Assignment 5: Data Visualization

Sebastian Bognar

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).

```
#1 get the working directory
getwd()
```

```
## [1] "/Users/Seabass/Documents/Duke/spring_2019/env_872L/lesson_2/ENV_872L/Assignments"
```

```
setwd("/Users/Seabass/Documents/Duke/spring_2019/env_872L/lesson_2/ENV_872L")

# load tidyverse package
install.packages("ggpubr",repos = "http://cran.us.r-project.org")
```

```
##
## The downloaded binary packages are in
##  /var/folders/2b/99qrr_4x56d__vh5gtz9dyhm0000gn/T//Rtmpd3ebto/downloaded_packages
```

```
library(ggpubr)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: magrittr
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------------------- tidyverse 1.2.1 --
## v tibble  2.0.1     v purrr   0.2.5
## v tidyr   0.8.2     v dplyr   0.7.8
## v readr   1.3.1     v stringr 1.3.1
## v tibble  2.0.1     v forcats 0.3.0
## -- Conflicts ------------------------------------------------------ tidyverse_conflicts() --
## x tidyr::extract()   masks magrittr::extract()
## x dplyr::filter()    masks stats::filter()
## x dplyr::lag()       masks stats::lag()
## x purrr::set_names() masks magrittr::set_names()
```

```
# upload the chemistry/physics for peter and paul lakes

peterpaul_physics <- read.csv("./Data/Processed/NTL-LTER_Lake_ChemistryPhysics_PeterPaul_Processed.csv"

peterpaul_chem <- read.csv("./Data/Processed/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Processed.csv"

peter.paul.gathered  <- read.csv("./Data/Processed/NTL-LTER_Lake_Nutrients_PeterPaulGathered_Processed.

#upload usgs dataset

USGS_flow_raw <- read.csv("./Data/Raw/USGS_Site02085000_Flow_Raw.csv", header = TRUE)

# upload the ecotox Dataset

EPA_ECOTOX <- read.csv("./Data/Raw/ECOTOX_Neonicotinoids_Mortality_raw.csv", header = TRUE)

#2
# check and format date column for peter paul lakes

peterpaul_chem$sampledate<- as.Date(peterpaul_chem$sampledate, "%Y-%m-%d")

peterpaul_physics$sampledate<- as.Date(peterpaul_physics$sampledate,"%d/%m/%y" )

peter.paul.gathered$sampledate<- as.Date(peter.paul.gathered$sampledate,"%Y-%m-%d" )


# check and format USGS date

USGS_flow_raw$datetime <- as.Date(USGS_flow_raw$datetime, "%m/%d/%y")

# format dates into single string dates
USGS_flow_raw$datetime <- format(USGS_flow_raw$datetime, "%y%m%d")

#  function to attach full year
create.early.dates <- (function(d) {
      paste0(ifelse(d > 181231,"19","20"),d)
      })
# use function to attach full year
USGS_flow_raw$datetime <- create.early.dates(USGS_flow_raw$datetime)

# reformat the date with slashes
```

```
#
USGS_flow_raw$datetime <- as.Date(USGS_flow_raw$datetime, format = "%Y%m%d")
```

Define your theme

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

```
library(ggplot2)
#3

theme_sb <- theme_gray(base_size = 12) +
  theme(axis.text = element_text(color = "black"),
        legend.position = "right",
        axis.ticks = element_line(colour = "black"),
        panel.border = element_rect(fill= NA,color="black", size=0.5,
                                    linetype="solid"),
        panel.grid.major = element_blank())

theme_dark <- theme_dark(base_size = 12) +
  theme(axis.text = element_text(color = "black"),
        legend.position = "right",
        axis.ticks = element_line(colour = "black"),
        panel.border = element_rect(fill= NA,color="black", size=0.5,
                                    linetype="solid"),
        panel.grid.major = element_blank())
```

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.

