Assignment 5: Data Visualization

Reino Hyyppa

OVERVIEW

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

Set up your session

1. Set up your session. Verify your working directory and load the tidyverse and 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_PeterPaul_Processed.csv`] version) and the processed data file for the Niwot Ridge litter dataset (use the [`NEON_NIWO_Litter_mass_trap_Processed.csv`] version).

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

```
#1
# check wd
getwd()
```

```
## [1] "/Users/reinohyyppa/Desktop/Duke MEM/Spring 22 /ENV872/Environmental_Data_Analytics_2022"
```

```
# load packages
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.4     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.0.2     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
#install.packages("cowplot")
library(cowplot)

# initialize data set
PeterPaul_chem <- read.csv("./Data/Processed/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv"

Litter <- read.csv("./Data/Processed/NEON_NIWO_Litter_mass_trap_Processed.csv", stringsAsFactors = TRUE)
```

```
#2

# check date class

class(Litter$collectDate)
```

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

```
Litter$collectDate <- as.Date(Litter$collectDate, format = "%Y-%m-%d")

class(PeterPaul_chem$sampledate)
```

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

```
PeterPaul_chem$sampledate <- as.Date(PeterPaul_chem$sampledate, format = "%Y-%m-%d")
```

Define your theme

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

```
#3

# create a new plot theme
new_theme <- theme_classic(base_size = 12) +
  theme(axis.text = element_text(color = "black"),
  legend.position = "right")

theme_set(new_theme)
```

Create graphs

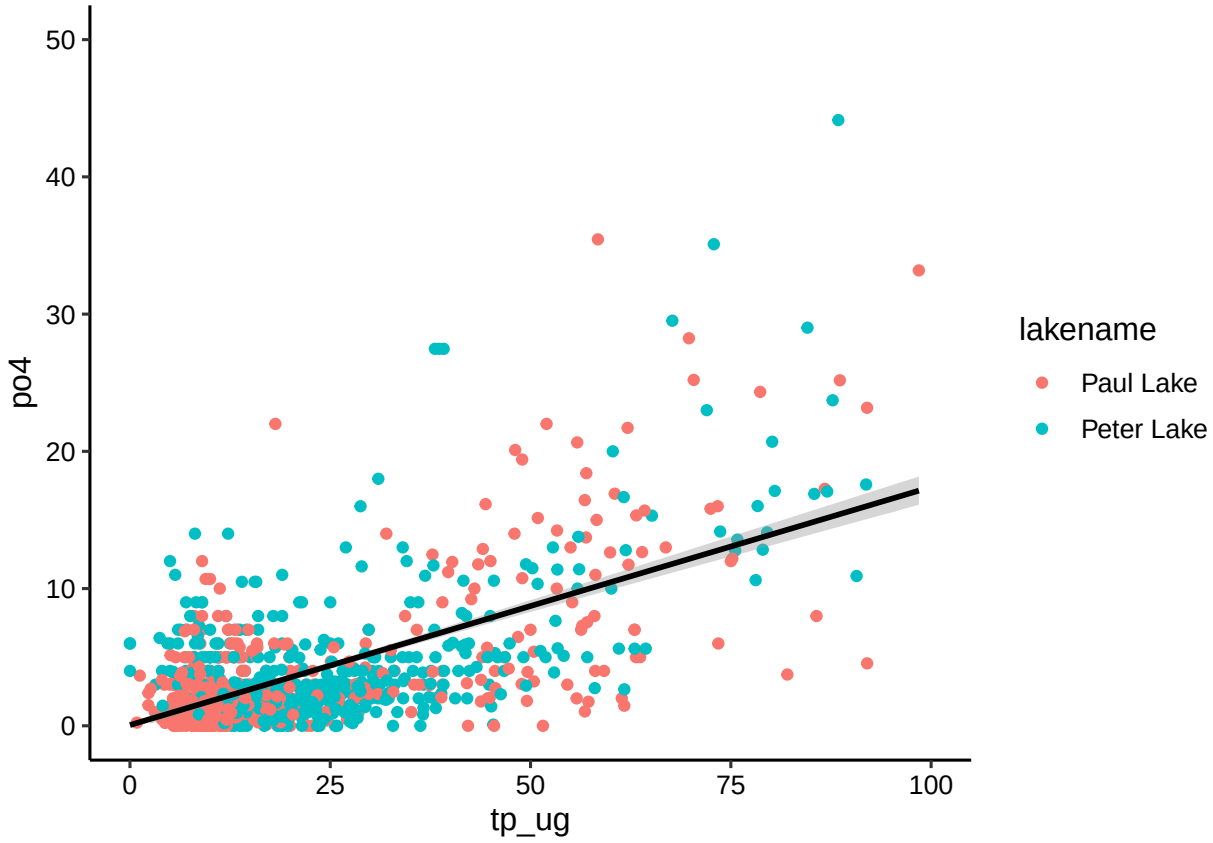
For numbers 4-7, create ggplot graphs and adjust aesthetics to follow best practices for data visualization. Ensure your theme, color palettes, axes, and additional aesthetics are edited accordingly.

