# Assignment 5: Data Visualization

*Laurie Muzzy*

## 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).

```
getwd()
```

```
## [1] "/Users/laurie/Desktop/Envtl_Data_Analytics/MuzzyGitFile/Assignments"
```

```
library(lubridate)
```

```
## Warning: package 'lubridate' was built under R version 3.4.4
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.4.4
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:lubridate':
##
##     intersect, setdiff, union

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.4.4
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 3.4.2

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

## v tibble  2.0.1     v purrr   0.3.0
## v tidyr   0.8.2     v stringr 1.3.1
## v readr   1.3.1     v forcats 0.3.0

## Warning: package 'tibble' was built under R version 3.4.4

## Warning: package 'tidyr' was built under R version 3.4.4

## Warning: package 'readr' was built under R version 3.4.4

## Warning: package 'purrr' was built under R version 3.4.4

## Warning: package 'stringr' was built under R version 3.4.4

## Warning: package 'forcats' was built under R version 3.4.3

## -- Conflicts -------------------------------------------- tidyverse_conflicts() --
## x lubridate::as.difftime() masks base::as.difftime()
## x lubridate::date()        masks base::date()
## x dplyr::filter()          masks stats::filter()
## x lubridate::intersect()   masks base::intersect()
## x dplyr::lag()             masks stats::lag()
## x lubridate::setdiff()     masks base::setdiff()
## x lubridate::union()       masks base::union()
```

```
library(tidyr)

#tidy Peter Paul
PeterPaul.chem.nutrients <- read.csv("../Data/Processed/NTL-LTER_Lake_Nutrient_Phys_PeterPaul_Processed
#View(PeterPaul.chem.nutrients) #23372 rows, 14 col

#gathered Peter Paul
PeterPaul.chem.nut.gathered <-   read.csv("../Data/Processed/NTL-LTER_Lake_Nutrients_PeterPaulGathered_
#View(PeterPaul.chem.nut.gathered) #7997 rows, 7 col

#USGS stream gauge
USGS.flow.data <-
read.csv("../Data/Raw/USGS_Site02085000_Flow_Raw.csv")
#View(USGS.flow.data) #33216 rows, 15 col, date is %m/%d/%y
```

```
#EPA Neonic
ECOTOX_Neonic <-
read.csv("../Data/Raw/ECOTOX_Neonicotinoids_Mortality_raw.csv", header = TRUE)
#View(ECOTOX_Neonic) #1283 rows, 13 col

#2
USGS.flow.data$datetime <- as.Date(USGS.flow.data$datetime, format = "%m/%d/%y")
```

```
## Warning in strptime(x, format, tz = "GMT"): unknown timezone 'zone/tz/
## 2018i.1.0/zoneinfo/America/New_York'
```

```
#turned the dates in 1900s into 2000s
class(USGS.flow.data$datetime) # Date
```

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

```
colnames(USGS.flow.data) <- c("agency_cd", "site_no", "Date",
                              "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")

USGS.flow.data$Date <- format(USGS.flow.data$Date, format = "%y%m%d")
#turning datetime into 6character string 280101 for jan 1, 1928
create.early.dates <- (function(d) {
       paste0(ifelse(d > 181231,"19","20"),d)
       })
USGS.flow.data$Date <- create.early.dates(USGS.flow.data$Date) #long character string
USGS.flow.data$Date <- as.Date(USGS.flow.data$Date, format = "%Y%m%d")

class(USGS.flow.data$Date) #Date
```

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

## Define your theme

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

```
#3
LFMtheme <- theme_light(base_size = 11) +
theme(axis.text = element_text(color = "dark gray"), legend.position = "right")
theme_set(LFMtheme)

#install.packages("viridis")
#install.packages("colormap")
#install.packages("RColorBrewer")
library(viridis)
```

```
## Warning: package 'viridis' was built under R version 3.4.4
```

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

```
## Warning: package 'viridisLite' was built under R version 3.4.3
```

```
#library(RColorBrewer)
#library(colormap)
```

