Assignment 5: Data Visualization

Yufan Du

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
# Load necessary libraries
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
```

```r
library(cowplot)
```

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

```r
# Verify home directory
here::here()
```

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

```r
# Set the file paths for the datasets
peter_data <- read_csv(here("Data","Processed","Processed_KEY", "NTL-LTER_Lake_Chemistry_Nutrients_Pete
```

```
## Rows: 23008 Columns: 15
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr   (1): lakename
## dbl  (13): year4, daynum, month, depth, temperature_C, dissolvedOxygen, irra...
## date  (1): sampledate
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
niwo_data <- read_csv(here("Data","Processed","Processed_KEY", "NEON_NIWO_Litter_mass_trap_Processed.cs
```

```
## Rows: 1692 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (7): plotID, trapID, functionalGroup, qaDryMass, nlcdClass, plotType, g...
## dbl  (5): dryMass, subplotID, decimalLatitude, decimalLongitude, elevation
## date (1): collectDate
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
#2
# Check the structure of the data to see how the date columns are being read
str(peter_data)
```

```
## spc_tbl_ [23,008 x 15] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ lakename       : chr [1:23008] "Paul Lake" "Paul Lake" "Paul Lake" "Paul Lake" ...
##  $ year4          : num [1:23008] 1984 1984 1984 1984 1984 ...
##  $ daynum         : num [1:23008] 148 148 148 148 148 148 148 148 148 148 ...
##  $ month          : num [1:23008] 5 5 5 5 5 5 5 5 5 5 ...
##  $ sampledate     : Date[1:23008], format: "1984-05-27" "1984-05-27" ...
##  $ depth          : num [1:23008] 0 0.25 0.5 0.75 1 1.5 2 3 4 5 ...
##  $ temperature_C  : num [1:23008] 14.5 NA NA NA 14.5 NA 14.2 11 7 6.1 ...
##  $ dissolvedOxygen: num [1:23008] 9.5 NA NA NA 8.8 NA 8.6 11.5 11.9 2.5 ...
##  $ irradianceWater: num [1:23008] 1750 1550 1150 975 870 610 420 220 100 34 ...
##  $ irradianceDeck : num [1:23008] 1620 1620 1620 1620 1620 1620 1620 1620 1620 1620 ...
##  $ tn_ug          : num [1:23008] NA NA NA NA NA NA NA NA NA NA ...
##  $ tp_ug          : num [1:23008] NA NA NA NA NA NA NA NA NA NA ...
##  $ nh34           : num [1:23008] NA NA NA NA NA NA NA NA NA NA ...
##  $ no23           : num [1:23008] NA NA NA NA NA NA NA NA NA NA ...
##  $ po4            : num [1:23008] NA NA NA NA NA NA NA NA NA NA ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   lakename = col_character(),
##   ..   year4 = col_double(),
##   ..   daynum = col_double(),
##   ..   month = col_double(),
##   ..   sampledate = col_date(format = ""),
##   ..   depth = col_double(),
##   ..   temperature_C = col_double(),
##   ..   dissolvedOxygen = col_double(),
##   ..   irradianceWater = col_double(),
##   ..   irradianceDeck = col_double(),
##   ..   tn_ug = col_double(),
##   ..   tp_ug = col_double(),
##   ..   nh34 = col_double(),
##   ..   no23 = col_double(),
##   ..   po4 = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```r
str(niwo_data)
```

```
## spc_tbl_ [1,692 x 13] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ plotID          : chr [1:1692] "NIWO_062" "NIWO_061" "NIWO_062" "NIWO_064" ...
##  $ trapID          : chr [1:1692] "NIWO_062_050" "NIWO_061_169" "NIWO_062_050" "NIWO_064_103" ...
##  $ collectDate     : Date[1:1692], format: "2016-06-16" "2016-06-16" ...
##  $ functionalGroup : chr [1:1692] "Seeds" "Other" "Woody material" "Seeds" ...
##  $ dryMass         : num [1:1692] 0 0.27 0.12 0 1.11 0 0 0 0.07 0.02 ...
##  $ qaDryMass       : chr [1:1692] "N" "N" "N" "N" ...
##  $ subplotID       : num [1:1692] 31 41 31 32 32 32 40 40 40 40 ...
##  $ decimalLatitude : num [1:1692] 40.1 40 40.1 40 40 ...
##  $ decimalLongitude: num [1:1692] -106 -106 -106 -106 -106 ...
##  $ elevation       : num [1:1692] 3477 3413 3477 3373 3446 ...
##  $ nlcdClass       : chr [1:1692] "shrubScrub" "evergreenForest" "shrubScrub" "evergreenForest" ...
##  $ plotType        : chr [1:1692] "tower" "tower" "tower" "tower" ...
##  $ geodeticDatum   : chr [1:1692] "WGS84" "WGS84" "WGS84" "WGS84" ...
##  - attr(*, "spec")=
##   .. cols(
```

