Assignment 5: Data Visualization

Gaby Czarniak

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 Load packages and verify home directory
# get working directory
getwd()
```

```
## [1] "/home/guest/gaby-cz_EDE_Fall2023"
```

```
# import basic libraries
library(tidyverse);library(lubridate);library(here)
```

```
## -- 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 1.9.2     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 errors
## here() starts at /home/guest/gaby-cz_EDE_Fall2023
```

```
#install.packages(colormap)
# Multiple plots on a page
library(cowplot)
```

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

```
library(ggridges)
#library(viridis)
#library(RColorBrewer)
#library(colormap)
library(ggthemes)
```

```
##
## Attaching package: 'ggthemes'
##
## The following object is masked from 'package:cowplot':
##
##     theme_map
```

```
PeterPaul_chem_nutrients <-
  read.csv("./Data/Processed_KEY/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv",
           stringsAsFactors = TRUE)
# PeterPaul_chem_nutrients

NIWO_litter_processed <-
  read.csv("./Data/Processed_KEY/NEON_NIWO_Litter_mass_trap_Processed.csv",
           stringsAsFactors = TRUE)
# NIWO_litter_processed

#2 Confirm dates are in date format
is.Date(PeterPaul_chem_nutrients$sampledate)
```

```
## [1] FALSE
```

```
# returns false - not yet formatted as date

is.Date(NIWO_litter_processed$collectDate)
```

```
## [1] FALSE
```

```
# returns false - not yet formatted as date

# changing to date format
PeterPaul_chem_nutrients$sampledate <-
  as.Date(PeterPaul_chem_nutrients$sampledate, format = "%Y-%m-%d")
NIWO_litter_processed$collectDate <-
  as.Date(NIWO_litter_processed$collectDate, format = "%Y-%m-%d")

# confirming change was successful
is.Date(PeterPaul_chem_nutrients$sampledate) #true
```

```
## [1] TRUE
```

```
is.Date(NIWO_litter_processed$collectDate) #true
```

```
## [1] TRUE
```

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
Gabystheme <- theme_wsj(base_size = 10) +
  theme(#reformatting axes
        axis.text = element_text(color = "black"),
        axis.title.x = element_text(size = rel(.6), angle = 0),
        axis.title.y = element_text(size = rel(.6), angle = 90),
        axis.line = element_line(linewidth = 3, colour = "grey70"),
        # setting base format for legend
        legend.position = "bottom",
        legend.justification = "left",
        legend.title.align = 1,
        # changing panel color
        panel.grid.major = element_line(color = "grey80"),
        # base setting for title
        plot.title = element_text(size = rel(1))
        )
# Gabystheme

theme_set(Gabystheme)
```

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.

