Assignment 5: Data Visualization

Livia Hoxha

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.

```
#1

#install.packages("cowplot")

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 1.9.3     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 error
```

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

```
## here() starts at C:/Users/Lenovo/Desktop/EDE_Fall2023
```

```r
library(cowplot)
```

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

```r
# I set the working directory
getwd()
```

```
## [1] "C:/Users/Lenovo/Desktop/EDE_Fall2023"
```

```r
here()
```

```
## [1] "C:/Users/Lenovo/Desktop/EDE_Fall2023"
```

```r
# I kept encountering the "No such file or directory
#error, so I verified the file path with the following code
list.files()
```

```
## [1] "Assignments"        "Data"                "EDE_Fall2023.Rproj"
## [4] "Lessons"            "Processed_KEY.zip"  "README.md"
```

```r
# Here I read in NTL-LTER processed data files for nutrients and
#chemistry/physics for Peter and Paul Lakes
peter_paul_data <-
  read.csv(
    "Data/Processed_KEY/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv",
    stringsAsFactors = TRUE)


# Now I read in the processed data file for the Niwot Ridge litter dataset
niwot_litter_data <-
  read.csv(
    "Data/Processed_KEY/NEON_NIWO_Litter_mass_trap_Processed.csv",
    stringsAsFactors = TRUE)

#2

#First we check the structure of the data
str(peter_paul_data)
```

```
## ’data.frame’:    23008 obs. of  15 variables:
##  $ lakename       : Factor w/ 2 levels "Paul Lake","Peter Lake": 1 1 1 1 1 1 1 1 1 1 ...
##  $ year4          : int  1984 1984 1984 1984 1984 1984 1984 1984 1984 1984 ...
##  $ daynum         : int  148 148 148 148 148 148 148 148 148 148 ...
##  $ month          : int  5 5 5 5 5 5 5 5 5 5 ...
##  $ sampledate     : Factor w/ 1103 levels "1984-05-27","1984-05-28",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ depth          : num  0 0.25 0.5 0.75 1 1.5 2 3 4 5 ...
##  $ temperature_C  : num  14.5 NA NA NA 14.5 NA 14.2 11 7 6.1 ...
##  $ dissolvedOxygen: num  9.5 NA NA NA 8.8 NA 8.6 11.5 11.9 2.5 ...
##  $ irradianceWater: num  1750 1550 1150 975 870 610 420 220 100 34 ...
##  $ irradianceDeck : num  1620 1620 1620 1620 1620 1620 1620 1620 1620 1620 ...
##  $ tn_ug          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ tp_ug          : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ nh34           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ no23           : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ po4            : num  NA NA NA NA NA NA NA NA NA NA ...
```

```
str(niwot_litter_data)
```

```
## ’data.frame’:    1692 obs. of  13 variables:
##  $ plotID          : Factor w/ 12 levels "NIWO_040","NIWO_041",..: 9 8 9 11 7 7 4 4 4 4 ...
##  $ trapID          : Factor w/ 15 levels "NIWO_040_139",..: 11 10 11 13 9 9 5 5 5 5 ...
##  $ collectDate     : Factor w/ 24 levels "2016-06-16","2016-07-14",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ functionalGroup : Factor w/ 8 levels "Flowers","Leaves",..: 6 5 8 6 4 2 2 6 7 8 ...
##  $ dryMass         : num  0 0.27 0.12 0 1.11 0 0 0 0.07 0.02 ...
##  $ qaDryMass       : Factor w/ 2 levels "N","Y": 1 1 1 1 2 1 1 1 1 1 ...
##  $ subplotID       : int  31 41 31 32 32 32 40 40 40 40 ...
##  $ decimalLatitude : num  40.1 40 40.1 40 40 ...
##  $ decimalLongitude: num  -106 -106 -106 -106 -106 ...
##  $ elevation       : num  3477 3413 3477 3373 3446 ...
##  $ nlcdClass       : Factor w/ 3 levels "evergreenForest",..: 3 1 3 1 3 3 2 2 2 2 ...
##  $ plotType        : Factor w/ 1 level "tower": 1 1 1 1 1 1 1 1 1 1 ...
##  $ geodeticDatum   : Factor w/ 1 level "WGS84": 1 1 1 1 1 1 1 1 1 1 ...
```

```
# Since we see that the format of the dates is factor,
#we convert date columns to date format in the NTL-LTER data
peter_paul_data$sampledate <- as.Date(peter_paul_data$sampledate, format = "%Y-%m-%d")

# Convert date columns to date format in the Niwot Ridge data
niwot_litter_data$collectDate <- as.Date(niwot_litter_data$collectDate, format = "%Y-%m-%d")
```

