Assignment 5: Data Visualization

Samantha Burch

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., "Salk_A05_DataVisualization.Rmd") prior to submission.

The completed exercise is due on Tuesday, February 11 at 1:00 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 (tidy and gathered) and the processed data file for the Niwot Ridge litter dataset.

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

```
#1
#load tidyverse and cowplot packages; upload NTL-LTER processed data files (Peter and Paul) and process
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------------------- tidyverse 1.3.0 --
## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
## -- Conflicts ------------------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(cowplot)
```

```
##
## ************************************************************
## Note: As of version 1.0.0, cowplot does not change the
##   default ggplot2 theme anymore. To recover the previous
##   behavior, execute:
##   theme_set(theme_cowplot())
```

```
## ********************************************************
```

```
library(ggplot2)

TIDYPeterPaulChem <- read.csv("./Data/Processed/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.cs
TIDYPeterPaulGathered <- read.csv("./Data/Processed/NTL-LTER_Lake_Nutrients_PeterPaulGathered_Processed

NiwotLitter <- read.csv("./Data/Processed/NEON_NIWO_Litter_mass_trap_Processed.csv")
View(NiwotLitter)

#2
class(TIDYPeterPaulChem$sampledate)
```

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

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

Define your theme

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

```
mytheme <- theme_classic(base_size = 14) +
  theme(axis.text = element_text(color = "blue"),
        legend.position = "top")

theme_set(mytheme)
```

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.

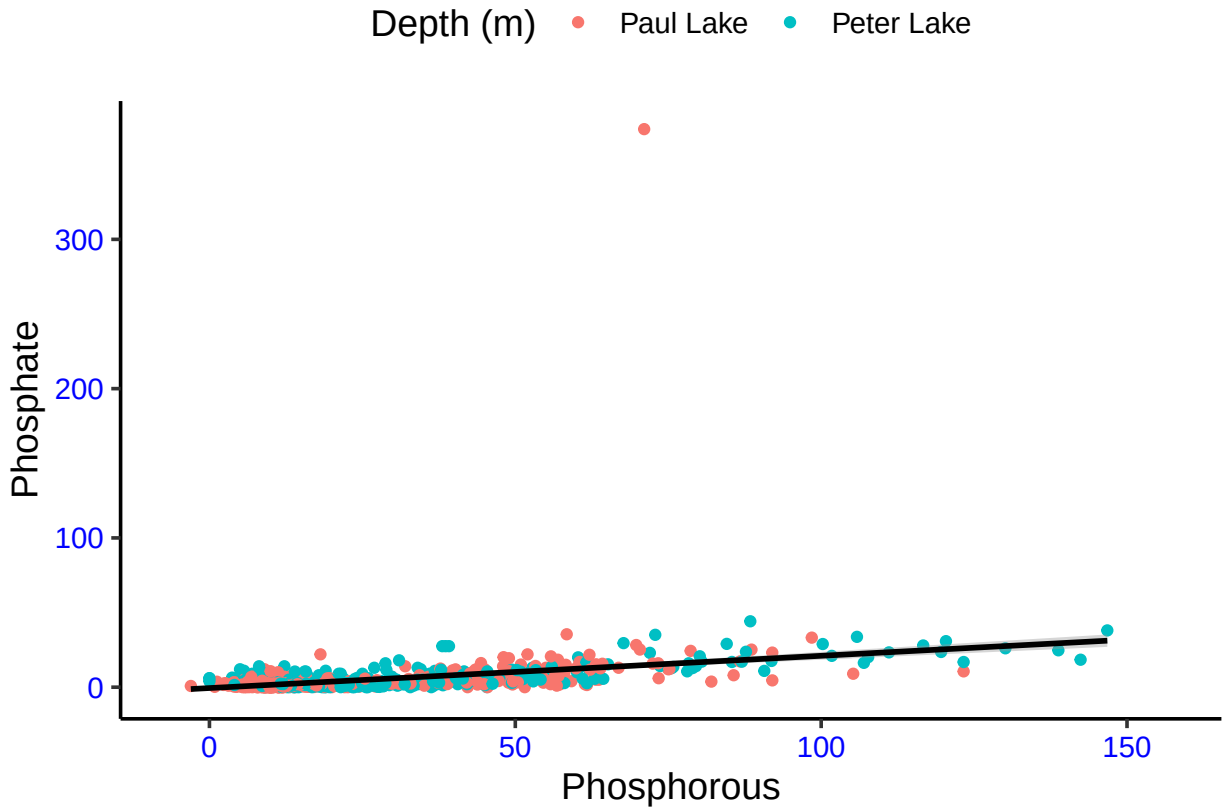
4. [NTL-LTER] Plot total phosphorus by phosphate, with separate aesthetics for Peter and Paul lakes. Add a line of best fit and color it black. Adjust your axes to hide extreme values.

```
library(viridisLite)
library(viridis)
library(RColorBrewer)
library(colormap)


NvsP <-
  ggplot(TIDYPeterPaulChem, aes(x = tp_ug, y = po4)) +
  geom_point(aes(color = lakename)) + #have to have aesthetics bc it was looking for a color named lake
  geom_smooth(method = lm, color = "black") +
  labs(x = "Phosphorous", y = "Phosphate",
       color = "Depth (m)", shape = "")
print(NvsP)
```

```
## Warning: Removed 21946 rows containing non-finite values (stat_smooth).
```

