Assignment 5: Data Visualization

Meilin Chan

OVERVIEW

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

Directions

1. Change "Student Name" on line 3 (above) with your name.
2. Work through the steps, **creating code and output** that fulfill each instruction.
3. Be sure to **answer the questions** in this assignment document.
4. When you have completed the assignment, **Knit** the text and code into a single PDF file.
5. After Knitting, submit the completed exercise (PDF file) to the dropbox in Sakai. Add your last name into the file name (e.g., "Fay_A05_DataVisualization.Rmd") prior to submission.

The completed exercise is due on Monday, February 14 at 7:00 pm.

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
getwd()
```

```
## [1] "C:/Users/meili/OneDrive - Duke University/EDA/Environmental_Data_Analytics_2022/Assignments"
```

```
library(tidyverse)
```

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

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.1     v forcats 0.5.1
```

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

```
library(cowplot)

PP_Nutrients <- read.csv("../Data/Processed/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv",

NN_Litter <- read.csv("../Data/Processed/NEON_NIWO_Litter_mass_trap_Processed.csv", stringsAsFactors =
```

```
#2
class(PP_Nutrients$sampledate)
```

```
## [1] "character"
```

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

```
## [1] "Date"
```

```
class(NN_Litter$collectDate)
```

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

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

```
## [1] "Date"
```

Define your theme

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

```
#3
A05_theme <- theme_bw(base_size = 12) +
  theme(axis.text = element_text(color = "black"), legend.position = "bottom")
theme_set(A05_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.

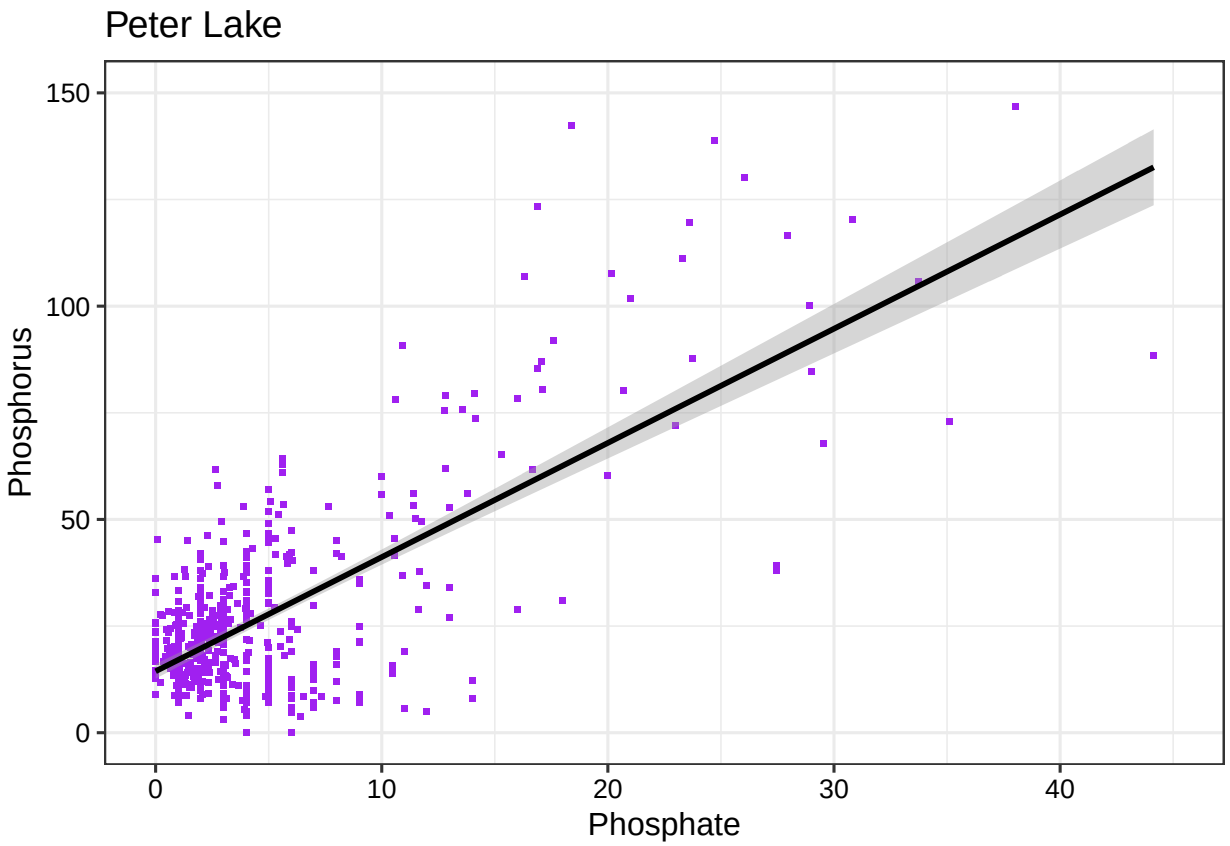
4. [NTL-LTER] Plot total phosphorus (`tp_ug`) by phosphate (`po4`), with separate aesthetics for Peter and Paul lakes. Add a line of best fit and color it black. Adjust your axes to hide extreme values (hint: change the limits using `xlim()` and `ylim()`).

