Assignment 5: Data Visualization

Analise Lindborg

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 Tuesday, February 23 at 11:59 pm.

```
## My document will not knit without this code

knitr::opts_knit$set(root.dir = '/Users/analiselindborg/Desktop/Desktop - Analise's MacBook Pro/Data Ana
```

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 (both the tidy [`NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv`] and the gathered [`NTL-LTER_Lake_Nutrients_PeterPaulGathered_Processed.csv`] versions) 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
getwd()
```

```
## [1] "/Users/analiselindborg/Desktop/Desktop - Analise's MacBook Pro/Data Analytics/Environmental_Data
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.0.5     v dplyr   1.0.3
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0

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

```
library(cowplot)
```

```
peterpaul.chem <- read.csv("./Data/Processed/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv"
                           stringsAsFactors = TRUE)
peterpaul.nut <- read.csv("./Data/Processed/NTL-LTER_Lake_Nutrients_PeterPaulGathered_Processed.csv",
                          stringsAsFactors = TRUE)
neon <- read.csv("./Data/Processed/NEON_NIWO_Litter_mass_trap_Processed.csv",
                 stringsAsFactors = TRUE)

#2
peterpaul.chem$sampledate <- as.Date(peterpaul.chem$sampledate, format = "%Y-%m-%d")
peterpaul.nut$sampledate <- as.Date(peterpaul.nut$sampledate, format = "%Y-%m-%d")
neon$collectDate <- as.Date(neon$collectDate, format = "%Y-%m-%d")
```

Define your theme

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

```
theme <-
  theme_bw() +
  theme(panel.grid.minor = element_blank(),
        text = element_text(color = "black", size = 10),
        axis.text.x = element_text(color = "black"),
        axis.text.y = element_text(color = "black"))

theme_set(theme)
```

