Assignment 5: Data Visualization

Student Name

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
getwd()
```

```
## [1] "/Users/xiaqianyi/Documents/current semester/ENV 872/ENV 872/Assignments"
```

```
library(tidyverse)
PeterPaul_Processed <-read.csv("../Data/Processed/NTL-LTER_Lake_Nutrients_PeterPaul_Processed.csv")
PeterPaul_Gathered <- read.csv("../Data/Processed/NTL-LTER_Lake_Nutrients_PeterPaulGathered_Processed.c
EPAtox <- read.csv("../Data/Raw/ECOTOX_Neonicotinoids_Mortality_raw.csv")
USGS.flow.data <- read.csv("../Data/Raw/USGS_Site02085000_Flow_Raw.csv")
#2 Format Date
USGS.flow.data$datetime <- as.Date(USGS.flow.data$datetime, format = "%m/%d/%y")
USGS.flow.data$datetime <- format(USGS.flow.data$datetime, "%y%m%d")

create.early.dates <- (function(d) {
       paste0(ifelse(d > 181231,"19","20"),d)
       })
USGS.flow.data$datetime <- create.early.dates(USGS.flow.data$datetime)

USGS.flow.data$datetime <- as.Date(USGS.flow.data$datetime, format = "%Y%m%d")
```

```
class(PeterPaul_Gathered$sampledate)
```

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

```
PeterPaul_Processed$sampledate<-as.Date(PeterPaul_Processed$sampledate, format = "%Y%m%d")
PeterPaul_Gathered$sampledate<-as.Date(PeterPaul_Gathered$sampledate, format = "%Y-%m-%d")
# EPAtox$Pub..Year <- factor(EPAtox$Pub..Year, format = '%Y')
```

Define your theme

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

```
#3
QYtheme <- theme_classic(base_size= 14) +
  theme(axis.text = element_text(color = "black"))
```

Create graphs

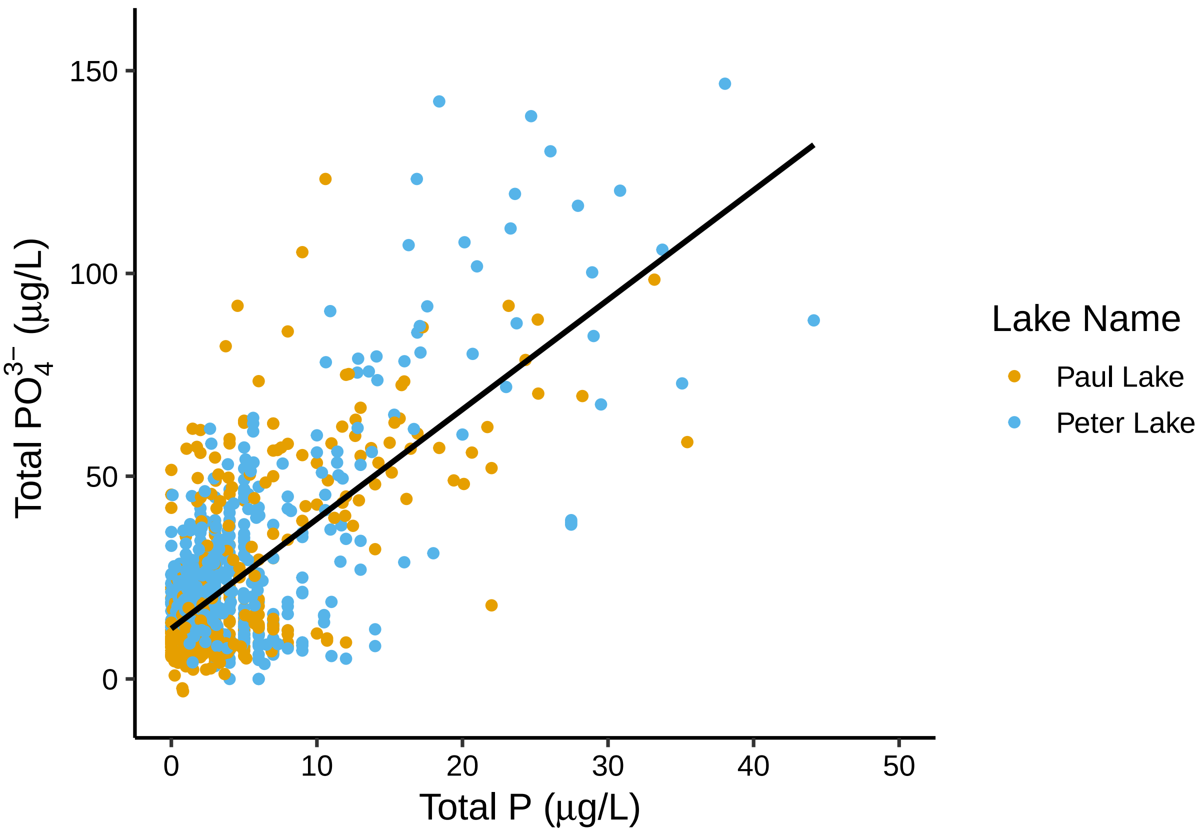
For numbers 4-7, create graphs that follow best practices for data visualization. To make your graphs "pretty," ensure your theme, color palettes, axes, and legends are edited to your liking.

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 PO4 (x)is phosphate, and TP (y) is phosphorus
library(viridis)
PeterPaulP_PO4 <-
  ggplot(PeterPaul_Processed, aes(x= po4, y=tp_ug,color=lakename))+
  xlim(0, 50)+
  geom_point()+
  scale_color_manual(values=c("#E69F00", "#56B4E9"))+
  # scale_shape_manual(values = c(21, 23))+
  # scale_fill_viridis(option = 'magma', direction = -1) +
  geom_smooth(method=lm, color="black", se=F)+
 labs(y=expression(paste('Total ',PO[4]^{'3-'}, ' (',mu, 'g/L) ')), x=expression(paste('Total P ', '(',
  #scale_color_discrete(name="Lake Name")+
  QYtheme
  #defaul color

print(PeterPaulP_PO4)
```

