Post1-heatmaps

Anant Kota

10/31/2017

Preliminary Packages

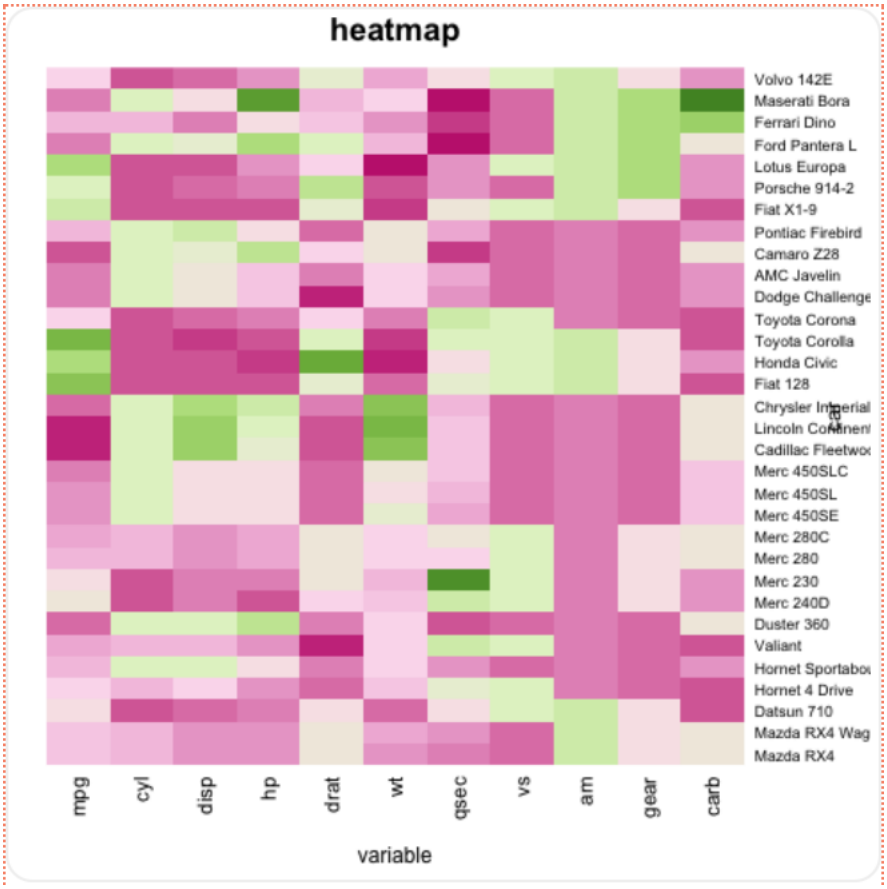
Before reading the post, please install and/or load the following packages as well as download the following data and set working directory to the Post1-heatmaps directory

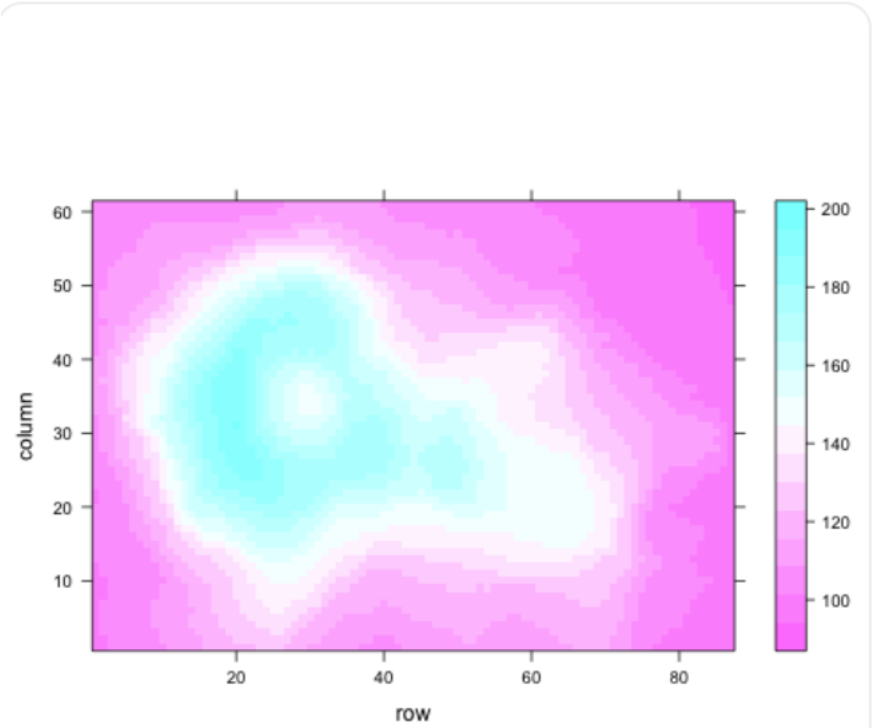
```
library(readr)
library(stargazer)
library(ggplot2)
library(dplyr)
library(gplots)
library(ggmap)

dat <- read_csv('data/nba2017-players.csv')
```

