Assignment 5: Data Visualization

Student Name

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 Project setup
##Load packages
library(tidyverse);library(lubridate);library(here);library(cowplot)
```

```
## -- 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
## here() starts at /home/guest/EDE
##
##
## Attaching package: 'cowplot'
##
##
## The following object is masked from 'package:lubridate':
##
##     stamp
```

```
## Check working directory
here()
```

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

```
##Load datasets
NTL_tidy <- read.csv(
  here('Data/Processed_KEY/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv'),
  stringsAsFactors = T)
Trap_data <- read.csv(
  here('Data/Processed_KEY/NEON_NIWO_Litter_mass_trap_Processed.csv'),
  stringsAsFactors = T)

#2 Check/convert date columns
NTL_tidy$sampledate <- ymd(NTL_tidy$sampledate)
Trap_data$collectDate <- ymd(Trap_data$collectDate)
```

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 Create my theme
my_theme = theme_bw() +
  theme(
    plot.background = element_rect(
      fill = 'Light gray',
      color = 'Black'
    ),
    axis.line = element_line(
      linewidth = 1,
      color='Blue'
    ),
    axis.text = element_text(
      family='serif'
    ),
```

```
    plot.title = element_text(
      face = 'bold',
      color='Blue',
      family='serif'
    )
  )

#Set my theme to the default
theme_set(my_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.

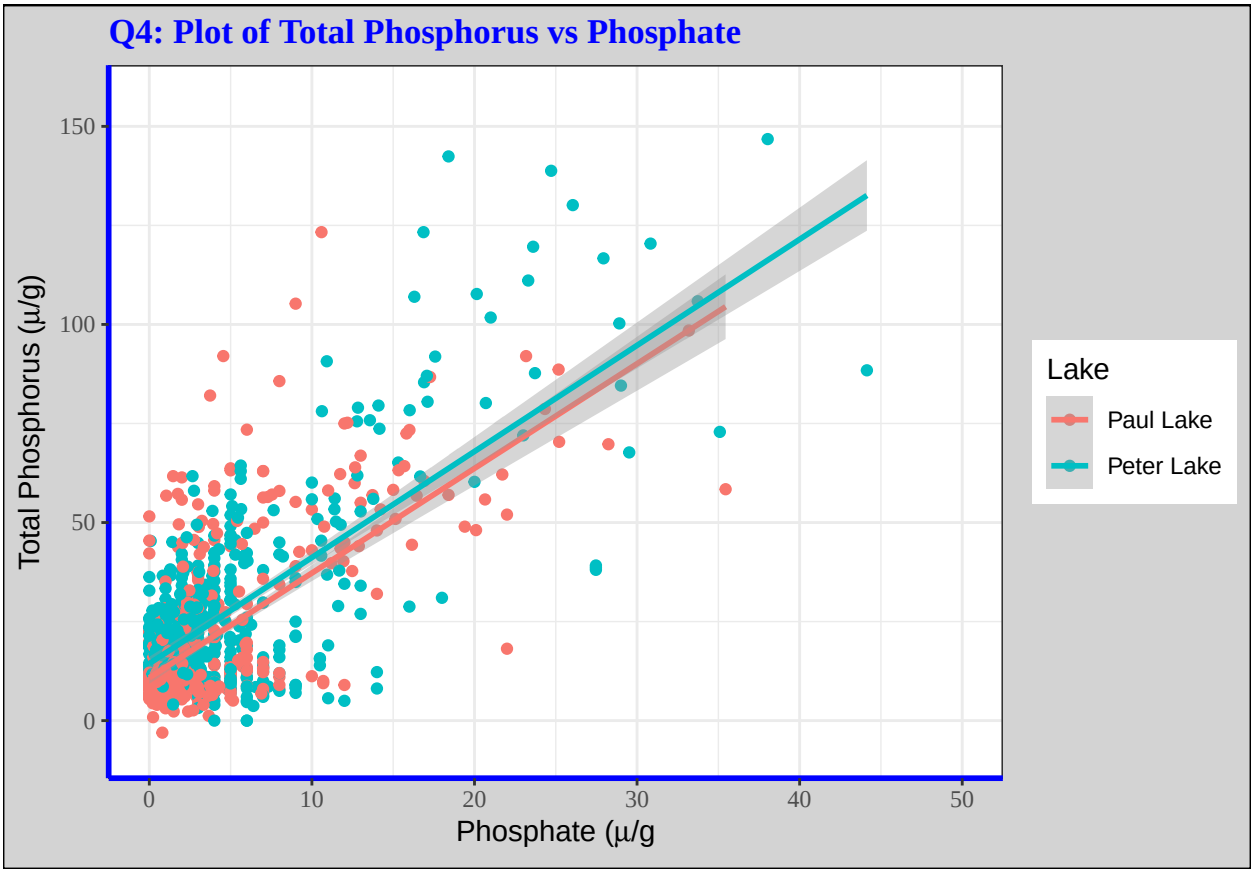
4. [NTL-LTER] Plot total phosphorus (`tp_ug`) by phosphate (`po4`), with separate aesthetics for Peter and Paul lakes. Add a line of best fit and color it black. Adjust your axes to hide extreme values (hint: change the limits using `xlim()` and/or `ylim()`).