## 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.

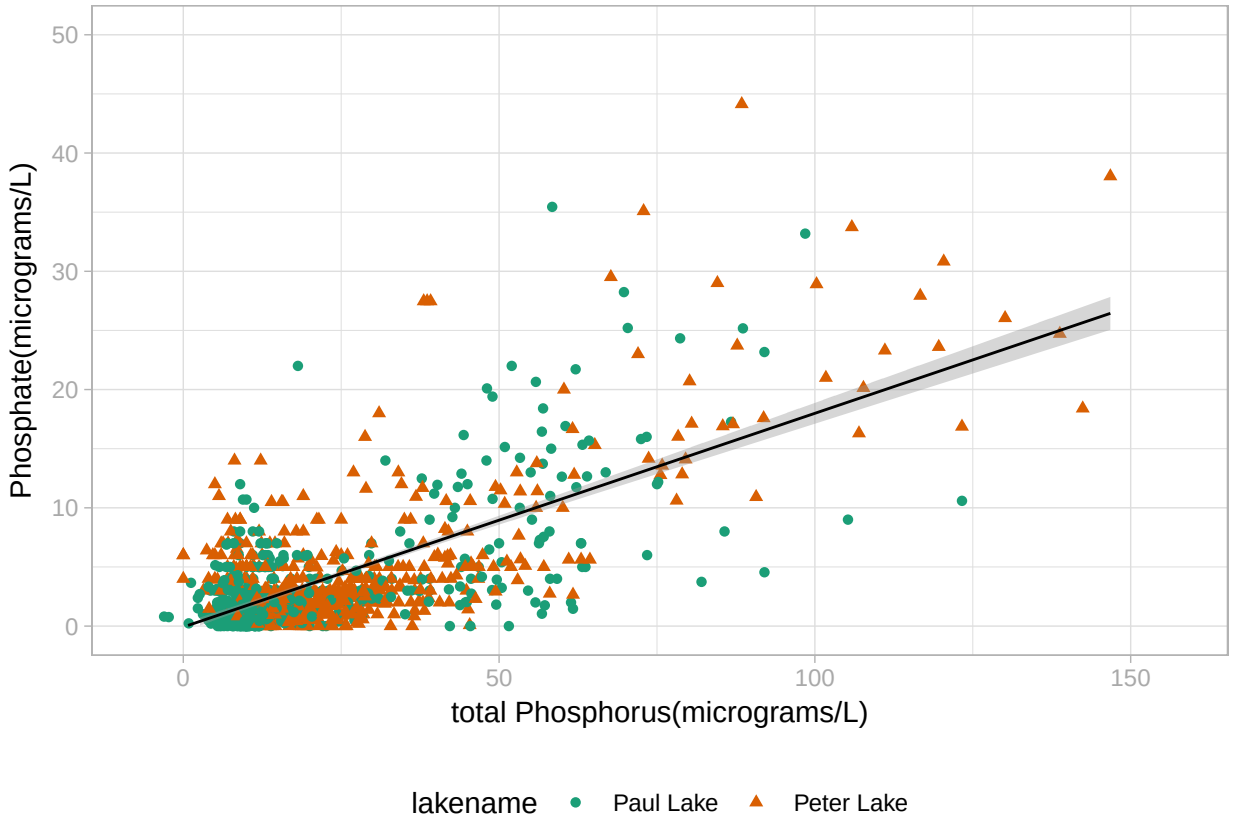
4. [NTL-LTER] Plot total phosphorus by phosphate, with separate aesthetics for Peter and Paul lakes. Add a line of best fit and color it black.

```
#4
PeterPaul.P.PO4 <- ggplot(PeterPaul.chem.nutrients, aes(x = tp_ug, y = po4)) +
geom_point(aes(color = lakename, shape = lakename)) +
geom_smooth(method = lm, size = 0.5, color = "black") +
ylim(0,50) +
xlab(expression("total Phosphorus(micrograms/L)")) +
ylab(expression("Phosphate(micrograms/L)")) +
scale_color_brewer(palette = "Dark2", direction = 1) +
theme(axis.text = element_text(color = "dark gray"), legend.position = "bottom")
print(PeterPaul.P.PO4)
```

```
## 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).
```

