EmmaBeyer_A05_DataVisualization.Rmd

Emma Beyer

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
# loading packages
library(tidyverse); library(lubridate); library(here); library(cowplot); library(ggthemes)
```

```
## -- 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/EDA/EDE_Fall2023
##
##
## Attaching package: 'cowplot'
##
##
## The following object is masked from 'package:lubridate':
##
##     stamp
##
##
##
## Attaching package: 'ggthemes'
##
##
## The following object is masked from 'package:cowplot':
##
##     theme_map
```

```
# checking working directory
here()
```

```
## [1] "/home/guest/EDA/EDE_Fall2023"
```

```
# reading data
PeterPaul.nutrients <-
  read.csv(here("Data/Processed_KEY/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv"),
           stringsAsFactors = TRUE)
PeterPaul.litter <-
  read.csv(here("Data/Processed_KEY/NEON_NIWO_Litter_mass_trap_Processed.csv"),
           stringsAsFactors = TRUE)

#2
# check dates
class(PeterPaul.nutrients$sampledate)
```

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

```
class(PeterPaul.litter$collectDate)
```

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

```
# change factors to dates
PeterPaul.nutrients$sampledate <- ymd(PeterPaul.nutrients$sampledate)
PeterPaul.litter$collectDate <- ymd(PeterPaul.litter$collectDate)
# recheck dates
class(PeterPaul.nutrients$sampledate)
```

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

```r
class(PeterPaul.litter$collectDate)
```

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

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

```r
#3
# assigning default theme
my_theme <- theme_base()  +
        # changing background color
  theme(plot.background =  element_rect(color = 'grey'),
        # changing legend position
        legend.position = 'right',
        # changing axis lines to blue
        panel.grid.major = element_line(color = "blue"),
        panel.grid.minor = element_line(color = "blue"))

# setting my_theme as default
theme_set(my_theme)
```

Create graphs

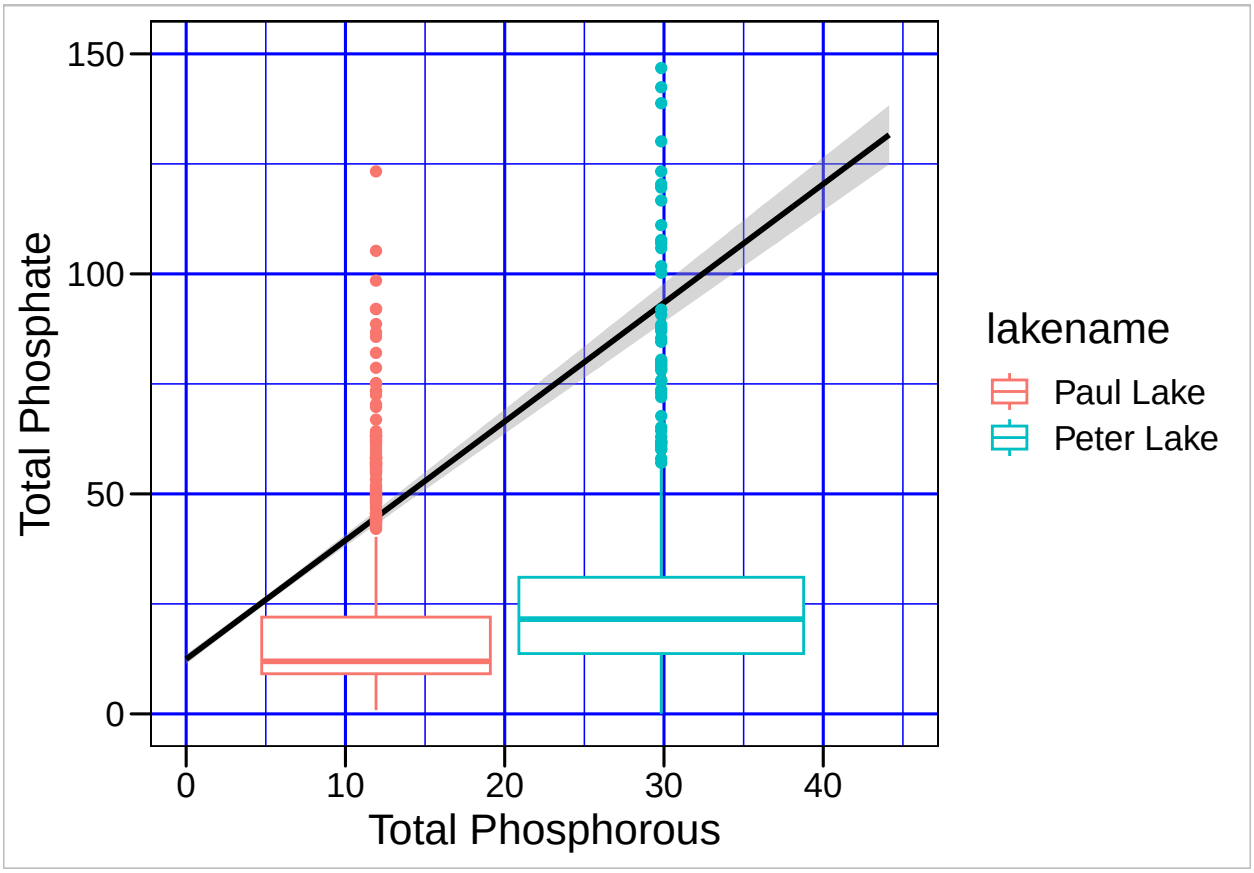
For numbers 4-7, create ggplot graphs and adjust aesthetics to follow best practices for data visualization. Ensure your theme, color palettes, axes, and additional aesthetics are edited accordingly.

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/or `ylim()`).

