9: Data Visualization

Environmental Data Analytics | Kateri Salk

Spring 2019

LESSON OBJECTIVES

1. Perform simple data visualizations in the R package `ggplot`
2. Develop skills to adjust aesthetics and layers in graphs
3. Apply a decision tree framework for appropriate graphing methods

SET UP YOUR DATA ANALYSIS SESSION

```
getwd()
```

```
## [1] "/Users/ks501/Documents/GithubRepos/ENV872"
```

```
library(tidyverse)

PeterPaul.chem.nutrients <- read.csv("./Data/Processed/NTL-LTER_Lake_Chemistry_Nutrients_PeterPaul_Proce
PeterPaul.nutrients.gathered <- read.csv("./Data/Processed/NTL-LTER_Lake_Nutrients_PeterPaulGathered_Pr
PeterPaul.chem.nutrients.summaries <- read.csv("./Data/Processed/NTL-LTER_Lake_Summaries_PeterPaul_Proce
EPAair <- read.csv("./Data/Processed/EPAair_O3PM25_3sites1718_processed.csv")

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

GGPLOT

ggplot, called from the package `ggplot2`, is a graphing and image generation tool in R. This package is part of tidyverse. While base R has graphing capabilities, ggplot has the capacity for a wider range and more sophisticated options for graphing. ggplot has only a few rules:

- The first line of ggplot code always starts with `ggplot()`
- A data frame must be specified within the `ggplot()` function. Additional datasets can be specified in subsequent layers.
- Aesthetics must be specified, most commonly x and y variables but including others. Aesthetics can be specified in the `ggplot()` function or in subsequent layers.
- Additional layers must be specified to fill the plot.

Geoms

Here are some commonly used layers for plotting in ggplot:

- geom_bar
- geom_histogram
- geom_freqpoly
- geom_boxplot
- geom_violin
- geom_dotplot

- geom_point
- geom_errorbar
- geom_smooth
- geom_line
- geom_area
- geom_abline (plus geom_hline and geom_vline)
- geom_text

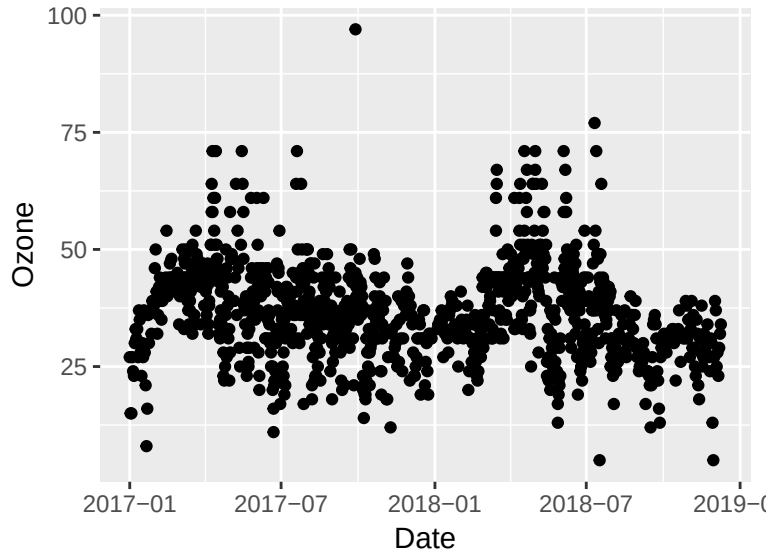
Aesthetics

Here are some commonly used aesthetic types that can be manipulated in ggplot:

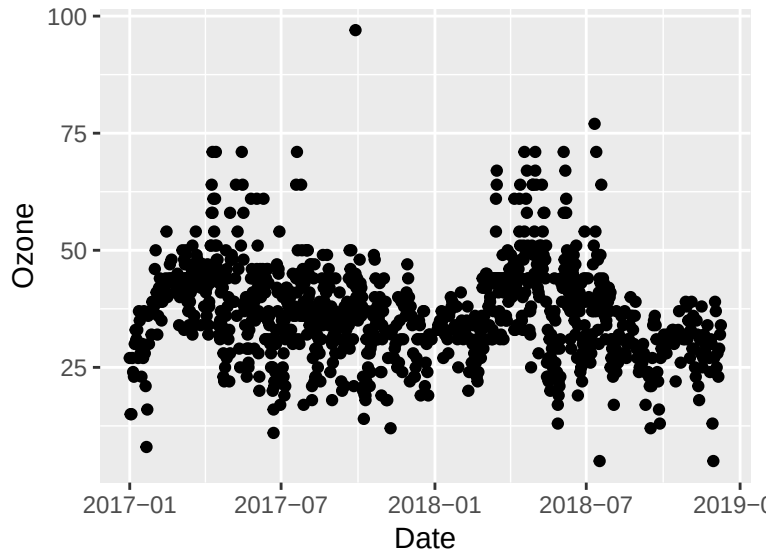
- color
- fill
- shape
- size
- transparency

Plotting continuous variables over time: Scatterplot

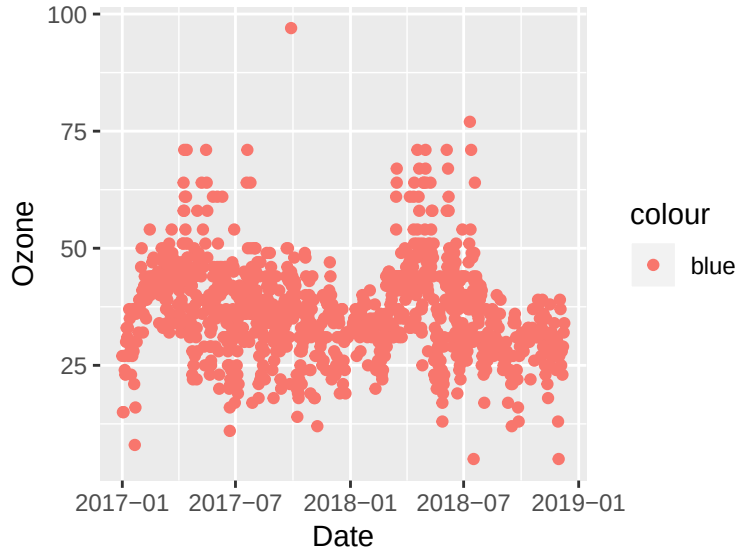
```
# Scatterplot
ggplot(EPAair, aes(x = Date, y = Ozone)) +
  geom_point()
```



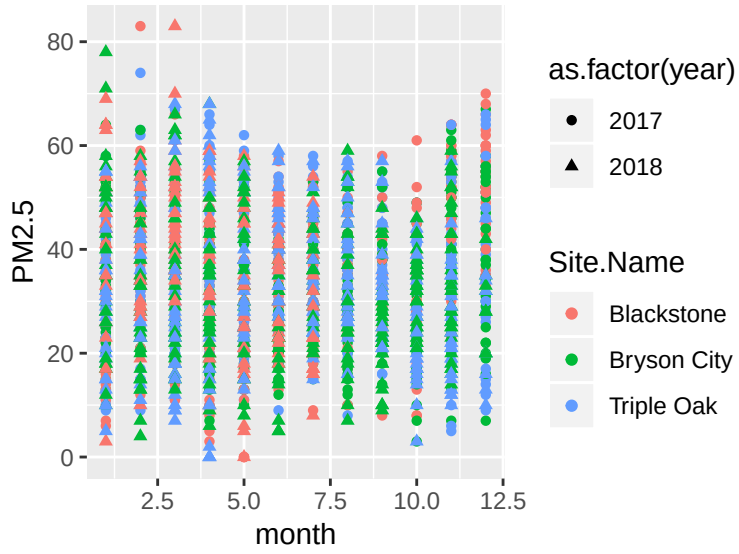