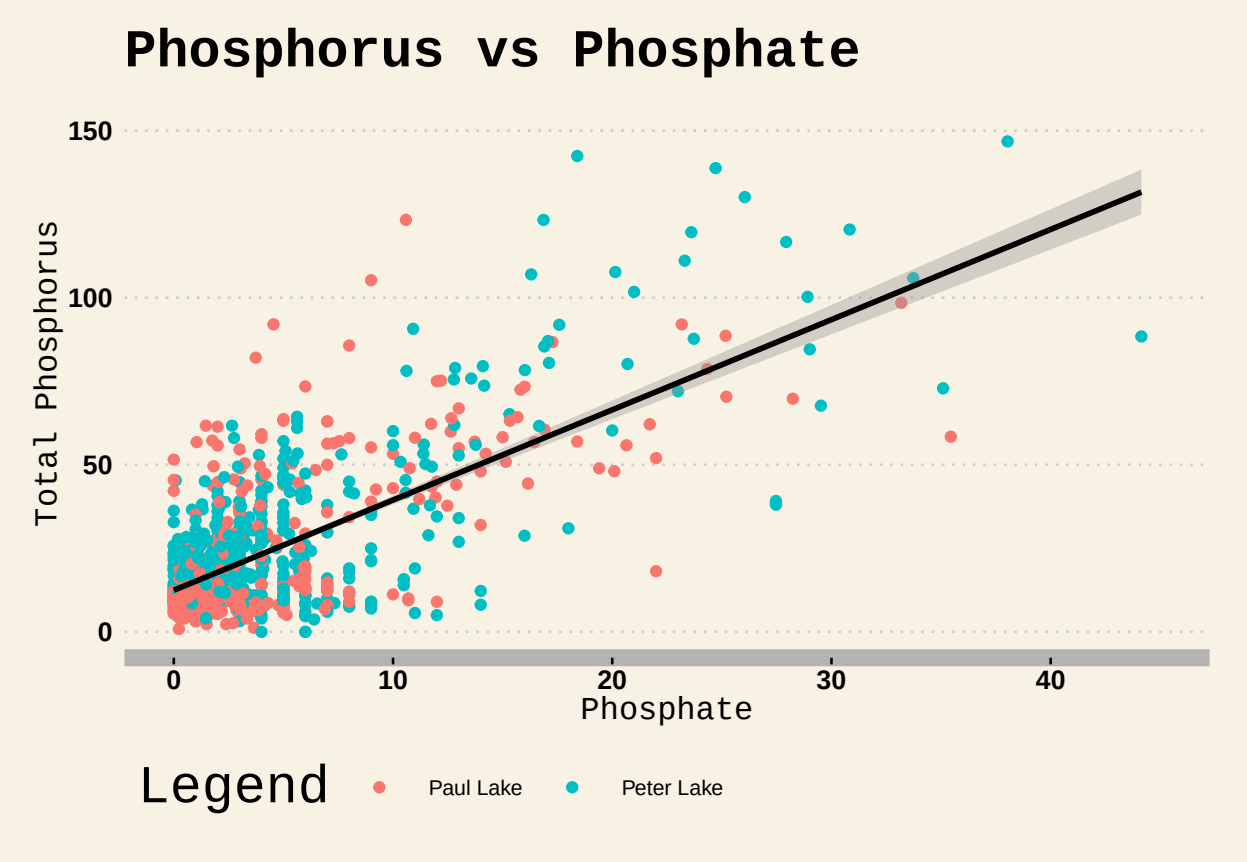
4. [NTL-LTER] Plot total phosphorus (`tp_ug`) by phosphate (`po4`), with separate aesthetics for Peter and Paul lakes. Add a line of best fit and color it black. Adjust your axes to hide extreme values (hint: change the limits using `xlim()` and/or `ylim()`).

```
#4
# creating object within which to store new plot
phosphorus_vs_phosphate <-
  # defining axes and asking r to choose color
  # based on whether lake is Peter or Paul
  ggplot(PeterPaul_chem_nutrients, aes(x=po4, y=tp_ug, color = lakename)) +
    # choosing scatterplot because: 2 variables, numeric, continuous
    geom_point() +
    # adjusting axes to hide extreme values
    xlim(0, 45) +
    ylim(0,155) +
    # finding a line of best fit
    geom_smooth(method = lm, color = "black") +
    ggtitle("Phosphorus vs Phosphate") +
    labs(x="Phosphate", y="Total Phosphorus", color="Legend")
print(phosphorus_vs_phosphate)
```

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

```
## Warning: Removed 21948 rows containing non-finite values (`stat_smooth()`).
```

```
## Warning: Removed 21948 rows containing missing values (`geom_point()`).
```