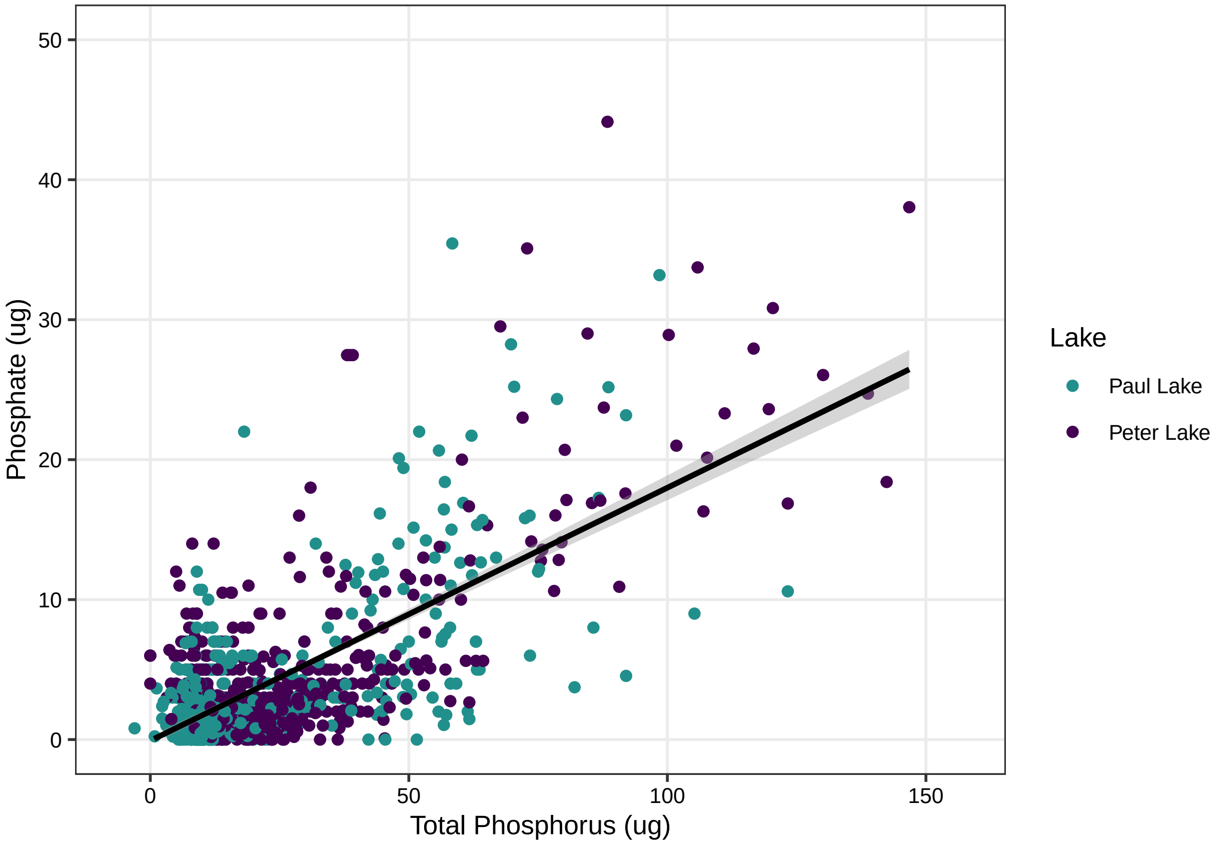
Create graphs

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.

```
ggplot(peterpaul.chem, aes(x = tp_ug, y = po4, color = lakename)) +
  geom_point()+
  scale_color_viridis_d(begin = 0.5, end = 0, name = "Lake") +
  geom_smooth(method = lm, color = "black") +
  ylim(0, 50) +
  labs(x = "Total Phosphorus (ug)",  y = "Phosphate (ug)") +
  theme
```

```
## `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).

## Warning: Removed 2 rows containing missing values (geom_smooth).
```



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.

```
#Create month abbreviations so that the x-axis is more easily interpreted (as opposed to numbers)
peterpaul.chem$month.name <- month.abb[peterpaul.chem$month]

#Factor so that months appear in order
peterpaul.chem$month.name = factor(peterpaul.chem$month.name,
                                   levels=c("Feb", "May", "Jun",
                                            "Jul", "Aug", "Sep",
                                            "Oct", "Nov"))

#temp plot
temp <- ggplot(peterpaul.chem, aes(x = month.name, y = temperature_C, fill = lakename))+
  geom_boxplot() +
   scale_fill_viridis_d(begin = 0.6, end = 0.2, name = "Lake") +
  labs(y = "Temperature (Celcius)") +
  ylim(0,30) +
  xlab(NULL) +
  theme+
  theme(panel.grid.major = element_blank(),
        legend.position = "none")

#TP plot
tp <- ggplot(peterpaul.chem, aes(x = month.name, y = tp_ug, fill = lakename))+
  geom_boxplot() +
```