```
#4
library(ggplot2)

NTL.LTER.phos.Peter <- ggplot(subset(PP_Nutrients, lakename == "Peter Lake"),
                              aes(x = po4, y = tp_ug)) +
  geom_point(color = "purple", size = 0.8, shape = "square")+
  xlim(0, 45) +
  ylim(0, 150) +
  geom_smooth(method = lm, color = "black") +
  xlab("Phosphate") +
  ylab("Phosphorus") +
  ggtitle("Peter Lake")
print(NTL.LTER.phos.Peter)
```

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

```
## Warning: Removed 11423 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 11423 rows containing missing values (geom_point).
```

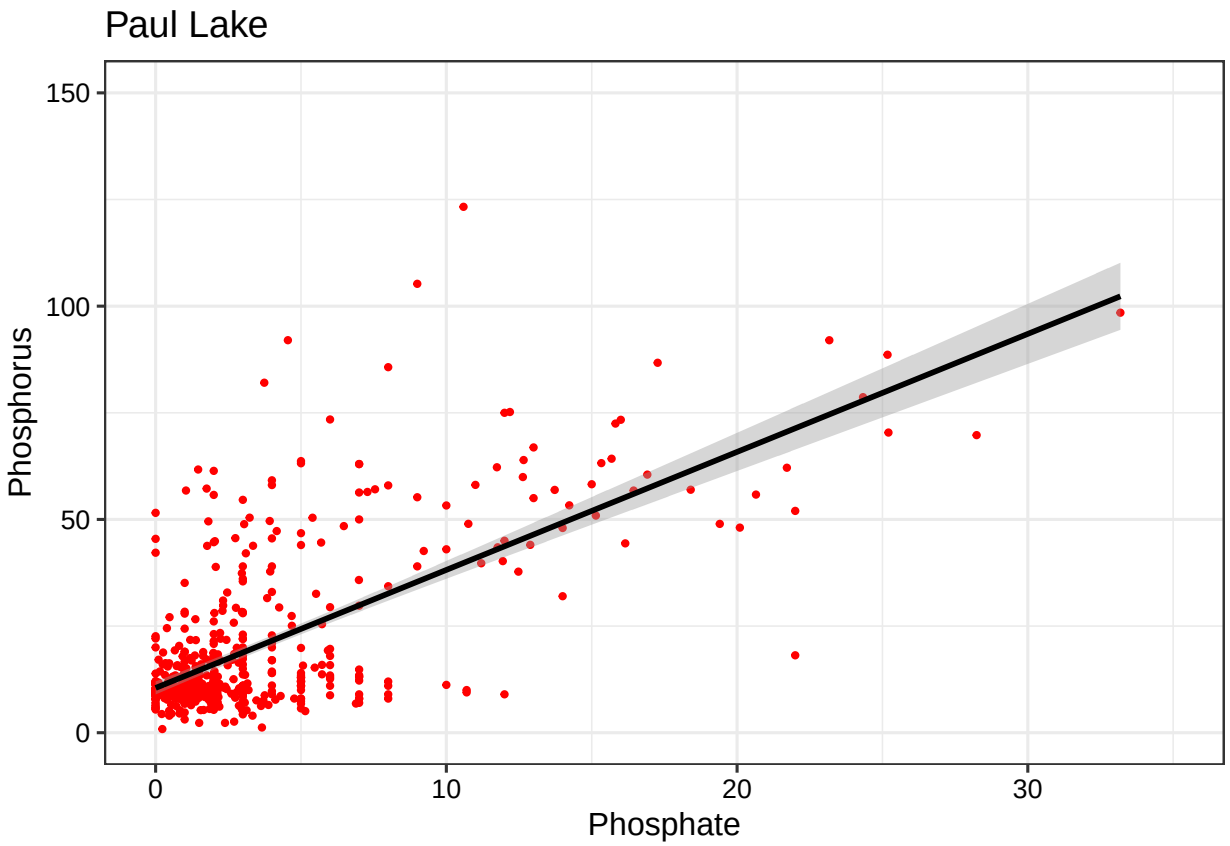



```
NTL.LTER.phos.Paul <- ggplot(subset(PP_Nutrients, lakename == "Paul Lake"),
                             aes(x = po4, y = tp_ug)) +
  geom_point(color = "red", size = 0.8) +
  xlim(0, 35) +
  ylim(0, 150) +
  geom_smooth(method = lm, color = "black") +
  xlab("Phosphate") +
  ylab("Phosphorus") +
  ggtitle("Paul Lake")
print(NTL.LTER.phos.Paul)
```

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