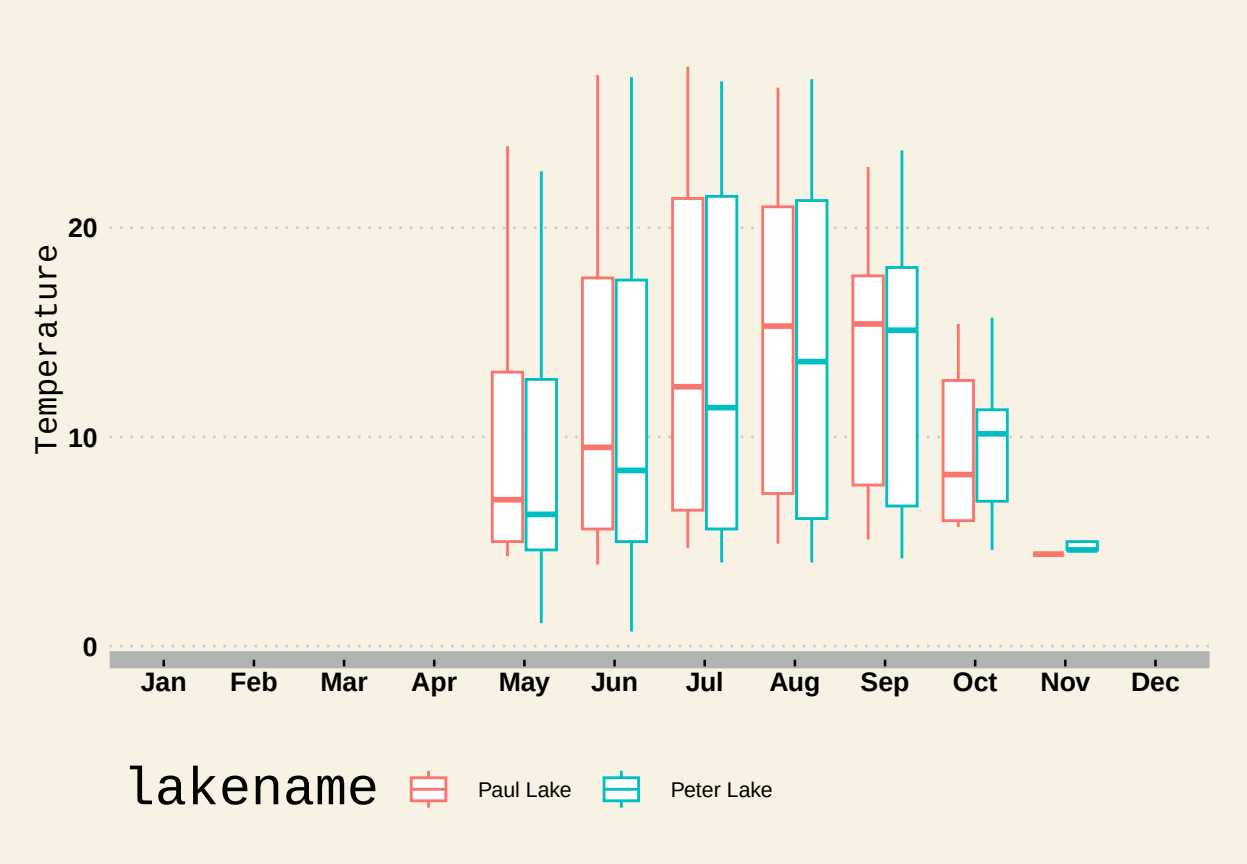


5. [NTL-LTER] Make three separate boxplots of (a) temperature, (b) TP, and (c) TN, with month as the x axis and lake as a color aesthetic. Then, create a cowplot that combines the three graphs. Make sure that only one legend is present and that graph axes are aligned.

