Heatmaps in R

Kayvan Jalali & Carlos Puerta

9/27/2023

Section 1

What is a heatmap?

What is a heatmap?

A heatmap is graphic that will display your data in a colorful grid. This is great for seeing trends and patterns in your data.

Generally, heatmaps are used to represent data where you have 2 categorical variables with a third continuous variable, though this is not explicitly necessary.
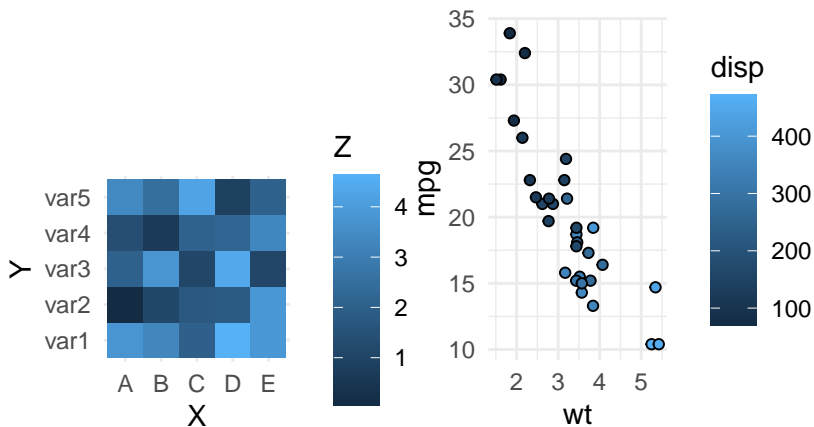
Advantages

- Adds another variable
- Easier to interpret
- Color is easy to parse

Disadvantages

- Need right choice of colors and data
- Only useful if clear
- A different plot may be better

# Examples

When to use a heatmap

Heatmaps are applicable to both continuous and discrete variables. When working with discrete variables, distinguishing between similar colors can be challenging. To improve visibility, it is recommended to use a color palette of no more than nine colors, such as ROYGBIV, black, and white.

In the case of continuous variables, you have the option to use a sequential color scale, transitioning from a lighter hue to a darker one or vice versa. Alternatively, you can create a divergent color scale, transitioning between two distinct colors. A divergent color scale may be more useful when the center and ends of the value range are meaning full.

ggplot2

You can create a simple and quick heatmap using the geom_tile() function in ggplot2, but these heatmaps are very limited in their functionality.

```
ggplot(data, aes(X, Y, fill = Z)) +
  geom_tile() + coord_equal()
```
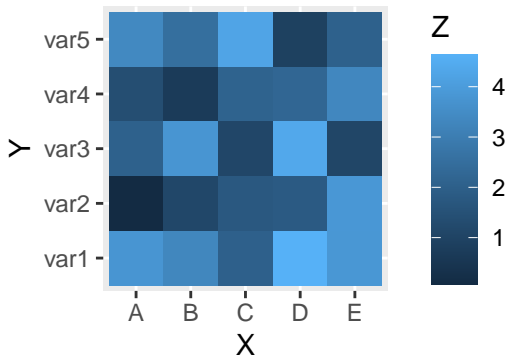
Section 2

Pheatmap Package

Pheatmap Package

Pheatmap (Pretty Heatmaps) is a package for R that supercharges heatmaps and allows you to create incredibly complex and helpful heatmaps. Although it is slightly more difficult to use when compared to ggplot, it is highly specialized and focused. It can plot more than 3 variables in a single heatmap, and more importantly, it groups rows and/or columns.

The grouping specifically can be incredibly helpful for visually spotting trends and similarities.