```
O3plot <- ggplot(EPAair) +
  geom_point(aes(x = Date, y = Ozone))
print(O3plot)
```



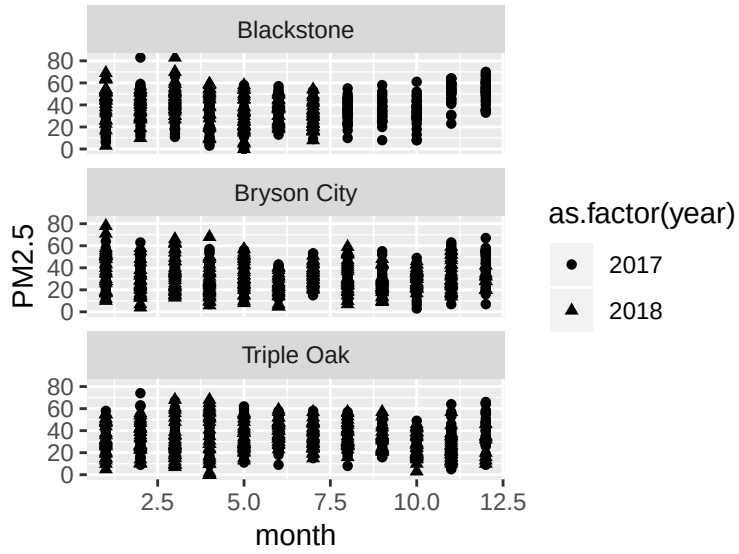