```
## Warning: Removed 10526 rows containing non-finite values (stat_smooth).
```

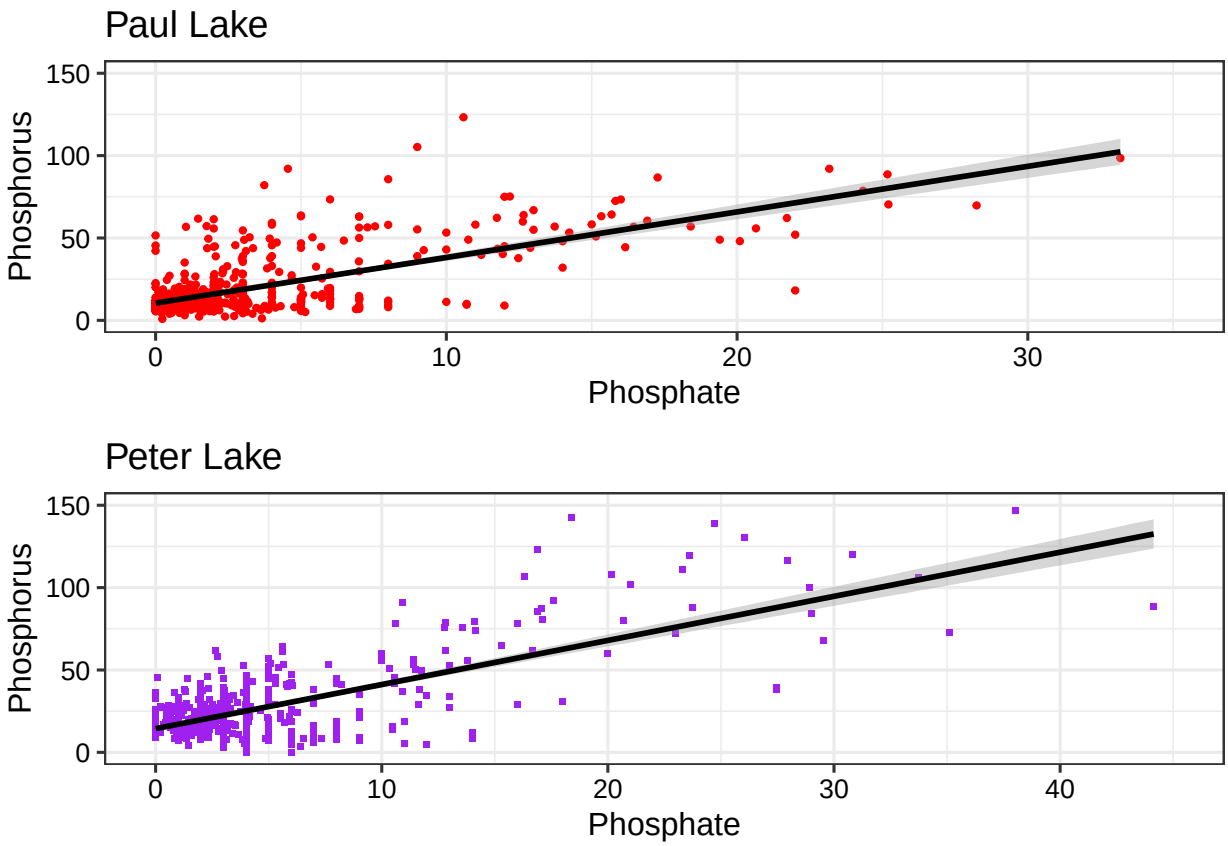
```
## Warning: Removed 10526 rows containing missing values (geom_point).
```




```
NTL.LTER.phos <- plot_grid(NTL.LTER.phos.Paul, NTL.LTER.phos.Peter,
                           nrow = 2, align = 'hv', rel_heights = c(3,3))
```

```
## `geom_smooth()` using formula 'y ~ x'
## Warning: Removed 10526 rows containing non-finite values (stat_smooth).
## Warning: Removed 10526 rows containing missing values (geom_point).
## `geom_smooth()` using formula 'y ~ x'
## Warning: Removed 11423 rows containing non-finite values (stat_smooth).
## Warning: Removed 11423 rows containing missing values (geom_point).
```

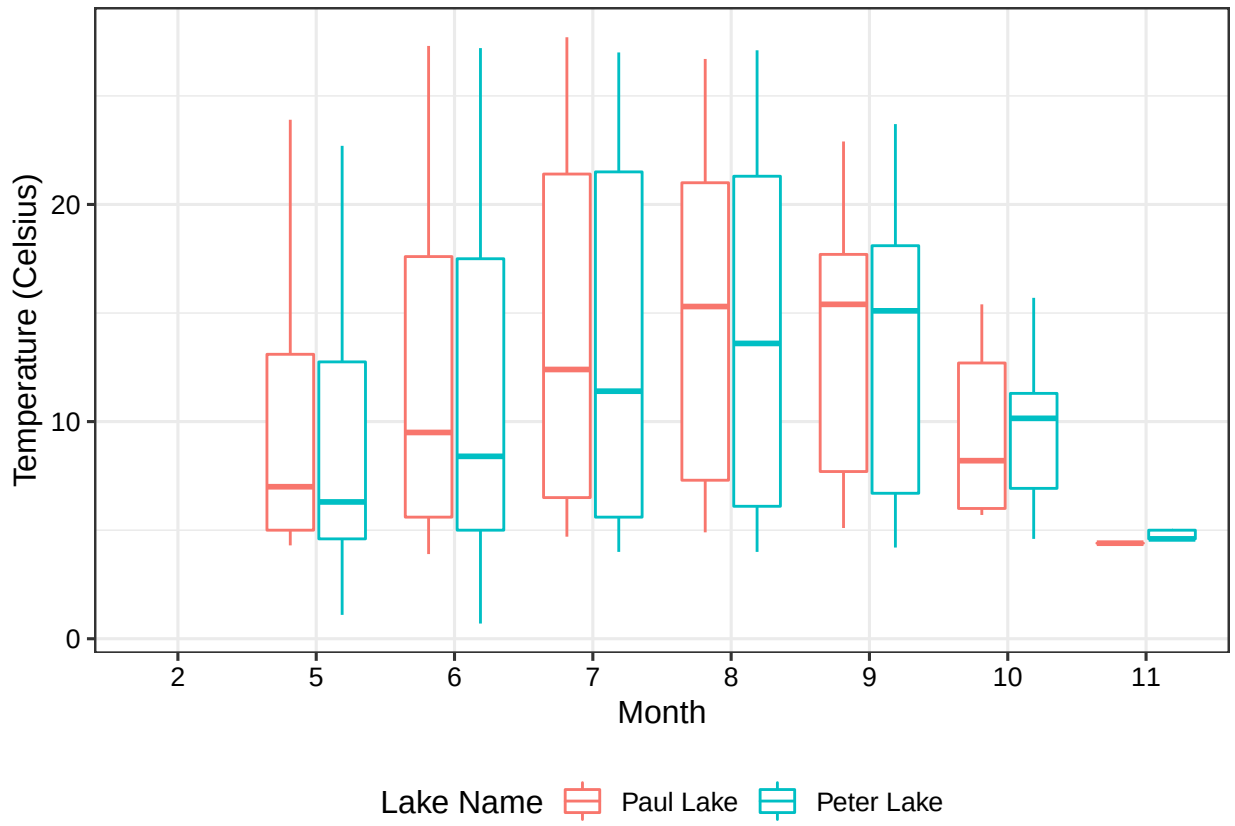
```
print(NTL.LTER.phos)
```



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
PP_Nutrients$month <- as.factor(PP_Nutrients$month)

NTL.temp <- ggplot(PP_Nutrients, aes(x = month, y = temperature_C)) +
  geom_boxplot(aes(color = lakename)) +
  xlab("Month") +
  ylab("Temperature (Celsius)") +
  labs(color = "Lake Name")

print(NTL.temp)
```

