Superheat Vignette

Rebecca Barter

Contents

Chapter 1

Downloading and installing the package

The `superheat` package was developed to produce customizable and extendable heatmaps which act as a tool for the visual exploration of complex datasets. Superheat enhances the traditional heatmap by providing a platform to visualize a wide range of data types simultaneously, adding to the heatmap a response variable as a scatterplot, model results as boxplots, correlation information as barplots, text information, and more. Superheat allows the user to explore their data to greater depths and to take advantage of the heterogeneity present in the data to inform analysis decisions.

The goal of this guide is to help you understand how to use the `superheat` package in R to visualize your data. First, you need to download and install the package. This can be done using the `devtools` package. If you have not yet done so, you will need to install it by simply typing the following code into your R console:

```
# install devtools
install.packages("devtools")
# use devtools to install superheat
devtools::install_github("rlbarter/superheat")
```

Next, load the `superheat` library into your workspace:

```
library(superheat)
```

Chapter 2

Basic Usage

The package consists of a single function: `superheat`.

The `superheat` function takes data objects, the most important of which are: - `X`: the heatmap matrix,

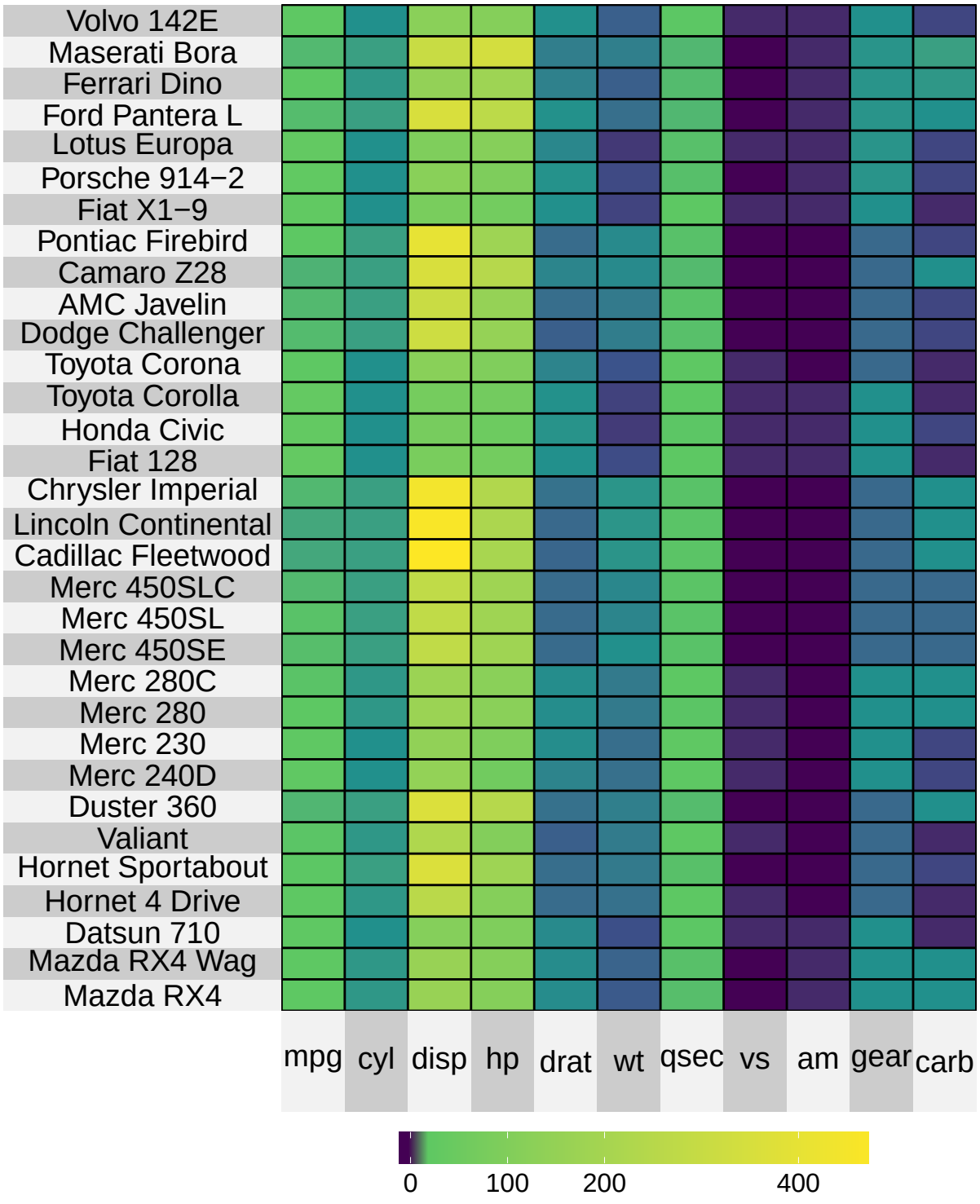
- (optional) `yr`: a vector of values to be plotted to the right of the heatmap,

- (optional) `yt`: a vector of values to be plotted above the heatmap.

As our running example, we will use superheat to visualize the mtcars dataset. For more complex examples, please see our accompanying website: https://rlbarter.github.io/superheat-examples/

A simple visualization without any additional arguments is presented below.

```
superheat(mtcars,
          # change the size of the labels
          left.label.size = 0.4,
          bottom.label.size = 0.1)
```

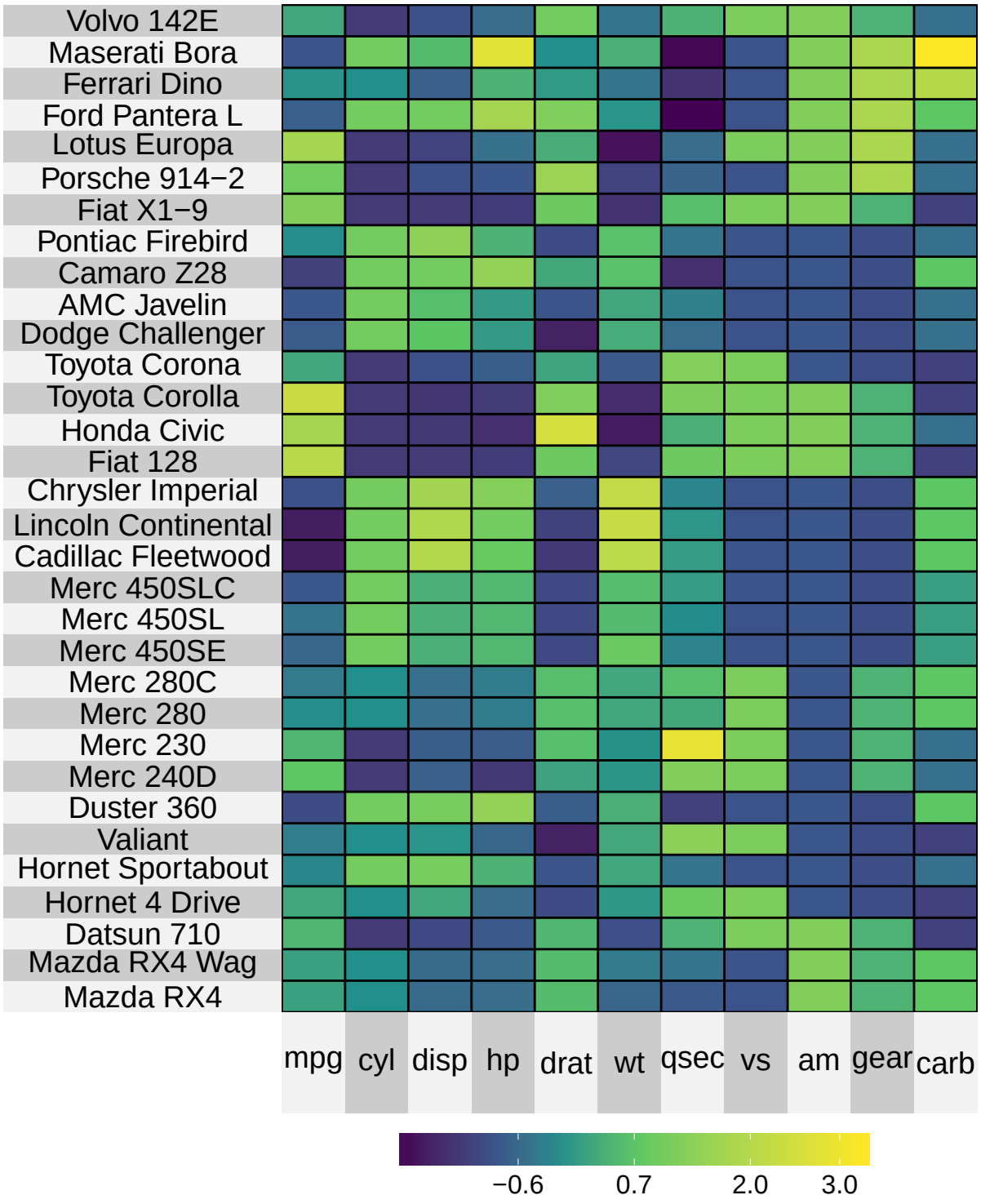


2.1 Heatmap scale

Notice that the variables in the mtcars dataset presented above each have very different scales, making comparisons between cars difficult. Fortunately, it is easy to scale the columns of the matrix (to mean 0 and standard deviation 1) using the `scale` argument.

It is important to be aware that they way in which you scale your data can alter the interpretation of your heatmap. It is always a good idea to scale the input matrix yourself using the method that makes the most sense for your data, for example by converting your data to a [0,1] quantile-preserving scale, or simply by mean-centering.

```
superheat(mtcars,
          # change the size of the labels
          left.label.size = 0.4,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE)
```


Volvo 142E
Maserati Bora
Ferrari Dino
Ford Pantera L
Lotus Europa
Porsche 914−2
Fiat X1−9
Pontiac Firebird
Camaro Z28
AMC Javelin
Dodge Challenger
Toyota Corona
Toyota Corolla
Honda Civic
Fiat 128
Chrysler Imperial
Lincoln Continental
Cadillac Fleetwood
Merc 450SLC
Merc 450SL
Merc 450SE
Merc 280C
Merc 280
Merc 230
Merc 240D
Duster 360
Valiant
Hornet Sportabout
Hornet 4 Drive
Datsun 710
Mazda RX4 Wag
Mazda RX4
mpg
cyl
disp
hp
drat
wt
qsec
vs
am
gear
carb
−0.6
0.7
2.0
3.0

Chapter 3

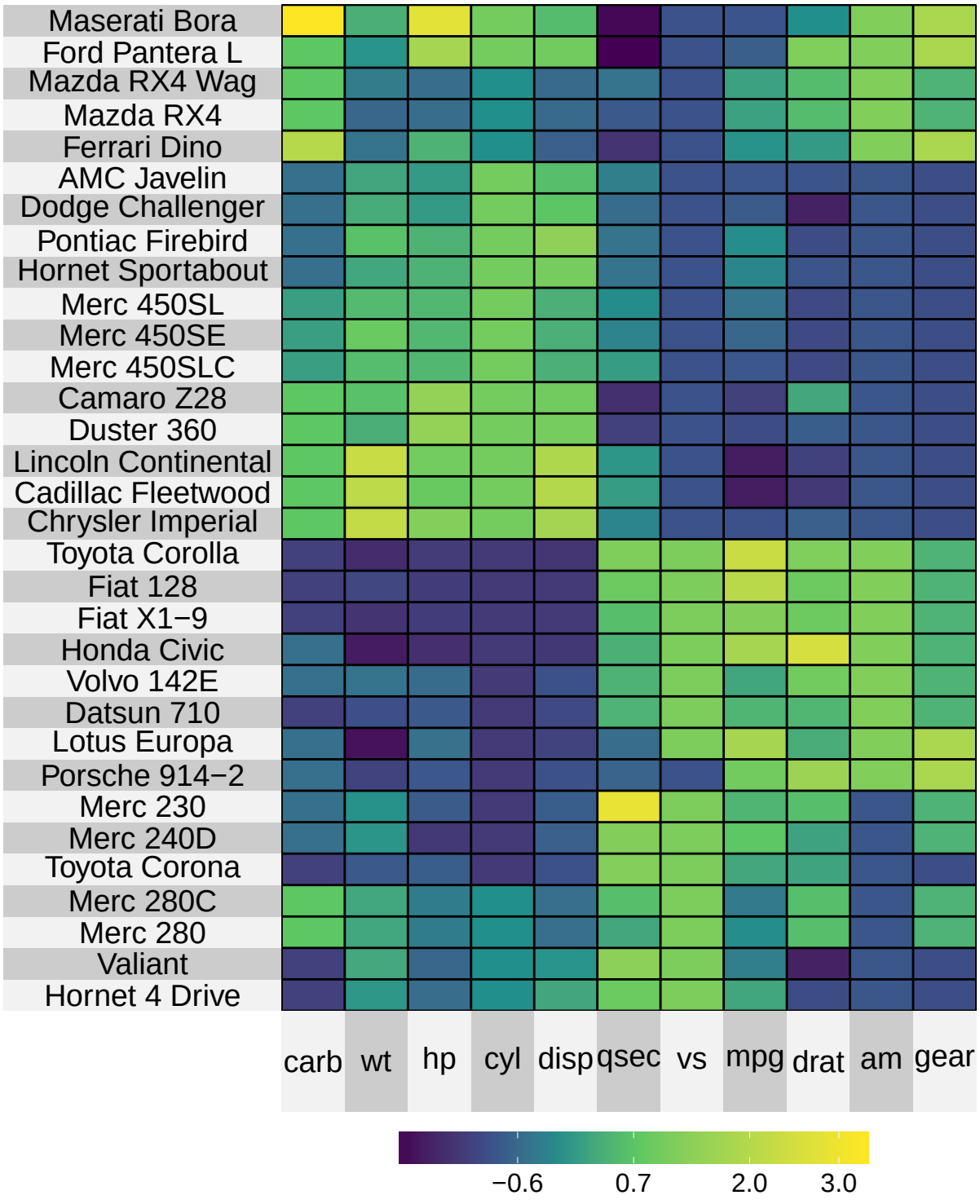
Ordering rows and columns

By default, the rows and columns are ordered as in the original input.

3.1 Using hierarchical clustering to order the rows/columns

You can set the ordering of the rows/columns based on a "pretty" hierarchical clustering by specifying `pretty.order.rows = TRUE` and `pretty.order.cols = TRUE`. Errors may arise when the matrix has missing values.

```
# generate the plot:
superheat(mtcars,
          # retain original order of rows/cols
          pretty.order.rows = TRUE,
          pretty.order.cols = TRUE,
          # change the size of the labels
          left.label.size = 0.4,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE)
```

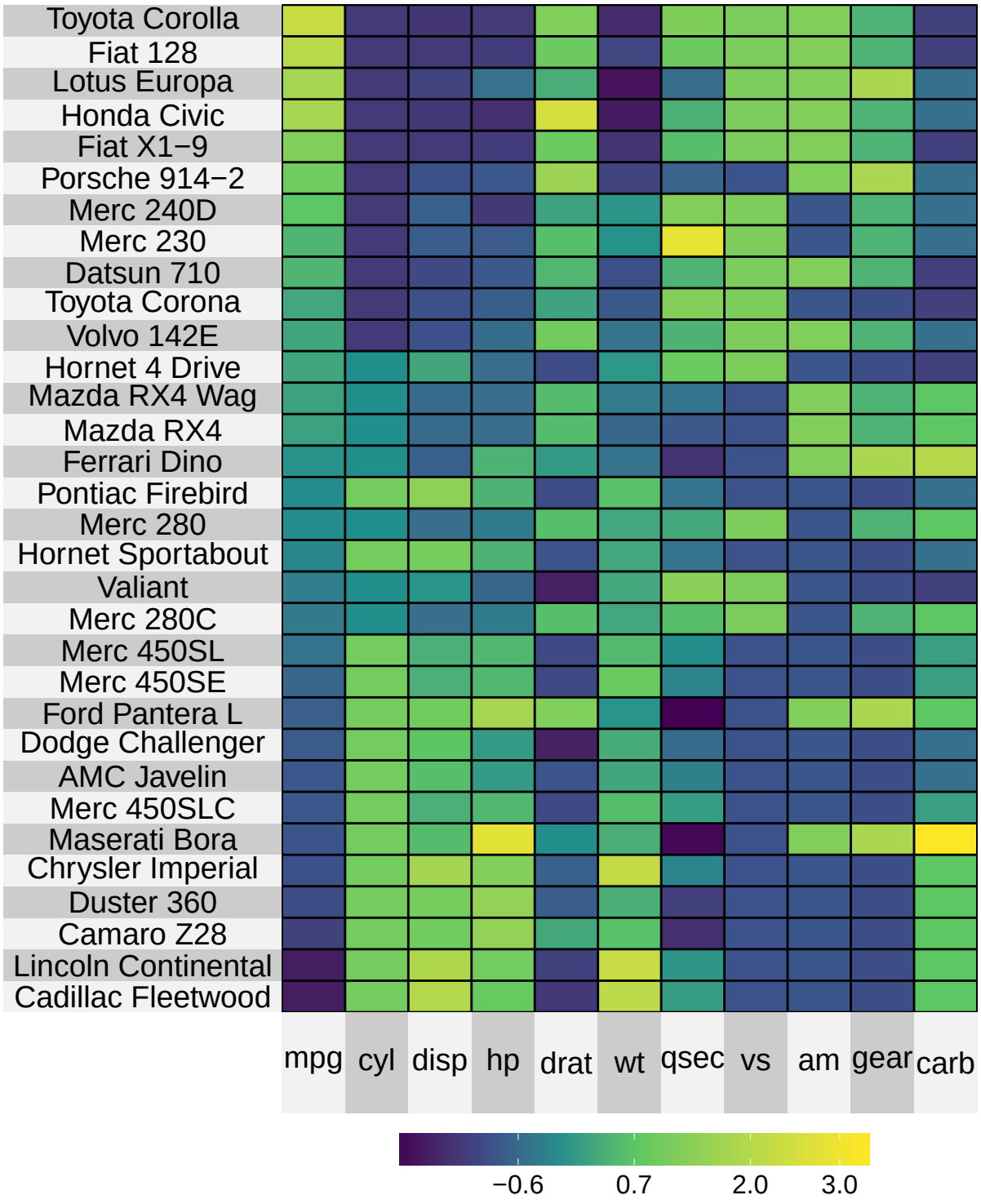


3.2 Specifying the ordering of the columns or rows

If, instead, you would like to specify a custom ordering of the rows and columns, you can simply provide the order vector to the `order.rows` and `order.cols` arguments. In the example below, we order the rows by the `mpg` variable from the original matrix. Note that we could order by any vector i.e. we are not restricted to

ordering by a column or row in our matrix.

```
# generate the plot:
superheat(mtcars,
          # order the rows by miles per gallon
          order.rows = order(mtcars$mpg),
          # change the size of the labels
          left.label.size = 0.4,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE)
```


Toyota Corolla
Fiat 128
Lotus Europa
Honda Civic
Fiat X1−9
Porsche 914−2
Merc 240D
Merc 230
Datsun 710
Toyota Corona
Volvo 142E
Hornet 4 Drive
Mazda RX4 Wag
Mazda RX4
Ferrari Dino
Pontiac Firebird
Merc 280
Hornet Sportabout
Valiant
Merc 280C
Merc 450SL
Merc 450SE
Ford Pantera L
Dodge Challenger
AMC Javelin
Merc 450SLC
Maserati Bora
Chrysler Imperial
Duster 360
Camaro Z28
Lincoln Continental
Cadillac Fleetwood
mpg
cyl
disp
hp
drat
wt
qsec
vs
am
gear
carb
−0.6
0.7
2.0
3.0

Chapter 4

Heatmap colormap

4.1 Heatmap Palette

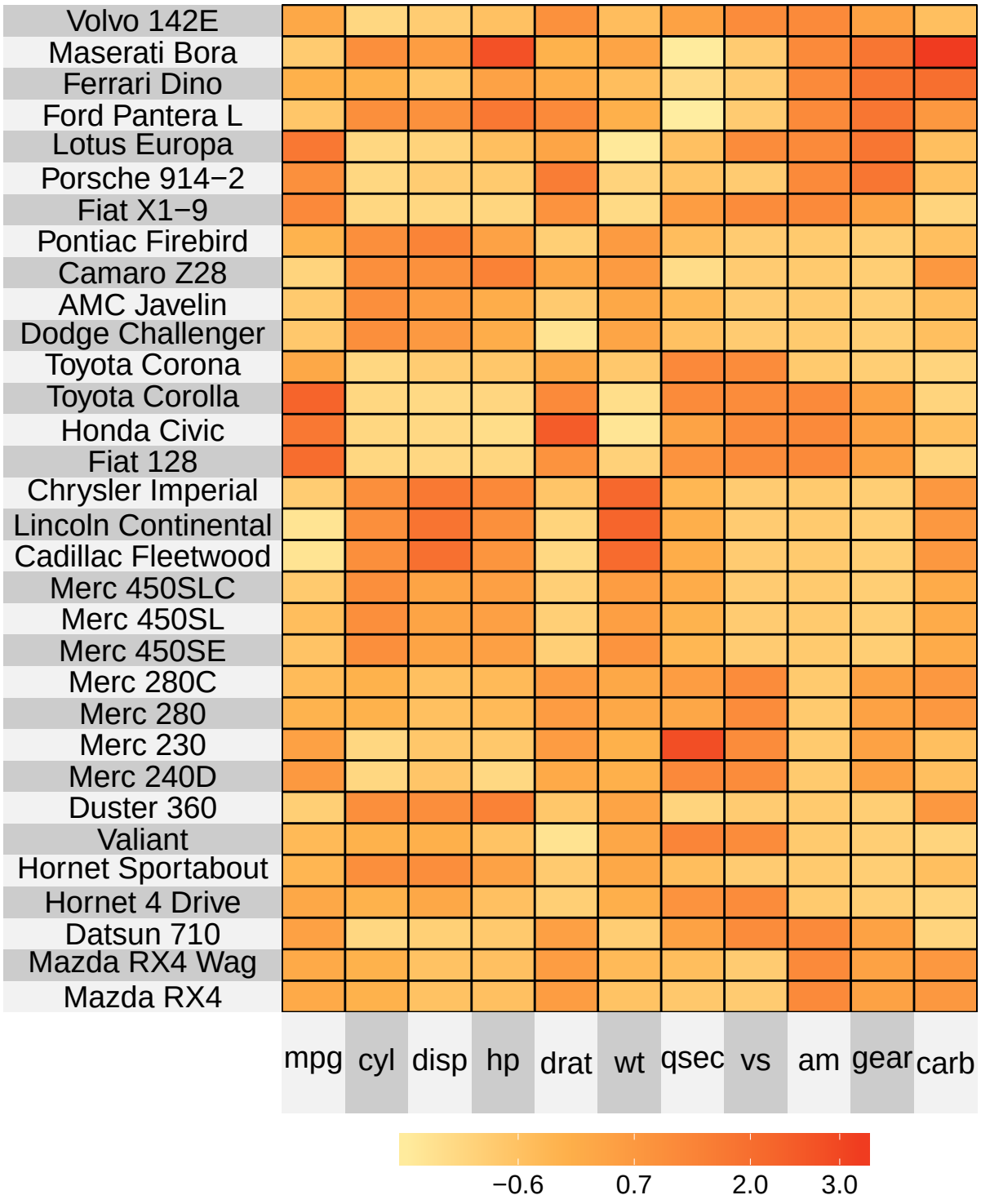
The default color palette is the **viridis** color map generated by Nathaniel Smith and Stéfan van der Walt. If for some reason, however, you'd like to change the color palette of your heatmap, you're in luck! Simply evoke one of the two following arguments:

- `heat.pal`: if you'd like to make your own color palette, or

- `heat.col.scheme`: if you'd like to select a color palette from the set of inbuilt choices.

For example, if you'd like to use the `red` inbuilt color palette, you can set `heat.col.scheme = "red"`:

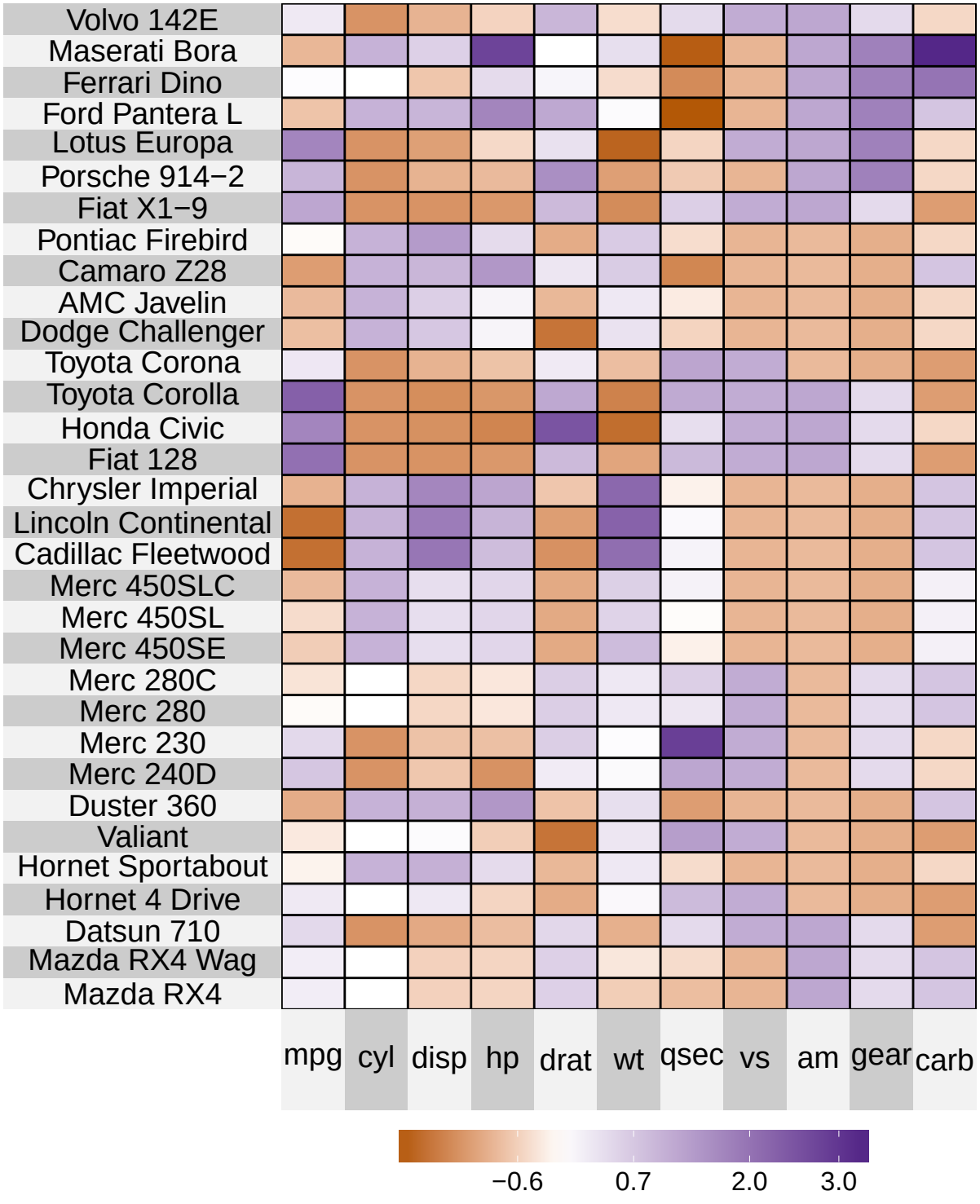
```
superheat(mtcars,
          # change the size of the labels
          left.label.size = 0.4,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE,
          # change the color
          heat.col.scheme = "red")
```



If you'd like your color palette to go from brown to purple by travelling through white, then you can simply set `heat.pal = c("brown", "white", "purple")` as follows

```
superheat(mtcars,
          # change the size of the labels
          left.label.size = 0.4,
          bottom.label.size = 0.1,
          # scale the matrix columns
```

```
          scale = TRUE,
          # change the color (#b35806 = brown and #542788 = purple)
          heat.pal = c("#b35806", "white", "#542788"))
```

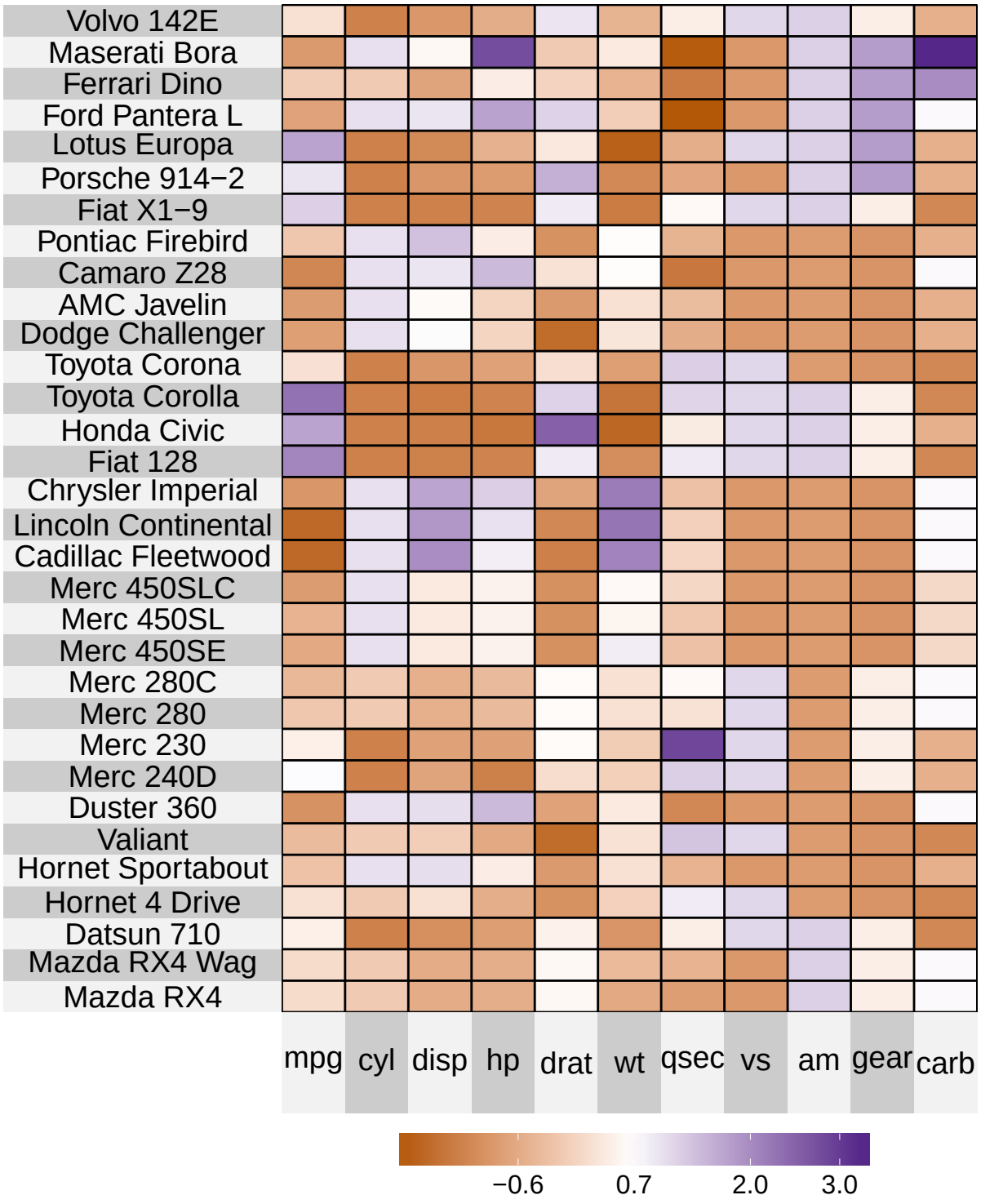


4.2 Color transitions

Note that by default, the color transitions take place at the appropriate quantile based on the number of colors provided in the palette. For example, if you have 6 colors the color transitions will be at the 0th, 20th, 40th, 60th, 80th and 100th quantiles, where the 0th quantile corresponds to the minimum value and the 100th quantile corresponds to the maximum value. For skewed data, this means that most of the color transitions will occur towards one end of the scale.

