Assignment 5: Data Visualization

Jiaqi Li

OVERVIEW

This exercise accompanies the lessons in Environmental Data Analytics (ENV872L) on data wrangling.

Directions

1. Change "Student Name" on line 3 (above) with your name.
2. Use the lesson as a guide. It contains code that can be modified to complete the assignment.
3. Work through the steps, **creating code and output** that fulfill each instruction.
4. Be sure to **answer the questions** in this assignment document. Space for your answers is provided in this document and is indicated by the ">" character. If you need a second paragraph be sure to start the first line with ">". You should notice that the answer is highlighted in green by RStudio.
5. When you have completed the assignment, **Knit** the text and code into a single PDF file. You will need to have the correct software installed to do this (see Software Installation Guide) Press the `Knit` button in the RStudio scripting panel. This will save the PDF output in your Assignments folder.
6. After Knitting, please submit the completed exercise (PDF file) to the dropbox in Sakai. Please add your last name into the file name (e.g., "Salk_A04_DataWrangling.pdf") prior to submission.

The completed exercise is due on Tuesday, 19 February, 2019 before class begins.

Set up your session

1. Set up your session. Upload the NTL-LTER processed data files for chemistry/physics for Peter and Paul Lakes (tidy and gathered), the USGS stream gauge dataset, and the EPA Ecotox dataset for Neonicotinoids.

2. Make sure R is reading dates as date format, not something else (hint: remember that dates were an issue for the USGS gauge data).

```r
#1 Set up
setwd("/Users/ljq/Desktop/Blue Devils/Data Analysis/ENV872_02")
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------- tidyverse 1.2.1 --

## v ggplot2 3.1.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.7
## v tidyr   0.8.1     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0

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

```r
PeterPaul <- read.csv("./Data/Processed/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv")
stream <- read.csv("./Data/Raw/USGS_Site02085000_Flow_Raw.csv")
tox <- read.csv("./Data/Raw/ECOTOX_Neonicotinoids_Mortality_raw.csv")
PeterPaul.gathered <-
  read.csv("./Data/Processed/NTL-LTER_Lake_Nutrients_PeterPaulGathered_Processed.csv")
#2 Format date
```

```
PeterPaul$sampledate <- as.Date(PeterPaul$sampledate, format = "%m/%d/%y")
stream$datetime <- as.Date(stream$datetime, format = "%m/%d/%y")
stream$datetime <- format(stream$datetime, "%y%m%d")
create.early.dates <- (function(d) {
       paste0(ifelse(d > 181231,"19","20"),d)
       })
stream$datetime <- create.early.dates(stream$datetime)
stream$datetime <- as.Date(stream$datetime, format = "%Y%m%d")
colnames(stream) <- c("agency_cd", "site_no", "datetime",
                              "discharge.max", "discharge.max.approval",
                              "discharge.min", "discharge.min.approval",
                              "discharge.mean", "discharge.mean.approval",
                              "gage.height.max", "gage.height.max.approval",
                              "gage.height.min", "gage.height.min.approval",
                              "gage.height.mean", "gage.height.mean.approval")
PeterPaul.gathered$sampledate <- as.Date(PeterPaul.gathered$sampledate, format = '%Y-%m-%d')
```

Define your theme

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

```
#3
top.theme <- theme_classic(base_size = 12) +
  theme(axis.text = element_text(color = "black"),
        legend.position = "top")
```

Create graphs

For numbers 4-7, create graphs that follow best practices for data visualization. To make your graphs "pretty," ensure your theme, color palettes, axes, and legends are edited to your liking.

Hint: a good way to build graphs is to make them ugly first and then create more code to make them pretty.

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.

```
#4 Total phosphorus level in Peter and Paul lakes
P.plot <- ggplot(PeterPaul, aes(x = po4, y = tp_ug)) +
  geom_point(aes(color = lakename, shape = lakename)) +
  xlim(0, 50) +
  xlab(expression(paste("Phosphate (", mu, "g/L)", sep = ""))) +
  ylab(expression(paste("Total phosphorus (", mu, "g/L)", sep = ""))) +
  geom_smooth(method = lm, color = 'black') +
  scale_color_manual(values=c("#E69F00", "#56B4E9"), name="Lake Name") +
  scale_shape_discrete(name="Lake Name") +
  top.theme

print(P.plot)
```

5. [NTL-LTER] Plot nutrients by date for Peter Lake, with separate colors for each depth. Facet your graph by the nutrient type.