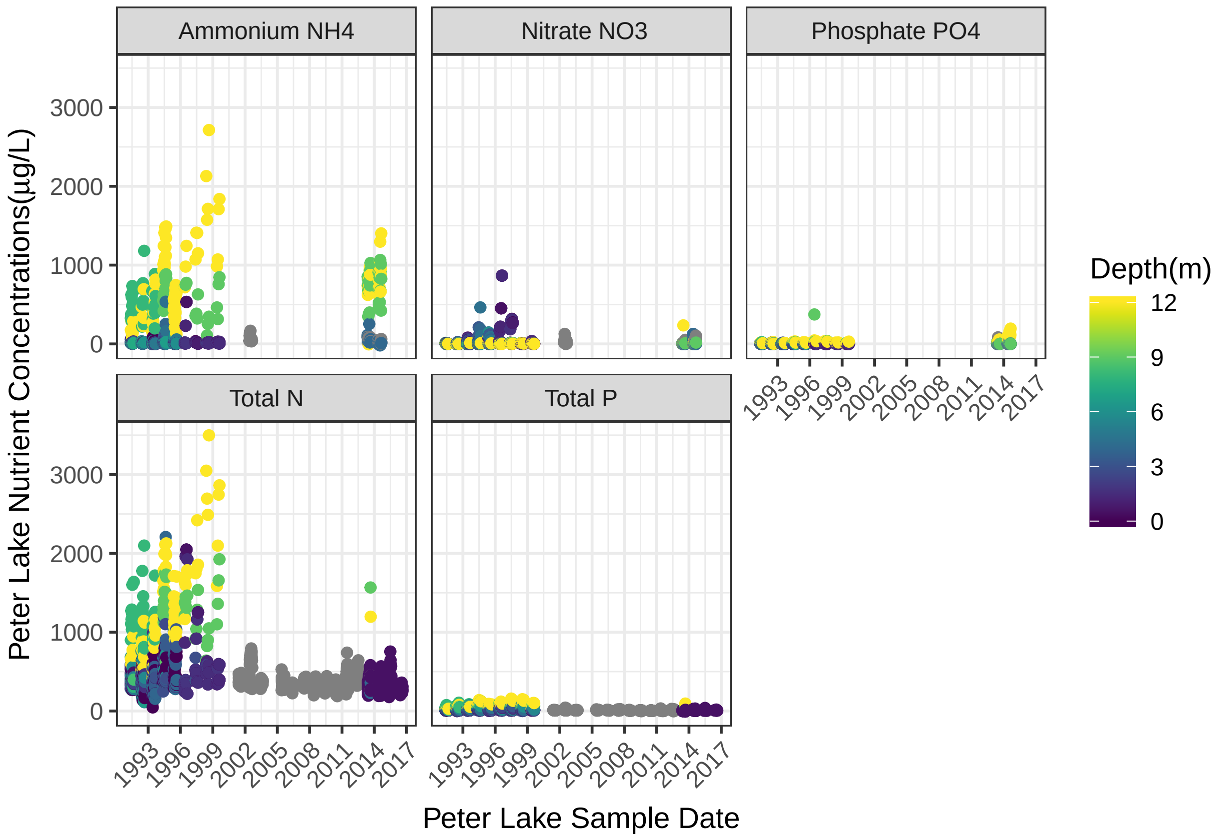


5. [NTL-LTER] Plot nutrients by date for Peter Lake, with separate colors for each depth. Facet your graph by the nutrient type.

```
#5
nutrient_name <- c(
  'nh34' = "Ammonium NH4",
  'no23' =" Nitrate NO3",
  'po4' = "Phosphate PO4",
  'tn_ug' = "Total N",
   'tp_ug' = "Total P"
)

PeterNutrients <-
  ggplot(subset(PeterPaul_Gathered, lakename = 'Paul Lake'))+
  geom_point(aes(x=PeterPaul_Gathered$sampledate, y= PeterPaul_Gathered$concentration, color=depth))+
   facet_wrap(vars(nutrient), labeller = as_labeller(nutrient_name))+