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 `ylim()`).

```
#4
PhPh <-
  ggplot(PeterPaul_chem, aes(x = tp_ug, y = po4, color = lakename)) +
  geom_point() +
  geom_smooth(method = lm, color = "black") +
  xlim(0, 100) +
  ylim(0, 50)
print(PhPh)
```

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

```
## Warning: Removed 21964 rows containing non-finite values (stat_smooth).

## Warning: Removed 21964 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.

```
#5

# create plot with legend
create_legend <-
  ggplot(PeterPaul_chem) +
  geom_boxplot(aes(x= as.factor(month), y = temperature_C, color = lakename)) +
  theme(legend.position = "bottom")

# create boxplot of temperature w/out legend
temp_plot <-
  ggplot(PeterPaul_chem) +
  geom_boxplot(aes(x= as.factor(month), y = temperature_C, color = lakename)) +
  theme(legend.position = "none")

# create boxplot of TP w/out legend
TP_plot <-
   ggplot(PeterPaul_chem) +
```

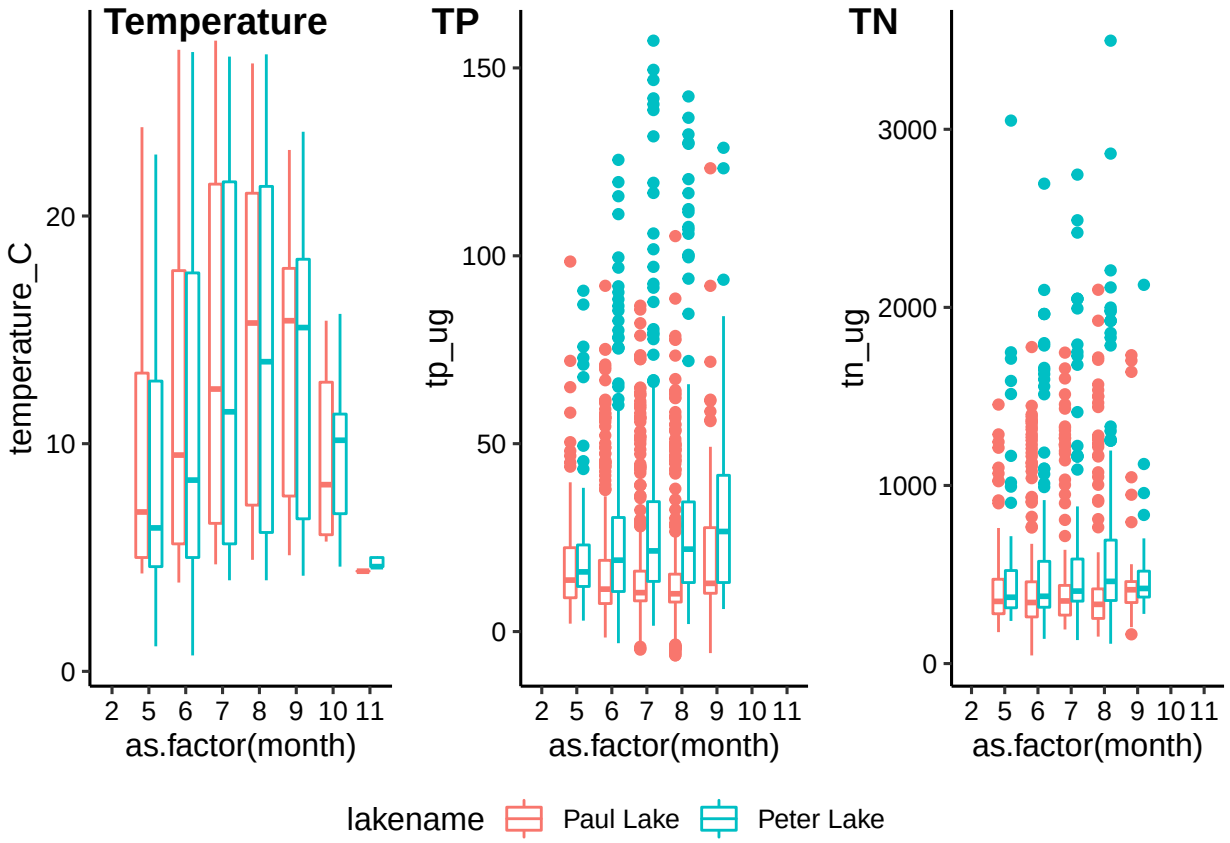
```
  geom_boxplot(aes(x= as.factor(month), y = tp_ug, color = lakename)) +
  theme(legend.position = "none")