```r
#4
tp_po4_plot <- ggplot(filter(PeterPaul.nutrients),
              # plotting phosphorus by phosphate and plotting lakes with different colors
              aes(y=tp_ug,
                  x=po4,
                  color=lakename)) +
              # change x axis label
              xlab("Total Phosphorous") +
              # change y axis label
              ylab("Total Phosphate") +
              # adding line of best fit colored black
              geom_smooth(method = "lm",
                          color="black") +
```

```
              geom_boxplot() +
              # changing the limits
              xlim(0, 45) +
              ylim(0, 150)

print(tp_po4_plot)
```

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

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

## Warning: Removed 21854 rows containing missing values (`stat_boxplot()`).

## Warning: Removed 94 rows containing non-finite values (`stat_boxplot()`).
```



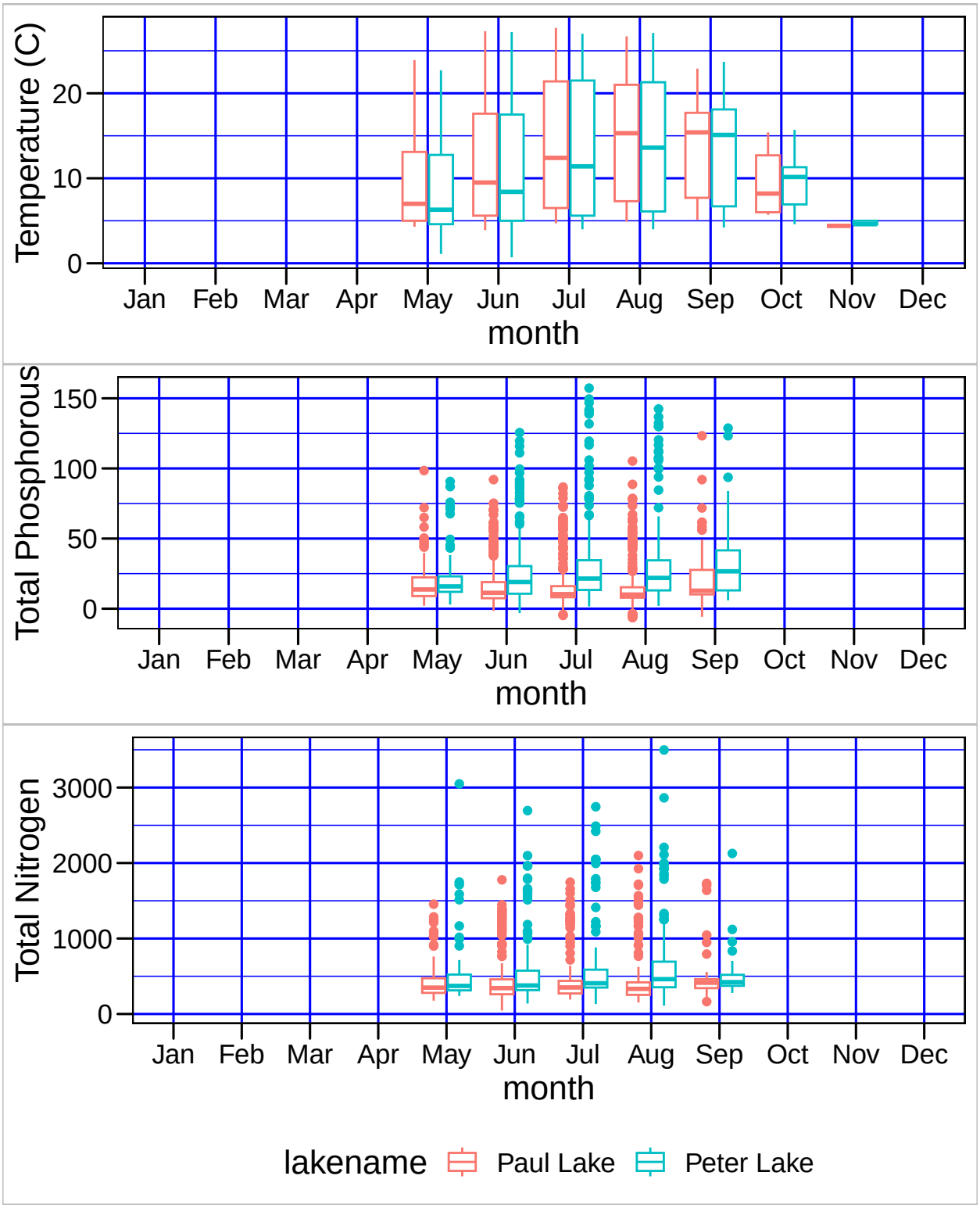
5. [NTL-LTER] Make three separate boxplots of (a) temperature, (b) TP, and (c) TN, with month as the x axis and lake as a color aesthetic. Then, create a cowplot that combines the three graphs. Make sure that only one legend is present and that graph axes are aligned.

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

```r
#5
temp_plot <- ggplot(filter(PeterPaul.nutrients),
           # plotting temperature by month
            aes(y=temperature_C,
                x=factor(month,
                         levels=1:12,
                         labels=month.abb),
                color=lakename)) +
          # change x axis label
          xlab("Month") +
          # change y axis label
          ylab("Temperature (C)") +
