Assignment 5: Data Visualization

Gaby Garcia

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.

1

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.

```
setwd("~/Desktop/Environmental Data Analytics/Environmental_Data_Analytics/Data/Processed")
library(tidyverse)
library(tidyr)
library(ggplot2)
library(viridis)
library(RColorBrewer)
library(colormap)
library(lubridate)


PeterPaul.chem.nutrients <- read.csv("LakeChemistryNutrientsPeterPaul.csv")
PeterPaul.nutrients.gathered <- read.csv("PeterPaulNutrientsGathered.csv")
EPAair <- read.csv("EPAair_O3PM25_3sites1718_processed.csv")
Neonicotinoids<-read.csv("ECOTOX_Neonicotinoids_Mortality_raw.csv")
USGS.flow.data <- read.csv("USGS_Site02085000_Flow_Raw.csv")
```

2

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).

```
PeterPaul.chem.nutrients$sampledate<-as.Date(PeterPaul.chem.nutrients$sampledate,
                                              format="%m/%d/%y")
PeterPaul.nutrients.gathered$sampledate<-as.Date(PeterPaul.nutrients.gathered$sampledate,
                                                  format="%m/%d/%y")


PeterPaul.chem.nutrients<- mutate(PeterPaul.chem.nutrients, month = month(sampledate))
PeterPaul.chem.nutrients<-mutate(PeterPaul.chem.nutrients, day=day(sampledate))
```

```
setwd("~/Desktop/Environmental Data Analytics/Environmental_Data_Analytics/Data/Processed")
USGS.flow.data <- read.csv("USGS_Site02085000_Flow_Raw.csv")
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")
##as.Date format argument should match the data table's format
```

Spread Nutrients Data

```
PeterPaulChemNutrientsSpread <- spread(PeterPaul.nutrients.gathered, nutrient, concentration)
class(PeterPaulChemNutrientsSpread$sampledate)
```

```
## [1] "Date"
```

```
PeterPaulChemNutrientsSpread$sampledate <- as.Date(PeterPaulChemNutrientsSpread$sampledate,
                                                    format="%m/%d/%y")
PeterPaulChemNutrientsSpread$lakename<-as.factor(PeterPaulChemNutrientsSpread$lakename)
```

3

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

```
library(ggplot2)
gabytheme <- theme_bw(base_size = 14) +
  theme(plot.title=element_text(face="bold", size="16", color="tomato2", hjust=0.5),
        axis.title=element_text(face="bold.italic", size=11, color="black"),
axis.text = element_text(face="bold", size=10, color = "black"),
panel.background=element_rect(fill="lightgray", color="darkblue"),
panel.border = element_rect(color = "black", size = 2),
legend.position = "top", legend.background = element_rect(fill="white", color="black"),
            legend.key = element_rect(fill="transparent", color="NA"))
```