```
# Fix this code
O3plot2 <- ggplot(EPAair) +
  geom_point(aes(x = Date, y = Ozone, color = "blue"))
print(O3plot2)
```



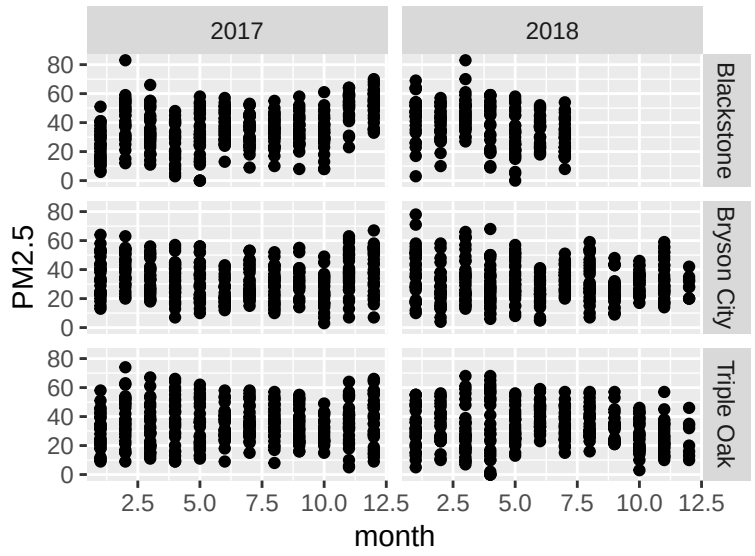

```
# Add additional variables
PMplot <-
  ggplot(EPAair, aes(x = month, y = PM2.5, shape = as.factor(year), color = Site.Name)) +
  geom_point()
print(PMplot)
```



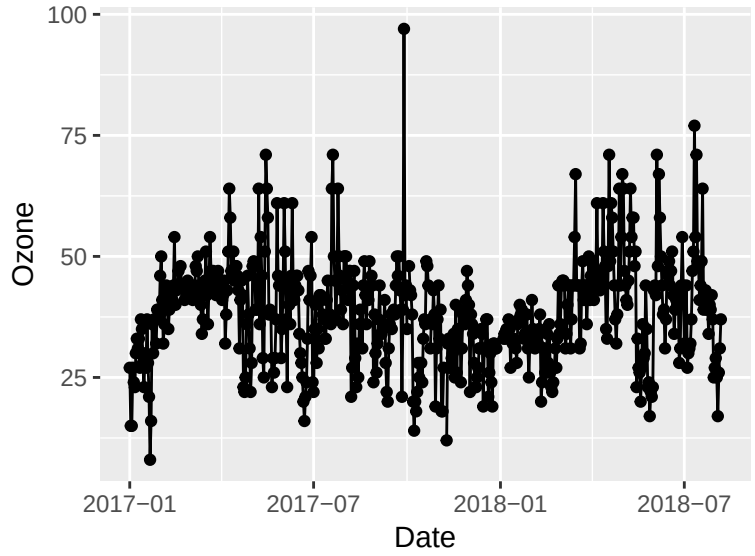