```
#4 Plot total P vs PO4
ggplot(NTL_tidy,aes(x=po4,y=tp_ug,color=lakename)) +
  geom_point() +
  geom_smooth(method='lm')+
  xlim(0,50)+
  labs(
    x = expression(paste('Phosphate (',mu,'/g')),
    y = expression(paste('Total Phosphorus (',mu,'/g)')),
    title = 'Q4: Plot of Total Phosphorus vs Phosphate',
    color='Lake'
    )
```

```
## `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()`).
```



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 Boxlots
base_plt <- NTL_tidy %>%
  mutate(the_month = factor(month,levels = 1:12, labels = month.abb)) %>%
  ggplot(aes(x=the_month,color=lakename)) #+
  scale_x_discrete(labels=month.abb)
```

```
## <ggproto object: Class ScaleDiscretePosition, ScaleDiscrete, Scale, gg>
##     aesthetics: x xmin xmax xend
##     axis_order: function
##     break_info: function
##     break_positions: function
##     breaks: waiver
##     call: call
##     clone: function
##     dimension: function
##     drop: TRUE
##     expand: waiver
##     get_breaks: function
```

```
##     get_breaks_minor: function
##     get_labels: function
##     get_limits: function
##     get_transformation: function
##     guide: waiver
##     is_discrete: function
##     is_empty: function
##     labels: Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
##     limits: NULL
##     make_sec_title: function
##     make_title: function
##     map: function
##     map_df: function
##     n.breaks.cache: NULL
##     na.translate: TRUE
##     na.value: NA
##     name: waiver
##     palette: function
##     palette.cache: NULL
##     position: bottom
##     range: environment
##     range_c: environment
##     rescale: function
##     reset: function
##     train: function
##     train_df: function
##     transform: function
##     transform_df: function
##     super:  <ggproto object: Class ScaleDiscretePosition, ScaleDiscrete, Scale, gg>
```

```
plt1 <- base_plt+
  geom_boxplot(aes(y=temperature_C),show.legend = T) +
  labs(
    x='',
    y='Temperature (C)',
    title='Temperature'
  ) +
  theme(
    legend.position = "top"
  )

plt2 <- base_plt+
  geom_boxplot(aes(y=tp_ug),show.legend = F) +
  labs(
    x='',
    y='Total P',
    title='Total P'
  )

plt3 <- base_plt+
  geom_boxplot(aes(y=tn_ug),show.legend = F) +
  labs(
    x='Month',
    y='Total N',
```