```
##   ..   plotID = col_character(),
##   ..   trapID = col_character(),
##   ..   collectDate = col_date(format = ""),
##   ..   functionalGroup = col_character(),
##   ..   dryMass = col_double(),
##   ..   qaDryMass = col_character(),
##   ..   subplotID = col_double(),
##   ..   decimalLatitude = col_double(),
##   ..   decimalLongitude = col_double(),
##   ..   elevation = col_double(),
##   ..   nlcdClass = col_character(),
##   ..   plotType = col_character(),
##   ..   geodeticDatum = col_character()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

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

```r
#3
# Load necessary libraries
library(ggplot2)

# Define a custom theme
my_custom_theme <- theme(
  # Customize plot background
  plot.background = element_rect(fill = "lightblue", color = NA),

  # Customize plot title
  plot.title = element_text(size = 12, face = "bold", hjust = 0.5, color = "darkblue"),

  # Customize axis labels
  axis.title = element_text(size = 10, color = "darkblue"),

  # Customize axis ticks and gridlines
  axis.text = element_text(size = 8, color = "black"),
  axis.ticks = element_line(color = "black"),
  panel.grid.major = element_line(color = "gray80", linewidth = 0.5),
  panel.grid.minor = element_line(color = "gray90", linewidth = 0.25),

  # Customize legend
  legend.background = element_rect(fill = "white", color = "black"),
  legend.title = element_text(face = "bold"),
  legend.text = element_text(size = 12)
)
```

```
# Set the custom theme as the default
theme_set(theme_bw() + 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 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
#decide the extreme values to hide
summary(peter_data$po4)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##  -0.233   1.000   2.324   5.919   5.000 373.836   21822
```

```
summary(peter_data$tp_ug)
```

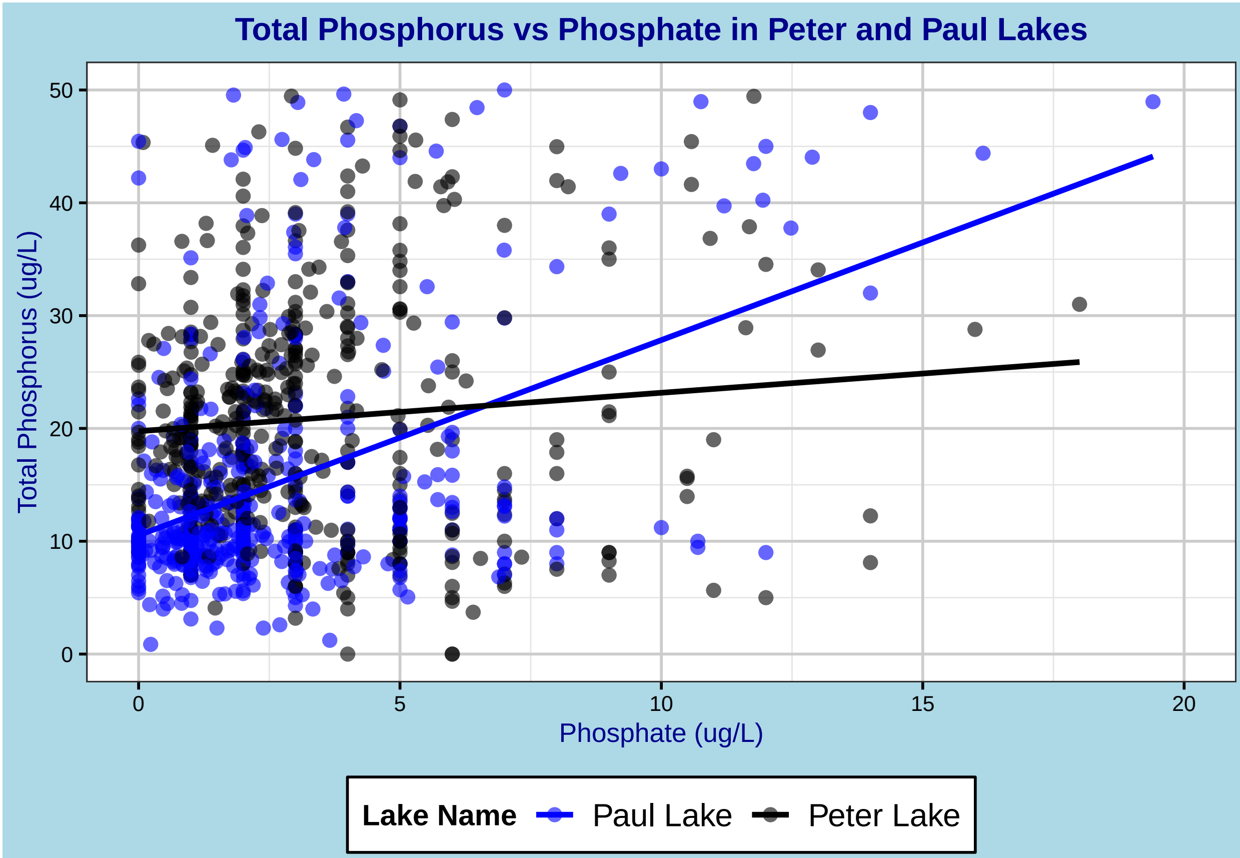
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##  -6.349   9.194  14.401  22.159  27.746 157.250   20729
```