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)

# Here I define my custom theme with updated colors
my_custom_theme <- function() {
  theme_minimal() +
    theme(
      # Customize the plot background
      plot.background = element_rect(fill = "lightgray"),

      # Customize the plot title
      plot.title = element_text(color = "darkblue", size = 16, face = "bold"),

      # Customize the axis labels
      axis.title = element_text(color = "darkgreen", size = 14, face = "italic"),

      # Customize axis ticks/gridlines
      axis.text = element_text(color = "purple", size = 12),
      axis.line = element_line(color = "darkorange"),
      axis.ticks = element_line(color = "darkred"),

      # Customize the legend
      legend.title = element_text(color = "darkblue", size = 12, face = "bold"),
      legend.text = element_text(color = "darkred", size = 10)
    )
} #I customized the look of all of the previous, even though
#we only had to do two, because I
#wanted to experiment with all of them and to see
#different color combinations

# Here I set my custom theme as the default theme
theme_set(my_custom_theme())
```

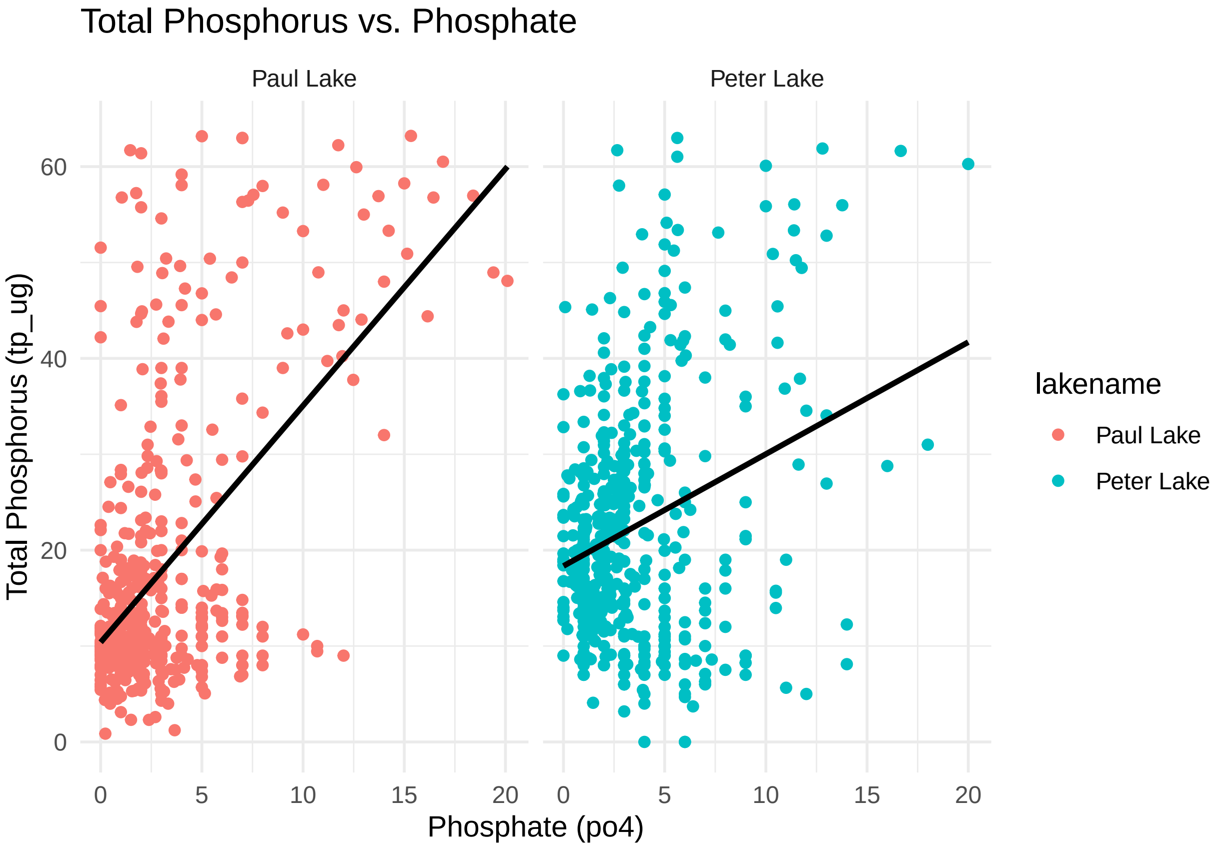
Create graphs

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

# I load the dplyr library
library(dplyr)

# Now I create separate plots for Peter and Paul lakes
ggplot(peter_paul_data, aes(x = po4, y = tp_ug, color = lakename)) +
  geom_point() +
  geom_smooth(method = "lm", formula = y ~ x, color = "black", se = FALSE) +
  labs(
    title = "Total Phosphorus vs. Phosphate",
    x = "Phosphate (po4)",
    y = "Total Phosphorus (tp_ug)"
  ) +
  scale_y_continuous(limits = c(0, quantile(peter_paul_data$tp_ug, 0.95, na.rm = TRUE))) +
  scale_x_continuous(limits = c(0, quantile(peter_paul_data$po4, 0.95, na.rm = TRUE))) +
  theme_minimal() +
  facet_wrap(~ lakename, ncol = 2)  # Separate plots for Peter and Paul
```

```
## Warning: Removed 22012 rows containing non-finite values (‘stat_smooth()‘).

## Warning: Removed 22012 rows containing missing values (‘geom_point()‘).
```



```
#I chose the theme_minimal, after receiving approval
#from Professor John, since my custom theme's color combinations
#were not aesthetically pleasing
```