Dendrogram

Dendrograms, create a higherarchy of similar groups. Eventhough they can be standalone, and are not always seen with heatmaps, pheatmap includes dendrograms by default. Groups that are most similar get pooled together until there are no more groupings possible. The distance between groups directly reflects the difference between groups. By default these groupings are made using the rows or columns euclidean distance.
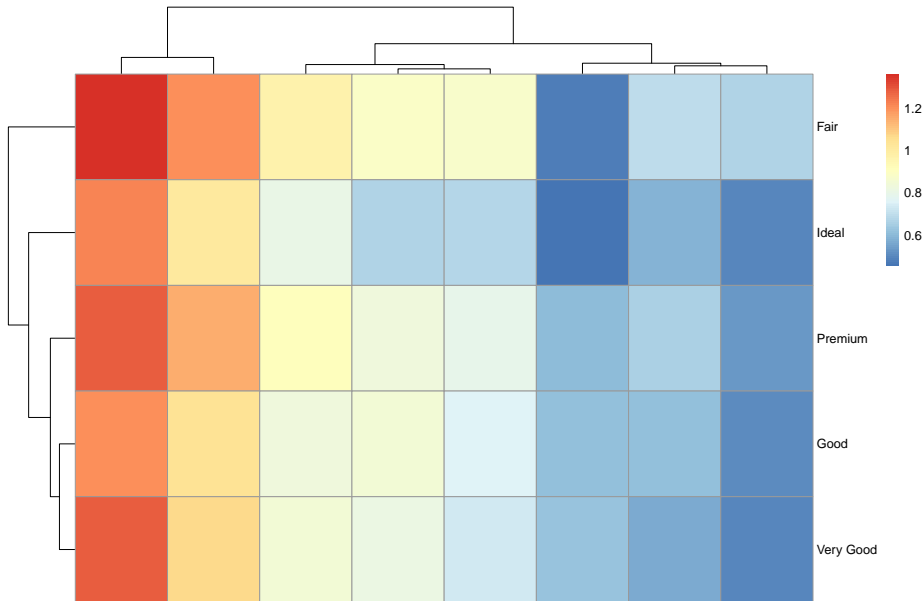
A Bad Tournament

You can think of it as a tournament bracket where seeding determines your matchups, your initial matches will be the strongest teams against each other and the weakest teams against each other. The distance between the connections symbolizes the overall difference in skill.

Pheatmap example data creation

Basic Functions

```
## 'summarise()' has grouped output by 'cut'. You can override using
## argument.
```
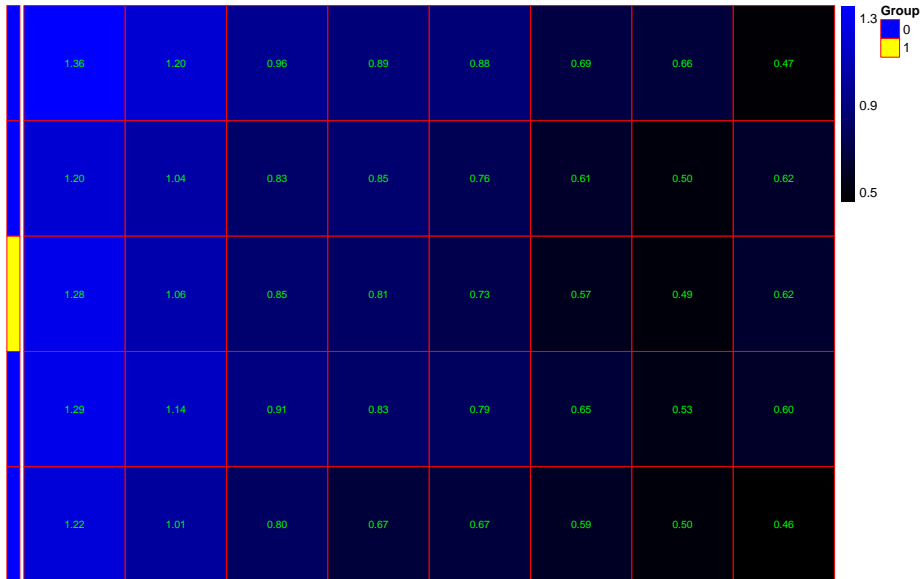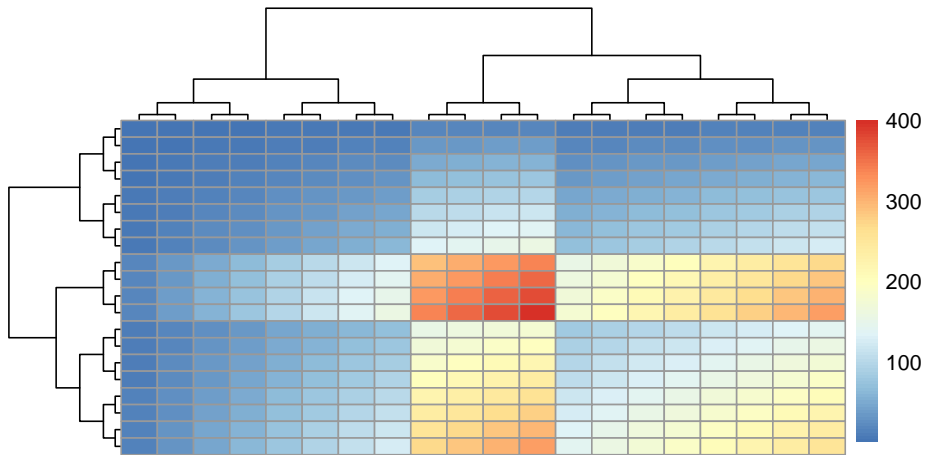
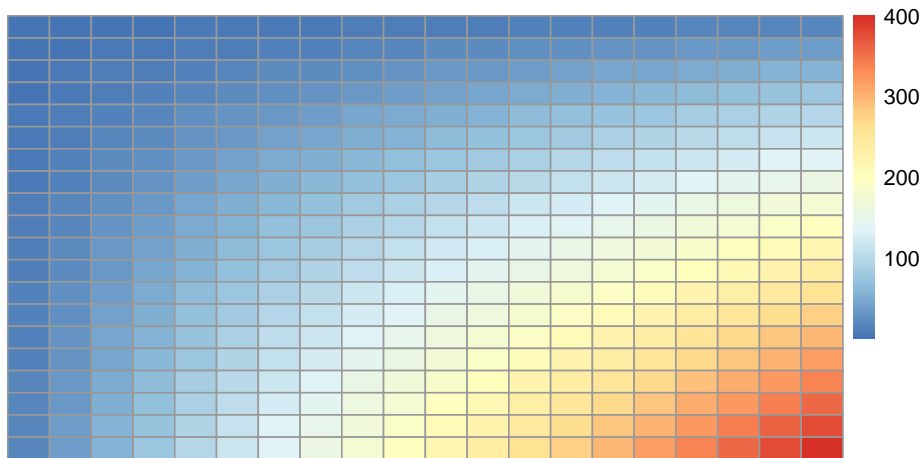# Pheatmap with no Arguments