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

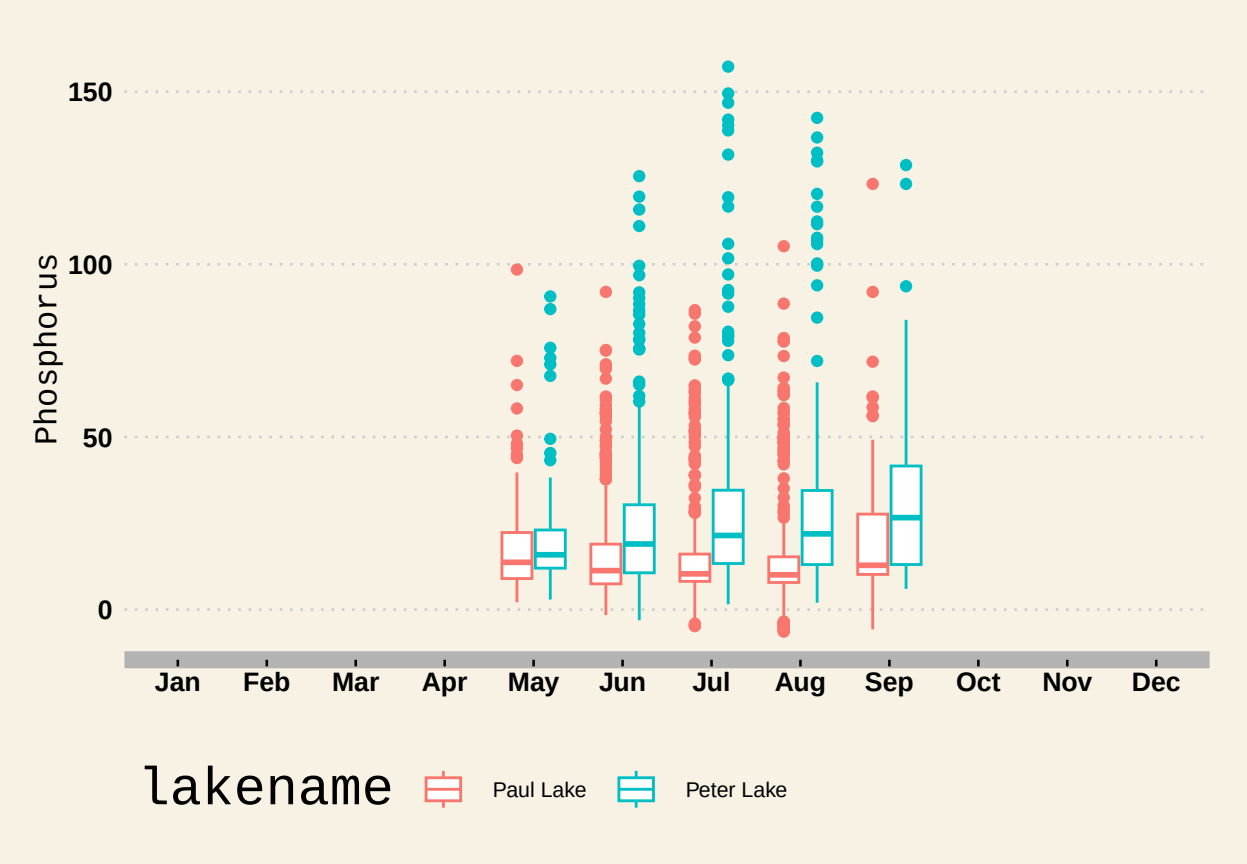
```
#5

# Plotting temperature in Celsius by month
temperature_plot <-
  ggplot(PeterPaul_chem_nutrients,
         aes(x=factor(month, levels = 1:12, labels = month.abb),
             y=temperature_C)) +
  # setting color aesthetic based on lake
  geom_boxplot(aes(color = lakename)) +
  scale_x_discrete(name = "", drop=FALSE) +
  # the scales allow us to override defaults in the different axes
  # setting axis title labels
  labs(x="",y="Temperature")
print(temperature_plot)
```