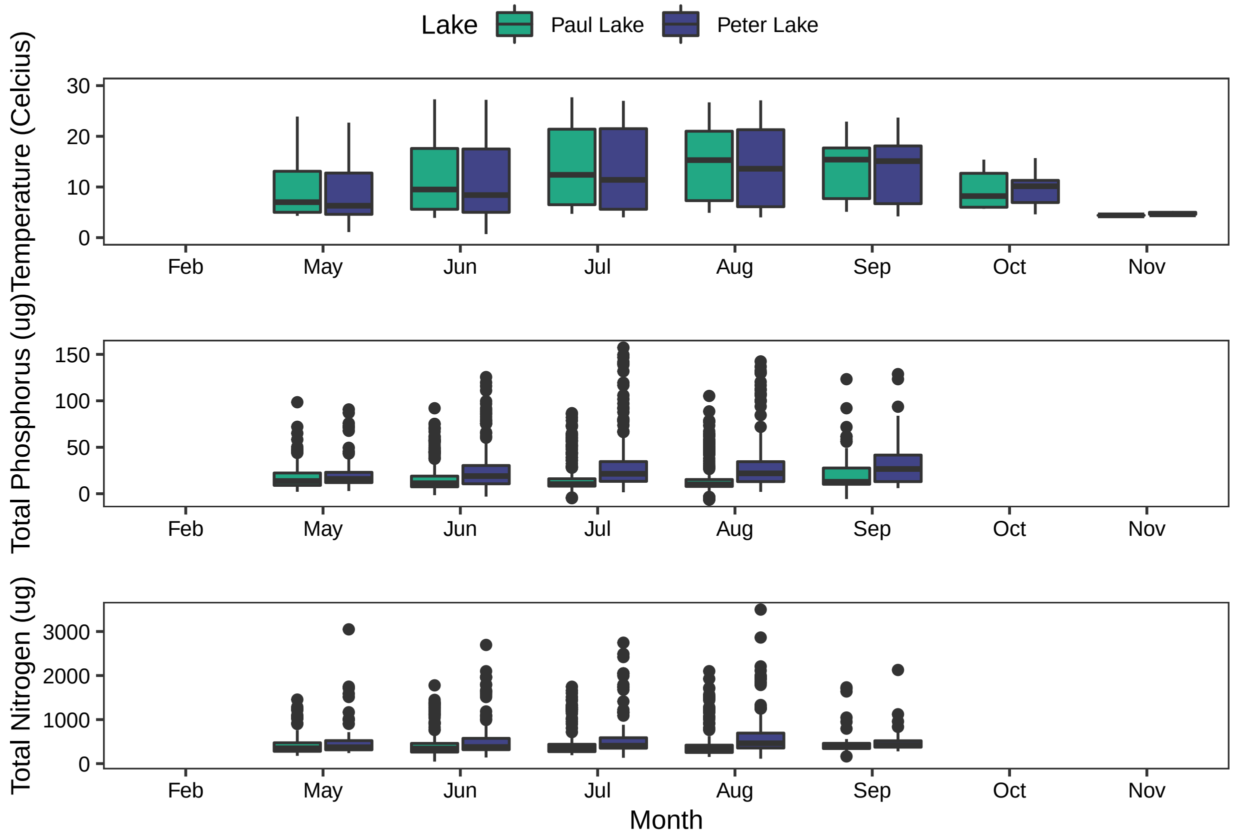
```r
   scale_fill_viridis_d(begin = 0.6, end = 0.2, name = "Lake") +
  labs(y = "Total Phosphorus (ug)") +
  xlab(NULL) +
  theme+
  theme(panel.grid.major = element_blank(),
        legend.position = "none")

#TN plot
tn <- ggplot(peterpaul.chem, aes(x = month.name, y = tn_ug, fill = lakename))+
  geom_boxplot() +
   scale_fill_viridis_d(begin = 0.6, end = 0.2, name = "Lake") +
  labs(x = "Month", y = "Total Nitrogen (ug)") +
  theme+
  theme(panel.grid.major = element_blank(),
        legend.position = "none")

#create plot
p <- plot_grid(temp, tp, tn, align = 'vh', 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).
```

```r
#Create legend
legend <- get_legend(
  temp +
    guides(color = guide_legend(nrow = 1)) +
    theme(legend.position = "top"))
```

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

```r
#add legend to plot
plot_grid(legend, p, ncol = 1, rel_heights = c(.1, 1))
```