Make Annotations

Lots of Function

Average Price by Cut and Clarity

Just Pheatmap

Dendrogram

Color
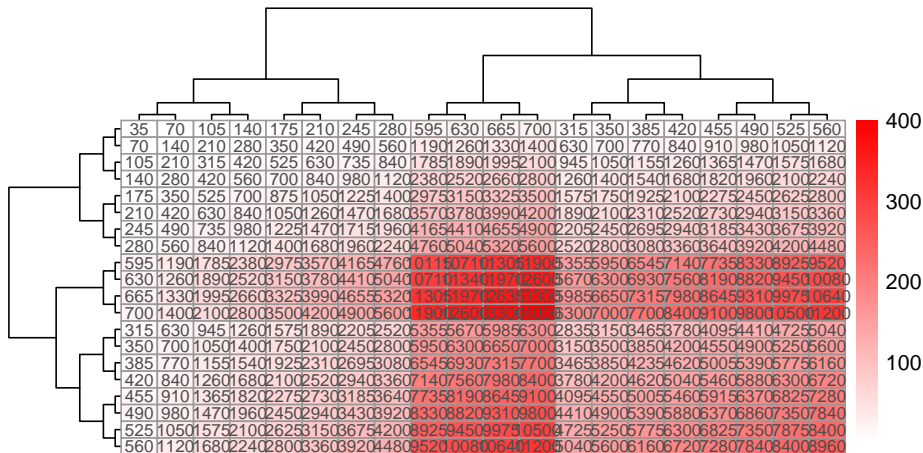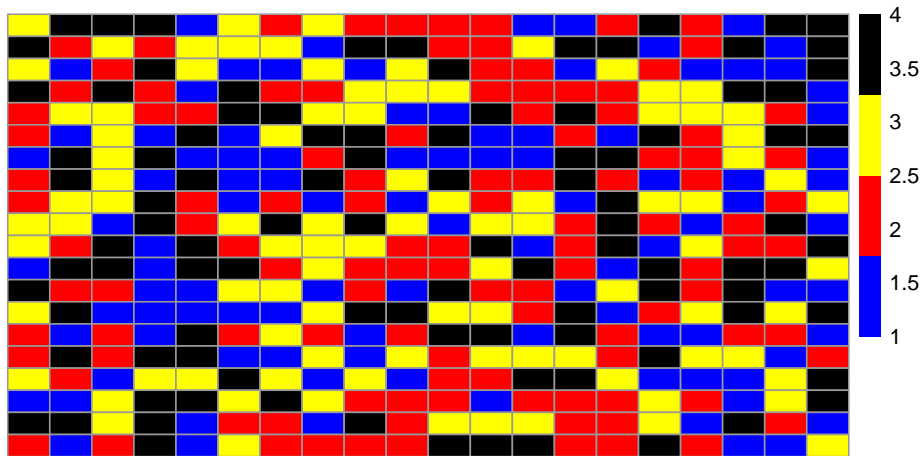