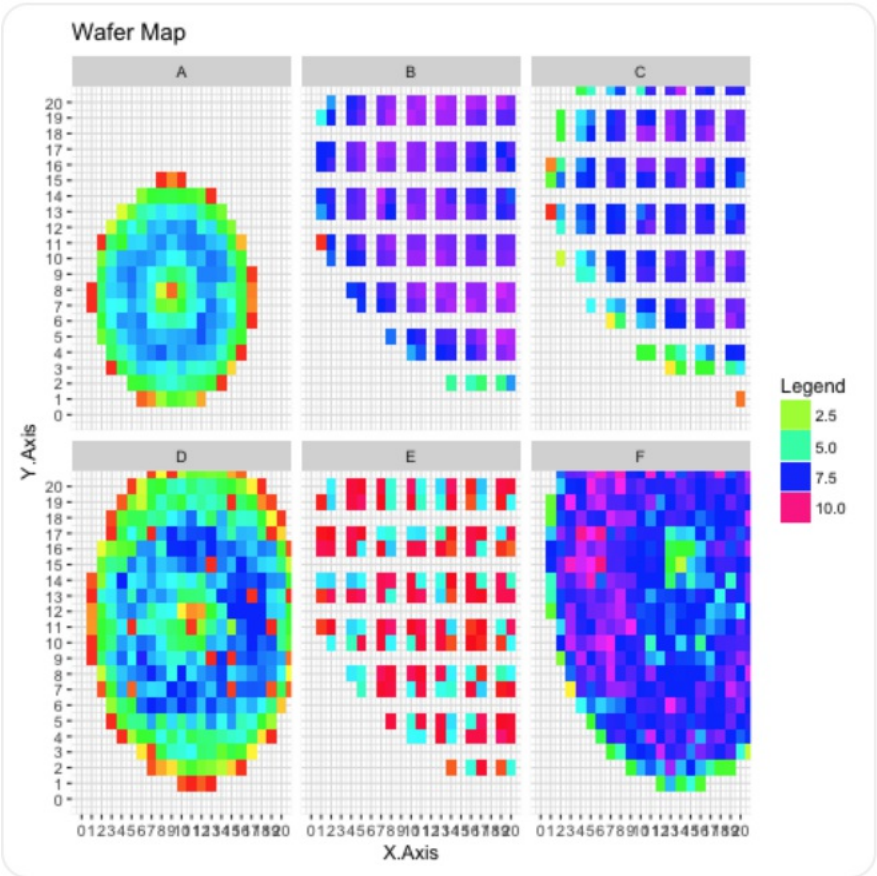
Heat Maps

One of the central uses of data visualization is to articulate some trend, characteristics, or other insight that a particular data set offers. Whther in journalism, professional reports, or scientific papers, the *clear* and *efficient* communication of statistical analysis is just as important as the analysis itself. Therefore, its important to choose the appropriate method of data visualization to convey the message you are trying to send with your data analysis.

Heatmaps have uses in various contexts as they also come in all shapes and sizes as the following examples show. As we progress through the different layers and implementations of heatmaps, the different uses will become apparent. First, lets discuss how to make the most basic of heat maps.







Basic Heatmap

Pretend you are a journalist or a scientific reporter of somekind and before conducting any analysis, you want the reader to notice some initial characteristic of a data set. Simply offering a table of numbers and asking the reading to notice a trend may not be the most efficient manner of comminicating this characteristic. For example, as a journalist, your table will be in the middle of an article, or, as a scientist, it may be cumbersome for the reader to take the time to look at each number in the table and metnally track what seems to be occuring. This is a scenario where heatmaps can be especially useful. Heatmaps essentially assign either different colors or different shades of one color to various values based on magnitude, concentration, proximity, etc. depending on the data. According to the National Institute of Health, humans are inherently visual learner with colors being the most potent and important visual experience for humans. Colors increase not only the speed at which we digest information, but also our attention span and memory performance! With such a powerful known trait of human psychology as it relates to information communication, it makes it all the more important that color is included in our data visualization.

To construct our first heatmap, we will be using the the nba player data from basketball-reference.com. We will want to create a heatmap that assigned colors of a darker shade for those number that are higher as compared to other values within a column. For example, players with higher salaries should have darker shades of a color compared to players with lower salaries. But first some cleaning to make the heatmap neater.

```
# Sort nba player data in ascending order for salary
dat <- arrange(dat, salary)
```

```
## Warning: package 'bindrcpp' was built under R version 3.3.2
```

For the sake of conciseness, we will limit the data frame to the 50 highest salaries,

```
dat <- dat[(nrow(dat)-49):nrow(dat), ]
```

Furthermore, because the base function *heatmap* needs the data in the form of a data matrix which is atomic, we need to get rid of character vectors within our data frame.

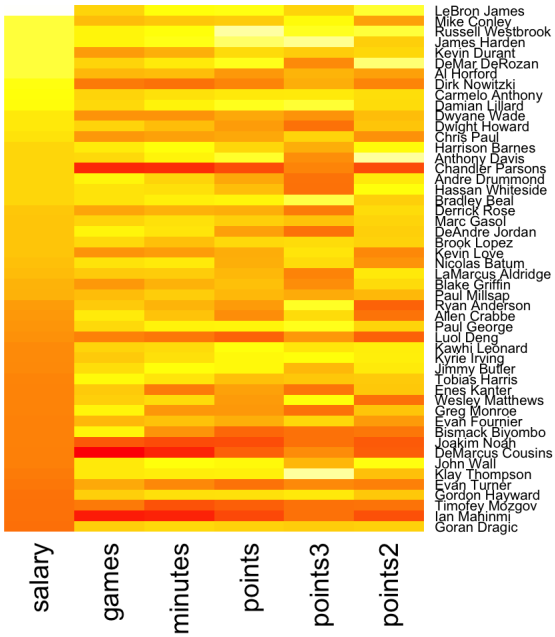
```
# Eliminate character vectors and non-performance stats
nba <- dat[,9:14]

# Name rows after players
row.names(nba) <- dat$player
```

Finally, the base function *heatmap* requires that the data be in the form of a matrix. Therefore, we will uses data.matrix to convert *nba* to a numeric matrix.

```
nba <- data.matrix(nba)
```

Alright! Now we're set to start creating our first heatmap. The first argument should be the matrix containing the data. The Colv and Rowv arguments produce a tree dendogram which is irrelevant to the discussion at hand. Providing NA as an argument eliminates this from being shown. Finally, the scale argument tells the function to scale the colors according the column that it is contained in. the "col" argument provided the color scheme that the heatmap will utilize. The scale argument tells the function which direction the tiles are going to be scaled ('row', 'column', or 'none'). Since our variables are columns, we'll plug 'column' for the scale argument. *Note: Try changing scale to 'row' and 'column' and see what happens to the corresponding heatmap.*

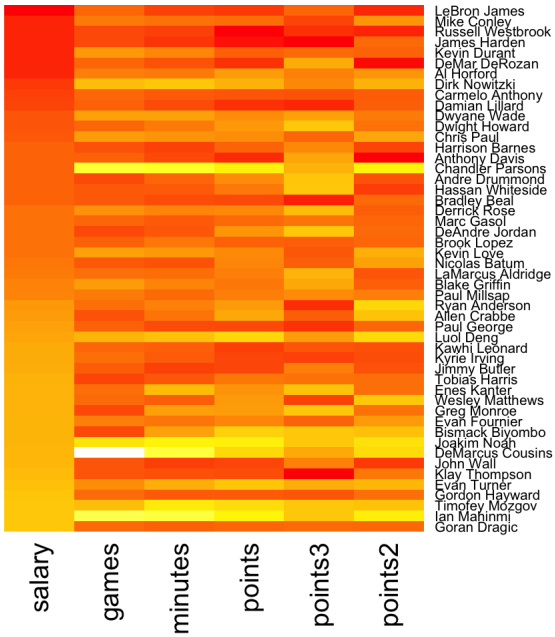
```
nba_heatmap <- heatmap(nba, Colv = NA, Rowv = NA ,scale = 'column', col = heat.colors(256), margins = c(5,10))
```




```
nba_heatmap
```

If you want larger values represented by darker colors simply replace the argument for col as such:

```
nba_heatmap <- heatmap(nba, Colv = NA, Rowv = NA ,scale = 'column', col = rev(heat.colors(256)), margins = c(5,10)
)
```




```
nba_heatmap
```

Now take a look at the resulting heatmap. Remember earlier how we ordered the data by salary (with Lebron being the highest player). Therefore, according to the heatmap, the lighter the color the larger it is within the context of the column that it is in. Now by quickly scanning the heatmap, one can see the relationships that exist between the columns. For example, look at Chandler Parsons. While his salary tile is a lighter shade, his tiles in the rest of the performance based columns (games, point, points2, points3, minutes) are consistently dark red, signaling that he has some of the lowest statistics among the players listed which informs the reader that he may not be living up to the salary he is paid. On the other hand, Lebron, who has the lightest shade salary tile, also has lighter shade tiles in the rest of the categories, delineating that although he is paid the most, he is also performing at higher level as well. Finally, Klay Thompson, who has one of the darkest salary tiles, has extremely lighter shade performance tiles, suggesting that he maybe he is underpaid. In an ideal world, where players are paid exactly in proportion to how they perform relative to the rest of the league, the color scheme of the salary column would carry on into the rest of the performance based columns.