To force the transition to occur at a particular location, you need to use the `heat.pal.values` argument. This argument takes a vector whose length is the same as `heat.pal` and specifies the position (within the range from 0 to 1) of each color. For example `heat.pal.values = c(0, 0.5, 1)` forces the first color to be at the minimum value, the second color to be exactly at the midpoint of the range (note: this is distinct from the median) and the last color to be at the maximum value.

```
superheat(mtcars,
          # change the size of the labels
          left.label.size = 0.4,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE,
          # change the color (#b35806 = brown and #542788 = purple)
          heat.pal = c("#b35806", "white", "#542788"),
          heat.pal.values = c(0, 0.5, 1))
```



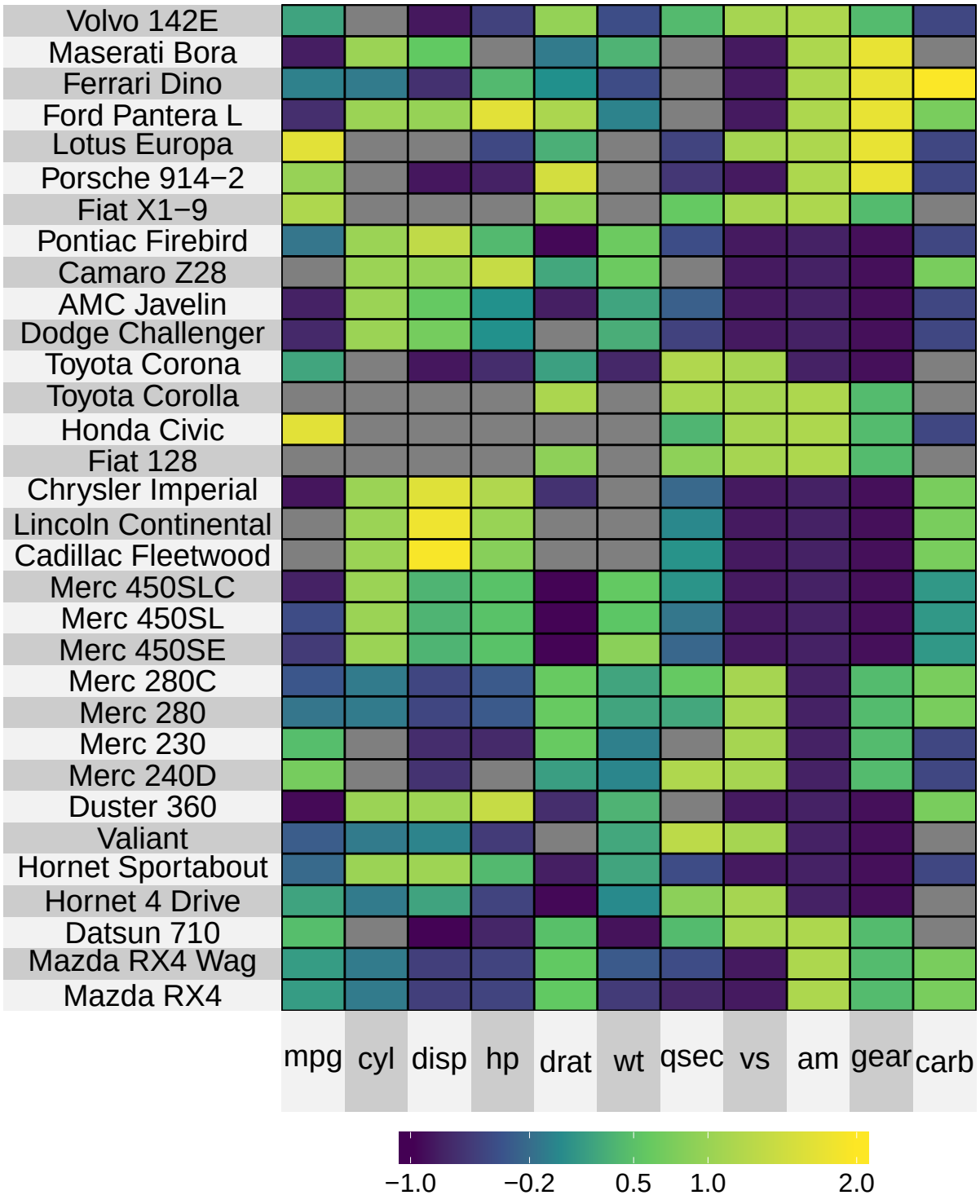
4.3 Color limits

It is possible to specify the minimum and maximum value for which you would like your colormap to be defined using the `heat.lim` argument. For example, if I would like to display only values from -1 to 2, then I would set `heat.lim = c(-1, 2)`. This means that

- each value outside this range will be presented as missing (i.e. a grey cell corresponding to `NA`), and

- the first color in the color range will correspond to the value -1 and the last color in the color range will correspond to the value 2.

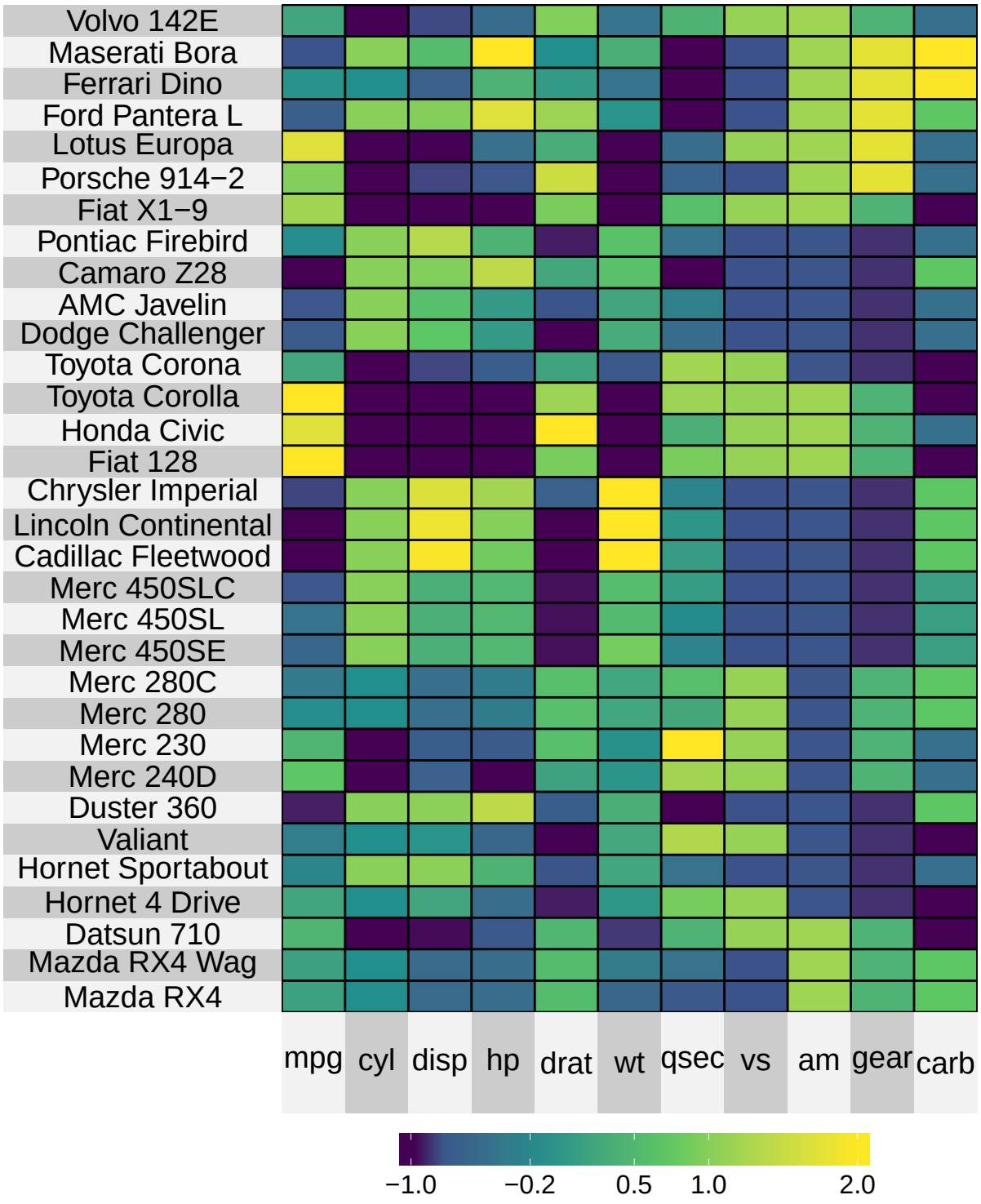
```
superheat(mtcars,
          # change the size of the labels
          left.label.size = 0.4,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE,
          heat.lim = c(-1, 2))
```



4.3.1 Extreme values

If you would prefer that values outside the `heat.lim` range be presented as the maximum/minimum color in the range (rather than as `NA`) then you can specify the argument `extreme.values.na = FALSE`.

```r
superheat(mtcars,
          # change the size of the labels
          left.label.size = 0.4,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE,
          heat.lim = c(-1, 2),
          extreme.values.na = FALSE)
```

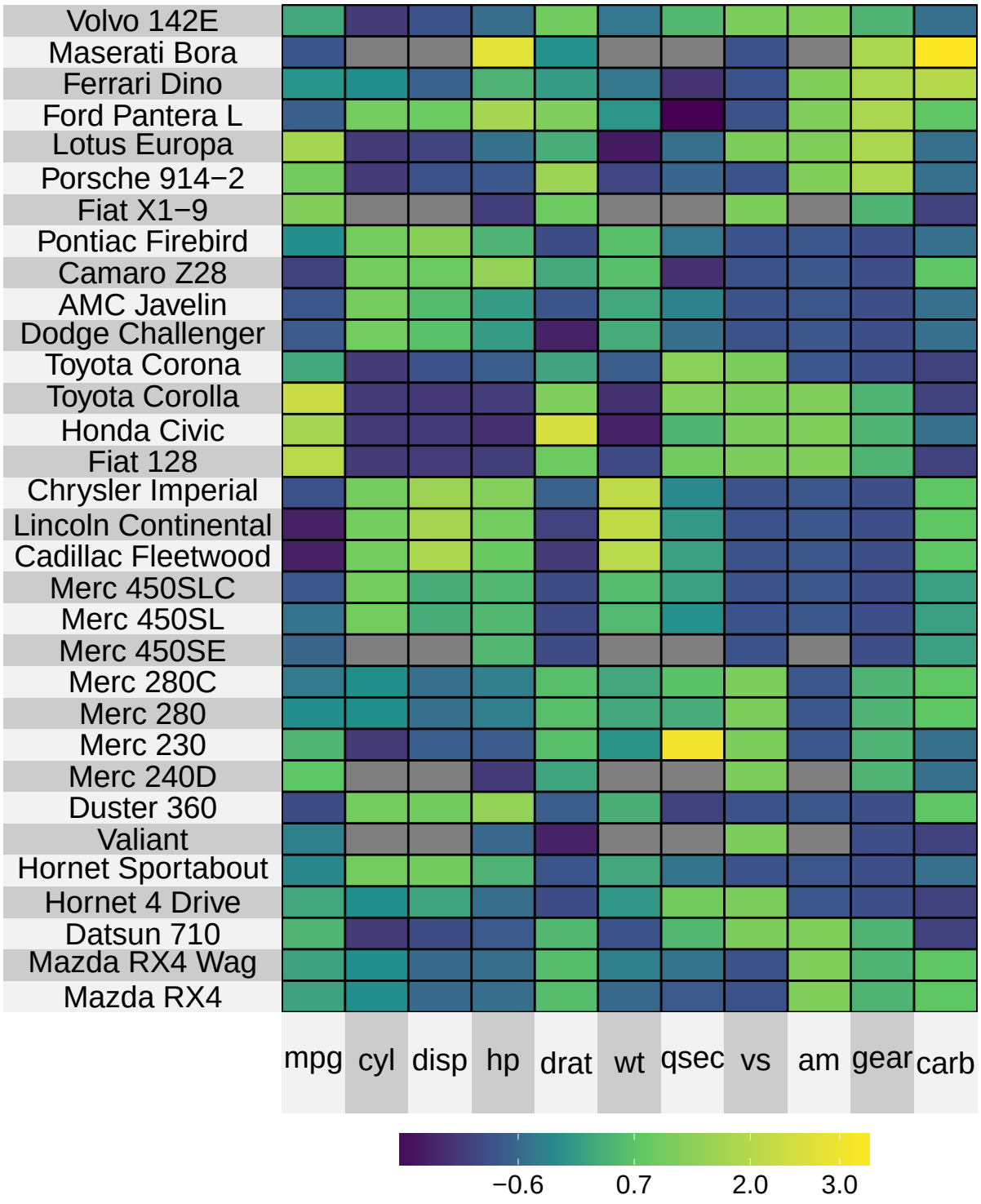

Volvo 142E
Maserati Bora
Ferrari Dino
Ford Pantera L
Lotus Europa
Porsche 914−2
Fiat X1−9
Pontiac Firebird
Camaro Z28
AMC Javelin
Dodge Challenger
Toyota Corona
Toyota Corolla
Honda Civic
Fiat 128
Chrysler Imperial
Lincoln Continental
Cadillac Fleetwood
Merc 450SLC
Merc 450SL
Merc 450SE
Merc 280C
Merc 280
Merc 230
Merc 240D
Duster 360
Valiant
Hornet Sportabout
Hornet 4 Drive
Datsun 710
Mazda RX4 Wag
Mazda RX4
mpg
cyl
disp
hp
drat
wt
qsec
vs
am
gear
carb
−1.0
−0.2
0.5
1.0
2.0

Chapter 5

Missing data

Superheat deals with missing values gracefully by filling them in with a color of your choice (the default is grey).

```
library(superheat)
# replace some values with missing values
mtcars.missing <- mtcars
mtcars.missing[sample(1:nrow(mtcars), 5),
               sample(1:ncol(mtcars), 5)] <- NA

superheat(mtcars.missing,
          # change the size of the labels
          left.label.size = 0.4,
          bottom.label.size = 0.1,
          # scale the matrix
          scale = T)
```

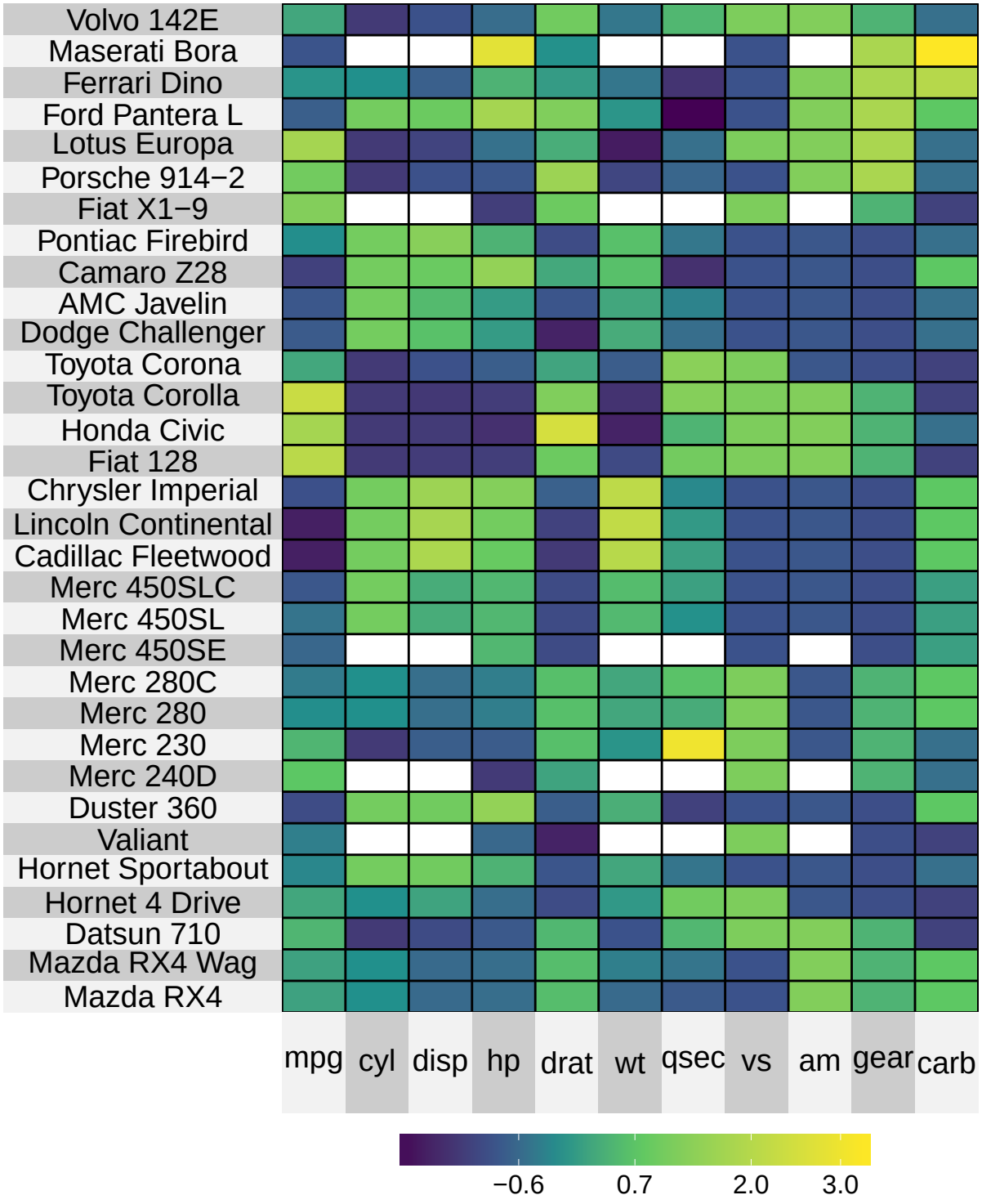


5.1 Color

You can set the color of missing values by setting `heat.na.col` to a color of your choice.

```
superheat(mtcars.missing,
          # change the size of the labels
```

```
        left.label.size = 0.4,
        bottom.label.size = 0.1,
        # scale the matrix
        scale = T,

        # change color of missing values
        heat.na.col = "white")
```



Chapter 6

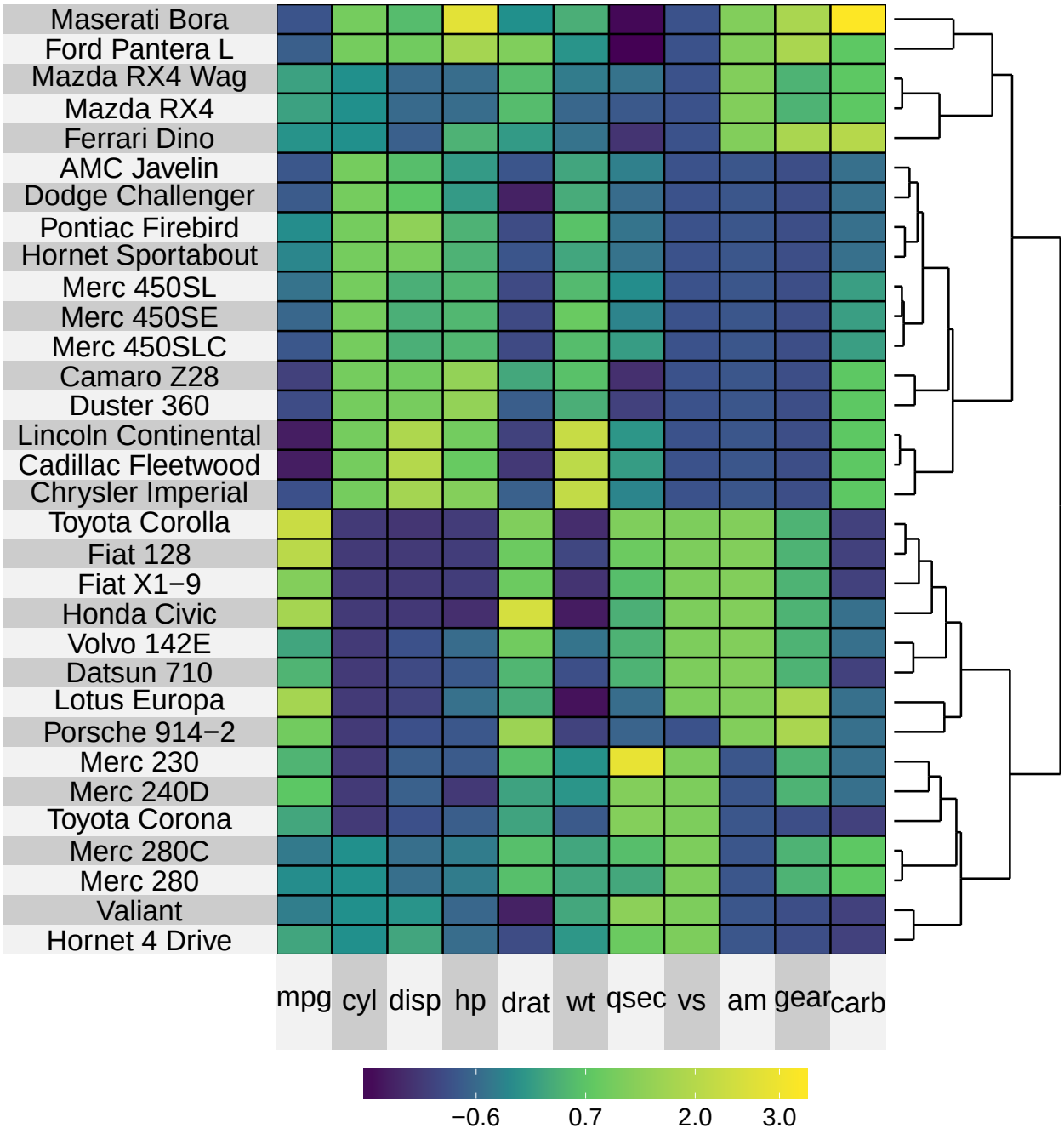
Clustering

The default option is to arrange the rows and columns in the same order as they are provided. We saw in a previous section that it is easy to arrange the rows and columns into a clustered formation using the `pretty.order.rows` and `pretty.order.cols` arguments.

6.1 Dendrogram

It is natural to supply a dendrogram that highlights the hierarchical clustering of the columns and/or rows using the `col.dendrogram` and `row.dendrogram` arguments. Note that if you want to implement the row or column ordering implied by the dendrogram, but to remove the dendrogram itself, you can use the `pretty.order.rows` and `pretty.order.cols` arguments.

```
superheat(mtcars,
          # change the size of the labels
          left.label.size = 0.45,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE,
          # add row dendrogram
          row.dendrogram = TRUE)
```

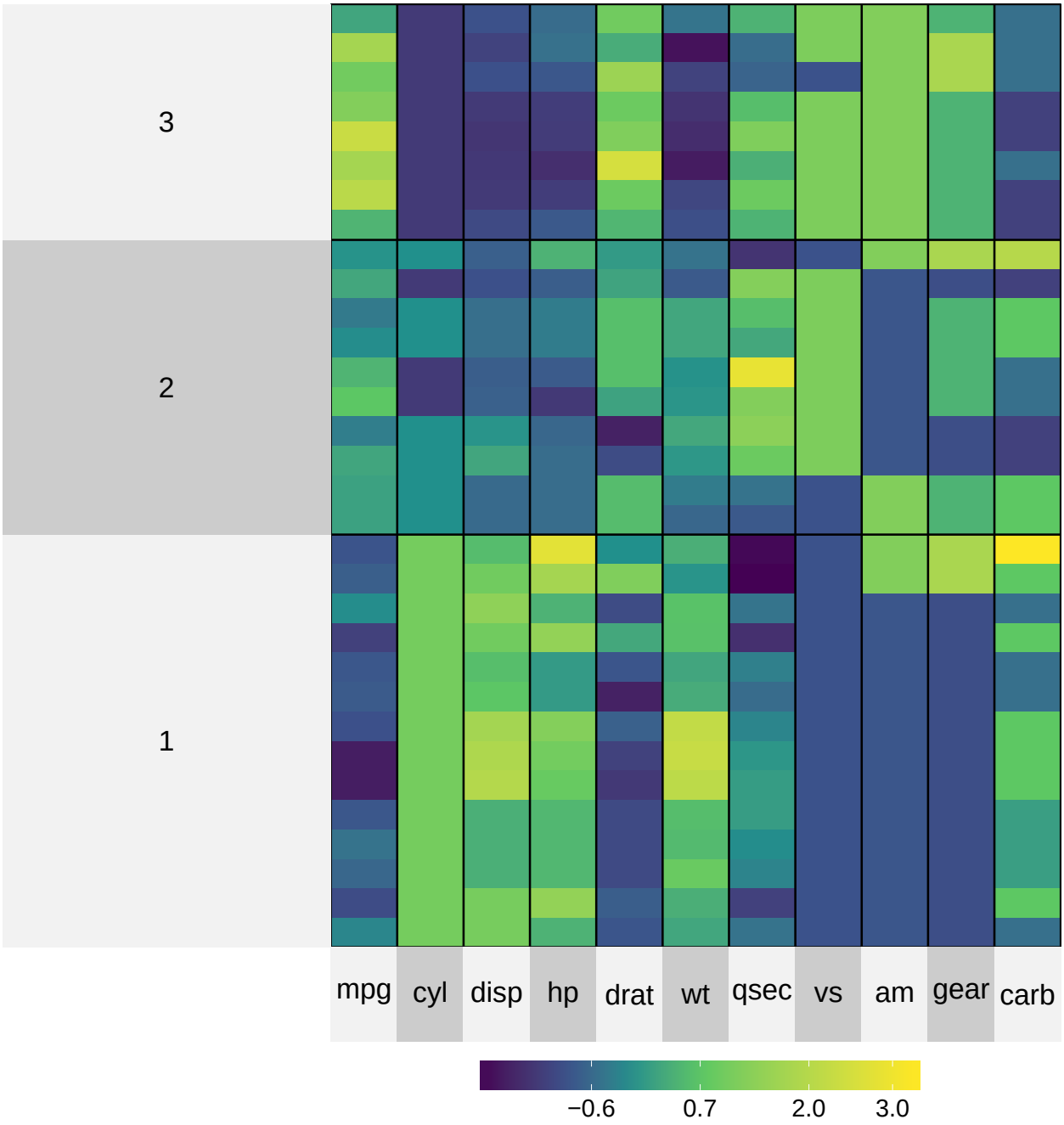


6.2 Generating clusters

Grouping the rows and/or columns into a pre-specified number of clusters is a nice way to highlight structure and simplify visualization. For example, we can group the rows into three groupings by specifying `n.clusters.rows = 3`. The underling clustering algorithm is `kmeans()`, but you can use hierarchical clustering by specifying `clustering.method = 'hierarchical'`.

In order to get the same clustering every time you must set the seed or provide your own clustering membership vector.

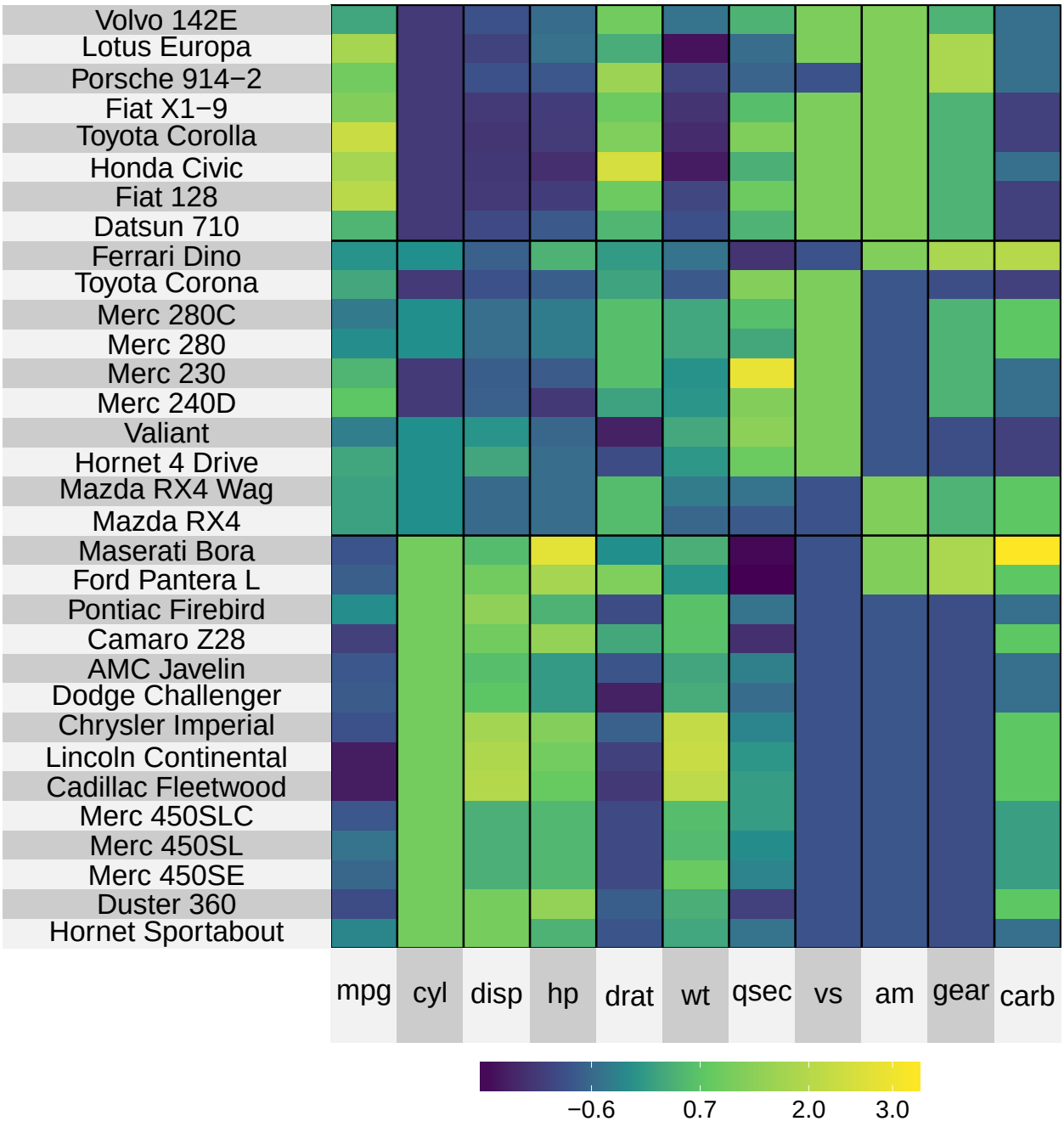
```
set.seed(2016113)
superheat(mtcars,
          # change the size of the labels
          left.label.size = 0.45,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE,
          # generate three column clusters
          n.clusters.rows = 3)
```



By default, when clustering the corresponding labels are grouped into the cluster name (typically 1, 2, 3, ...

etc). If you would like to force the labels to be the original variable names, you can specify `left.label = 'variable'` or `bottom.label = 'variable`, depending on whether it is the left or bottom labels, respectively.

```
set.seed(2016113)
superheat(mtcars,
          # change the size of the labels
          left.label.size = 0.45,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE,
          # generate three column clusters
          n.clusters.rows = 3,
          left.label = 'variable')
```



6.2.1 Extracting the clusters

If you would like to be able to extract the clusters generated by the `superheat()` function, then you need to first save the superheat object as a variable. From this variable, you can access the clusters that are stored in the `membership.rows` and `membership.cols` entries.

```
set.seed(2016113)
superheatmap <- superheat(mtcars,
                          # change the size of the labels
                          left.label.size = 0.45,
                          bottom.label.size = 0.1,
```