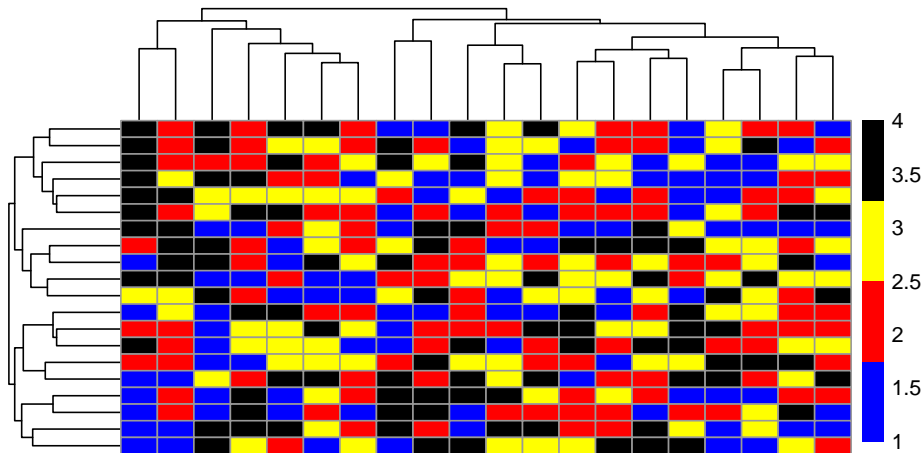| 35	70	105	140	175	210	245	280	315	350	385	420	455	490	525	560	595	630	665	700
70	140	210	280	350	420	490	560	630	700	770	840	910	980	1050	1120	1190	1260	1330	1400
105	210	315	420	525	630	735	840	945	1050	1155	1260	1365	1470	1575	1680	1785	1890	1995	2100
140	280	420	560	700	840	980	1120	1260	1400	1540	1680	1820	1960	2100	2240	2380	2520	2660	2800
175	350	525	700	875	1050	1225	1400	1575	1750	1925	2100	2275	2450	2625	2800	2975	3150	3325	3500
210	420	630	840	1050	1260	1470	1680	1890	2100	2310	2520	2730	2940	3150	3360	3570	3780	3990	4200
245	490	735	980	1225	1470	1715	1960	2205	2450	2695	2940	3185	3430	3675	3920	4165	4410	4655	4900
280	560	840	1120	1400	1680	1960	2240	2520	2800	3080	3360	3640	3920	4200	4480	4760	5040	5320	5600
315	630	945	1260	1575	1890	2205	2520	2835	3150	3465	3780	4095	4410	4725	5040	5355	5670	5985	6300
350	700	1050	1400	1750	2100	2450	2800	3150	3500	3850	4200	4550	4900	5250	5600	5950	6300	6650	7000
385	770	1155	1540	1925	2310	2695	3080	3465	3850	4235	4620	5005	5390	5775	6160	6545	6930	7315	7700
420	840	1260	1680	2100	2520	2940	3360	3780	4200	4620	5040	5460	5880	6300	6720	7140	7560	7980	8400
455	910	1365	1820	2275	2730	3185	3640	4095	4550	5005	5460	5915	6370	6825	7280	7735	8190	8645	9100
490	980	1470	1960	2450	2940	3430	3920	4410	4900	5390	5880	6370	6860	7350	7840	8330	8820	9310	9800
525	1050	1575	2100	2625	3150	3675	4200	4725	5250	5775	6300	6825	7350	7875	8400	8925	9450	9975	10500
560	1120	1680	2240	2800	3360	3920	4480	5040	5600	6160	6720	7280	7840	8400	8960	9520	10080	10640	11200
595	1190	1785	2380	2975	3570	4165	4760	5355	5950	6545	7140	7735	8330	8925	9520	10115	10710	11305	11900
630	1260	1890	2520	3150	3780	4410	5040	5670	6300	6930	7560	8190	8820	9450	10080	10710	11340	11970	12600
665	1330	1995	2660	3325	3990	4655	5320	5985	6650	7315	7980	8645	9310	9975	10640	11305	11970	12635	13300
700	1400	2100	2800	3500	4200	4900	5600	6300	7000	7700	8400	9100	9800	10500	11200	11900	12600	13300	14000