# create boxplot of TN w/out legend
TN_plot <-
  ggplot(PeterPaul_chem) +
  geom_boxplot(aes(x= as.factor(month), y = tn_ug, color = lakename)) +
  theme(legend.position = "none")

# combine three plots (temp, TP, TN) using plot_grid
combined_plot <-
  plot_grid(temp_plot, TP_plot, TN_plot, nrow = 1,
            labels = c("Temperature", "TP", "TN"), rel_heights = c(3, 0.3))
```

```
## 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).
```

```
# create legend
plot_legend <- get_legend(create_legend)
```

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

```
# plot three plots with a master legend
final_plot <-
  plot_grid(combined_plot, plot_legend,
            nrow =2, rel_heights = c(3, 0.3))

# check results
print(final_plot)
```



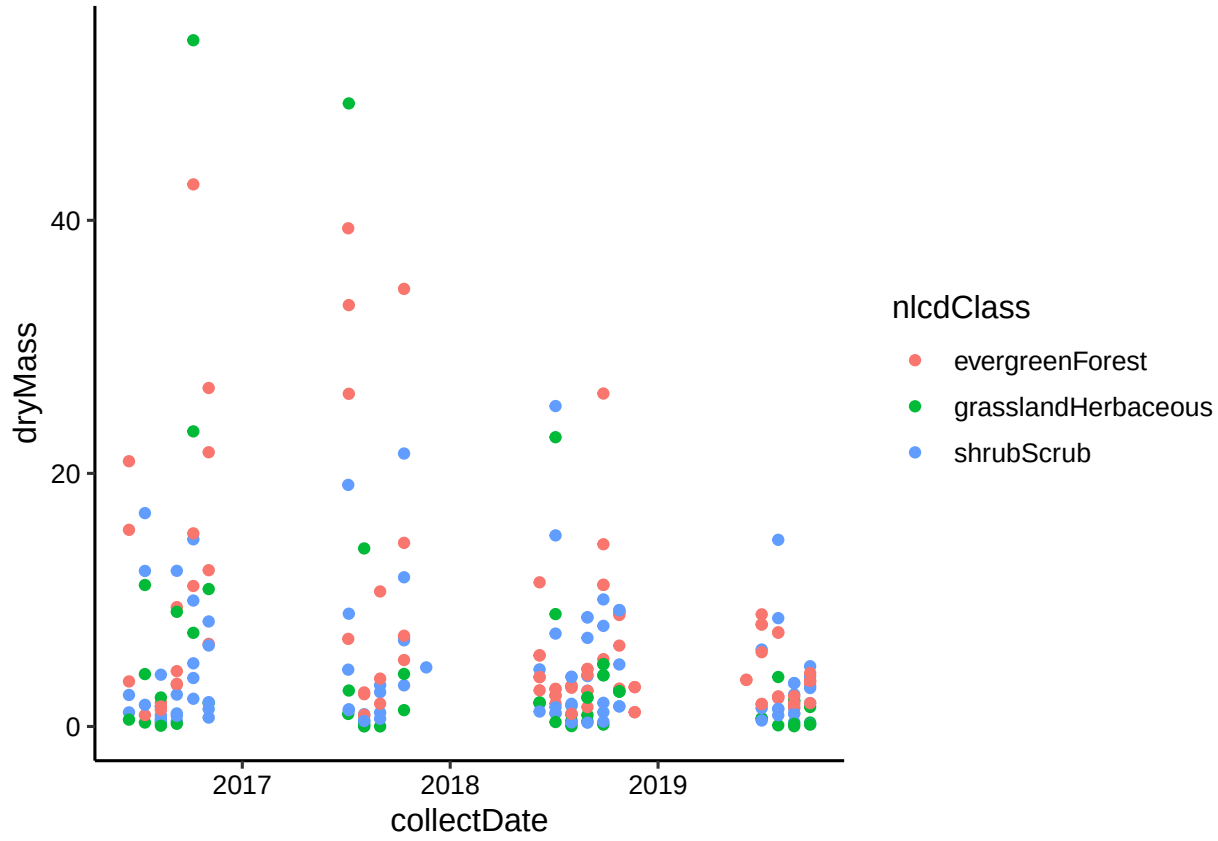
Question: What do you observe about the variables of interest over seasons and between lakes?

> Answer: Temperatures for each of the two lakes varies in relationship with one another over the different seasons, getting warmer during the summer months, and cooling off in the fall. TP averages differ more for Peter Lake than Paul Lake over the seasons with a larger spread in the data for Paul Lake. Based on the boxplot for TN, levels for this variable have less variance for the two lakes than TP. However, there seems to larger swings in TN levels for Peter Lake than Paul Lake.