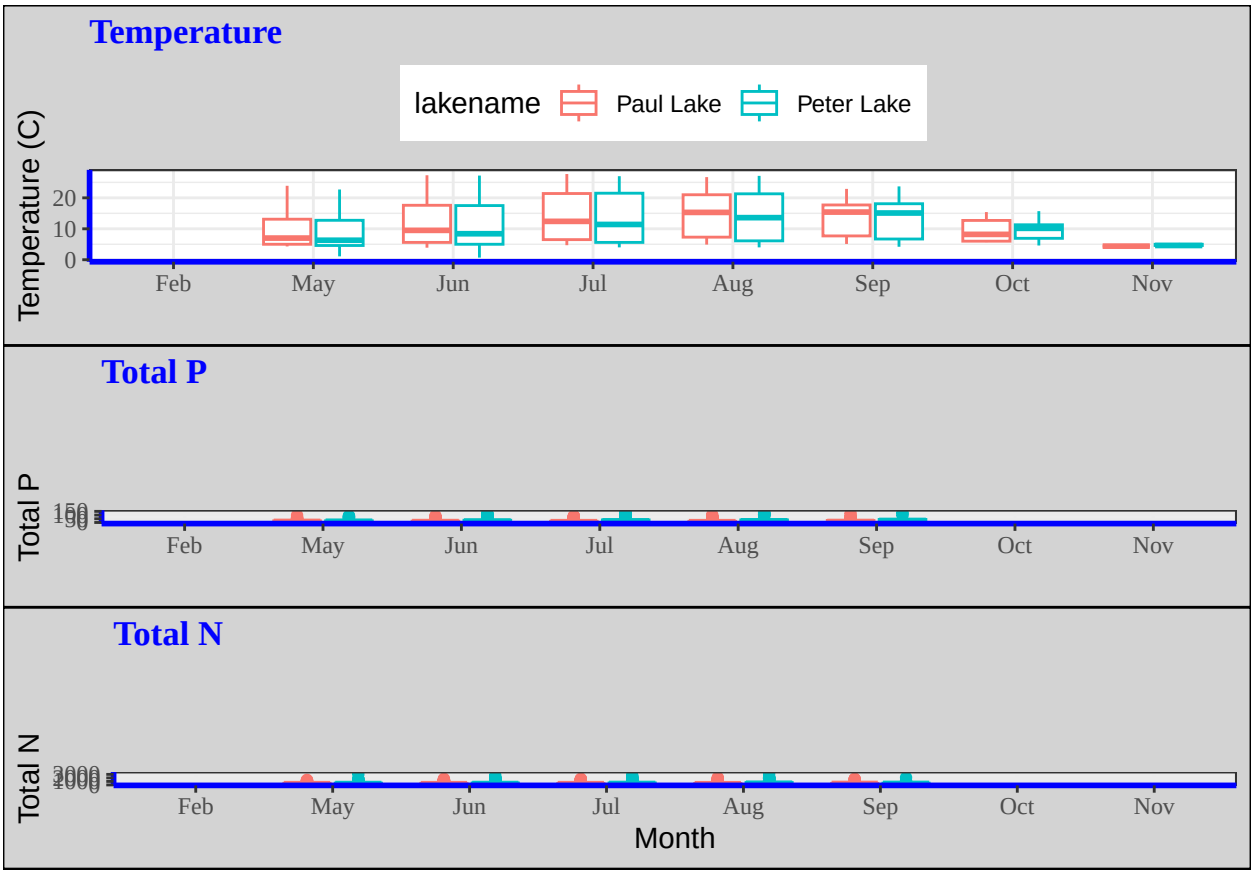
```
    title='Total N',
    color='Lake'
  )

plot_grid(plt1,plt2,plt3,
          nrow = 3,
          align = 'h',
          rel_heights = c(1.3,1,1))
```

```
## 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()`).
```