Pheatmap example

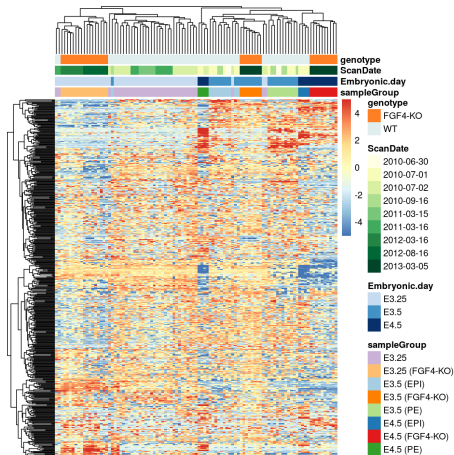# Pheatmap example

Pheatmap example

More Complex Example

Genes

Code for Genes

```
topGenes = order(rowVars(Biobase::exprs(x)),
                 decreasing = TRUE)[1:500]
rowCenter = function(x) { x - rowMeans(x) }
pheatmap(rowCenter(dfx[topGenes, ]),
  show_rownames = FALSE,
  show_colnames = FALSE,
  breaks = seq(-5, +5, length = 101),
  annotation_col = pData(x)[, c("sampleGroup", "Embryonic.day",
                                "ScanDate", "genotype")],
  annotation_colors = list(
    sampleGroup = groupColor,
    genotype = c(`FGF4-KO` = "chocolate1", `WT` = "azure2"),
    Embryonic.day = setNames(brewer.pal(9, "Blues")[c(3, 6, 9)],
                             c("E3.25", "E3.5", "E4.5")),
    ScanDate = setNames(brewer.pal(nlevels(x$ScanDate), "YlGn"),
                        levels(x$ScanDate))
  )
)
```

Section 3

Applying Heatmaps

MPG

Let's do it with a simple dataset we're familiar with.

```
mpg2 <- mpg %>%
  group_by(manufacturer, class) %>%
  mutate(avg_mpg = (cty + hwy) / 2) %>%
  summarize(avg_mpg = mean(avg_mpg)) %>%
  spread(class, avg_mpg) %>%
  column_to_rownames("manufacturer")
mpg2[is.na(mpg2)] = 0
mpg2[1:5, 1:6]
```

```
##           2seater  compact midsize  minivan   pickup subcompact
## audi          0.0 22.43333    20.0  0.00000  0.00000    0.00000
## chevrolet    20.1  0.00000    23.2  0.00000  0.00000    0.00000
## dodge         0.0  0.00000     0.0 19.09091 14.07895    0.00000
## ford          0.0  0.00000     0.0  0.00000 14.71429   19.55556
## honda         0.0  0.00000     0.0  0.00000  0.00000   28.50000
```

MPG

`pheatmap`(mpg2)
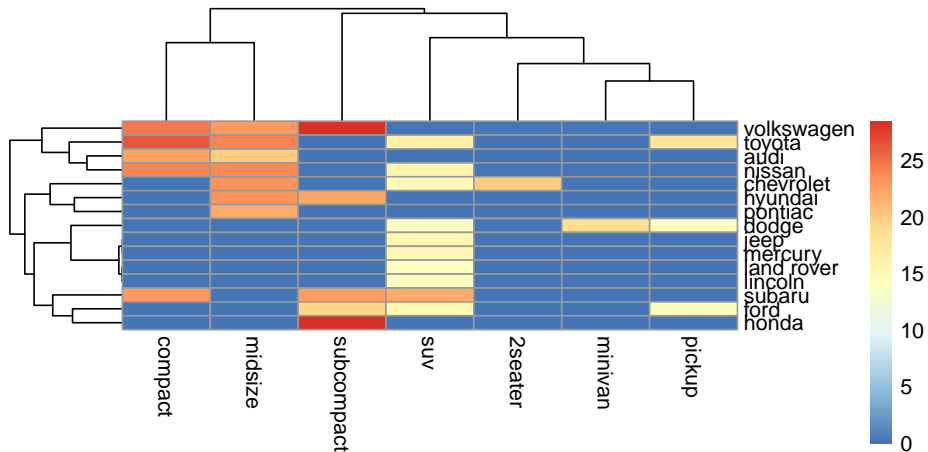
MPG

That didn't look great. Let's clean it up.

```
pheatmap(mpg2,
  angle_col = 45,
  color = colorRampPalette(
    c("gray", "red", "orange", "yellow", "darkgreen")
  )(10),
)
```