          # choosing box plot
          geom_boxplot() +
          # adding x axis name
          scale_x_discrete(name="month",
                           drop=FALSE)

tp_plot <- ggplot(filter(PeterPaul.nutrients),
           # plotting temperature by month
            aes(y=tp_ug,
                x=factor(month,
                         levels=1:12,
                         labels=month.abb),
                color=lakename)) +
          # change x axis label
          xlab("Month") +
          # change y axis label
          ylab("Total Phosphorous") +
          # choosing box plot
          geom_boxplot() +
          # adding x axis name
          scale_x_discrete(name="month",
                           drop=FALSE)

tn_plot <- ggplot(filter(PeterPaul.nutrients),
           # plotting temperature by month
            aes(y=tn_ug,
                x=factor(month,
                         levels=1:12,
                         labels=month.abb),
                color=lakename)) +
          # change x axis label
          xlab("Month") +
          # change y axis label
          ylab("Total Nitrogen") +
          # choosing box plot
          geom_boxplot() +
          # adding x axis name
          scale_x_discrete(name="month",
                           drop=FALSE)

# adding above graphs to one plot
```

```r
plot_grid(temp_plot + theme(legend.position = "none"),
          tp_plot + theme(legend.position = "none"),
          tn_plot + theme(legend.position = "bottom"),
          nrow = 3,
          align = 'h',
          rel_heights = c(1.5, 1.5, 2)) +
  theme(axis.text = element_text(size = 10))
```

```
## 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.
```