```
#5 Nutrient facet
```

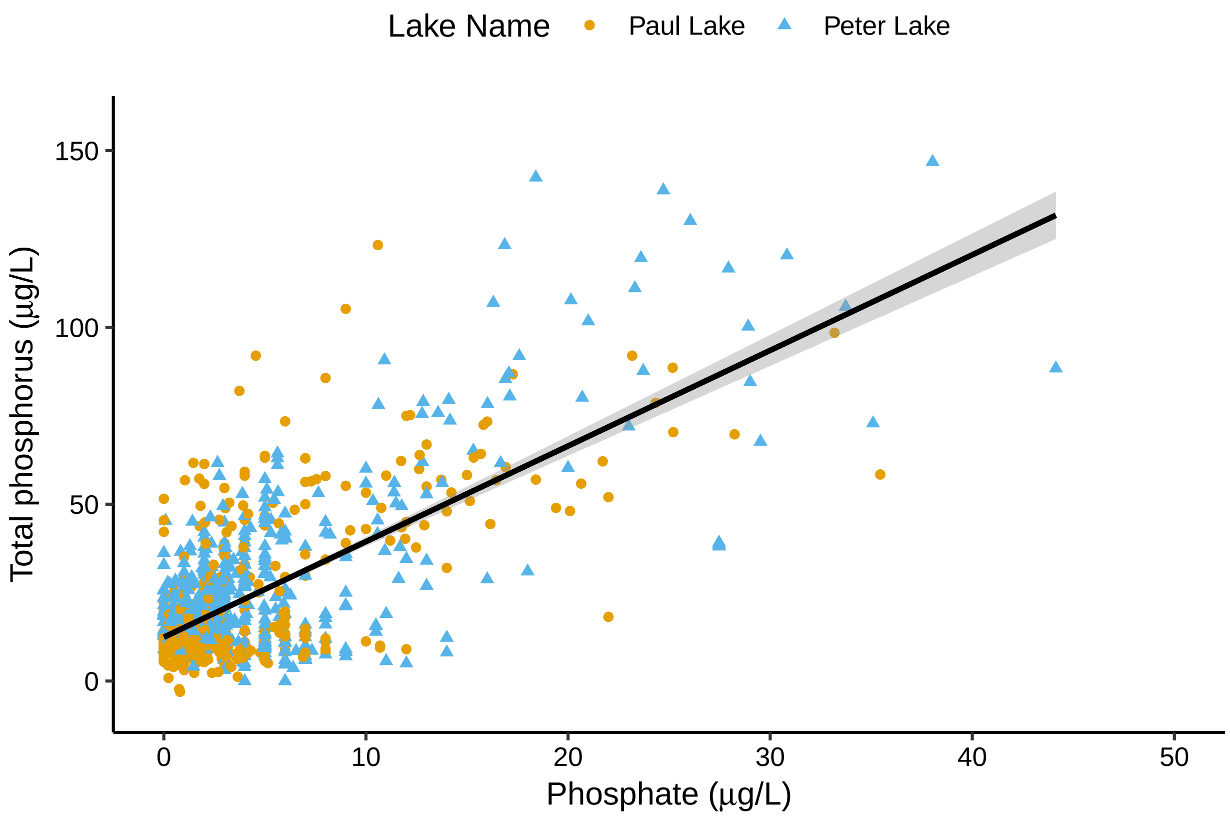

Figure 1: Total phosphorus level in Peter Lake and Paul Lake

```
library(viridis)
theme.facet <-
  theme_light() +
  theme(strip.background = element_rect(fill = "white"), strip.text =
          element_text(color = "black")) +
  theme(axis.text.x = element_text(angle = 90,  hjust = 1))

nutrient <- c("nh34" = "Ammonia", "no23" = "Nitrite", "po4" = "Phosphate",
              "tn_ug" = "Total N", "tp_ug" = "Total P")

nutrient.facet <-
  ggplot(subset(PeterPaul.gathered, lakename == 'Peter Lake'),
         aes(x = sampledate, y = concentration, color = depth)) +
  geom_point(size = 1) +
  facet_wrap(vars(nutrient), nrow = 5, scales="free_y", strip.position="right",
             labeller = as_labeller(nutrient)) +
  scale_x_date(limits = as.Date(c("1991-01-01", "2016-12-31")),
    date_breaks = "12 months", date_labels = "%b %y") +
  xlab("Sample Date") +
  ylab(expression(paste("Concentration (", mu, "g/L)", separate = ""))) +
  scale_color_viridis(name="Depth (m)") +
  theme.facet

print(nutrient.facet)
```

6. [USGS gauge] Plot discharge by date. Create two plots, one with the points connected with geom_line and one with the points connected with geom_smooth (hint: do not use method = "lm"). Place these graphs on the same plot (hint: ggarrange or something similar)