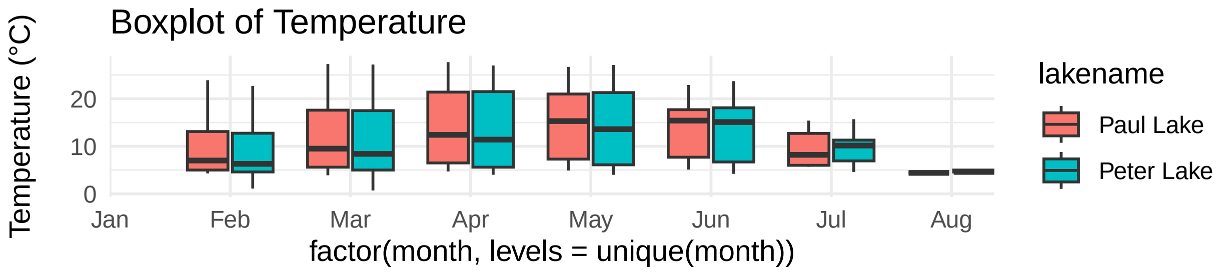
5. [NTL-LTER] Make three separate boxplots of (a) temperature, (b) TP, and (c) TN, with month as the x axis and lake as a color aesthetic. Then, create a cowplot that combines the three graphs. Make sure that only one legend is present and that graph axes are aligned.