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



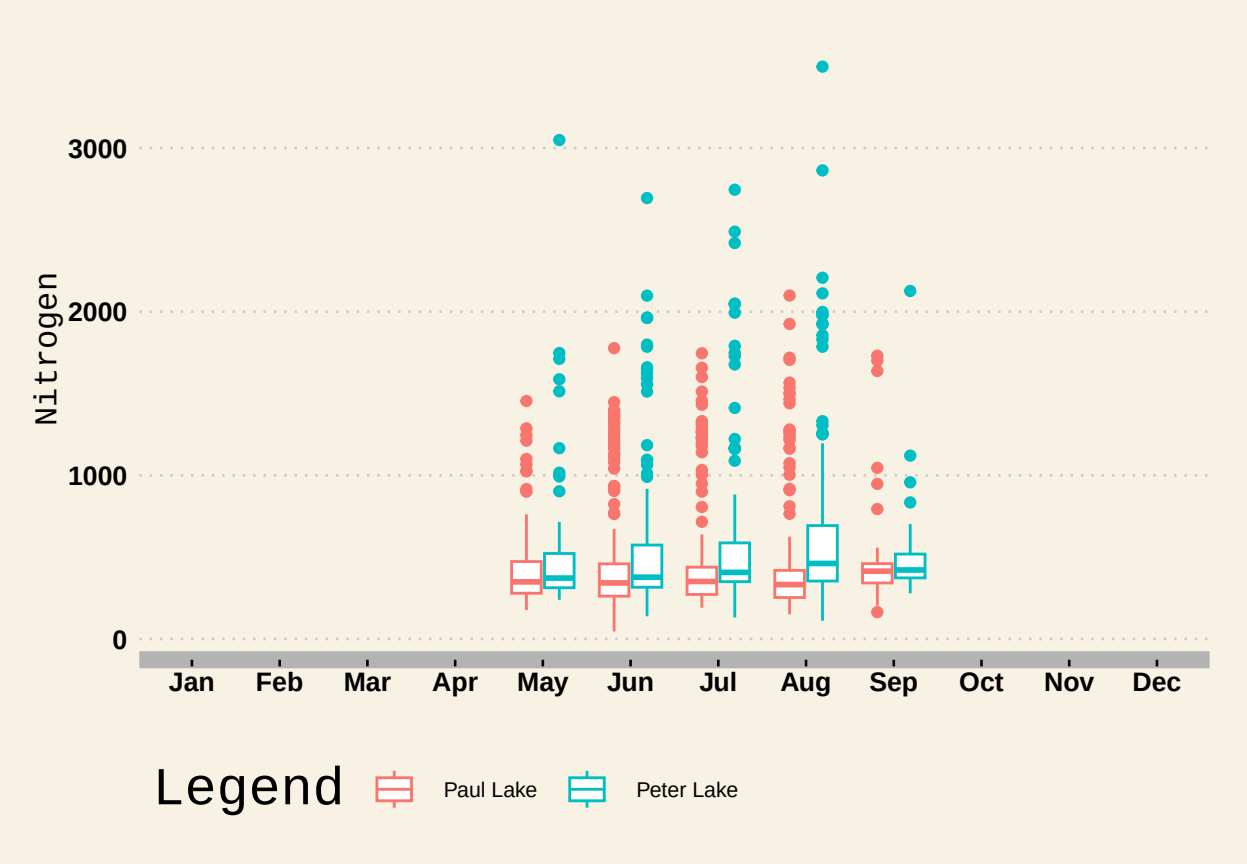

```
# Plotting total phosphorus by month
TP_plot <-
    ggplot(PeterPaul_chem_nutrients,
           aes(x=factor(month, levels = 1:12, labels = month.abb),
               y=tp_ug)) +
    scale_x_discrete(drop=FALSE) +
    geom_boxplot(aes(color = lakename)) +
    labs(x="",y="Phosphorus")
print(TP_plot)
```

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




```
# Plotting total nitrogen by month
TN_plot <-
    ggplot(PeterPaul_chem_nutrients,
           aes(x=factor(month, levels = 1:12, labels = month.abb),
               y=tn_ug)) +
    scale_x_discrete(drop=FALSE) +
    geom_boxplot(aes(color = lakename)) +
    labs(x="",y="Nitrogen", color="Legend")
print(TN_plot)
```