MPG
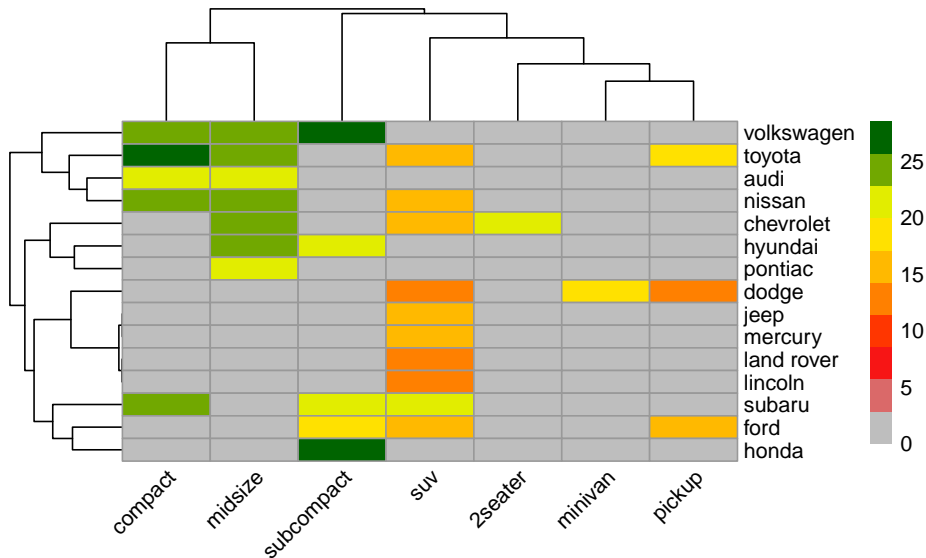
NYC13Flights

We want to plot the total number of hours flights were late arrivals each month, broken up by each carrier.

```
flights2 <- flights %>%
  filter(arr_delay > 0) %>%
  group_by(month, carrier) %>%
  summarize(total_delay = sum(arr_delay / 60)) %>%
  spread(carrier, total_delay) %>%
  column_to_rownames("month")
```

NYC13Flights

We can now plot it.

```
pheatmap(flights2)
```
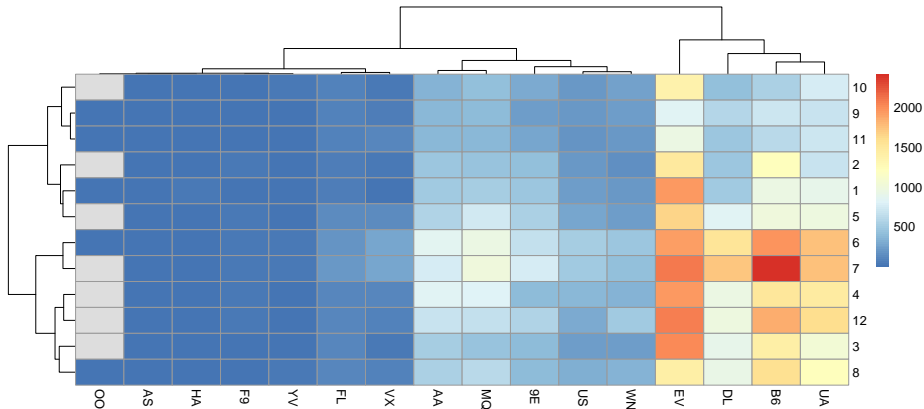
NYC13Flights