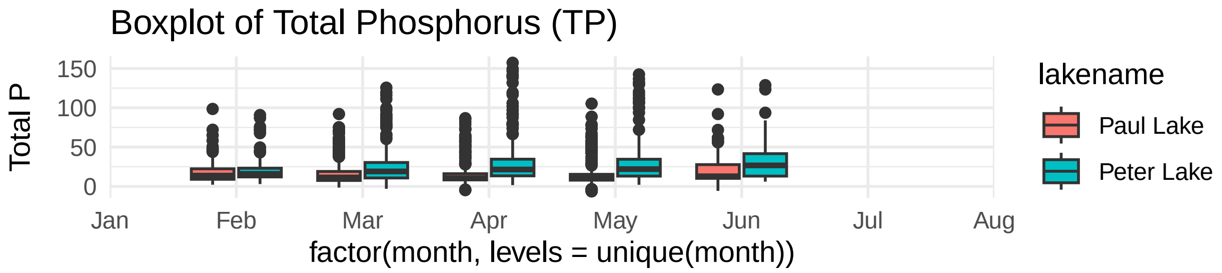
Tip: * Recall the discussion on factors in the previous section as it may be helpful here. * R has a built-in variable called `month.abb` that returns a list of months;see https://r-lang.com/month-abb-in-r-with-example

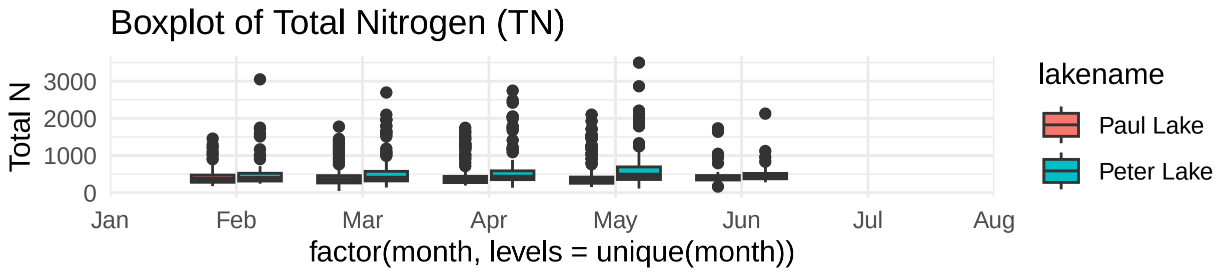
```
#5

# I create a data frame with all months
all_months <- data.frame(month = unique(peter_paul_data$month))

# Here I merge my original data with the data frame containing all months
peter_paul_data <- merge(all_months, peter_paul_data, by = "month", all.x = TRUE)

# Now I define a vector of abbreviated month labels
month_labels <- c("Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec")

# Next, I modify my boxplot code
boxplot_temp <- ggplot(peter_paul_data, aes
                       (x = factor(month, levels = unique(month)),
                         y = temperature_C, fill = lakename)) +
  geom_boxplot() +
  labs(title = "Boxplot of Temperature", y = "Temperature (°C)") +
  theme_minimal() +
  scale_x_discrete(expand = c(0, 0), labels = month_labels)

boxplot_tp <- ggplot(peter_paul_data,
                     aes(x = factor(month, levels = unique(month)),
                         y = tp_ug, fill = lakename)) +
  geom_boxplot() +
  labs(title = "Boxplot of Total Phosphorus (TP)", y = "Total P") +
  theme_minimal() +
  scale_x_discrete(expand = c(0, 0), labels = month_labels)

boxplot_tn <- ggplot(peter_paul_data,
                     aes(x = factor(month, levels = unique(month)),
                         y = tn_ug, fill = lakename)) +
  geom_boxplot() +
  labs(title = "Boxplot of Total Nitrogen (TN)", y = "Total N") +
  theme_minimal() +
  scale_x_discrete(expand = c(0, 0), labels = month_labels)

# Here I combine the three boxplots into a single cowplot with one legend and aligned axes
combined_plot <- plot_grid(boxplot_temp, boxplot_tp,
                           boxplot_tn, ncol = 1, align = "v",
                           rel_heights = c(1, 1, 1))
```

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

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

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

```
# Finally, I display the combined plot
print(combined_plot)
```