Create graphs

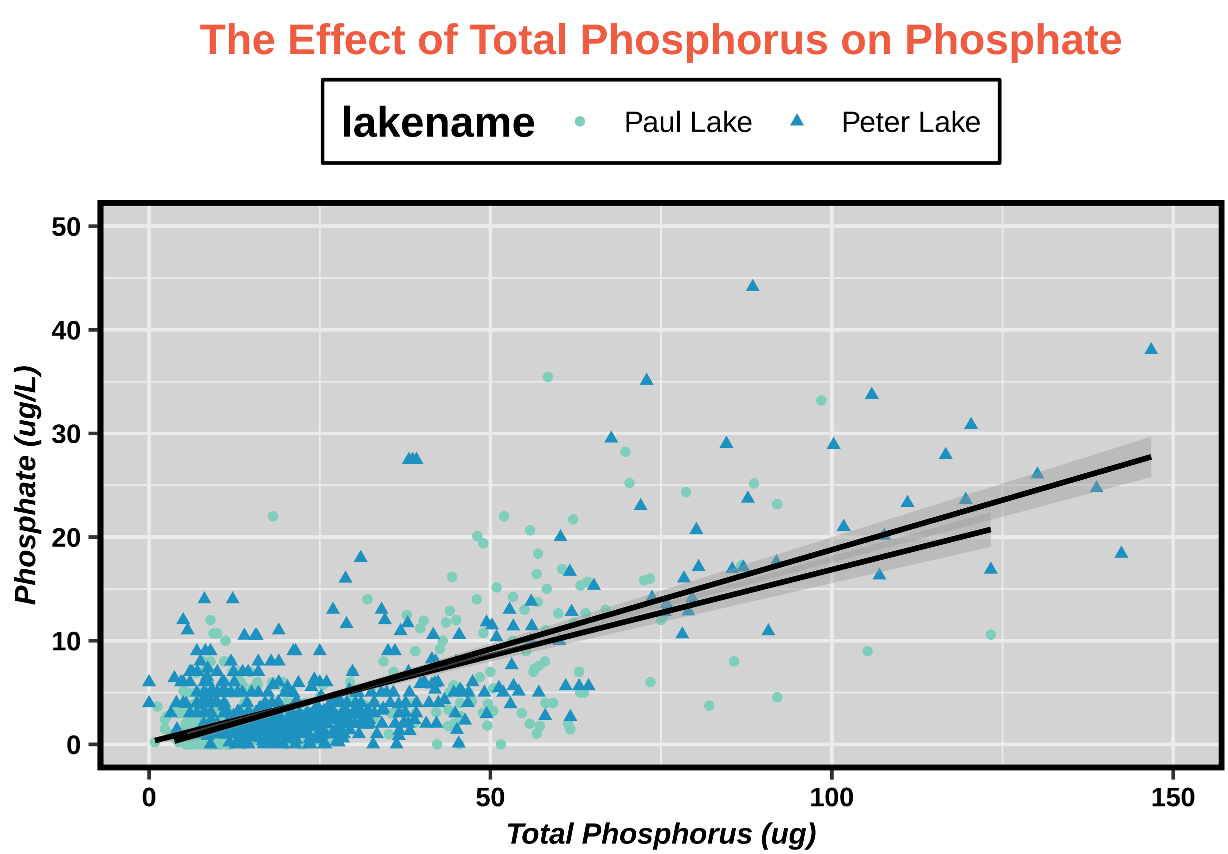
For numbers 4-7, create graphs that follow best practices for data visualization. To make your graphs "pretty," ensure your theme, color palettes, axes, and legends are edited to your liking.

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

4

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.