Additionally, we would like to overlay the total airtime (in days) for each month. We can do this like as well.

```r
airtime <- flights %>%
  filter(arr_delay > 0) %>%
  group_by(month) %>%
  summarize(airtime = sum(air_time / (60 * 24))) %>%
  column_to_rownames("month")


pheatmap(flights2, annotation_row = airtime)
```
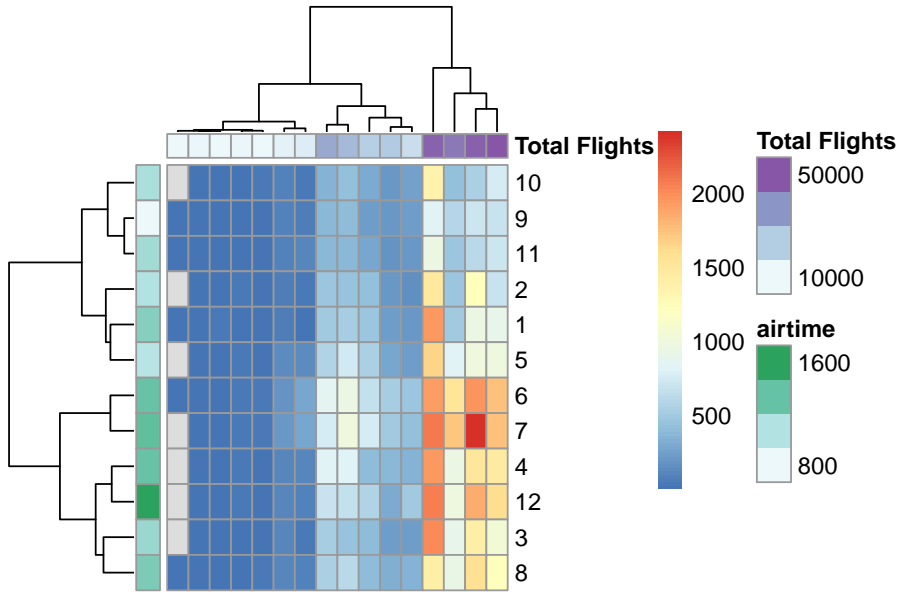
NYC13Flights
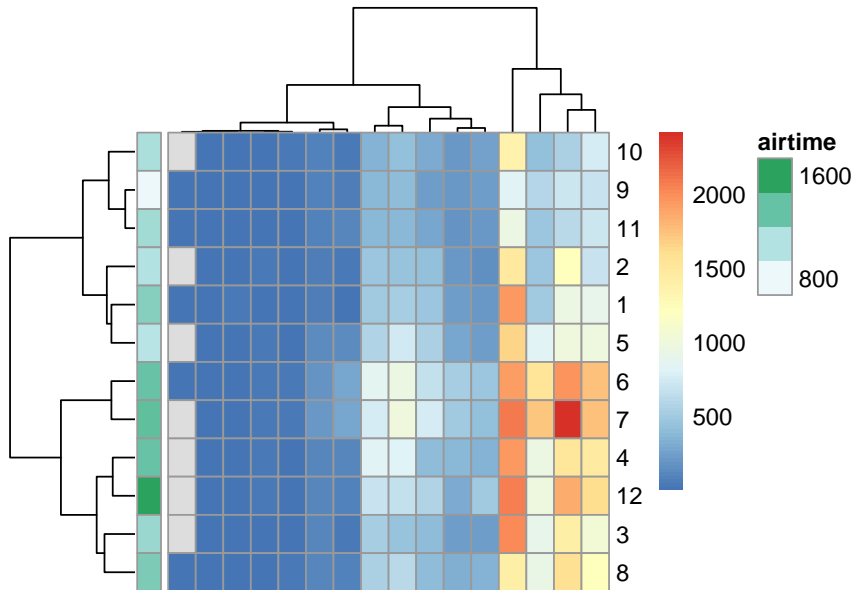
NYC13Flights

We also know certain airlines are preferred, so we would like to see the total
number of flights by airline.

```
flightcounts <- flights %>%
  group_by(carrier) %>%
  summarize(total_flights = n()) %>%
  column_to_rownames("carrier")

colnames(flightcounts) = "Total Flights"


pheatmap(flights2, annotation_row = airtime, annotation_col = flight
```
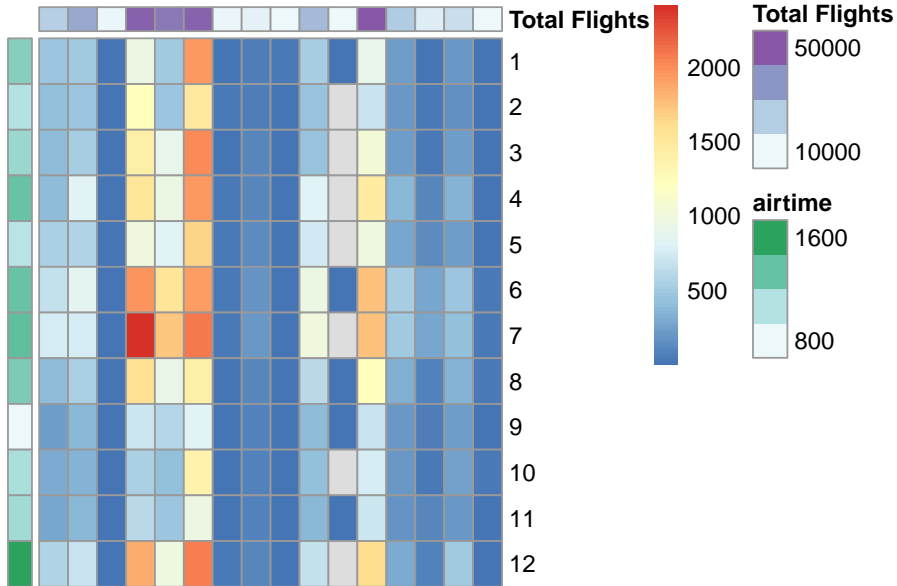
NYC13Flights