```
#plot the graph
ggplot(peter_data, aes(x = po4, y = tp_ug, color = lakename)) +
  geom_point(alpha = 0.6, size = 2) +  # Add scatter plot points with transparency
  geom_smooth(method = "lm", se = FALSE) +  # Add linear regression line (line of best fit)
  scale_color_manual(values = c("Peter Lake" = "black", "Paul Lake" = "blue")) +  # Customize colors fo
  labs(title = "Total Phosphorus vs Phosphate in Peter and Paul Lakes",
       x = "Phosphate (ug/L)",
       y = "Total Phosphorus (ug/L)",
       color = "Lake Name") +
  xlim(0, 20) +  # Adjust x-axis limits to hide extreme values
  ylim(0, 50) +  # Adjust y-axis limits to hide extreme values
  theme(legend.position = "bottom")  # Move legend to bottom
```

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

```
## Warning: Removed 22067 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

```
## Warning: Removed 22067 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`.

```
#5

# Ensure the 'month' column is treated as a factor with numeric labels for proper ordering
peter_data$month <- factor(peter_data$month, levels = 1:12, labels = 1:12)

# Plot 1: Temperature boxplot
plot_temp <- ggplot(peter_data, aes(x = month, y = temperature_C, fill = lakename)) +
  geom_boxplot() +
  labs(title = "Temperature by Month", y = "Temperature (°C)") +
  theme(legend.position = "none", axis.title.x = element_blank())  # Remove x-axis title

# Plot 2: Total Phosphorus (TP) boxplot
plot_tp <- ggplot(peter_data, aes(x = month, y = tp_ug, fill = lakename)) +
  geom_boxplot() +
  labs(title = "Total Phosphorus by Month", y = "Total Phosphorus (ug/L)") +
  theme(legend.position = "none", axis.title.x = element_blank())  # Remove x-axis title
```

```r
# Plot 3: Total Nitrogen (TN) boxplot
plot_tn <- ggplot(peter_data, aes(x = month, y = tn_ug, fill = lakename)) +
  geom_boxplot() +
  labs(title = "Total Nitrogen by Month", x = "Month", y = "Total Nitrogen (ug/L)") +
  theme(legend.position = "none")  # Remove legend

# Extract the legend from one of the plots
legend_plot <- get_legend(
  ggplot(peter_data, aes(x = month, y = temperature_C, fill = lakename)) +
    geom_boxplot() +
    theme(legend.position = "bottom")  # Place legend at the bottom
)
```

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

## Warning in get_plot_component(plot, "guide-box"): Multiple components found;
## returning the first one. To return all, use `return_all = TRUE`.
```

```r
# Combine the three plots into one cowplot, ensuring axes are aligned
combined_plots <- plot_grid(
  plot_temp, plot_tp, plot_tn,
  ncol = 1,  # Arrange plots vertically
  align = "v"  # Align the y-axis
)
```