```
# Separate plot with facets
PMplot.faceted <-
  ggplot(EPAair, aes(x = month, y = PM2.5, shape = as.factor(year))) +
  geom_point() +
  facet_wrap(vars(Site.Name), nrow = 3)
print(PMplot.faceted)
```




```
PMplot.faceted2 <-
  ggplot(EPAair, aes(x = month, y = PM2.5)) +
  geom_point() +
  facet_grid(Site.Name ~ year)
print(PMplot.faceted2)
```



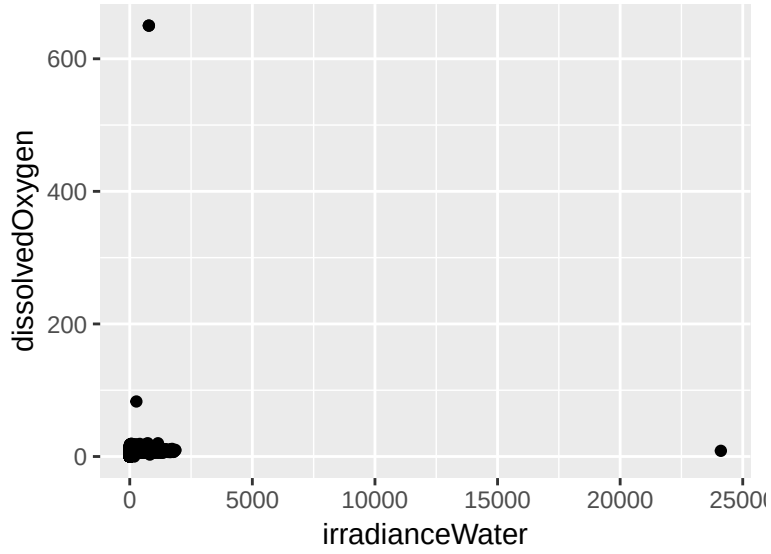

```
# Filter dataset within plot building
O3plot.Blackstone <-
  ggplot(subset(EPAair, Site.Name == "Blackstone"), aes(x = Date, y = Ozone)) +
  geom_point() +
  geom_line()
print(O3plot.Blackstone)
```



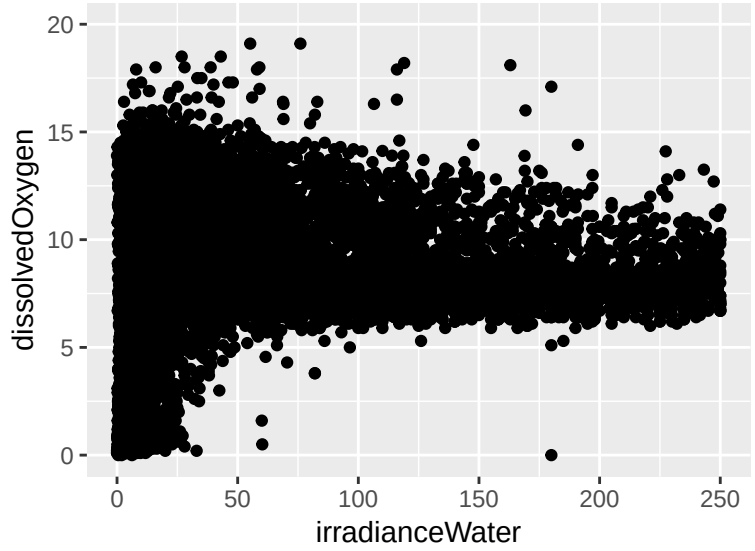