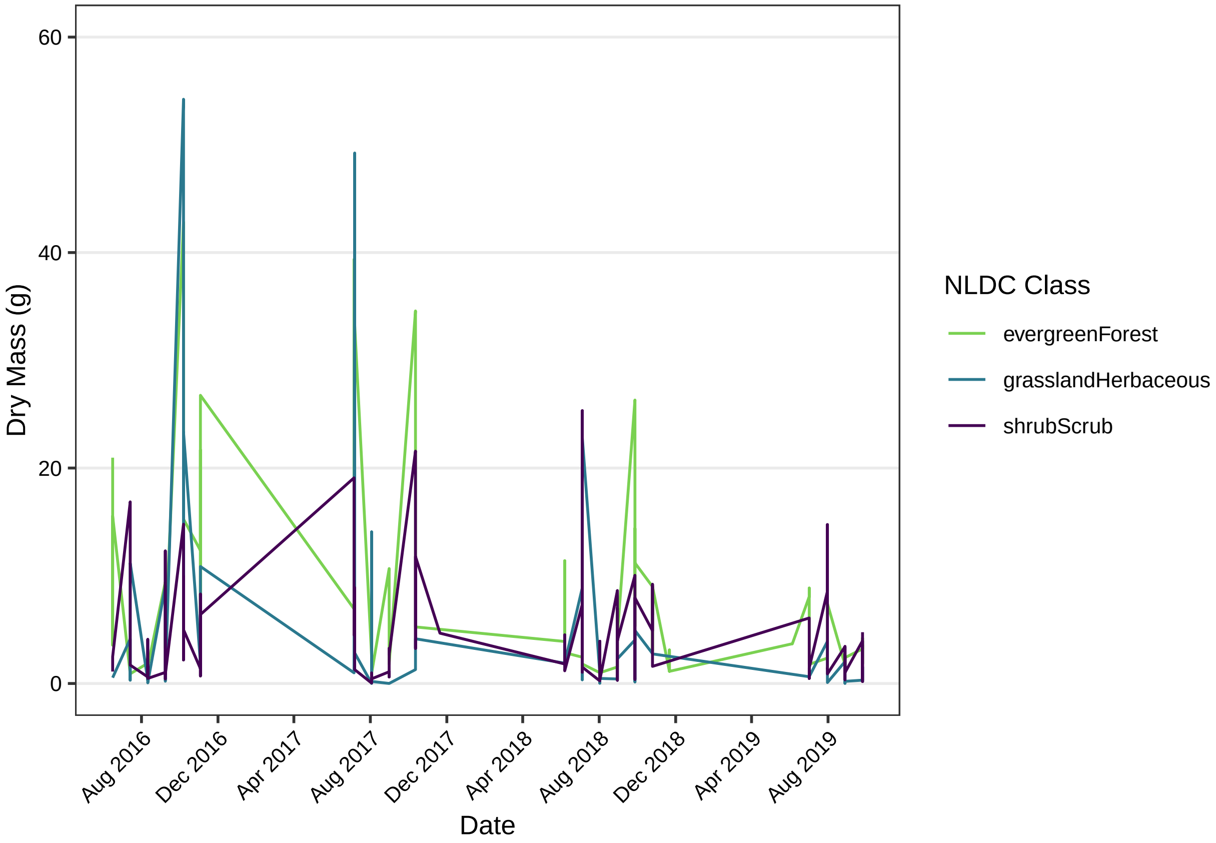


Question: What do you observe about the variables of interest over seasons and between lakes?