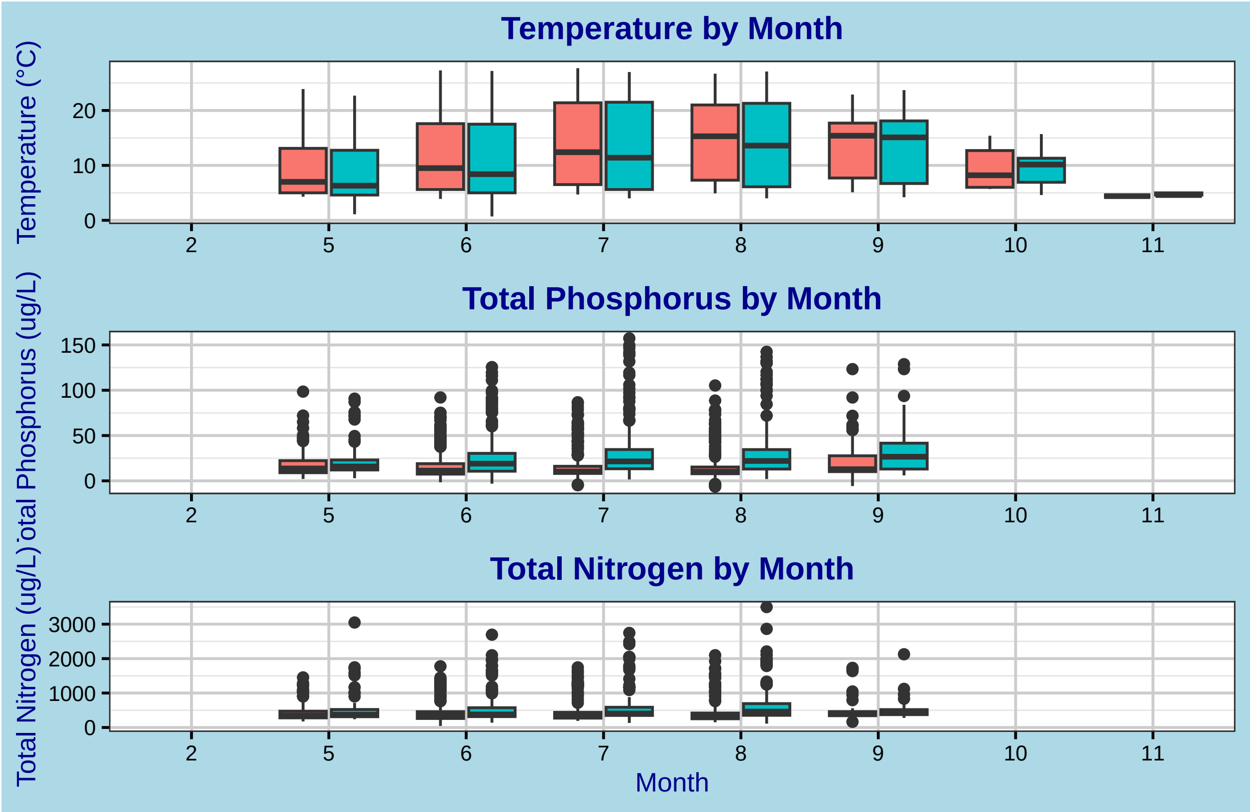
```
## Warning: Removed 3566 rows containing non-finite outside the scale range
## (`stat_boxplot()`).

## 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
# Add the shared legend at the bottom
final_plot <- plot_grid(
  combined_plots, legend_plot,
  ncol = 1,  # Place legend below the plots
  rel_heights = c(3, 0.2)  # Adjust the size ratio between the plots and the legend
)

# Display the final combined plot
print(final_plot)
```