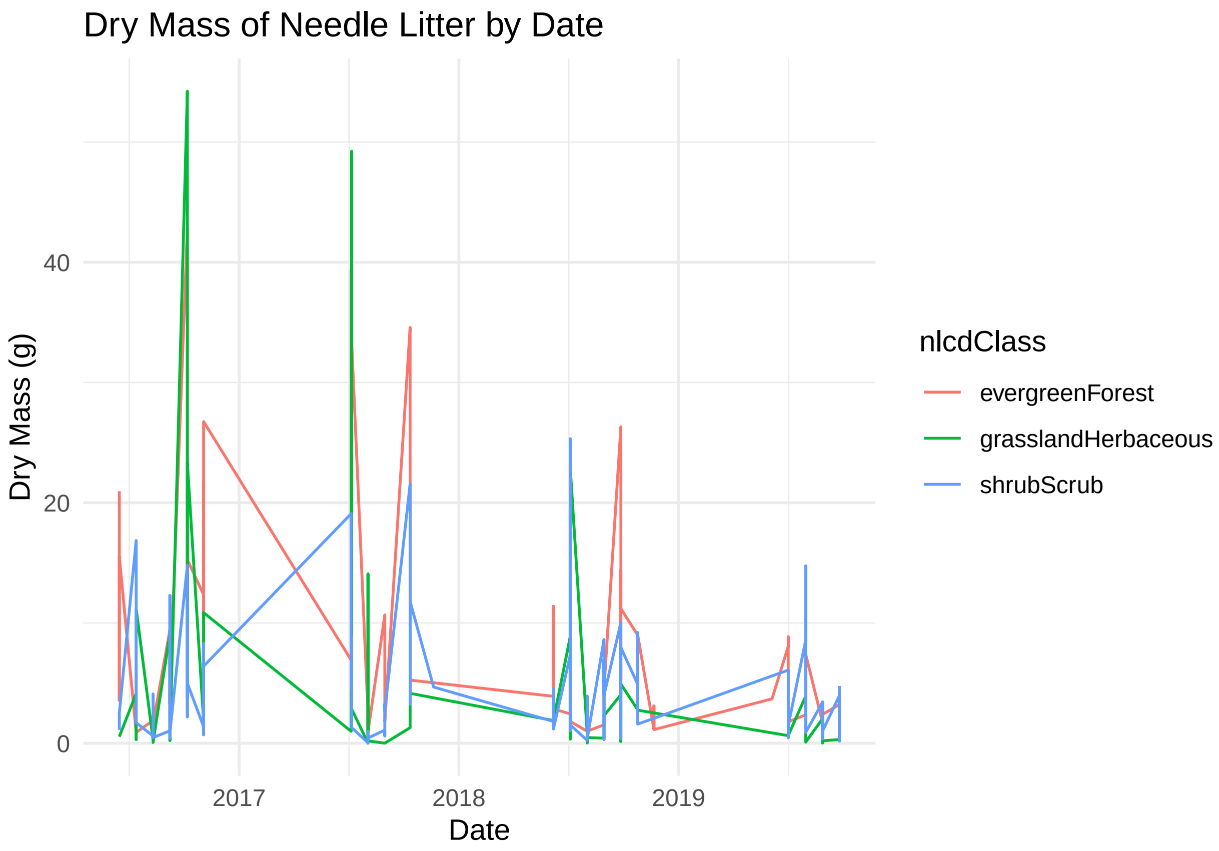


Question: What do you observe about the variables of interest over seasons and between lakes?

