

plotly

The Johns Hopkins Data Science Lab

February 28, 2017

# What is Plotly?

Plotly is a web application for creating and sharing data visualizations. Plotly can work with several programming languages and applications including R, Python, and Microsoft Excel. We're going to concentrate on creating different graphs with Plotly and sharing those graphs.

# Installing Plotly

Installing Plotly is easy:

```
install.packages("plotly")  
library(plotly)
```

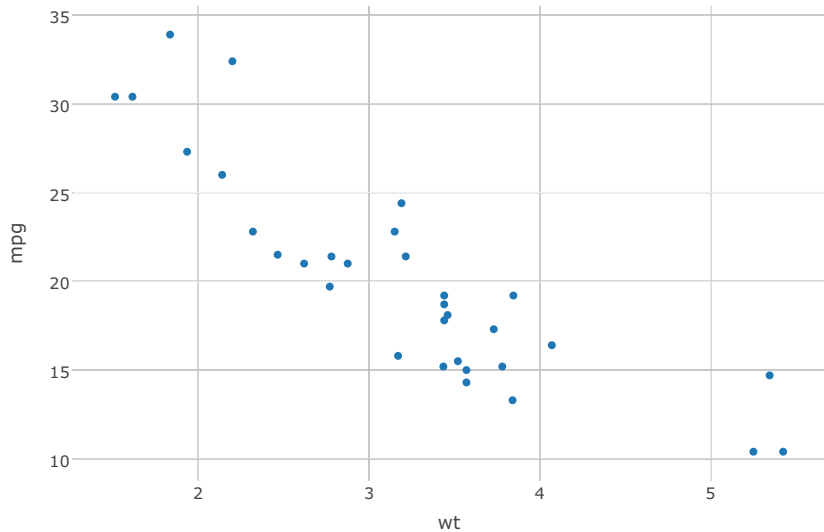
If you want to share your visualizations on <https://plot.ly/> you should make an account on their site.

## Basic Scatterplot

A basic scatterplot is easy to make, with the added benefit of tooltips that appear when your mouse hovers over each point. Specify a scatterplot by indicating `type = "scatter"`. Notice that the arguments for the x and y variables are specified as formulas, with the tilde operator (`~`) preceding the variable that you're plotting.

```
library(plotly)
plot_ly(mtcars, x = ~wt, y = ~mpg, type = "scatter")
```

# Basic Scatterplot

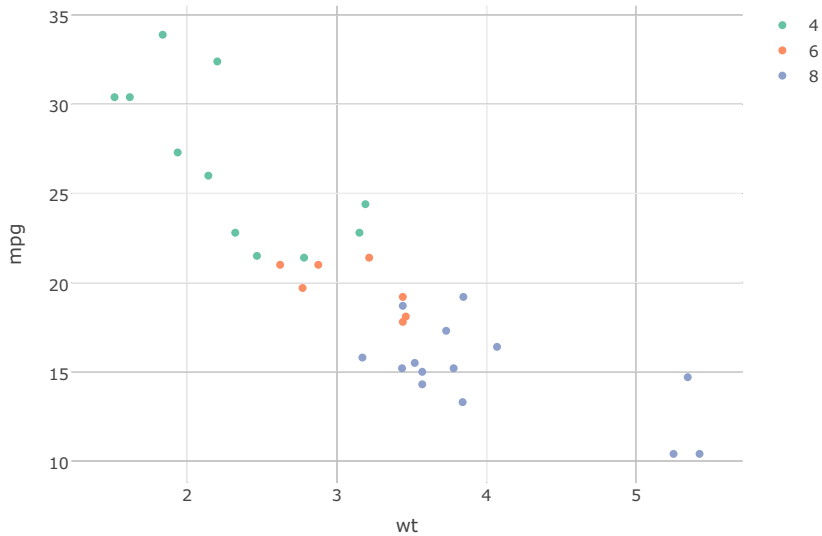


## Scatterplot Color

You can add color to your scatterplot points according to a categorical variable in the data frame you use with `plot_ly()`.

```
plot_ly(mtcars, x = ~wt, y = ~mpg, type = "scatter", color
```

