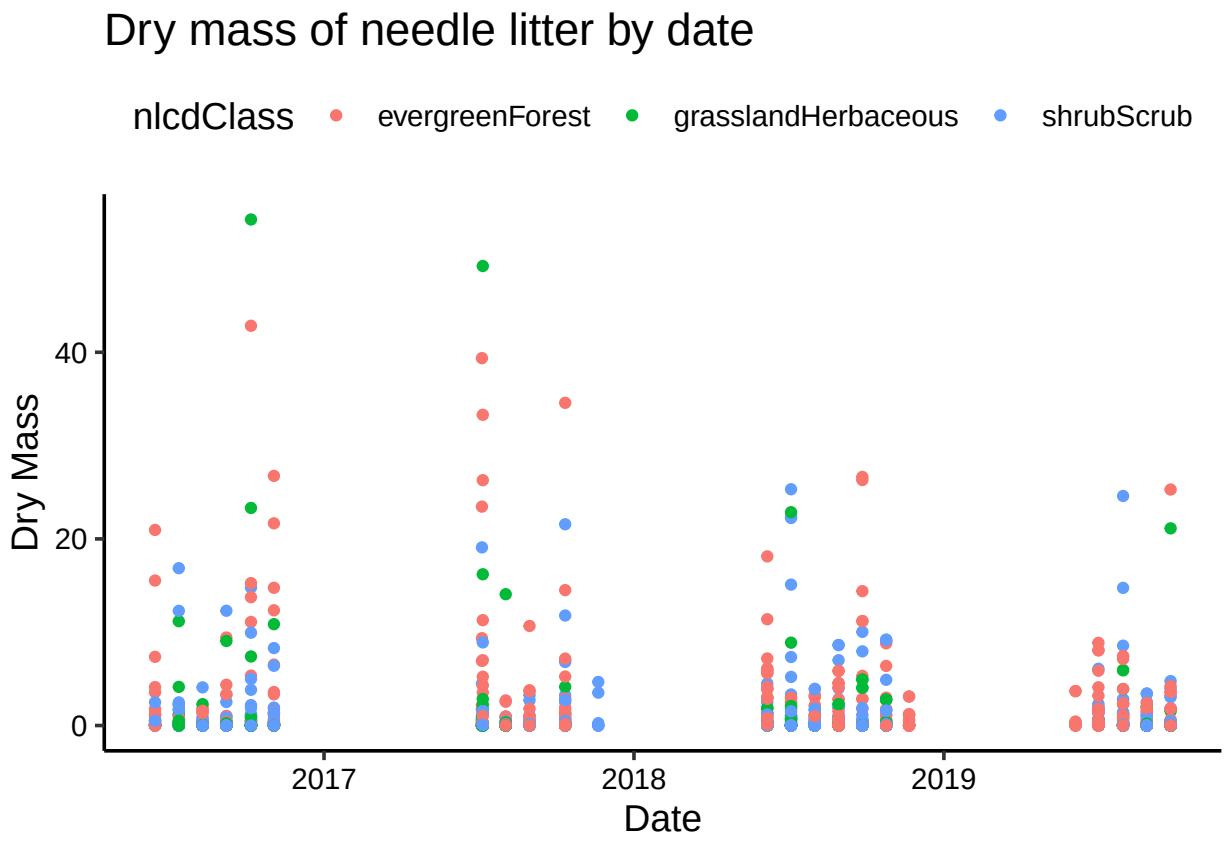
Question: What do you observe about the variables of interest over seasons and between lakes?

> Answer: All three variables of interest are most prevalent during summer months, with little to no appearance during the winter months. Levels for all three variables of interest are highest in July and August. Between the two lakes, Peter Lake generally has higher levels across the three variables of interest in nearly every month (with the exception of Temperature where Paul Lake shows slightly greater levels in some months).