NYC13Flights

If we're not interested in the dendrogram, we can remove it easily.

```
pheatmap(flights2,
  annotation_row = airtime,
  annotation_col = flightcounts,
  cluster_row = FALSE,
  cluster_cols = FALSE
)
```
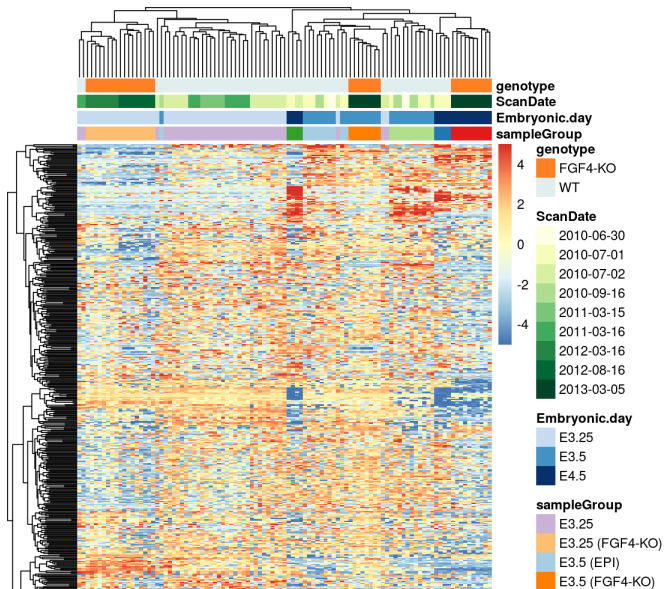
NYC13Flights

NYC13Flights

Finally, we can clean up our main legend, change the color scheme, and very clearly show missing values for our final figure.

```
pheatmap(flights2,
  annotation_row = airtime,
  annotation_col = flightcounts,
  cluster_row = FALSE,
  cluster_cols = FALSE,
  legend_breaks = c(150, 1200, 2400),
  legend_labels = c("Low", "Medium", "High"),
  color = RColorBrewer::brewer.pal(9, "PuRd"),
  na_col = "black"
)
```

NYC13Flights
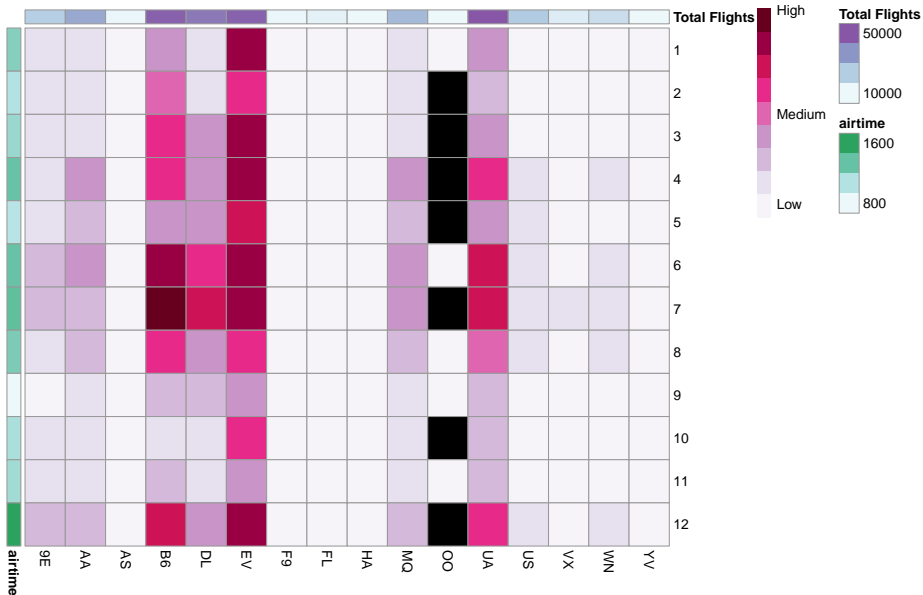
You can also choose to go bananas

Section 4

Wrapping up