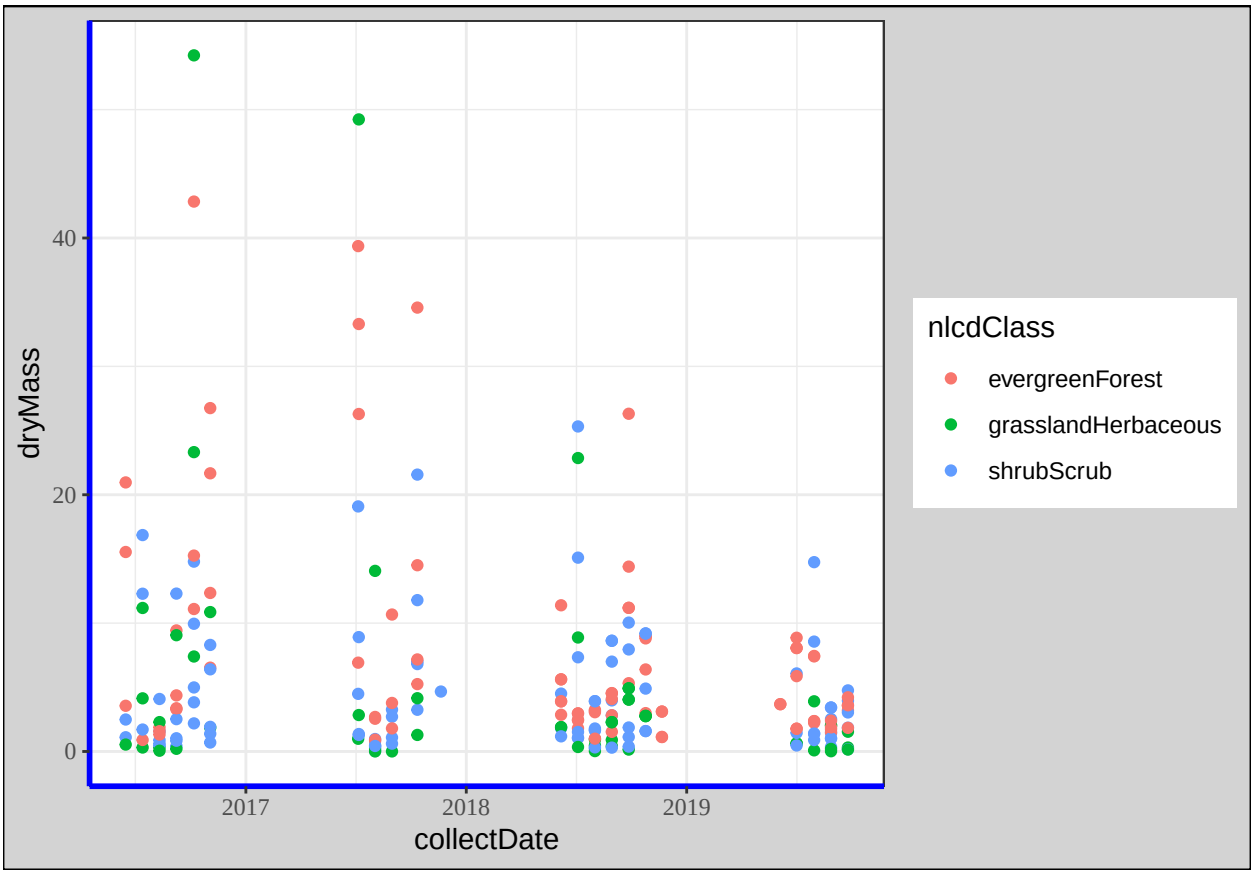
Question: What do you observe about the variables of interest over seasons and between lakes?