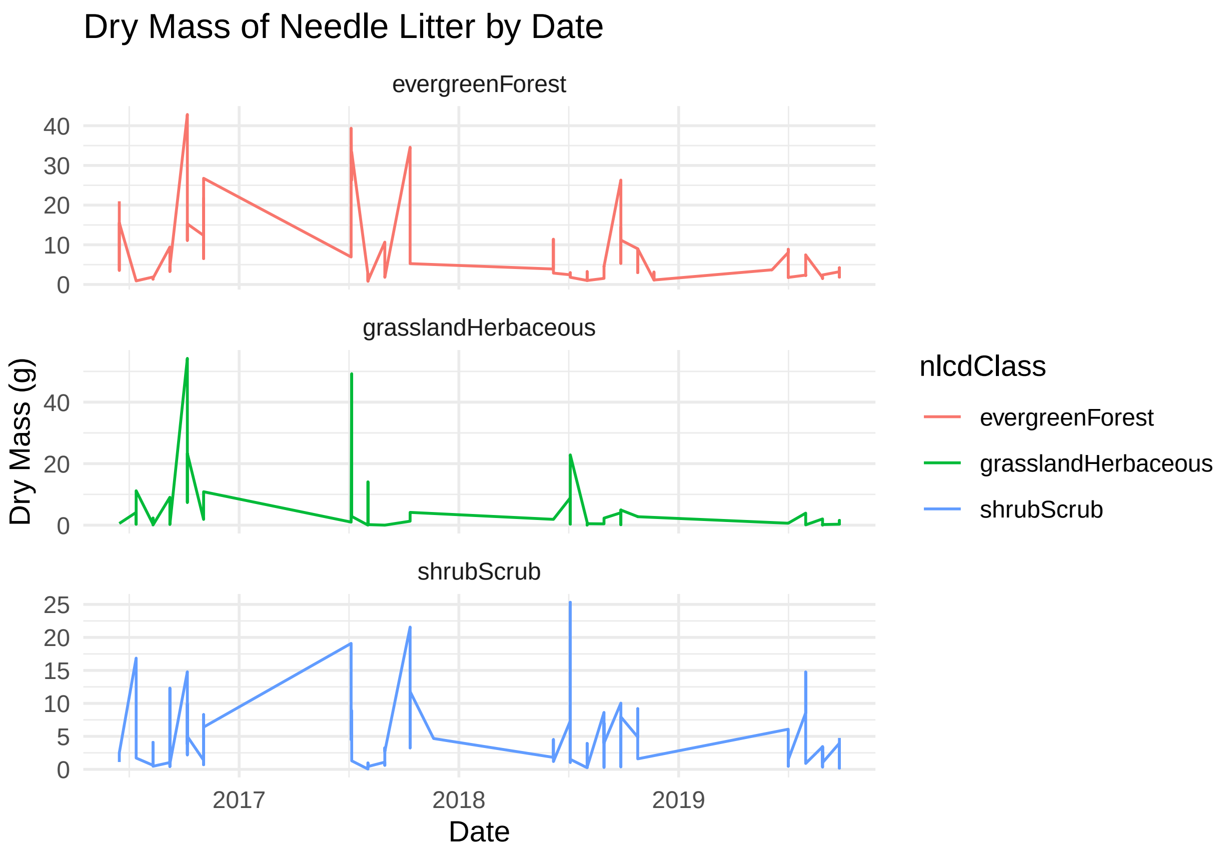
> Answer: From the boxplot of Temperature, we can see that for all the months, except for July, Paul Lake has higher temperatures in general than Peter Lake. Regarding the boxplot of Total Phosphorus, we notice that Paul Lake has lower Phosphorus levels. Lastly, regarding the Boxplot of Total Nitrogen, when we zoom in, we can see that Paul Lake has lower levels of 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.

```
#6

# Here I filter the dataset to include only the "Needles" functional group
needles_subset <- niwot_litter_data %>%
  filter(functionalGroup == "Needles")

# Now I create the plot
ggplot(needles_subset, aes(x = collectDate, y = dryMass, color = nlcdClass)) +
  geom_line() +
  labs(
    title = "Dry Mass of Needle Litter by Date",
    x = "Date",
    y = "Dry Mass (g)"
  ) +
  theme_minimal()
```



```
#7

# I filter the dataset to include only the "Needles" functional group
needles_subset <- niwot_litter_data %>%
  filter(functionalGroup == "Needles")

# Now I create the plot with facets
ggplot(needles_subset, aes(x = collectDate, y = dryMass)) +
  geom_line(aes(color = nlcdClass)) +
  labs(
    title = "Dry Mass of Needle Litter by Date",
    x = "Date",
    y = "Dry Mass (g)"
  ) +
  facet_wrap(~ nlcdClass, scales = "free_y", ncol = 1) +  # Separate by nlcdClass into facets
  theme_minimal()
```



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

> Answer: Visually, Plot 7 is more appealing and easier to understand, and we can see clearly the oscillations of the dry mass between different years. However, when we have a small number of unique NLCD classes (only 3) and if we want to see trends within each class, plot 6 is better than plot 7.