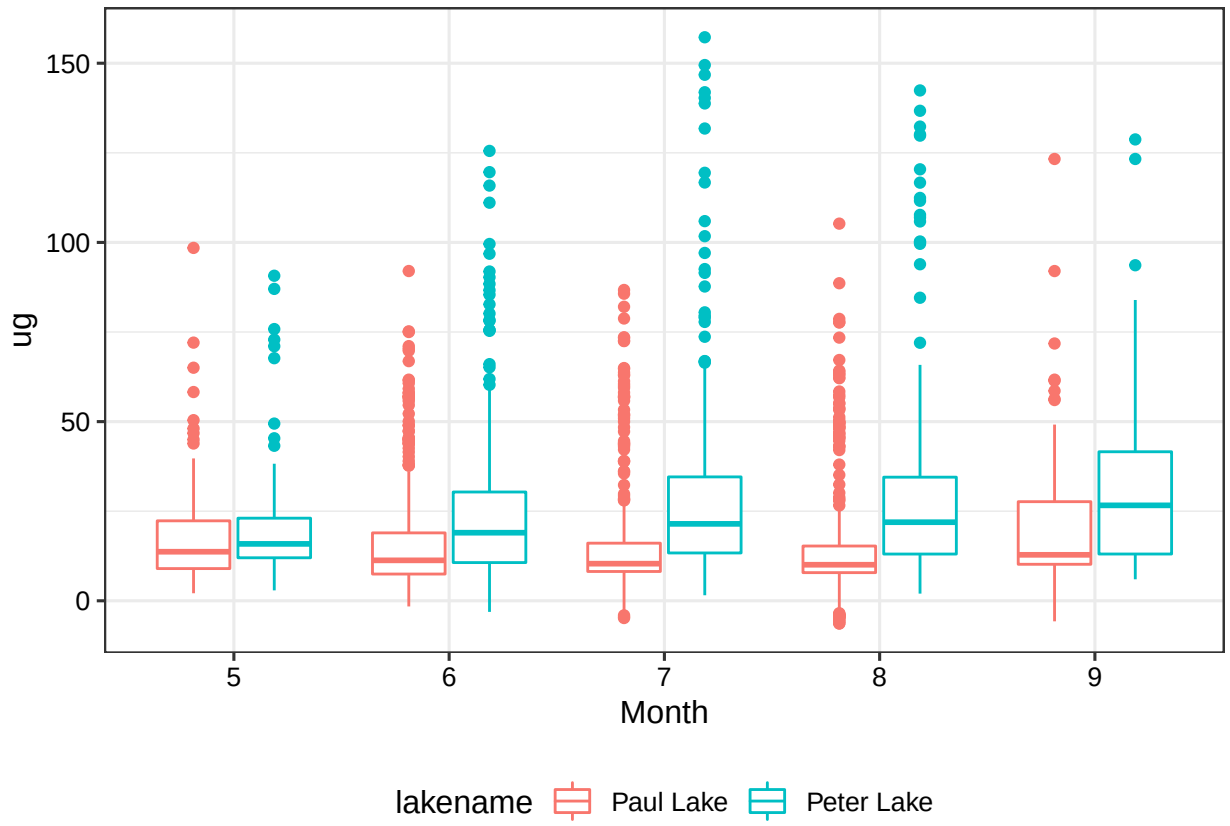
```
## Warning: Removed 3566 rows containing non-finite values (stat_boxplot).
```




```
NTL.TP <- ggplot(PP_Nutrients, aes(x = month, y = tp_ug)) +
  geom_boxplot(aes(color = lakename)) +
  xlab("Month") +
  ylab("ug") +
  xlim("5","6","7","8","9")

print(NTL.TP)
```

```
## Warning: Removed 205 rows containing missing values (stat_boxplot).
```