# Scatterplot Color



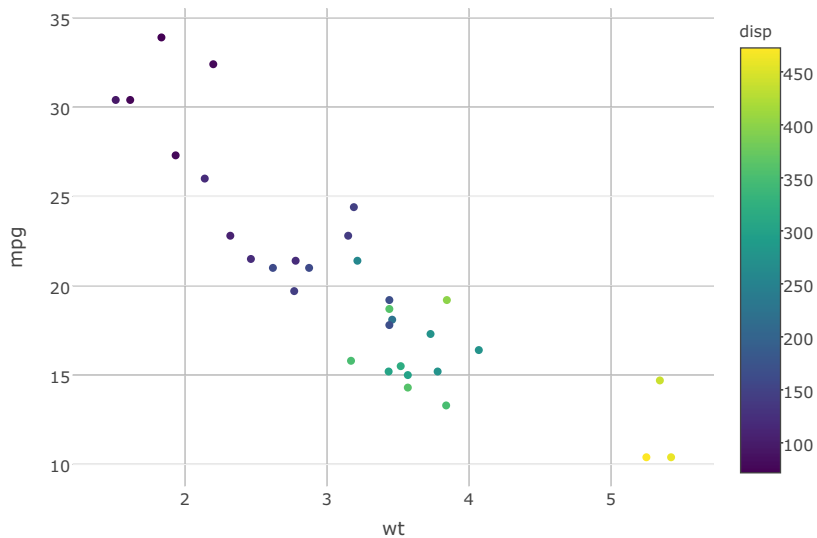
## Continuous Color

You can also show continuous variables with color in scatterplots.

```
plot_ly(mtcars, x = ~wt, y = ~mpg, type = "scatter", color
```



# Continuous Color

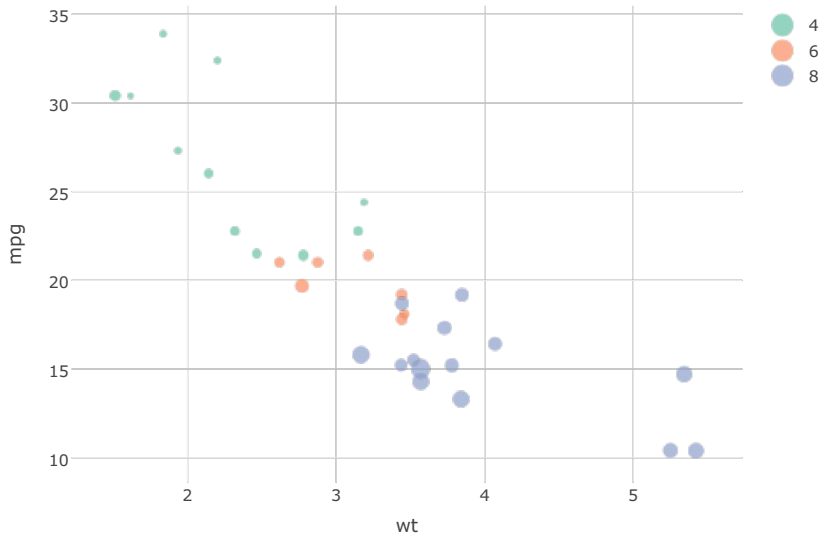


## Scatterplot Sizing

By specifying the size argument you can make each point in your scatterplot a different size.

```
plot_ly(mtcars, x = ~wt, y = ~mpg, type = "scatter",  
        color = ~factor(cyl), size = ~hp)
```

# Scatterplot Sizing



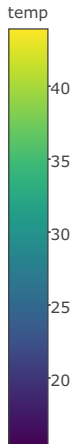
## 3D Scatterplot

You can create a three-dimensional scatterplot with the `type = "scatter3d"` argument. If you click and drag these scatterplots you can view them from different angles.

```
set.seed(2016-07-21)
temp <- rnorm(100, mean = 30, sd = 5)
pressue <- rnorm(100)
dtime <- 1:100
plot_ly(x = ~temp, y = ~pressue, z = ~dtime,
        type = "scatter3d", color = ~temp)
```

# 3D Scatterplot

Webgl is not supported by your browser - visit <http://get.webgl.org> for more info

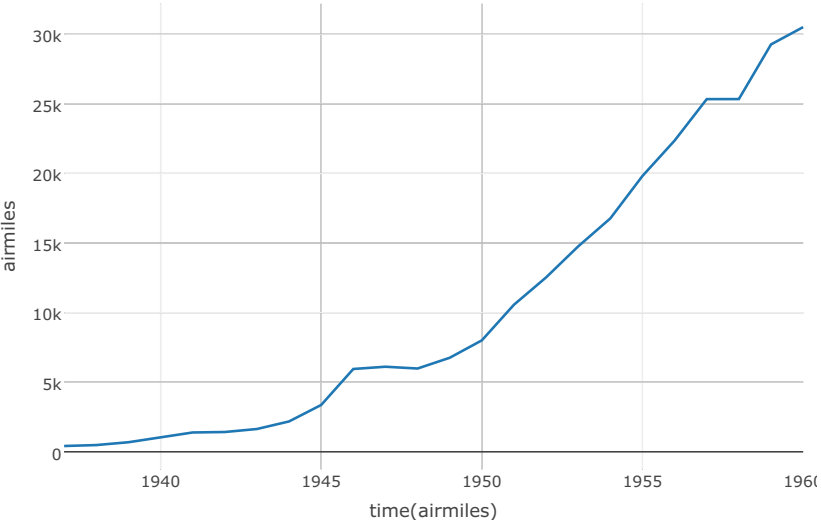


## Line Graph

Line graphs are the default graph for `plot_ly()`. They're of course useful for showing change over time:

```
data("airmiles")  
plot_ly(x = ~time(airmiles), y = ~airmiles, type = "scatter")
```

