Assignment 5: Data Visualization

Yangsen Zhang

Spring 2026

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, 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 Load required packages and read in datasets
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## 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
```

```
library(here)
```

```
## here() starts at /home/guest/EDE_Spring2026-yz
```

```
library(cowplot)
```

```
##
## Attaching package: 'cowplot'
##
## The following object is masked from 'package:lubridate':
##
##     stamp
```

```
library(lubridate)

here()
```

```
## [1] "/home/guest/EDE_Spring2026-yz"
```

```
PeterPaul.nutrients <- read_csv(
  here("Data","Processed_KEY",
       "NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv")
)
```

```
## Rows: 23008 Columns: 15
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr   (1): lakename
## dbl  (13): year4, daynum, month, depth, temperature_C, dissolvedOxygen, irra...
## date  (1): sampledate
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
Niwot.litter <- read_csv(
  here("Data","Processed_KEY",
       "NEON_NIWO_Litter_mass_trap_Processed.csv")
)
```

```
## Rows: 1692 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (7): plotID, trapID, functionalGroup, qaDryMass, nlcdClass, plotType, g...
## dbl  (5): dryMass, subplotID, decimalLatitude, decimalLongitude, elevation
## date (1): collectDate
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
#2 Ensure date columns are in Date format
str(PeterPaul.nutrients)
```

```
## spc_tbl_ [23,008 x 15] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ lakename       : chr [1:23008] "Paul Lake" "Paul Lake" "Paul Lake" "Paul Lake" ...
##  $ year4          : num [1:23008] 1984 1984 1984 1984 1984 ...
##  $ daynum         : num [1:23008] 148 148 148 148 148 148 148 148 148 148 ...
##  $ month          : num [1:23008] 5 5 5 5 5 5 5 5 5 5 ...
##  $ sampledate     : Date[1:23008], format: "1984-05-27" "1984-05-27" ...
##  $ depth          : num [1:23008] 0 0.25 0.5 0.75 1 1.5 2 3 4 5 ...
##  $ temperature_C  : num [1:23008] 14.5 NA NA NA 14.5 NA 14.2 11 7 6.1 ...
##  $ dissolvedOxygen: num [1:23008] 9.5 NA NA NA 8.8 NA 8.6 11.5 11.9 2.5 ...
##  $ irradianceWater: num [1:23008] 1750 1550 1150 975 870 610 420 220 100 34 ...
##  $ irradianceDeck : num [1:23008] 1620 1620 1620 1620 1620 1620 1620 1620 1620 1620 ...
##  $ tn_ug          : num [1:23008] NA NA NA NA NA NA NA NA NA NA ...
##  $ tp_ug          : num [1:23008] NA NA NA NA NA NA NA NA NA NA ...
##  $ nh34           : num [1:23008] NA NA NA NA NA NA NA NA NA NA ...
##  $ no23           : num [1:23008] NA NA NA NA NA NA NA NA NA NA ...
##  $ po4            : num [1:23008] NA NA NA NA NA NA NA NA NA NA ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   lakename = col_character(),
##   ..   year4 = col_double(),
##   ..   daynum = col_double(),
##   ..   month = col_double(),
##   ..   sampledate = col_date(format = ""),
##   ..   depth = col_double(),
##   ..   temperature_C = col_double(),
##   ..   dissolvedOxygen = col_double(),
##   ..   irradianceWater = col_double(),
##   ..   irradianceDeck = col_double(),
##   ..   tn_ug = col_double(),
##   ..   tp_ug = col_double(),
##   ..   nh34 = col_double(),
##   ..   no23 = col_double(),
##   ..   po4 = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
str(Niwot.litter)
```

```
## spc_tbl_ [1,692 x 13] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ plotID          : chr [1:1692] "NIWO_062" "NIWO_061" "NIWO_062" "NIWO_064" ...
##  $ trapID          : chr [1:1692] "NIWO_062_050" "NIWO_061_169" "NIWO_062_050" "NIWO_064_103" ...
##  $ collectDate     : Date[1:1692], format: "2016-06-16" "2016-06-16" ...
##  $ functionalGroup : chr [1:1692] "Seeds" "Other" "Woody material" "Seeds" ...
##  $ dryMass         : num [1:1692] 0 0.27 0.12 0 1.11 0 0 0 0.07 0.02 ...
##  $ qaDryMass       : chr [1:1692] "N" "N" "N" "N" ...
##  $ subplotID       : num [1:1692] 31 41 31 32 32 32 40 40 40 40 ...
##  $ decimalLatitude : num [1:1692] 40.1 40 40.1 40 40 ...
##  $ decimalLongitude: num [1:1692] -106 -106 -106 -106 -106 ...
##  $ elevation       : num [1:1692] 3477 3413 3477 3373 3446 ...
##  $ nlcdClass       : chr [1:1692] "shrubScrub" "evergreenForest" "shrubScrub" "evergreenForest" ...
```