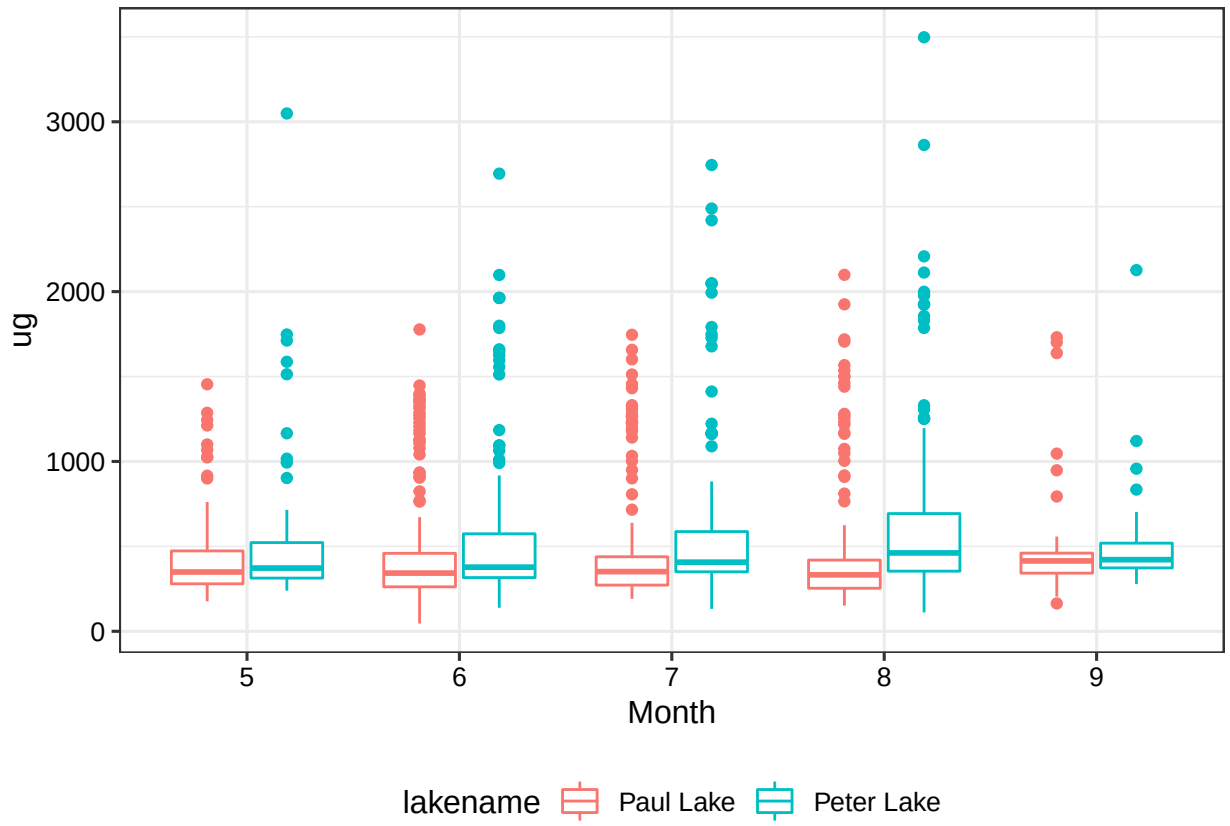
```
## Warning: Removed 20524 rows containing non-finite values (stat_boxplot).
```




```
NTL.TN <- ggplot(PP_Nutrients, aes(x = month, y = tn_ug)) +
  geom_boxplot(aes(color = lakename)) +
  xlab("Month") +
  ylab("ug") +
  xlim("5","6","7","8","9")

print(NTL.TN)
```

```
## Warning: Removed 205 rows containing missing values (stat_boxplot).
```

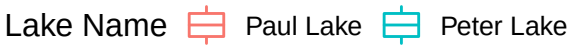
```
## Warning: Removed 21378 rows containing non-finite values (stat_boxplot).
```




```
NTL.legend <- get_legend(NTL.temp)
```

