Assignment 5: Data Visualization

Catherine Otero

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/Catherine/OneDrive - Duke University/GitHub_Spot/Environmental_Data_Analytics_2021/Envi
```

```
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)
library(rlang)
```

```
##
## Attaching package: 'rlang'

## The following objects are masked from 'package:purrr':
##
##     %@%, as_function, flatten, flatten_chr, flatten_dbl, flatten_int,
##     flatten_lgl, flatten_raw, invoke, list_along, modify, prepend,
##     splice
```

```
peterpaul <- read.csv("../Data/Processed/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv", st

PeterPaul.chem.nutrients <- read.csv("./Data/Processed/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Proc

processed_litter <- read.csv("../Data/Processed/NEON_NIWO_Litter_mass_trap_Processed.csv", stringsAsFac

#2
class(peterpaul$sampledate)
```

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

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

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

```
PeterPaul.chem.nutrients$sampledate <- as.Date(PeterPaul.chem.nutrients$sampledate, format = "%Y-%m-%d"

class(processed_litter$collectDate)
```

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

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

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

Define your theme

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

```
#3

theme_set(theme_classic())
```

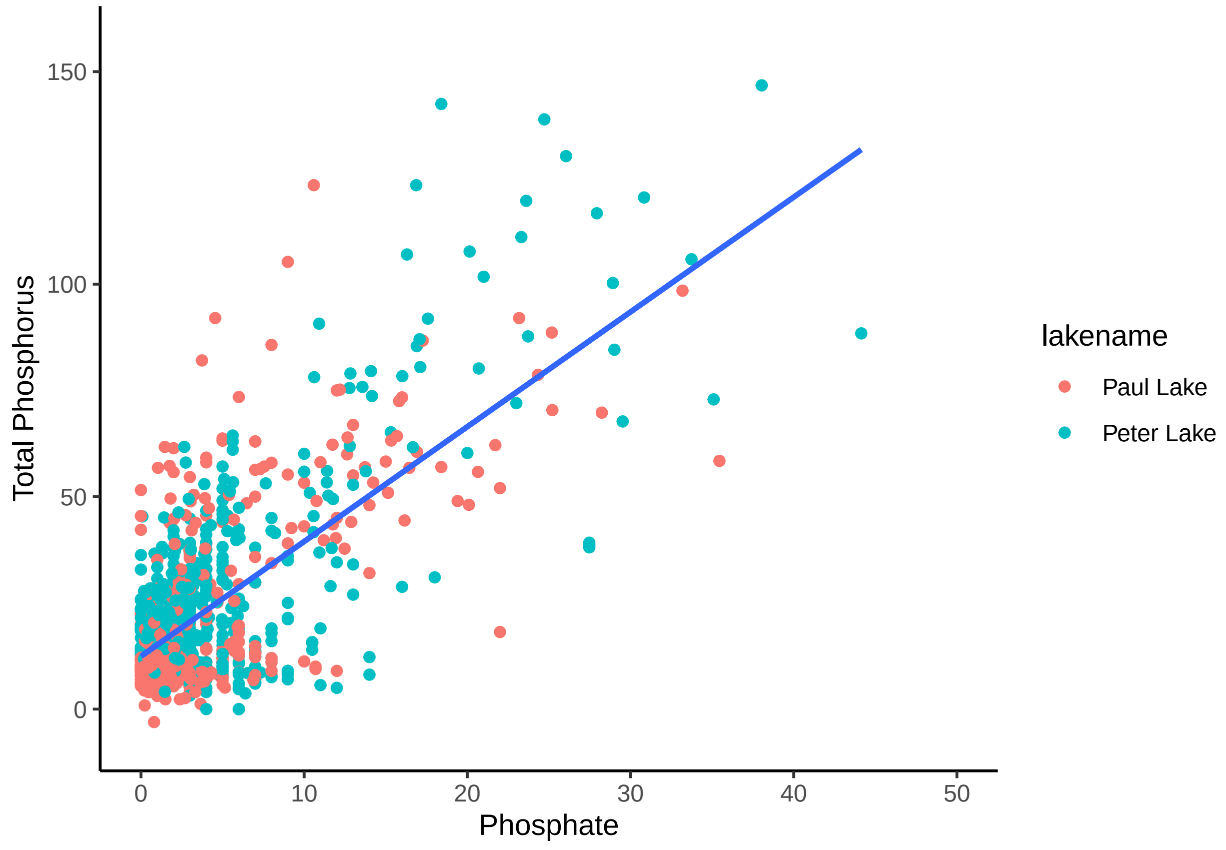
Create graphs

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
phosphorusvsphosphate <-
  ggplot(peterpaul, aes(x = po4, y = tp_ug)) +
  geom_point(aes(colour = lakename)) +
  geom_smooth(method = lm, se=FALSE) +
  xlim(0, 50) +
  labs(x = "Phosphate", y = "Total Phosphorus")


print(phosphorusvsphosphate)
```