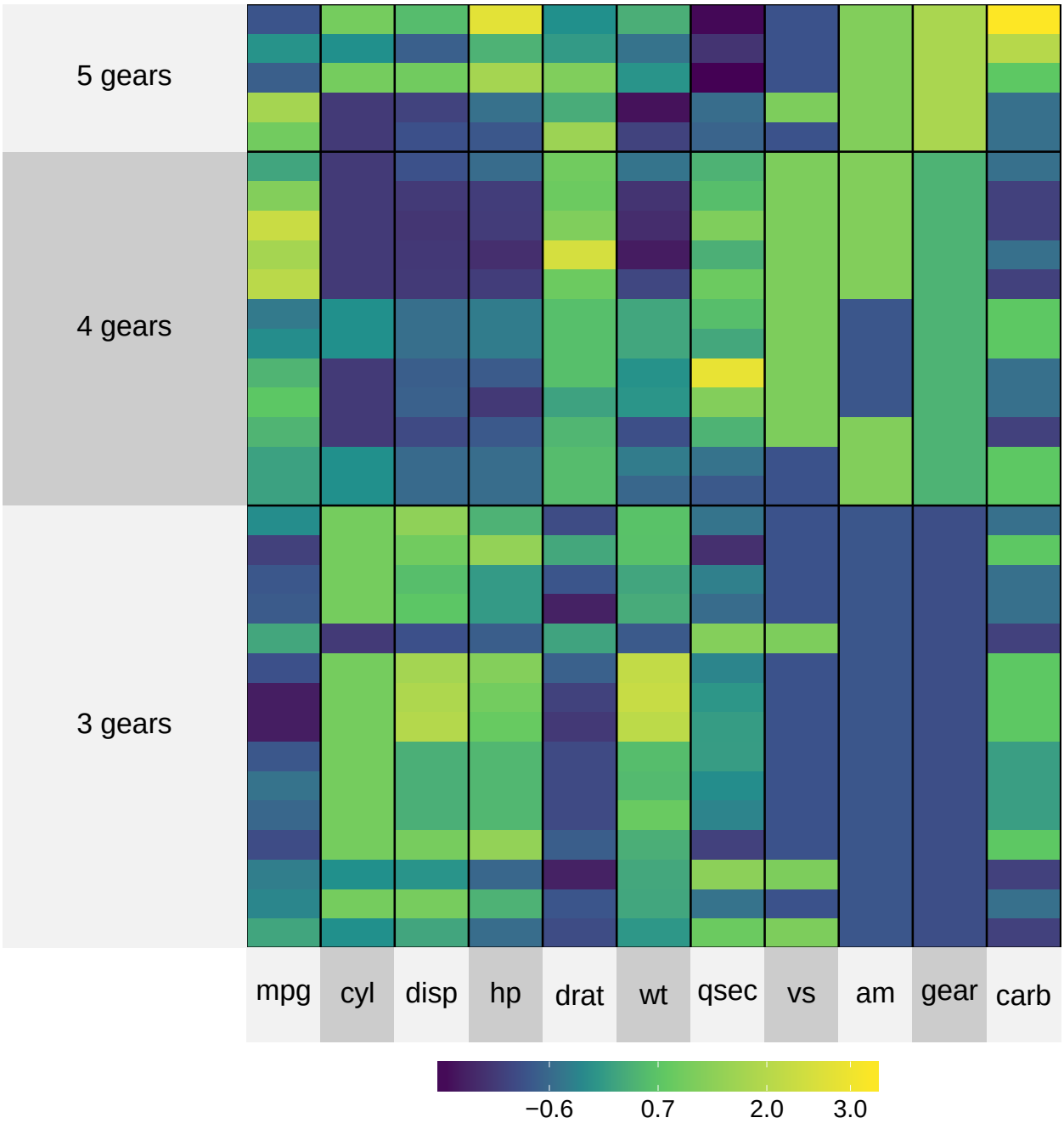
```
                          # scale the matrix columns
                          scale = TRUE,
                          # generate three column clusters
                          n.clusters.rows = 3,
                          left.label = 'variable',
                          print.plot = F)

# extract the clusters
superheatmap$membership.rows
```

```
##   Hornet Sportabout          Duster 360          Merc 450SE
##                   1                   1                   1
##          Merc 450SL         Merc 450SLC  Cadillac Fleetwood
##                   1                   1                   1
## Lincoln Continental   Chrysler Imperial    Dodge Challenger
##                   1                   1                   1
##         AMC Javelin          Camaro Z28    Pontiac Firebird
##                   1                   1                   1
##      Ford Pantera L       Maserati Bora           Mazda RX4
##                   1                   1                   2
##       Mazda RX4 Wag      Hornet 4 Drive             Valiant
##                   2                   2                   2
##           Merc 240D            Merc 230            Merc 280
##                   2                   2                   2
##           Merc 280C       Toyota Corona        Ferrari Dino
##                   2                   2                   2
##          Datsun 710            Fiat 128         Honda Civic
##                   3                   3                   3
##      Toyota Corolla           Fiat X1-9       Porsche 914-2
##                   3                   3                   3
##        Lotus Europa          Volvo 142E
##                   3                   3
```

6.3 User-supplied clusters

The best way to conduct clustering on your matrix is to provide a pre-specified membership vector using the `membership.rows`/`membership.cols` argument. Suppose, for our mtcars example, we wanted to group by number of gears.

```
gears <- paste(mtcars$gear, "gears")

set.seed(2016113)
superheat(mtcars,
          # change the size of the labels
          left.label.size = 0.3,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE,
          # cluster by gears
          membership.rows = gears)
```


5 gears
4 gears
3 gears
mpg
cyl
disp
hp
drat
wt
qsec
vs
am
gear
carb
−0.6
0.7
2.0
3.0

Chapter 7

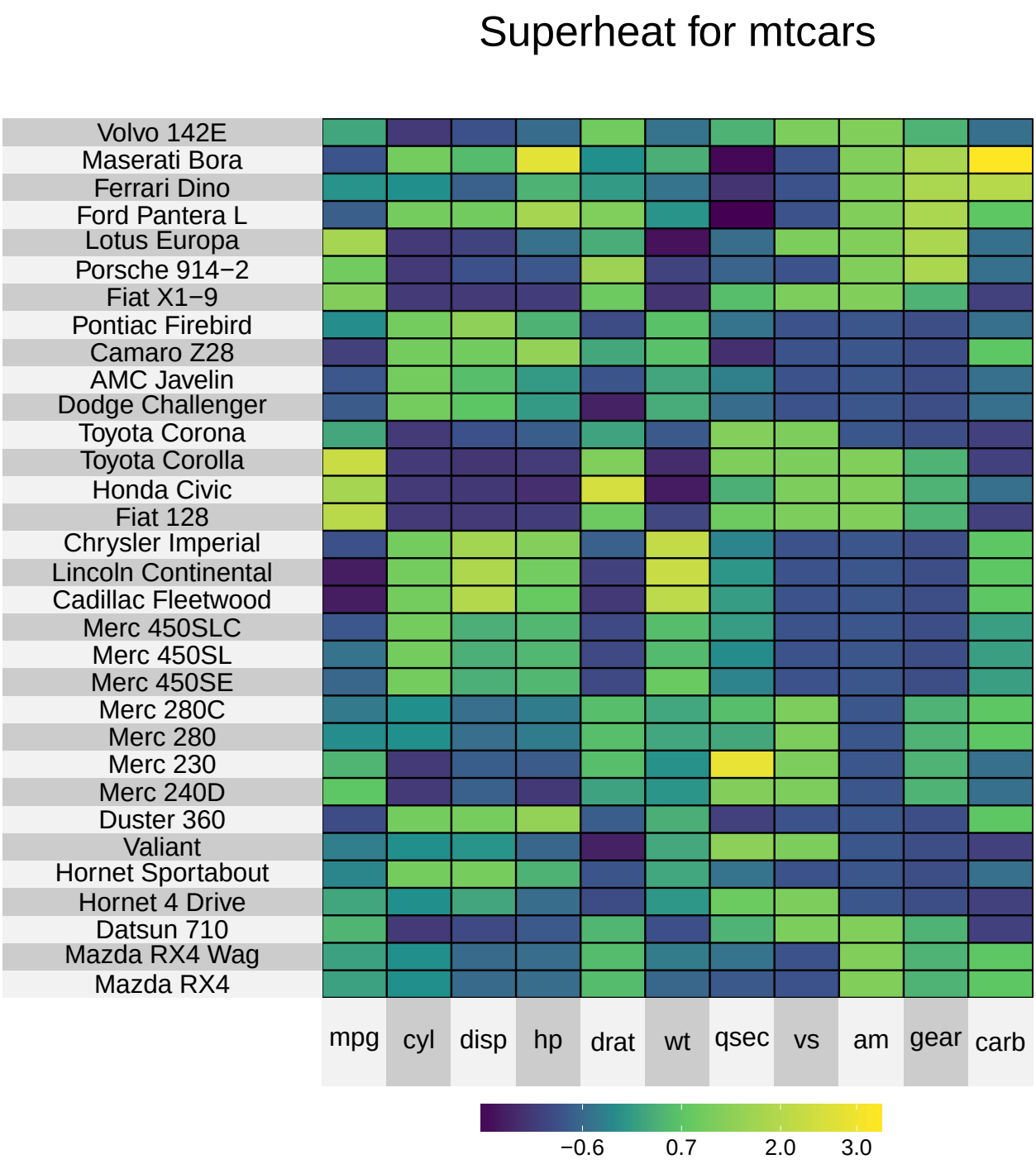
Titles

Adding row, column and plot titles to your heatmap is easy.

7.1 Plot title

Plot titles are very important for presenting your visualizations. You can set the plot title by `title`.

```
superheat(mtcars,
          # change the size of the labels
          left.label.size = 0.45,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE,
          # plot title
          title = "Superheat for mtcars",
          title.size = 8)
```

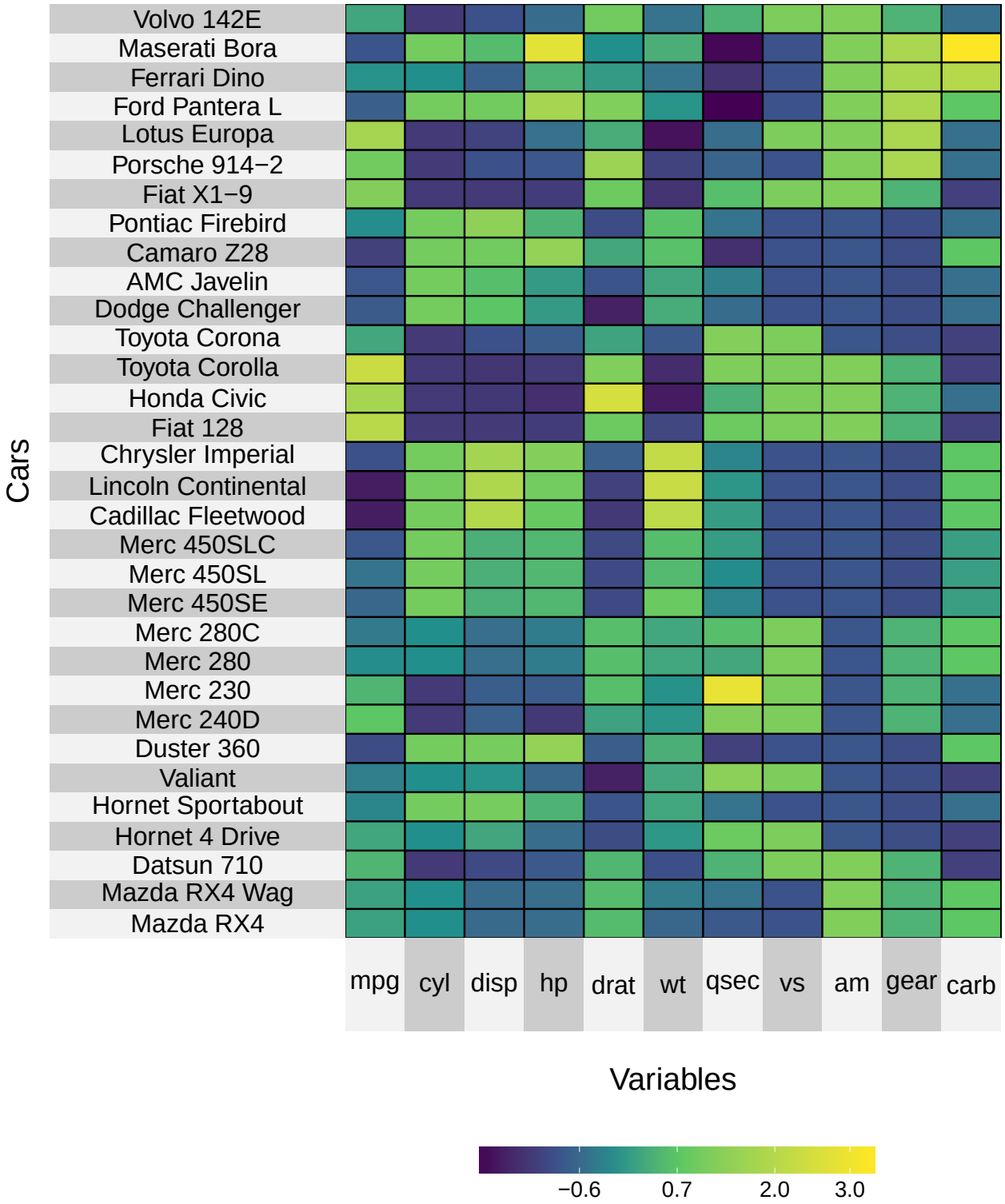


7.2 Row and column titles

Adding titles to the columns and rows is easy too. Simply supply the desired row and column titles for the `row.title` and `column.title` arguments

```
superheat(mtcars,
          # change the size of the labels
```

```
          left.label.size = 0.45,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE,
          # row title
          row.title = "Cars",
          row.title.size = 6,
          # col title
          column.title = "Variables",
          column.title.size = 6)
```


Volvo 142E
Maserati Bora
Ferrari Dino
Ford Pantera L
Lotus Europa
Porsche 914−2
Fiat X1−9
Pontiac Firebird
Camaro Z28
AMC Javelin
Dodge Challenger
Toyota Corona
Toyota Corolla
Honda Civic
Fiat 128
Chrysler Imperial
Lincoln Continental
Cadillac Fleetwood
Merc 450SLC
Merc 450SL
Merc 450SE
Merc 280C
Merc 280
Merc 230
Merc 240D
Duster 360
Valiant
Hornet Sportabout
Hornet 4 Drive
Datsun 710
Mazda RX4 Wag
Mazda RX4
Cars
mpg
cyl
disp
hp
drat
wt
qsec
vs
am
gear
carb
Variables
−0.6
0.7
2.0
3.0

Chapter 8

Adjacent plots

Adding adjacent plots to the heatmap is easy with superheat using the `yt` ('y top') and `yr` ('y right'). `yr` and `yt` must have the same length as either:

1. the number of rows/columns, or
2. the number of row clusters/column clusters (for scatterplots, barplots, and boxplots only).

The plot types available for the adjacent plots are

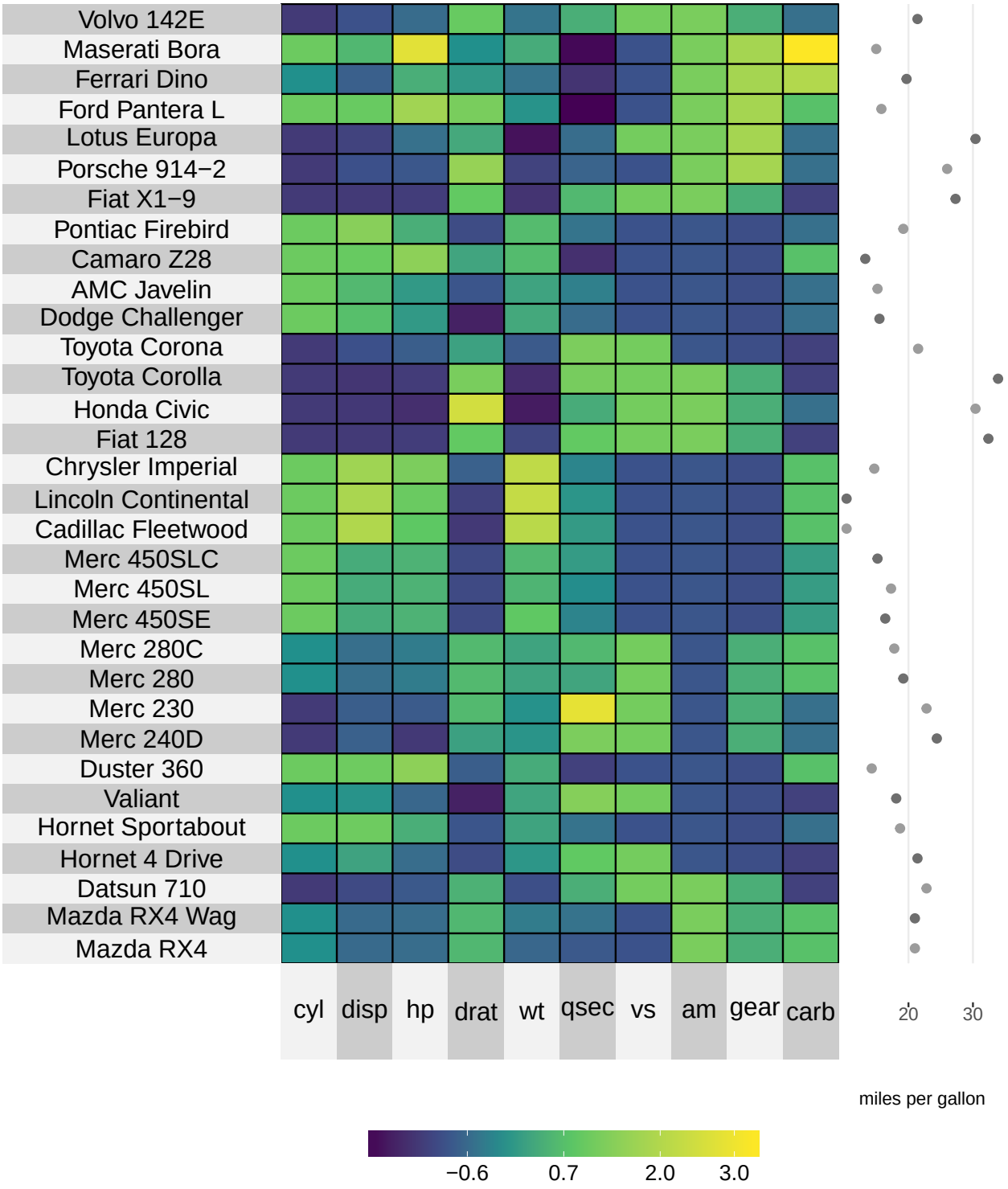
- `scatter`: scatterplot (default)
- `line`: line plot
- `smooth`: smoothed line
- `scattersmooth`: scatterplot with smoothed line
- `scatterline`: scatterplot with connecting lines
- `bar`: barplot
- `boxplot`: boxplot (with clusters)

The plot type can be specified using `yt.plot.type = 'line'`, for example.

8.1 Scatterplots

The following example adds the miles per gallon (`mpg`) variable as a scatterplot next to the rows, and then orders the rows by the `mpg` variable. The `yr` argument takes a vector to plot next to the rows, while the `yt` argument takes a vector to plot next to the columns.

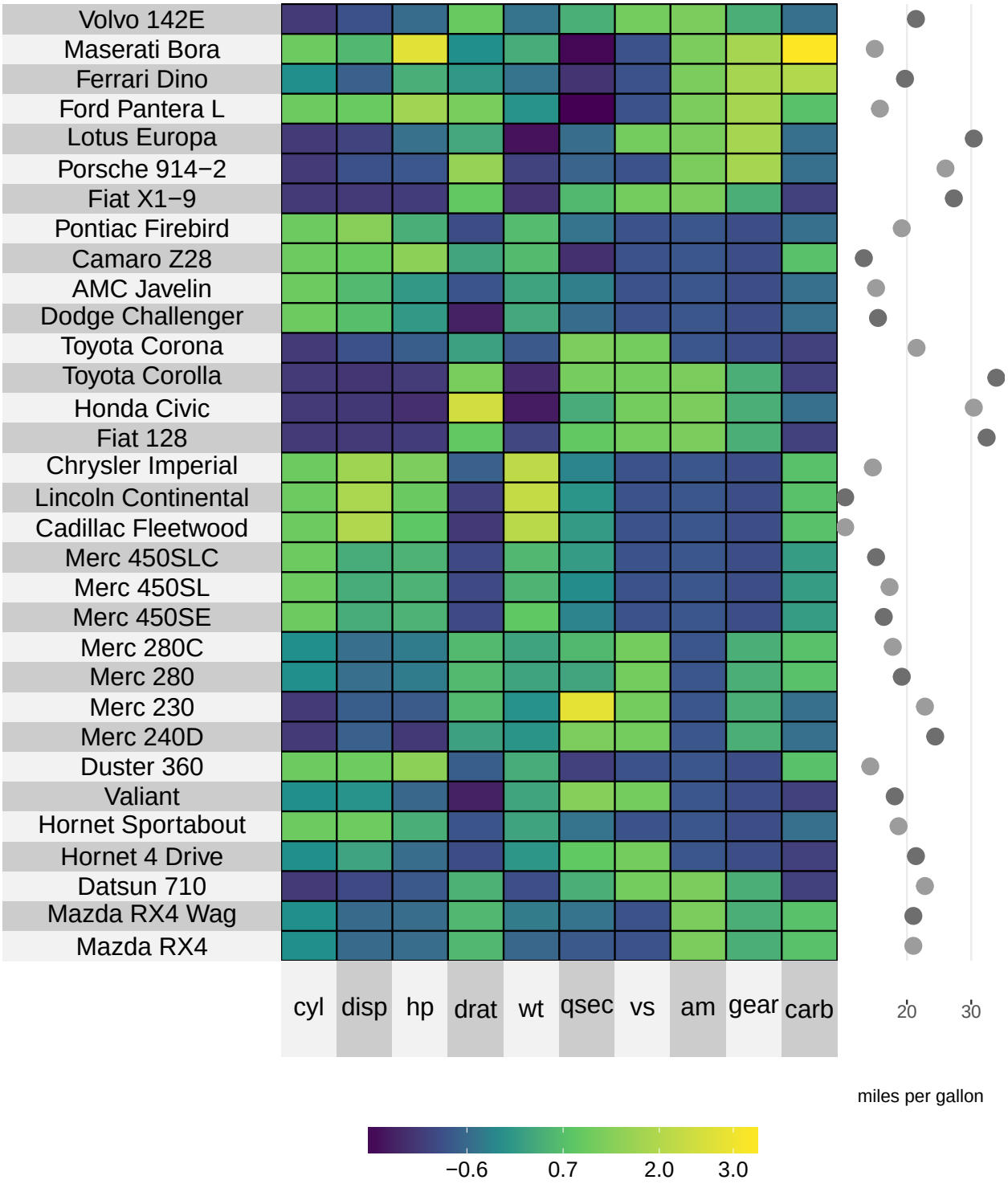
```
# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg),
          # scale the variables/columns
          scale = T,

          # add mpg as a scatterplot next to the rows
          yr = mtcars$mpg,
          yr.axis.name = "miles per gallon",

          left.label.size = 0.5,
          bottom.label.size = 0.1)
```



8.1.1 Size

You can change the size of the scatterplot points using the `yr.point.size` or `yt.point.size` arguments.

```r
# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg),
          # scale the variables/columns
          scale = T,

          # add mpg as a scatterplot next to the rows
          yr = mtcars$mpg,
          yr.axis.name = "miles per gallon",
          yr.point.size = 4,

          left.label.size = 0.5,
          bottom.label.size = 0.1)
```



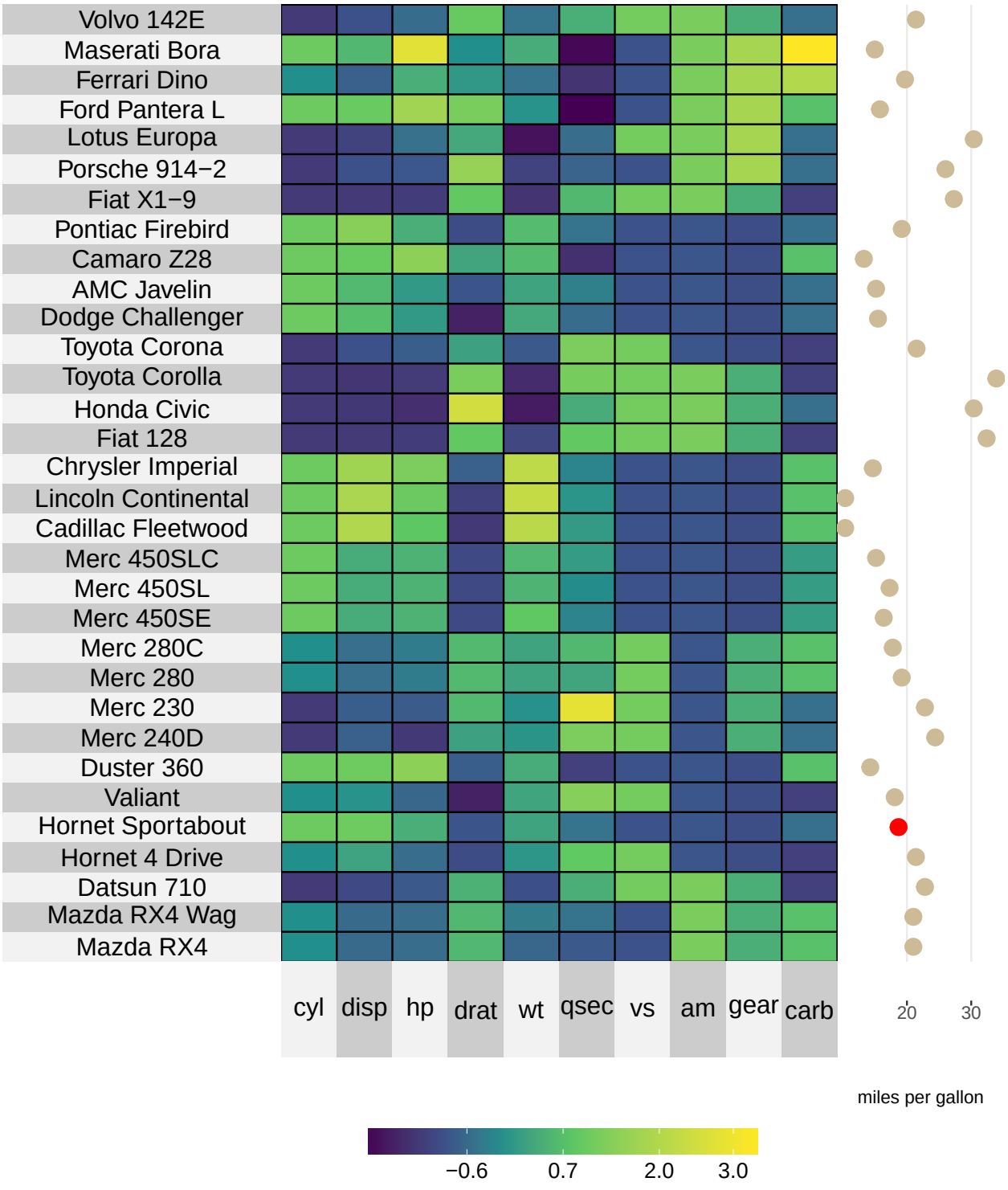
8.1.2 Color

Changing the color of the points in the scatterplot can be achieved using the `yr.obs.col` and `yt.obs.col` arguments, which are designed for specifying the color of individual data points.

For example, in the plot below, we are setting the fifth data point to be red, while the rest are grey. Note that

the "fifth" data point corresponds to the fifth data point in the original matrix X, rather than the re-ordered matrix (recall that the default order corresponds to a hierarchical clustering. To remove this ordering, specify `pretty.order.rows = FALSE`).

```
# set a color vector
point.col <- rep("wheat3", nrow(mtcars))
point.col[5] <- "red"

# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg),
          # scale the variables/columns
          scale = T,

          # add mpg as a scatterplot next to the rows
          yr = mtcars$mpg,
          yr.axis.name = "miles per gallon",
          # change the color of the points
          yr.obs.col = point.col,
          yr.point.size = 4,

          left.label.size = 0.5,
          bottom.label.size = 0.1)
```

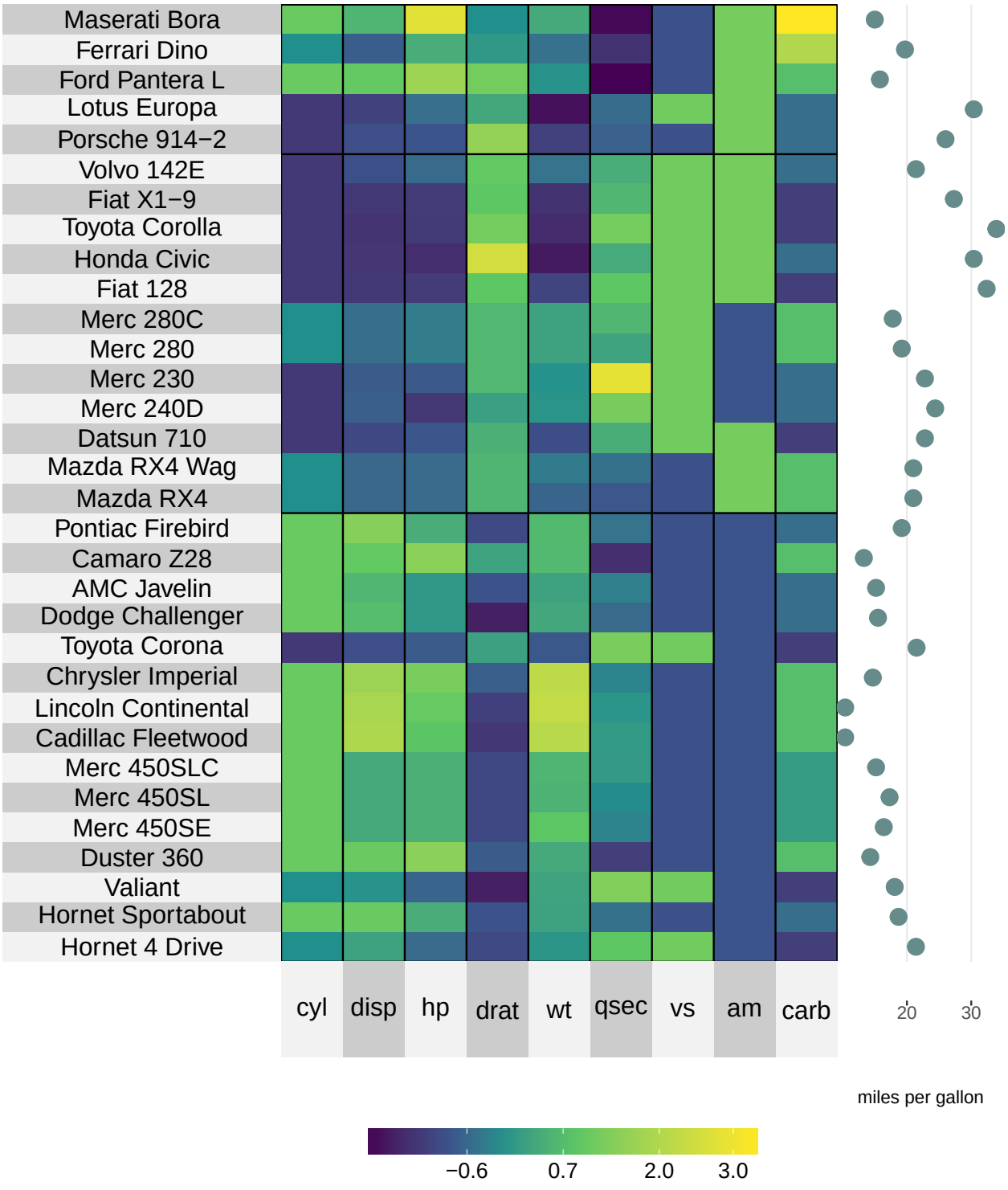


8.1.3 Clustering

If we cluster the cars into three groups based on the number of gears, then we can provide a `yr` whose length is either equal to `nrow(X)` or is equal to the number of clusters (`length(membership.rows)`), which in this case is equal to 3.

If `yr` has length equal to `nrow(X)` then we can specify the point colors using `yr.obs.col` as above.