# Line Graph



## Multi Line Graph

You can show multiple lines by specifying the column in the data frame that separates the lines:

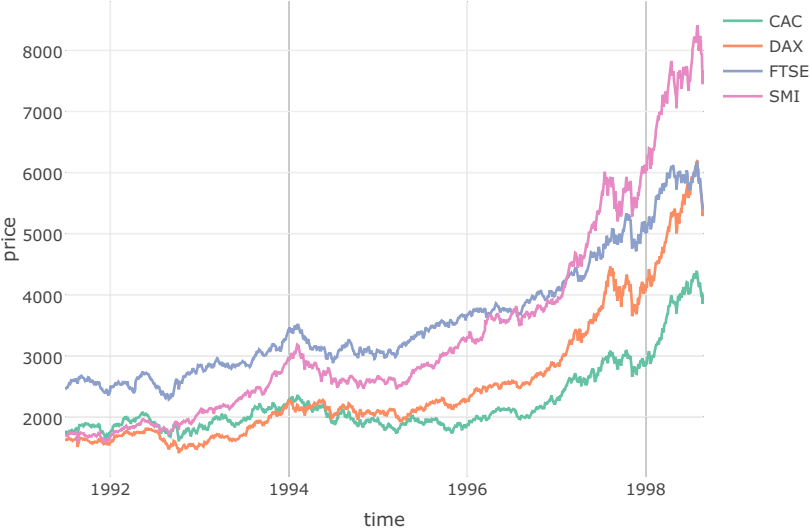
```
library(plotly)
library(tidyr)
library(dplyr)
data("EuStockMarkets")

stocks <- as.data.frame(EuStockMarkets) %>%
  gather(index, price) %>%
  mutate(time = rep(time(EuStockMarkets), 4))

plot_ly(stocks, x = ~time, y = ~price, color = ~index, type
```



# Multi Line Graph

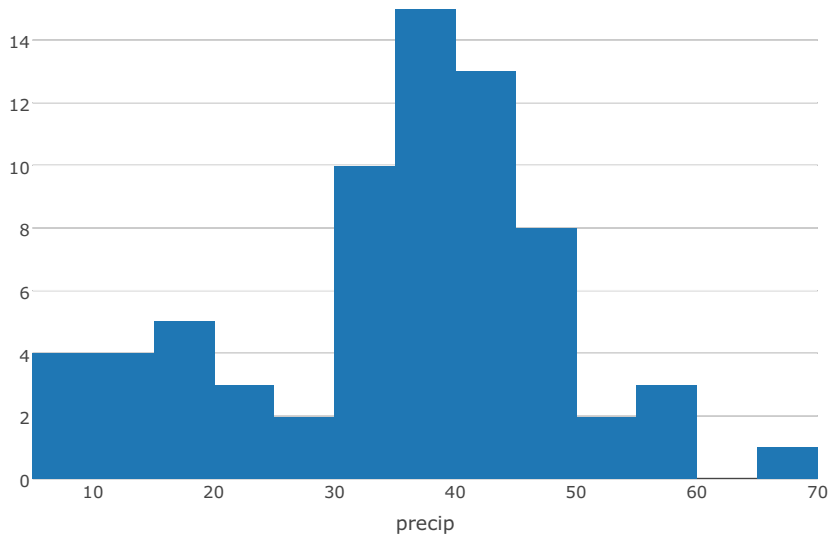


# Histogram

A histogram is great for showing how counts of data are distributed. Use the `type = "histogram"` argument.

```
plot_ly(x = ~precip, type = "histogram")
```

