Assignment 5: Data Visualization

Sky Volz

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.

```r
#1
#install packages
library(here); library(tidyverse); library(cowplot)
```

```
## here() starts at /home/guest/EDE_Spring2026_personal
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.2.0     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   4.0.1     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
##
## Attaching package: 'cowplot'
##
##
## The following object is masked from 'package:lubridate':
##
##     stamp
```

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

```
## [1] "/home/guest/EDE_Spring2026_personal"
```

```
#reading in files
Peter.Paul.chem.nutrients <-
  read.csv(
    here("Data/Processed_KEY/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv"),
    stringsAsFactors = TRUE)

Litter.processed <-
 read.csv(
    here("Data/Processed_KEY/NEON_NIWO_Litter_mass_trap_Processed.csv"),
    stringsAsFactors = TRUE)

#2
#checking dates
class(Litter.processed$collectDate)
```

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

```
class(Peter.Paul.chem.nutrients$sampledate)
```

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

```
#changing to dates using lubridate
Peter.Paul.chem.nutrients$sampledate <- ymd(Peter.Paul.chem.nutrients$sampledate)
Litter.processed$collectDate <- ymd(Litter.processed$collectDate)
```

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
#install more themes and colors
library(ggthemes); library(viridis); library(RColorBrewer); library(colormap)
```

```
##
## Attaching package: 'ggthemes'

## The following object is masked from 'package:cowplot':
##
##     theme_map

## Loading required package: viridisLite
```

```r
#writing my theme, based on theme_economist

my_theme <- theme_economist_white()  +
  theme(
     plot.title = element_text(
       size = 14, color = "navy", face = "bold", family = "serif"),
     axis.title = element_text(
       size = 12, color = "navy"),
     legend.position = "right",
     legend.background = element_rect(linewidth = 10)
  )
```

Create graphs

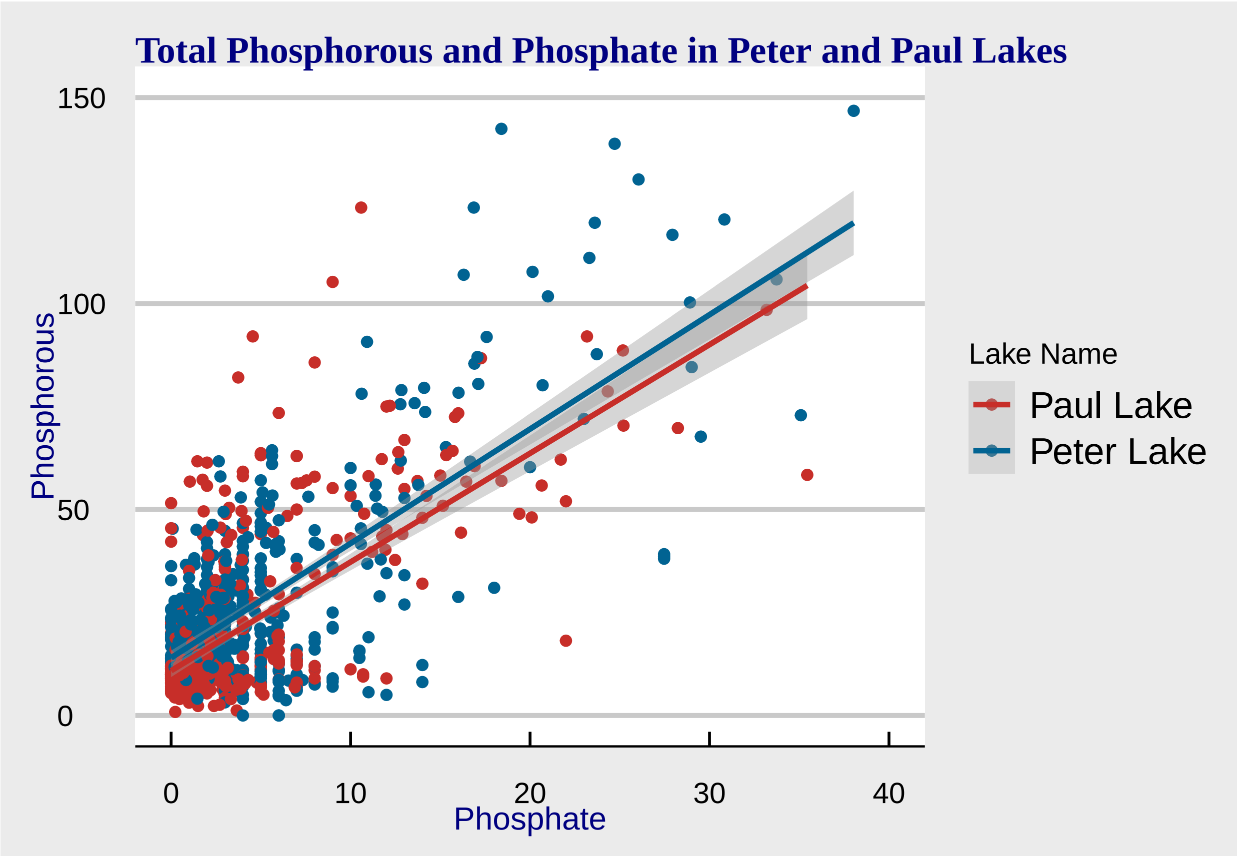
For numbers 4-7, create ggplot graphs and adjust aesthetics to follow best practices for data visualization. Ensure your theme, color palettes, axes, and additional aesthetics are edited accordingly.

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
po4.tp_ug <- ggplot(Peter.Paul.chem.nutrients, aes(
  x=po4, y=tp_ug, color = lakename)) +
  geom_point() +
  my_theme +
  geom_smooth (method = "lm") +
  ylim(0, 150) + xlim(0, 40) +
  labs(title = "Total Phosphorous and Phosphate in Peter and Paul Lakes",
       x = "Phosphate", y = "Phosphorous", color = "Lake Name") +
  scale_color_wsj()

print(po4.tp_ug)
```

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

## Warning: Removed 21949 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

```
## Warning: Removed 21949 rows containing missing values or values outside the scale range
## (`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. 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
#temperature graph
Temp <- Peter.Paul.chem.nutrients %>%
  mutate(month_f = factor(
    month,                      #make it tidy - organize code
    level=1:12,
    labels=month.abb)) %>%
  ggplot(aes(x=month_f, y = temperature_C, color = lakename)) +
  geom_boxplot() +
  labs(x = NULL, y = "Temperature (C)", color = "Lake Name") +
  scale_x_discrete(drop=F) +   #fix scale by not dropping months with empty values
  my_theme +  scale_color_wsj() +
```

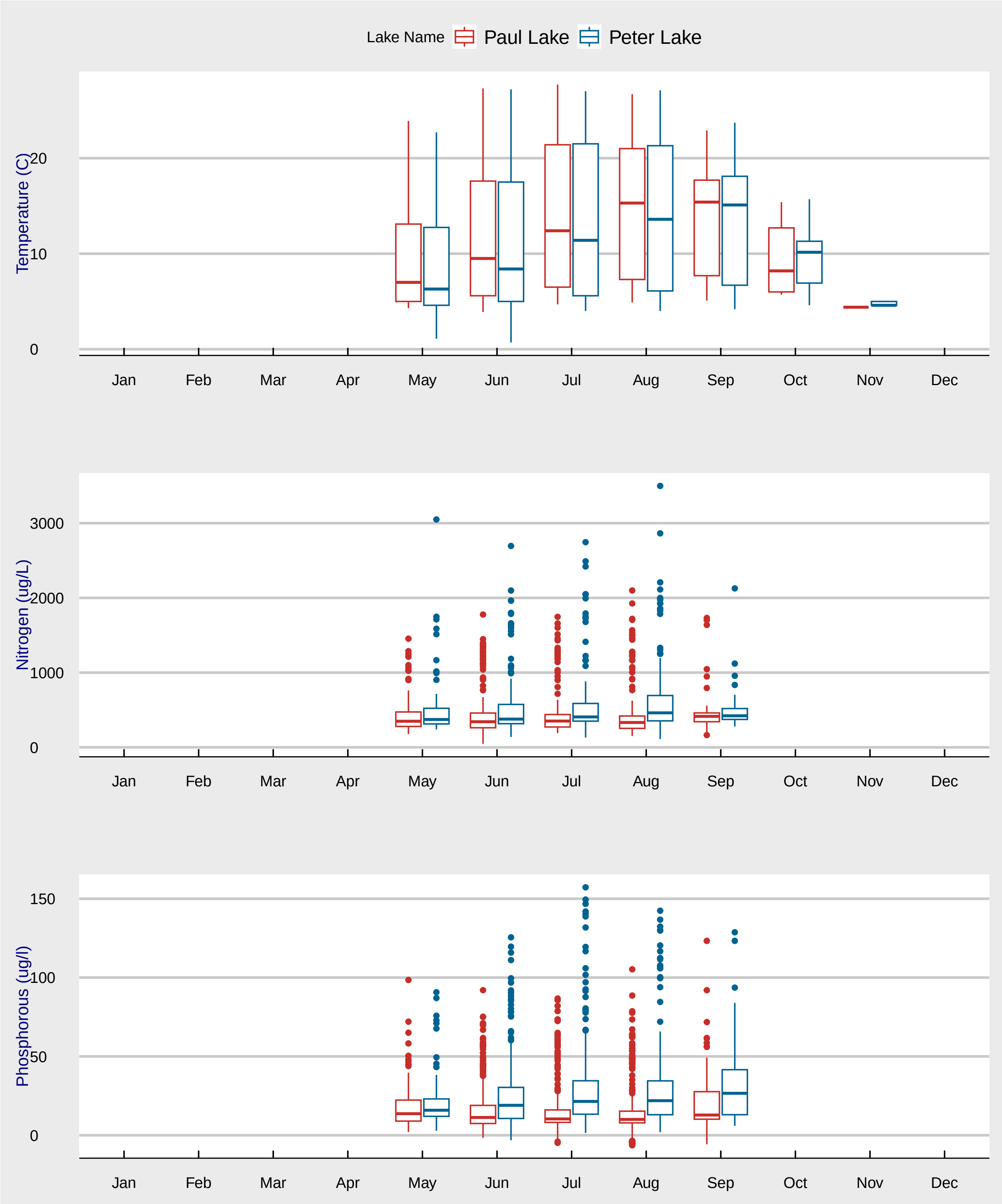
```r
  theme(legend.position = "top")

#phosphorous graph
TP <- Peter.Paul.chem.nutrients %>%
  mutate(month_f = factor(
    month,                        #make it tidy - organize code
    level=1:12,
    labels=month.abb)) %>%
  ggplot(aes(x=month_f, y = tp_ug, color = lakename)) +
  geom_boxplot() +
  labs(x = NULL, y = "Phosphorous (ug/l)") +
  scale_x_discrete(drop=F) +   #fix scale by not dropping months with empty values
  my_theme +  scale_color_wsj() +
  theme(legend.position = "none")

#nitrogren graph
TN <- Peter.Paul.chem.nutrients %>%
  mutate(month_f = factor(
    month,                        #make it tidy - organize code
    level=1:12,
    labels=month.abb)) %>%
  ggplot(aes(x=month_f, y = tn_ug, color = lakename)) +
  geom_boxplot() +
  labs(x = NULL, y = "Nitrogen (ug/L)") +
  scale_x_discrete(drop=F) +   #fix scale by not dropping months with empty values
  my_theme +  scale_color_wsj() +
  theme(legend.position = "none")

lake.boxplots <- plot_grid(Temp, TN, TP, nrow =  3, align="hv")
```

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

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

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

```r
print(lake.boxplots)
```



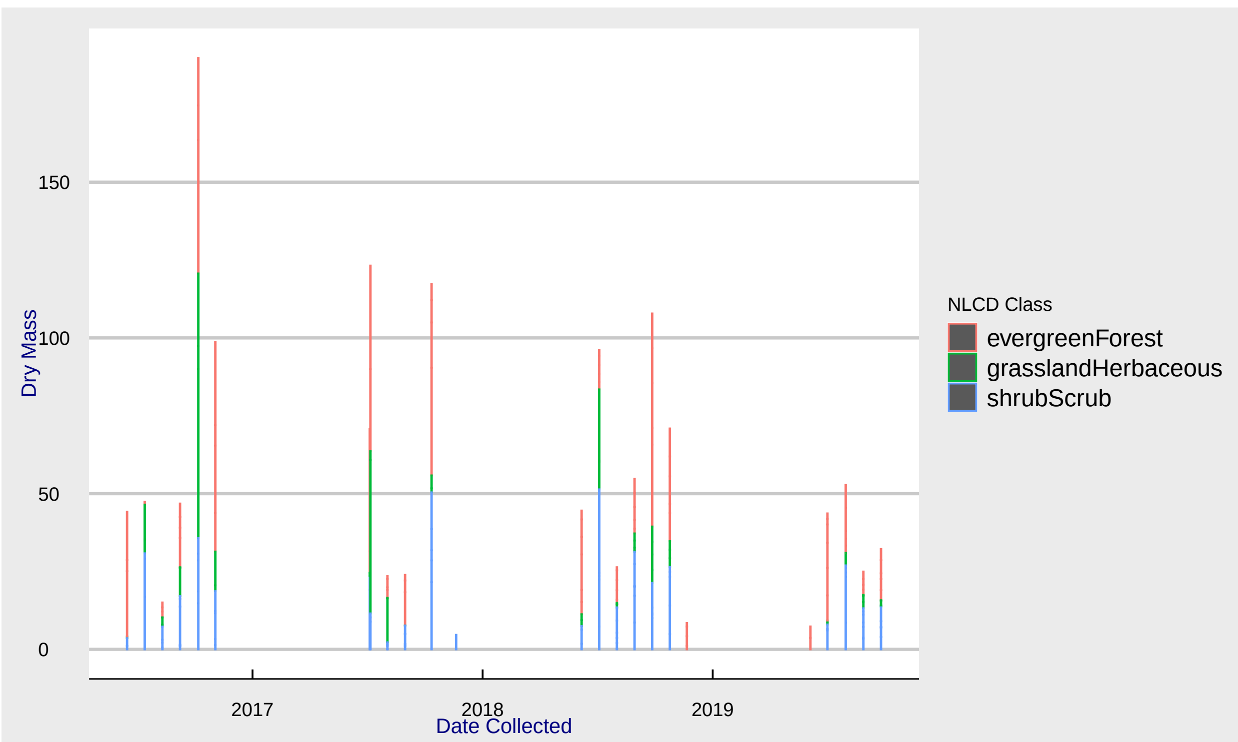
Question: What do you observe about the variables of interest over seasons and between lakes?

> Answer: The temperature is highest during the summer, which is when the nitrogen and phosphorous levels peak as well. The hottest months tend to have the highest nutrient levels, although it varies by lake. When the weather begins to cool as it turns into fall, the nutrient levels decline. Further, although Paul Lake has higher mean temperatures, Peter Lake tend to have higher mean

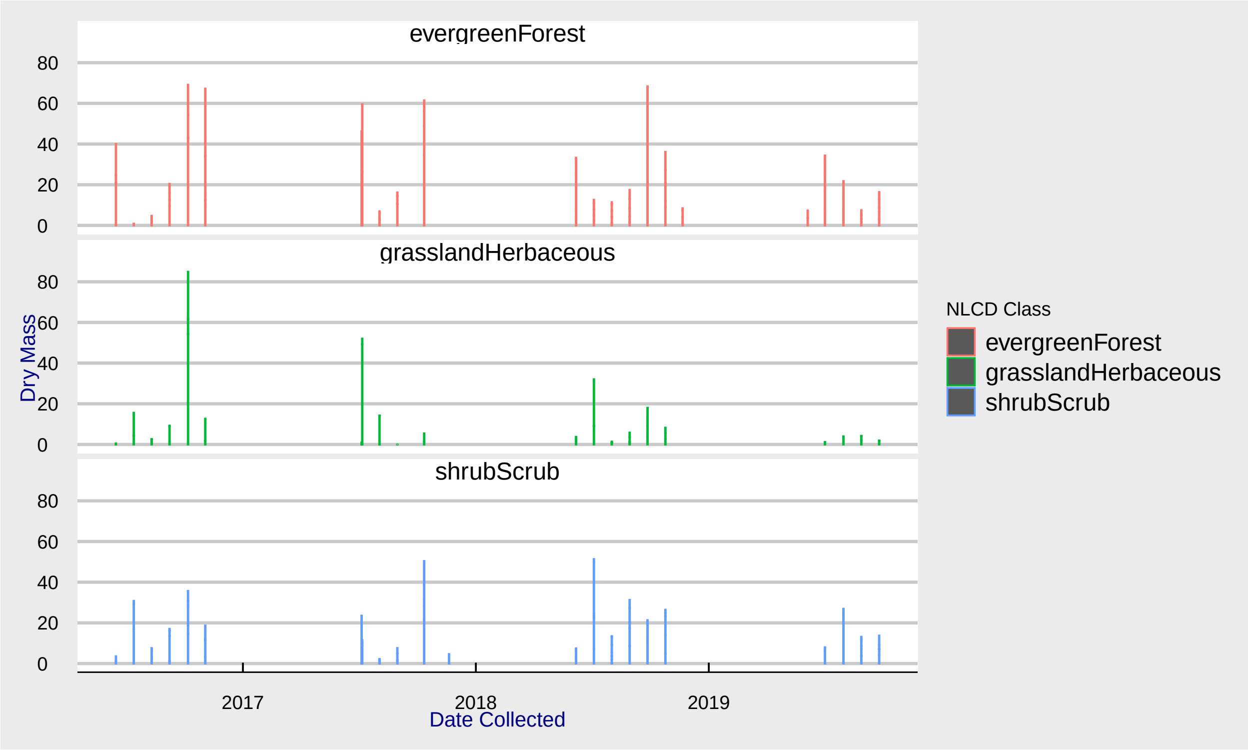
nutrient levels, especially for phosphorous.

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
color.nlcd.class.plot <- Litter.processed %>%
  filter(functionalGroup == "Needles") %>%
  ggplot(aes(x = collectDate, y = dryMass, color = nlcdClass)) +
  geom_col() +
  my_theme +
  labs(x = "Date Collected", y = "Dry Mass", color = "NLCD Class")

print(color.nlcd.class.plot)
```




```
#7
facet.nlcd.class.plot <- Litter.processed %>%
  filter(functionalGroup == "Needles") %>%
  ggplot(aes(x = collectDate, y = dryMass, color = nlcdClass)) +
  geom_col() +
  my_theme +
  labs(x = "Date Collected", y = "Dry Mass", color = "NLCD Class") +
  facet_wrap(vars(nlcdClass), nrow = 3)

print(facet.nlcd.class.plot)
```



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

Answer: I think the plot 7 is more effective. Showing each NCLD class in it's own panel makes it easier to see trends and seasonal changes without all the points overlapping. The long time period makes plot 6 especially hard to see, because the columns end up being very skinny so it is difficult to see the transition in color. Plot 6 is okay for comparing values directly, but it gets pretty crowded and harder to follow the patterns over time. Plot 7 also allows for a comparison of specific time periods because the axes are aligned vertically.