Compare the heat map to simply posting the matrix table:

	salary	games	minutes	points	points3	points2
Goran Dragic	15,890,000	73	2,459	1,483	117	417
Ian Mahinmi	15,944,154	31	555	173	0	65
Timofey Mozgov	16,000,000	54	1,104	401	0	169
Gordon Hayward	16,073,140	73	2,516	1,601	149	396
Evan Turner	16,393,443	65	1,658	586	31	204
Klay Thompson	16,663,575	78	2,649	1,742	268	376
John Wall	16,957,900	78	2,836	1,805	89	558
DeMarcus Cousins	16,957,900	17	574	414	36	106
Joakim Noah	17,000,000	46	1,015	232	0	99
Bismack Biyombo	17,000,000	81	1,793	483	0	179
Evan Fournier	17,000,000	68	2,234	1,167	128	280
Greg Monroe	17,100,000	81	1,823	951	0	387
Wesley Matthews	17,100,000	73	2,495	986	174	159
Enes Kanter	17,145,838	72	1,533	1,033	5	397
Tobias Harris	17,200,000	82	2,567	1,321	109	402
Jimmy Butler	17,552,209	76	2,809	1,816	91	479
Kyrie Irving	17,638,063	72	2,525	1,816	177	494
Kawhi Leonard	17,638,063	74	2,474	1,888	147	489
Luol Deng	18,000,000	56	1,486	425	51	113
Paul George	18,314,532	75	2,689	1,775	195	427
Allen Crabbe	18,500,000	79	2,254	845	134	169
Ryan Anderson	18,735,364	72	2,116	979	204	119
Paul Millsap	20,072,033	69	2,343	1,246	75	355
Blake Griffin	20,140,838	61	2,076	1,316	38	441
LaMarcus Aldridge	20,575,005	72	2,335	1,243	23	477
Nicolas Batum	20,869,566	77	2,617	1,164	135	258
Kevin Love	21,165,675	60	1,885	1,142	145	225
Brook Lopez	21,165,675	75	2,222	1,539	134	421
DeAndre Jordan	21,165,675	81	2,570	1,029	0	412
Marc Gasol	21,165,675	74	2,531	1,446	104	428
Derrick Rose	21,323,250	64	2,082	1,154	13	447
Bradley Beal	22,116,750	77	2,684	1,779	223	414
Hassan Whiteside	22,116,750	77	2,513	1,309	0	542
Andre Drummond	22,116,750	81	2,409	1,105	2	481
Chandler Parsons	22,116,750	34	675	210	25	50
Anthony Davis	22,116,750	75	2,708	2,099	40	730
Harrison Barnes	22,116,750	79	2,803	1,518	78	521
Chris Paul	22,868,828	61	1,921	1,104	124	250
Dwight Howard	23,180,275	74	2,199	1,002	0	388
Dwyane Wade	23,200,000	60	1,792	1,096	45	369
Damian Lillard	24,328,425	75	2,694	2,024	214	447
Carmelo Anthony	24,559,380	74	2,538	1,659	151	451
Dirk Nowitzki	25,000,000	54	1,424	769	79	217
Al Horford	26,540,100	68	2,193	952	86	293
DeMar DeRozan	26,540,100	74	2,620	2,020	33	688
Kevin Durant	26,540,100	62	2,070	1,555	117	434
James Harden	26,540,100	81	2,947	2,356	262	412
Russell Westbrook	26,540,100	81	2,802	2,558	200	624
Mike Conley	26,540,100	69	2,292	1,415	171	293
LeBron James	30,963,450	74	2,794	1,954	124	612

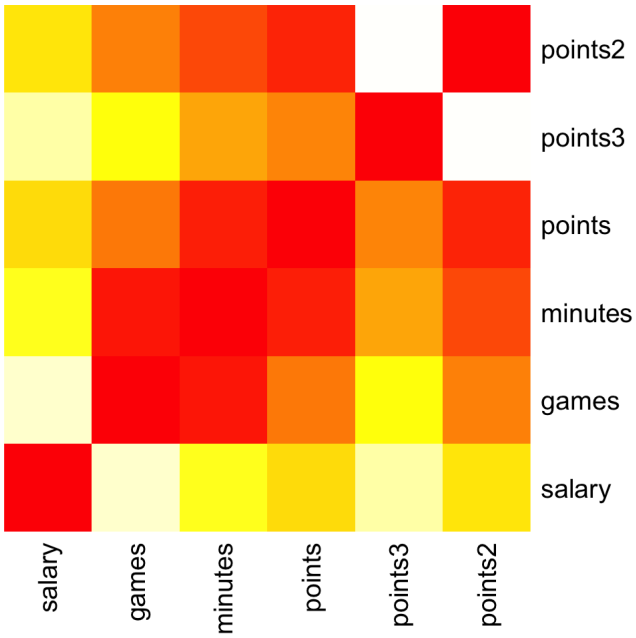
{

width = 50%}

One can easily see how cumbersome it would be for a reader of say, a sports blog, to make the same inferences as we made before with the heatmap. This is just one example of how powerful a heatmap, with the aid of colors, can be in not only communicating but also in increasing the expediency at which readers can analyze data.