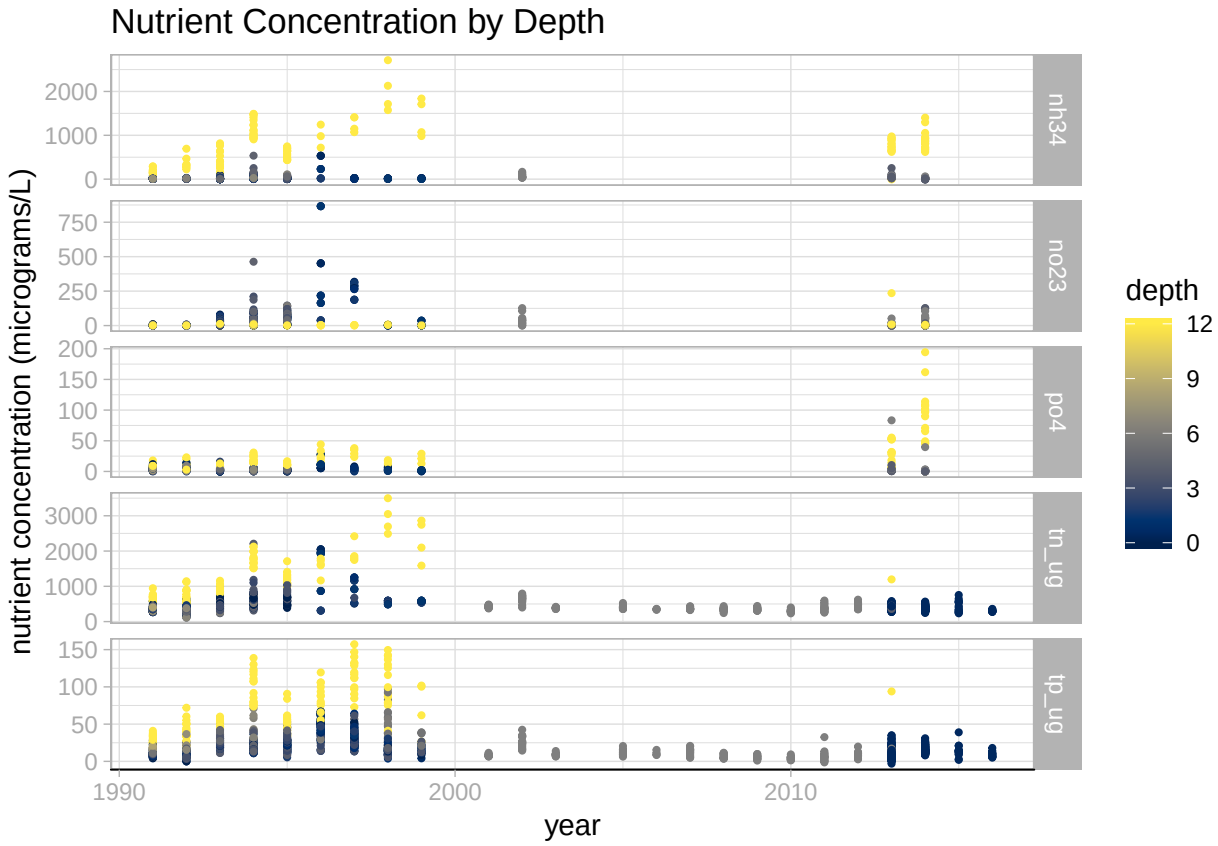


5. [NTL-LTER] Plot nutrients by date for Peter Lake, with separate colors for each depth. Facet your graph by the nutrient type.

```
#5

Peter.lake <- filter(PeterPaul.chem.nut.gathered, lakename == "Peter Lake")

Peter.nutrient <- ggplot(Peter.lake, aes(x = year4, y = concentration, color = depth)) +
geom_point(aes(), size = 0.75) +
facet_wrap(vars(nutrient), scales = "free_y", ncol = 1, strip.position = "right") +
scale_colour_viridis(option = "cividis") +
labs(x = "year", y = "nutrient concentration (micrograms/L)", title = "Nutrient Concentration by Depth"
theme(axis.line.x = element_line(color = "black"))
print(Peter.nutrient)
```




```
# can't figure out how to change facet wrap labels; levels = c("Ammonium", "Nitrate", "Phosphate", "Tot
```

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.max by datetime w/geom_line

USGS.flow.discharge1 <-
ggplot(USGS.flow.data, aes(x = Date, y = discharge.max)) +
geom_point(size = 0.5, alpha = 0.3) +
geom_line(alpha = 0.3) +
```

```
scale_y_log10()

#discharge.max by datetime w/geom_smooth
USGS.flow.discharge2 <-
ggplot(USGS.flow.data, aes(x = Date, y = discharge.max)) +
geom_point(size = 0.5, alpha = 0.3) +
geom_smooth(color = "magenta") +
scale_y_log10()

#same grid
#install.packages("gridExtra")
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