  labs(x= expression("Peter Lake Sample Date"), y= expression(paste('Peter Lake Nutrient Concentrations
  scale_x_date(limits = as.Date(c("1991-05-01", "2016-08-31")),
date_breaks = "36 months", date_labels = " %Y")+
  scale_color_viridis()+
   theme_bw()+
  theme(axis.text.x=element_text(angle = 45, hjust = 1))


print(PeterNutrients)
```



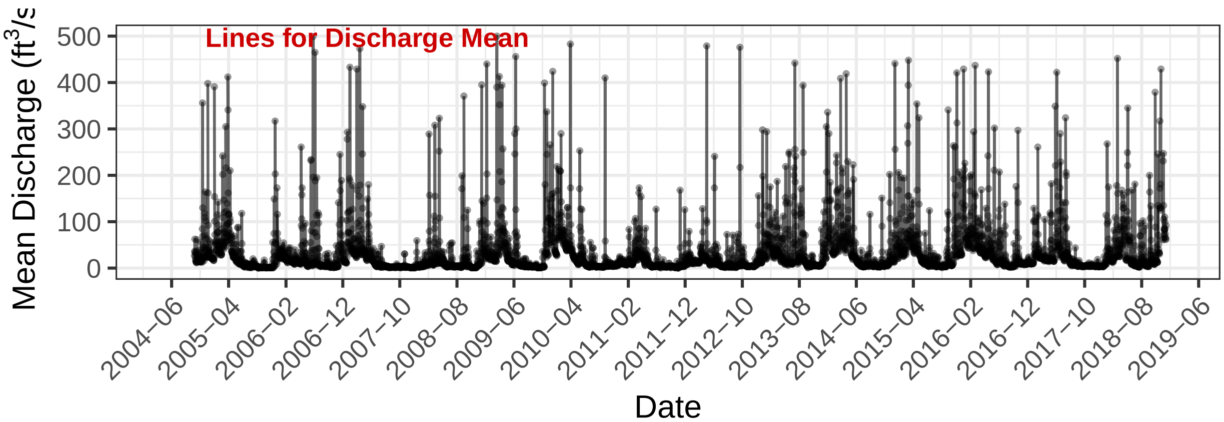
6. [USGS gauge] Plot discharge by date. Create two plots, one with the points connected with geom_line and one with the points connected with geom_smooth (hint: do not use method = "lm"). Place these graphs on the same plot (hint: ggarrange or something similar)