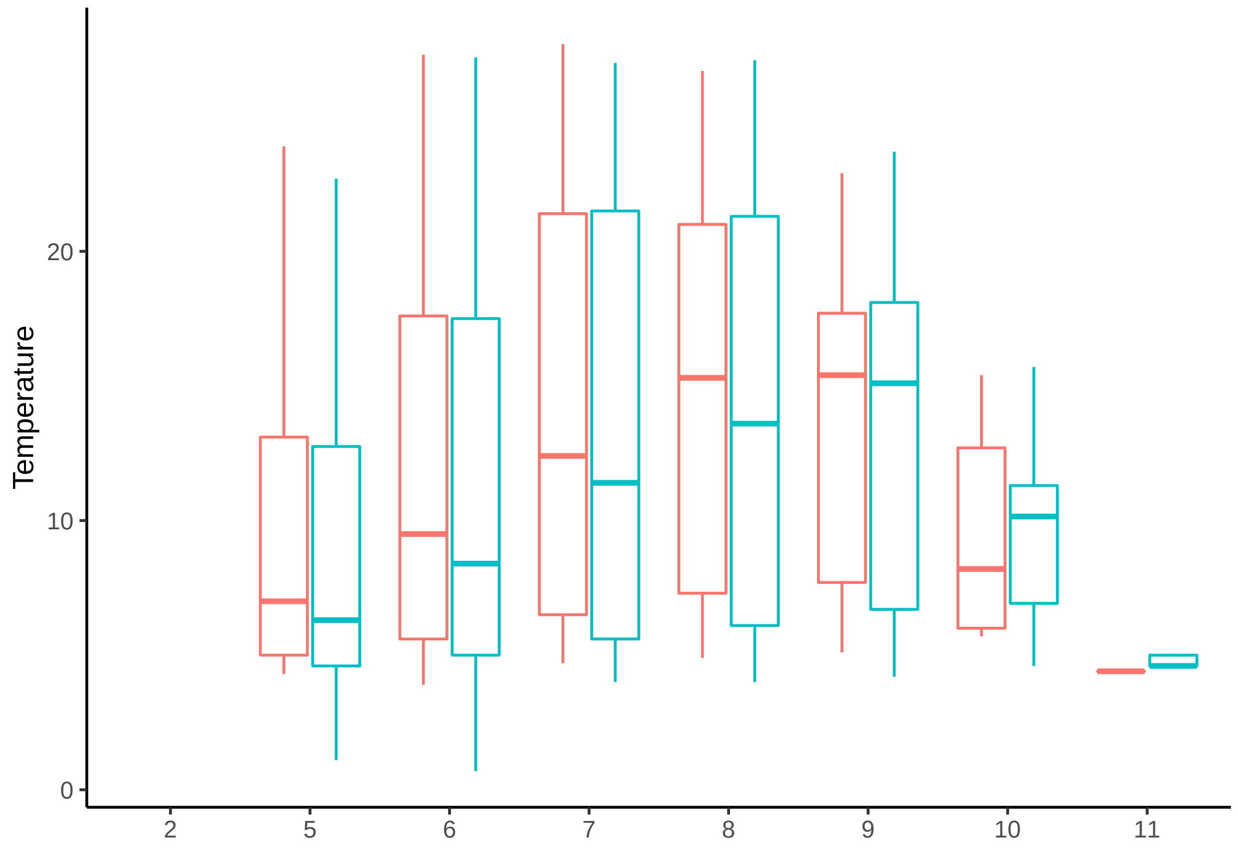
```
## 'geom_smooth()' using formula 'y ~ x'

## Warning: Removed 21947 rows containing non-finite values (stat_smooth).

## Warning: Removed 21947 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.

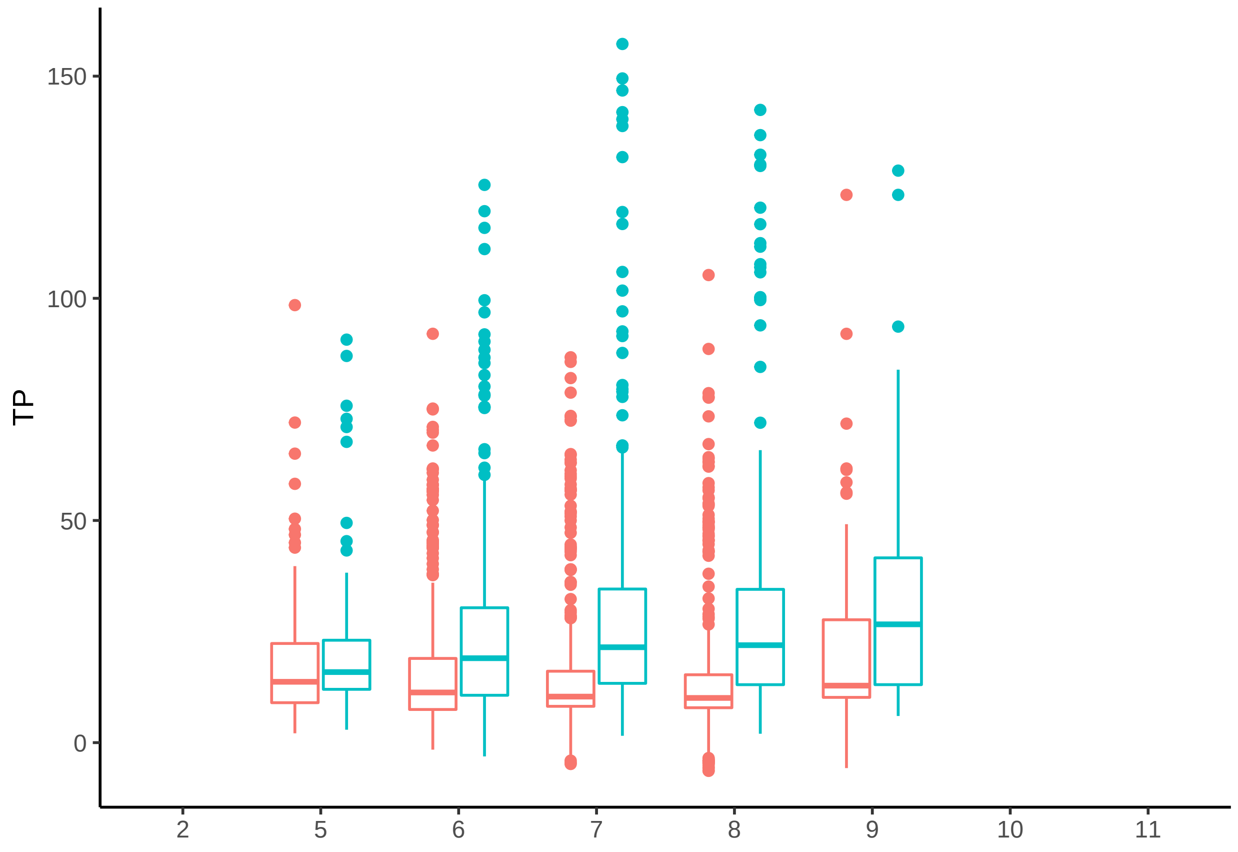
```
#5
temp <- ggplot(PeterPaul.chem.nutrients) +
  geom_boxplot(aes(x = as.factor(month), y = temperature_C, color = lakename), show.legend = FALSE) +