```
#6 Discharge plot
stream.theme <- theme_classic() +
  theme(axis.text.x = element_text(angle = 45,  hjust = 1))

stream.plot <- ggplot(stream, aes(x = datetime, y = discharge.mean)) +
  geom_point(size = 1) +
  geom_line() +
  scale_x_date(limits = as.Date(c("2004-07-01", "2018-12-31")),
date_breaks = "6 months", date_labels = "%b %y") +
  xlab("Date") +
  ylab(bquote('Discharge' ~ (ft^3/s))) +
  stream.theme

stream.plot.2 <- ggplot(stream, aes(x = datetime, y = discharge.mean)) +
  geom_point(size = 1) +
  geom_smooth() +
  scale_x_date(limits = as.Date(c("2004-07-01", "2018-12-31")),
date_breaks = "6 months", date_labels = "%b %y") +
  xlab("Date") +
  ylab(bquote('Discharge' ~ (ft^3/s))) +
  stream.theme

library(gridExtra)
grid.arrange(stream.plot, stream.plot.2)
```

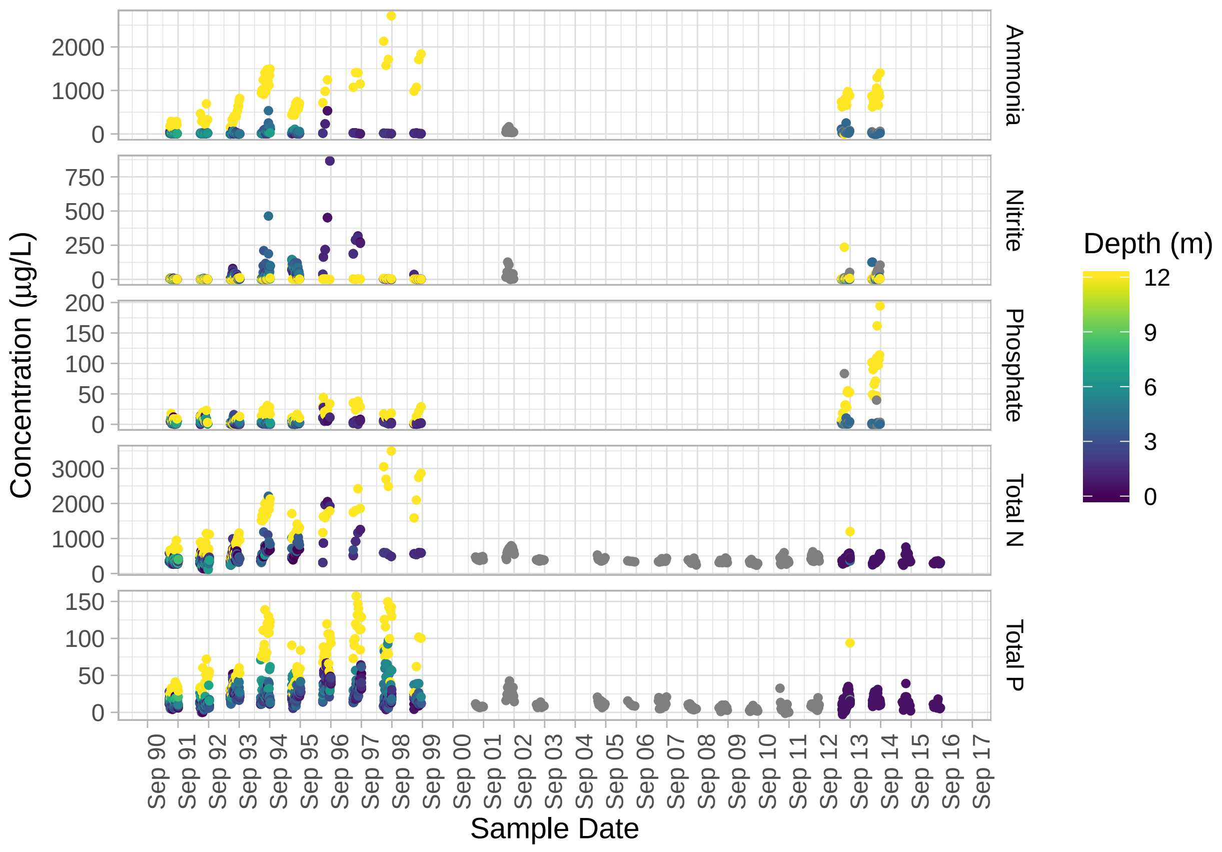