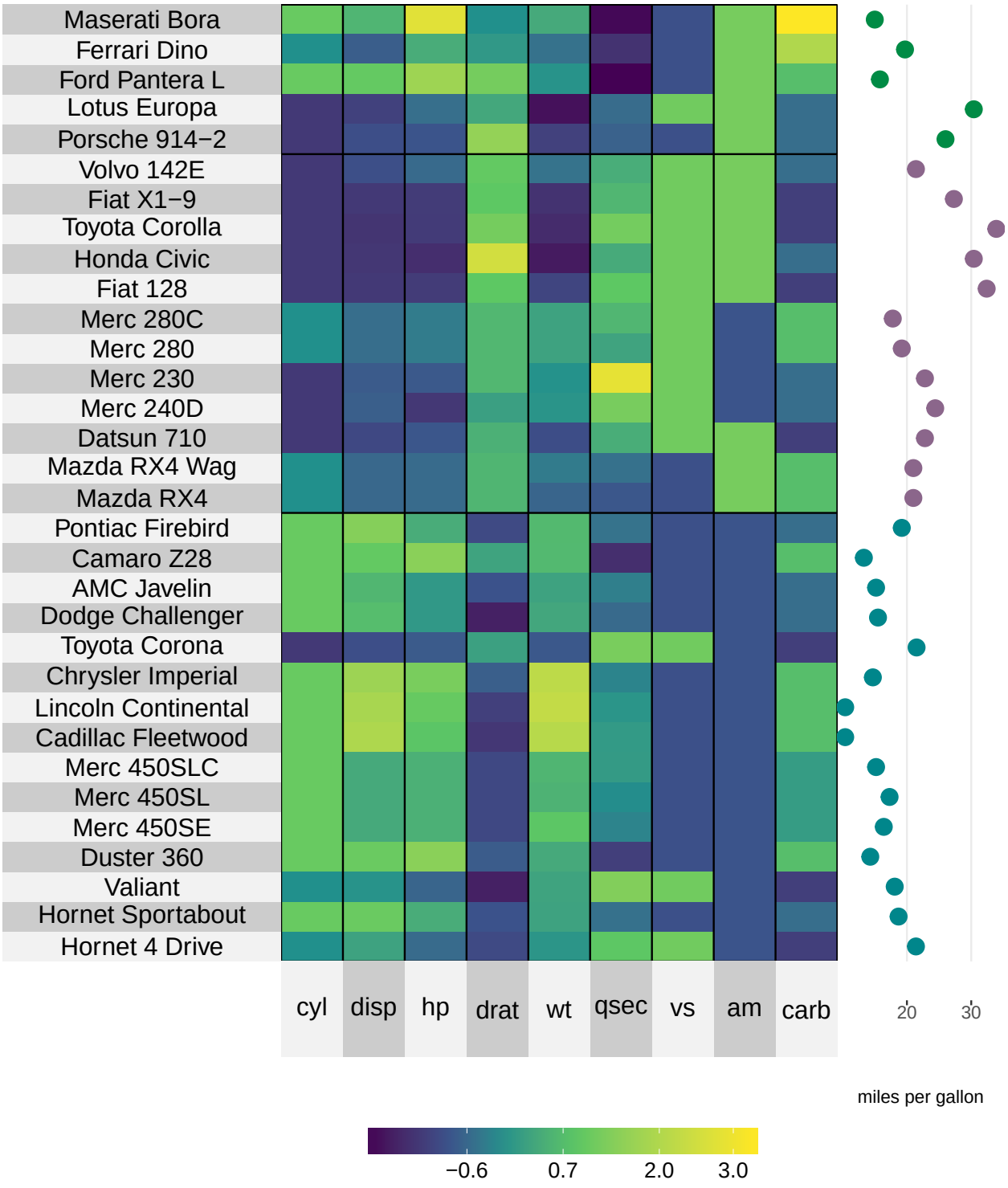
```
# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg, -gear),
          # scale the variables/columns
          scale = T,

          # cluster the rows
          membership.rows = paste(mtcars$gear, "gears"),
          left.label = "variable",

          # add mpg as a scatterplot next to the rows
          yr = mtcars$mpg,
          yr.axis.name = "miles per gallon",
          yr.obs.col = rep("paleturquoise4", nrow(mtcars)),
          yr.point.size = 4,

          left.label.size = 0.5,
          bottom.label.size = 0.1)
```



Setting the color for each cluster can be achieved using the `yr.cluster.col`/`yt.cluster.col` arguments.

```
# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg, -gear),
          # scale the variables/columns
          scale = T,
```

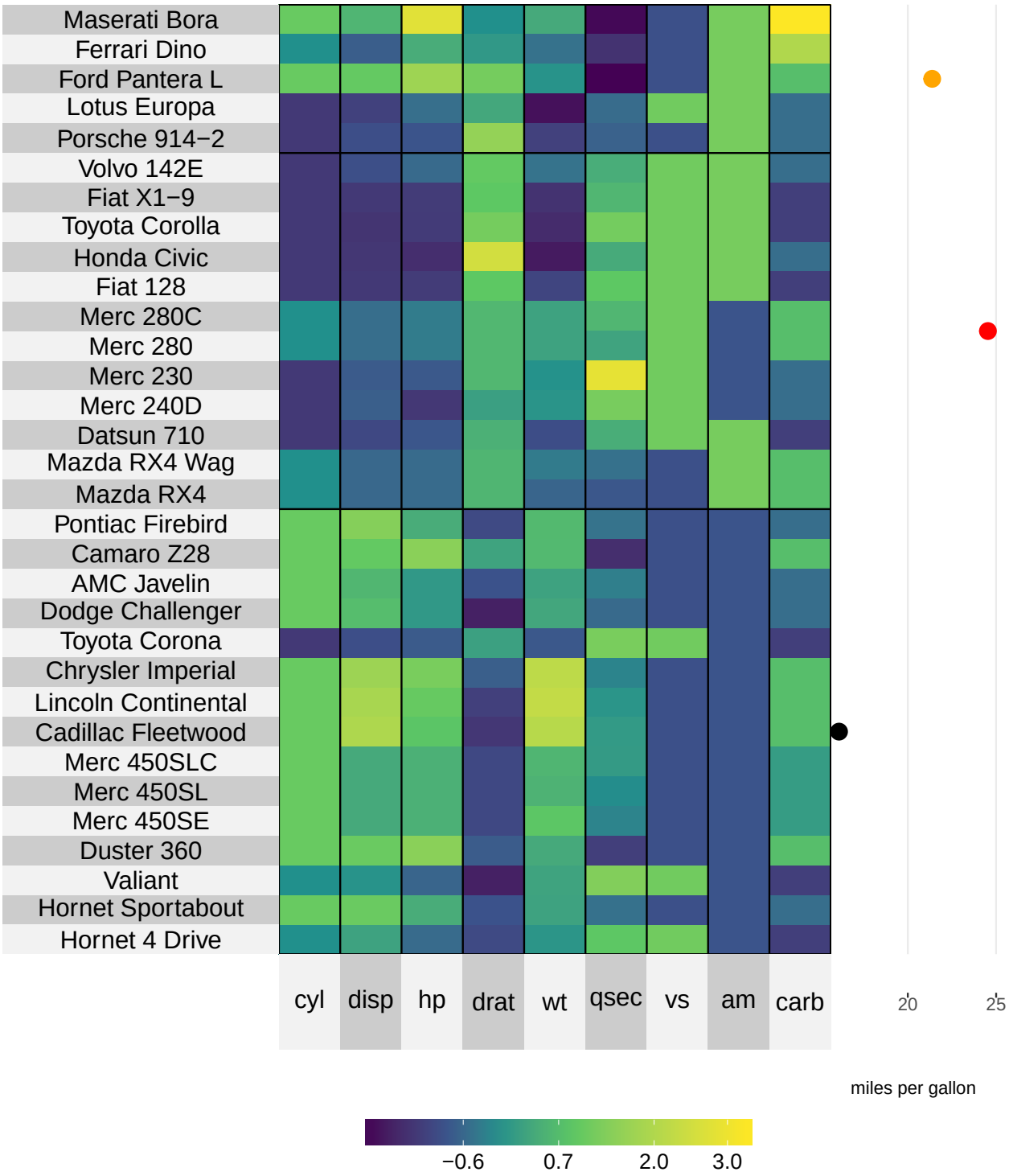
```
          # cluster the rows
          membership.rows = paste(mtcars$gear, "gears"),
          left.label = "variable",

          # add mpg as a scatterplot next to the rows
          yr = mtcars$mpg,
          yr.axis.name = "miles per gallon",
          yr.cluster.col = c("turquoise4", "plum4", "springgreen4"),
          yr.point.size = 4,

          left.label.size = 0.5,
          bottom.label.size = 0.1)
```



If `yr` has length equal to the number of clusters (which in this case would correspond to a vector of length three), then the three points are placed next to each cluster and `yr.cluster.col`/`yt.cluster.col` should be used to define the color for the points at the cluster-level.

```
# average the miles per gallon in each gear cluster
library(dplyr)
mpg.per.cluster <- mtcars %>%
  group_by(gear) %>%
```

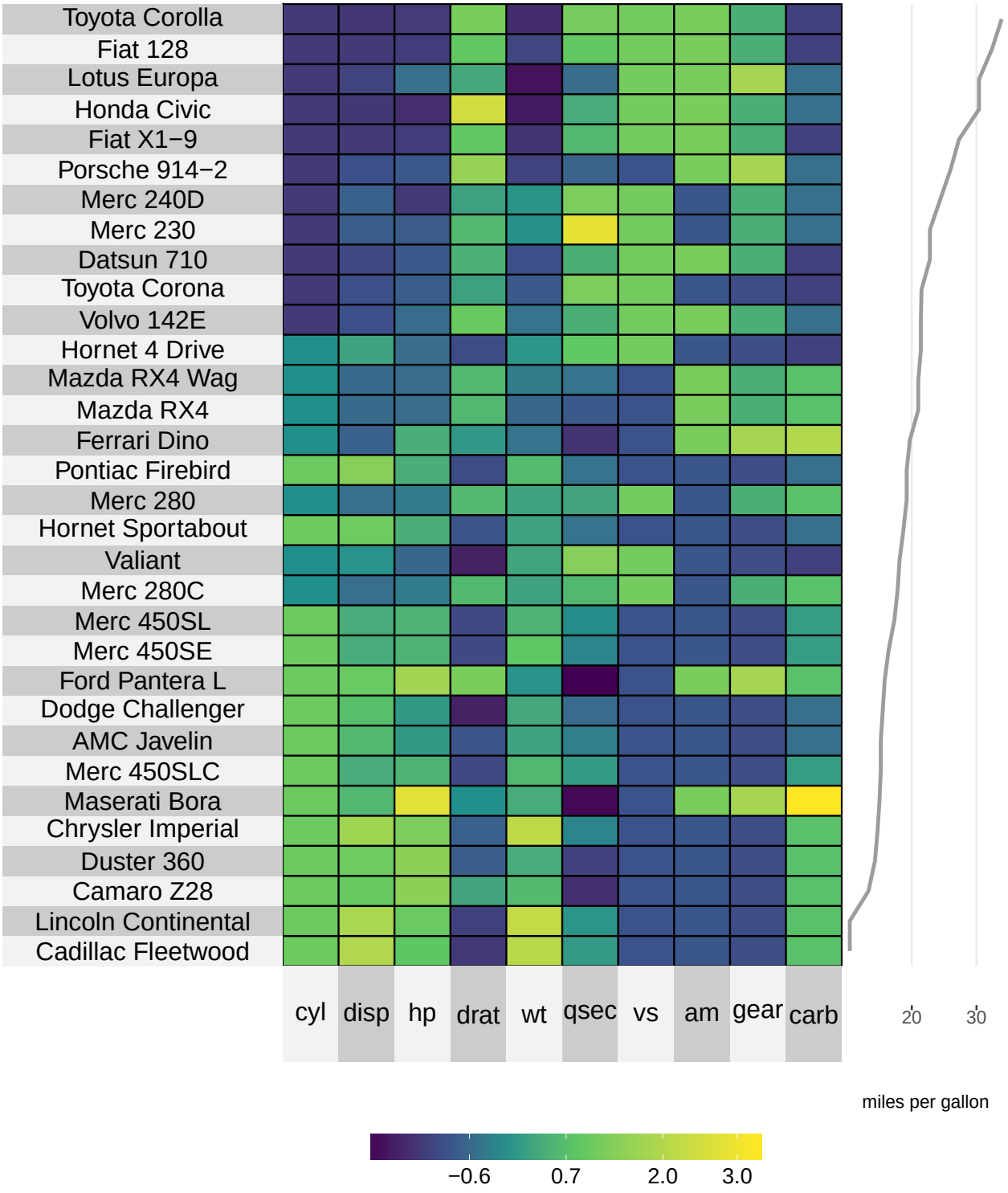
```
  summarize(mpg.avg = mean(mpg)) %>%
  select(mpg.avg) %>%
  unlist

# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg, -gear),
          # scale the variables/columns
          scale = T,

          # cluster the rows
          membership.rows = paste(mtcars$gear, "gears"),
          left.label = "variable",

          # add mpg as a scatterplot next to the rows
          yr = mpg.per.cluster,
          yr.axis.name = "miles per gallon",
          yr.cluster.col = c("black", "red", "orange"),
          yr.point.size = 4,

          left.label.size = 0.5,
          bottom.label.size = 0.1)
```



8.2 Line plot

The line plot is a nice way of depicting a trend. Instead of plotting each data unit as a point connects the points via a continuous line. In the example below, we are plotting the miles per gallon as a line plot, and simultaneously ordering the rows by miles per gallon.

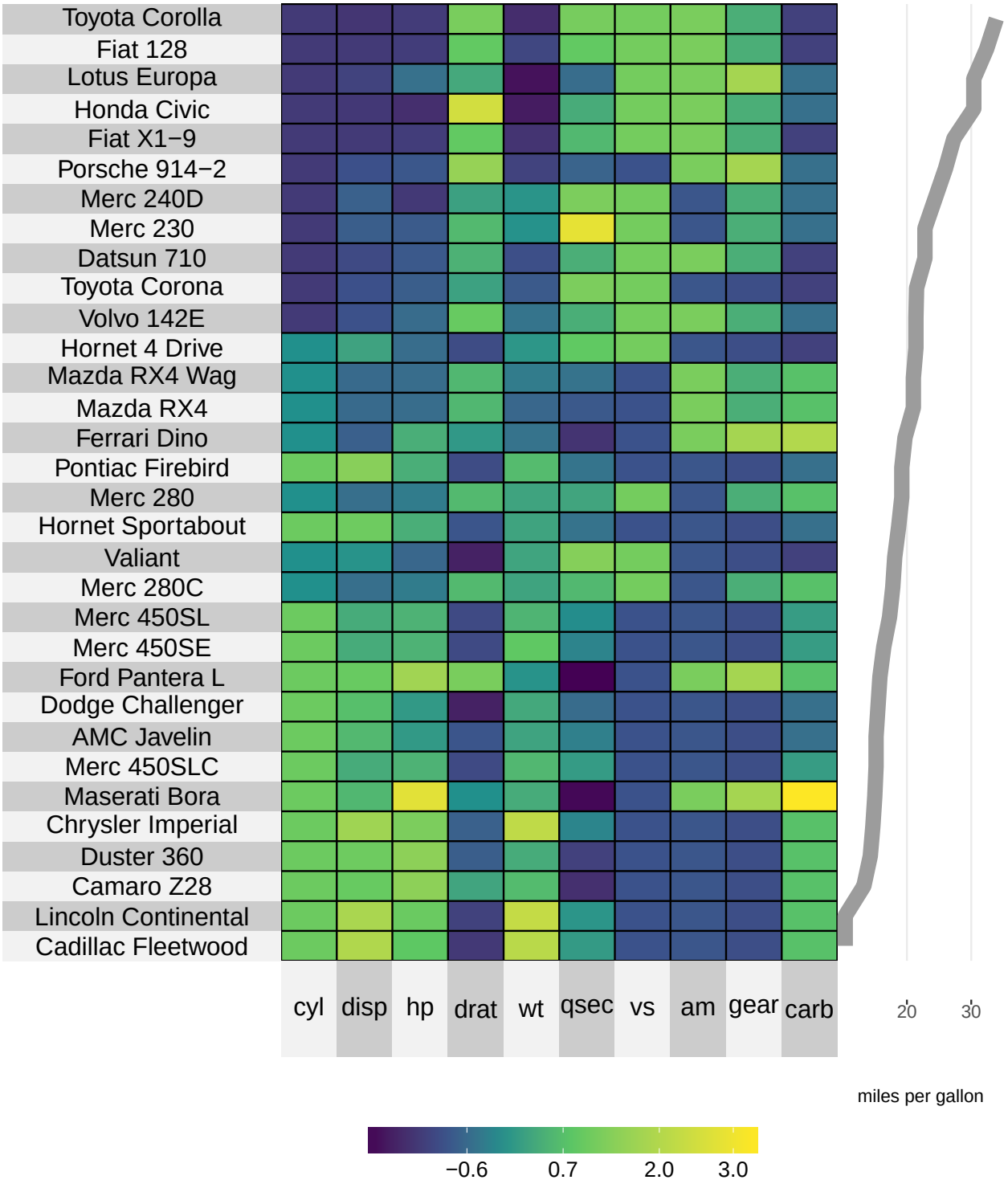
```
# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg),
          # scale the variables/columns
          scale = T,

          # add mpg as a line plot next to the rows
          yr = mtcars$mpg,
          yr.axis.name = "miles per gallon",
          yr.plot.type = "line",
          # order the rows by mpg
          order.rows = order(mtcars$mpg),

          left.label.size = 0.5,
          bottom.label.size = 0.1)
```



8.2.1 Size

The `yr.line.size`/`yt.line.size` arguments determine the thickness of the line plot.

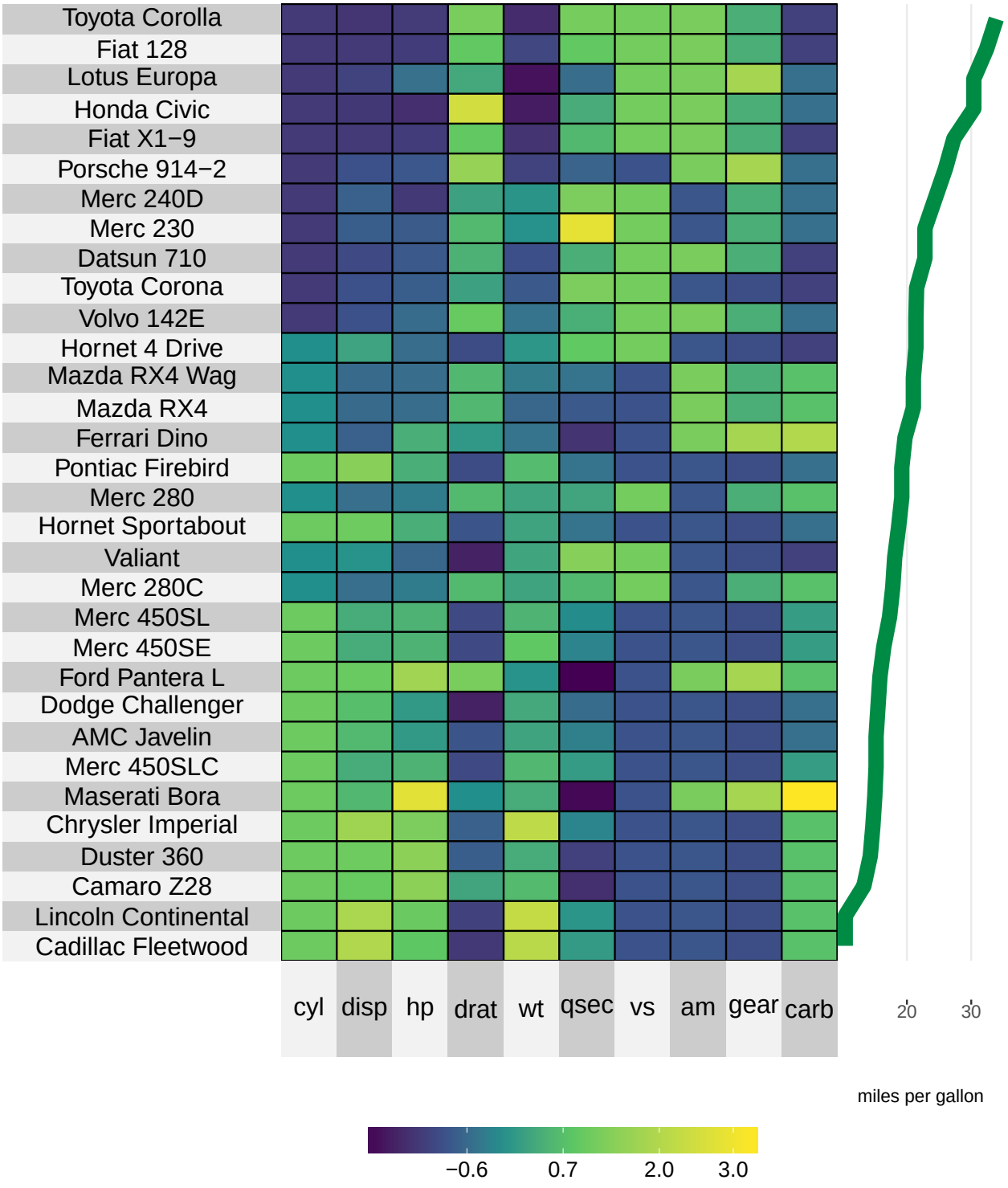
```r
# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg),
          # scale the variables/columns
          scale = T,

          # add mpg as a line plot next to the rows
          yr = mtcars$mpg,
          yr.axis.name = "miles per gallon",
          yr.plot.type = "line",
          # change the line thickness
          yr.line.size = 4,
          # order the rows by mpg
          order.rows = order(mtcars$mpg),

          left.label.size = 0.5,
          bottom.label.size = 0.1)
```



8.2.2 Color

The color can be changed using the `yr.line.col`/`yt.line.col` arguments.

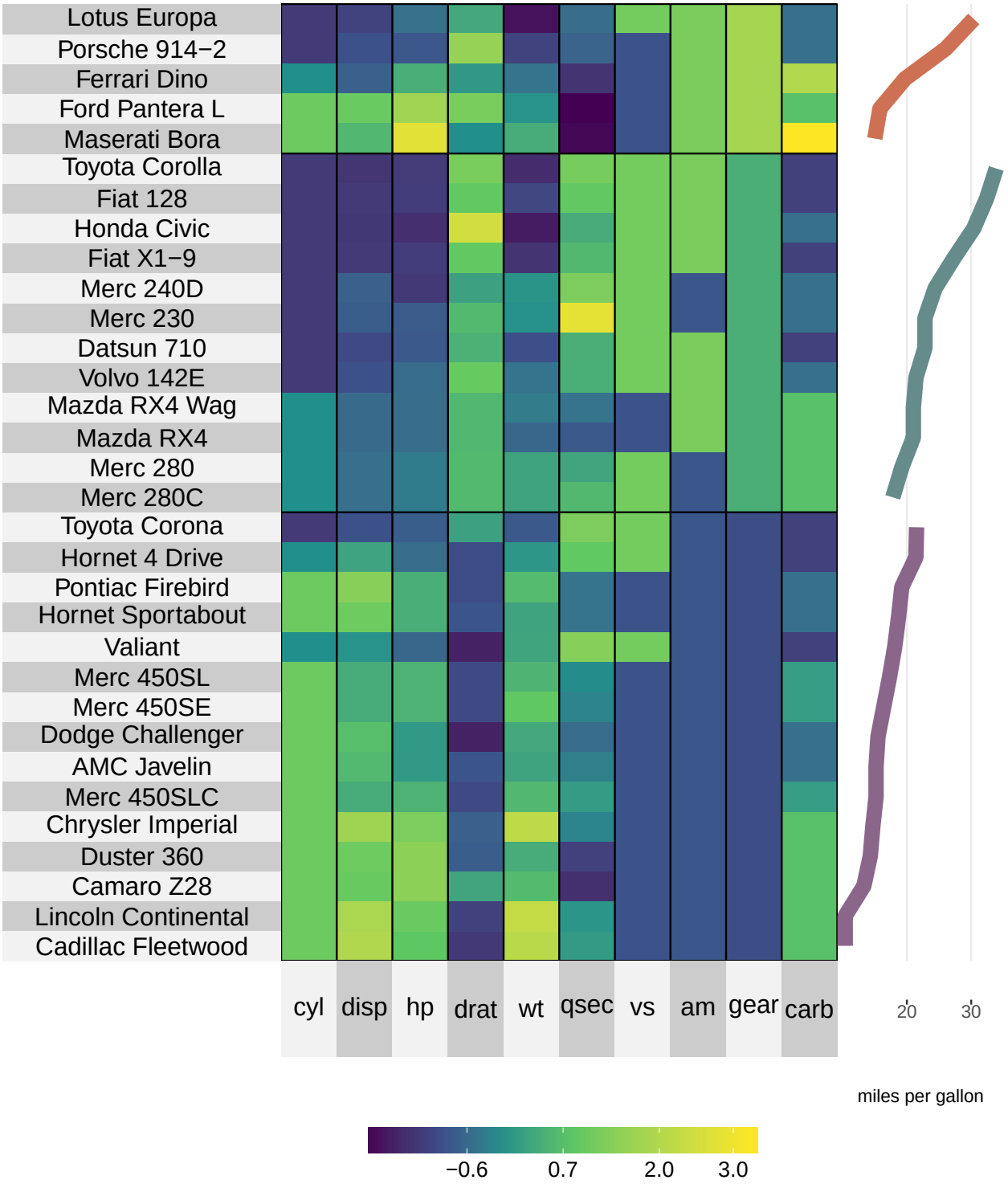
```
# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg),
          # scale the variables/columns
          scale = T,

          # add mpg as a line plot next to the rows
          yr = mtcars$mpg,
          yr.axis.name = "miles per gallon",
          yr.plot.type = "line",
          # change the line thickness
          yr.line.size = 4,
          # change the line color
          yr.line.col = "springgreen4",
          # order the rows by mpg
          order.rows = order(mtcars$mpg),

          left.label.size = 0.5,
          bottom.label.size = 0.1)
```



8.2.3 Clustering

When clustering, the line will be grouped by cluster and the cluster-wise color can be set using `yr.clust.col`/`yt.clust.col` (rather than `yr.line.col` etc). Note that you cannot have aggregated line plots at the cluster level, implying that `yr` and `yt` must have the same length as `nrow(X)` and `ncol(X)`.

```
# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg),
          # scale the variables/columns
          scale = T,

          # cluster the rows
          membership.rows = paste(mtcars$gear, "gears"),
          left.label = "variable",

          # add mpg as a line plot next to the rows
          yr = mtcars$mpg,
          yr.axis.name = "miles per gallon",
          yr.plot.type = "line",
          # change the line thickness
          yr.line.size = 4,
          # change the line color
          yr.cluster.col = c("plum4", "paleturquoise4", "salmon3"),
          # order the rows by mpg
          order.rows = order(mtcars$mpg),

          left.label.size = 0.5,
          bottom.label.size = 0.1)
```



8.3 Smoothed line

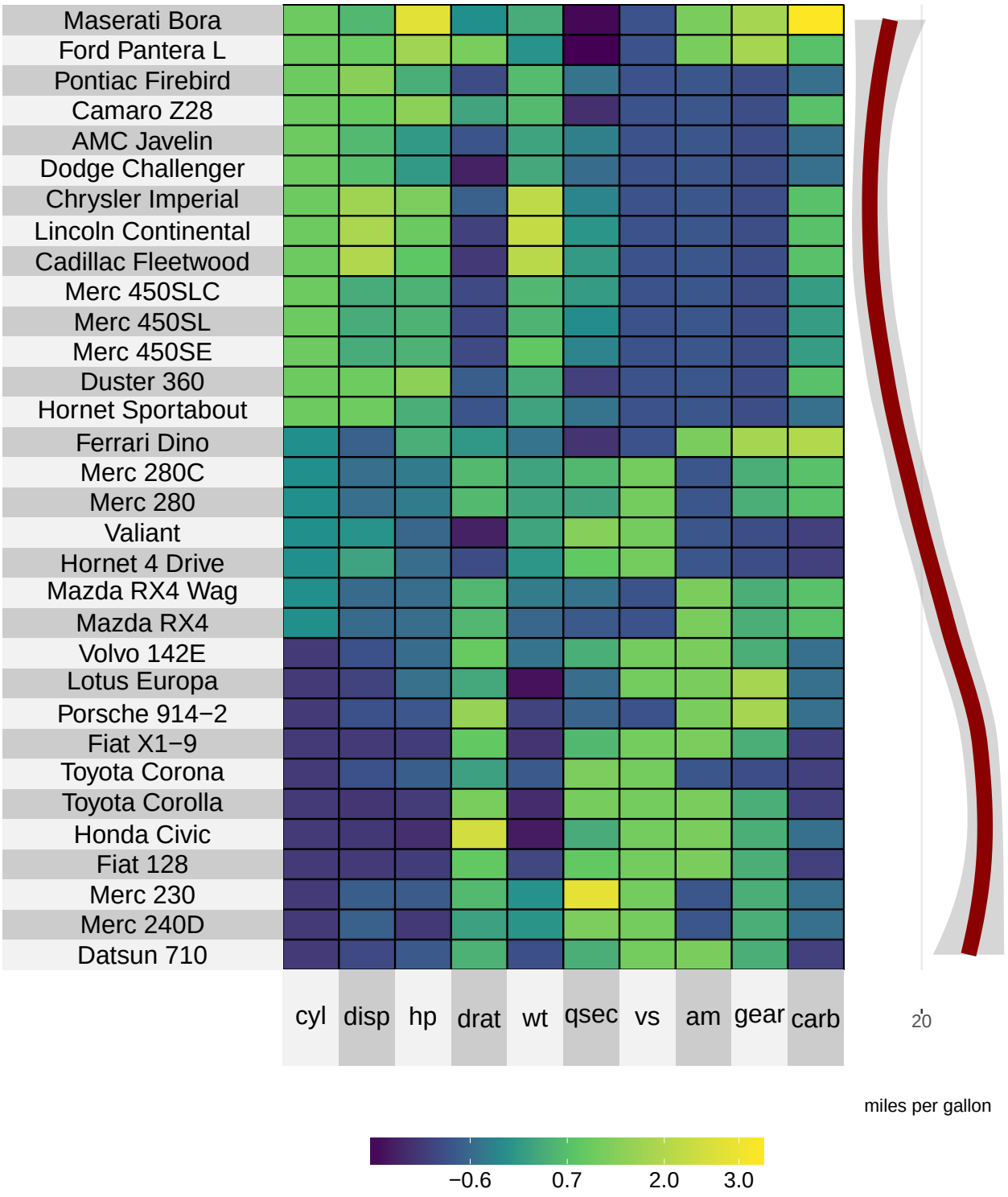
The options for a smoothed line are much line those for the line plot above. Setting `yt.plot.type`/`yr.plot.type` to `"smooth"` will provide a loess smoothed line (default) or linear regression line based (set `smoothing.method`

= "lm"). Color can be specified using yr.line.col/yt.line.col.

8.3.1 Loess curve

In the example below we produce a loess smoothed curve for miles per gallon versus number of cylinders (order.rows = order(mtcars$cyl)). The standard error shading can be removed by specifying smooth.se = FALSE.

```
# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg),
          # scale the variables/columns
          scale = T,

          # add mpg as a smoothed line plot next to the rows
          yr = mtcars$mpg,
          yr.axis.name = "miles per gallon",
          yr.plot.type = "smooth",
          # change the line thickness and color
          yr.line.size = 4,
          yr.line.col = "red4",
          # order the rows by mpg
          order.rows = order(mtcars$cyl),


          left.label.size = 0.5,
          bottom.label.size = 0.1)
```

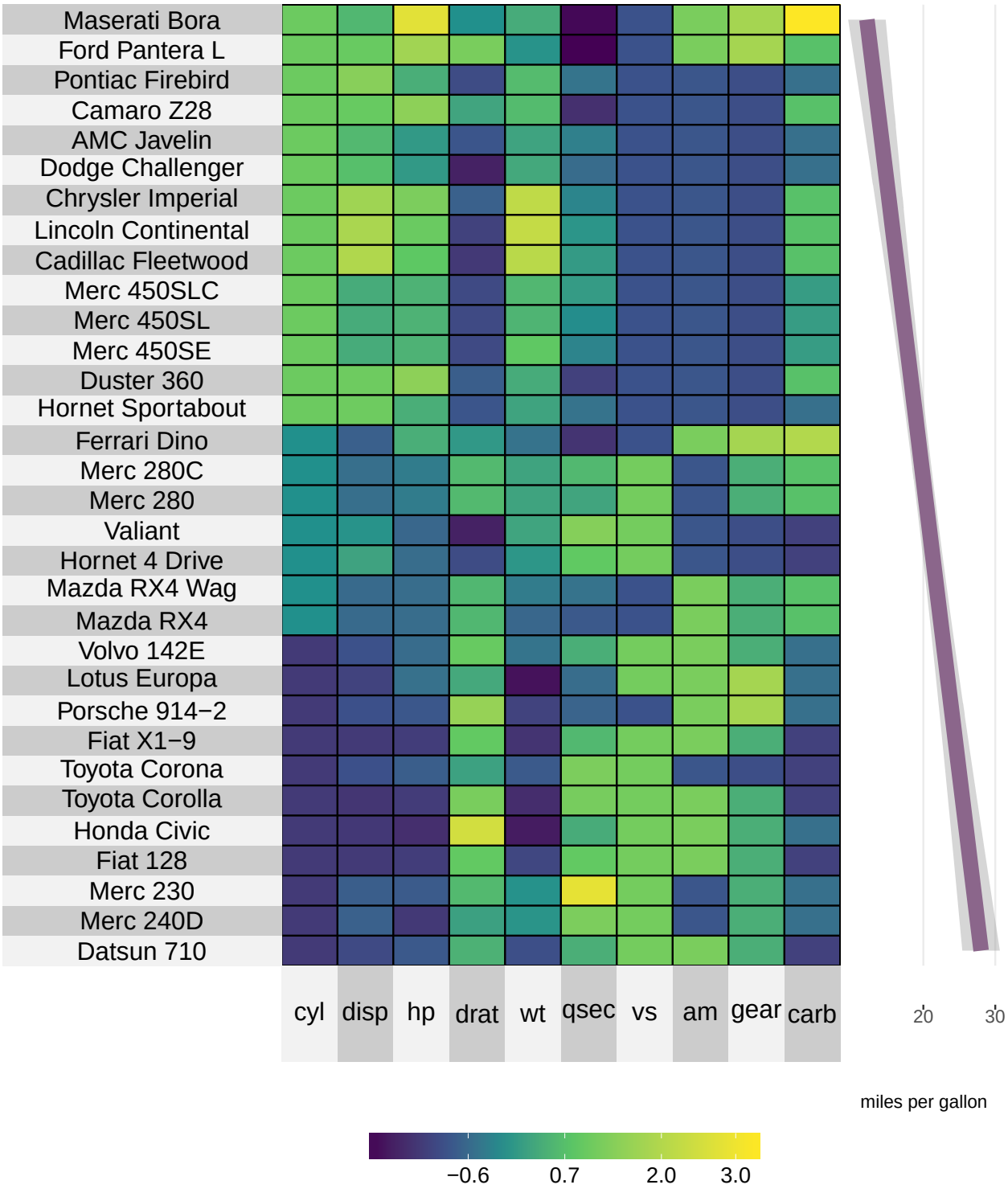


8.3.2 Linear regression line

A linear regression of miles per gallon versus number of cylinders (`order.rows = order(mtcars$cyl)`) is specified similarly with the additional argument `smoothing.method = "lm"`.