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

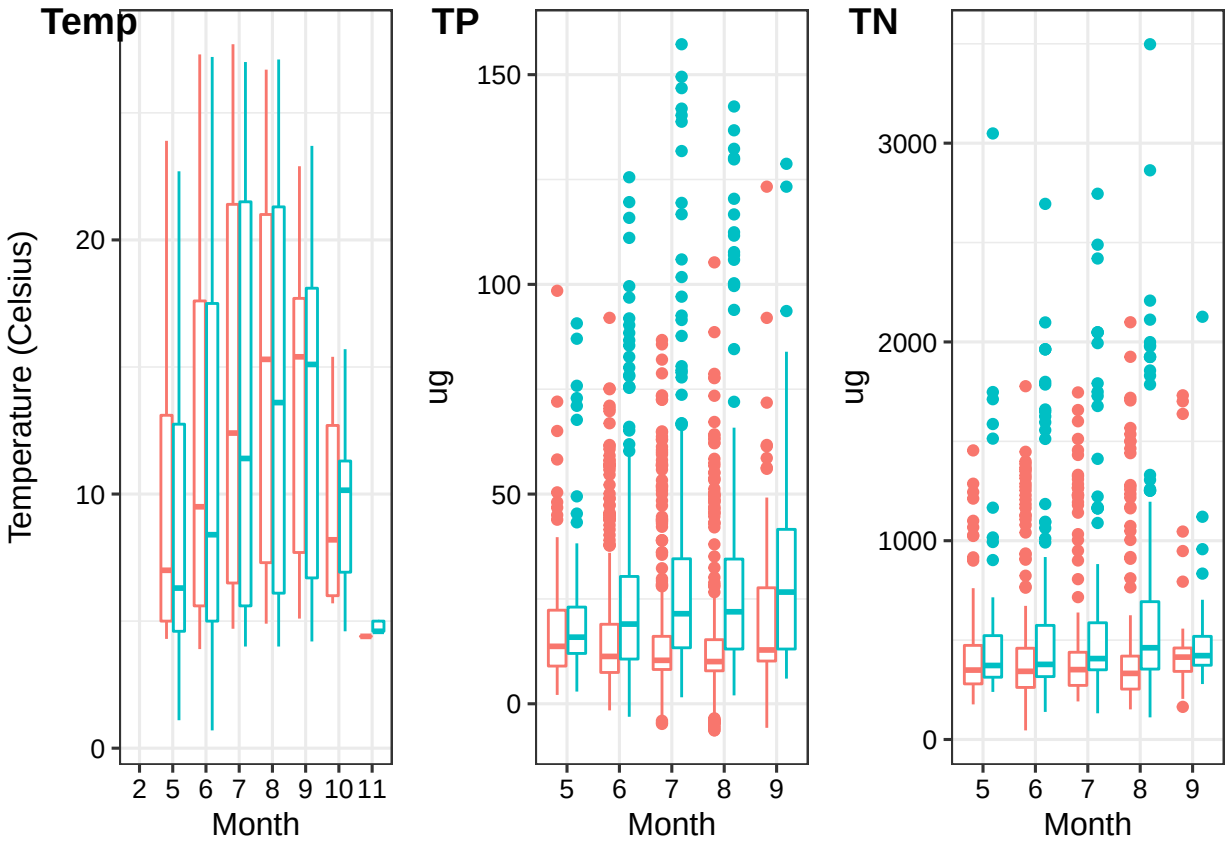
```
print(plot_grid(NTL.legend)) #only shows legend - no axes
```



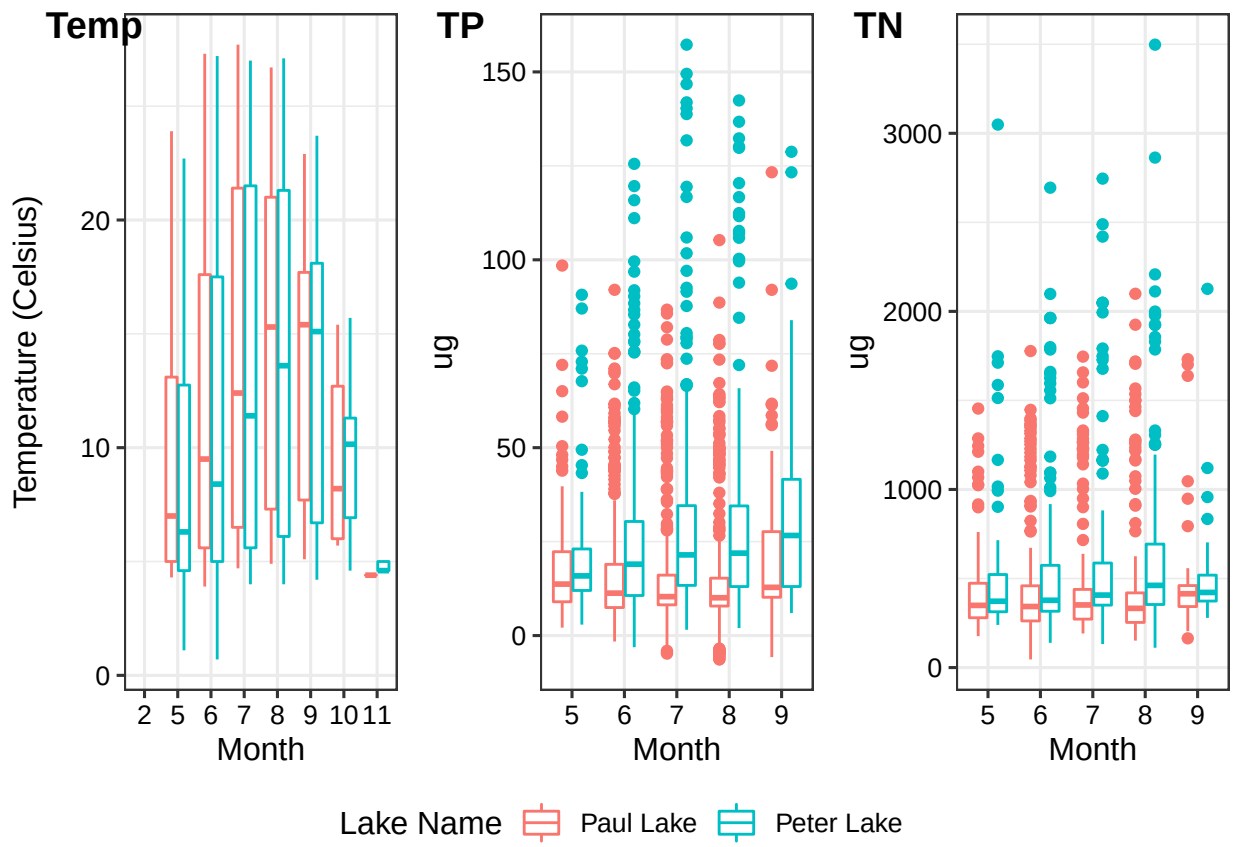

```
NTL.three <- plot_grid(NTL.temp + theme(legend.position = "none"),
                       NTL.TP + theme(legend.position = "none"),
                       NTL.TN + theme(legend.position = "none"),
                       nrow = 1,
                       align = 'hv',
                       labels = c("Temp", "TP", "TN"))
```

```
## Warning: Removed 3566 rows containing non-finite values (stat_boxplot).
## Warning: Removed 205 rows containing missing values (stat_boxplot).
## Warning: Removed 20524 rows containing non-finite values (stat_boxplot).
## Warning: Removed 205 rows containing missing values (stat_boxplot).
## Warning: Removed 21378 rows containing non-finite values (stat_boxplot).
```

```
print(NTL.three)
```




```
NTL.three.leg <- plot_grid(NTL.three, NTL.legend,nrow = 2,
                           rel_heights = c(3,0.3))

print(NTL.three.leg)
```