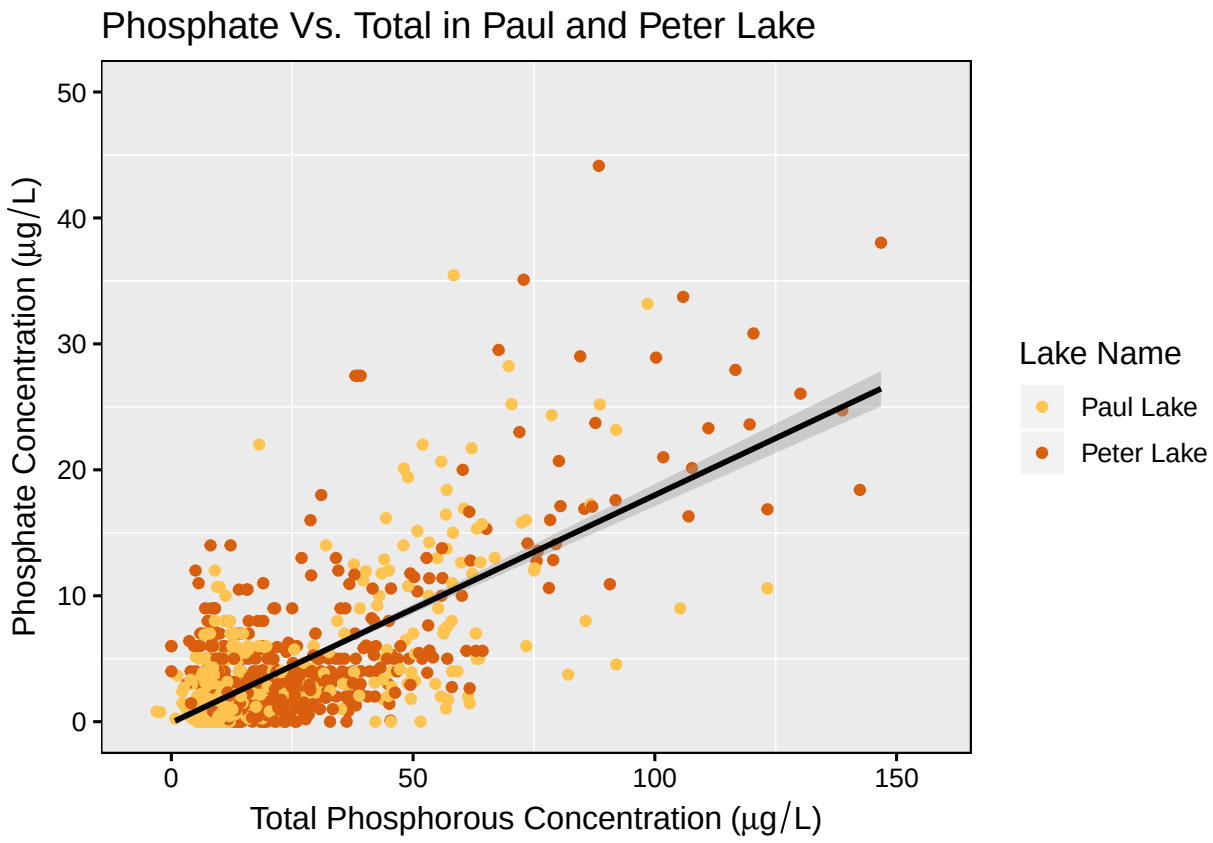
```
#load Color
library(viridis)
```

```
## Loading required package: viridisLite
```

```
library(RColorBrewer)
library(colormap)


# create color palette
Palette <- c("#fec44f","#d95f0e")

#4
Peter_paul_TP.by.P <- ggplot(peterpaul_chem)+
  geom_point(aes(x= tp_ug, y= po4, col = lakename))+
  scale_y_continuous(limits =c(0, 50))+
   scale_colour_manual(values=Palette)+
  xlab(expression("Total Phosphorous Concentration"~"("*mu*g/L*")"))+
```

```
  ylab(expression("Phosphate Concentration"~("*mu*g/L*")"))+
  labs(color='Lake Name')+
  geom_smooth(aes(x= tp_ug, y= po4 ),method = "lm", color ="black")+
  ggtitle("Phosphate Vs. Total in Paul and Peter Lake")+
  theme_sb



print(Peter_paul_TP.by.P)
```