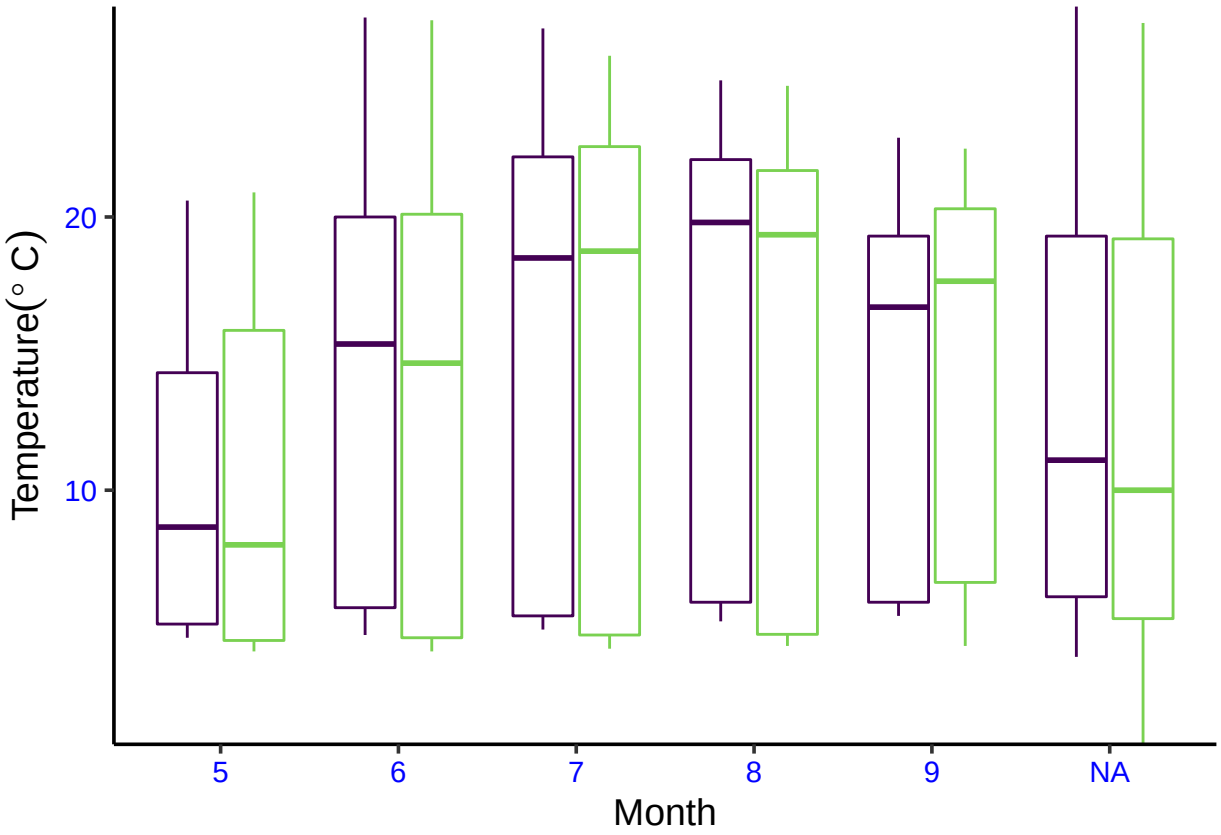
```
## Warning: Removed 21946 rows containing missing values (geom_point).
```



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.