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

> Answer: The temperature of both Peter Lake and Paul Lake increase as the season shifts from winter to summer and then decreases again as the season shifts back into autumn. TP in Peter Lake increases in ug as the season shifts into summer and then autumn. TP in Paul Lake decreases as the season shifts into summer but then increases again as it becomes autumn. TN in Peter Lake increases and peaks as winter shifts into summer (around August), but then decreases once the season starts to roll into autumn. TN in Paul Lake seems to remain at a steady ug level until around September when it increases.

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)

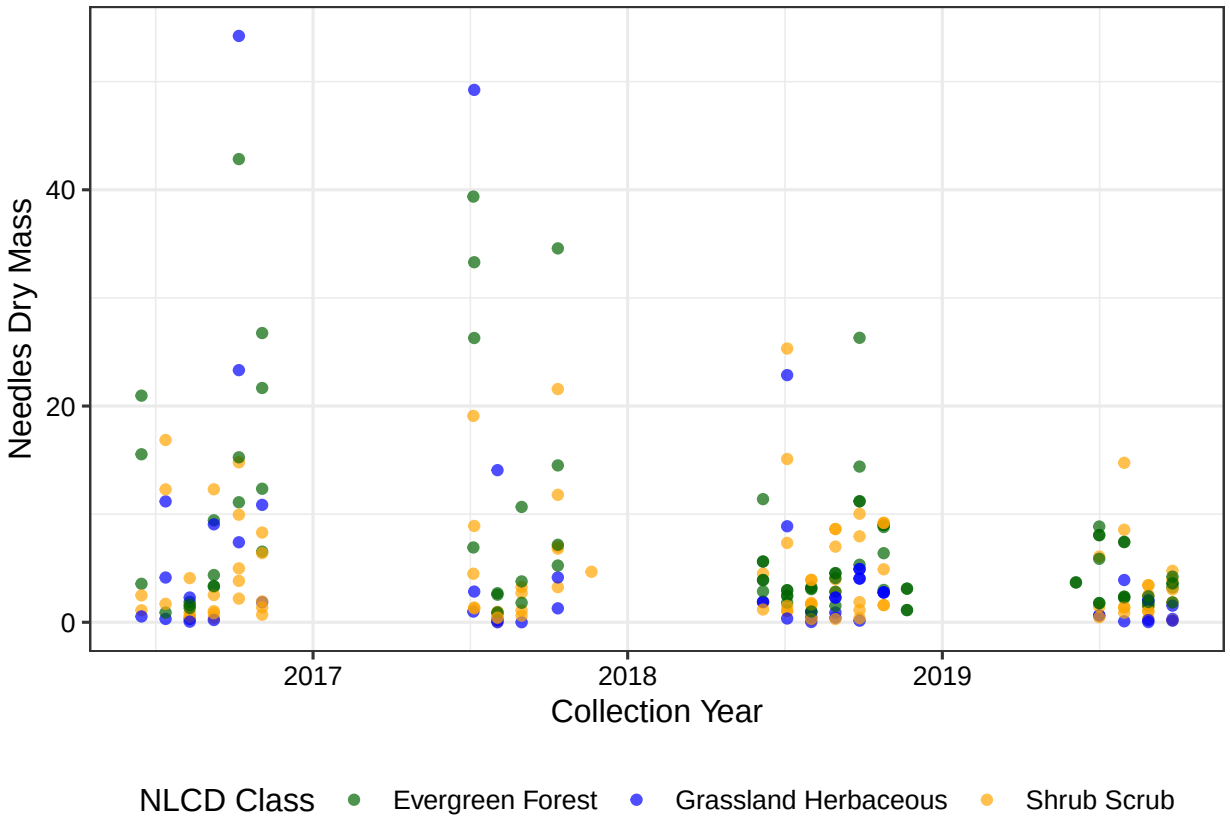
7. [Niwot Ridge] Now, plot the same plot but with NLCD classes separated into three facets rather than separated by color.

```
#6
class(NN_Litter$collectDate)
```