```
# Exercise: build your own scatterplots of PeterPaul.chem.nutrients

# 1.
# Plot surface temperatures by day of  year.
# Color your points by year, and facet by lake in two rows.


#2.
# Plot temperature by date. Color your points by depth.
# Change the size of your point to 0.5
```

Plotting the relationship between two continuous variables: Scatterplot

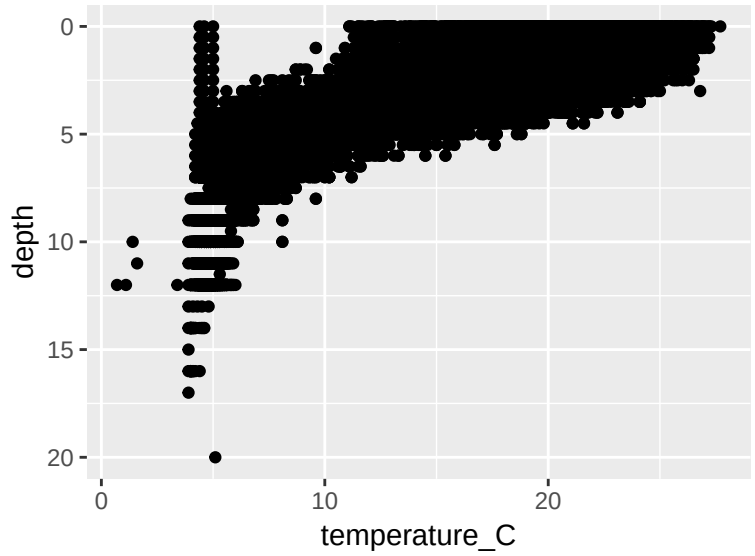
```
# Scatterplot
lightvsDO <-
  ggplot(PeterPaul.chem.nutrients, aes(x = irradianceWater, y = dissolvedOxygen)) +
  geom_point()
print(lightvsDO)
```



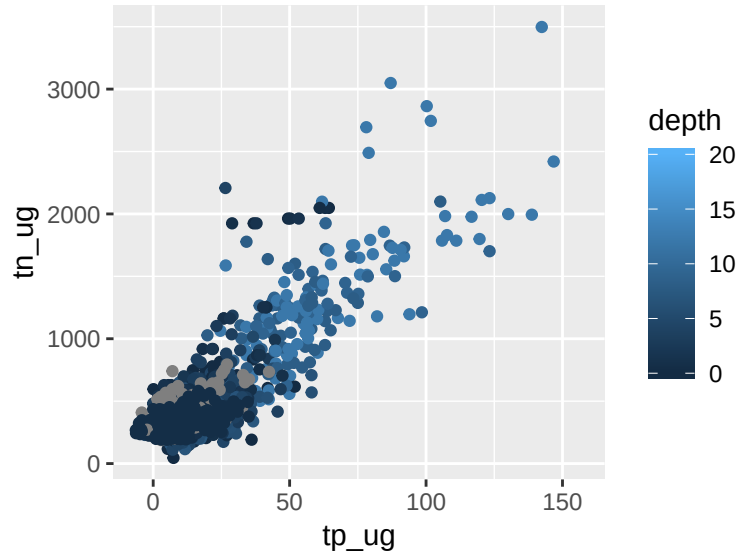

```
# Adjust axes
lightvsDOfixed <-
  ggplot(PeterPaul.chem.nutrients, aes(x = irradianceWater, y = dissolvedOxygen)) +
  geom_point() +
  xlim(0, 250) +
  ylim(0, 20)
print(lightvsDOfixed)
```



```r
# Depth in the fields of limnology and oceanography is on a reverse scale
tempvsdepth <-
  ggplot(PeterPaul.chem.nutrients, aes(x = temperature_C, y = depth)) +
  #ggplot(PeterPaul.chem.nutrients, aes(x = temperature_C, y = depth, color = daynum)) +
  geom_point() +
  scale_y_reverse()
print(tempvsdepth)
```