Again, the standard error shading can be removed by specifying `smooth.se = FALSE`.

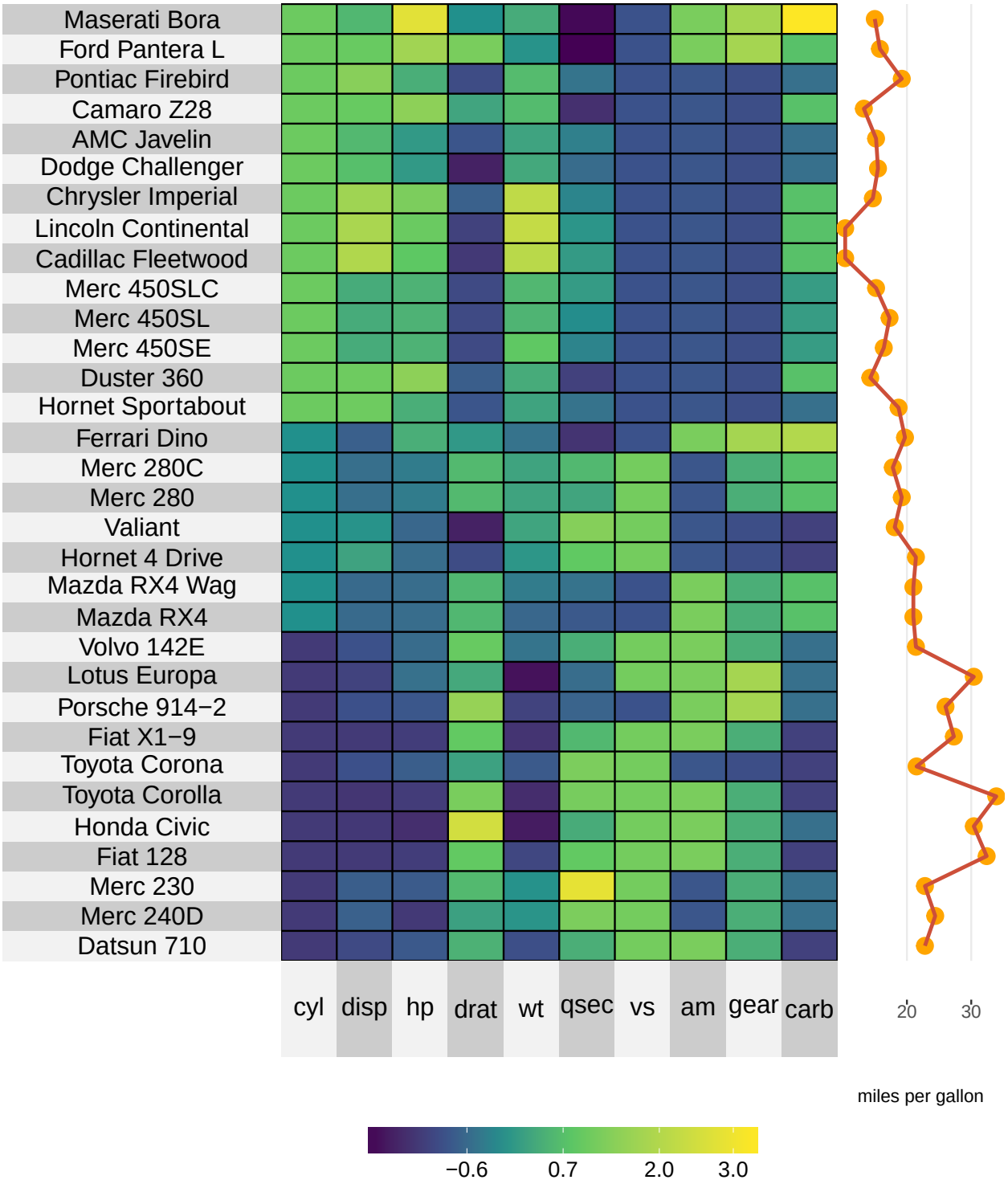
```r
# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg),
          # scale the variables/columns
          scale = T,

          # add mpg as a smoothed line plot  next to the rows
          yr = mtcars$mpg,
          yr.axis.name = "miles per gallon",
          yr.plot.type = "smooth",
          smoothing.method = "lm",
          # change the line thickness and color
          yr.line.size = 4,
          yr.line.col = "plum4",
          # order the rows by mpg
          order.rows = order(mtcars$cyl),

          left.label.size = 0.5,
          bottom.label.size = 0.1)
```



8.4 Scatterplot with connecting line plot

The scatterline plot combines the line plot and the scatter plot. The arguments that can be used separately for the line plot and the scatterplot can be used for the scatterline plot.

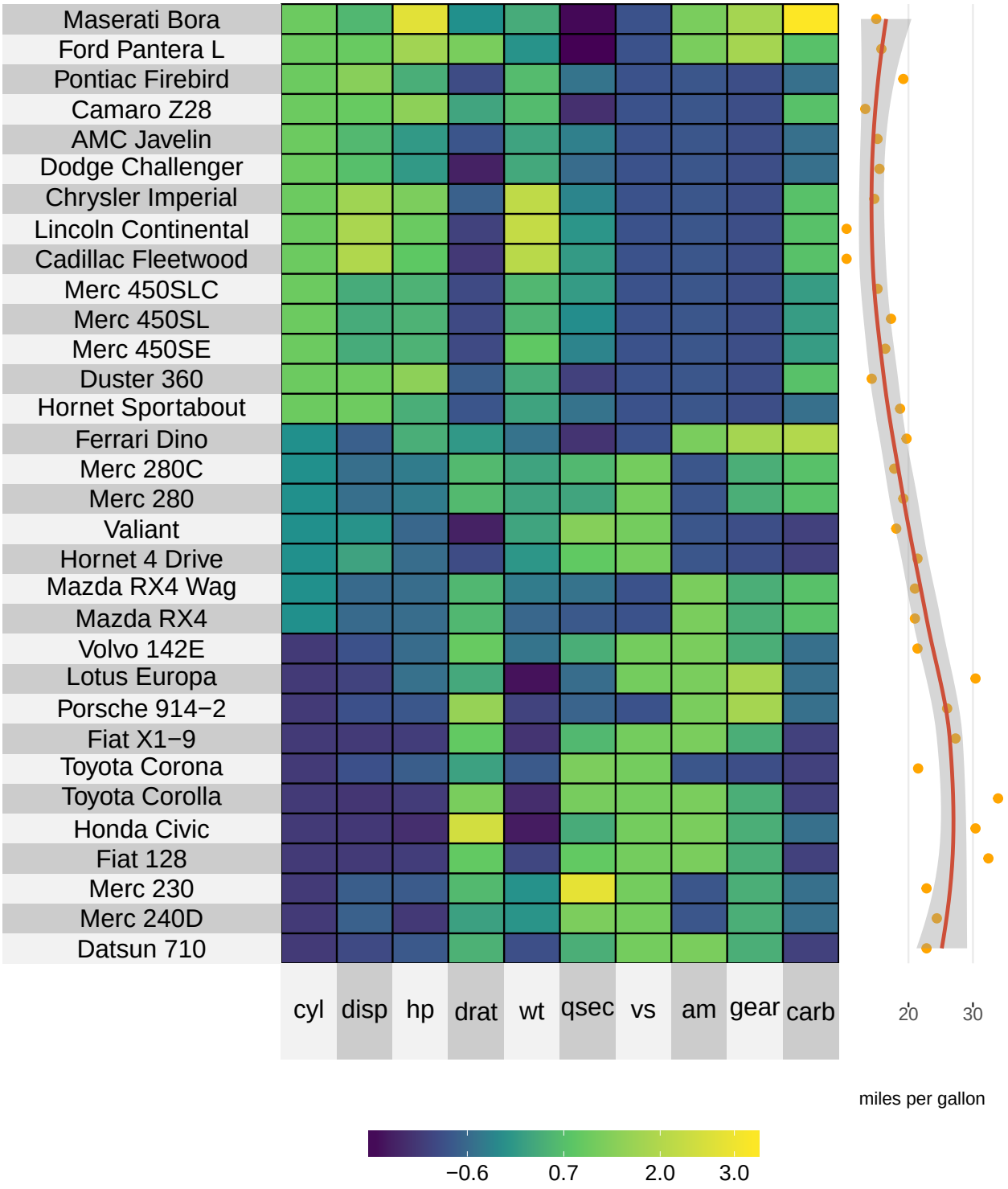
```r
# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg),
          # scale the variables/columns
          scale = T,

          # add mpg as a scatter line plot next to the rows
          yr = mtcars$mpg,
          yr.axis.name = "miles per gallon",
          yr.plot.type = "scatterline",
          # change the line color
          yr.line.col = "tomato3",
          yr.obs.col = rep("orange", nrow(mtcars)),
          yr.point.size = 4,
          # order the rows by mpg
          order.rows = order(mtcars$cyl),

          left.label.size = 0.5,
          bottom.label.size = 0.1)
```



8.5 Scatterplot with smoothed line

The scattersmooth plot combines the functionality of the scatter plot with the smoothed curve. The aesthetic arguments that apply for the scatterplot and the smoothed curve apply for the scattersmooth plot too.

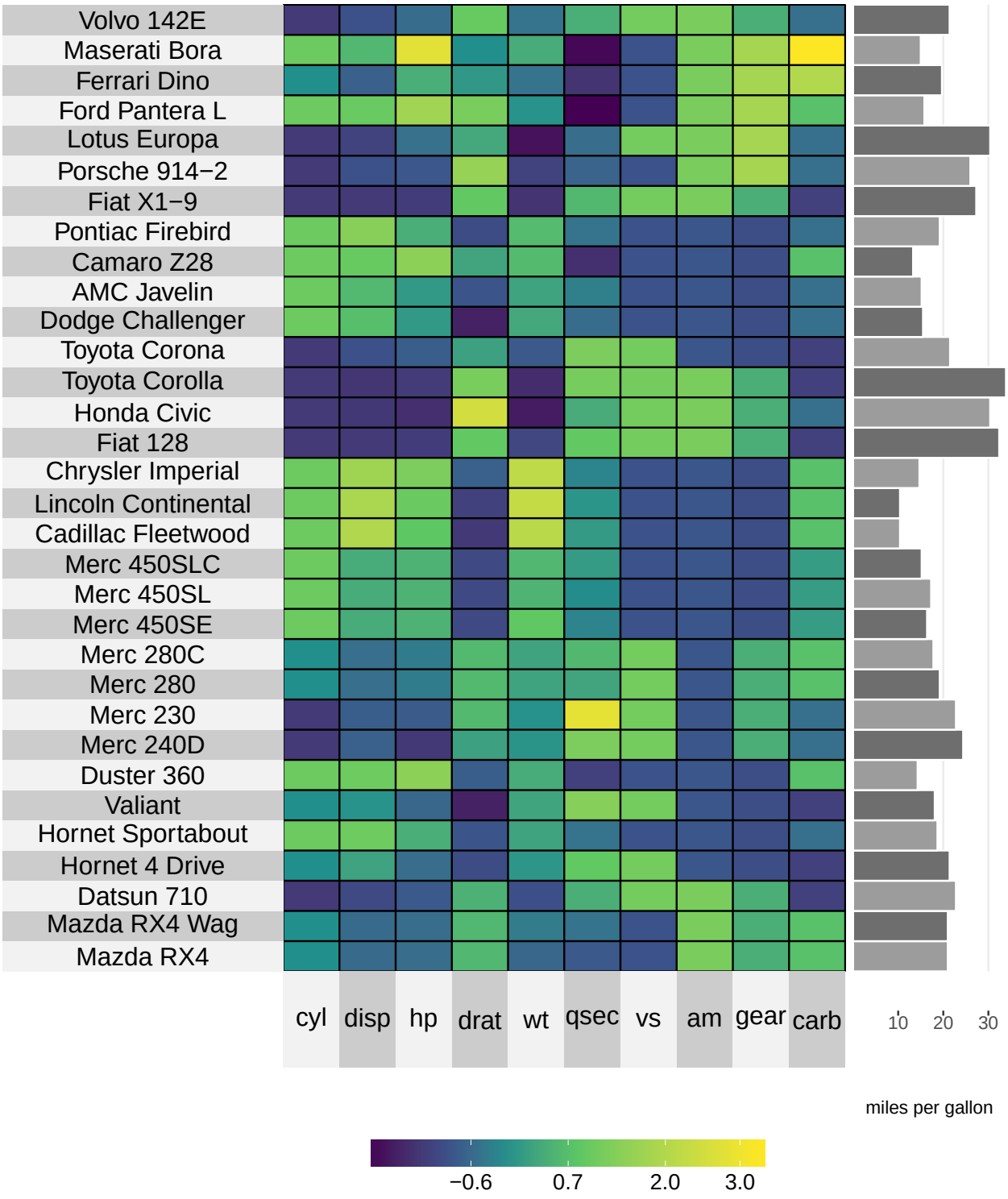
```
# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg),
          # scale the variables/columns
          scale = T,

          # add mpg as a scatter smoothed plot next to the rows
          yr = mtcars$mpg,
          yr.axis.name = "miles per gallon",
          yr.plot.type = "scattersmooth",
          # change the line color
          yr.line.col = "tomato3",
          yr.obs.col = rep("orange", nrow(mtcars)),
          # order the rows by mpg
          order.rows = order(mtcars$cyl),

          left.label.size = 0.5,
          bottom.label.size = 0.1)
```



8.6 Barplot

Barplots are a particularly nice way of presenting and comparing values of a variable. Adding a barplot next to the columns and/or rows can be achieved by setting `yr.plot.type = "bar"` or `yt.plot.type = "bar"`.

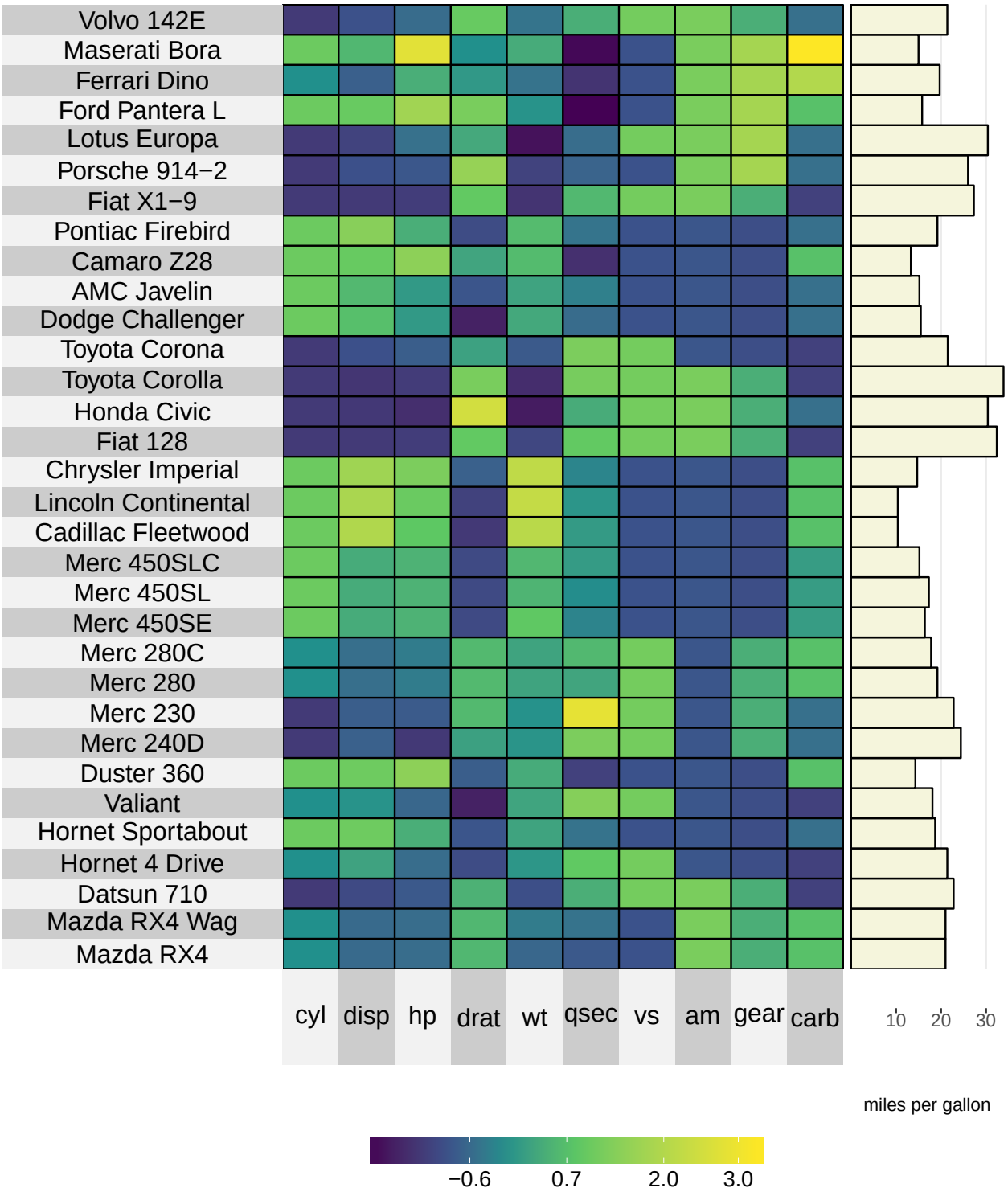
```r
# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg),
          # scale the variables/columns
          scale = T,

          # add mpg as a barplot next to the rows
          yr = mtcars$mpg,
          yr.axis.name = "miles per gallon",
          yr.plot.type = "bar",

          # change the label size
          left.label.size = 0.5,
          bottom.label.size = 0.1)
```



8.6.1 Color

The bar fill color can be set using the standard `yr.obs.col`/`yt.obs.col` arguments. The outline of each bar can be set using the `yr.bar.col`/`yr.bar.col` arguments.

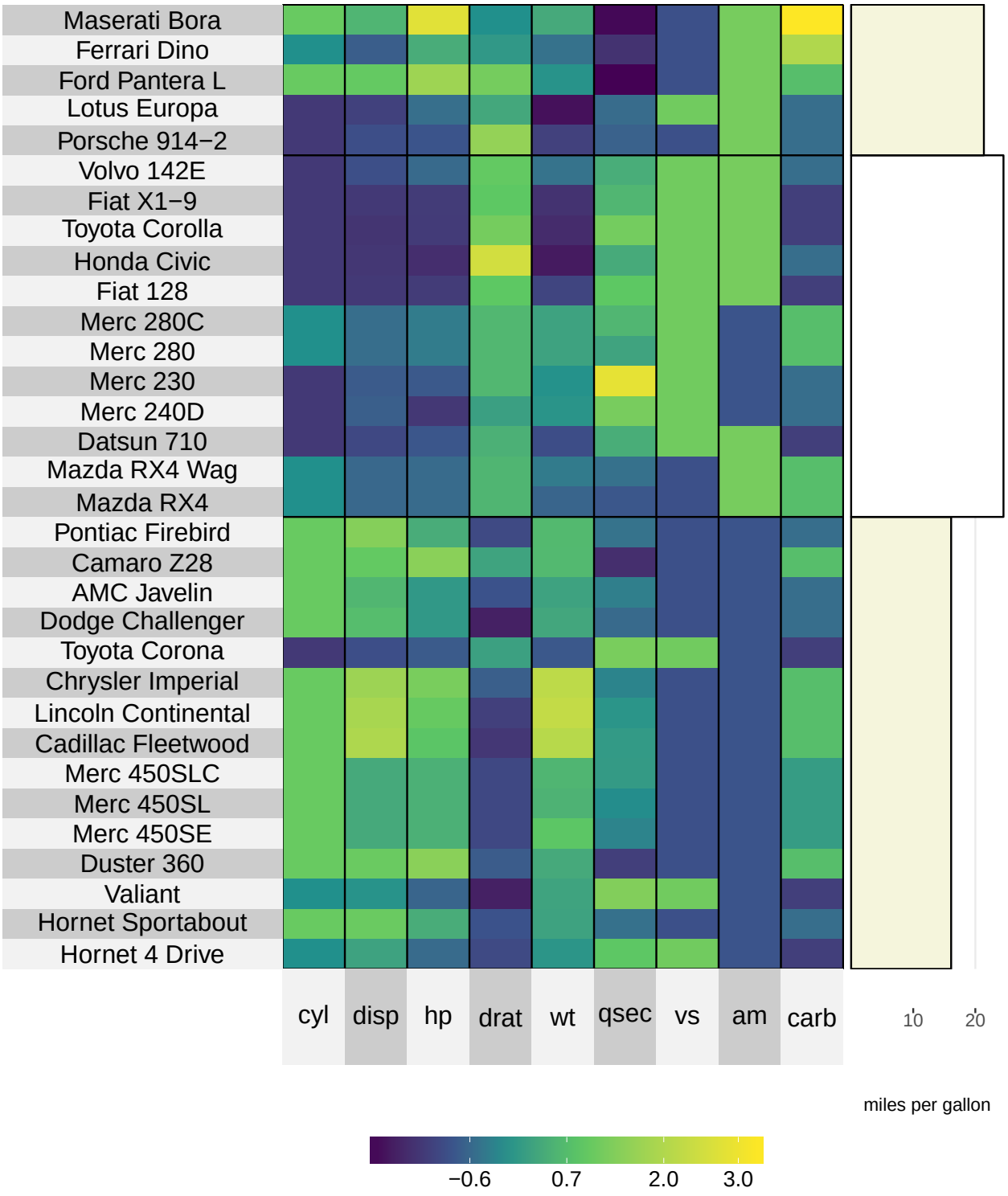
```r
# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg),
          # scale the variables/columns
          scale = T,

          # add mpg as a barplot next to the rows
          yr = mtcars$mpg,
          yr.axis.name = "miles per gallon",
          yr.plot.type = "bar",
          # set bar colors
          yr.bar.col = "black",
          yr.obs.col = rep("beige", nrow(mtcars)),

          # change the label size
          left.label.size = 0.5,
          bottom.label.size = 0.1)
```



8.6.2 Clustering

Bar plots can present values aggregated across clusters. In this situation, as with the other plots, the fill color of the bars is set using the `yr.cluster.col`/`yt.cluster.col` arguments (instead of the `yr.obs.col`/`yt.obs.col` arguments for the unclustered heatmap).

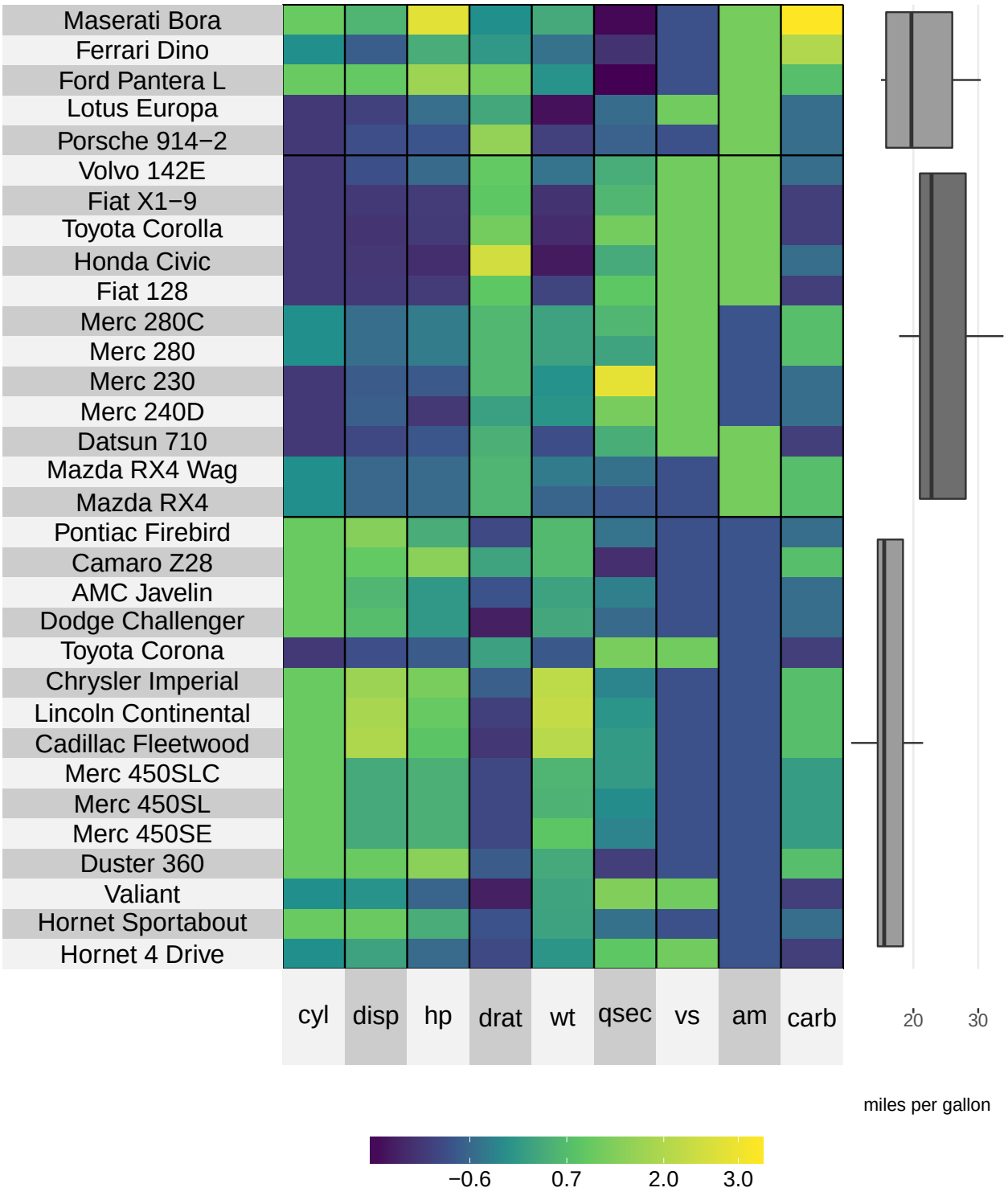
```r
library(dplyr)
mpg.per.cluster <- mtcars %>%
  group_by(gear) %>%
  summarize(mpg.avg = mean(mpg)) %>%
  select(mpg.avg) %>%
  unlist

# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg, -gear),
          # scale the variables/columns
          scale = T,

          # cluster the rows
          membership.rows = paste(mtcars$gear, "gears"),
          left.label = "variable",

          # add mpg per cluster as a barplot
          yr = mpg.per.cluster,
          yr.axis.name = "miles per gallon",
          yr.plot.type = "bar",
          # set bar colors
          yr.bar.col = "black",
          yr.cluster.col = c("beige", "white", "beige"),

          # change the label size
          left.label.size = 0.5,
          bottom.label.size = 0.1)
```



8.7 Boxplot

Boxplots are a bit different to the other plot types presented above. In particular, they can only be used on clustered matrices. The reason for this is that a boxplot must consist of many data points.

```r
library(dplyr)
mpg.per.cluster <- mtcars %>%
  group_by(gear) %>%
  summarize(mpg.avg = mean(mpg)) %>%
  select(mpg.avg) %>%
  unlist

# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg, -gear),
          # scale the variables/columns
          scale = T,

          # cluster the rows
          membership.rows = paste(mtcars$gear, "gears"),
          left.label = "variable",

          # add mpg per cluster as a boxplot
          yr = mtcars$mpg,
          yr.axis.name = "miles per gallon",
          yr.plot.type = "boxplot",

          # change the label size
          left.label.size = 0.5,
          bottom.label.size = 0.1)
```

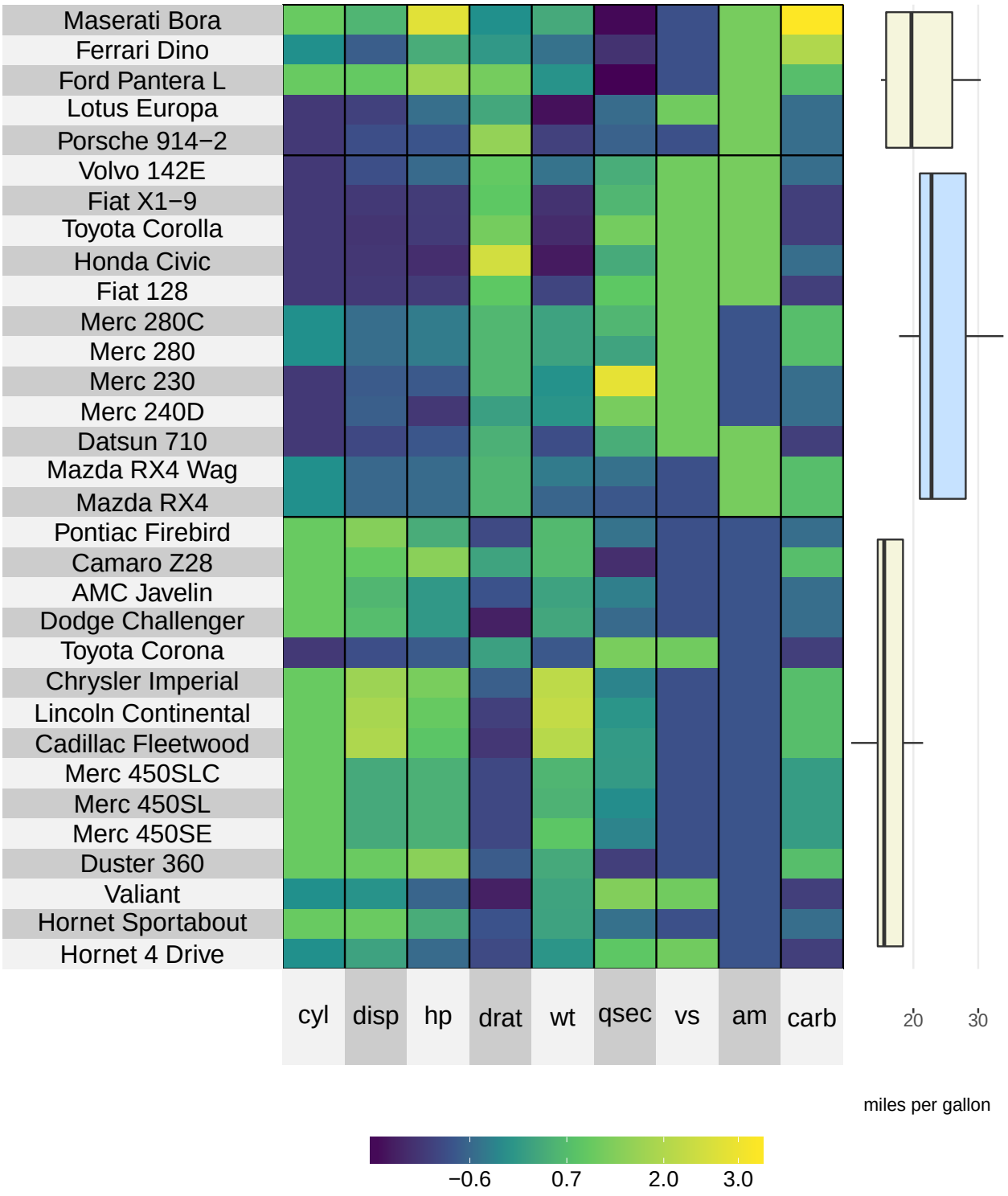


8.7.1 Color

Setting the color can be achieved using the `yr.cluster.col`/`yt.cluster.col` arguments.

```
library(dplyr)
mpg.per.cluster <- mtcars %>%
```

```
  group_by(gear) %>%
  summarize(mpg.avg = mean(mpg)) %>%
  select(mpg.avg) %>%
  unlist

# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg, -gear),
          # scale the variables/columns
          scale = T,

          # cluster the rows
          membership.rows = paste(mtcars$gear, "gears"),
          left.label = "variable",

          # add mpg per cluster as a boxplot
          yr = mtcars$mpg,
          yr.axis.name = "miles per gallon",
          yr.plot.type = "boxplot",
          yr.cluster.col = c("beige", "slategray1", "beige"),

          # change the label size
          left.label.size = 0.5,
          bottom.label.size = 0.1)
```