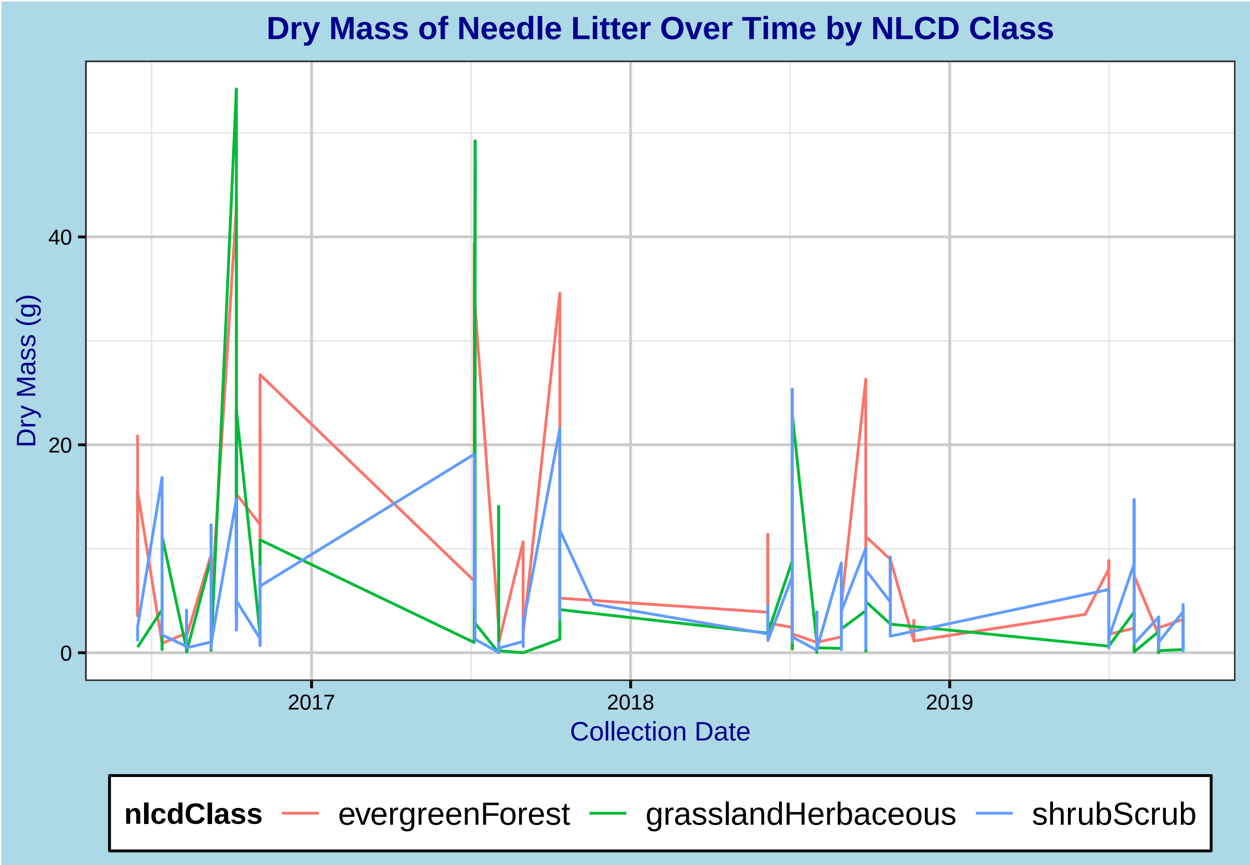
Question: What do you observe about the variables of interest over seasons and between lakes?

> Answer: Temperature: Shows a clear seasonal pattern, with higher values in warmer months (May–August) and lower in colder months. Both lakes exhibit similar temperature trends. Total Phosphorus (TP): TP levels vary across months without a strong seasonal pattern. There are notable differences between the lakes, with one lake having higher TP levels at certain times. Total Nitrogen (TN): TN shows some seasonal variation, with higher levels in late spring/summer. Differences between lakes suggest varying nutrient dynamics.