# Histogram

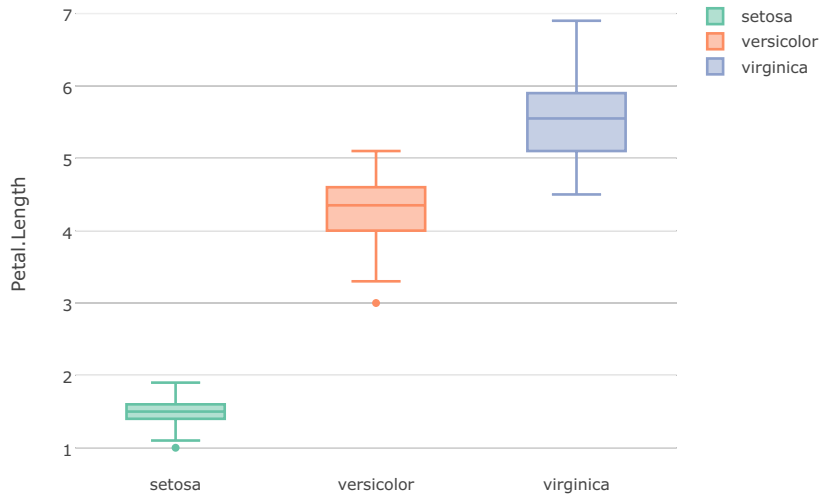


# Boxplot

Boxplots are wonderful for comparing how different datasets are distributed. Specify `type = "box"` to create a boxplot.

```
plot_ly(iris, y = ~Petal.Length, color = ~Species, type = 'box')
```

# Boxplot

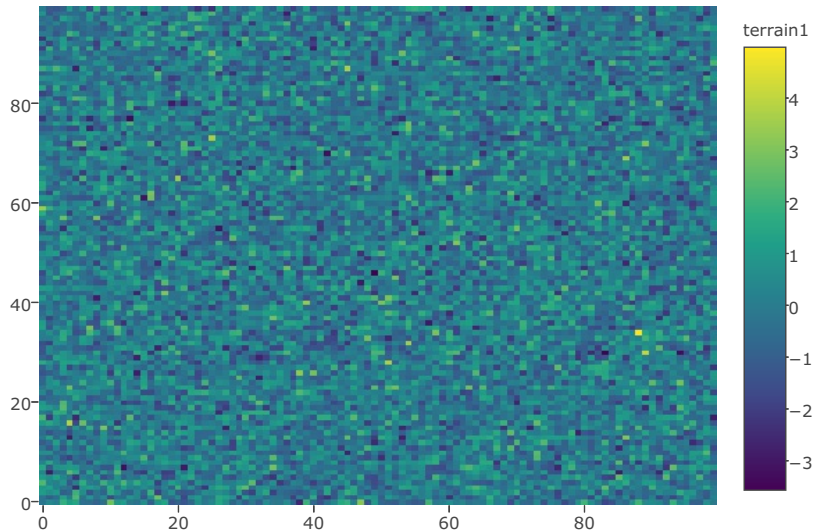


# Heatmap

Heatmaps are useful for displaying three dimensional data in two dimensions, using color for the third dimension. Create a heatmap by using the `type = "heatmap"` argument.

```
terrain1 <- matrix(rnorm(100*100), nrow = 100, ncol = 100)
plot_ly(z = ~terrain1, type = "heatmap")
```

# Heatmap



## 3D Surface

Why limit yourself to two dimensions when you can render three dimensions on a computer!? Create moveable 3D surfaces with `type = "surface"`.

```
terrain2 <- matrix(sort(rnorm(100*100))), nrow = 100, ncol = 100  
plot_ly(z = ~terrain2, type = "surface")
```



# 3D Surface

Webgl is not supported by your browser - visit <http://get.webgl.org> for more info



## Choropleth Maps: Setup

Choropleth maps illustrate data across geographic areas by shading regions with different colors. Choropleth maps are easy to make with Plotly though they require more setup compared to other Plotly graphics.

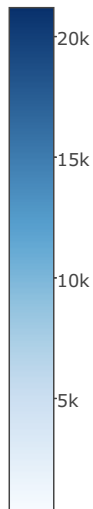
```
# Create data frame
state_pop <- data.frame(State = state.abb, Pop = as.vector(
# Create hover text
state_pop$hover <- with(state_pop, paste(State, '<br>', "Po
# Make state borders white
borders <- list(color = toRGB("red"))
# Set up some mapping options
map_options <- list(
  scope = 'usa',
  projection = list(type = 'albers usa'),
  showlakes = TRUE,
  lakecolor = toRGB('white')
)
```

## Choropleth Maps: Mapping

```
plot_ly(z = ~state_pop$Pop, text = ~state_pop$hover, locationmode = 'USA-states',  
        type = 'choropleth', locationmode = 'USA-states',  
        color = state_pop$Pop, colors = 'Blues', marker = 1)  
layout(title = 'US Population in 1975', geo = map_options)
```

# Choropleth Maps

US Population in 1975



## More Resources

- ▶ The Plotly Website
- ▶ The Plotly R API
- ▶ The Plotly R Package on GitHub
- ▶ The Plotly R Cheatsheet
- ▶ “Plotly for R” book by Carson Sievert