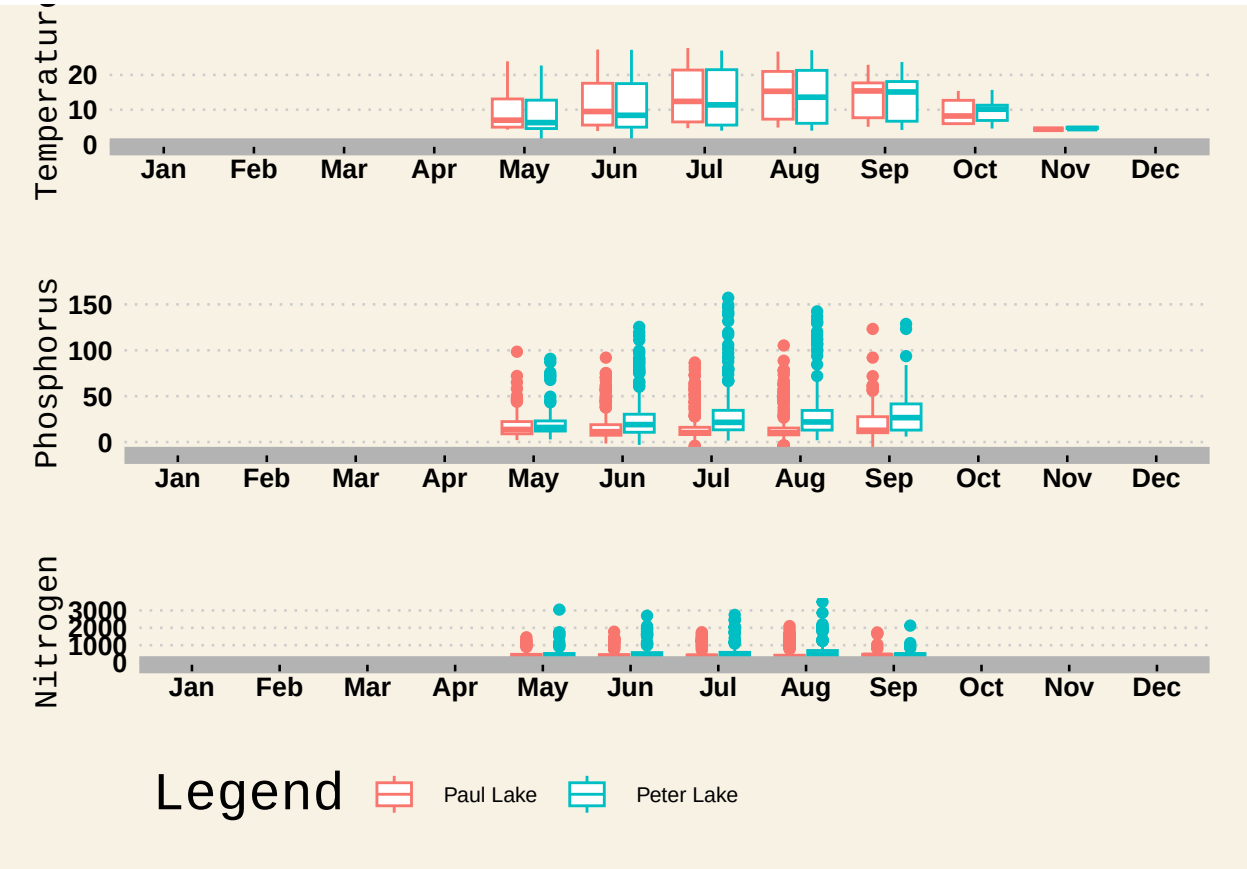
```
## Warning: Removed 21583 rows containing non-finite values (‘stat_boxplot()‘).
```




```
# Cowplot combining the three plots into one view
cowplot <- plot_grid(
  #arranging three plots in single column with shared legend
  temperature_plot + theme(legend.position="none"),
  TP_plot + theme(legend.position="none"),
  # only using the legend for one plot, dropping the rest
  TN_plot,
  nrow = 3,
  align = 'h',
  rel_heights = c(4,5,5))
```

```
## 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()‘).

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