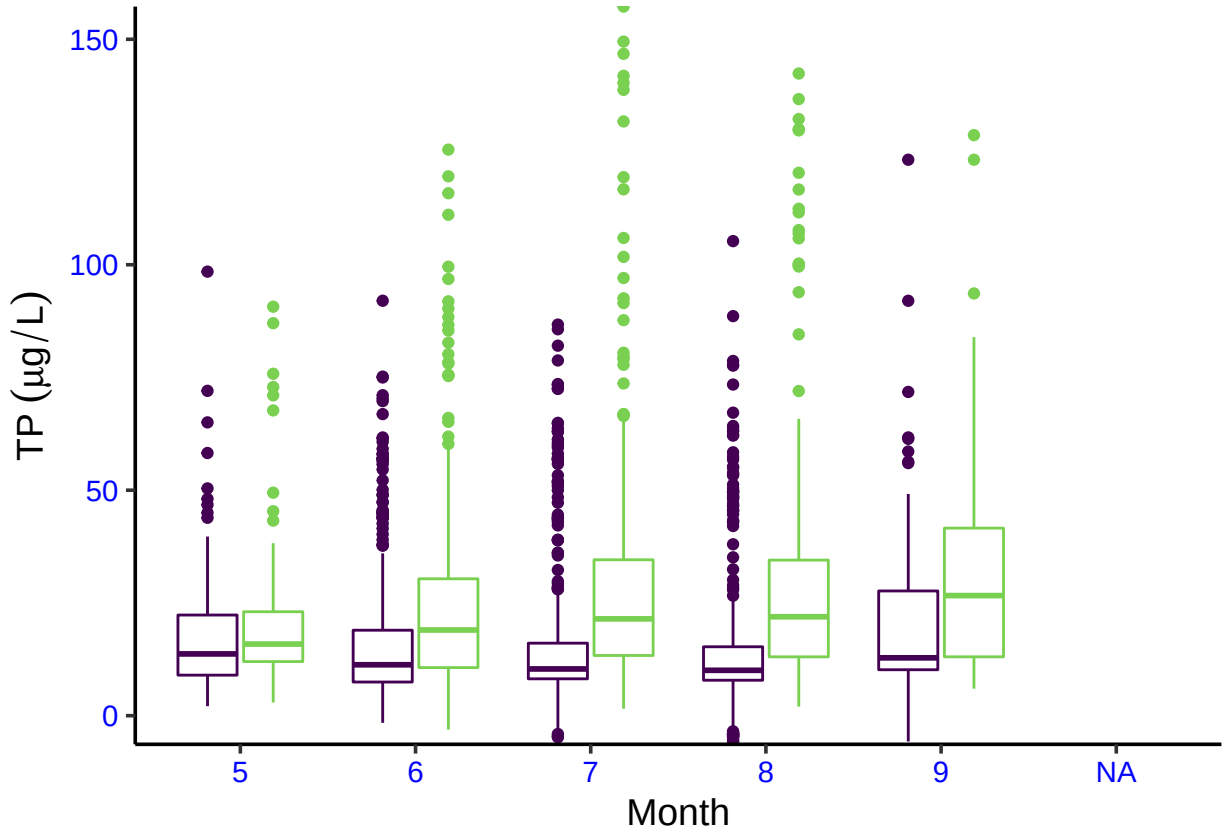
```
#5-1 Temperature Boxplot
Tempboxplot <-
  ggplot(TIDYPeterPaulChem, aes(x = as.factor(month), y = temperature_C, color = lakename)) +
  geom_boxplot() +
  labs(x = expression(paste("Month")),
       y = expression(paste("Temperature" (degree~C))), color = "Lake") +
  scale_y_continuous(expand = c(0,0)) +
  scale_color_viridis(discrete = TRUE, option = "viridis", end = 0.8) +
  theme(legend.position = "none")
print(Tempboxplot)
```