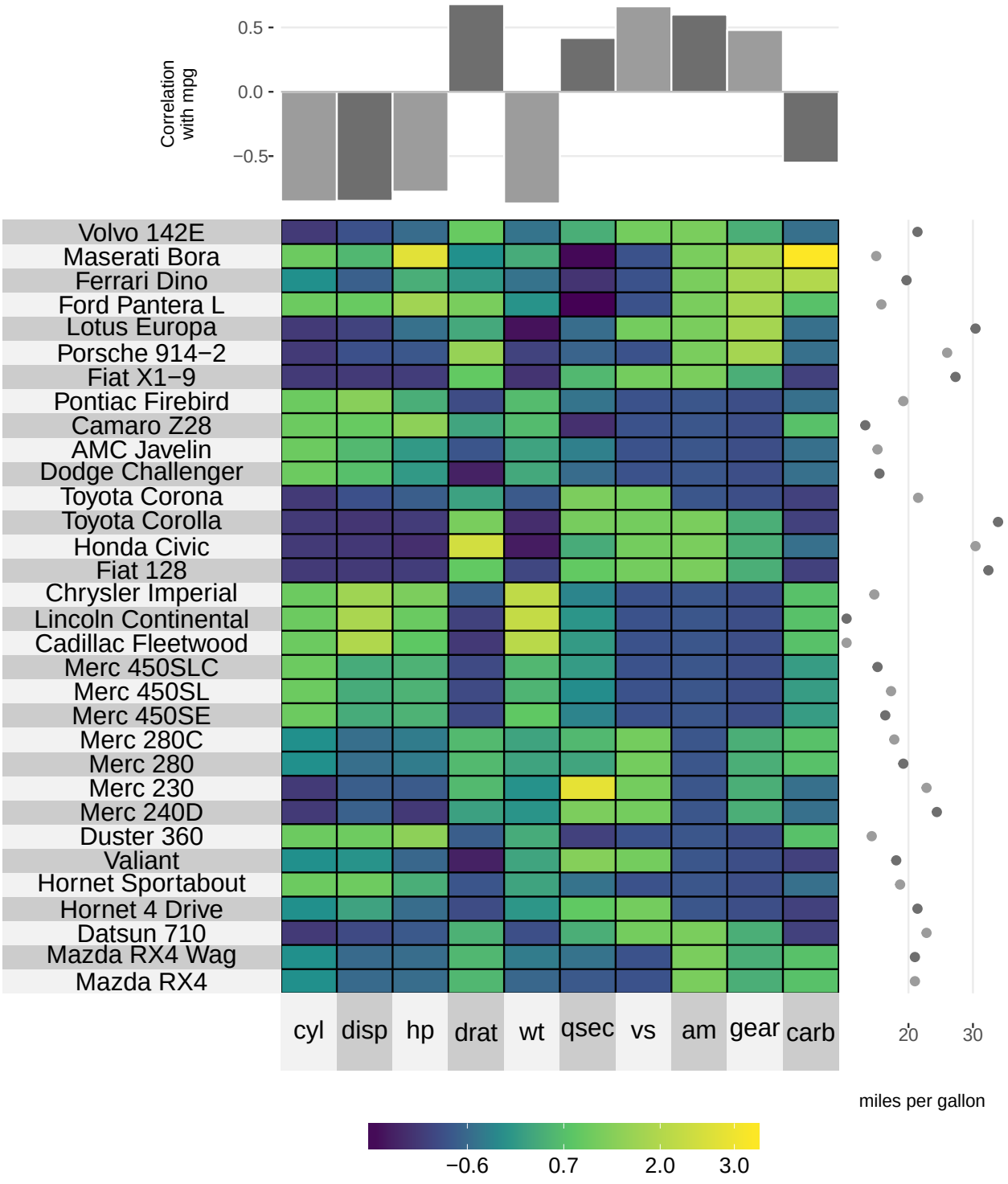


8.8 Axis options

8.8.1 Name

The axis name can be specified using `yr.axis.name`/`yt.axis.name`.

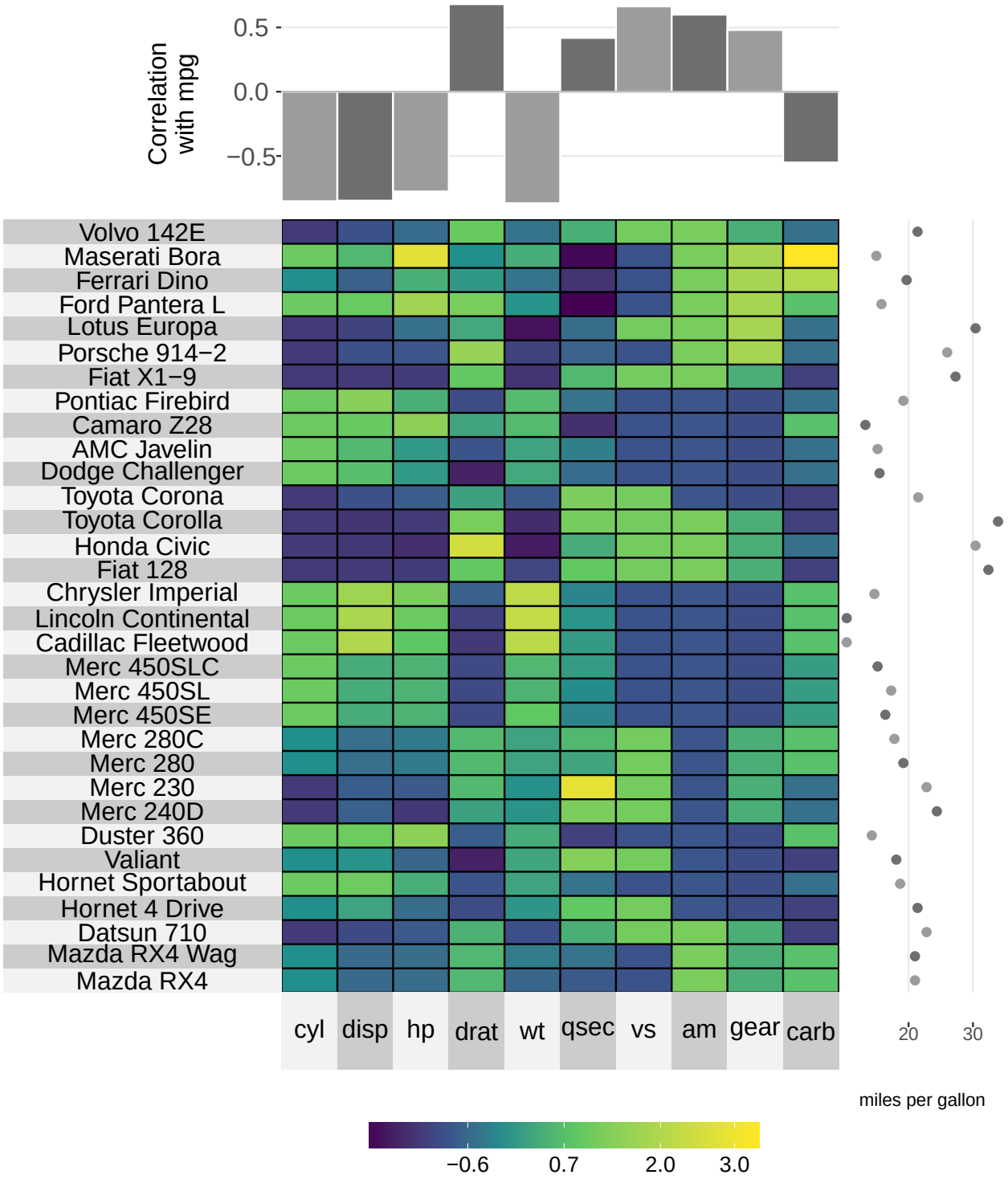
```
# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg),
          # scale the variables/columns
          scale = T,

          # add mpg as a scatterplot next to the rows
          yr = mtcars$mpg,
          yr.axis.name = "miles per gallon",

          # add correlation between each variable and miles per gallon
          yt = cor(mtcars)[-1,"mpg"],
          yt.plot.type = "bar",
          yt.axis.name = "Correlation\nwith mpg",

          left.label.size = 0.5,
          bottom.label.size = 0.1)
```



8.8.2 Size

The size of the axis name can be set using `yr.axis.name.size`/`yt.axis.name.size`, while the size of the axis numbers can be set using `yr.axis.size`/`yt.axis.size`.

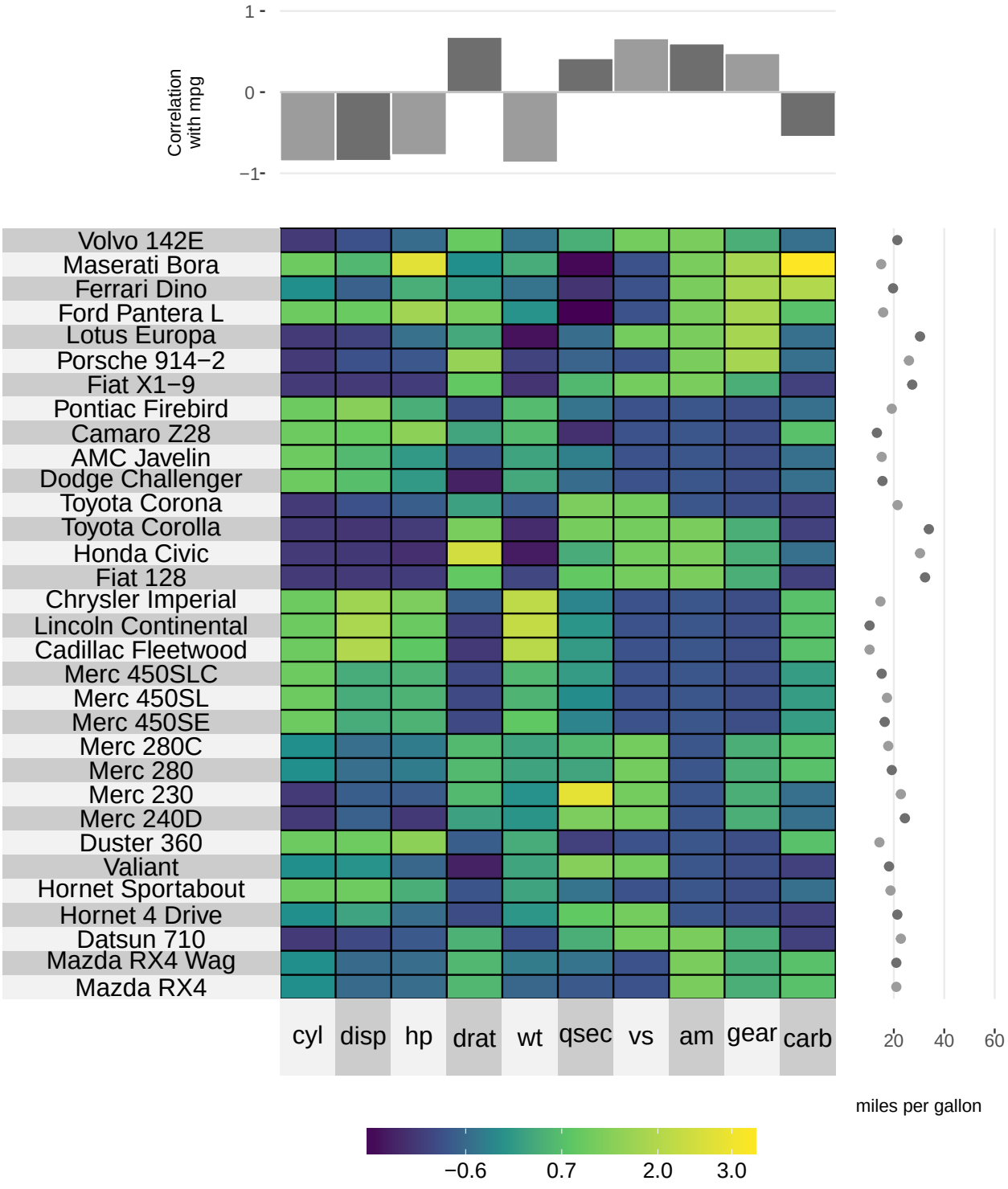
```
# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg),
          # scale the variables/columns
          scale = T,

          # add mpg as a scatterplot next to the rows
          yr = mtcars$mpg,
          yr.axis.name = "miles per gallon",

          # add correlation between each variable and miles per gallon
          yt = cor(mtcars)[-1,"mpg"],
          yt.plot.type = "bar",
          yt.axis.name = "Correlation\nwith mpg",
          yt.axis.size = 14,
          yt.axis.name.size = 14,

          left.label.size = 0.5,
          bottom.label.size = 0.1)
```



8.8.3 Limits

You can set the y-axis limits by using the `yr.lim` and `yt.lim` arguments. You must provide a vector of length 2 specifying the minimum and maximum values for the range respectively.

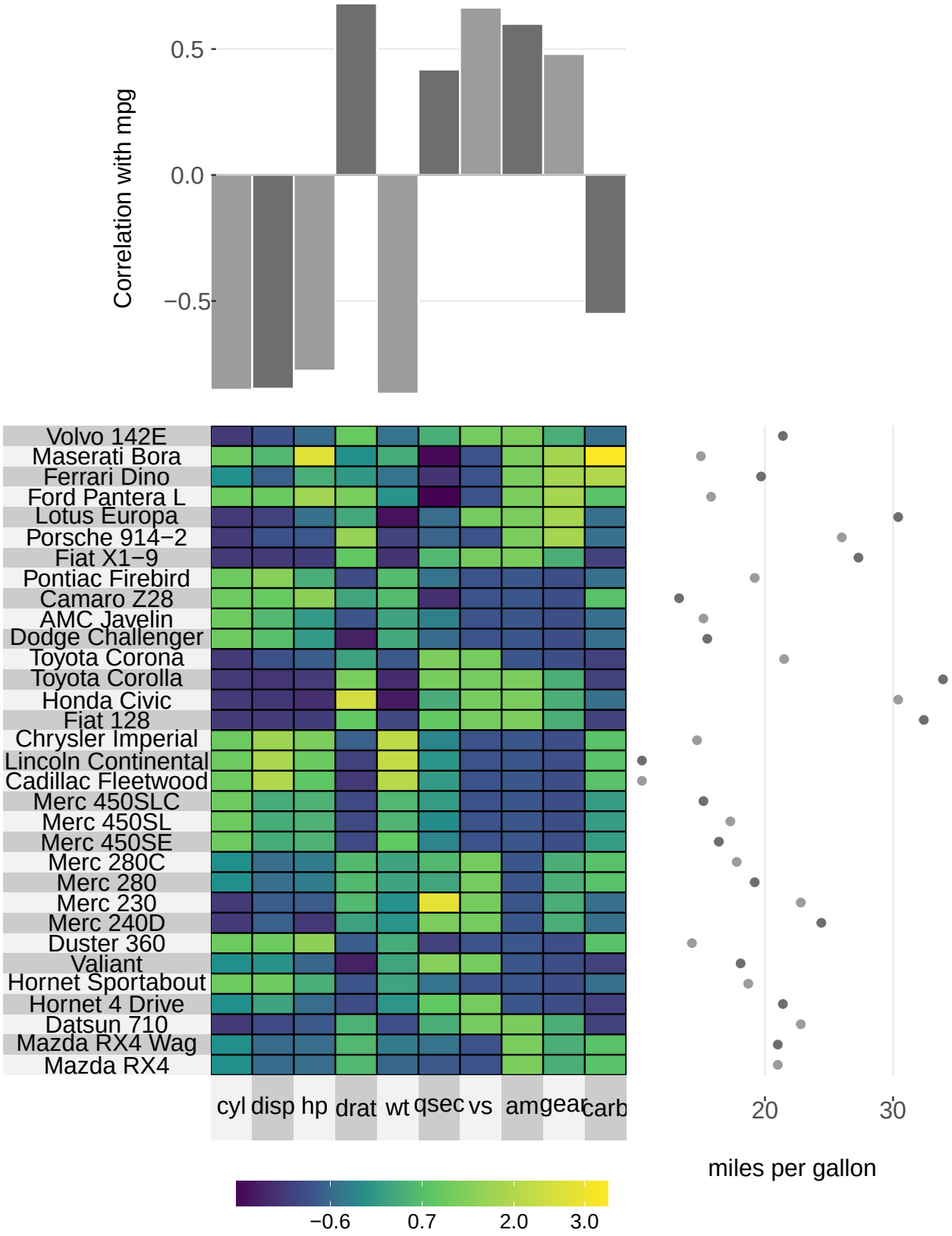
```r
# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg),
          # scale the variables/columns
          scale = T,

          # add mpg as a scatterplot next to the rows
          yr = mtcars$mpg,
          yr.axis.name = "miles per gallon",
          yr.lim = c(0, 60),

          # add correlation between each variable and miles per gallon
          yt = cor(mtcars)[-1,"mpg"],
          yt.plot.type = "bar",
          yt.axis.name = "Correlation\nwith mpg",
          yt.lim = c(-1.5, 1),

          left.label.size = 0.5,
          bottom.label.size = 0.1)
```



8.9 Plot size

The size of the entire adjacent plot can be determined using the `yr.plot.size`/`yr.plot.size` arguments.

```r
# plot a super heatmap
superheat(dplyr::select(mtcars, -mpg),
          # scale the variables/columns
          scale = T,

          # add mpg as a scatterplot next to the rows
          yr = mtcars$mpg,
          yr.axis.name = "miles per gallon",
          yr.axis.size = 14,
          yr.axis.name.size = 14,
          yr.plot.size = 0.8,

          # add correlation between each variable and miles per gallon
          yt = cor(mtcars)[-1,"mpg"],
          yt.plot.type = "bar",
          yt.axis.name = "Correlation with mpg",
          yt.axis.size = 14,
          yt.axis.name.size = 14,
          yt.plot.size = 0.7,

          left.label.size = 0.5,
          bottom.label.size = 0.1)
```

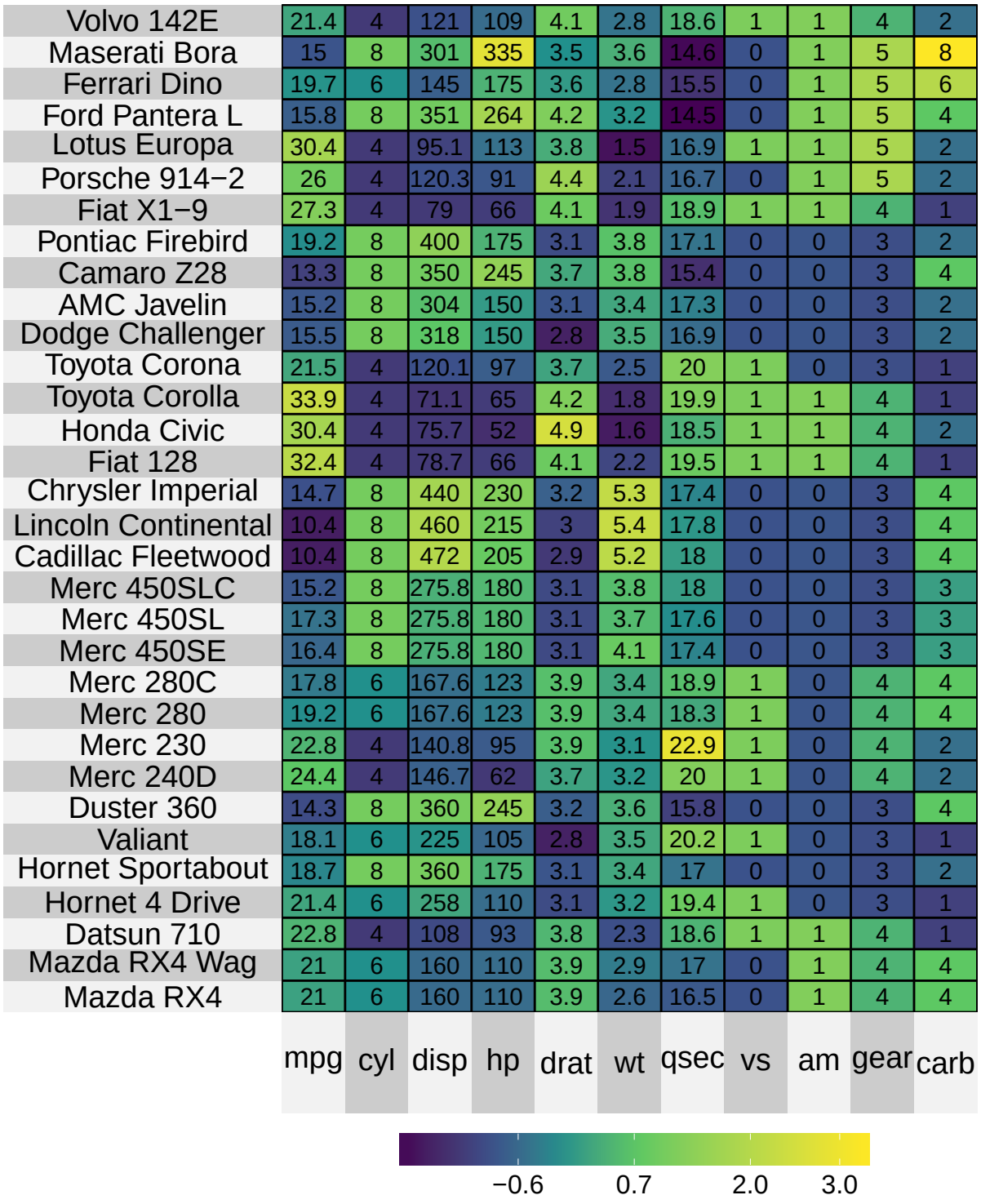

0.5
0.0
−0.5
Correlation with mpg
Volvo 142E
Maserati Bora
Ferrari Dino
Ford Pantera L
Lotus Europa
Porsche 914−2
Fiat X1−9
Pontiac Firebird
Camaro Z28
AMC Javelin
Dodge Challenger
Toyota Corona
Toyota Corolla
Honda Civic
Fiat 128
Chrysler Imperial
Lincoln Continental
Cadillac Fleetwood
Merc 450SLC
Merc 450SL
Merc 450SE
Merc 280C
Merc 280
Merc 230
Merc 240D
Duster 360
Valiant
Hornet Sportabout
Hornet 4 Drive
Datsun 710
Mazda RX4 Wag
Mazda RX4
cyl disp hp drat wt qsec vs am gear carb
−0.6 0.7 2.0 3.0
20 30
miles per gallon

Chapter 9

Adding text

It is easy to add text to each cell in the heatmap using the `X.text` argument. Below we simply plot the raw matrix over the top of the heatmap, but you could add any matrix of numbers or character strings.

```
superheat(X = mtcars, # heatmap matrix
          # change the size of the labels
          left.label.size = 0.4,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE,
          # add text matrix
          X.text = round(as.matrix(mtcars), 1),
          X.text.size = 4)
```



9.1 Text color

You can change the colors of the text by providing a single number or a matrix of numbers to the argument `X.text.col`. If providing a matrix, the dimension must be identical to that of the X matrix provided.

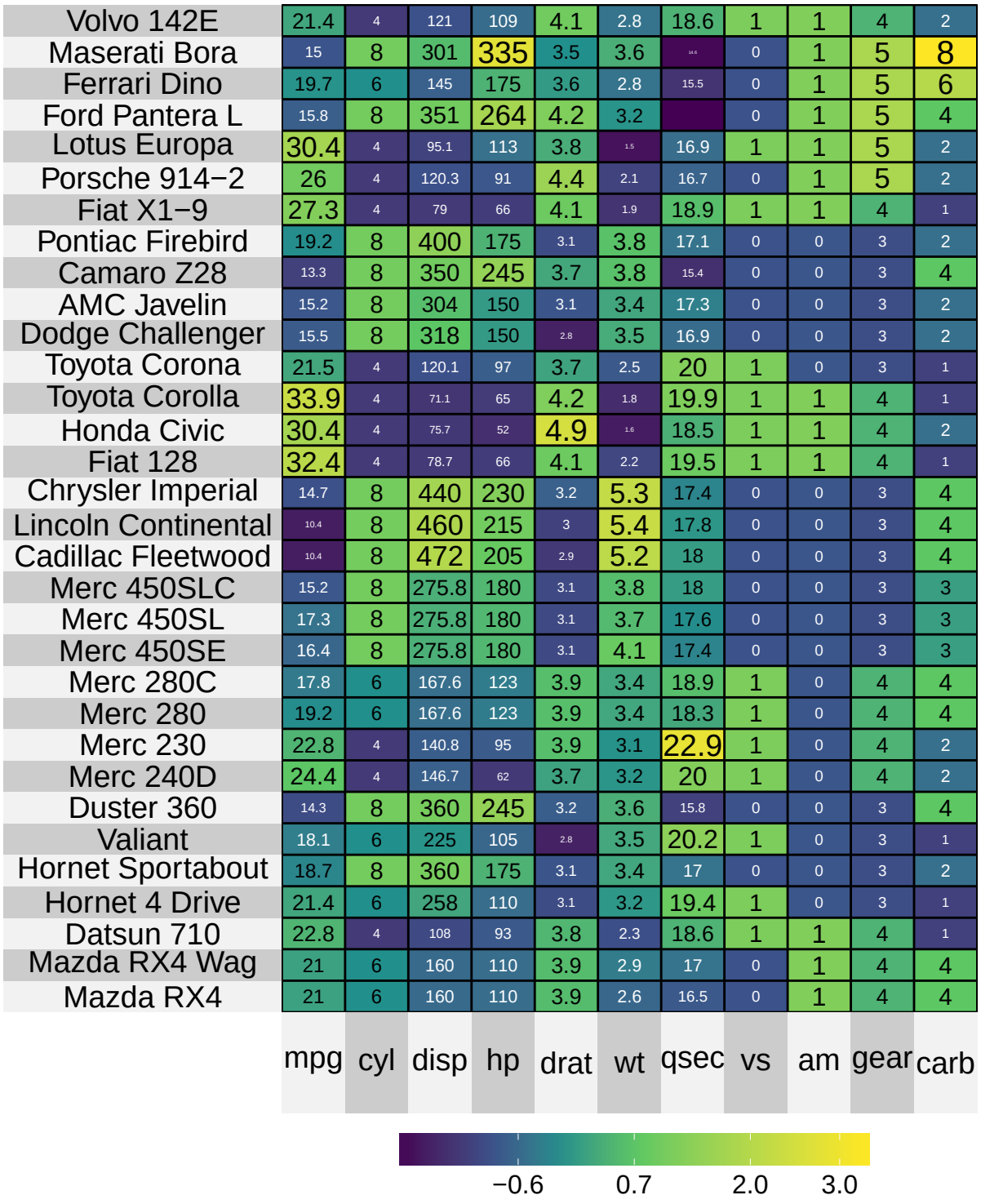
```
# set the text colors
# identify all scaled values that fall below -0.3
mtcars.col <- scale(mtcars) < -0.3
# set all values that satisfy the condition to "white"
mtcars.col <- gsub("TRUE", "white", mtcars.col)
# set all values that do not satisfy the condition to "black"
mtcars.col <- gsub("FALSE", "black", mtcars.col)
# convert to matrix
mtcars.col <- matrix(mtcars.col, ncol = ncol(mtcars))

superheat(X = mtcars, # heatmap matrix
          # change the size of the labels
          left.label.size = 0.4,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE,
          # add text matrix
          X.text = round(as.matrix(mtcars), 1),
          X.text.col = mtcars.col,
          X.text.size = 4)
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Volvo 142E	21.4	4	121	109	4.1	2.8	18.6	1	1	4	2
Maserati Bora	15	8	301	335	3.5	3.6	14.6	0	1	5	8
Ferrari Dino	19.7	6	145	175	3.6	2.8	15.5	0	1	5	6
Ford Pantera L	15.8	8	351	264	4.2	3.2	14.5	0	1	5	4
Lotus Europa	30.4	4	95.1	113	3.8	1.5	16.9	1	1	5	2
Porsche 914−2	26	4	120.3	91	4.4	2.1	16.7	0	1	5	2
Fiat X1−9	27.3	4	79	66	4.1	1.9	18.9	1	1	4	1
Pontiac Firebird	19.2	8	400	175	3.1	3.8	17.1	0	0	3	2
Camaro Z28	13.3	8	350	245	3.7	3.8	15.4	0	0	3	4
AMC Javelin	15.2	8	304	150	3.1	3.4	17.3	0	0	3	2
Dodge Challenger	15.5	8	318	150	2.8	3.5	16.9	0	0	3	2
Toyota Corona	21.5	4	120.1	97	3.7	2.5	20	1	0	3	1
Toyota Corolla	33.9	4	71.1	65	4.2	1.8	19.9	1	1	4	1
Honda Civic	30.4	4	75.7	52	4.9	1.6	18.5	1	1	4	2
Fiat 128	32.4	4	78.7	66	4.1	2.2	19.5	1	1	4	1
Chrysler Imperial	14.7	8	440	230	3.2	5.3	17.4	0	0	3	4
Lincoln Continental	10.4	8	460	215	3	5.4	17.8	0	0	3	4
Cadillac Fleetwood	10.4	8	472	205	2.9	5.2	18	0	0	3	4
Merc 450SLC	15.2	8	275.8	180	3.1	3.8	18	0	0	3	3
Merc 450SL	17.3	8	275.8	180	3.1	3.7	17.6	0	0	3	3
Merc 450SE	16.4	8	275.8	180	3.1	4.1	17.4	0	0	3	3
Merc 280C	17.8	6	167.6	123	3.9	3.4	18.9	1	0	4	4
Merc 280	19.2	6	167.6	123	3.9	3.4	18.3	1	0	4	4
Merc 230	22.8	4	140.8	95	3.9	3.1	22.9	1	0	4	2
Merc 240D	24.4	4	146.7	62	3.7	3.2	20	1	0	4	2
Duster 360	14.3	8	360	245	3.2	3.6	15.8	0	0	3	4
Valiant	18.1	6	225	105	2.8	3.5	20.2	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.1	3.4	17	0	0	3	2
Hornet 4 Drive	21.4	6	258	110	3.1	3.2	19.4	1	0	3	1
Datsun 710	22.8	4	108	93	3.8	2.3	18.6	1	1	4	1
Mazda RX4 Wag	21	6	160	110	3.9	2.9	17	0	1	4	4
Mazda RX4	21	6	160	110	3.9	2.6	16.5	0	1	4	4

9.2 Font size

You can change size of the text by providing a single number or a matrix of numbers to the argument `X.text.size`. If providing a matrix, the dimension must be identical to that of the X matrix provided.

```
# Set the size to simply be proportional to the scaled value
mtcars.size <- scale(mtcars) + 2
superheat(X = mtcars, # heatmap matrix
               # change the size of the labels
          left.label.size = 0.4,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE,
          # add text matrix
          X.text = round(as.matrix(mtcars), 1),
          X.text.col = mtcars.col,
          X.text.size = mtcars.size)
```



9.3 Text angle

You can change the angle of the text by providing a single number (the number of degrees between 0 and 360) or a matrix of numbers to the argument `X.text.angle`. If providing a matrix, the dimension must be identical to that of the X matrix provided.