```
##  $ plotType        : chr [1:1692] "tower" "tower" "tower" "tower" ...
##  $ geodeticDatum   : chr [1:1692] "WGS84" "WGS84" "WGS84" "WGS84" ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   plotID = col_character(),
##   ..   trapID = col_character(),
##   ..   collectDate = col_date(format = ""),
##   ..   functionalGroup = col_character(),
##   ..   dryMass = col_double(),
##   ..   qaDryMass = col_character(),
##   ..   subplotID = col_double(),
##   ..   decimalLatitude = col_double(),
##   ..   decimalLongitude = col_double(),
##   ..   elevation = col_double(),
##   ..   nlcdClass = col_character(),
##   ..   plotType = col_character(),
##   ..   geodeticDatum = col_character()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

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

```
#3 Build default theme
theme1 <- theme_minimal() +
  theme(plot.background = element_rect(fill = "white", color = NA),
        panel.grid.major = element_line(color = "grey"),
        panel.grid.minor = element_blank(),
        legend.position = "right",
        legend.title = element_text(face = "bold")
        )

theme_set(theme1)
```

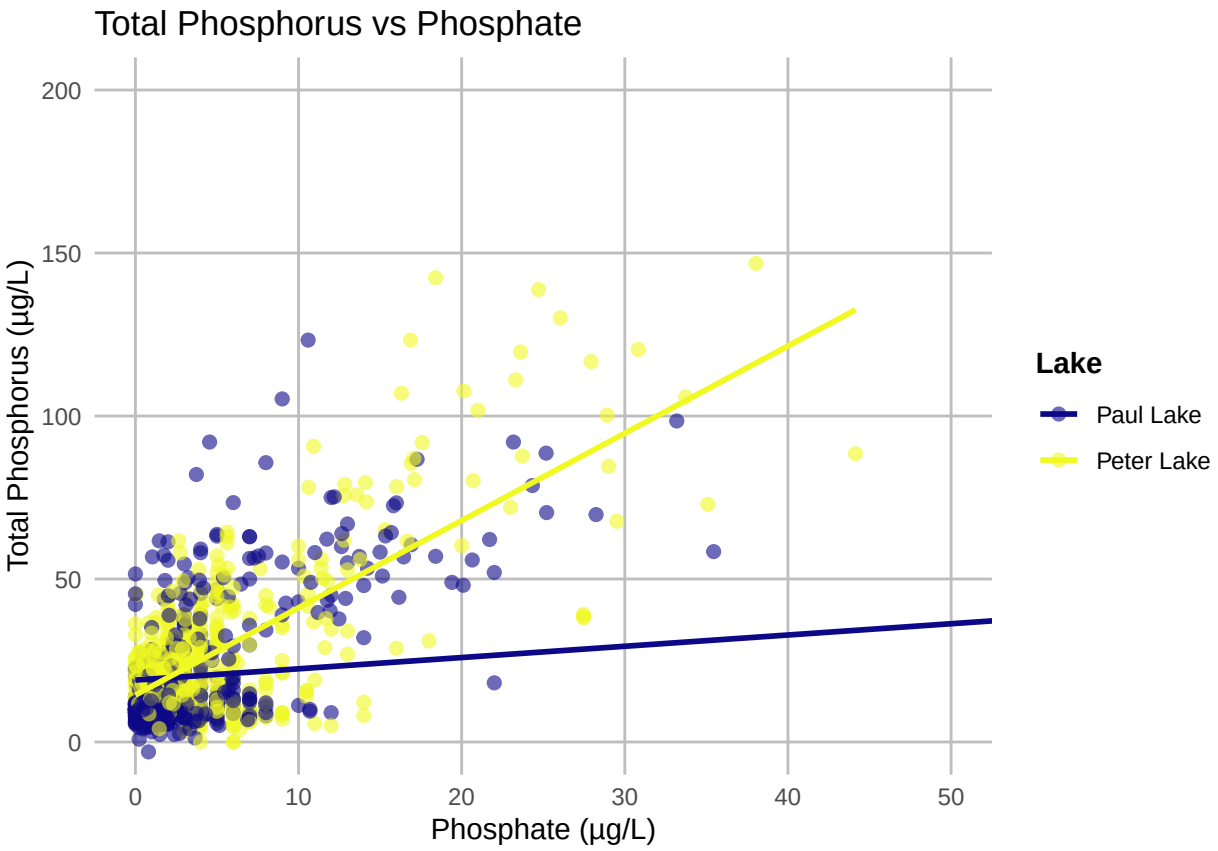
Create graphs

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 line(s) of best fit using the `lm` method. Adjust your axes to hide extreme values (hint: change the limits using `xlim()` and/or `ylim()`).

```r
#4
PeterPaul.nutrients %>%
  filter(!is.na(po4), !is.na(tp_ug)) %>%
  ggplot(aes(x = po4,
             y = tp_ug,
             color = lakename)) +
  geom_point(alpha = 0.6, size = 2) +
  geom_smooth(method = "lm", se = FALSE) +
  coord_cartesian(
    xlim = c(0, 50),
    ylim = c(0, 200)
  ) +
  labs(
    x = "Phosphate (µg/L)",
    y = "Total Phosphorus (µg/L)",
    color = "Lake",
    title = "Total Phosphorus vs Phosphate"
  ) +
  scale_color_viridis_d(option = "plasma")
```

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



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. Show all months, even those where no data was collected.