  labs(x = NULL, y = "Temperature")
print(temp)
```

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



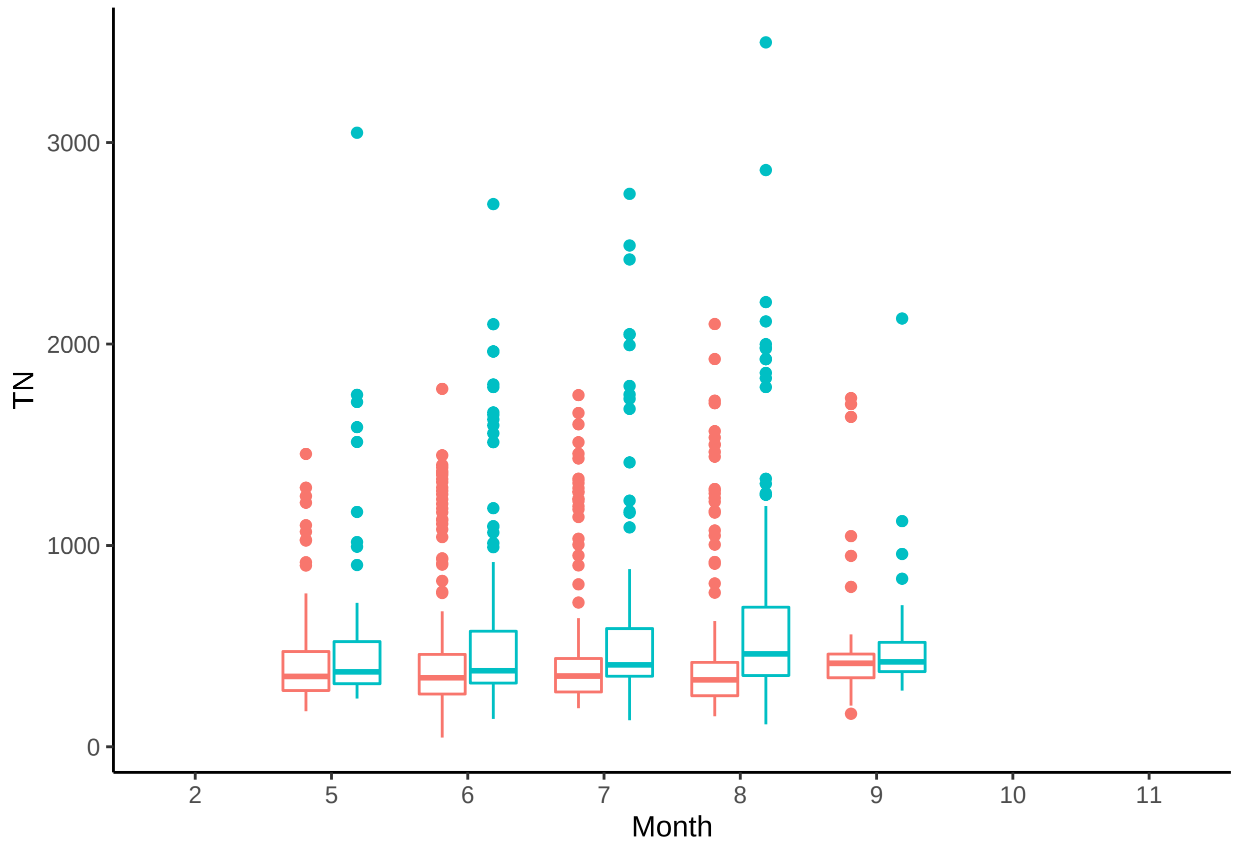
```
tp <- ggplot(PeterPaul.chem.nutrients) +
  geom_boxplot(aes(x = as.factor(month), y = tp_ug, color = lakename), show.legend = FALSE) +
  labs(x = NULL, y = "TP")
print(tp)
```

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