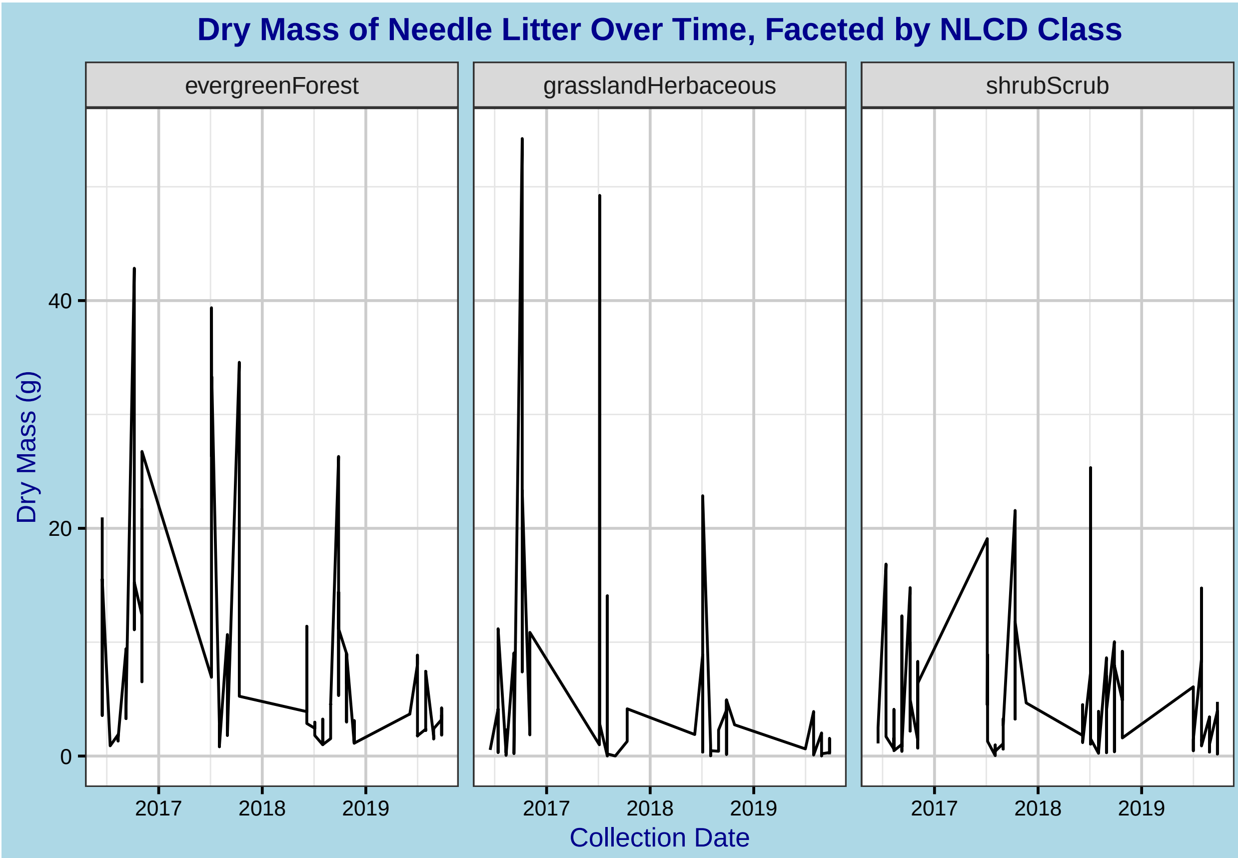
6. [Niwot Ridge] Plot a subset of the litter dataset by displaying only the "Needles" functional group. Plot the dry mass of needle litter by date and separate by NLCD class with a color aesthetic. (no need to adjust the name of each land use)
7. [Niwot Ridge] Now, plot the same plot but with NLCD classes separated into three facets rather than separated by color.

```
#6
# Filter the dataset for "Needles" functional group
needles_data <- niwo_data %>% filter(functionalGroup == "Needles")

# Plot: Dry mass of needle litter by date, colored by NLCD class
ggplot(needles_data, aes(x = collectDate, y = dryMass, color = nlcdClass)) +
  geom_line() +
  labs(title = "Dry Mass of Needle Litter Over Time by NLCD Class",
       x = "Collection Date",
       y = "Dry Mass (g)") +
  theme(legend.position = "bottom")
```




```r
#7
# Plot: Dry mass of needle litter by date, faceted by NLCD class
ggplot(needles_data, aes(x = collectDate, y = dryMass)) +
  geom_line() +
  facet_wrap(~nlcdClass, ncol = 3) +
  labs(title = "Dry Mass of Needle Litter Over Time, Faceted by NLCD Class",
       x = "Collection Date",
       y = "Dry Mass (g)") +
  theme(legend.position = "bottom")
```



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

Answer: 6. Becuase it's easier to compare in a same graph.