```
print(cowplot)
```



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

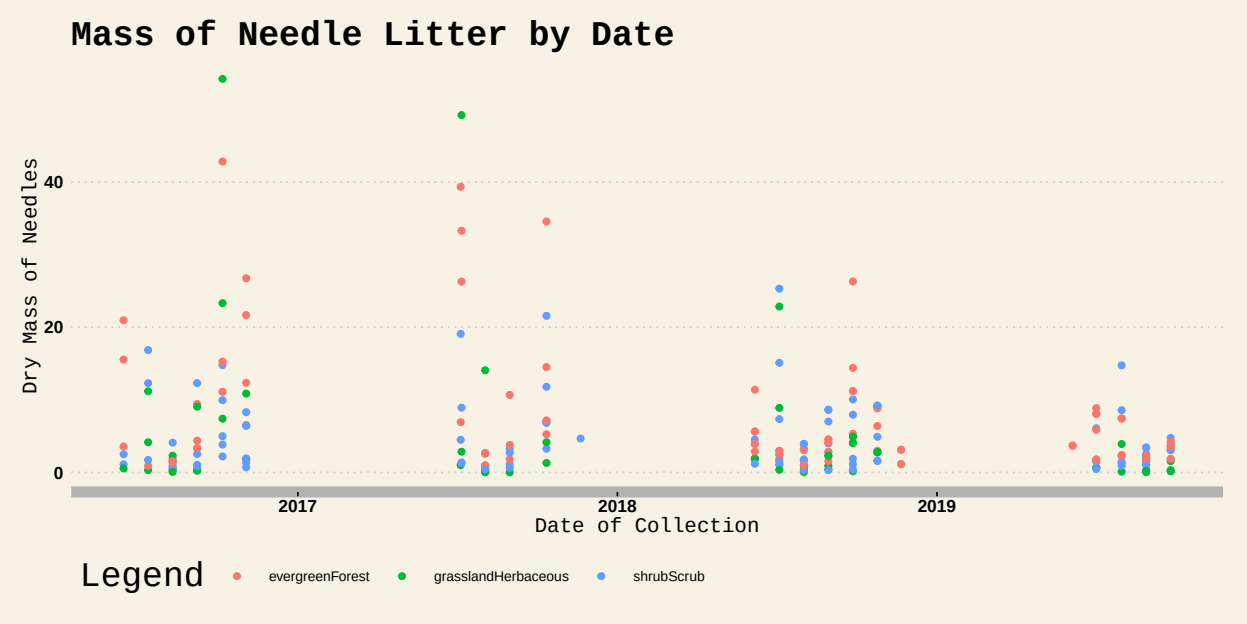
> Answer: The subsets of the data have very different distributions along the variables of interest. Peter Lake has higher total nitrogen means across most months (with the exception of September, which is closer to Paul Lake's), and its values are more distributed. The same goes for total phosphorus–Peter Lake has higher means across months and is slightly to significantly more distributed in total phosphorus values within each month measured, with the exception of May, where Paul Lake had a higher distribution, and September, where they were quite similar. From a temperature perspective, the two lakes are a bit more similar. Peter Lake has a slightly lower mean temperature than Paul Lake for most months measured, but it has a notably higher mean temperature in October 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)

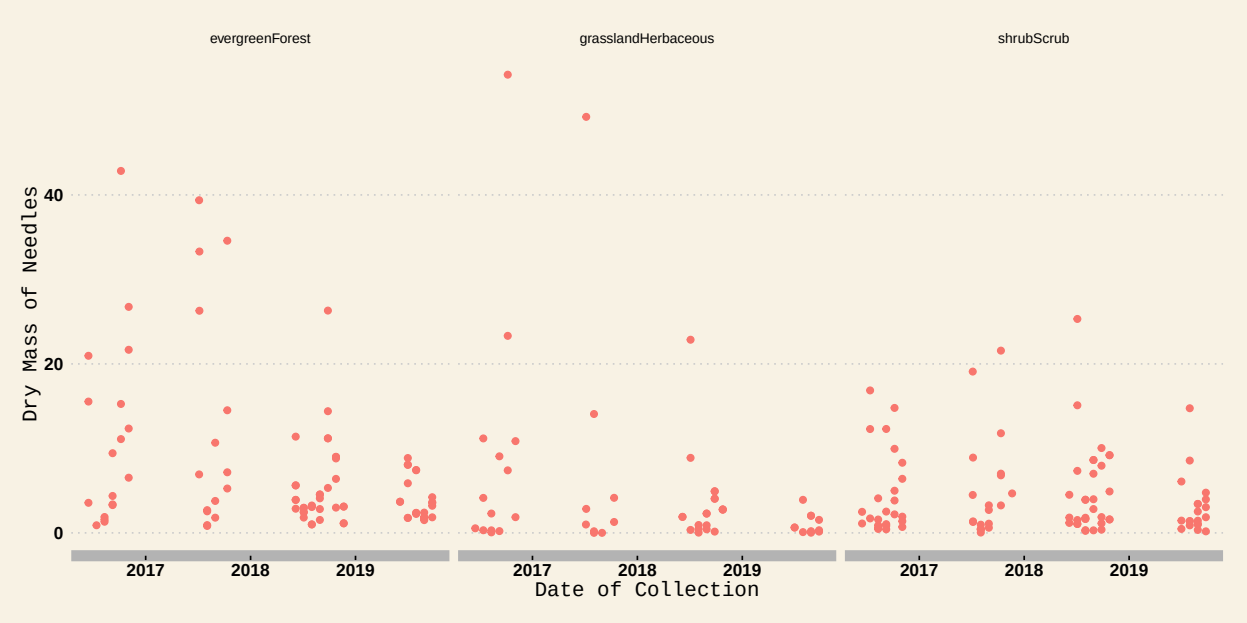
7. [Niwot Ridge] Now, plot the same plot but with NLCD classes separated into three facets rather than separated by color.