What other forms of information could one illustrate with a heatmap. Another useful application could be converting a correlation matrix from simply a bunch of numbers to a heatmap indicating the strength of the correlation between variabels. The first step is to create a corrlation matrix from the data matrix that we previously created.

```
# Create correlation matrix
cor_matrix <- cor(nba)

# Create heatmap of correlation matrix
cor_heatmap <- heatmap(cor_matrix, Colv = NA, Rowv = NA, col = rev(heat.colors(256)), scale = 'none')
```



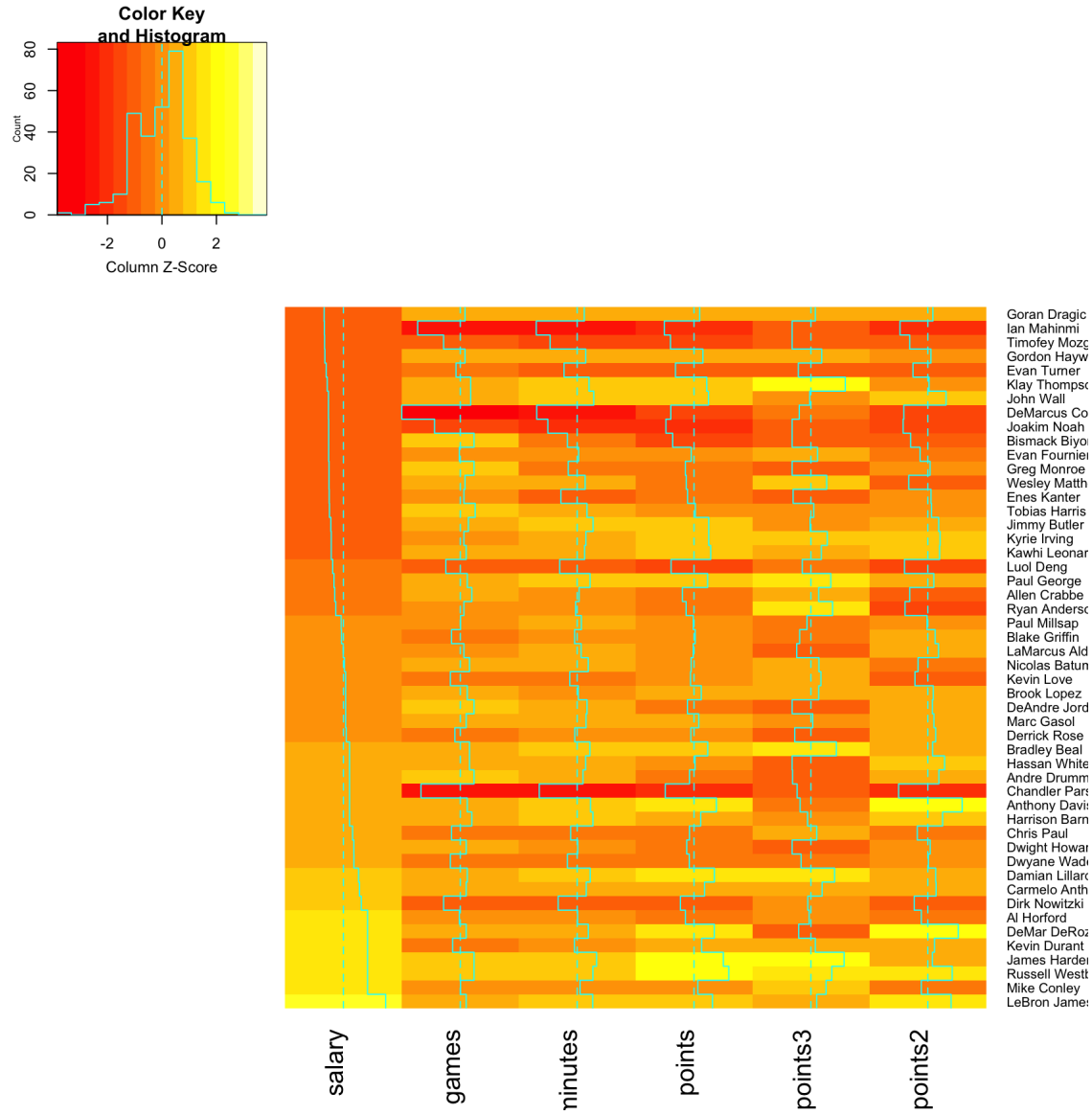

```
cor_heatmap
```

From the heatmap displayed, one can see that the stronger the correlation, the darker the color. Along the diagonal consisting of values of 1 since each variable will have a perfect correlation of 1 with itself, there are only dark red colors. In addition, one can notice the mirror image of color shades across the diagonal just as the values of a correlation amtrix are mirrored across the diagonal. From the white tiles, a very weak correlation exists between salary and the number of games played as well as between the 3 pointers scored and the 2 pointers scored. On the other hand, a very strong correlation exists between the number of minutes a player is on the court and the number of 2 pointers he scores. For readers that may not be well versed in statistics, displays of data that are more aesthetic than a matrix of correlation values (especially when the number of variables are extremeley large), will be much more visually appealing while still communicating the neccesary information.

heatmap.2()

While the the base function for a heatmap is useful, it is not very customizable. For example, what if we want data values printed within tiles in addition to the heatmap? What about different colors? This is where the gplot package comes in handy. The heatmap.2() function within this package was built as an extension from the original heatmap function as its name suggests.

```
nba_heatmap2 <- heatmap.2(nba, Colv = NA, Rowv = NA, scale = 'column')
```