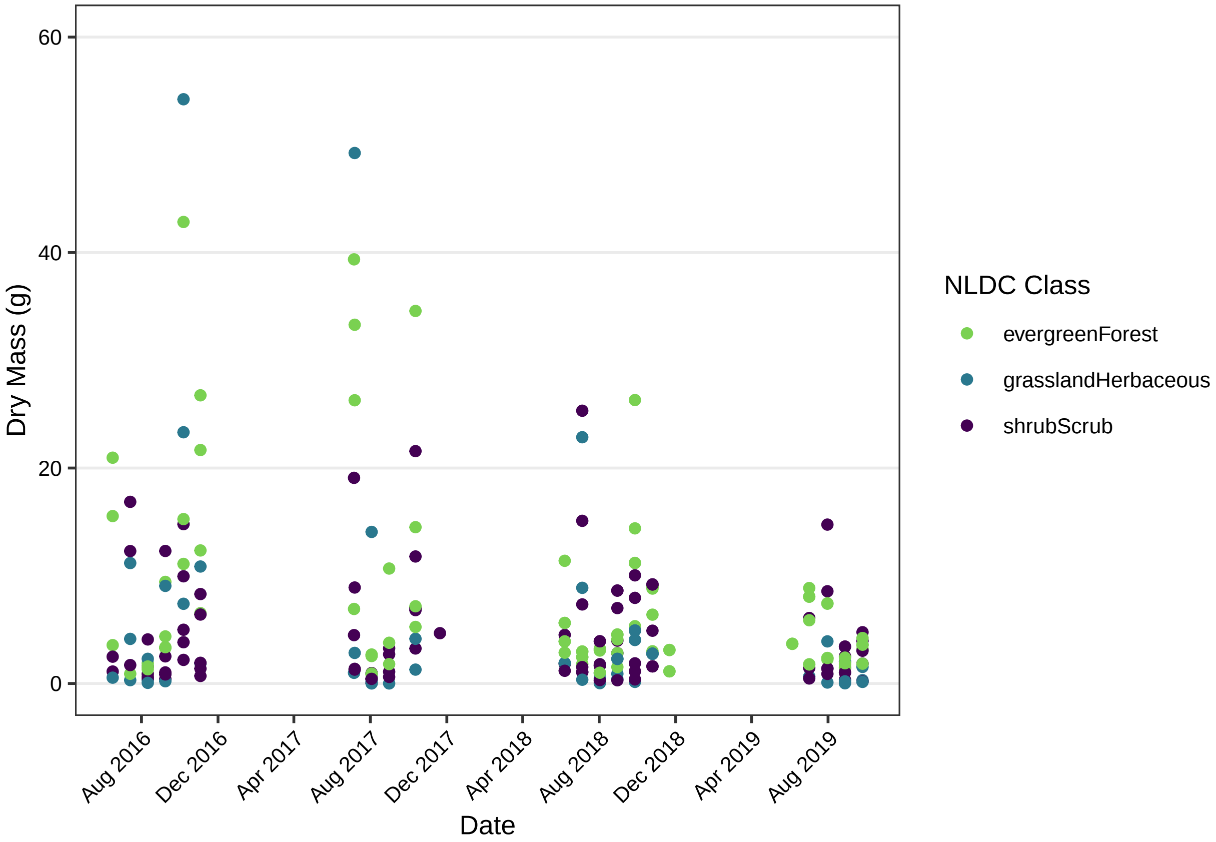
> Answer: Variables of interest are all relatively similar between lakes. Temperature peaks in summer, which would be expected. TN and TP have many outliers and don't seem to fluctuate with season.

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.