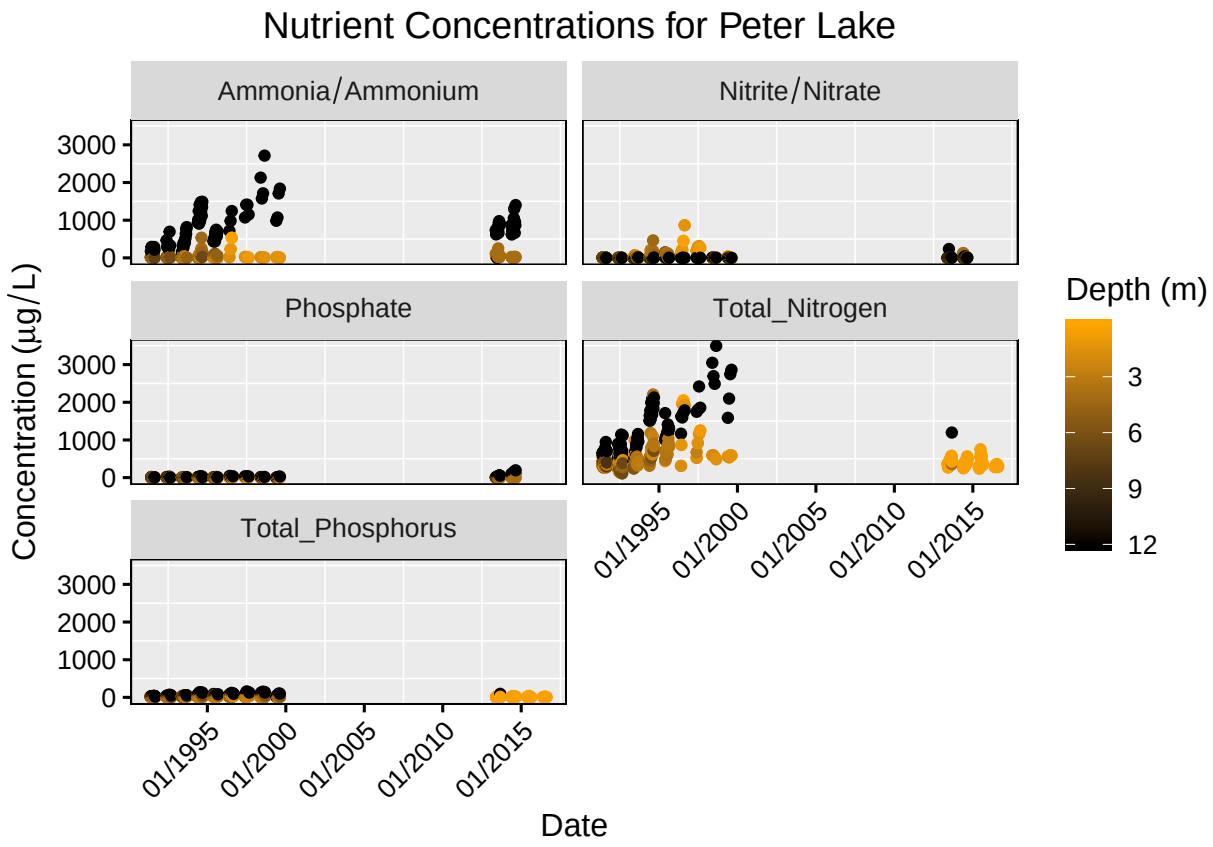
```
## Warning: Removed 22310 rows containing non-finite values (stat_smooth).
## Warning: Removed 22310 rows containing missing values (geom_point).
## Warning: Removed 2 rows containing missing values (geom_smooth).
```



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

```
# pull out paul lake nutrients from dataset and Nas for depth

peter.nutrients.data<-
  peter.paul.gathered %>%
  filter(lakename =="Peter Lake", depth>0)

#5 plots nutrients for paul lake by depth
```

```r
# create variable to story facet headers

levels(peter.nutrients.data$nutrient) <-  c("Ammonia/Ammonium", "Nitrite/Nitrate", "Phosphate", "Total_

peter.lake.nutrients <- ggplot(peter.nutrients.data, aes(x= sampledate, y = concentration,color =depth)
  geom_point()+
  facet_wrap(vars(nutrient), nrow = 3, labeller = label_parsed)+
  xlab("Date")+
  ylab(expression("Concentration"~"("*mu*g/L*")"))+
  theme_sb+
  labs(color='Depth (m)')+
  scale_color_gradient(low="black",high="orange", trans = "reverse")+
  theme(axis.text.x = element_text(angle = 45,  hjust = 1))+
  scale_x_date(date_breaks = "5 year", date_labels = "%m/%Y")+
  ggtitle(" Nutrient Concentrations for Peter Lake")+
  theme(plot.title = element_text(hjust = 0.5))




print(peter.lake.nutrients)
```