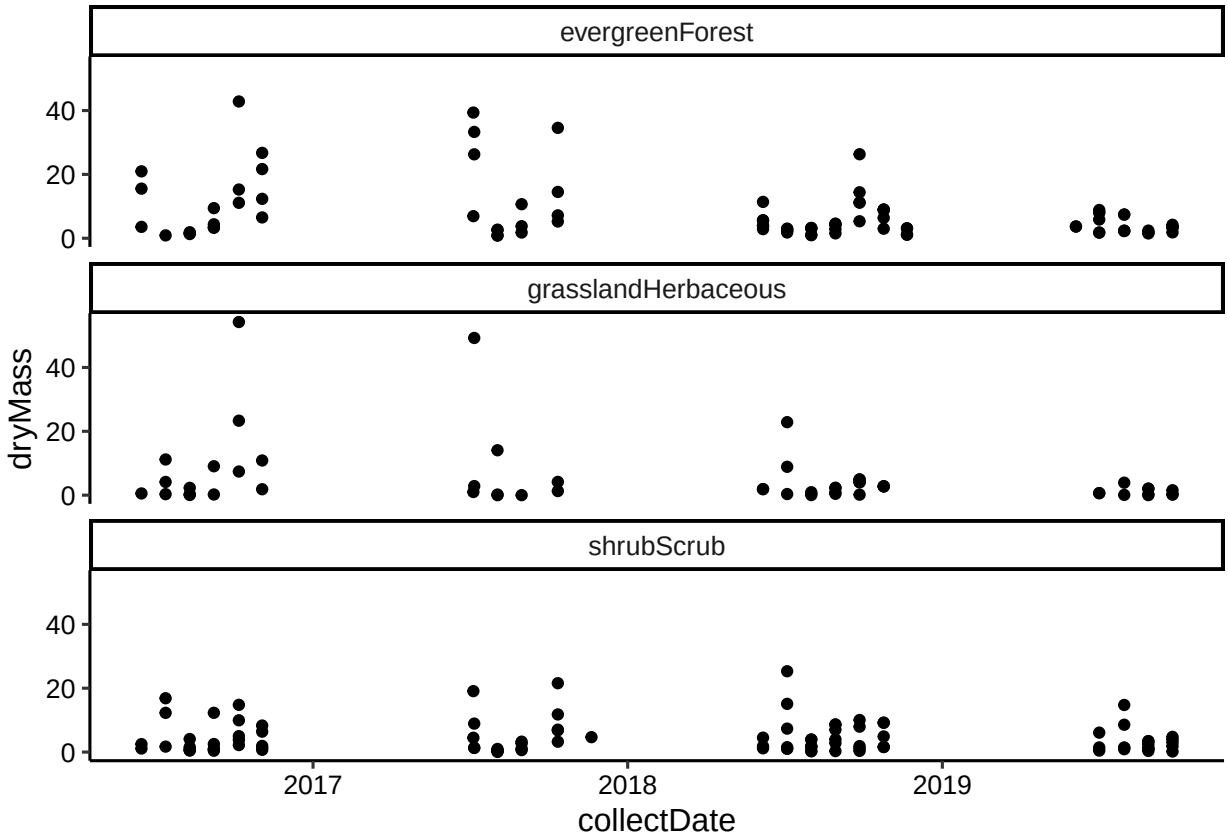
6. [Niwot Ridge] Plot a subset of the litter dataset by displaying only the "Needles" functional group. Plot the dry mass of needle litter by date and separate by NLCD class with a color aesthetic. (no need to adjust the name of each land use)

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

```
#6
# plot the dry mass of needle litter by date
needle_plot <-
  ggplot(subset(Litter, functionalGroup == "Needles"),
         aes(x = collectDate, y = dryMass, color = nlcdClass)) +
  geom_point()
print(needle_plot)
```




```
#7
# separate needle litter into three different facets
needle_plot_faceted <-
  ggplot(subset(Litter, functionalGroup == "Needles"),
         aes(x = collectDate, y = dryMass)) +
  geom_point() +
  facet_wrap(vars(nlcdClass), nrow = 3)
print(needle_plot_faceted)
```



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

Answer: I find it much easier to distinguish between different functional groups using the facet method. Rather than plotting all three types of needles on the same plot, the facet method plots each type of needle on a different plot. Then, you can compare the body mass for each needle type more easily and it is more visually appealing.