Exploring ggplot2

A post by John Karis, johntkaris@berkeley.edu

Introduction

In R programming, ggplot2 is an invaluable tool that can be used in the field of statistics and data science. It facilitates data visualization, and allows for data analysis to easily take place. In this post I am going to explain what I discovered while exploring ggplot2 and try to give some insight to some of the really interesting functions that were not discussed in my Statistics 133 class. Specifically, I am going to deal with functions that involve scatterplots and how to more easily analyze densities in these plots.

Review

First, I have to load ggplot2 (obviously) and a data set to the current session in order to show some examples of how to use functions. I am also downloading the gridExtra package to help with some visualizations.

```
library(ggplot2)
library(readr)
library(gridExtra)
```

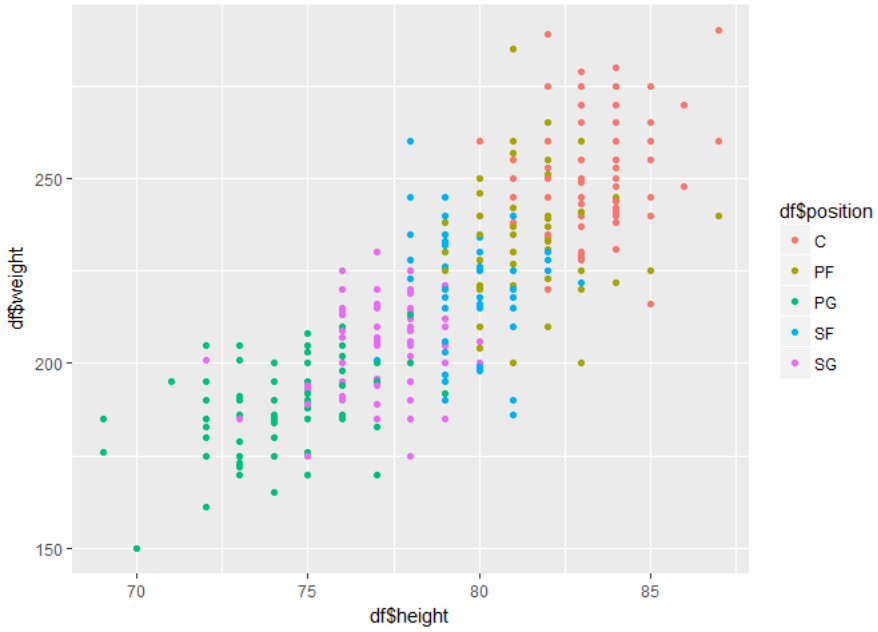
```
## Warning: package 'gridExtra' was built under R version 3.4.2
```

```
df = read_csv('nba2017-players.csv')
```

```
## Parsed with column specification:
## cols(
##   player = col_character(),
##   team = col_character(),
##   position = col_character(),
##   height = col_integer(),
##   weight = col_integer(),
##   age = col_integer(),
##   experience = col_integer(),
##   college = col_character(),
##   salary = col_double(),
##   games = col_integer(),
##   minutes = col_integer(),
##   points = col_integer(),
##   points3 = col_integer(),
##   points2 = col_integer(),
##   points1 = col_integer()
## )
```