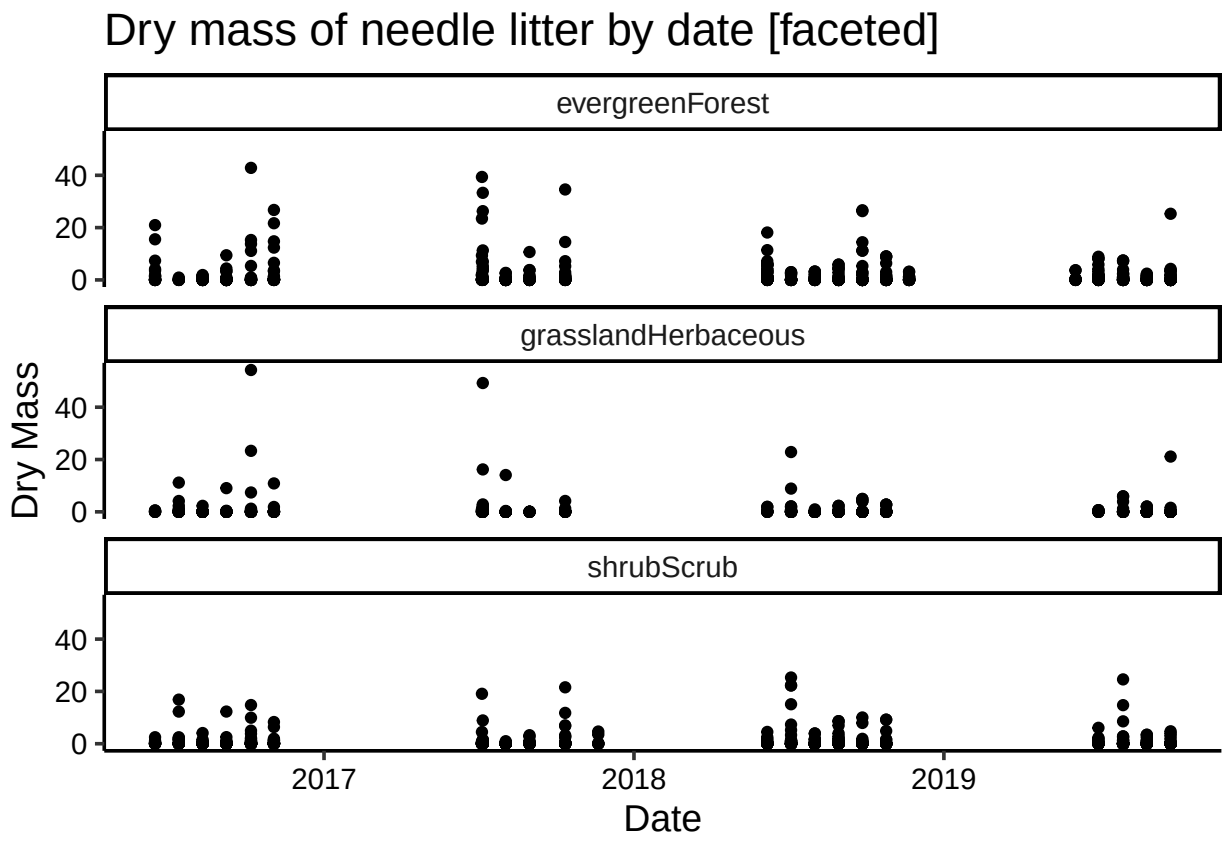
6. [Niwot Ridge] Plot a subset of the litter dataset by displaying only the "Needles" functional group. Plot the dry mass of needle litter by date and separate by NLCD class with a color aesthetic. (no need to adjust the name of each land use)

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

```
# 6

# creating Needles plot by nlcd color
Litter.Needles.plot <- ggplot(Litter, aes(x = collectDate, y = dryMass, color = nlcdClass)) +
    geom_point() + ggtitle("Dry mass of needle litter by date") + xlab("Date") +
    ylab("Dry Mass")
print(Litter.Needles.plot)
```




```
# 7

# creating Needles plot by nlcd faceted
Litter.Needles.facet.plot <- ggplot(Litter, aes(x = collectDate, y = dryMass)) +
    geom_point() + facet_wrap(vars(nlcdClass), nrow = 3) + ggtitle("Dry mass of needle litter by date [
    xlab("Date") + ylab("Dry Mass")
print(Litter.Needles.facet.plot)
```



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

Answer: I think the plot in #6 is more effective because it shows the data points from all NLCD classes on one plot. Because each class has its own color, it's easier to compare the different NLCD classes against each other than it is in the faceted plot where each class has its own graph.