```r
NvsP <-
  ggplot(PeterPaul.chem.nutrients, aes(x = tp_ug, y = tn_ug, color = depth)) +
  geom_point() #+
  #geom_smooth(method = lm) +
  #geom_abline(aes(slope = 16, intercept = 0))
print(NvsP)
```

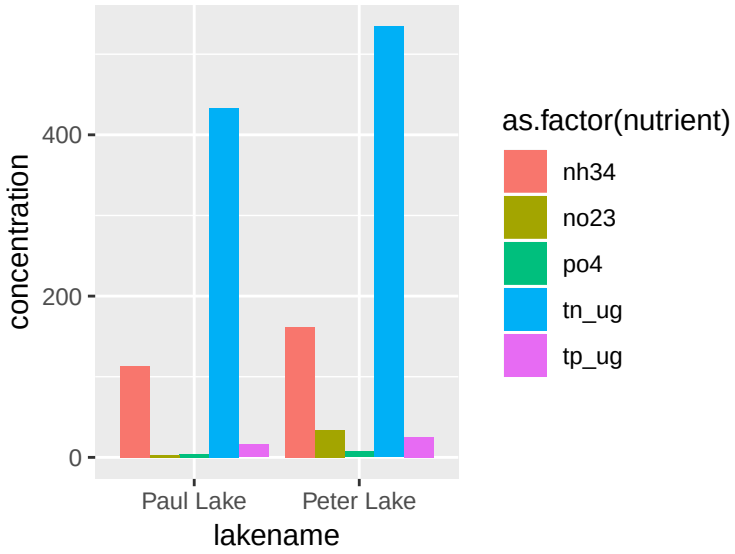



```r
# Exercise: Plot relationships between air quality measurements
```

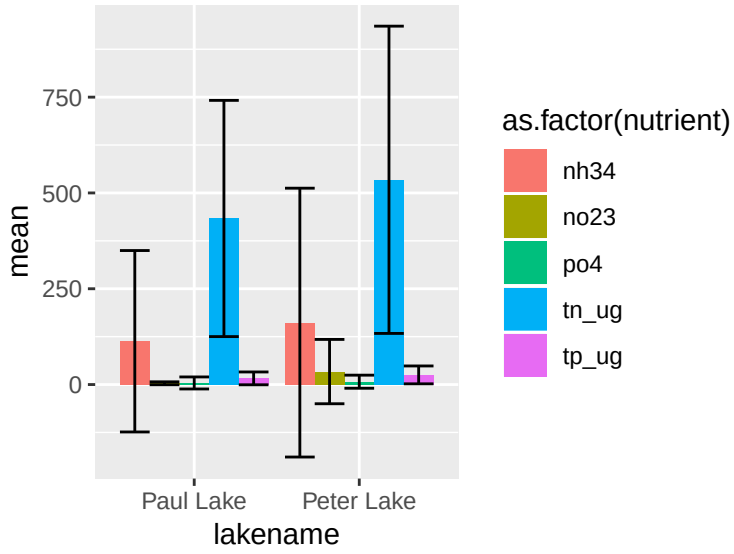
```
# 1.
# Plot AQI values for ozone by PM2.5, colored by site.
# Add a line of best fit for the linear regression of these variables.
```

Plotting continuous vs. categorical variables

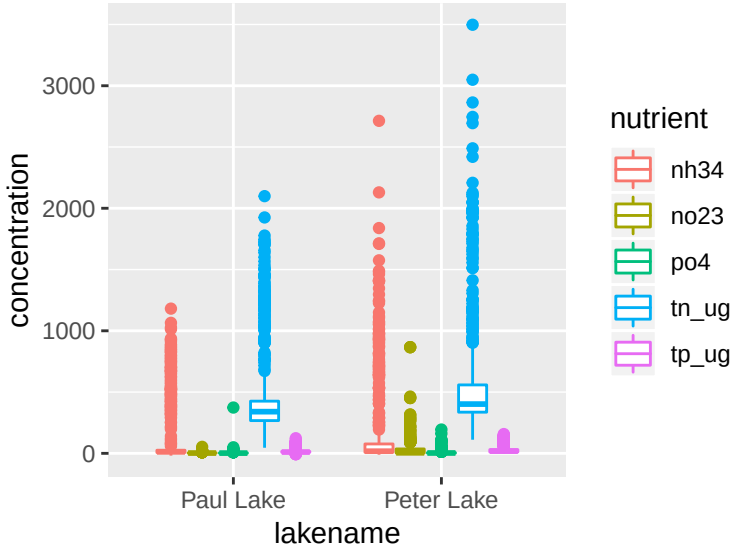
```
 # Barplot + error bars
PeterPaul.nutrient.summaries <- PeterPaul.nutrients.gathered %>%
  group_by(lakename, nutrient) %>%
  summarise(sd = sd(concentration),
            mean = mean(concentration))