```
#6
# using only the needles subset of data's functionalGroup column
needle_mass <-
  ggplot(subset(NIWO_litter_processed, functionalGroup == "Needles"),
         aes(x = collectDate, y = dryMass)) +
  # delineating by NLCD class using color aesthetic
  geom_point(aes(color = nlcdClass)) +
```

```
  # setting axis title labels and legend title label
  labs(x="Date of Collection", y="Dry Mass of Needles", color="Legend") +
  ggtitle("Mass of Needle Litter by Date")
print(needle_mass)
```




```
#7
# Filter dataset within plot building and facet by multiple variables
needle_mass_faceted <-
  ggplot(subset(NIWO_litter_processed, functionalGroup == "Needles"),
         aes(x = collectDate, y = dryMass)) +
  theme(legend.position="none") +
  # moving back to single color across all points, per instructions
  geom_point(aes(color = "black")) +
  # facetting evergreenForest, grasslandHerbaceous, shrubScrub
  facet_wrap(vars(nlcdClass)) +
  # setting axis title labels; legend would be redundant here
  labs(x="Date of Collection", y="Dry Mass of Needles")
print(needle_mass_faceted)
```



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

Answer: If we're looking at needle mass by date because we're interested in trends over time, the plot from #7 makes it easier to see within evergreen forest, herbaceous grasslands, and shrub area, how needle mass is changing within each of those areas. The plot in #6 gives an idea that needle mass at large is changing from one year to the next, but I think plot #7 is better for these purposes because it allows you to compare landcover types more easily and clearly.