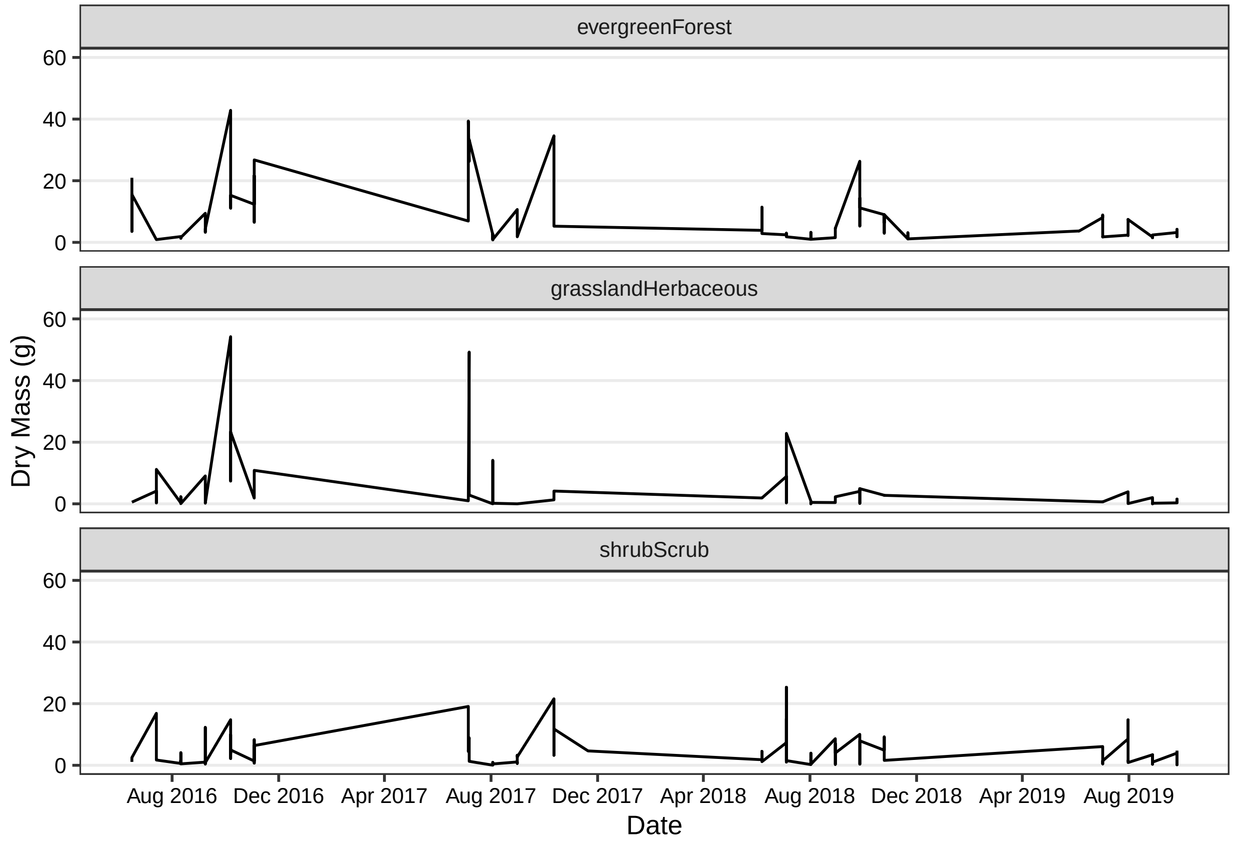
```r
#6
#Used line plot because this is typically best representation of continuous variable over time.
# However, a scatterplot may be better in this instance because of large breaks in time periods,
# so I also plotted that.
ggplot(subset(neon, functionalGroup == "Needles"), aes(x = collectDate, y = dryMass, color = nlcdClass)
  geom_line() +
  scale_x_date(date_breaks = "4 months", date_labels = "%b %Y") +
  scale_color_viridis_d(begin = .8, end = 0, name = "NLDC Class") +
  labs(x = "Date", y = "Dry Mass (g)")+
  ylim(0,60)+
  theme +
  theme(panel.grid.major.x = element_blank(),
        axis.text.x = element_text(angle = 45, hjust = 1))
```