Figure 2: Nutrients for Peter Lake

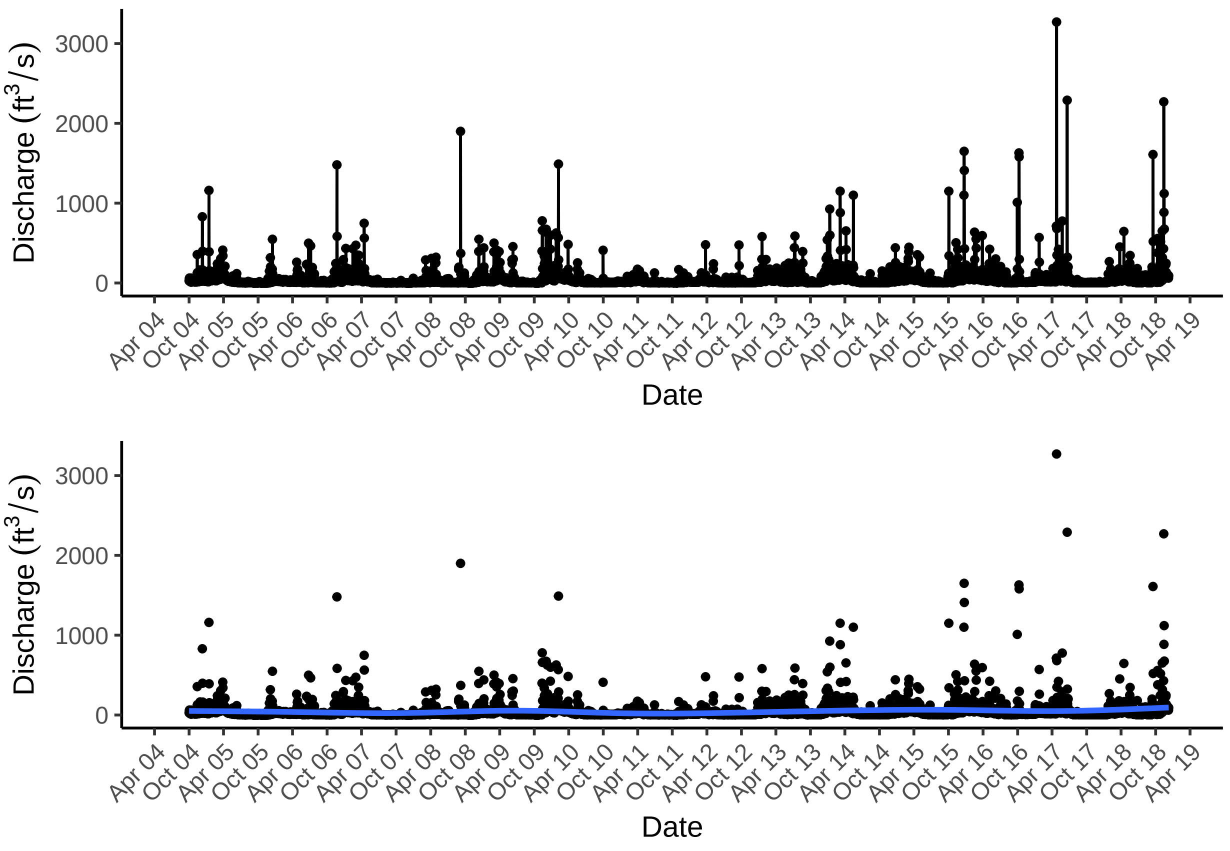

Figure 3: Discharge for Eno River

Question: How do these two types of lines affect your interpretation of the data?

> Answer: geom_line function connects all the observation points and shows the pattern of discharge variation. However, it is not very useful in this case since there are so many observations and it is hard to see the variation. On the other hand, geom_smooth function shows an overall relationship between date and discharge rate. In this data set, the line can tell us the discharge rate does not change a lot and is considerably stable over time.

7. [ECOTOX Neonicotinoids] Plot the concentration, divided by chemical name. Choose a geom that accurately portrays the distribution of data points.