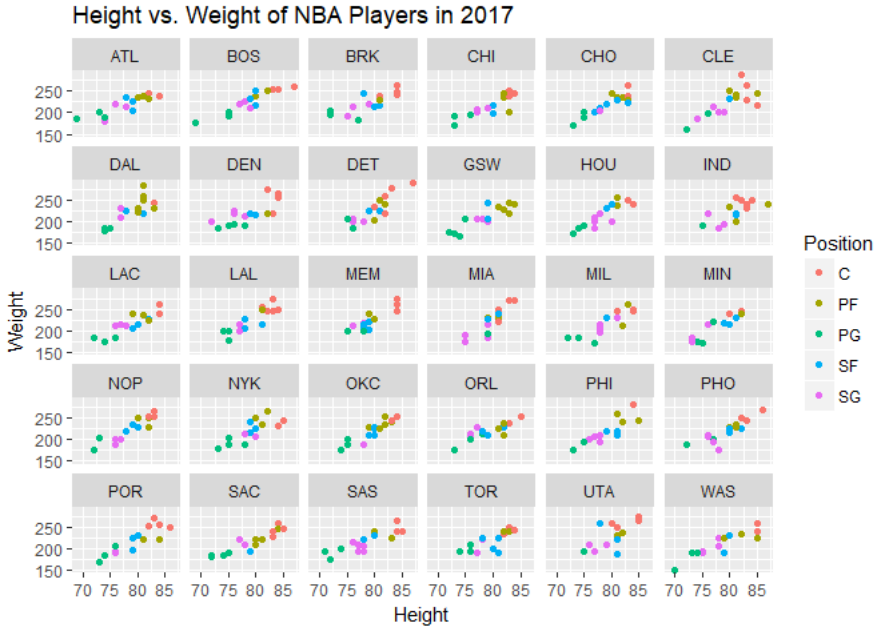
Next, I am going to do a quick run through most of the functions that have been dicussed and used in the class, just to remind people of how ggplot2 works and to review some basics. For the entirety of this post, I will be using a data set containing the information for players in the NBA in 2017. ggplot2 requires and aesthetic or aes() to determine its arguments and how to represent them. For these brief review examples, I am graphing a scatterplot of height versus weight of players, denoting each player's position with a different color. Remember that in order to plot points, '+ geom_point()' is required.

```
ggplot(df, aes(x = df$height, y = df$weight, color = df$position)) +
  geom_point()
```