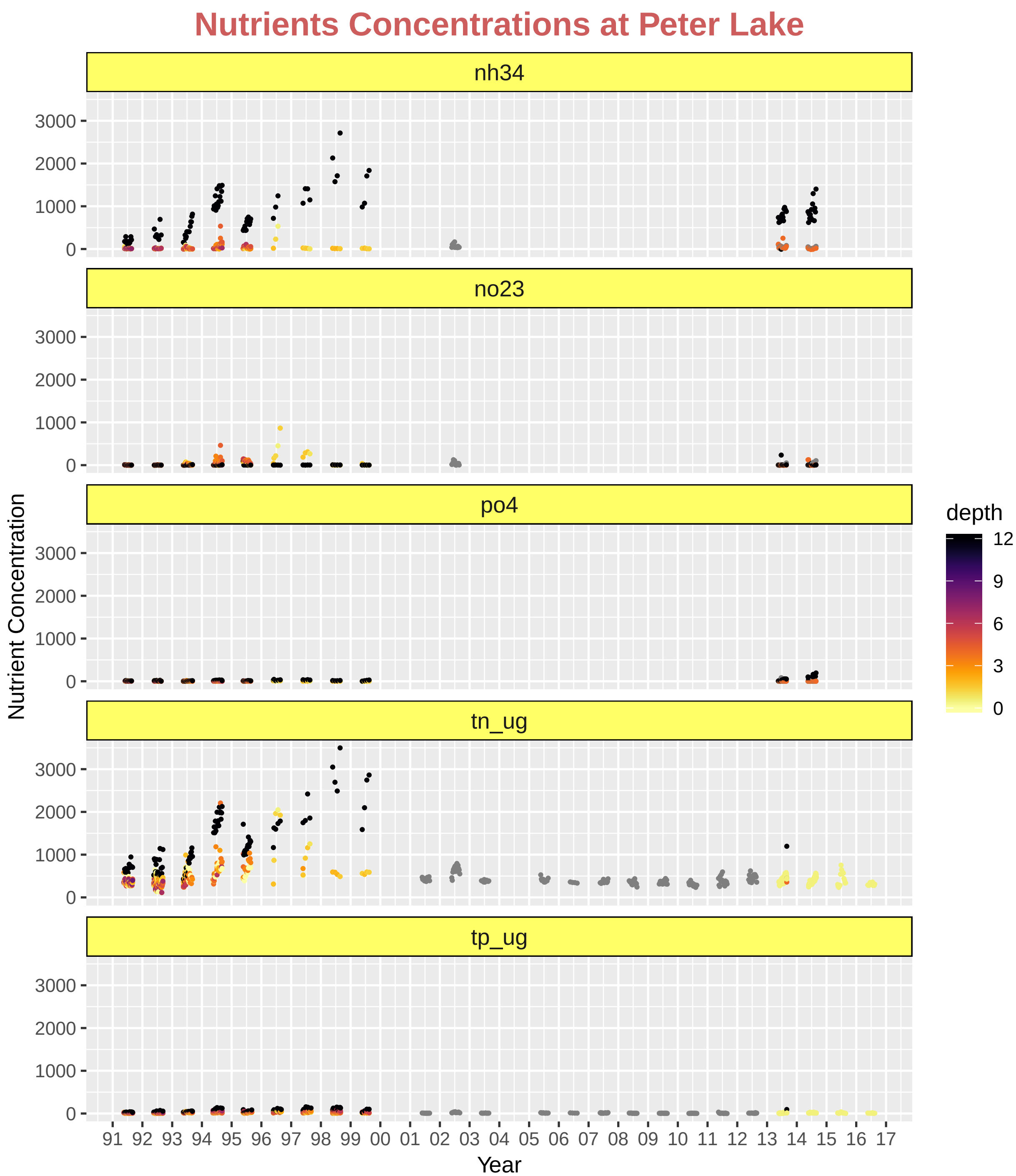
```
Plot4 <- ggplot(PeterPaulChemNutrientsSpread, aes(x = tp_ug, y = po4,
                                                    color=lakename, shape=lakename)) +
  geom_point() +
  geom_smooth(method="lm", color="black") +
  labs(title="The Effect of Total Phosphorus on Phosphate", x="Total Phosphorus (ug)",
      y="Phosphate (ug/L)") + xlim(0, 150) +ylim(0,50) +
    gabytheme +
    scale_color_manual(values = c("#7fcdbb", "#1d91c0")) +
  theme(legend.title = element_text(colour="Black", size=16, face="bold"))


##PO4 microg/L
print(Plot4)
```