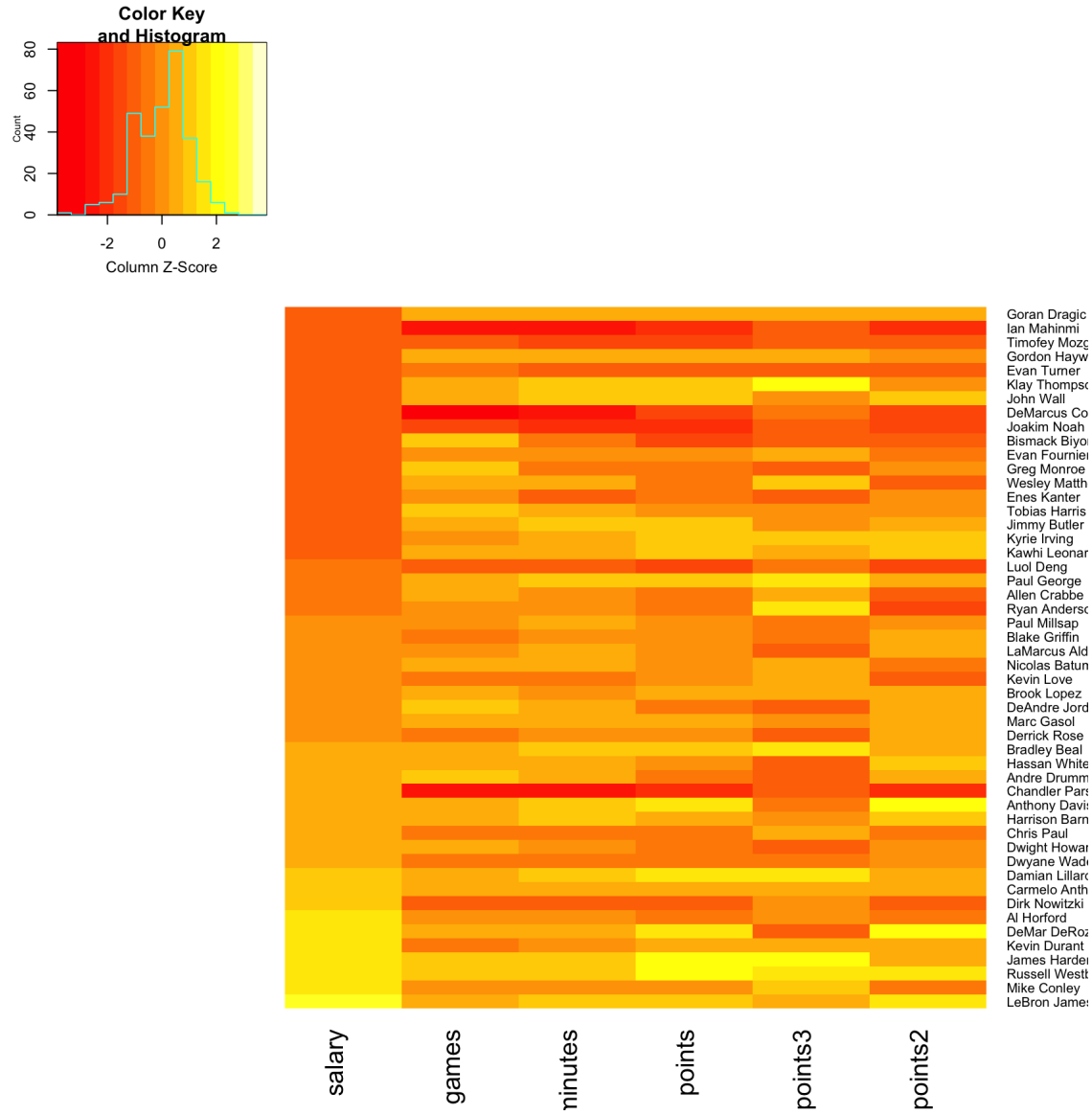



```
nba_heatmap2
```

From the resulting output, we essentially recreated the heatmap from before. However, we now have an interesting addition to it. For each column, there exists a blue line. The dotted line represents the center of the values of each variable. The solid line represents the distance of the tile's value from median of the variable. This is called a "trace" line. If this sounds confusing, just look at the salary column. According to the heatmap, the larger the salary relative to other salaries, the lighter the tile shade. Since we ordered the data set based on salary, the tiles get progressively lighter in shade as you move down the column. Similarly, the trace line is almost linear as it starts off to the left of the dotted line for the darker shades, indicating it is one of the lower values in the vector, and ends to the right of the dotted line for lighter shades, indicating it is one of the larger values in the vector. The same pattern holds for the other categories. The trace line is also an example about how a visual dimension can be used to highlight trends that occur within a data set

The default argument for this is 'column', but if one wants the trace line to be drawn along the rows or for both columns and rows simply plug in *trace = 'rows'* or *trace = 'both'* respectively. In addition, to simply remove the trace line:

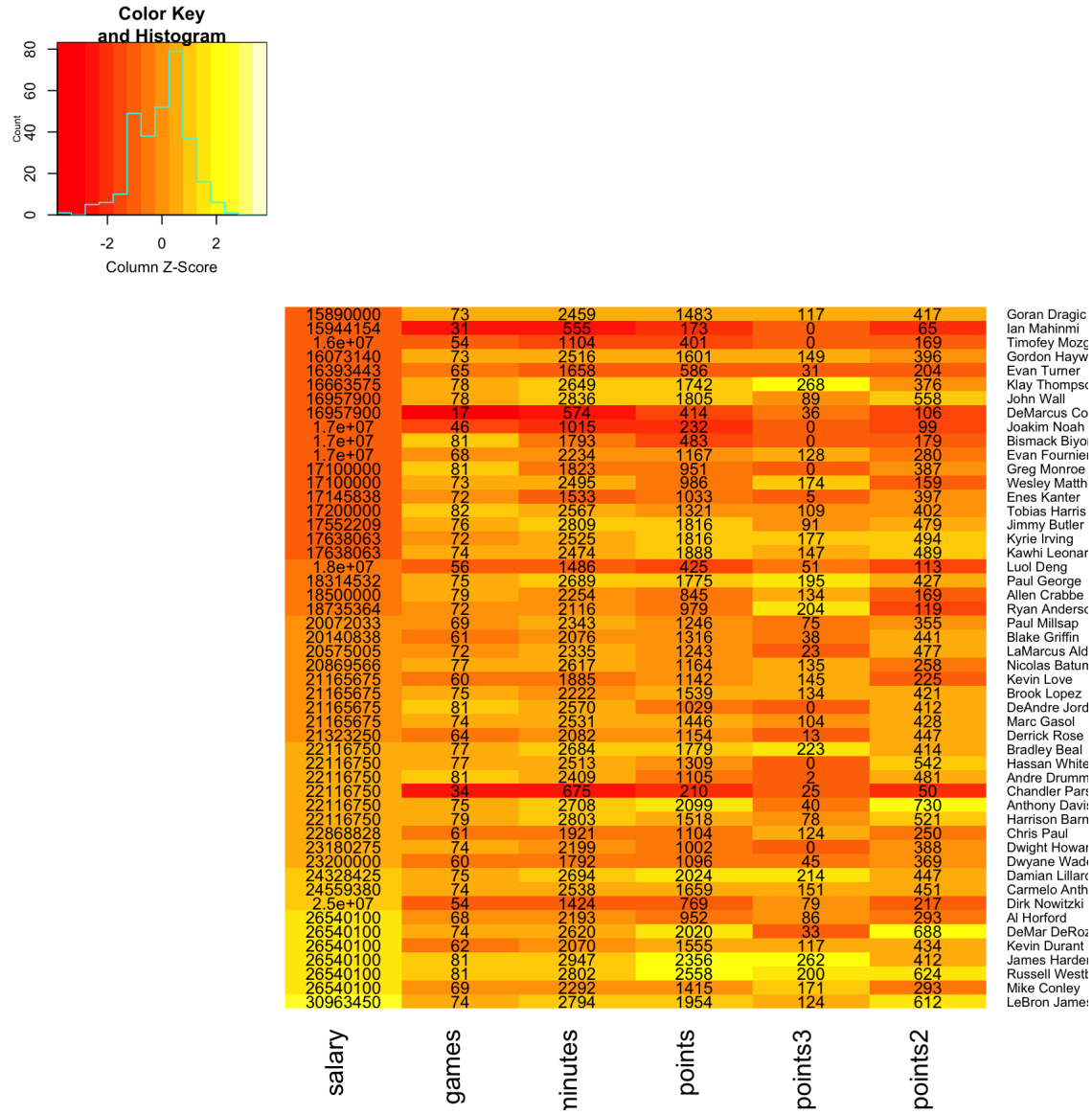
```
nba_heatmap2 <- heatmap.2(nba, trace = 'none',Colv = NA, Rowv = NA, scale = 'column')
```




```
nba_heatmap2
```

As asked before, what if we want to display the actual value of each tile in addition to the color. The cellnote argument accomplishes. By simply substituting in the following argument for 'cellnote' which tells the function that the notes of the heatmap match up with the data we used to create the heatmap, the corresponding values will be displayed:

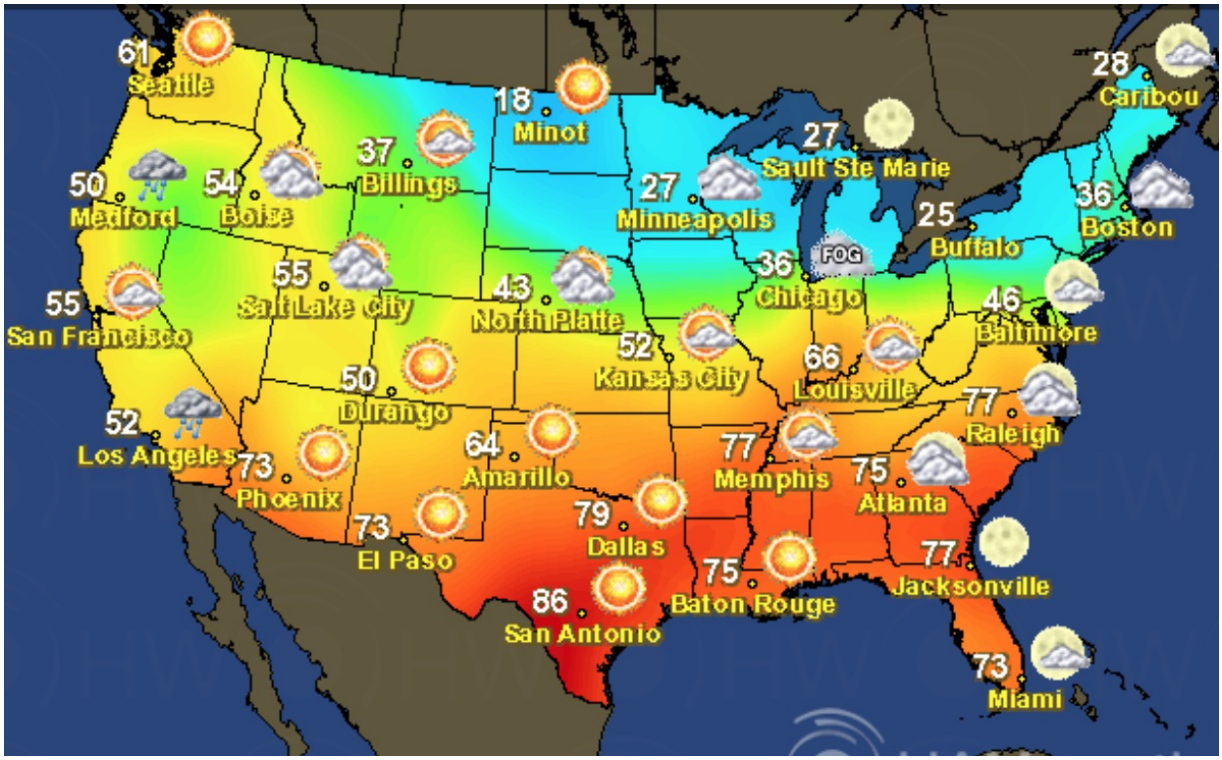
```
nba_heatmap2 <- heatmap.2(nba, trace = 'none',Colv = NA, Rowv = NA,scale = 'column', cellnote = nba, notecol = 'bl
ack')
```



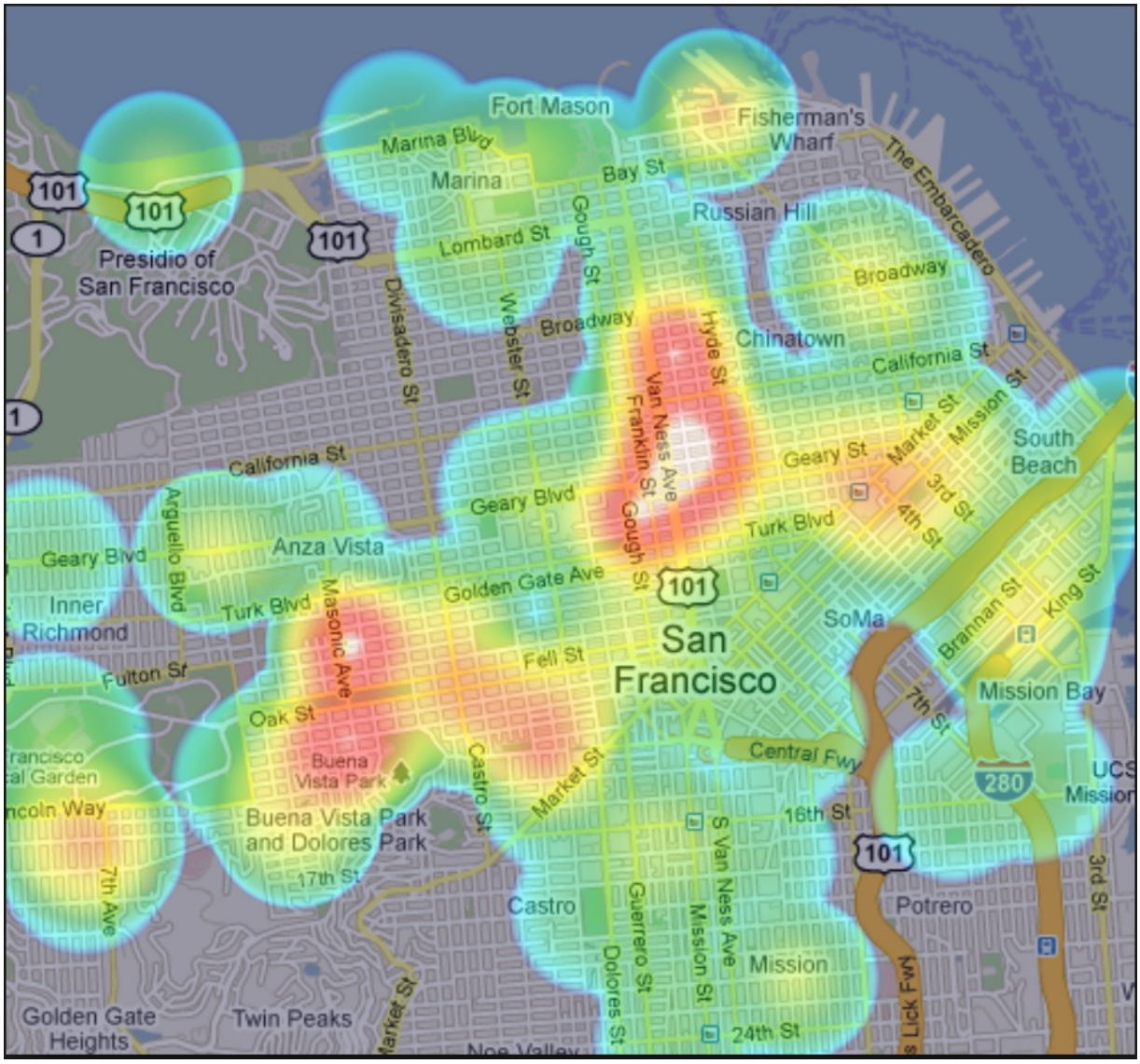

```
nba_heatmap2
```

Heatmaps applied to spatial environments

The final application of heatmaps is in relation to spatial maps. The most notable use of heatmaps are actually within literal maps of cities, countrys, etc. The most common heat map that you've probably seen has to do with actual heat! Whenever you watch the weather man, you are looking at a heat map with temperature as the data:



In addition, they can also be used to convey the frequency a certain occurence happens in a particular area. for example, the following is a heat map visualizing the frequency of crimes within the San Francisco area:



What we're going to do is recreate the heatmap showing the number of noise complaints in Berkeley from the Daily Cal

First load the relevant package (ggmap), read in the data and inspect it:

```
noise <- read_csv('data/noise_citation_coords.csv')
```

```
## Warning: Missing column names filled in: 'X3' [3]
```

```
## Parsed with column specification:
## cols(
##   Latitude = col_double(),
##   Longitude = col_double(),
##   X3 = col_character()
## )
```

```
summary(noise)
```

```
##     Latitude       Longitude           X3
##  Min.   :37.78   Min.   :-122.4   Length:3006
##  1st Qu.:37.86   1st Qu.:-122.3   Class :character
##  Median :37.87   Median :-122.3   Mode  :character
##  Mean   :37.87   Mean   :-122.3
##  3rd Qu.:37.87   3rd Qu.:-122.3
##  Max.   :37.92   Max.   :-122.2
##  NA's   :39      NA's   :39
```

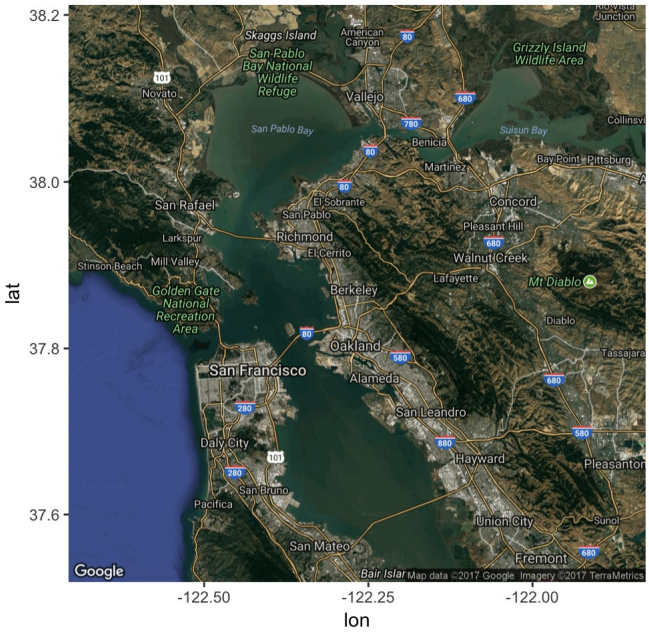
```
# Remove missing values
noise <- na.omit(noise)
```

Take a look at the summary of the data set "noise". There are 3 columns ("Latitude", "Longitude", and "X3"). The data contained within the first 2 columns are self explanatory. The third column simply states the address at which the noise complaint was issued. Before we begin constructing the heat map, we will need to retrieve the actual map on which we are going to be plotting our coordinate points. The package ggmap, is a package that is used to retrieve these maps that we will be plotting points on. Using the function "get_map", ggmap will retrieve a map from Google Maps based on the parameters we place within it.

```
noise <- noise[,1:2]
noise_map <- get_map(location = c(lon = mean(noise$Longitude), lat = mean(noise$Latitude)), maptype = "hybrid", zo
om = 'auto', color = "color" )
```

```
## Source : https://maps.googleapis.com/maps/api/staticmap?center=37.866597,-122.267789&zoom=10&size=640x640&scale
=2&maptype=hybrid&language=en-EN
```

```
ggmap(noise_map)
```



For the get_map function, the first argument is location. This argument must be in the form of a vector of length 2 with the latitude and longitude of the center of the map needed. By the using the mean function on the Longitude and Latitude column of "noise", I extracted those two values as the center.

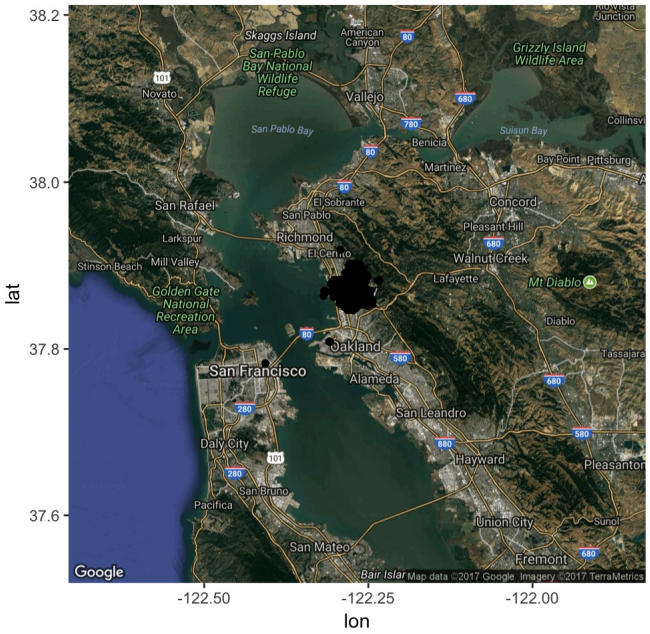
There are also the following types of maps that one can retrieve from Google: terrain",”terrain-background",”satellite",”roadmap", and”hybrid" (google maps), "terrain", "watercolor", and "toner".

Zoom is the degree to which one wants the zoom the map (an integer from 3 to 18) with the larger integers zooming in. I placed “auto” preliminarily to see what the map would look like. After plotting the points we can adjust it accordingly.

Finally, color is simply a choice between black and white (“bw”) or color (“color”).

ggmap is a function within the package ggplot2 that provides a map against which we can plot our coordinate points. Just as ggplot() is used as a precursor for what we are going to plot scatter points on, ggmap() is a precursor for what we are going to plot coordinate points but isntead of a blank canvas, we will be using a map we retrieved with get_map. Let now add our coordinate points.

```
ggmap(noise_map) + geom_point(data = noise, aes(x = Longitude, y = Latitude))
```

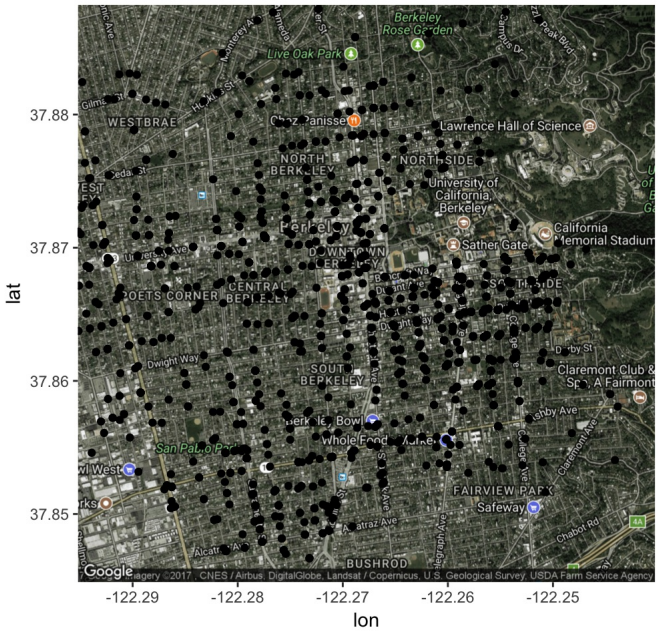


Clearly, this map is useless to us since the zoom isn’t properly adjusted. After toggling for a bit, you should settle on a zoom of around 14 as such.

```
noise_map <- get_map(location = c(lon = mean(noise$Longitude), lat = mean(noise$Latitude)), maptype = "hybrid", zo
om = 14, color = "color" )
```

```
## Source : https://maps.googleapis.com/maps/api/staticmap?center=37.866597,-122.267789&zoom=14&size=640x640&scale
=2&maptype=hybrid&language=en-EN
```

```
ggmap(noise_map) + geom_point(data = noise, aes(x = Longitude, y = Latitude))
```

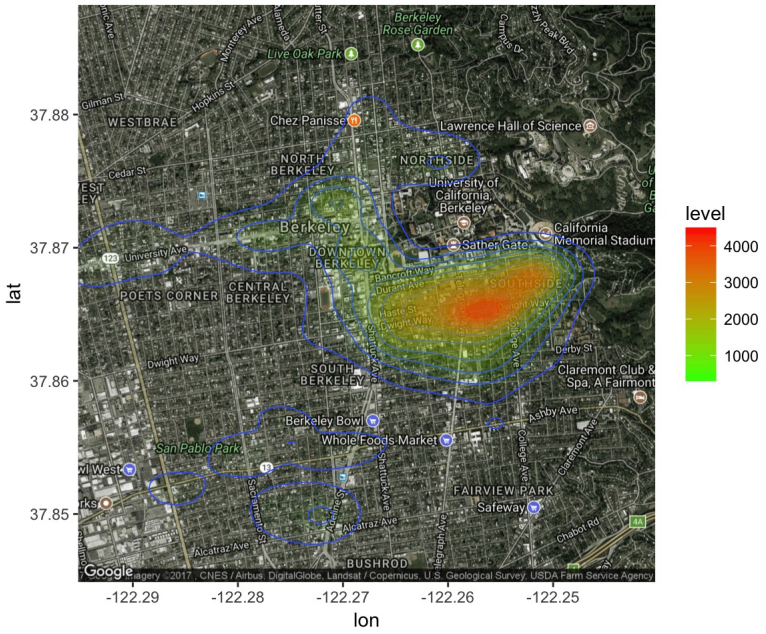


There you have it! A map with all the locations of where noise complaints occured. However, that’s not the point of this post. We want to create a heatmap showing the concentration of noise complaints in a certain area. This is simliar to the crime heat map of San Francisco we saw, except the location is now Berkeley and instead of crime, we are looking at noise complaints.

The proper function for this sort of task is the stat_density2d function within ggplot

```
ggmap(noise_map) + geom_density2d(data = noise, aes(x = Longitude, y = Latitude), size = 0.3) +
  stat_density2d(data = noise, aes(x = Longitude, y = Latitude, fill = ..level.., alpha = ..level..), size = 0.01,
bins = 16, geom = "polygon") + scale_fill_gradient(low = "green", high = "red") + scale_alpha(range = c(0, 0.3), g
uide = FALSE)
```

```
## Warning: Removed 177 rows containing non-finite values (stat_density2d).

## Warning: Removed 177 rows containing non-finite values (stat_density2d).
```



The first addition to the ggmap plot is “geom_density2d. This performs a two dimensional kernel density estimation that displays the contours (lines) of these varying densities. In addition, the the stat density2d function plots the actual density estimates (colors) on the map. You can think of the lines as outlining a certain region in accordance to having a certain concentration of noise complaints. Notice how as the contour lines get bigger, the color shifts to cooler ones and the colors also become more transparent..

Within the stat_density2d function, the initial paramters are the same as any other plots. First, the data comes from the data.frame noise. In addition, because we are plotting based on Latitude and Longitude from the noise data frame, those two variabels are entered within aes().

In addition, notice the fill and alpha arguments. The reason why these have “..level..” as an argument is because when we call stat_density2d, we want the color shift according to the ‘level’ of density in the map. Alpha represent the transparency of the colors. To make it more similar to a weather map like we’re used to seeing, we adjust alpha so that the colors become more transparent as the ‘level’ of density decreases.

The “scale fill gradient”" function requires two character arguments that are colors for the highest density and lowest density. In addition, the scale_alpha does the same except instead of colors, the corresponding levels of transparency are provided. To what effect these, try removing the from the plot and toggle the colors and/or levels.

With the information provided, we can see from our heatmap that the highest concentration of noise complaints occur (not surprisngly) in the Southside of Berkeley where there is a higher concentration of undergraduate students living as well as Greek houses.

Conclusion

Heatmaps are an incredibly powerful visual tool that can be used to immediately convey conclusion that may not otherwise be apparent in bland displays of numbers and tables. The most common heatmaps we see on a day to day basis are weather maps describing the various temperatures on a map. However, the application of such a tool can be used in the context of noise complaints, as we performed, as well as crime statistics. But another use can be found in the simple delineation of tables. As we discovered with the nba players data, without the need of scatterplots, correlation matrices, and regression lines, we were able to deduce which variables of player stats were strongly (or weakly) correlated with other variables. When one needs to convey immediate points or provide a more colloquial way of highlighting trends to the general public, heat maps are one of the most effective ways of accomplishing such a task.

References

http://www.dailycal.org/2017/04/23/data-shows-berkeley-noise-complaints-underage-drinking-citations-largely-concentrated-southside/

http://www.geo.ut.ee/aasa/LOOM02331/heatmap_in_R.html

https://flowingdata.com/2010/01/21/how-to-make-a-heatmap-a-quick-and-easy-solution/

https://learnr.wordpress.com/2010/01/26/ggplot2-quick-heatmap-plotting/

http://www.exegetic.biz/blog/2013/12/contour-and-density-layers-with-ggmap/