Nutrientplot <-
  ggplot(PeterPaul.nutrients.gathered) +
  geom_bar(aes(x = lakename, y = concentration, fill = as.factor(nutrient)), # why did we use fill?
           position = "dodge", stat = "summary", fun.y = "mean")             # what's happening here?
print(Nutrientplot)
```



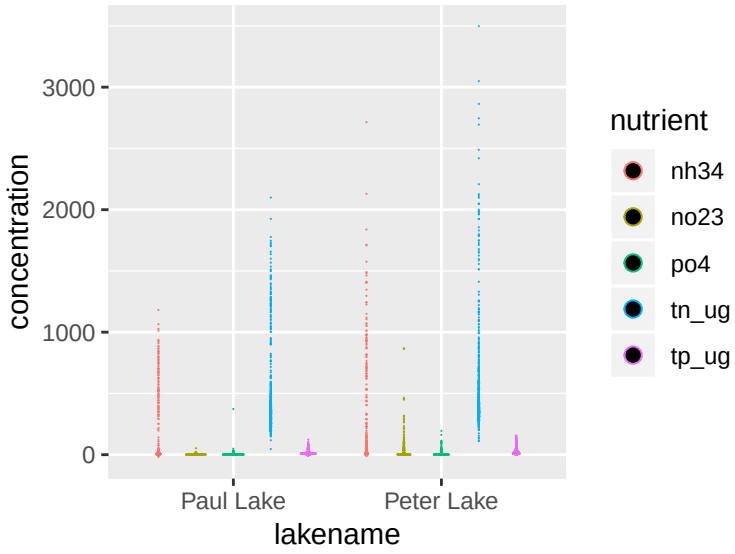

```
Nutrientplot2 <-
  ggplot(PeterPaul.nutrient.summaries, aes(x = lakename, y = mean, fill = as.factor(nutrient))) + #
  geom_bar(stat = "identity", position = "dodge") + # what does the stat command do?
  geom_errorbar(aes(ymin = mean-sd, ymax = mean+sd), # how do we specify error bars?
                position = "dodge")
print(Nutrientplot2)
```



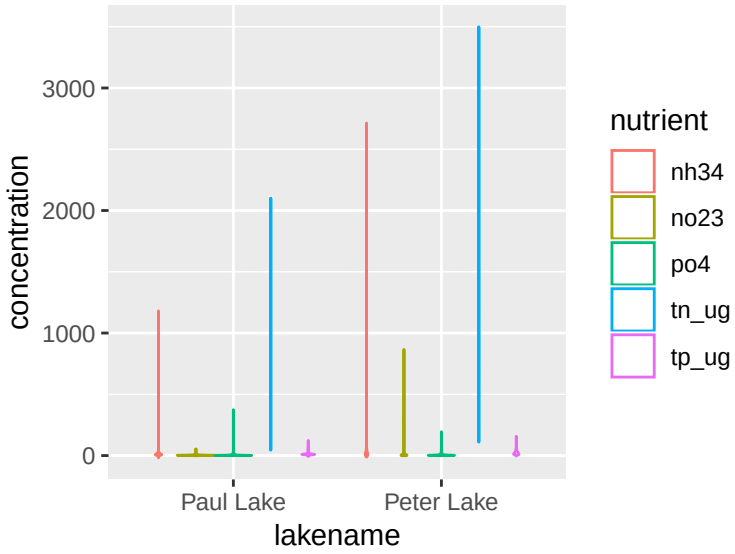

```
# Are there more effective ways to produce summary stats for categories?