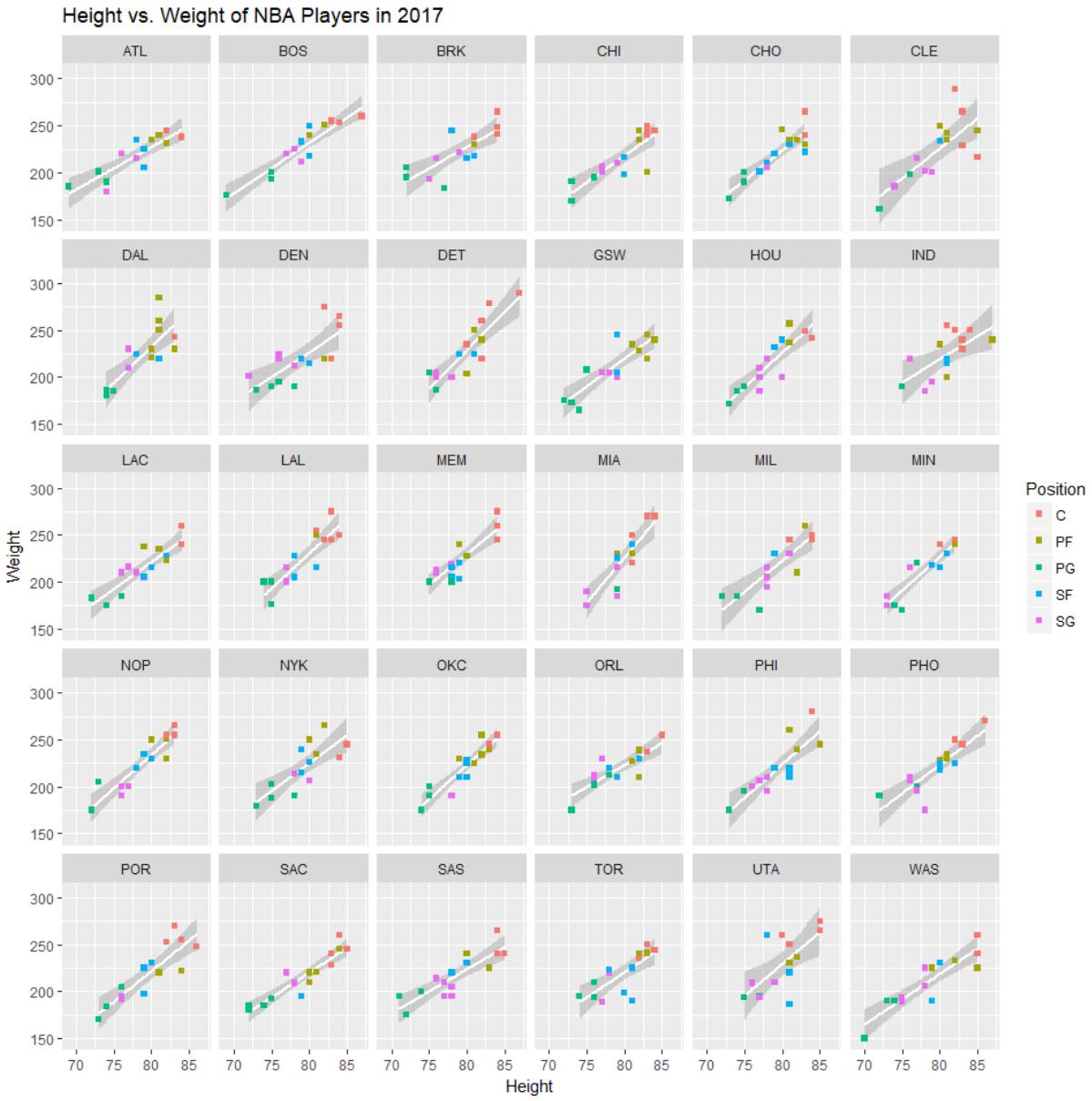
That wasn't so bad for a graph that is reasonable complicated. That being said, let's add a few more things to make the plot more interesting and clean it up a bit. Remember that there must be a '+' in between each function in order to run them all on the same plot.

```
ggplot(df, aes(x = df$height, y = df$weight, color = df$position)) +
  #I am going to facet the plot by teams so we can see each team's height-weight plot
  facet_wrap(~ df$team) +
  geom_point() +
  #let'd add a title and some better looking axis labels
  ggtitle('Height vs. Weight of NBA Players in 2017') +
  labs(x = 'Height', y = 'Weight') +
  #let's also give the key a cleaner name so its easier to read and understand
  scale_color_discrete(name = 'Position')
```