5

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

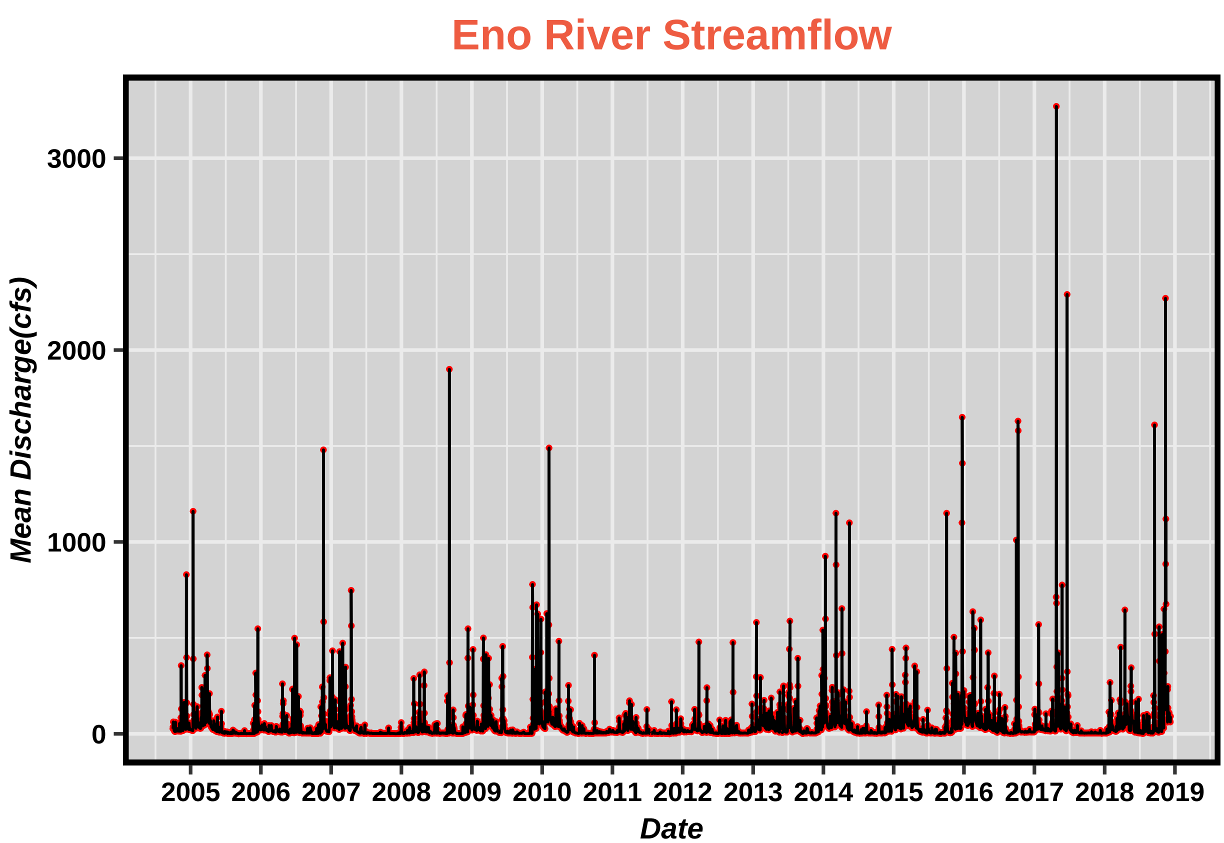
```
library(scales)
library(ggplot2)
Plot5<- ggplot(subset(PeterPaul.nutrients.gathered, lakename =="Peter Lake"),
               aes(x = sampledate, y = concentration, color=depth)) +
  geom_point(aes(x = sampledate, y = concentration),size=0.5) +
  facet_wrap(vars(nutrient), nrow=5) +
  theme(strip.text.x = element_text(size=11, angle=360),
          strip.text.y = element_text(size=10, face="bold", color="black"),
          strip.background = element_rect(colour="black", fill="#FFFF66")) +
  labs(title="Nutrients Concentrations at Peter Lake", x="Year",
       y="Nutrient Concentration", hjust=0.5) +
  scale_x_date(labels = date_format("%y"),breaks = date_breaks("1 year")) +
  theme(plot.title=element_text(face="bold", size="16", color="IndianRed", hjust=0.5))+
    scale_color_viridis(option = "inferno", direction = -1)
print(Plot5)
```


Nutrients Concentrations at Peter Lake
nh34
no23
po4
tn_ug
tp_ug
3000
2000
1000
0
Nutrient Concentration
depth
12
9
6
3
0
91 92 93 94 95 96 97 98 99 00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17
Year