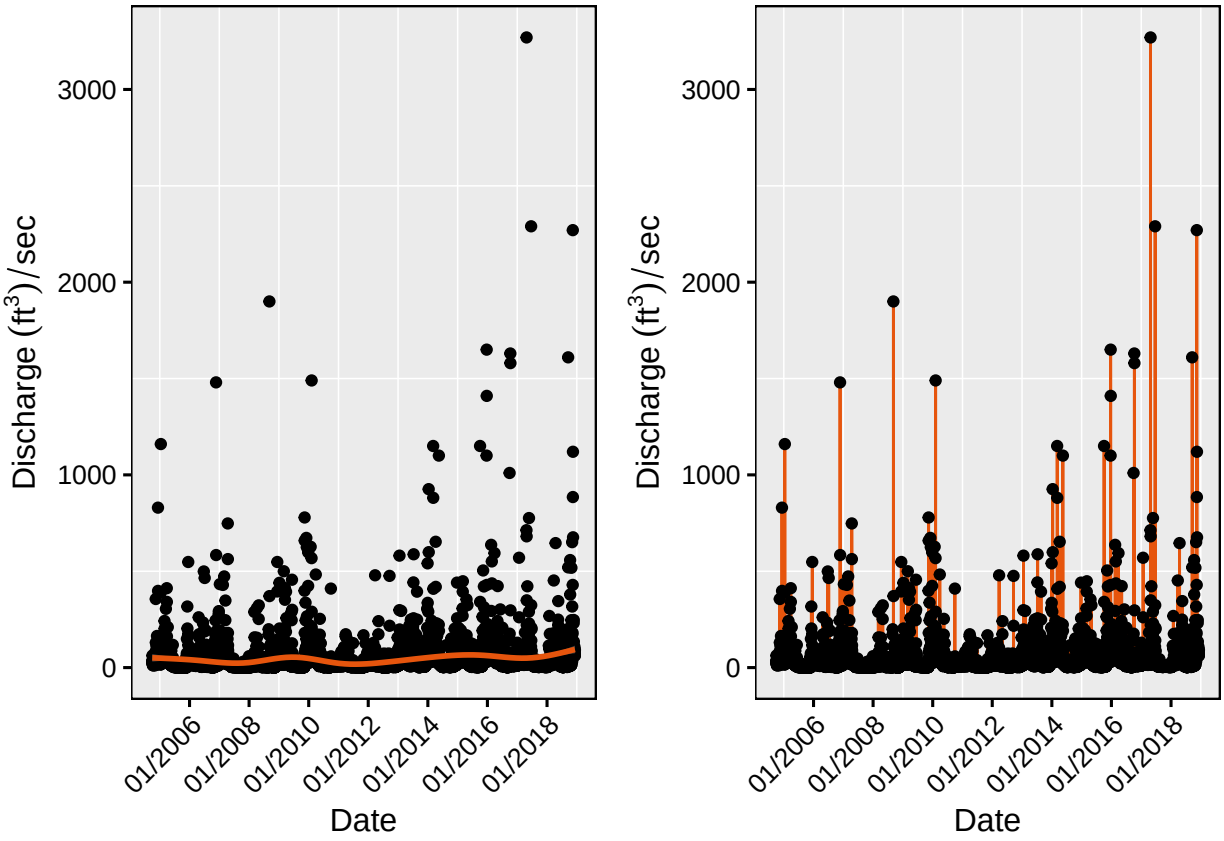
6. [USGS gauge] Plot discharge by date. Create two plots, one with the points connected with geom_line and one with the points connected with geom_smooth (hint: do not use method = "lm"). Place these

graphs on the same plot (hint: ggarrange or something similar)

```
# remove NAs from USGS data

USGS_NO_NAS <-
  USGS_flow_raw%>%
  filter(X84936_00060_00003 > 0)


#6

#connected with simple line
USGS.discharge.line.plot <- ggplot(USGS_NO_NAS, aes( x= datetime, y = X84936_00060_00003 ))+
geom_line(color = "#e6550d")+
geom_point()+
theme_sb+
ylab(expression(Discharge~(ft^{3})/sec))+
xlab("Date")+
scale_x_date(date_breaks = "2 year", date_labels = "%m/%Y")+
theme(axis.text.x = element_text(angle = 45,  hjust = 1))



#connected with smoothed line
USGS.discharge.smooth.plot <- ggplot(USGS_NO_NAS, aes( x= datetime, y = X84936_00060_00003 ))+
geom_point()+
geom_smooth(color = "#e6550d")+
theme_sb+
ylab(expression(Discharge~(ft^{3})/sec))+
xlab("Date")+
scale_x_date(date_breaks = "2 year", date_labels = "%m/%Y")+
theme(axis.text.x = element_text(angle = 45,  hjust = 1))




# ggarange them together

USGS_Double_plot <- ggarrange(USGS.discharge.smooth.plot,USGS.discharge.line.plot)
```

```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```

```
print(USGS_Double_plot)
```



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

Answer:The points that are connected by geom_line show oscillation of the data points from day to day whereas the points that are connected by geom_smooth show mean oscillation with time that is not effected by extreme storm events.

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

```
# pull out only the ones that dont have mg/L

eco_tox_mg <-
  EPA_ECOTOX %>%
  filter(Conc..Units..Std. == "AI mg/L")

#7

ECO_TOX_plot <- ggplot(eco_tox_mg, aes( x = Chemical.Name, y = Conc..Mean..Std., color = Chemical.Name)
  geom_boxplot()+
  theme_dark+
   xlab("Chemical Name")+
  ylab("Concentration (mg/L)")+
  labs(color='Chemical Name')+
  scale_colour_manual(values = c('#8c510a',"#bf812d","#dfc27d","#f6e8c3","#f5f5f5","#c7eae5","#80cdc1",
                                 "#01665e"))+
scale_x_discrete(labels = abbreviate)+
  ggtitle("Mean Concentrations of Chemicals")+
  theme(plot.title = element_text(hjust = 0.5))
```

```
print(ECO_TOX_plot)
```