Last, just for fun, lets make the graph look extra interesting by making the points different shapes and adding a smooth curve to show the regression line for each team. To make each graph easier to see, I will use 'geom_point()' after 'geom_smooth()' so the points appear on top of the line and its error area. I am also going to use 'fig.height=10' and 'fig.width=10' in the code chunk of the Rmd file to make the graph bigger and easier to read.

```
ggplot(df, aes(x = df$height, y = df$weight, color = df$position)) +
  facet_wrap(~ df$team) +
  #this next line of code plots a white regression line with an error region around it
  geom_smooth(method = 'lm', color = 'white', formula = y~x) +
  #you change the points different shapes by defining 'shape' inside 'geom_point()'. Different shapes have differe
nt code numbers
  geom_point(shape = 15) +
  ggtitle('Height vs. Weight of NBA Players in 2017') +
  labs(x = 'Height', y = 'Weight') +
  scale_color_discrete(name = 'Position')
```



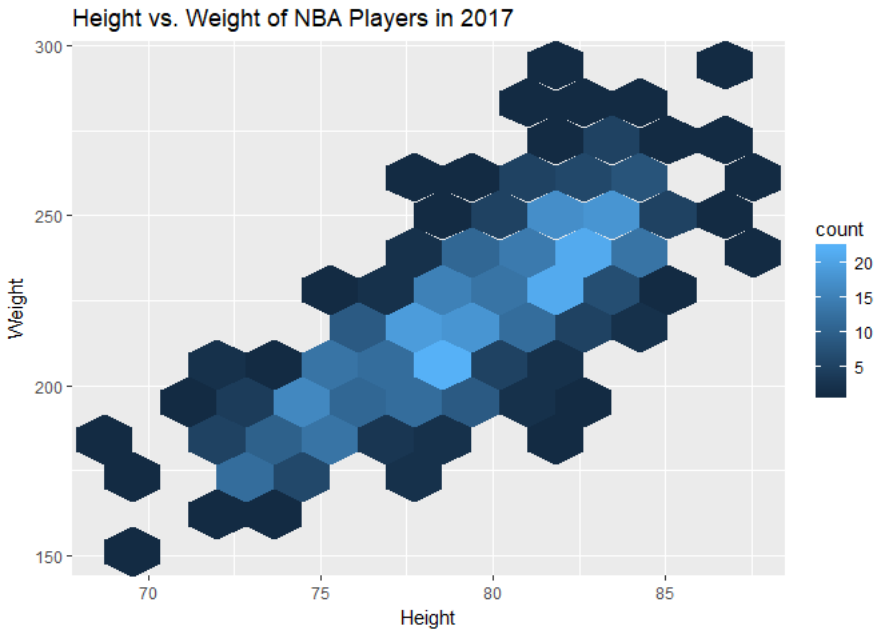
New Stuff

Hexplot

Now that all the basic stuff is over, let get into some new stuff that exists in the large world of ggplot2. One of my favorite functions is 'geom_hex()', because it looks like honeycomb and its default is in my favorite color of blue. A hexplot is a type of density plot that shows the count of points that exist in a hexagon gridded area of a scatterplot, and they often more easily show density than normal scatterplots. This description may seem a little wordy, and I think that simply looking at an example of it is far better at explaining its function. To make a hexplot, all you have to do is set up the ggplot like a scatterplot, but use 'geom_hex()' instead of 'geom_point()'. For this example, I set the bins to 11 to better show the data.

```
ggplot(df, aes(x = df$height, y = df$weight)) +
  #notice that you set the labels in the same way for a hexplot as a scatterplot as well
  ggtitle('Height vs. Weight of NBA Players in 2017') +
  labs(x = 'Height', y = 'Weight') +
  geom_hex(bins = 11)
```

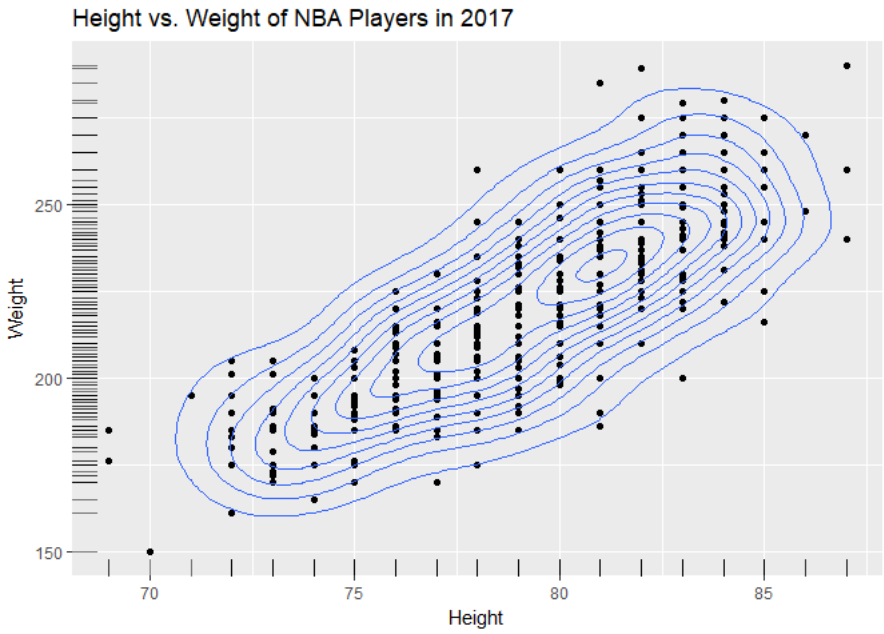
```
## Warning: package 'hexbin' was built under R version 3.4.2
```



2D Density

The next useful graphing tool that ggplot2 has to offer is two dimensional density plots. Another one of my favrite tools that we never got to learn about, 2D density plots shows density in a scatterplot by drawing areas of differing densities. This allows statisticians and data analysts to easily see the center of data on a 2 variable scatterplot. In the eample below, it is easy to see that the center of both distributions is at about 81 inches tall and between 230 and 235 pounds. Moreover, this function is super easy to use because all I have to do is add '+ geom_density_2d' to the end of the ggplot code.