```r
#7 Neonicotinoids

tox.theme <- theme_light() +
  theme(axis.title.x=element_blank(), axis.text.x=element_blank(),
        axis.ticks.x=element_blank())

tox.plot <- ggplot(subset(tox, Conc..Units..Std. == "AI mg/L"),
                   aes(x = Chemical.Name, y = Conc..Mean..Std.)) +
  geom_boxplot(aes(color =Chemical.Name)) +
  xlab(element_blank()) +
  ylab("Concentration (mg/L)") +
  scale_color_brewer(palette = "YlGnBu", name="Chemical Name") +
  tox.theme

print(tox.plot)
```

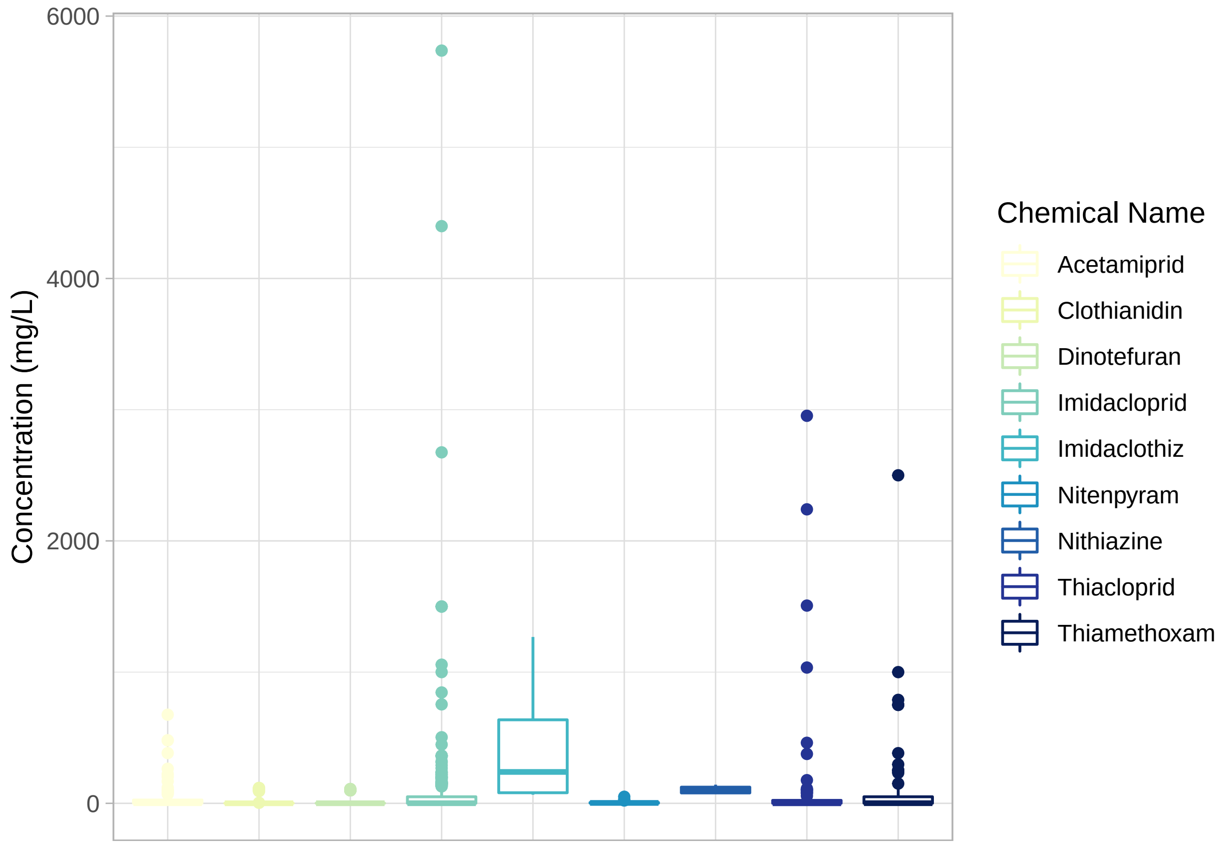

Figure 4: Neonicotinoids concentrations by chemical