```r
superheat(X = mtcars, # heatmap matrix
          # change the size of the labels
          left.label.size = 0.4,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE,
          # add text matrix
          X.text = round(as.matrix(mtcars), 1),
          X.text.col = mtcars.col,
          X.text.size = 4,
          X.text.angle = 12)
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Volvo 142E	21.4	4	121	109	4.1	2.8	18.6	1	1	4	2
Maserati Bora	15	8	301	335	3.5	3.6	14.6	0	1	5	8
Ferrari Dino	19.7	6	145	175	3.6	2.8	15.5	0	1	5	6
Ford Pantera L	15.8	8	351	264	4.2	3.2	14.5	0	1	5	4
Lotus Europa	30.4	4	95.1	113	3.8	1.5	16.9	1	1	5	2
Porsche 914−2	26	4	120.3	91	4.4	2.1	16.7	0	1	5	2
Fiat X1−9	27.3	4	79	66	4.1	1.9	18.9	1	1	4	1
Pontiac Firebird	19.2	8	400	175	3.1	3.8	17.1	0	0	3	2
Camaro Z28	13.3	8	350	245	3.7	3.8	15.4	0	0	3	4
AMC Javelin	15.2	8	304	150	3.1	3.4	17.3	0	0	3	2
Dodge Challenger	15.5	8	318	150	2.8	3.5	16.9	0	0	3	2
Toyota Corona	21.5	4	120.1	97	3.7	2.5	20	1	0	3	1
Toyota Corolla	33.9	4	71.1	65	4.2	1.8	19.9	1	1	4	1
Honda Civic	30.4	4	75.7	52	4.9	1.6	18.5	1	1	4	2
Fiat 128	32.4	4	78.7	66	4.1	2.2	19.5	1	1	4	1
Chrysler Imperial	14.7	8	440	230	3.2	5.3	17.4	0	0	3	4
Lincoln Continental	10.4	8	460	215	3	5.4	17.8	0	0	3	4
Cadillac Fleetwood	10.4	8	472	205	2.9	5.2	18	0	0	3	4
Merc 450SLC	15.2	8	275.8	180	3.1	3.8	18	0	0	3	3
Merc 450SL	17.3	8	275.8	180	3.1	3.7	17.6	0	0	3	3
Merc 450SE	16.4	8	275.8	180	3.1	4.1	17.4	0	0	3	3
Merc 280C	17.8	6	167.6	123	3.9	3.4	18.9	1	0	4	4
Merc 280	19.2	6	167.6	123	3.9	3.4	18.3	1	0	4	4
Merc 230	22.8	4	140.8	95	3.9	3.1	22.9	1	0	4	2
Merc 240D	24.4	4	146.7	62	3.7	3.2	20	1	0	4	2
Duster 360	14.3	8	360	245	3.2	3.6	15.8	0	0	3	4
Valiant	18.1	6	225	105	2.8	3.5	20.2	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.1	3.4	17	0	0	3	2
Hornet 4 Drive	21.4	6	258	110	3.1	3.2	19.4	1	0	3	1
Datsun 710	22.8	4	108	93	3.8	2.3	18.6	1	1	4	1
Mazda RX4 Wag	21	6	160	110	3.9	2.9	17	0	1	4	4
Mazda RX4	21	6	160	110	3.9	2.6	16.5	0	1	4	4

Chapter 10

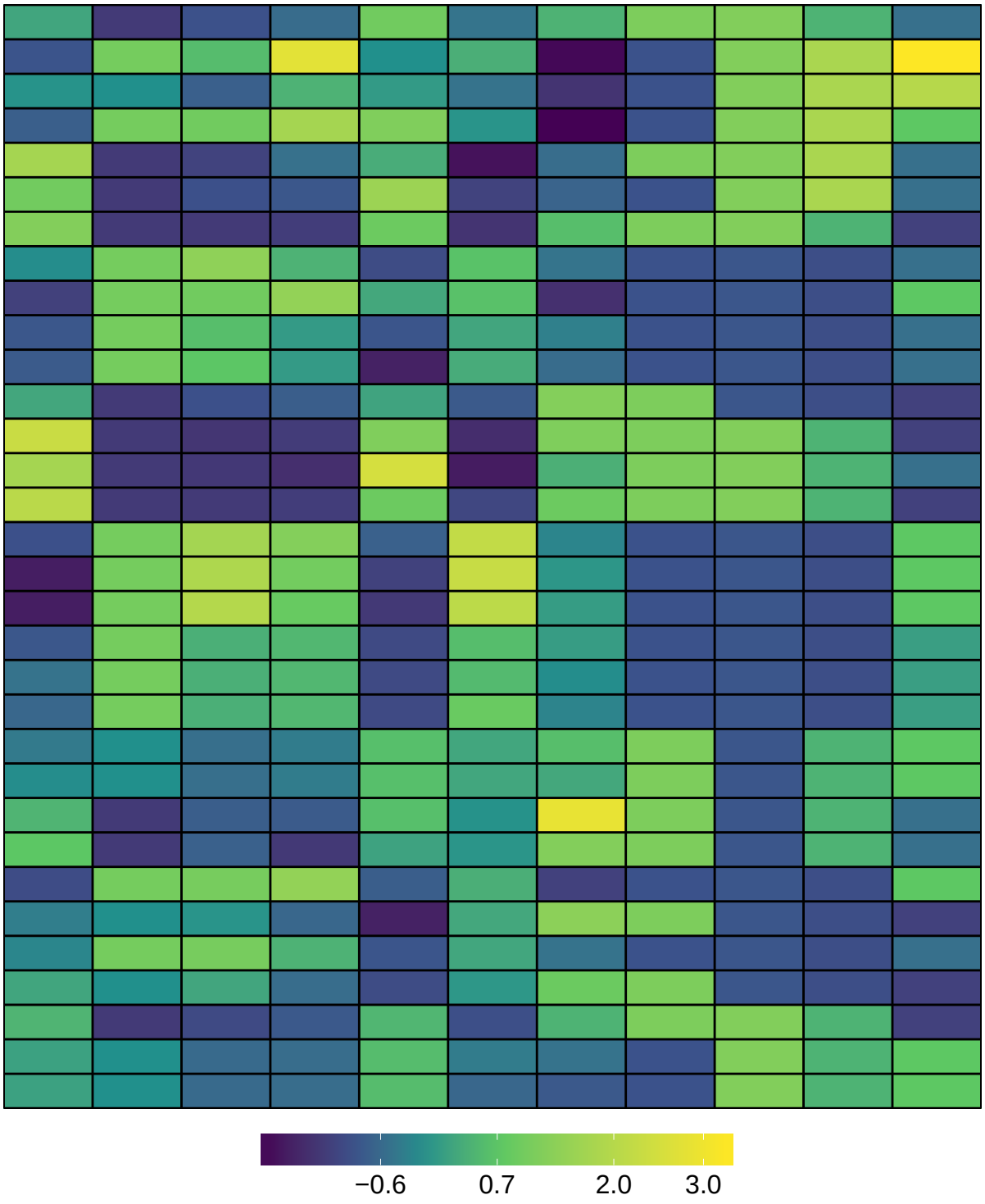
Labels

The default labels alternate between light and dark grey, and disappear when you have more than 100 rows or columns. In this situation, to force the row or column labels to appear, you can set `force.left.label = TRUE`/`force.bottom.label = TRUE`.

10.1 Removing the labels

To remove either the row or column labels, you can use the `left.label` and `bottom.label` arguments.

```
superheat(X = mtcars, # heatmap matrix
          # remove the labels
          left.label = "none",
          bottom.label = "none",
          # scale the matrix columns
          scale = TRUE)
```



Recall that if you have clustered your matrix and you want the labels to show the variable names, you can set `left.label = "variable"` and `bottom.label = "variable"`

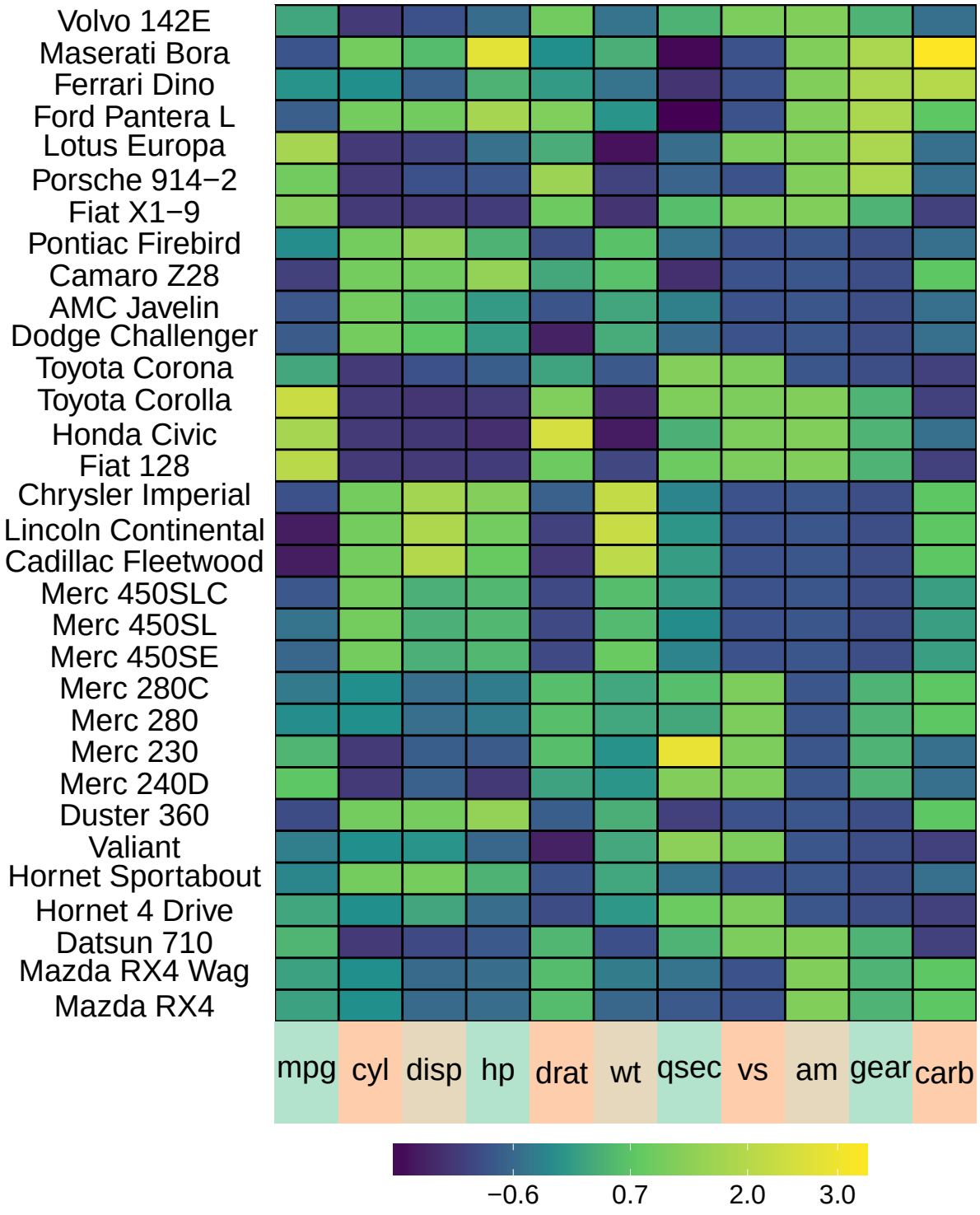
10.2 Label size

We have already seen changing the size of the labels using `left.label.size` and `bottom.label.size`. Note that this changes the size of the label bar but does not change the text size.

10.3 Label color

Changing the color of the labels can be done using the `left.label.col` and `bottom.label.col` arguments. You can provide a single color or you can provide a vector of colors (in which case this vector will be cycled through to fill the length of the labels.)

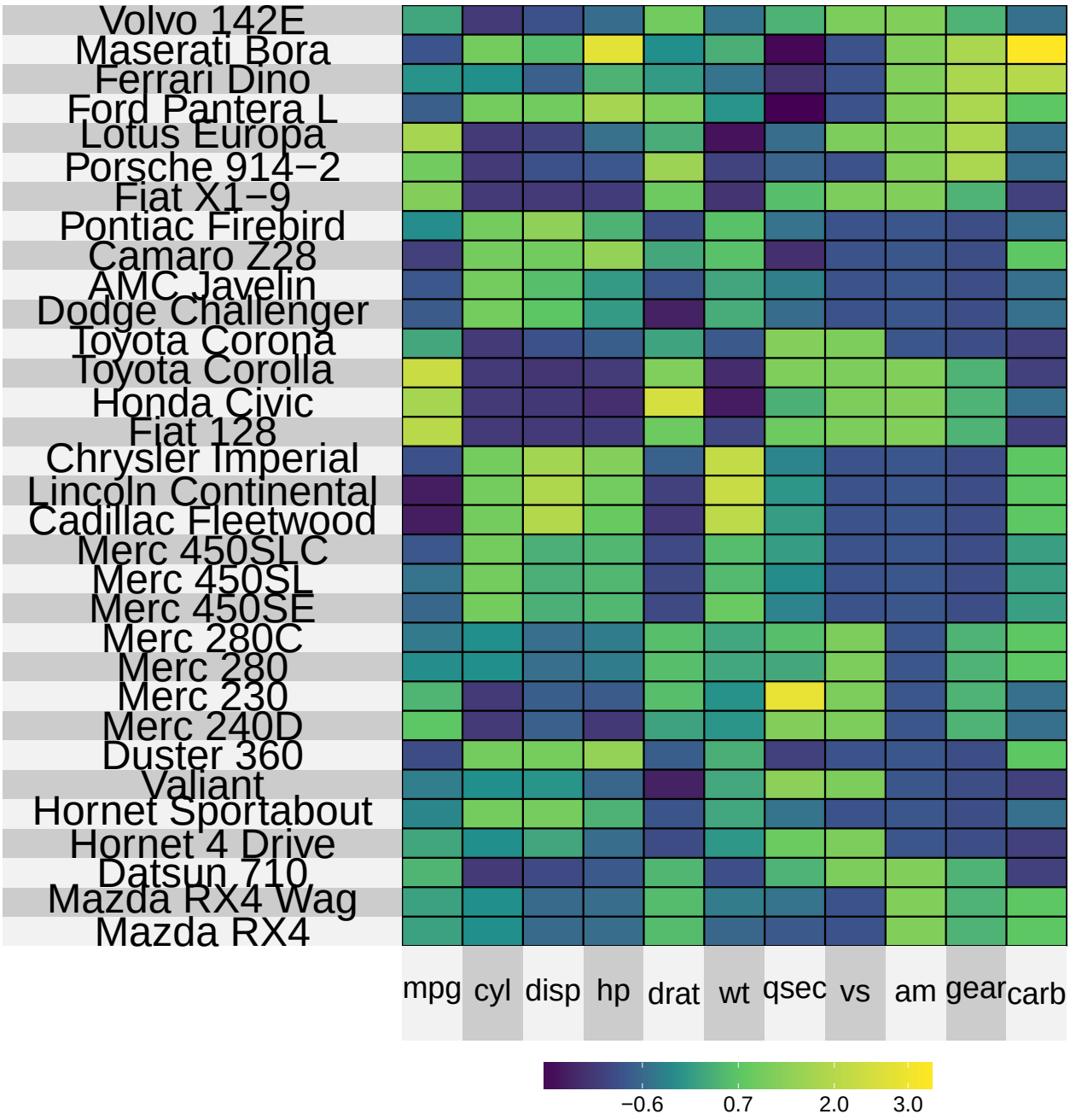
```
superheat(X = mtcars, # heatmap matrix
          # change the size of the labels
          left.label.size = 0.4,
          bottom.label.size = 0.1,
          # change the color of the labels
          left.label.col = "white",
          bottom.label.col = c("#b3e2cd","#fdcdac","#e5d8bd"),
          # scale the matrix columns
          scale = TRUE)
```



10.4 Text size

The size of the label text can be changed using the `left.label.text.size` and `bottom.label.text.size` arguments.

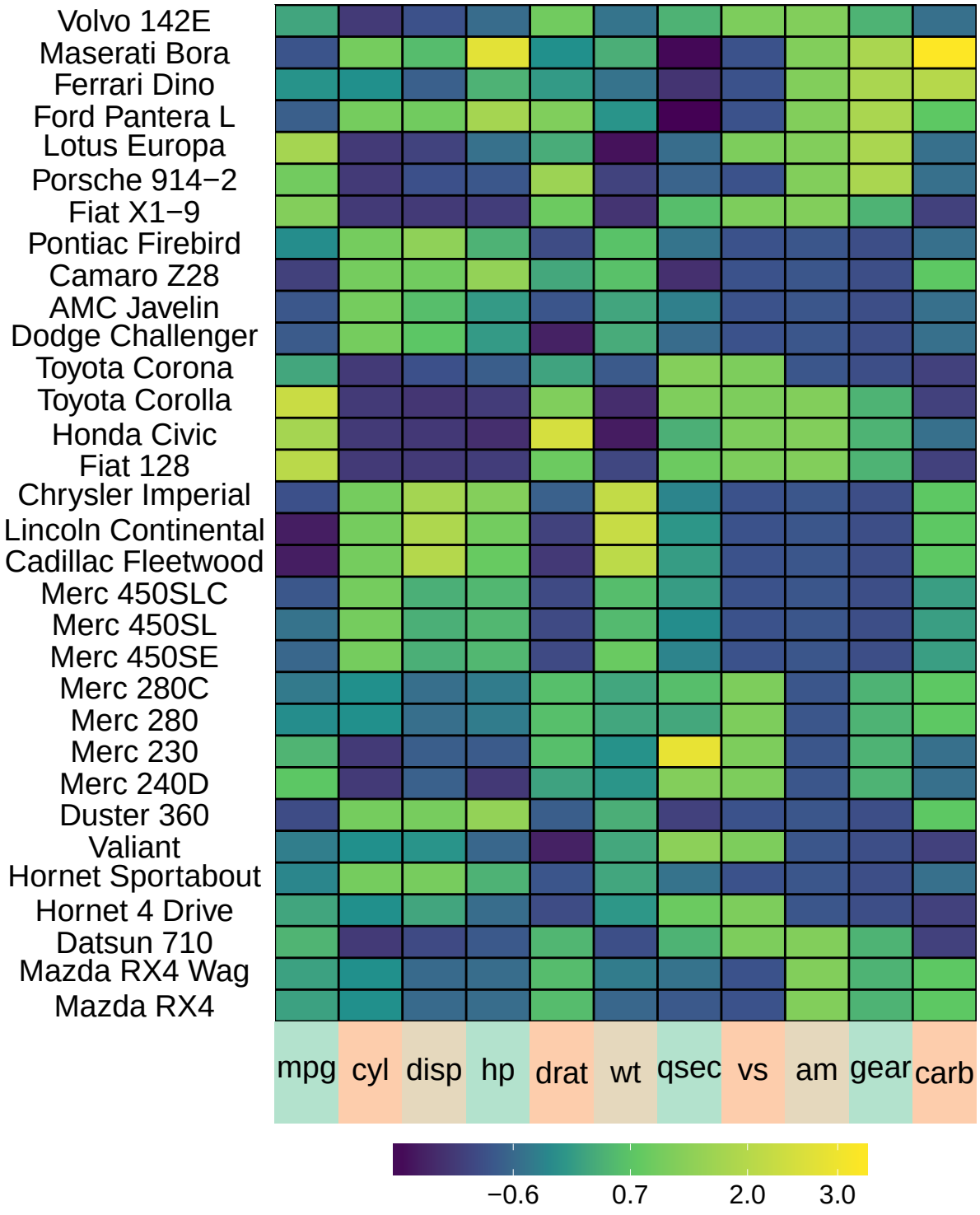
```
superheat(X = mtcars, # heatmap matrix
          # change the size of the labels
          left.label.size = 0.6,
          bottom.label.size = 0.1,
          # change the size of the label text
          left.label.text.size = 8,
          bottom.label.text.size = 6,
          # scale the matrix columns
          scale = TRUE)
```



10.5 Text color

Changing the color of the text can be achieved using the `left.label.text.col` and `bottom.label.text.col` arguments.

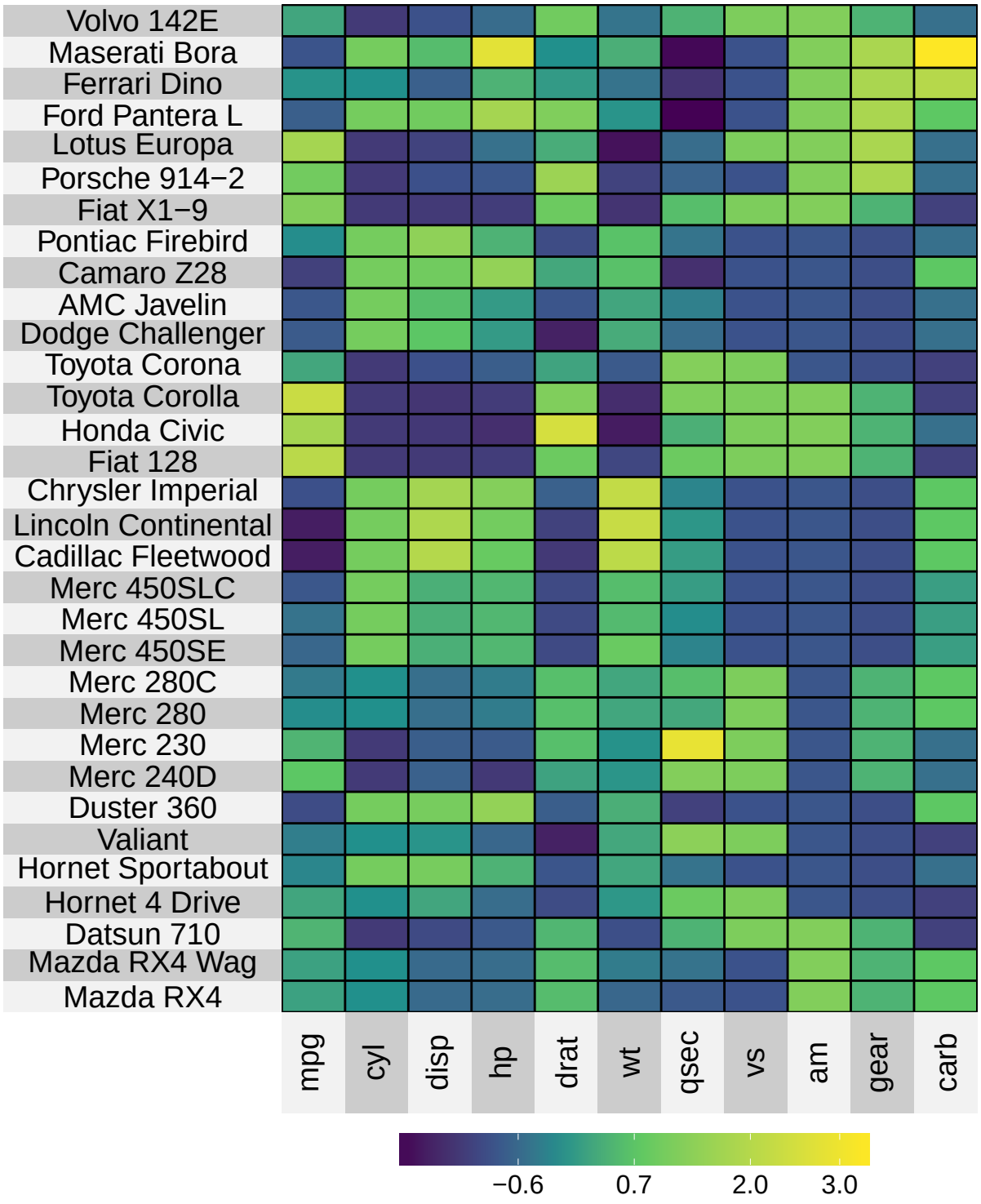
```
superheat(X = mtcars, # heatmap matrix
          # change the size of the labels
          left.label.size = 0.4,
          bottom.label.size = 0.1,
          # change the color of the label text
          left.label.col = "white",
          bottom.label.col = c("#b3e2cd","#fdcdac","#e5d8bd"),
          # scale the matrix columns
          scale = TRUE)
```



10.6 Text angle

By default, the text has angle 0. Rotating the text by x degrees can be achieved using `left.label.text.angle = x` or `bottom.label.text.angle = x`.

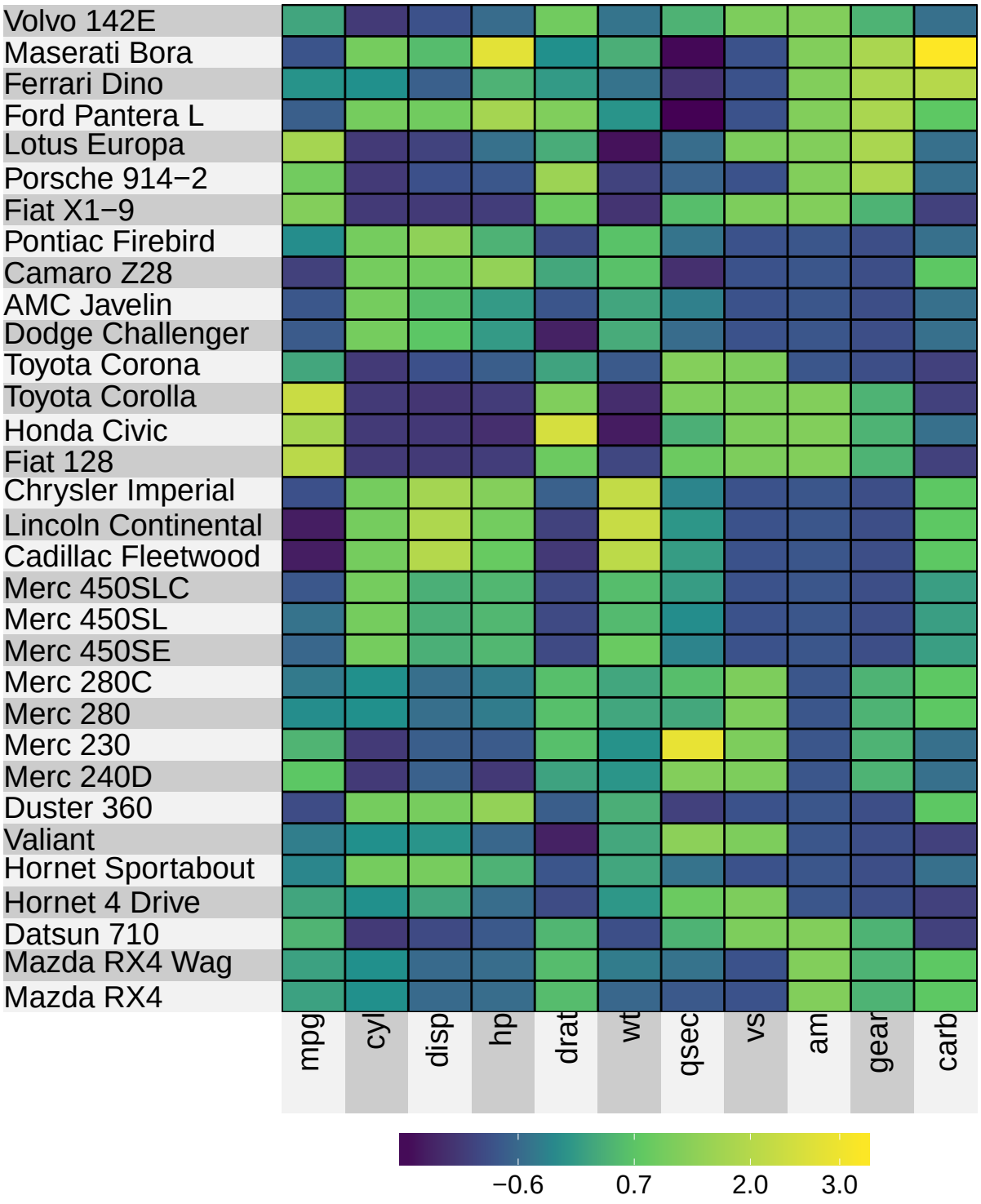
```
superheat(X = mtcars, # heatmap matrix
          # change the size of the labels
          left.label.size = 0.4,
          bottom.label.size = 0.1,
          # change the angle of the label text
          bottom.label.text.angle = 90,
          # scale the matrix columns
          scale = TRUE)
```



10.7 Text alignment

By default, the text is center-aligned. Changing to right-aligned or left-aligned can be achieved using the `left.label.text.alignment` and `bottom.label.text.alignment` arguments.

```
superheat(X = mtcars, # heatmap matrix
          # change the size of the labels
          left.label.size = 0.4,
          bottom.label.size = 0.1,
          # change the angle of the label text
          bottom.label.text.angle = 90,
          left.label.text.alignment = "left",
          bottom.label.text.alignment = "right",
          # scale the matrix columns
          scale = TRUE)
```


Volvo 142E
Maserati Bora
Ferrari Dino
Ford Pantera L
Lotus Europa
Porsche 914−2
Fiat X1−9
Pontiac Firebird
Camaro Z28
AMC Javelin
Dodge Challenger
Toyota Corona
Toyota Corolla
Honda Civic
Fiat 128
Chrysler Imperial
Lincoln Continental
Cadillac Fleetwood
Merc 450SLC
Merc 450SL
Merc 450SE
Merc 280C
Merc 280
Merc 230
Merc 240D
Duster 360
Valiant
Hornet Sportabout
Hornet 4 Drive
Datsun 710
Mazda RX4 Wag
Mazda RX4
mpg
cyl
disp
hp
drat
wt
qsec
vs
am
gear
carb
−0.6
0.7
2.0
3.0

Chapter 11

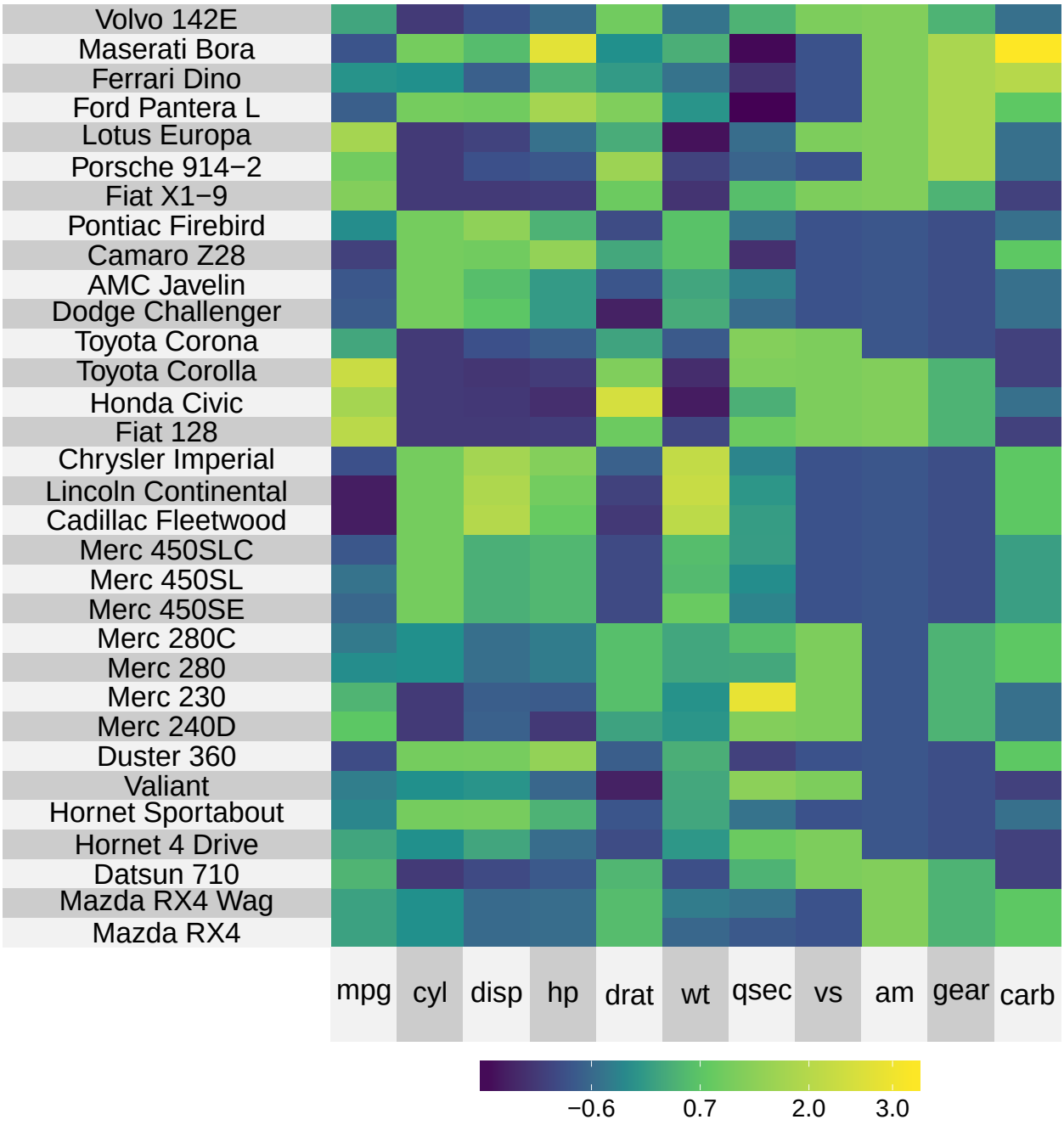
Grid aesthetics

The cell grid can be customized as you like in terms of color, size.