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



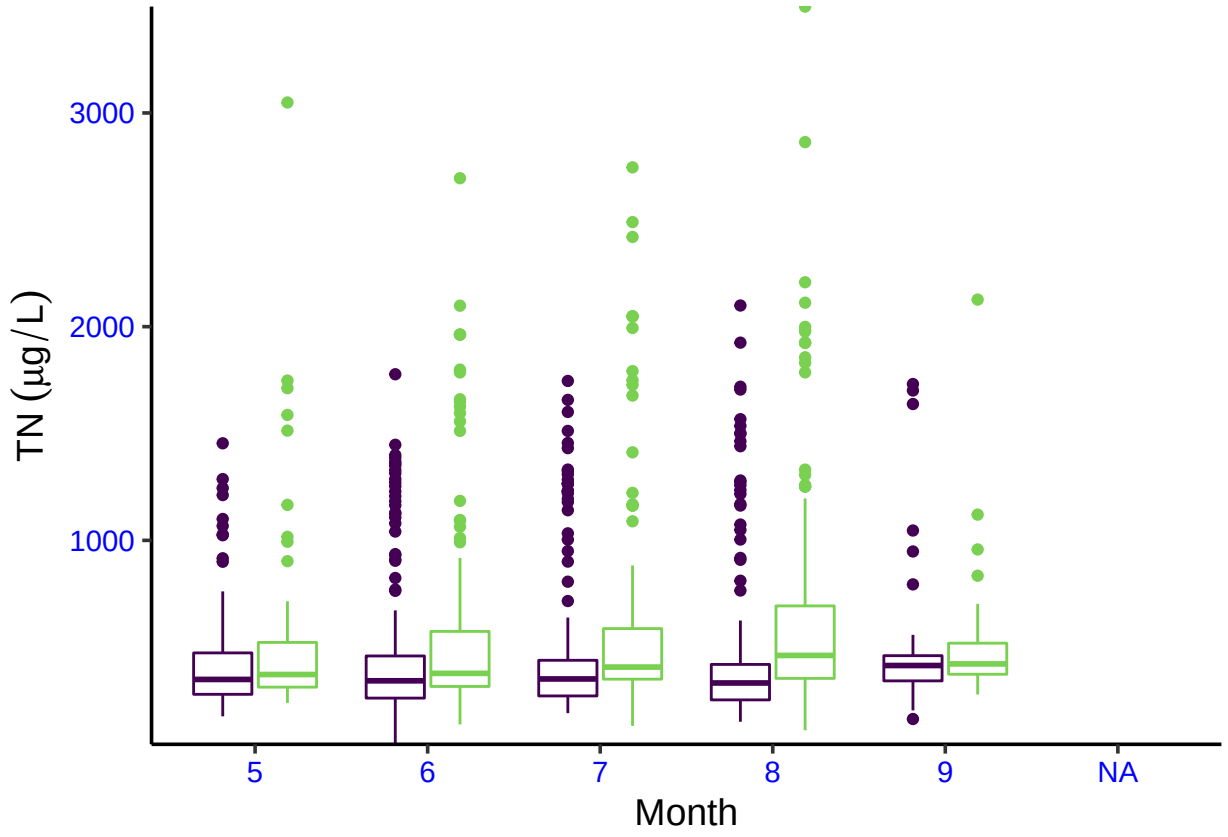

```
#5-2 TP Boxplot
TPboxplot <-
  ggplot(TIDYPeterPaulChem, aes(x = as.factor(month), y = tp_ug, color = lakename)) +
  geom_boxplot() +
  labs(x = expression(paste("Month")),
       y = expression(paste("TP" ~ (mu*g / L))), color = "Lake") +
  scale_y_continuous(expand = c(0,0)) +
  scale_color_viridis(discrete = TRUE, option = "viridis", end = 0.8) +
  theme(legend.position = "none")
print(TPboxplot)
```

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




```
#5-3 TN Boxplot
TNboxplot <-
  ggplot(TIDYPeterPaulChem, aes(x = as.factor(month), y = tn_ug, color = lakename)) +
  geom_boxplot() +
  labs(x = expression(paste("Month")),
       y = expression(paste("TN" ~ (mu*g / L))), color = "Lake") +
  scale_y_continuous(expand = c(0,0)) +
  scale_color_viridis(discrete = TRUE, option = "viridis", end = 0.8) +
  theme(legend.position = "none")
print(TNboxplot)
```