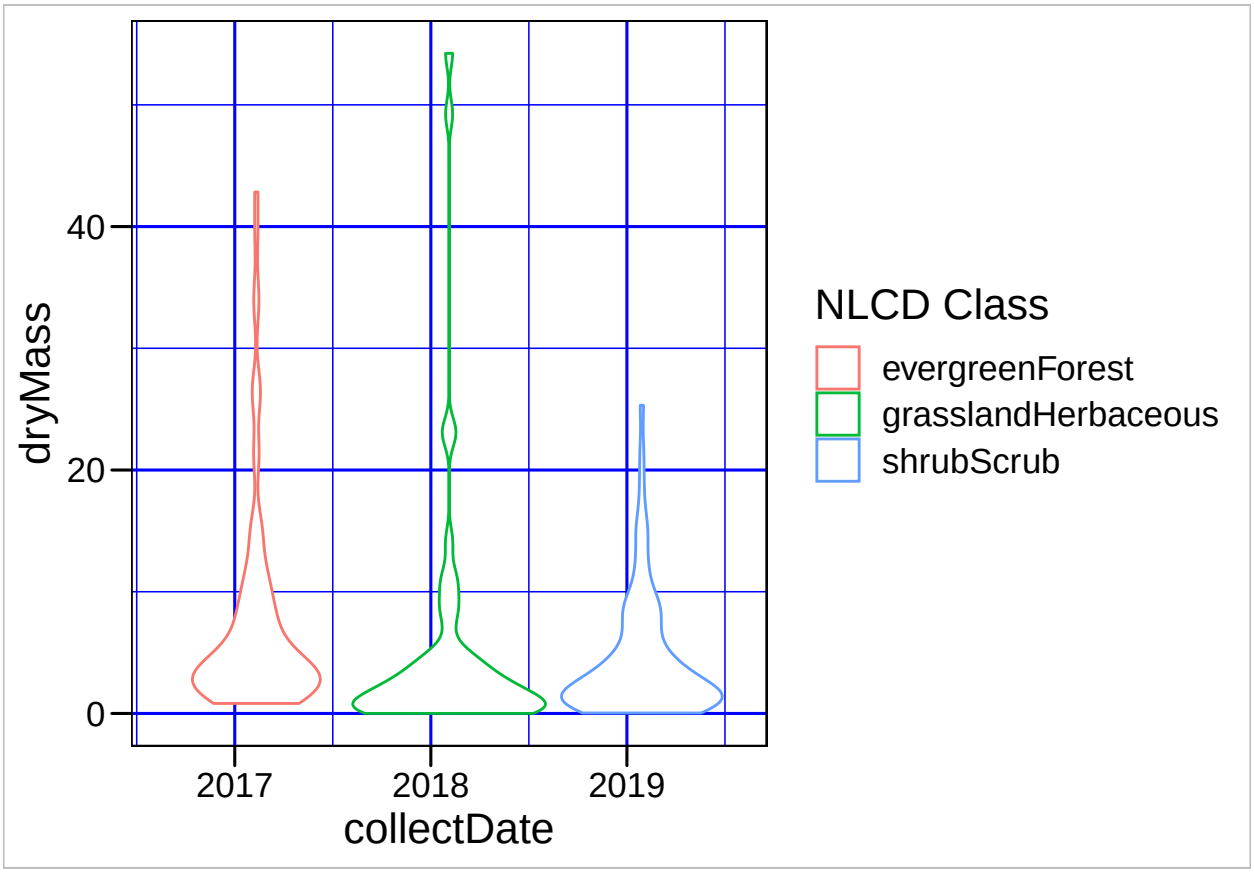
Question: What do you observe about the variables of interest over seasons and between lakes?

Answer: Summer was the ideal time to take nutrient samples, whereas there was no data for the other seasons. The nitrogen and temperatures are relativly the same across the two lakes, but there is a slight difference in phosphorous levels between the lakes.