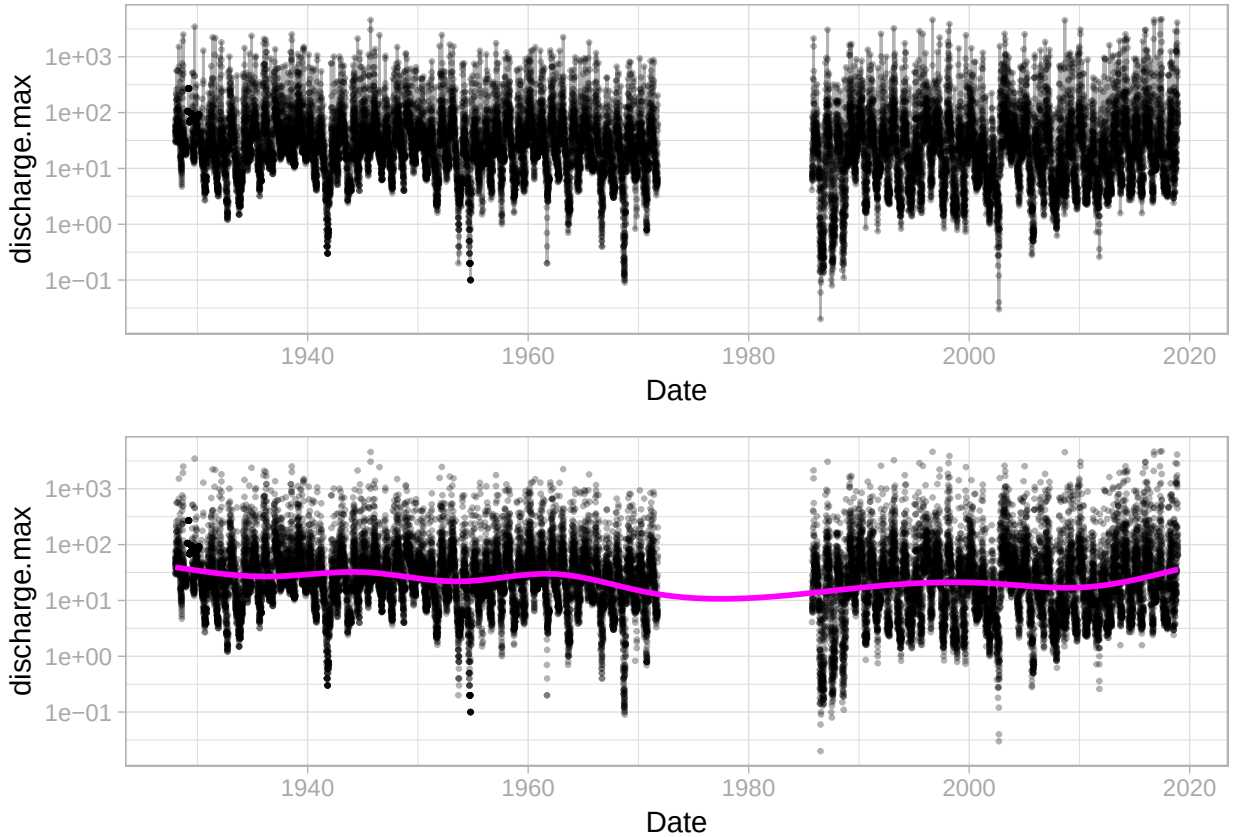
```
grid.arrange(USGS.flow.discharge1, USGS.flow.discharge2, newpage = TRUE)
```

```
## Warning: Removed 5113 rows containing missing values (geom_point).
```

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

```
## Warning: Removed 5113 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 5113 rows containing missing values (geom_point).
```



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

Answer: The linear method shows seasonal differences in discharge, as would be expected; the smooth trend line just basically adds a line, though it is easier to see the outliers with no lines connecting points.(Also, there are many years of data are missing, from about 1970 to 1988.)

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

```
#7
Ecotox.chem.concn <- ggplot(subset(ECOTOX_Neonic, Conc..Units..Std. == "AI mg/L")) +
geom_boxplot(aes(x = Chemical.Name, y = Conc..Mean..Std., fill = Chemical.Name)) +
labs(x = "", y = "Mean Chemical Concentration", title = "Pesticide Concentrations") +
theme(axis.text.x = element_text(angle = 45, hjust = 1, size = 8)) +
scale_y_log10()
print(Ecotox.chem.concn)
```