6

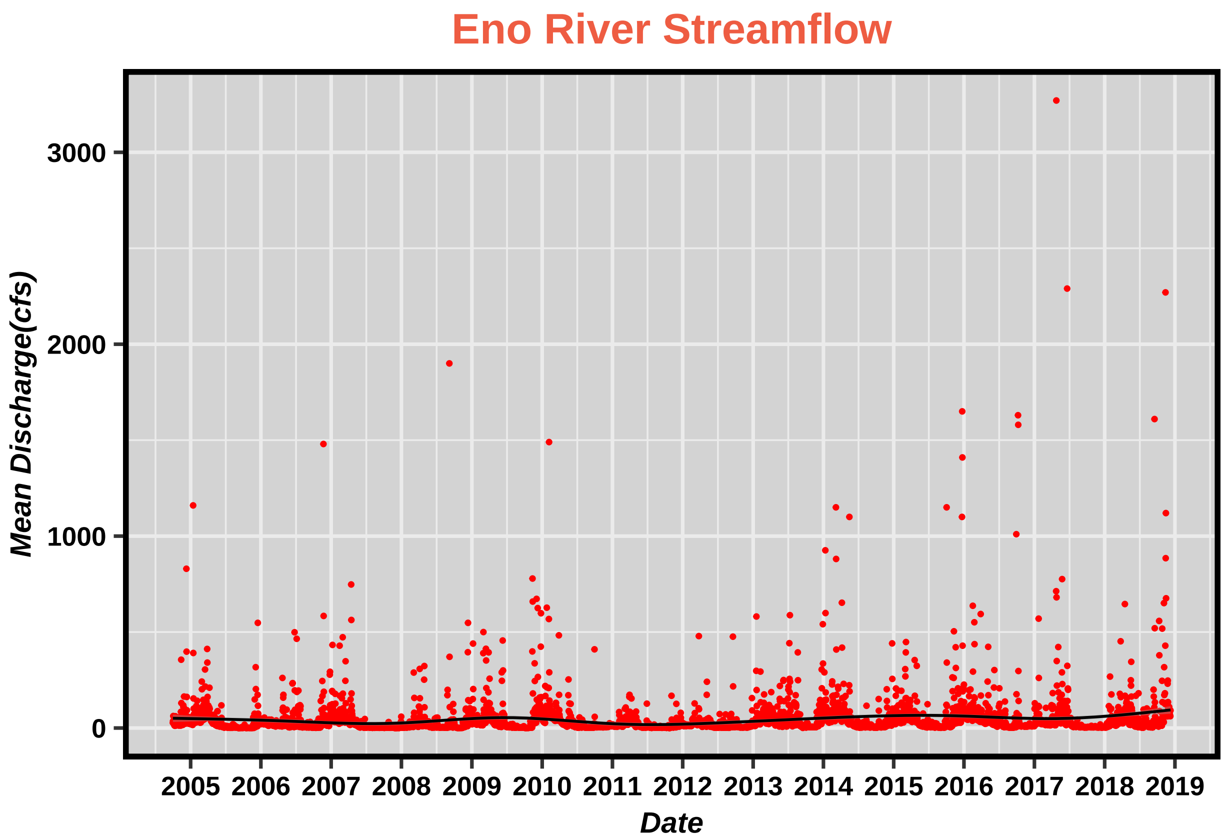
6. [USGS gauge] Plot discharge by date. Create two plots, one with the points connected with geom_line and one with the points connected with geom_smooth (hint: do not use method = "lm"). Place these graphs on the same plot (hint: ggarrange or something similar)

```
library(gridExtra)
library(scales)
library(dplyr)
library(tidyverse)

setwd("~/Desktop/Environmental Data Analytics/Environmental_Data_Analytics/Data/Processed")

USGS.flow.data2<-USGS.flow.data%>% slice(28034:33216)
USGS.flow.data2$datetime<-as.Date(USGS.flow.data2$datetime, format="%m/%d/%y")

Plot6<-ggplot(USGS.flow.data2, aes(x = datetime, y =X84936_00060_00003)) +
  geom_point(aes(x = datetime, y = X84936_00060_00003),size=0.5, color="red") +
   geom_line(aes(x = datetime, y =X84936_00060_00003), color="black", linetype=1, size=0.5) +
  labs(title="Eno River Streamflow", x="Date", y="Mean Discharge(cfs)") + gabytheme +
  scale_x_date(date_breaks = "1 year", date_labels = "%Y")


  print(Plot6)
```