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)

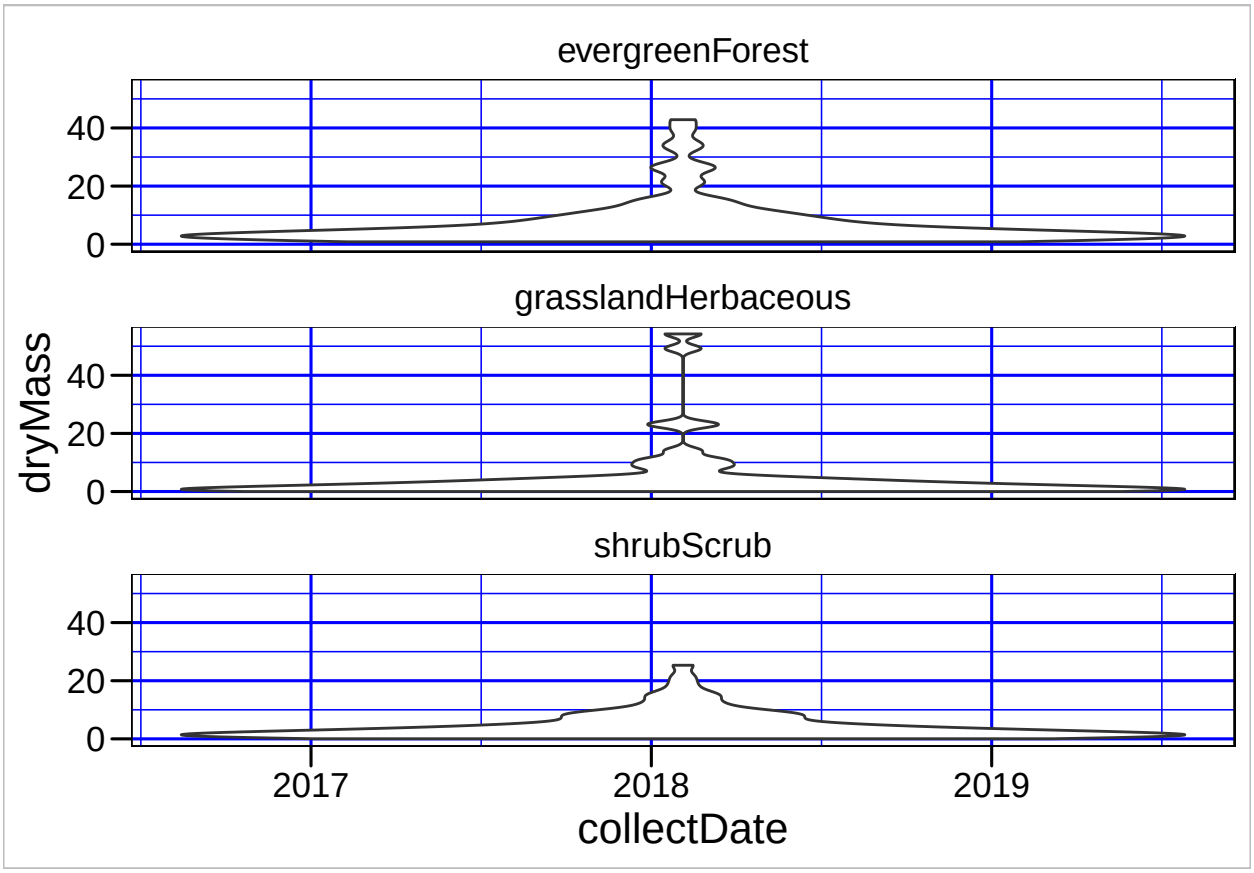
7. [Niwot Ridge] Now, plot the same plot but with NLCD classes separated into three facets rather than separated by color.

```
#6
# displaying only Needles subset
needles_plot <- ggplot(filter(PeterPaul.litter, functionalGroup=="Needles"),
                       # plotting dry mass of needles by date and coloring nlcdClass
                       aes(y=dryMass,
                           x=collectDate,
                           color=nlcdClass)) +
                          # choosing violin plot
                          geom_violin() +
                          # changing the name of the legend
                          labs(color = "NLCD Class")
print(needles_plot)
```




```
#7
# displaying only Needles subset
needles_plot_facet <- ggplot(filter(PeterPaul.litter, functionalGroup=="Needles"),
                             # plotting dry mass of needles by date and coloring nlcdType
                             aes(y=dryMass,
                                 x=collectDate,)) +
```

```
                        # choosing violin plot
                        geom_violin() +
                        # separate NLCD classes into 3 facets
                        facet_wrap(vars(nlcdClass), nrow = 3, )
print(needles_plot_facet)
```



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

Answer: It's easier to compare the NLCD Classes in 6, since they are next to each other on the same plot. Separated the graph gets squished and it's harder to see the differences between them.