Answer:

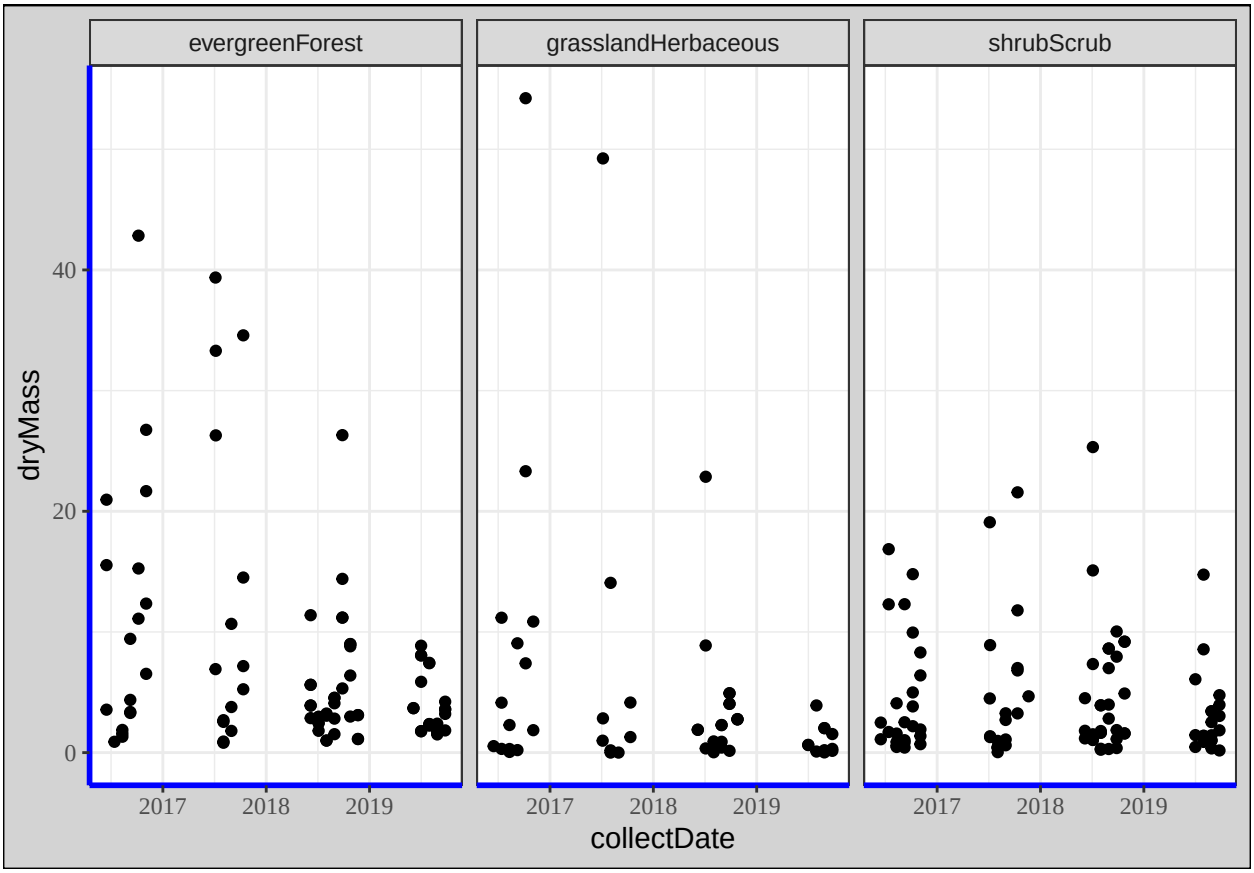
6. [Niwot Ridge] Plot a subset of the litter dataset by displaying only the "Needles" functional group. Plot the dry mass of needle litter by date and separate by NLCD class with a color aesthetic. (no need to adjust the name of each land use)

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

```
#6
Trap_data %>%
  filter(functionalGroup == 'Needles') %>%
  ggplot(aes(x=collectDate,y=dryMass,color=nlcdClass)) +
  geom_point()
```




```
#7
Trap_data %>%
  filter(functionalGroup == 'Needles') %>%
  ggplot(aes(x=collectDate,y=dryMass)) +
  geom_point() +
  facet_wrap(vars(nlcdClass))
```



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

Answer: