Heatmaps in R: Colored Representations of Data

Katherine Li

October 29, 2017

My post is about colored representations of data in R, specifically through heatmaps. I want to explore this topic because I am a visual learner who finds color-coding extremely helpful, and this technique presents a visual aspect of data manipulation that complements the subsetting and wrangling methods we studied in class. One of my favorite tutorials on heatmaps is by seasoned programming languages author Sebastian Raschka. He explains that this is a great tool for exploratory data analysis as it allows for surface-level identification of numerical patterns through color that enables us to "dig deeper… to further customize our plots and maybe even polish them for publication." In this post, I discuss what a heatmap is, how to create one, and useful applications of this method. I also provide examples of heatmaps with the NBA data we've worked with in class. My analysis focuses on the methodology of heatmaps rather than its application to different datasets because I see techniques for analyzing heatmaps as being constant across different sets of data.

What is a heatmap?

A heatmap makes it easier for users to determine patterns in their data by substituting numbers with colors denoting the level of measurement. Users can apply heatmaps to data so long as the data can be wrangled into matrix form. In class, we learned that a matrix is an atomic vector; it is therefore important to keep in mind that the heatmap function only works on data that is of one type.

How do I create a heatmap?

There are multiple heatmap functions in R, including the heatmap() function from the *stats* package and the heatmap.2() function from the *gplot* package. In this post, I demonstrate how heatmap works through the heatmap.2() function but the two functions are similar. The main difference is the default values for the scaling and clustering arguments of either function.
Keeping in mind that heatmaps must be applied to matrices, we create a heatmap by applying the function heatmap.2() to our data matrix. While this approach is itself fairly straightforward, it is more difficult to understand the process behind creating a heatmap, which can be divided into the clustering and coloring parts:

1. Clustering in Heatmaps
There are countless ways to cluster data. One simple way to do so – also the default heatmap function way – is by finding the average pairwise Euclidean distance between data points in different groups and joining groups that are the shortest distance apart. Mick Watson illustrates this process in his post on heatmapping gene expressions, in which he uses a heatmap to cluster together four genes by their distance. This is done specifically through the dendrogram argument of the heatmap.2() function, which offers options on clustering by row, column, both, or neither. He then determines scaling of his subgroups; this can occur before or after clustering, or not at all. Depending on how the user wishes to display data, this will affect the coloring of the graph because the heatmap colors cell data depending on how closely matched they are.

2. Coloring in Heatmaps
I use the package RColorBrewer to select colors for my heatmap. The most common colors, according to Raschka, are red, green, and yellow. In practice, the coloring of your heatmap depends on your preferences (I've seen heatmaps with all sorts of color combinations and I consider this the most fun part of heatmaps because I love pairing colors together). Once the colors for the heatmap are selected, the heatmap.2() function will naturally generate a key specifying what the color gradient represents for your map.

What are some practical applications of heatmaps?

We've seen from Mike Watson's blog that heatmaps can be applied to biology and gene expressions to determine high and medium and low expressions of genes and similarities between genes. In fact, many of the use cases for the heatmap functions that I've researched online have been about organizing data in biological studies. But heatmaps can be used to color any dataset – to sort cars or to sort people, as in the example I will provide later in this post coloring the NBA dataset.

Other applications of heatmapping exist through the ggplot and ShinyApp functions that we learned about in class. Reading about this intersection between heatmaps and the latter two functions was exciting for me because this class also first exposed me to ggplot and ShinyApp, two programs I find easier to comprehend because they are also both visual.

Unlike heatmap.2(), which can only create a heatmap out of a data matrix, ggplot2() enables users to heatmap data frames because it requires a data frame argument. There is no heatmap layer in ggplot, so this is done using the geom_tile() and smoothing gradient layers along with ggplot2(). This article about using ggplot to perform heatmapping applies this technique to the set of NBA data. It is clear that the two approaches have the same effect.

The Shiny application shinyHeatmaply enables users to interact with their heatmaps by adjusting features such as scaling, margins and the dendrogram in the sidebar panel. This was very cool to use because we are currently learning about how widgets are programmed into our ShinyApp on the user interface and server sides. The finished product looks very much like the examples with Old Faithful Geyser data that we covered in class.

Lastly, I would like to point out that heatmaps are not limited to data matrices. They can also be applied to color points in Principle Component Analysis. R-blogger Ming Tang provides several examples of applications of heatmapping. I found this incredibly interesting because it helps with visualizing a highly technical analysis method, which took me a great deal of time to grasp, that we learned about in class and had practice with for one of our assignments.

What are some examples of how to use the heatmap.2() function?

I would like to discuss a basic implementation of the heatmap.2() function by using the NBA data sets that we've seen before in class. I set up my example by downloading the data file, setting my working directory to the location of my downloaded data file, and creating a table from that file using the read.csv() function that we've had a lot of practice with in class.

```
# set working directory to where data file is
# setwd("../../Desktop/stat133/stat133-hws-fall17/posts")

# download data code
# download.file("https://raw.githubusercontent.com/ucb-stat133/
# stat133-fall-2017/master/data/nba2017-player-statistics.csv", destfile = "nba2017data.csv")

# create data table
dat <- read.csv("nba2017data.csv", stringsAsFactors = FALSE)
```

I then install and load all my packages. In this case, because I am implementing heatmaps, I need the packages gplot and RColorBrewer. As I mentioned above, gplot provides me access to the heatmap.2() function and RColorBrewer allows me to control the colors that show up on my heatmap. I install the pheatmap package to show ways to enhance further the visualizations of heatmap.

```
# install and load package gplots
# install.packages("gplots")
library(gplots)
```

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

```
##
## Attaching package: 'gplots'
```

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

```
# install and load package RColorBrewer
# install.packages("RColorBrewer")
library(RColorBrewer)

# install and load package pheatmap
# install.packages("pheatmap")
library(pheatmap)
```

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

Because I am using the heatmap.2() function and not creating my heatmap through ggplot, I need to coerce my data into matrix form. To simplify my example, I've selected the last few columns of my data set, dealing with points scored, blocks, steals, and assist data to be part of my data matrix. I label these statistics with player names and shrink my matrix so that it only displays Golden State Warrior data. My goal is to color-compare Golden State Warrior players with each other.

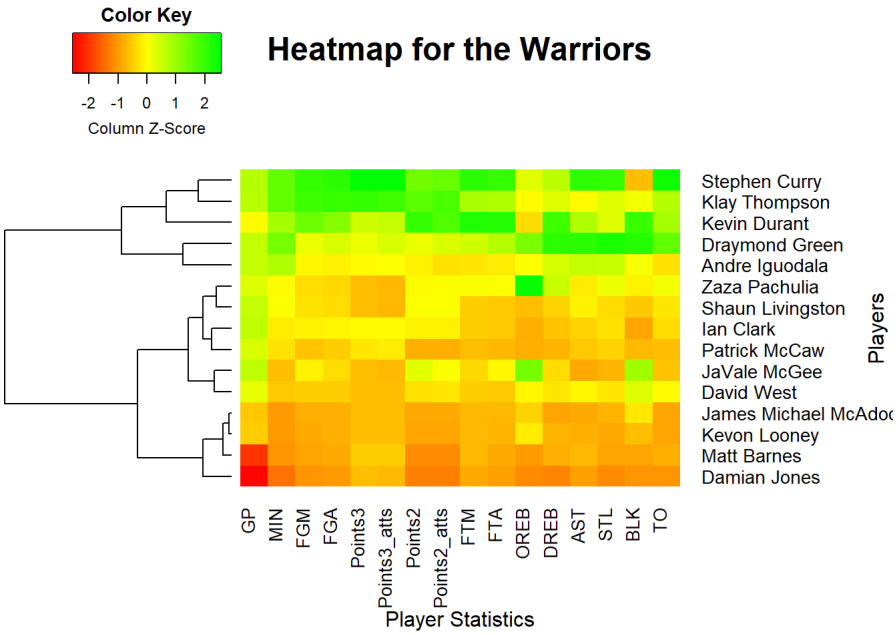
```
# separating row names
rnames <- dat$Player[dat$Team == "GSW"]

# creating a matrix of only numerical values
mat_dat <- data.matrix(dat[dat$Team == "GSW" , c(8, 10:ncol(dat))])

# reassigning row names
rownames(mat_dat) <- rnames
```

I chose the colors in my color palette to be the colors Raschka said were the most commonly used colors – red, yellow, and green.

```
# using RColorBrewer to select my color palette for my heatmap
my_palette <- colorRampPalette(c("red", "yellow", "green")) (n = 100)
```

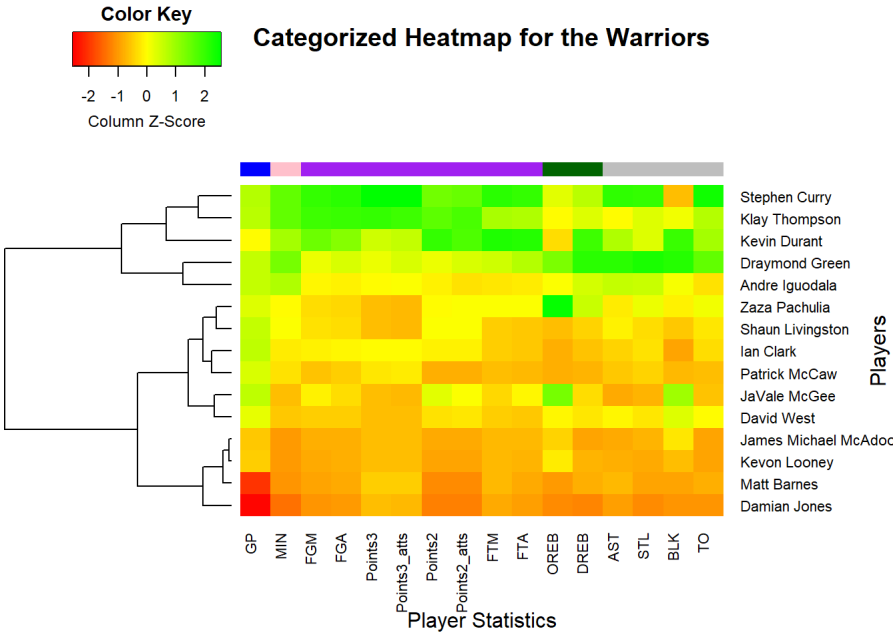
This is my heatmap code. I created a row dendrogram and scaled by columns for Warriors data, so players with the closest datapoints are grouped together and player are reordered depending on these groups. I commented out the "cellnote" argument, which labels each cell by its datapoint, to provide better visualization of the groupings but it is possible to add greater clarity to the coloring by including the data.

```
# heatmap for Warriors data
heatmap.2(mat_dat,         # my data matrix
  #cellnote = mat_dat,     # same data set for cell labels
  main = "Heatmap for the Warriors", # heatmap title
  notecol = "black",       # change font color of cell labels to black
  density.info = "none",   # turns off density plot inside color legend
  trace = "none",          # turns off trace lines inside the heatmap
  xlab = "Player Statistics", # x-axis label
  ylab = "Players",        # y-axis label
  margins = c(7,10),      # widens margins around plot
  col = my_palette,        # use on color palette defined earlier
  Colv ="NA",              # says that the column dendrogram should not
                           # be reordered
  dendrogram = "row",      # how the dendrogram should be applied
  scale = "column")        # how the matrix should be scaled
```



With my heatmap, I'm also able to categorize groups of rows and columns by color. I ended up grouping together similar columns in my NBA example. For example, I colored the sections related to points scored – free throws made, free throws attempted, points3 made, etc. – purple because I considered these measures to be fairly similar. This technique can further assist with the visualization of data and make the different components easier to interpret. Both Raschka and this heatmap article discuss color categorizing in more detail.

```
# heatmap for Warriors with colored categories
heatmap.2(mat_dat,
  main = "Categorized Heatmap for the Warriors",
  notecol = "black",
  density.info = "none",
  trace = "none",
  margins = c(7,10),
  xlab = "Player Statistics",
  ylab = "Players",
  col = my_palette,
  Colv ="NA",
  dendrogram = "row",
  scale = "column",
  ColSideColors = c("blue", "pink", rep("purple", 8),        # where you specify
                    rep("dark green", 2), rep("gray", 4)))  # your category colors
```



As I mentioned before, heatmaps consist of the clustering and coloring components. In this example, I try out methods for clustering, other than the most commonly used Euclidean distance method. The resulting graph shows a different ordering of the columns than the one we had previously witnessed with the Euclidean distance heatmap generator: the user can customize the method of clustering depending on what they wish to gain from the data visualization.

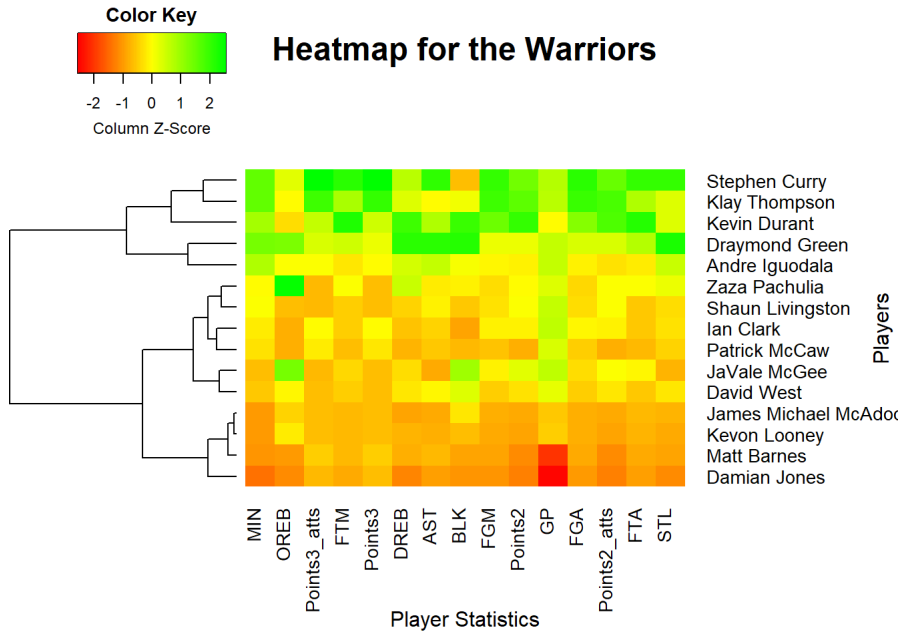
My favorite feature that I've come across while researching heatmaps is the pheatmap function, which allows you to slice your data matrix into groups of players with similar statistics. It creates a neater and even more beautiful version of the original heatmap in the process. A tutorial by Kamil Slowikowski demonstrates pheatmapping using quantile breaks, which allows us to visualize the unequal proportions of data from each color and reposition them so we can compare equal proportions of color in the data. I don't implement this method myself because I think it extends beyond the foundational level of heatmapping I aim to describe in this post, but I found it to be an interesting application.

```
# trying out other clustering methods that do not use Euclidean distance
distance = dist(mat_dat, method = "manhattan")
cluster = hclust(distance, method = "ward.D2")
heatmap.2(mat_dat,
  main = "Heatmap for the Warriors",
  notecol = "black",
  density.info = "none",
  trace = "none",
  xlab = "Player Statistics",
  ylab = "Players",
  margins = c(7,10),
  col = my_palette,
  Colv = as.dendrogram(cluster), # for applying another column cluster method
  Rowb = as.dendrogram(cluster), # for applying another row cluster method
  dendrogram = "row",
  scale = "column")
```

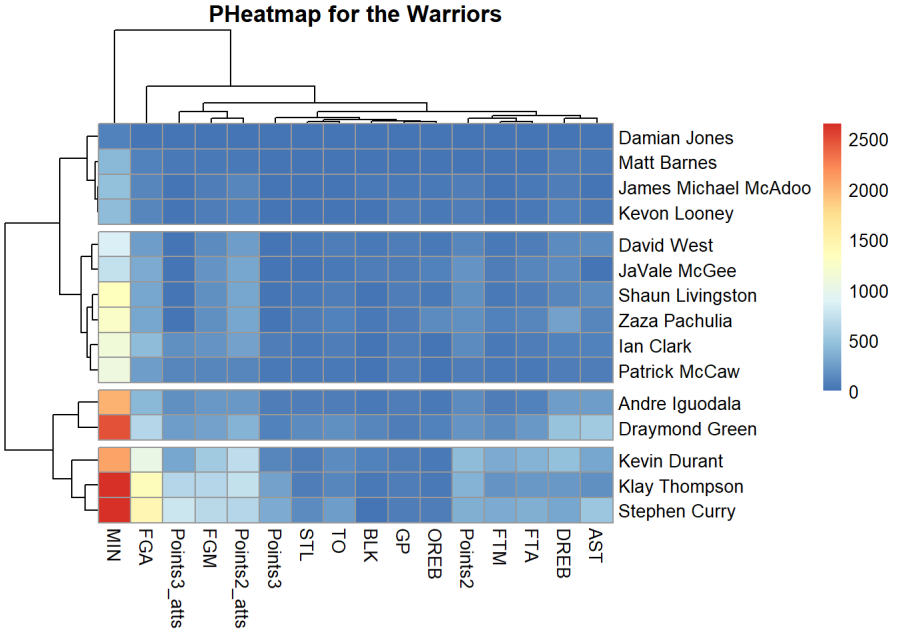
```
## Warning in plot.window(...): "Rowb" is not a graphical parameter
```

```
## Warning in plot.xy(xy, type, ...): "Rowb" is not a graphical parameter
```

```
## Warning in title(...): "Rowb" is not a graphical parameter
```




```
# trying out the pretty heatmap function
pheatmap(mat_dat, cutree_rows = 4, main = "PHeatmap for the Warriors")
```



Conclusion

Heatmapping therefore is a quick and easy method of determining patterns in data that can lead to meaningful analysis. This method is implemented by clustering and coloring a data matrix; in fact, many of the methods we've learned about in class – like ggplot, ShinyApp, and Principal Component Analysis – can be enhanced through it. I believe it is a topic worth posting about not only because it connects many of the elements we've already studied but also because it is a fun, visualization technique that people use in real-life data analysis initially to interpret their data. One thing to keep in mind is that: while the function itself is not difficult to implement and call, adding more features to the heatmap can get more complex. In this case, spend some time studying the specific parameters and you'll have your color-organized data matrix in no time!

References

http://sebastianraschka.com/Articles/heatmaps_in_r.html

http://www.opiniomics.org/you-probably-dont-understand-heatmaps/

http://www.sthda.com/english/articles/28-hierarchical-clustering-essentials/93-heatmap-static-and-interactive-absolute-guide/

http://slowkow.com/notes/heatmap-tutorial/

https://rpubs.com/crazyhottommy/heatmap_demystified

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

https://digitheadslabnotebook.blogspot.com.au/2011/06/drawing-heatmaps-in-r.html?m=1

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

https://www.r-bloggers.com/choosing-colour-palettes-part-i-introduction/

https://www.r-statistics.com/2017/03/shinyheatmaply-a-shiny-app-for-creating-interactive-cluster-heatmaps/

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.