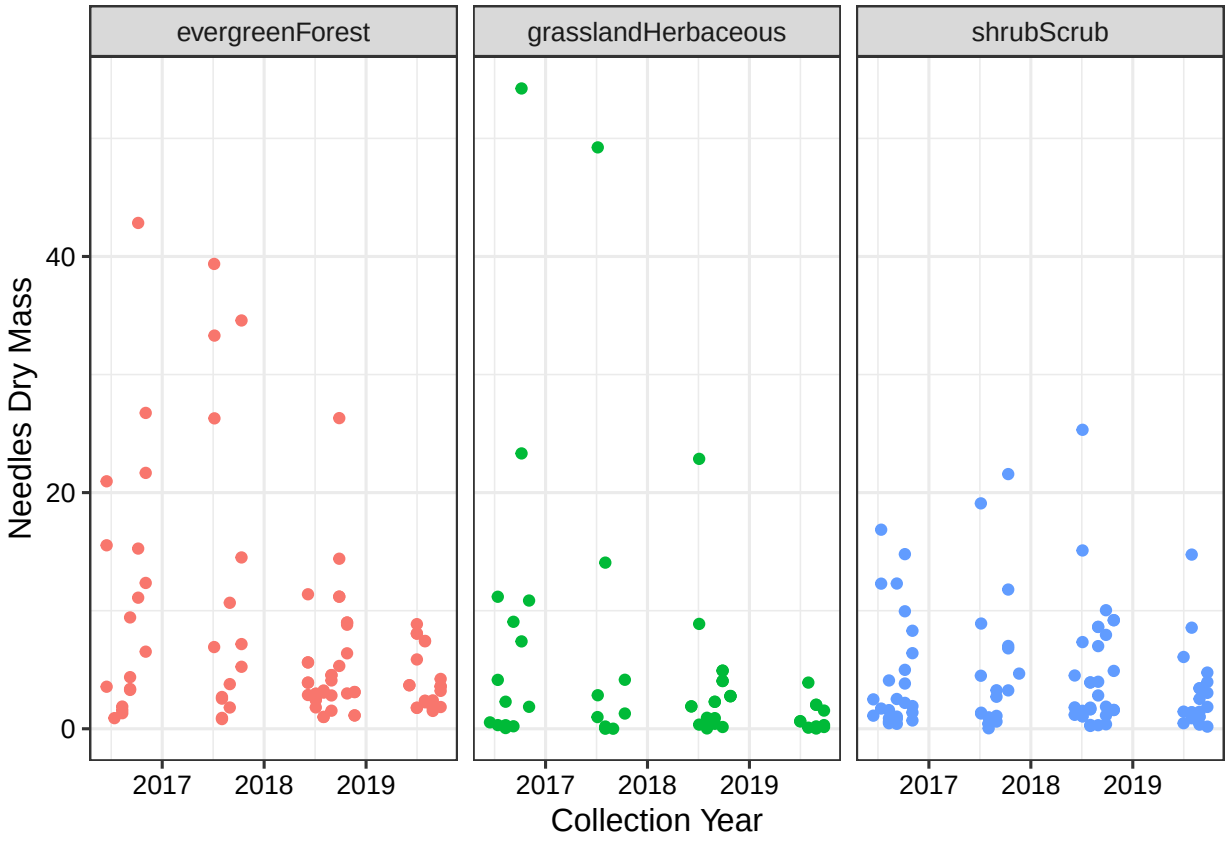
```
## [1] "Date"
```

```
NN_Litter$collectDate <- as.Date(NN_Litter$collectDate)


needles.dry.point <- ggplot(subset(NN_Litter, functionalGroup == "Needles")) +
  geom_point(aes(x = collectDate, y = dryMass,
                 color = nlcdClass), alpha = 0.7) +
  theme(legend.position = "bottom") +
```

```
  xlab("Collection Year") +
  ylab("Needles Dry Mass") +
  labs(color = "NLCD Class") +
  scale_color_manual(labels = c("Evergreen Forest", "Grassland Herbaceous",
                                "Shrub Scrub"),
                     values = c("dark green", "blue", "orange"))

print(needles.dry.point)
```




```
#7

pointfacet.needles.dry <- ggplot(subset(NN_Litter, functionalGroup == "Needles")) +
  geom_point(aes(x = collectDate, y = dryMass, color = nlcdClass)) +
  theme(legend.position = "none") +
  xlab("Collection Year") +
  ylab("Needles Dry Mass") +
  facet_grid(~ nlcdClass)

print(pointfacet.needles.dry)
```



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

Answer: Plot 7 is more effective. By separating NLCD classes into 3 facets, we are able to better see the overall spread of amount of needle dry mass collected over the timespan of the data. From Plot 7, we can see that across all NLCD classes, that most of the dry mass collected over the course of the time period was below 10 units of dry mass. When we observe Plot 6, the plot is difficult to read as some of the points overlap with each other and it is difficult to see the full distribution of each NLCD Class during each collection periods.