Post 02: Plotting with Plotly: Scatter Plot, Line Plot, and 3D Plots

Introduction

Besides the basic graphing functions in R, R users have developed many useful packages that expands and goes beyond the basic graphical techniques. Plotly is a widely used R package that creates interactive graphs.
Plotly was built using D3.js, which is a JavaScript library that can be used to create interactive plots. It is a great, simple way to visualize data, as compared to the alternative of learning D3.js. Not to mention, Plotly is compatible with not just R, but also Python, MATLAB, Pearl, Julia, and Arduino.
Plotly is a great choice for graphing simple functions, such as scatter plots and line plots, as well as complex 3D graphs. Regardless of what type of graph it is, plotly will render it so that users can easily interact with the graph and all of its functions.

First, load the package plotly.

```r
library(plotly)
```

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

```
## Loading required package: ggplot2
```

```
##
## Attaching package: 'plotly'
```

```
## The following object is masked from 'package:ggplot2':
##
##     last_plot
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following object is masked from 'package:graphics':
##
##     layout
```

Basic Plots

Let's take a look at what plotly can do! We'll start with some basic plots to work with.

Scatterplot

Using the data set, cars, we can produce a simple scatterplot of speed versus stopping distance in cars.

```r
data(cars) #loading data set
plot_ly(data=cars, x=~speed, y=~dist, type="scatter",
        mode="marker")
```

```
## A marker object has been specified, but markers is not in the mode
## Adding markers to the mode...
```

Using your cursor, you can move along the points and see their exact values. In addition to this useful interactive feature, users can also zoom in and zoom out of the graph, compare data on hover, download the photo as a png, and much more. These features can be found in the modebar at the top of the graph.

```r
#Adding details to the scatterplot

cars %>%
  plot_ly(x=~speed, y=~dist, type="scatter",
          color="orange", alpha=0.7) %>%
  layout(title="cars")
```

```
## No scatter mode specifed:
##   Setting the mode to markers
##   Read more about this attribute -> https://plot.ly/r/reference/#scatter-mode
```

```
## Warning in RColorBrewer::brewer.pal(N, "Set2"): minimal value for n is 3, returning requested palette with 3 di
fferent levels
```

You can also use pipe operators within plotly to add operators as we have done with ggplot and ggvis. In this example, we use the data set cars, which we pass into plot_ly(), which we pass into layout().
We can use various arguments in plot_ly, such as color, which specifies the color of the plot points, the alpha, which controls the opacity, and we can also use other options such as type, aymbols, size, and more. The layout() is also useful in labeling the title as well as the x and y axes.

More with Scatterplots

Plotly is also a great tool for data visualization. By using colors and labels, scatterplots made through plotly can provide visually appealing graphs that cleanly represent the data in an easy to understand manner.

```r
#The Iris data set can be organized by species

data(iris)
p <- plot_ly(data=iris, x=~Sepal.Length, y=~Petal.Length,
        color=~Species,
        colors = c("red", "blue", "purple"),
        alpha=0.8)

#Properly lapeling plot
p %>% layout(title ="Iris", xaxis=list(title="Sepal Length"), yaxis=list(title="Petal Length"))
```

```
## No trace type specified:
##   Based on info supplied, a 'scatter' trace seems appropriate.
##   Read more about this trace type -> https://plot.ly/r/reference/#scatter
```

```
## No scatter mode specifed:
##   Setting the mode to markers
##   Read more about this attribute -> https://plot.ly/r/reference/#scatter-mode
```

```
#Adding Hover text to label plot points
plot_ly(trees, x=~Girth, y=~Height, type= "scatter",
        text= ~paste("Volume: ", Volume), #Hover text
        color=~Volume)
```

```
## No scatter mode specifed:
##   Setting the mode to markers
##   Read more about this attribute -> https://plot.ly/r/reference/#scatter-mode
```

Using arguments such as text, we can add hover text. In the examples with the trees data, we can visuzlize the volume using the color scale, and we can also learn the exact volume by hovering over the point.

Line Graphs

Another basic graph plotly can produce is line graphs. There are many different ways to graph the lines using the mode attribute. For example, you can make the lines smooth or you can include the data points as well by using the "lines+markers" options for mode.

```
#nhtemp = average yearly temp in New Haven
plot_ly(x=1912:1971, y=nhtemp, mode="lines")
```

```
## No trace type specified:
##   Based on info supplied, a 'scatter' trace seems appropriate.
##   Read more about this trace type -> https://plot.ly/r/reference/#scatter
```

```
#Different options for "mode"
plot_ly(x=1912:1971, y=nhtemp, mode="lines+markers")
```

```
## No trace type specified:
##   Based on info supplied, a 'scatter' trace seems appropriate.
##   Read more about this trace type -> https://plot.ly/r/reference/#scatter
```

More graphs: 3 Dimensional

In addition to the basic plots, such as scatterplots, line plots, bar charts, and histograms, plotly is great for visualizing 3D data as well.
The interactive portion of the data visualization allows us to locate points with the helper lines and see the 3D plots in different perspectives by changing the angle from which we view the data. The modebar at the top provides different options for users to visualize the data. These options include orbital rotation (rotates around moddle point), and turntable rotation (rotates around middle while slightly constraining one axis). If users want to undo any changes to the perspective, they can reset the camera to default.

```
#Use "type" to change the type of plot

#3D Line Plots
plot_ly(type= "scatter3d", x = c(7, 8, 3, 2),
        y=c(1, 4, 3, 9),
        z= c(11, 6, 15, 3),
        mode = 'lines')
```

```r
#3D Scatter Plots
plot_ly(type= "scatter3d", x = c(7, 8, 3, 2),
        y=c(1, 4, 3, 9),
        z= c(11, 6, 15, 3),
        mode = 'markers')
```

```r
#Using commonly used data set "volcano" for 3D graphs
#volcano has topographical info on the Maunga Whau Volcano
#Heatmaps
plot_ly(z=volcano, type='heatmap')
```

```r
#3D Surface Plots
plot_ly(type="surface", z=~volcano)
```

Conclusion/Take Away

With the countless of packages in R, there is a plethora of graphical tool options for R users. Plotly is one such tool that can be great for producing visually pleasing graphs that can contain a lot of the useful information from the data in one easy to view source. Plotly’s interactive functions makes it easy for viewers to process the information as well. While different data sets require different modes of visualization, Plotly is a great graphical tool for creating complex and detailed, yet easy to view graphs, in a simple manner without the complexities of having to learn D3.js.

References

1. plot.ly line and scatter
2. R bloggers, Pipe operator
3. Plotly cheat sheet
4. Analytics Vidhya
5. R Data Set
6. R Bloggers
7. R Visualization
8. Plotly Modebar