Tips: * Recall the discussion on factors in the lab section as it may be helpful here. * Setting an axis text in your theme to `element_blank()` removes the axis text (useful when multiple, aligned plots use the same axis values) * Setting a legend's position to "none" will remove the legend from a plot. * Individual plots can have different relative sizes when combined using `cowplot`.

```
#5
PeterPaul.nutrients <- PeterPaul.nutrients %>%
  mutate(month_f = factor(month, levels = 1:12,
                          labels = month.abb),
         lakename = as.factor(lakename))

fill_pal <- scale_fill_viridis_d(option = "plasma")

p_temp <- PeterPaul.nutrients %>%
  ggplot(aes(x = month_f, y = temperature_C, fill = lakename)) +
  geom_boxplot(outlier.alpha = 0.3) +
  scale_x_discrete(drop = FALSE) +
  labs(x = NULL, y = "Temperature (°C)", fill = "Lake",
       title = "Seasonality: Temperature / TP / TN") +
  fill_pal +
  theme(axis.text.x = element_blank())

p_tp <- PeterPaul.nutrients %>%
  ggplot(aes(x = month_f, y = tp_ug, fill = lakename)) +
  geom_boxplot(outlier.alpha = 0.3) +
  scale_x_discrete(drop = FALSE) +
  labs(x = NULL, y = "Total Phosphorus (µg/L)", fill = "Lake") +
  fill_pal +
  theme(axis.text.x = element_blank(),
        plot.title = element_blank())

p_tn <- PeterPaul.nutrients %>%
  ggplot(aes(x = month_f, y = tn_ug, fill = lakename)) +
  geom_boxplot(outlier.alpha = 0.3) +
  scale_x_discrete(drop = FALSE) +
  labs(x = "Month", y = "Total Nitrogen (µg/L)", fill = "Lake") +
  fill_pal +
  theme(plot.title = element_blank())

legend1 <- get_legend(p_temp + theme(legend.position = "right"))
```

```
## Warning: Removed 3566 rows containing non-finite outside the scale range
## (`stat_boxplot()`).
```

```
## Warning in get_plot_component(plot, "guide-box"): Multiple components found;
## returning the first one. To return all, use `return_all = TRUE`.
```

```
p_temp_noleg <- p_temp + theme(legend.position = "none")
p_tp_noleg   <- p_tp   + theme(legend.position = "none")
p_tn_noleg   <- p_tn   + theme(legend.position = "none")
```

```
panels <- plot_grid(
  p_temp_noleg, p_tp_noleg, p_tn_noleg,
  ncol = 1, align = "v", axis = "l",
  rel_heights = c(1, 1, 1)
)
```