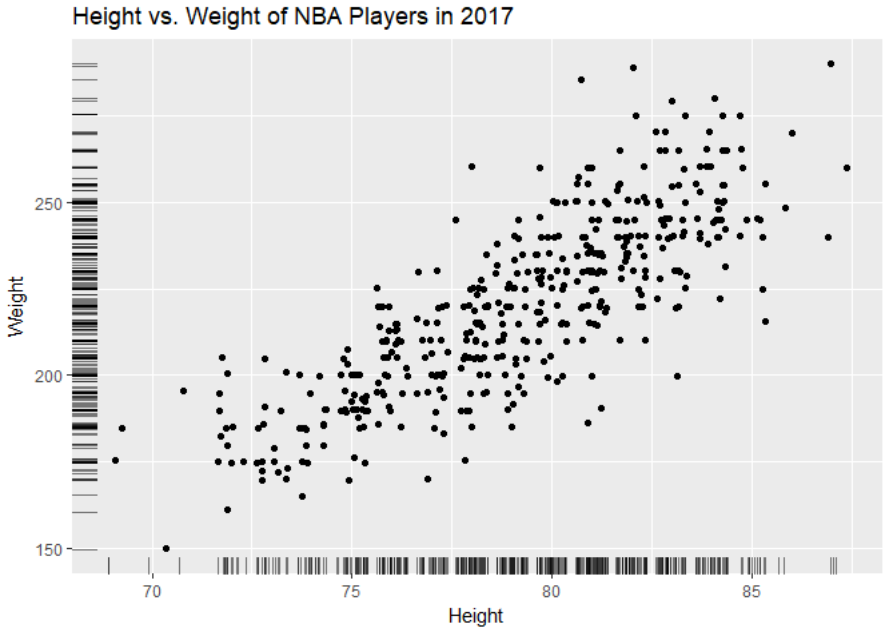
```
ggplot(df, aes(x = df$height, y = df$weight)) +
  ggtitle('Height vs. Weight of NBA Players in 2017') +
  labs(x = 'Height', y = 'Weight') +
  #Ignore this next line of code for now. It will be used in the next part.
  geom_rug(alpha = .5) +
  #I decided to add the points for this data to better understand the density plot
  geom_point() +
  geom_density_2d()
```



Jitter

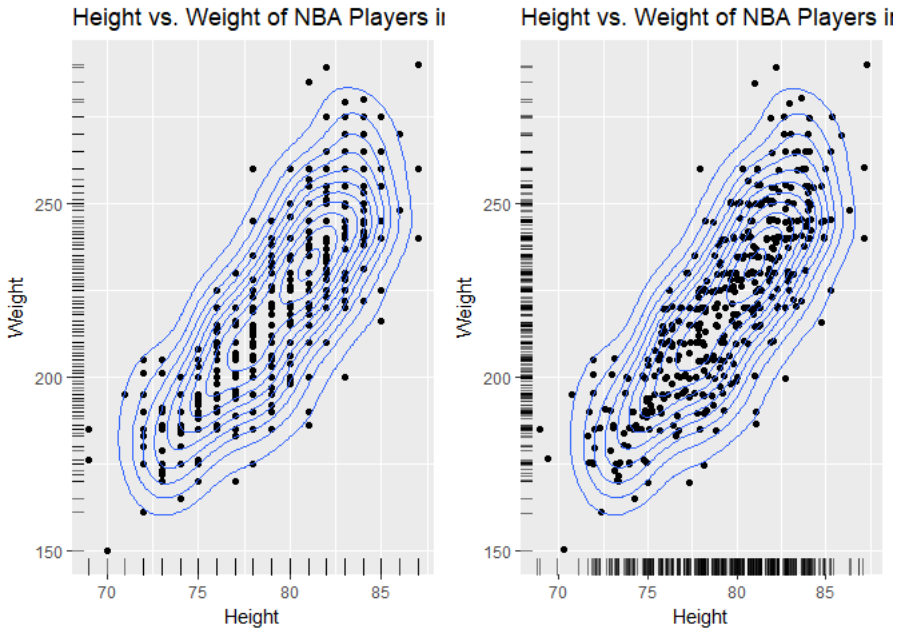
Another tool is the the jitter function. It may seem strange and borderline useless, but the jitter function actually has some usful properties. Jitter adds some variation to points on a scatterplot, and helps show density for points that are close to the same value. In the height versus weight scatterplot in the last plot, many of the points are close to each other, and it is hard to tell just how many there are in a single height value. Jitter allows us to more easily see the densities of the points. I used the 'geom_rug()' function on this plot and the last one to show the variation change, especially in the height values, since they were integers.

```
ggplot(df, aes(x = df$height, y = df$weight)) +
  ggtitle('Height vs. Weight of NBA Players in 2017') +
  labs(x = 'Height', y = 'Weight') +
  #it is possible to set the amount of variation in 'geom_jitter()' but I did not change it for this example
  geom_jitter() +
  geom_rug(alpha = .5, position = 'jitter')
```



Now, you may be thinking, after using the jitter function and adding all this variation to the plot, doesn't that make the plot less accurate? Well, literally yes, it does. However, the variation added is exteremely small. In the big picture of things, the distribution will remain relatively similar. I plotted the density function we used before onto a jitter plot and placed it next to the exact plot from the geom_density_2d plot to prove that the overall distribution remained almost the same.