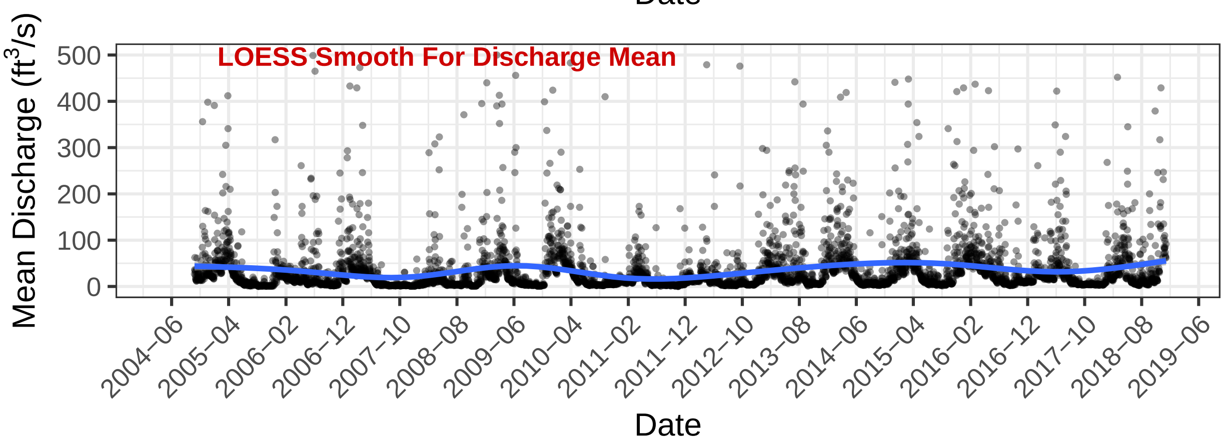
```
#6
library(ggpubr)
library(gridExtra)
colnames(USGS.flow.data) <- c("agency_cd", "site_no", "datetime", "discharge.max", "discharge.max.appro
"discharge.mean", "discharge.mean.approval",
"gage.height.max", "gage.height.max.approval",
"gage.height.min", "gage.height.min.approval",
"gage.height.mean", "gage.height.mean.approval")
DischargeLine <-
  ggplot(USGS.flow.data, aes(x= datetime, y=discharge.mean, na.rm=TRUE))+
  ylim(0,500)+
  scale_x_date(limits = as.Date(c("2004-05-01", "2018-12-31")),
date_breaks = "10 months", date_labels = "%Y-%m")+
  geom_point(alpha = 0.4, size=0.6)+
  geom_line(color='black', alpha=0.6)+
  labs(x= expression("Date"), y= expression(paste('Mean Discharge (', ft^3,'/s)')))+
theme_bw(base_size= 12)+
    theme(axis.text.x=element_text(angle = 45, hjust = 1))


DischargeSmooth <-
    ggplot(USGS.flow.data, aes(x= datetime, y=discharge.mean, na.rm=TRUE))+
  ylim(0,500)+
  scale_x_date(limits = as.Date(c("2004-05-01", "2018-12-31")),
```

```
date_breaks = "10 months", date_labels = "%Y-%m")+
  geom_point(alpha = 0.4, size=0.6)+
  geom_smooth(alpha=0.6)+
  labs(x= expression("Date"), y= expression(paste('Mean Discharge (', ft^3,'/s)')))+
theme_bw(base_size= 12)+
    theme(axis.text.x=element_text(angle = 45, hjust = 1))

ggarrange(DischargeLine,DischargeSmooth, nrow=2,labels = c("     Lines for Discharge Mean", "LOESS Smoo
          font.label = list(size = 10, color = "red3"), vjust=2)
```





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

> Answer: From the plot data, we know that most of the points are concentrate at a low value of y axis, therefore, when using geom line to connect them from point to point, we can hardly see the distinction at the bottom of the graph; When applying the geom smooth function, it generates the locally estimated scatterplot smoothing (LOESS), which allows to see a moving regression of the data point trend. The geom smooth may provide better interpretation of the data.

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

```
#7
class(EPAtox$Conc..Mean..Std.)
```

```
## [1] "numeric"
```

```
ChemicalConc <- EPAtox %>%
  group_by(Chemical.Name, Conc..Mean..Std.) %>%
  summarise(sd=sd(Conc..Mean..Std.),
```

```r
            mean=mean(Conc..Mean..Std.))
ChemicalC <-
  ggplot(EPAtox, aes(x=Chemical.Name, y=Conc..Mean..Std.))+
  geom_boxplot(fill='#A4A4A4', color="black", outlier.size = 1)+
  ylim(0,1500)+
  labs(x= expression("Chemicals"), y= expression('Mean Concentration (mg/L)'), hjust=1)+
  QYtheme+
  theme(axis.text.x=element_text(angle = 45, hjust = 1))

# plot1 <-
#   ggplot(ChemicalConc, aes(x=Chemical.Name, y=mean))+
#   geom_bar(stat = "identity", position = "dodge") +
# geom_errorbar(aes(ymin = mean-sd, ymax = mean+sd),position = "dodge")

print(ChemicalC)
```