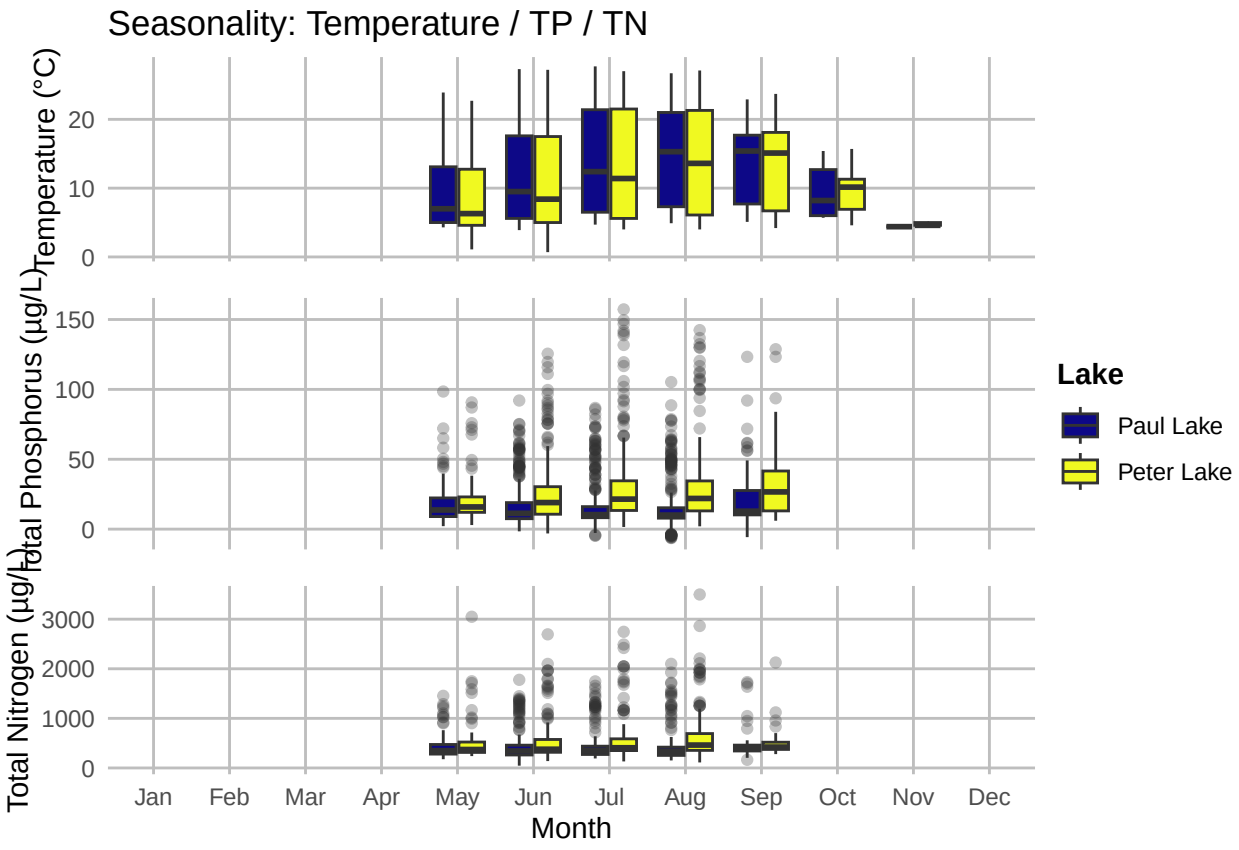
```
## Warning: Removed 3566 rows containing non-finite outside the scale range
## (‘stat_boxplot()‘).
```

```
## Warning: Removed 20729 rows containing non-finite outside the scale range
## (‘stat_boxplot()‘).
```

```
## Warning: Removed 21583 rows containing non-finite outside the scale range
## (‘stat_boxplot()‘).
```

```
combined_plot <- plot_grid(
  panels, legend1,
  ncol = 2,
  rel_widths = c(1, 0.18)
)

combined_plot
```



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

Answer: Temperature shows a strong seasonal trend with summer peaks and winter lows. TP and TN show higher variability and weaker seasonality, with more outliers in summer months. Peter Lake generally exhibits slightly higher nutrient concentrations than Paul Lake.