```
tn <- ggplot(PeterPaul.chem.nutrients) +
  geom_boxplot(aes(x = as.factor(month), y = tn_ug, color = lakename), show.legend = FALSE) +
  labs(x = "Month", y = "TN")
print(tn)
```

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



```
Legend <- get_legend(tn)
```

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

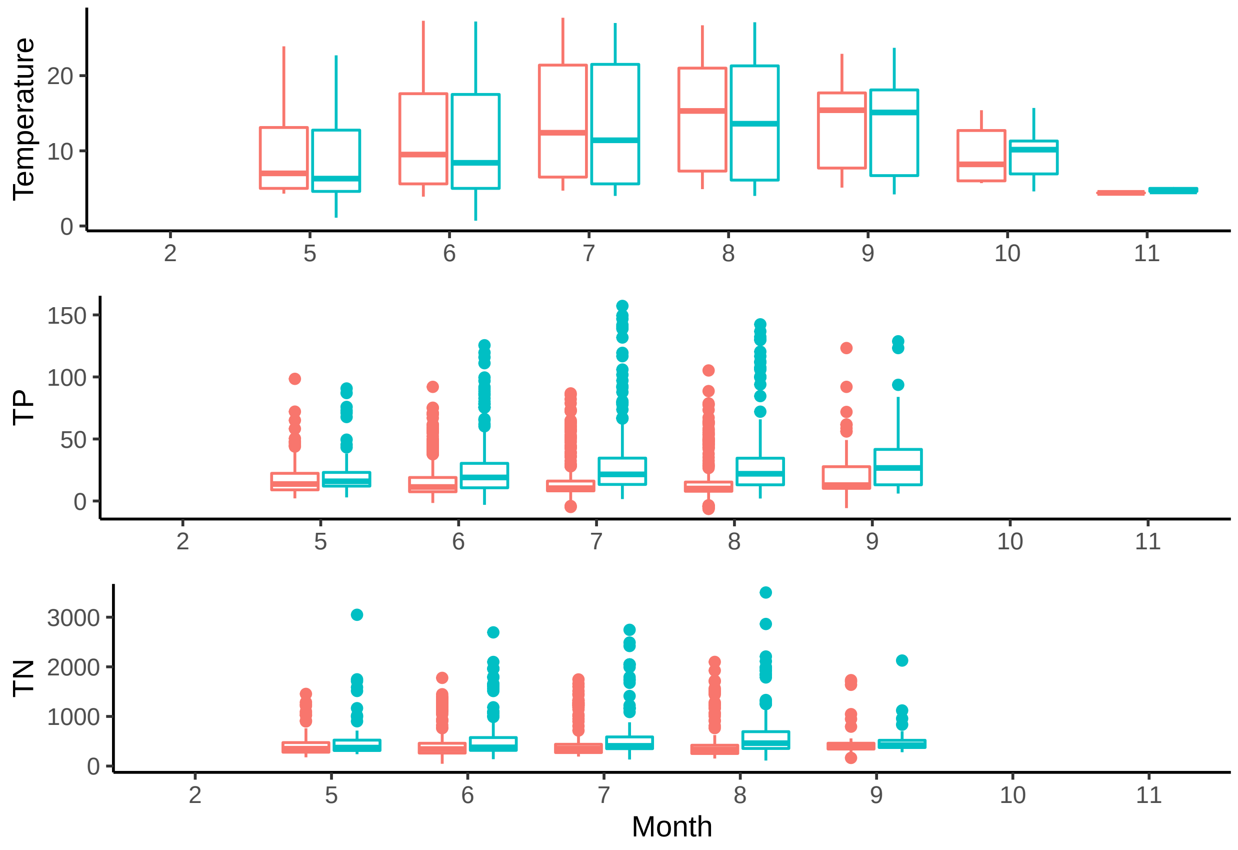
```
CombinedPlot <-
  plot_grid(temp, tp, tn,
            ncol = 1)
```

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

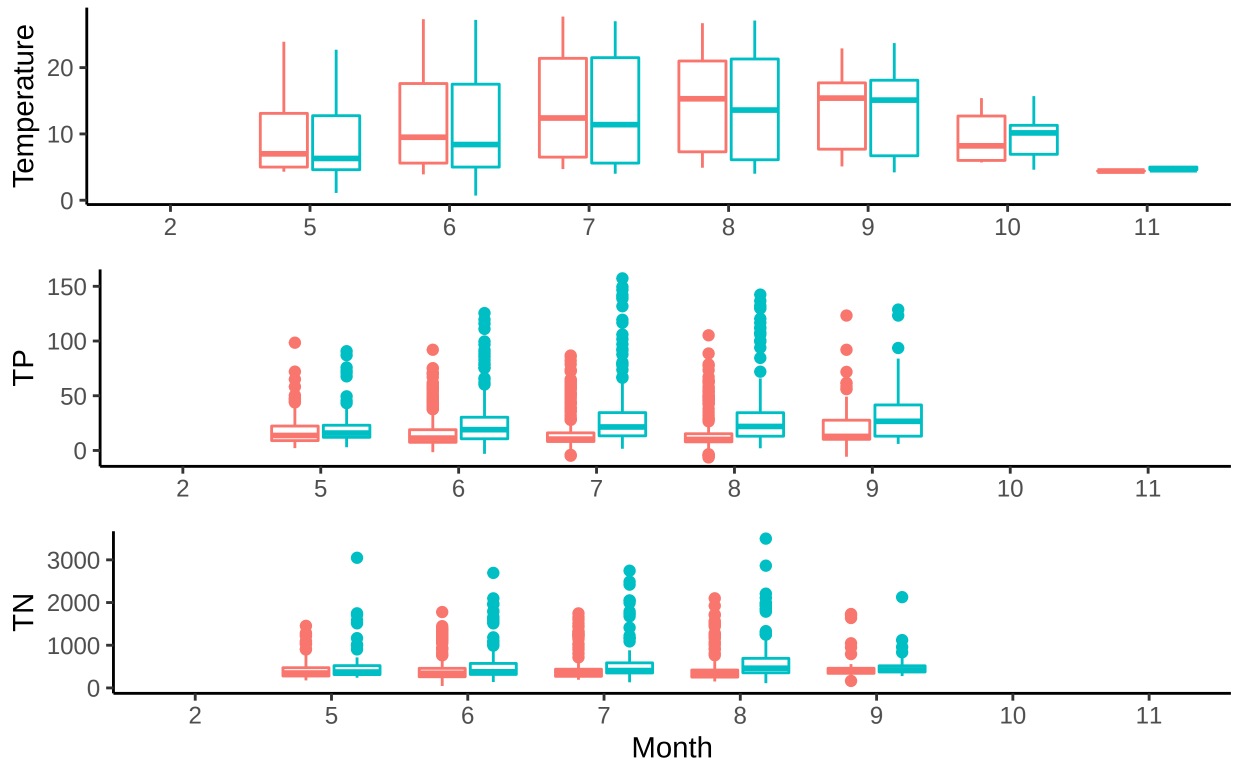
```
## Warning: Removed 20729 rows containing non-finite values (stat_boxplot).
```

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