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




```
#5-4 Cowplot that combines all three

threeplots <- plot_grid(Tempboxplot, TPboxplot, TNboxplot, Labels = c('Temp', 'TP', 'TN'),
                    nrow = 3, align = 'v', rel_heights = c(1.25, 1, 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).

## Warning in as_grob.default(plot): Cannot convert object of class character into
## a grob.

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

```
view(threeplots) #still issues
```

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

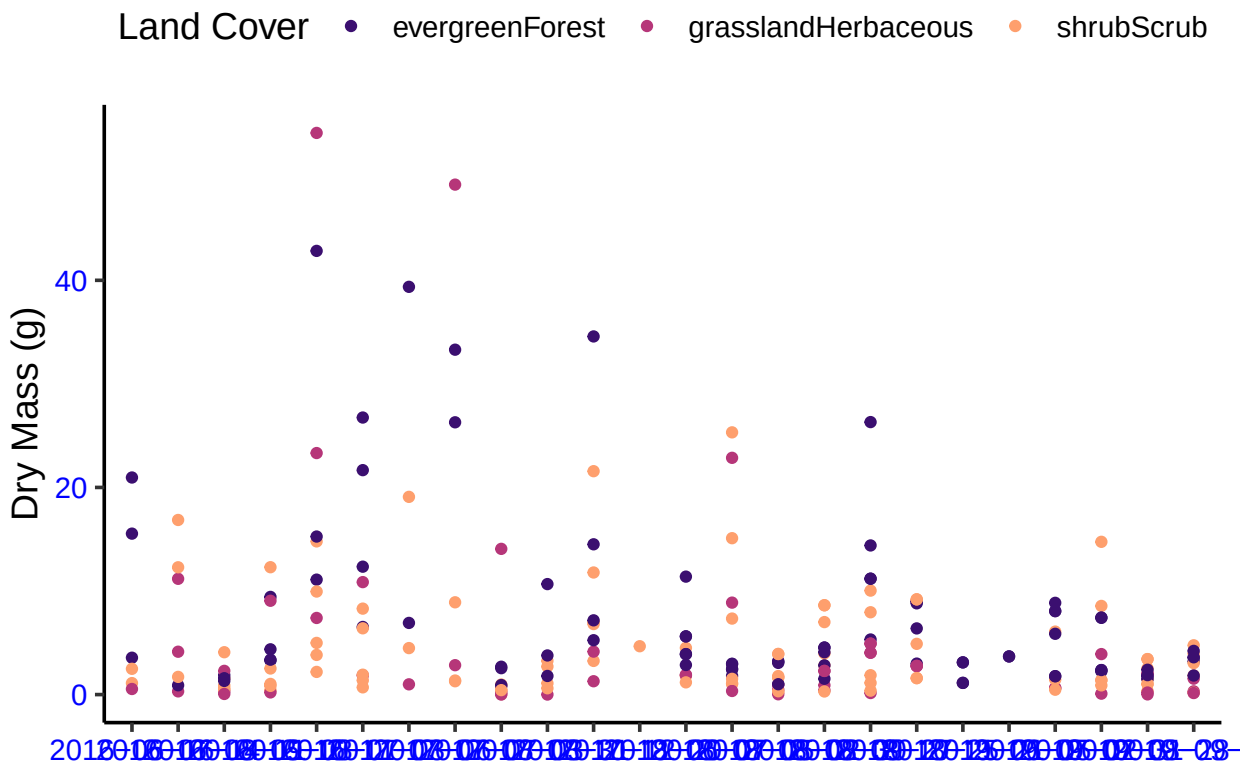
> Answer: When looking across the seasons, it becomes apparent that Paul Lake has higher temperatures and lower TP and TN levels during the summer months. It is possible that this is due to the fact that samples were taken during months warmer than those taken from Peter Lake (May/June vs July/August).

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.

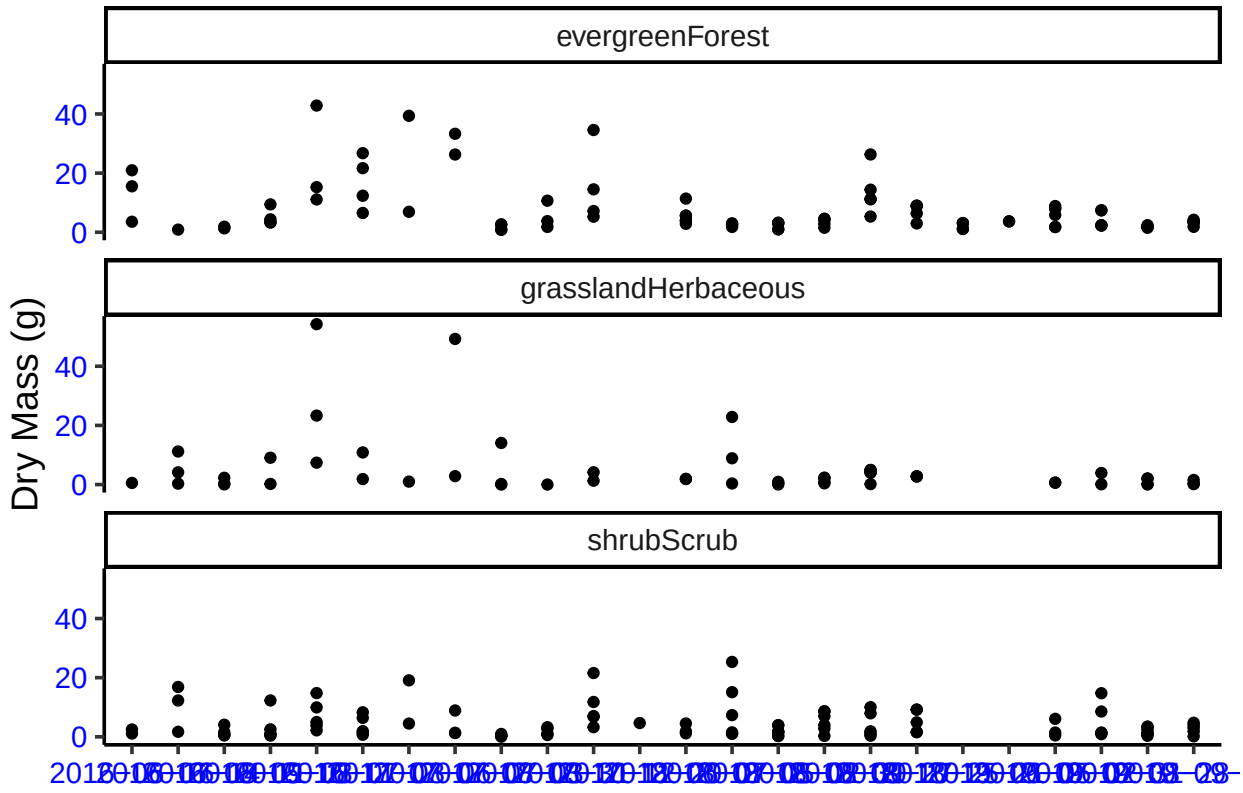
```
#6
NIWOneedles <-
  ggplot(subset(NiwotLitter, functionalGroup == "Needles")) + #why double equal signs?
           geom_point(aes(x = collectDate, y = dryMass, color = nlcdClass)) +
           scale_color_viridis_d(option = "magma", begin = 0.2, end = 0.8) +
           theme(legend.position = "top") +
           labs(x = "", y = "Dry Mass (g)", color = "Land Cover")

print(NIWOneedles) #how to fix x axis so I can see all collectDates? niwo needles not found
```




```
#7
NIWOneedlesFacets <-
  ggplot(subset(NiwotLitter, functionalGroup == "Needles")) +
  geom_point(aes(x = collectDate, y = dryMass)) +
  facet_wrap(vars(nlcdClass), nrow = 3) +
  labs(x = "", y = "Dry Mass (g)")

print(NIWOneedlesFacets)
```



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

Answer: The use of colored dots can be somewhat difficult to see, and prove challenging to compare the data across the different land covers. If one wanted to look directly at a specific land cover type and see the data points more clearly, the facet option in #7 is most effective.