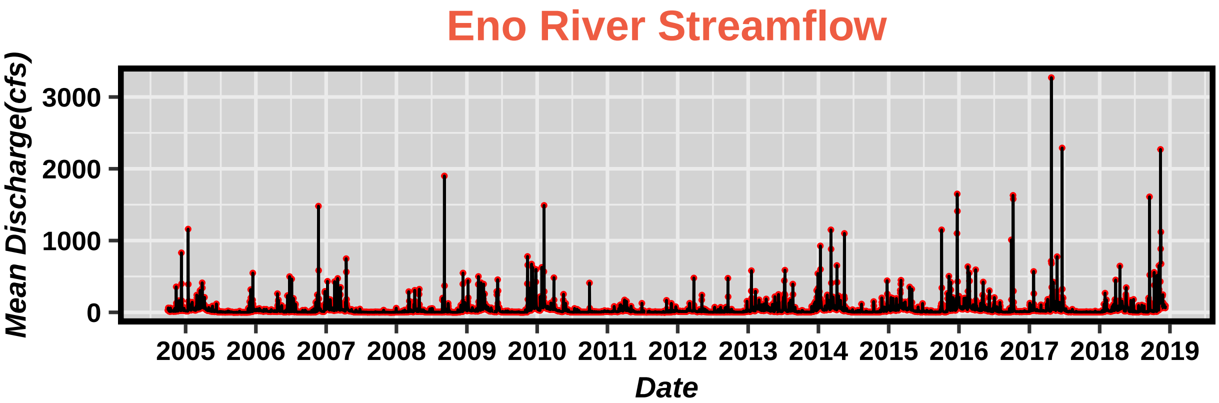

```
  Plot7<-ggplot(USGS.flow.data2, aes(x = datetime, y = X84936_00060_00003)) +
  geom_point(aes(x = datetime, y = X84936_00060_00003),size=0.5, color="red") +
```

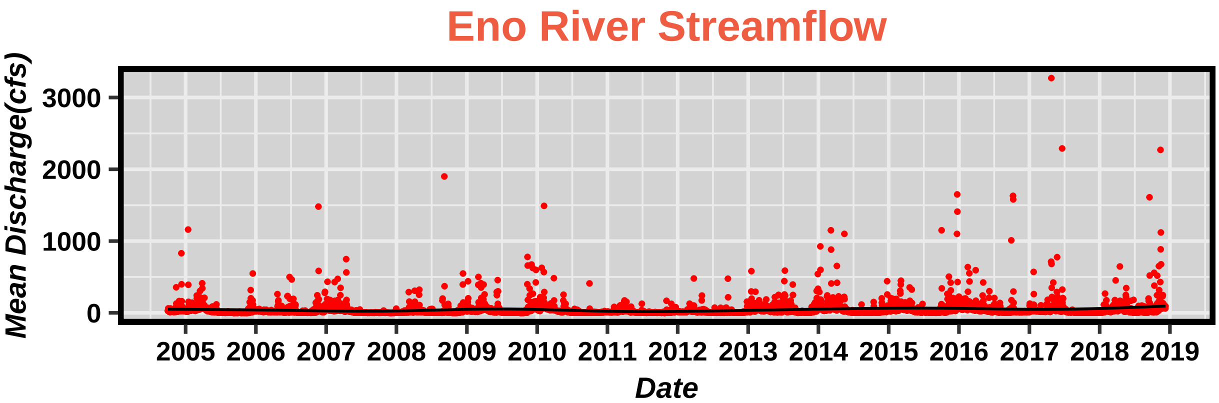
```r
  geom_smooth(aes(x = datetime, y = X84936_00060_00003, span=0.1), color="black", linetype=1, size=0.5)
 labs(title="Eno River Streamflow", x="Date", y="Mean Discharge(cfs)") +gabytheme +
scale_x_date(date_breaks = "1 year", date_labels = "%Y")


print(Plot7)
```



Use gg.arrange function to display graphs on the same plot

```r
grid.arrange(Plot6, Plot7, nrow=2)
```