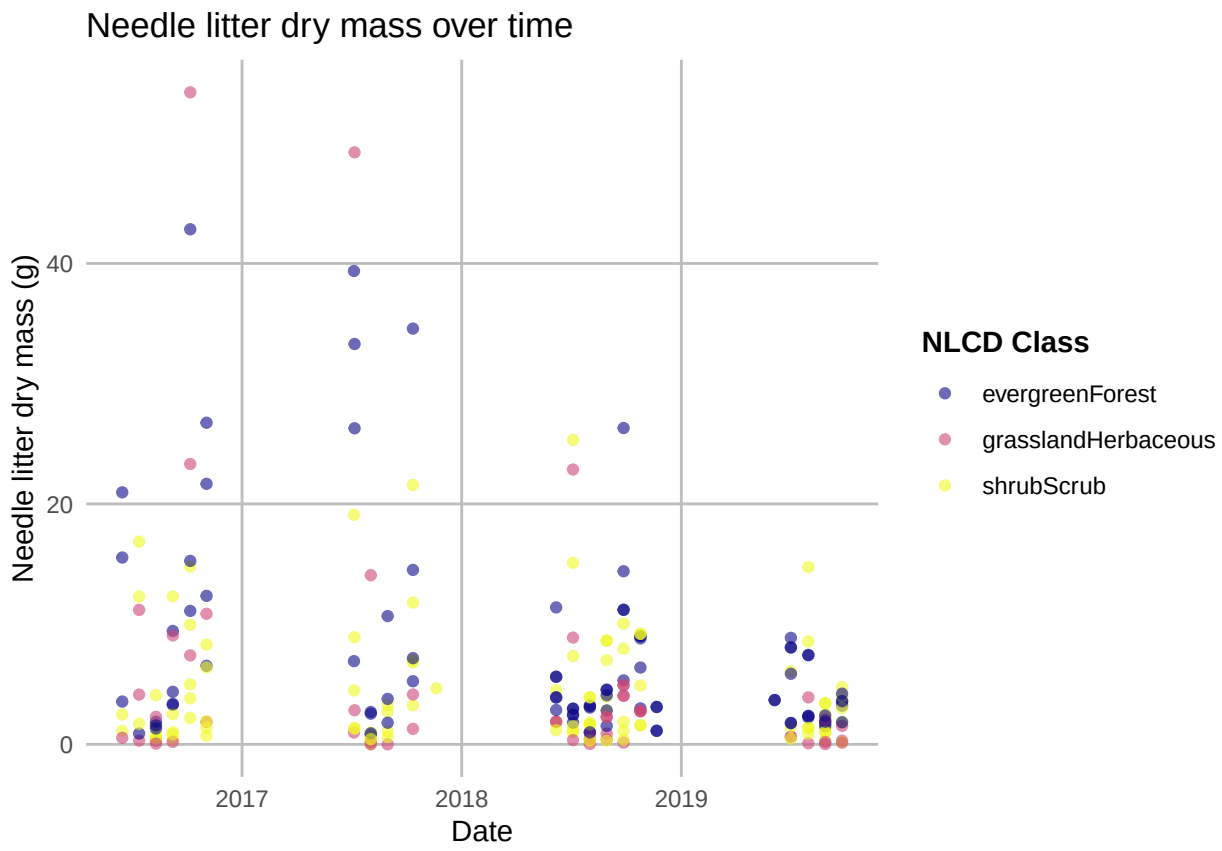
6. [Niwot Ridge] Plot a subset of the litter dataset by displaying only the "Needles" functional group. Plot the dry mass of needle litter by date and separate by NLCD class with a color aesthetic. (no need to adjust the name of each land use)

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

```
#6
needles_dat <- Niwot.litter %>%
  filter(functionalGroup == "Needles") %>%
  filter(!is.na(collectDate),
         !is.na(dryMass),
         !is.na(nlcdClass))

p6 <- ggplot(needles_dat,
             aes(x = collectDate,
                 y = dryMass,
                 color = nlcdClass)) +
  geom_point(alpha = 0.6) +
  labs(
    x = "Date",
    y = "Needle litter dry mass (g)",
    color = "NLCD Class",
    title = "Needle litter dry mass over time"
  ) +
  scale_color_viridis_d(option = "plasma")

p6
```