```
CombinedPlot
```



```
CombinedPlotwithLegend <- plot_grid(CombinedPlot, Legend, nrow = 2, rel_heights = c(3, 0.3))

CombinedPlotwithLegend
```



```
#Can't figure out why my legend isn't showing up

#to get the degree symbol
#ylab(expression(paste("Temperature, ", degree, "C")))
```

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

> Answer: Mean temperatures fluctuate as one might expect in the northern hemisphere with hotter summer months in both lakes. TP and TN do not fluctuate as much as temperature across both lakes. Peter Lake's mean values for TP and TN are consistently slightly above Paul Lake's mean values.

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.

```
#Number 6 and 7 are commented out because
#I keep getting this error message:
#Error in seq.int(0, to0 - from, by) : 'to' must be a finite number.
#I had to comment the lines out to knit my file.
#6
#class(processed_litter$collectDate)
```

```
#
# needles <-
#   ggplot(subset(processed_litter, functionalGroup == "Needles"),
#   aes(x = collectDate, y = dryMass, color = nlcdClass)) +
#   geom_point()
#
# print(needles)

#I keep receiving this Error:
#Error in seq.int(0, to0 - from, by) : 'to' must be a finite number

#7 Getting same error as number 6
# Separate plot with facets
# needles.faceted <-
# ggplot(subset(processed_litter, functionalGroup == "Needles"),
#  aes(x = # collectDate, y = dryMass, color = nlcdClass)) +
#   geom_point()
#   facet_wrap(vars(nlcdClass), nrow = 3)
# print(needles.faceted)
```

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

> Answer: I'd say number 6 because I like how graphs look when variables are separated by color and it gives and clear visual cue of differences.