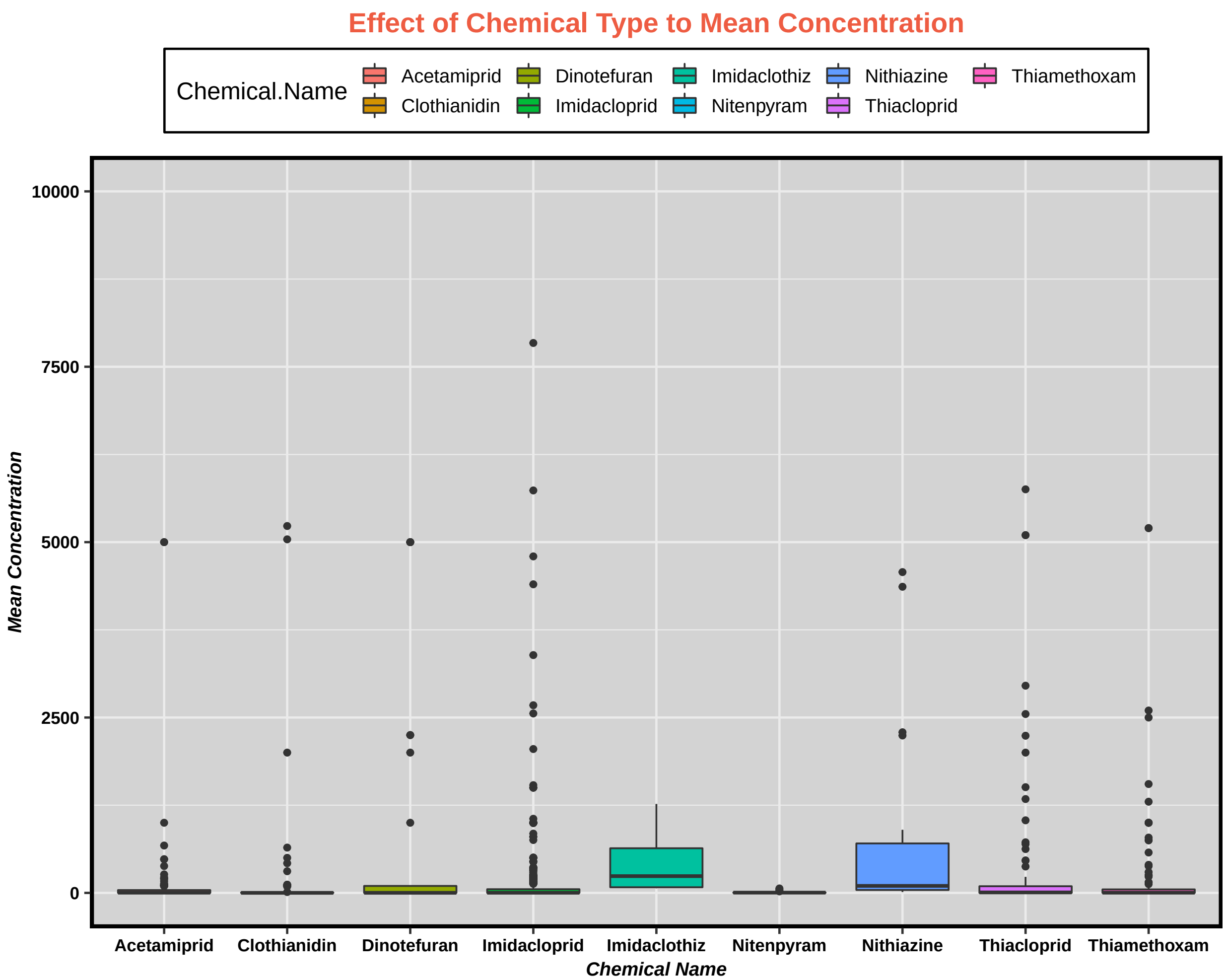
Question: How do these two types of lines affect your interpretation of the data?

> Answer: Geom_line connects the data's observations in the order of the variable on the x-axis. Using geom_lines, I can't see data patterns across the years. The geom_smooth line is for regression lines on a scatter plot, and shows us how the concentration data changes over time. Therefore, geom_smooth is a more useful geom in this case.

7

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

```
Plot8<-ggplot(Neonicotinoids) +
  geom_boxplot(aes(x=Chemical.Name, y=Conc..Mean..Std., fill=Chemical.Name)) + ylim(0,10000) +
  labs(title="Effect of Chemical Type to Mean Concentration", x="Chemical Name", y="Mean Concentration"
theme(legend.title = element_text(colour="IndianRed", size=16, face="bold")) +gabytheme




print(Plot8)
```


Effect of Chemical Type to Mean Concentration
Chemical.Name
Acetamiprid
Clothianidin
Dinotefuran
Imidacloprid
Imidaclothiz
Nitenpyram
Nithiazine
Thiacloprid
Thiamethoxam
Mean Concentration
10000
7500
5000
2500
0
Chemical Name