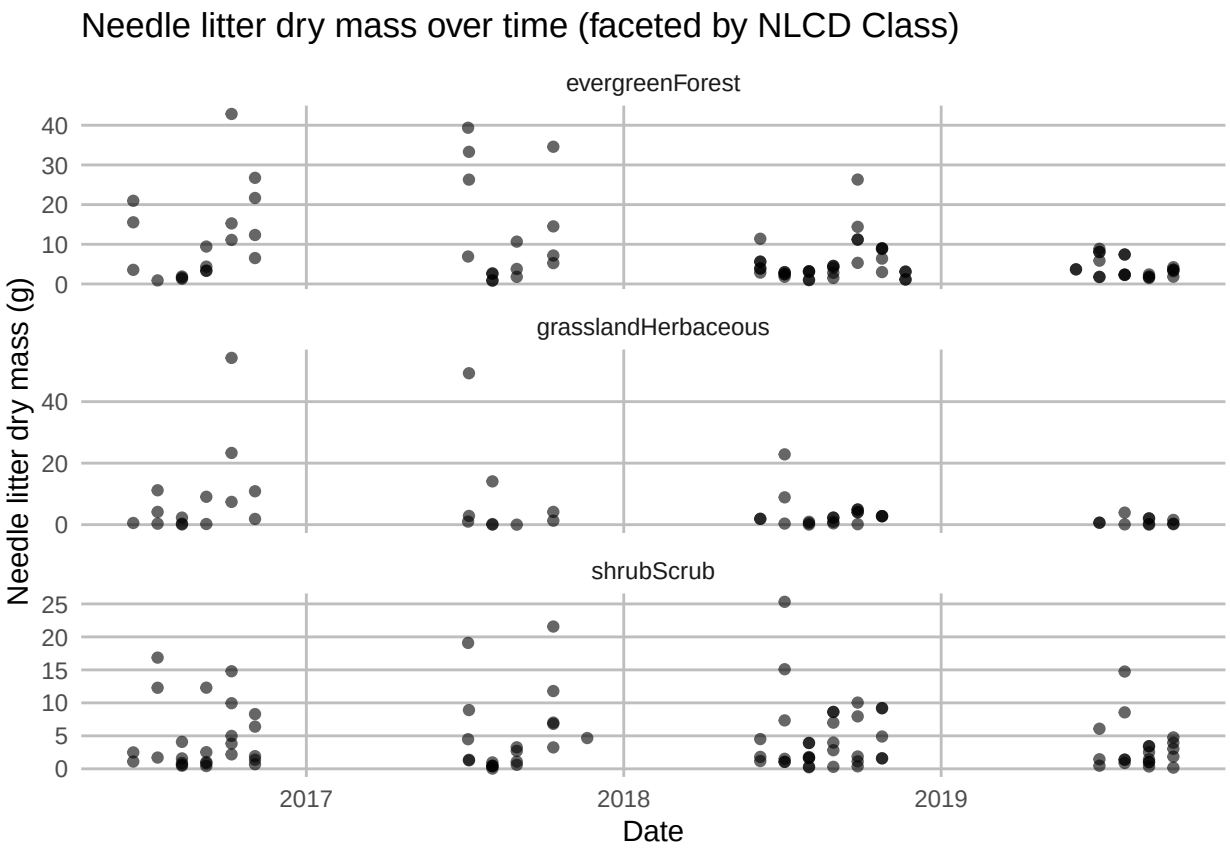



```
#7
classes_3 <- needles_dat %>%
  count(nlcdClass, sort = TRUE) %>%
  slice_head(n = 3) %>%
  pull(nlcdClass)

needles_3 <- needles_dat %>%
  filter(nlcdClass %in% classes_3)

p7 <- ggplot(needles_3,
             aes(x = collectDate,
                 y = dryMass)) +
  geom_point(alpha = 0.6) +
  facet_wrap(~ nlcdClass, ncol = 1, scales = "free_y") +
  labs(
    x = "Date",
    y = "Needle litter dry mass (g)",
    title = "Needle litter dry mass over time (faceted by NLCD Class)"
  )

p7
```



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

Answer: Plot 7 (the faceted version) is more effective than Plot 6 (the colored version). In Plot 6, all NLCD classes are displayed on the same axis using different colors. Although this allows direct comparison, overlapping observations reduce visual clarity. It is more difficult to clearly distinguish the trend of each land cover type, especially when the value overlaps. In contrast, Plot 7 divides each NLCD class into its own panel. This reduces visual overlap and makes the time trend in each class easier to explain. Variations and scale differences between land cover types are more obvious.