# Box and whiskers plot
Nutrientplot3 <-
  ggplot(PeterPaul.nutrients.gathered, aes(x = lakename, y = concentration)) +
  geom_boxplot(aes(color = nutrient)) # Why didn't we use "fill"?
print(Nutrientplot3)
```




```
# Dot plot
Nutrientplot4 <-
  ggplot(PeterPaul.nutrients.gathered, aes(x = lakename, y = concentration)) +
  geom_dotplot(aes(color = nutrient), binaxis = "y", binwidth = 1,
               stackdir = "center", position = "dodge") #
print(Nutrientplot4)
```

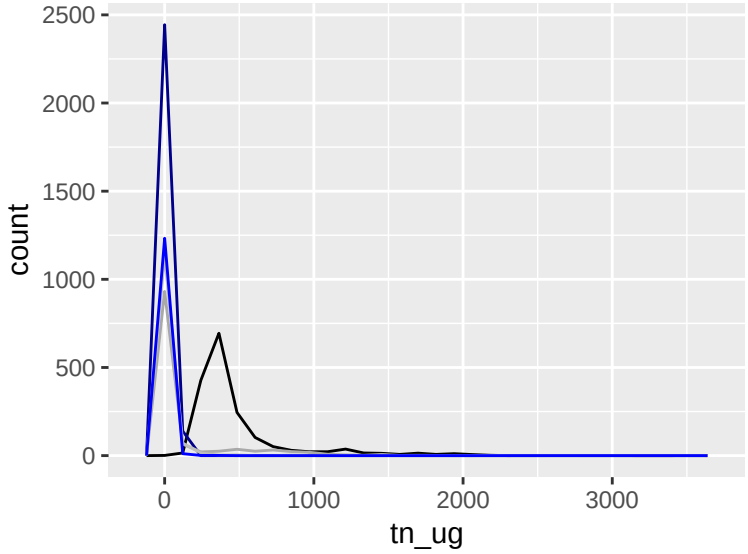



```
# Violin plot
Nutrientplot5 <-
  ggplot(PeterPaul.nutrients.gathered, aes(x = lakename, y = concentration)) +
  geom_violin(aes(color = nutrient)) #
print(Nutrientplot5)
```



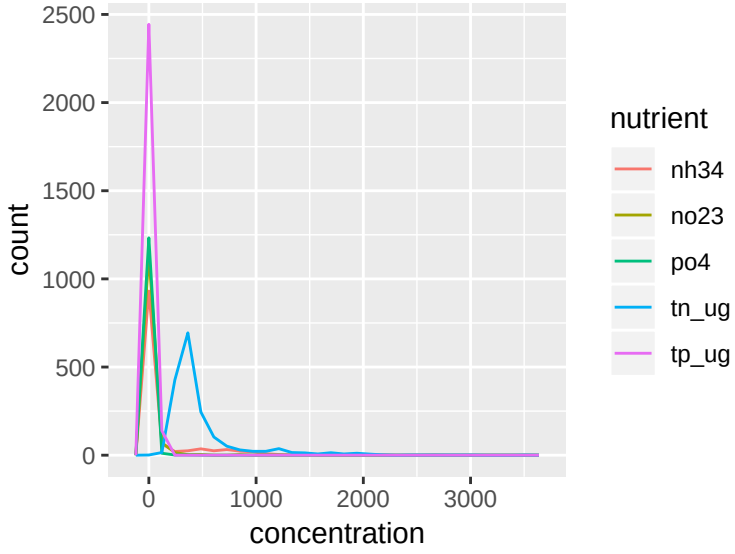

```
# Frequency polygons
# Using a tidy dataset
Nutrientplot6 <-
  ggplot(PeterPaul.chem.nutrients) +
  geom_freqpoly(aes(x = tn_ug), color = "black") +
  geom_freqpoly(aes(x = tp_ug), color = "darkblue") +
  geom_freqpoly(aes(x = nh34), color = "darkgray") +
  geom_freqpoly(aes(x = no23), color = "gray") +
  geom_freqpoly(aes(x = po4), color = "blue")
print(Nutrientplot6)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
# Using a gathered dataset
Nutrientplot7 <-
  ggplot(PeterPaul.nutrients.gathered) +
  geom_freqpoly(aes(x = concentration, color = nutrient))
print(Nutrientplot7)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
# Exercise: Plot distributions of AQI values for EPAair

# 1.
# Create a bar chart plus standard deviation error bars for PM2.5, divided by year.
```

```
# Create separate bars for each site.

# 2.
# Create a new plot that better depicts the distribution of PM2.5 concentrations.
# Divide your graph by year and site.
```