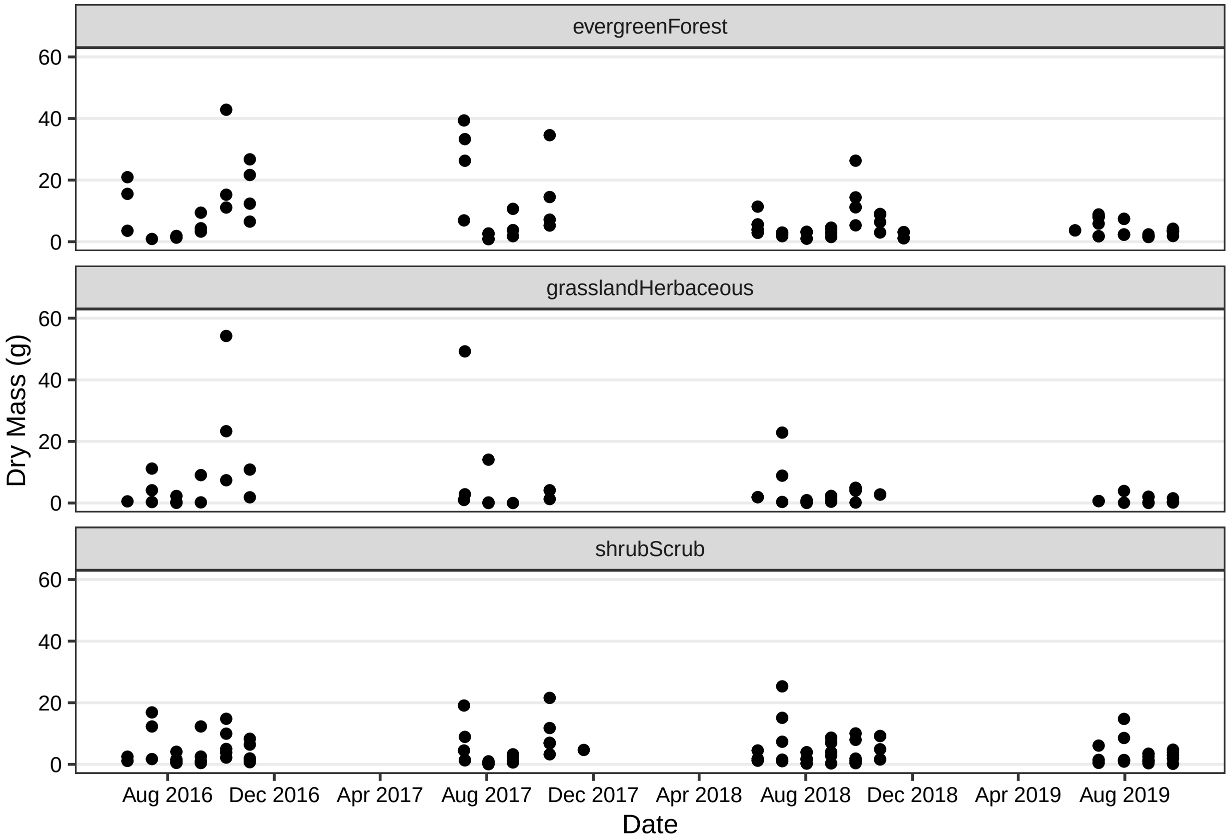
```
ggplot(subset(neon, functionalGroup == "Needles"), aes(x = collectDate, y = dryMass, color = nlcdClass)
  geom_point() +
  scale_x_date(date_breaks = "4 months", date_labels = "%b %Y") +
  scale_color_viridis_d(begin = .8, end = 0, name = "NLDC Class") +
  labs(x = "Date", y = "Dry Mass (g)")+
  ylim(0,60)+
  theme +
  theme(panel.grid.major.x = element_blank(),
        axis.text.x = element_text(angle = 45, hjust = 1))
```



```
#7
ggplot(subset(neon, functionalGroup == "Needles"), aes(x = collectDate, y = dryMass)) +
  geom_line() +
  scale_x_date(limits = as.Date(c("2016-06-16", "2019-09-25")),
    date_breaks = "4 months", date_labels = "%b %Y") +
  labs(x = "Date", y = "Dry Mass (g)")+
  ylim(0,60)+
  theme +
  theme(panel.grid.major.x = element_blank()) +
  facet_wrap(~nlcdClass, ncol = 1)
```



```
ggplot(subset(neon, functionalGroup == "Needles"), aes(x = collectDate, y = dryMass)) +
  geom_point() +
  scale_x_date(limits = as.Date(c("2016-06-16", "2019-09-25")),
    date_breaks = "4 months", date_labels = "%b %Y") +
  labs(x = "Date", y = "Dry Mass (g)")+
  ylim(0,60)+
  theme +
  theme(panel.grid.major.x = element_blank()) +
  facet_wrap(~nlcdClass, ncol = 1)
```



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

Answer:Facet is more effective because the lines/points overlap frequently and it is hard to make any distinction between the three NLCD classes. Viewing on a facetted plot allows for better visual interpretation of the individual trends. Comparisons can be made by keeping the y-axis constant (not free scaling).