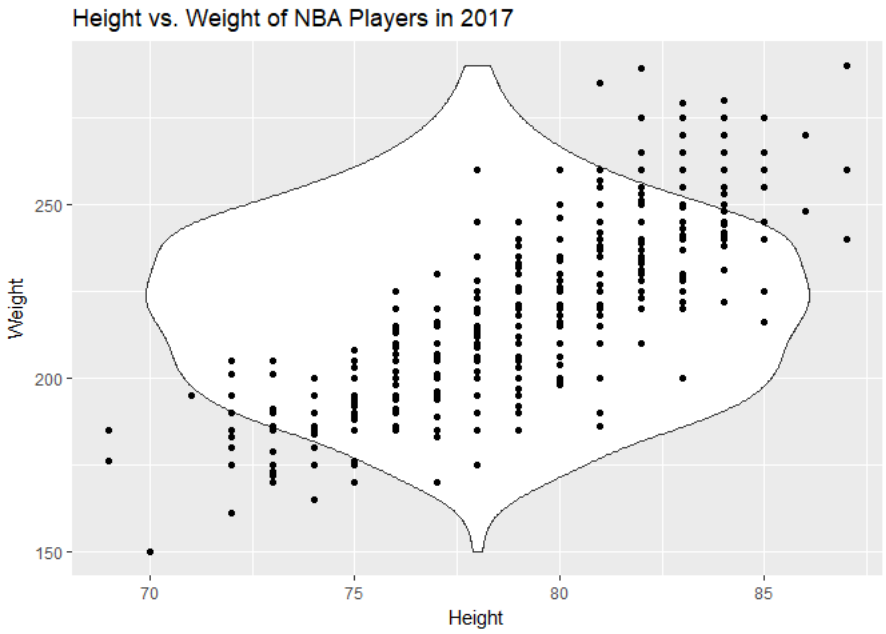
```
area_density_plot = ggplot(df, aes(x = df$height, y = df$weight)) +
  ggtitle('Height vs. Weight of NBA Players in 2017') +
  labs(x = 'Height', y = 'Weight') +
  #Ignore this next line of code for now. It will be used in the next part.
  geom_rug(alpha = .5) +
  #I decided to add the points for this data to better understand the density plot
  geom_point() +
  geom_density_2d()

jitterplot = ggplot(df, aes(x = df$height, y = df$weight)) +
  ggtitle('Height vs. Weight of NBA Players in 2017') +
  labs(x = 'Height', y = 'Weight') +
  geom_jitter() +
  geom_rug(alpha = .5, position = 'jitter') +
  geom_density_2d()

grid.arrange(area_density_plot, jitterplot, ncol=2)
```



Violin

There is one final tool to show density in graphs that I explored. The violin (not the instrument) function shows the density of points of specific y-axis values with a symmetrical shape that gets longer the more points fall in that value. The violin function is a great way of seeing spread in data because it is often difficult to determine distributions only with points.

```
ggplot(df, aes(x = df$height, y = df$weight)) +
  ggtitle('Height vs. Weight of NBA Players in 2017') +
  labs(x = 'Height', y = 'Weight') +
  geom_violin() +
  #I am going to add the points on top in order to more easily visualize what this function does
  geom_point()
```



Although we learned about a great many different kinds of graphing techniques in my stat 133 class, there were quite a few functions that we did not get to cover. All of the functions I went over have great uses for those interested in data visialization, and I hope I brought to light some of the more interesting functions in ggplot2.

References

1. https://www.statmethods.net/advgraphs/ggplot2.html
2. http://ggplot2.tidyverse.org/index.html
3. http://www.sthda.com/english/wiki/ggplot2-point-shapes
4. https://www.r-bloggers.com/how-to-format-your-chart-and-axis-titles-in-ggplot2/
5. https://www.r-bloggers.com/15-questions-all-r-users-have-about-plots/
6. https://sebastiansauer.github.io/figure_sizing_knitr/
7. https://stackoverflow.com/questions/19699858/ggplot-adding-regression-line-equation-and-r2-with-facet
8. http://www.sthda.com/english/wiki/ggplot2-stripchart-jitter-quick-start-guide-r-software-and-data-visualization
9. https://www.r-statistics.com/2016/07/using-2d-contour-plots-within-ggplot2-to-visualize-relationships-between-three-variables/
10. http://ggplot2.tidyverse.org/reference/geom_density_2d.html
11. https://stackoverflow.com/questions/17547699/explanation-for-jitter-function-in-r