11.1 Removing the grid

Removing the grid lines is achieved by setting `grid.hline = FALSE` and `grid.vline = FALSE`.

```
set.seed(2016113)
superheat(mtcars,
          # change the size of the labels
          left.label.size = 0.45,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE,
          # remove the grid
          grid.hline = FALSE,
          grid.vline = FALSE)
```

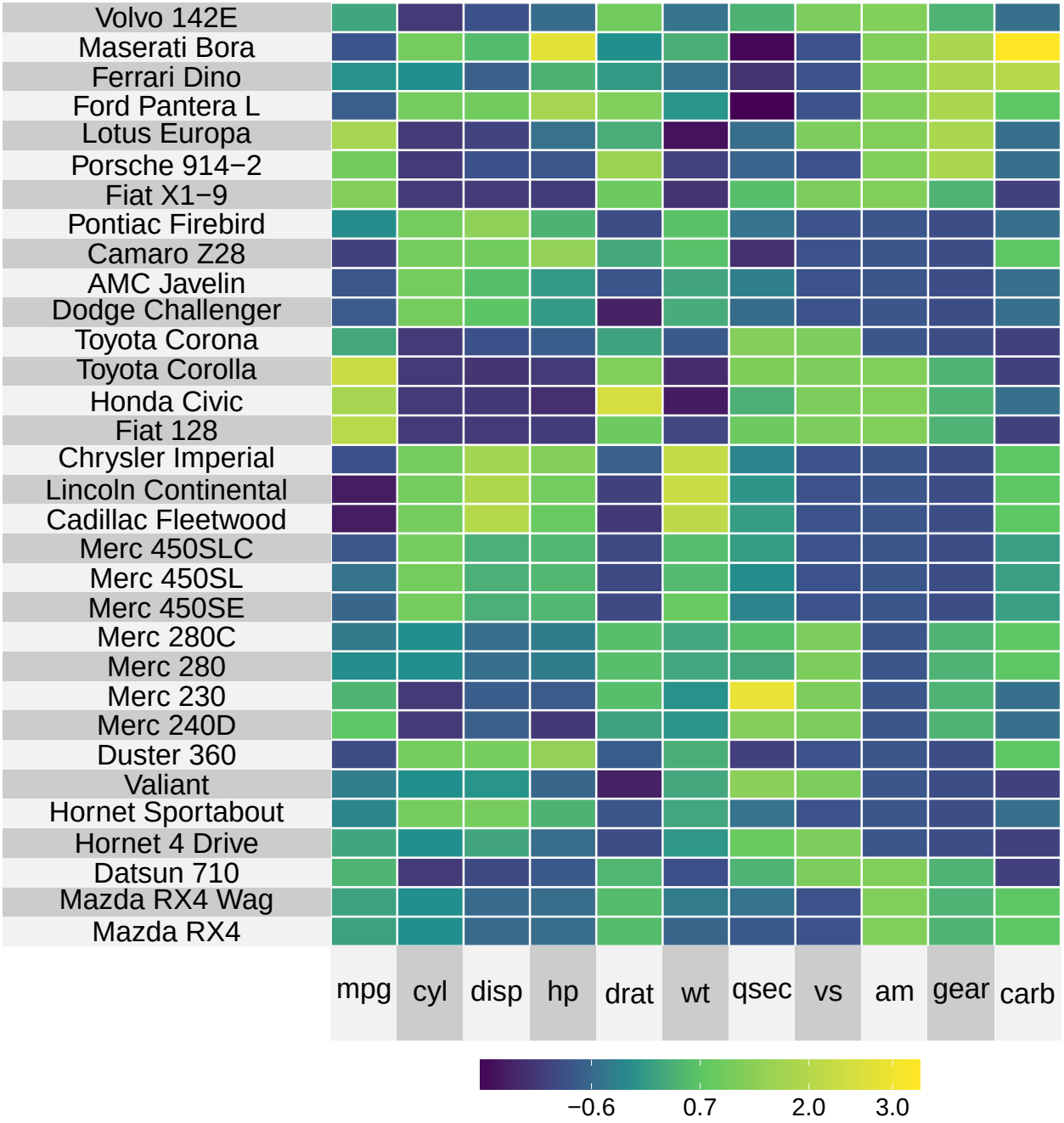


11.2 Grid color

One particularly nice setting is to make the grid lines white. Changing the color can be done using `grid.hline.col` and `grid.vline.col`.

```
set.seed(2016113)
superheat(mtcars,
          # change the size of the labels
          left.label.size = 0.45,
          bottom.label.size = 0.1,
```

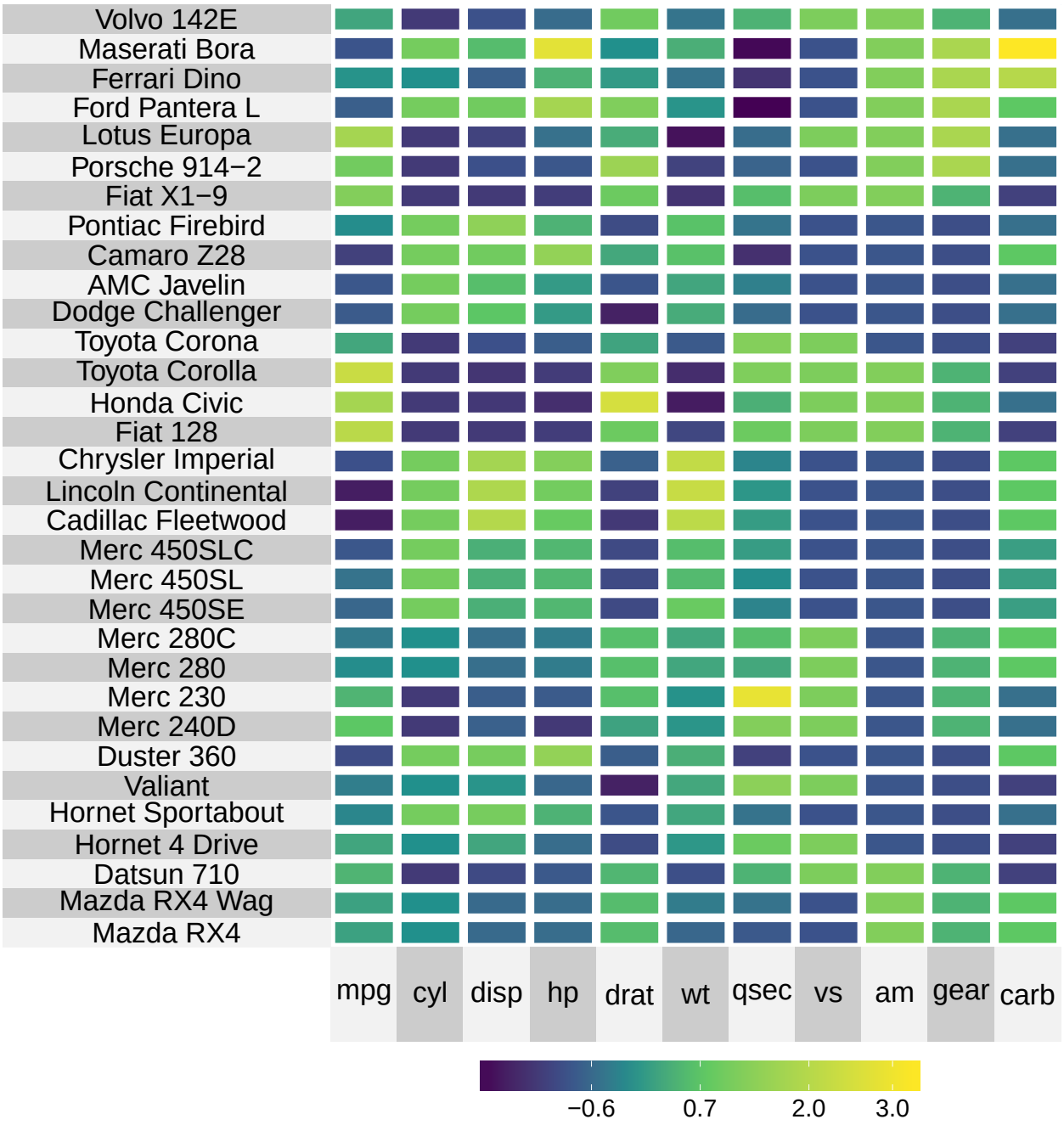
```
        # scale the matrix columns
        scale = TRUE,
        # change the grid color
        grid.hline.col = "white",
        grid.vline.col = "white")
```



11.3 Grid size

The grid lines can be made thicker or thinner using the `grid.hline.size` and `grid.vline.size` arguments.

```
set.seed(2016113)
superheat(mtcars,
          # change the size of the labels
          left.label.size = 0.45,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE,
          # change the grid size
          grid.hline.col = "white",
          grid.vline.col = "white",
          grid.hline.size = 2,
          grid.vline.size = 2)
```

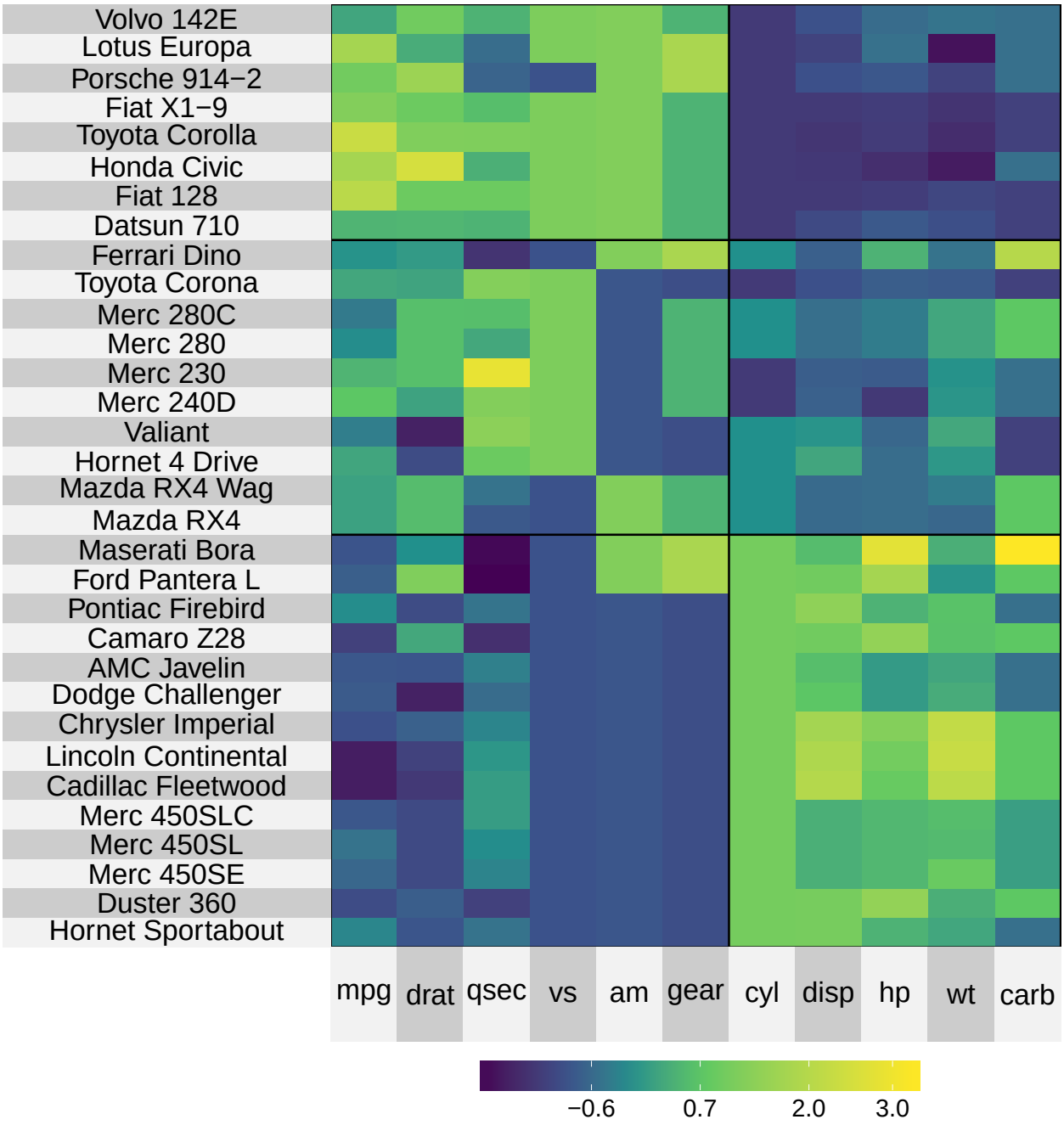


11.4 Clustered grid

When clustering the heatmap, the grid lines are placed around the clusters rather than the individual rows and columns. Specifically, this helps present the clusters when forcing the left and bottom labels to be the variable names.

```
set.seed(2016113)
superheat(mtcars,
          # change the size of the labels
          left.label.size = 0.45,
```

```
        bottom.label.size = 0.1,
        # scale the matrix columns
        scale = TRUE,

        # cluster the heatmap
        n.clusters.rows = 3,
        left.label = "variable",
        n.clusters.cols = 2,
        bottom.label = "variable")
```



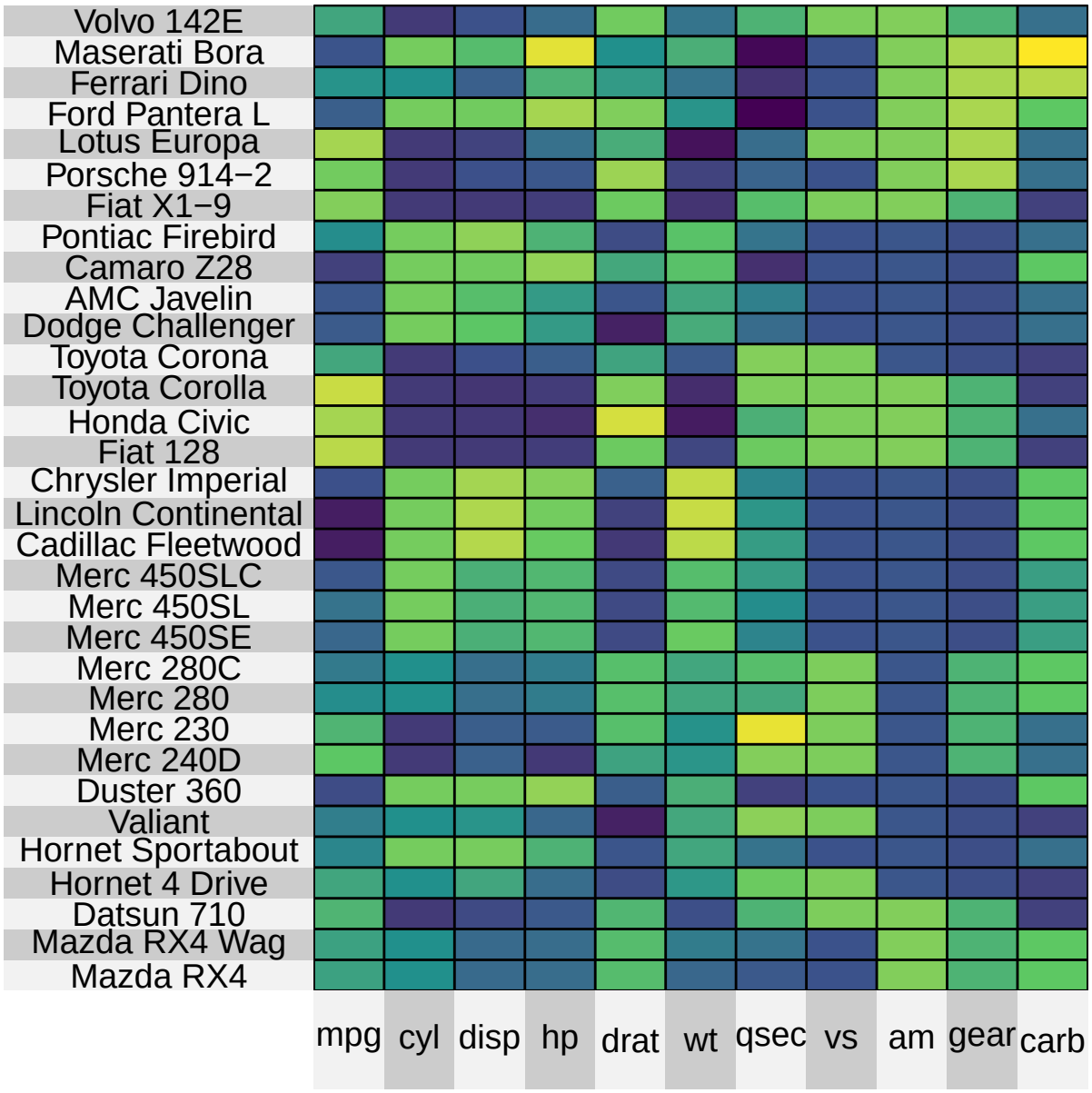
Chapter 12

Legend

12.1 Removing the legend

Removing the legend entirely can be achieved by setting `legend = FALSE`.

```
superheat(mtcars,
          # change the size of the labels
          left.label.size = 0.4,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE,

          # remove the legend
          legend = FALSE)
```

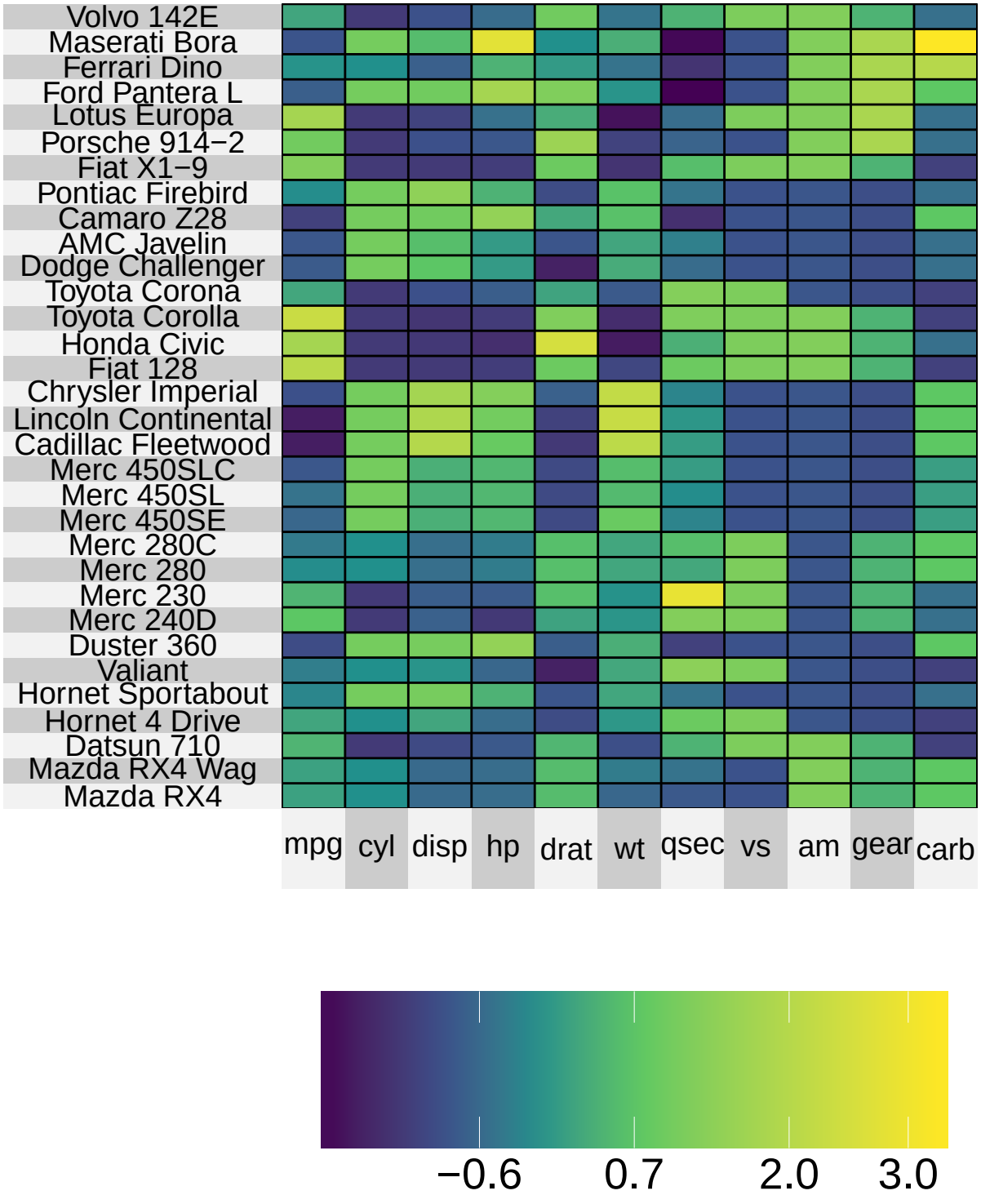


12.2 Size

Changing the size of the legend can be achieved by setting `legend.height` and `legend.width`. The size of the text can be set using `legend.text.size`.

```
superheat(mtcars,
          # change the size of the labels
          left.label.size = 0.4,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE,

          # make the legend bigger
          legend.height = 0.5,
```

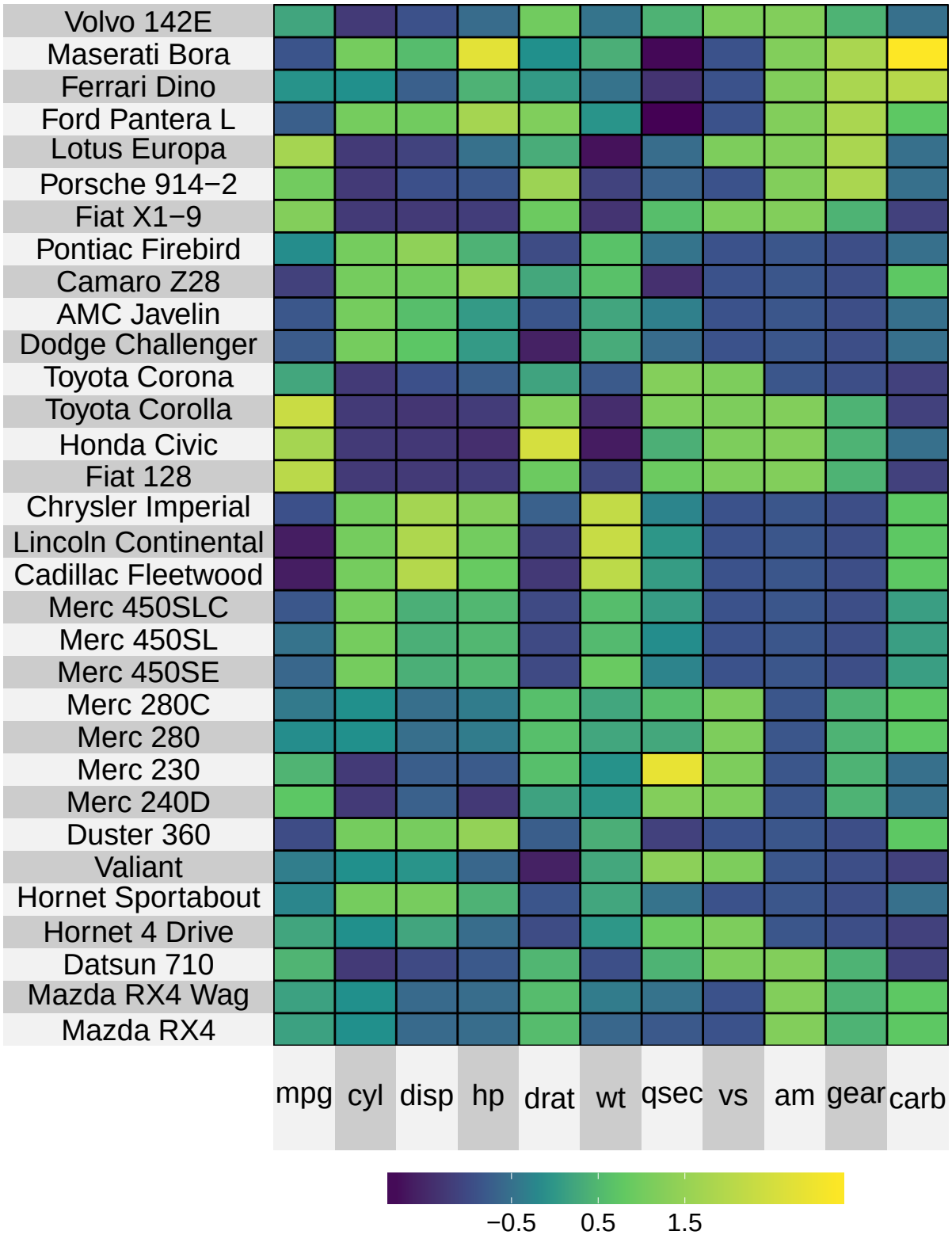
```
          legend.width = 2,
          legend.text.size = 20)
```



12.3 Legend breaks

Sometimes it is nice to customize the legend even further. The `legend.breaks` argument can be used to specify the legend break values (i.e. where the numbers and ticks appear). This argument does not mess with the actual color range or presentation.

```
superheat(mtcars,
          # change the size of the labels
          left.label.size = 0.4,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE,

          # specify the legend breaks
          legend.breaks = c(-0.5, 0.5, 1.5))
```


Volvo 142E
Maserati Bora
Ferrari Dino
Ford Pantera L
Lotus Europa
Porsche 914−2
Fiat X1−9
Pontiac Firebird
Camaro Z28
AMC Javelin
Dodge Challenger
Toyota Corona
Toyota Corolla
Honda Civic
Fiat 128
Chrysler Imperial
Lincoln Continental
Cadillac Fleetwood
Merc 450SLC
Merc 450SL
Merc 450SE
Merc 280C
Merc 280
Merc 230
Merc 240D
Duster 360
Valiant
Hornet Sportabout
Hornet 4 Drive
Datsun 710
Mazda RX4 Wag
Mazda RX4
mpg
cyl
disp
hp
drat
wt
qsec
vs
am
gear
carb
−0.5
0.5
1.5

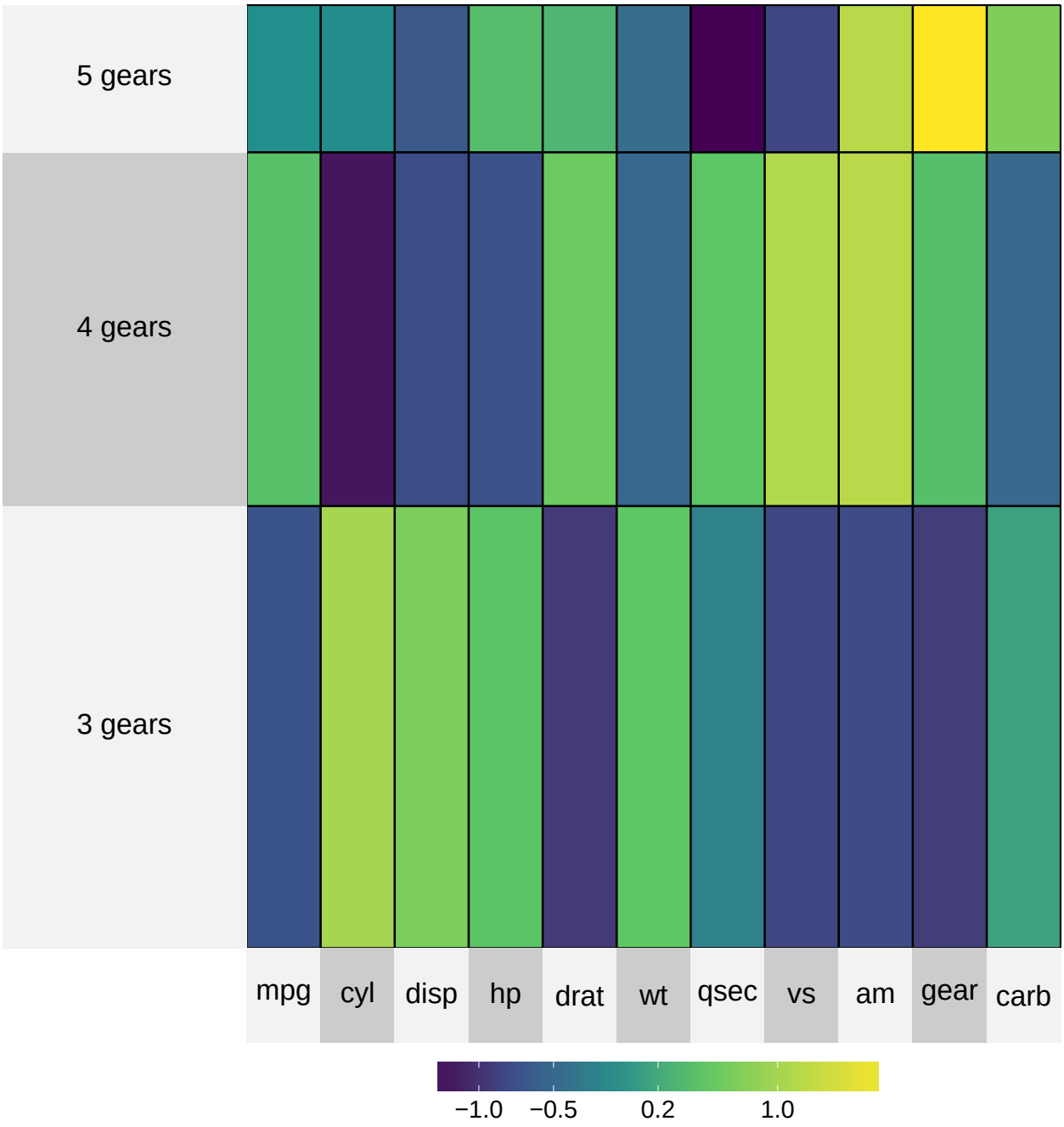
Chapter 13

Smoothing in high dimensions

In situations where you are plotting a very large matrix, often the heatmap becomes obscured by noise due to the sheer amount of information being presented relative to the number of pixels available.

To address this common issue, we have allowed for smoothing by summarizing values within a cluster by the median value. This can be achieved using the `smooth.heat` argument.

```
gears <- paste(mtcars$gear, "gears")

set.seed(2016113)
superheat(mtcars,
          # change the size of the labels
          left.label.size = 0.3,
          bottom.label.size = 0.1,
          # scale the matrix columns
          scale = TRUE,

          # cluster by gears
          membership.rows = gears,

          # place each variable in its own cluster
          membership.cols = 1:ncol(mtcars),
          bottom.label = "variable",

          # smooth the heatmap within clusters
          smooth.heat = TRUE)
```


5 gears
4 gears
3 gears
mpg
cyl
disp
hp
drat
wt
qsec
vs
am
gear
carb
−1.0
−0.5
0.2
1.0

Chapter 14

Saving superheatmaps

The best format for saving superheat images is as a .png file. To do this in R, the easiest way is to use the `png()` function (remember to call `dev.off()` when you're done!)

```
png("superheat.png", height = 900, width = 800)
superheat(X = mtcars, scale = T)
dev.off()
```

Chapter 15

Conclusion

Thanks for using the package! I hope you find it helpful in your data exploration adventures. The github development page can be found at https://github.com/rlbarter/superheat. For pull requests and suggestions please follow the standard protocol. For pressing questions or comments feel free to email me at rebeccabarter@berkeley.edu.