Notes on pheatmap

Code

In this note on pheatmap, I will create a dataset (a surrogate matrix) containing random numbers, with arbitrary column names and row names. Then I will add three different layer of information (1 layer along the columns and 2 layers along the rows) on the heatmap. The cells of the heatmap will be ordered according to the value of one layer of information (group1) along the column.

To begin with, let's load necessary packages -

Hide

```
# if (!requireNamespace("BiocManager", quietly = TRUE))
#   install.packages("BiocManager")
#
# BiocManager::install("DESeq")
library(pheatmap)
library(RColorBrewer)
library(tibble)
library(dplyr)
```

```
Registered S3 method overwritten by 'dplyr':
  method           from
  print.rowwise_df

Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

Hide

```
library(repr)
```

```
Registered S3 methods overwritten by 'htmltools':
  method               from
  print.html           tools:rstudio
  print.shiny.tag      tools:rstudio
  print.shiny.tag.list tools:rstudio
```

Hide

```
library(viridis)
```

```
Loading required package: viridisLite
```

I'll set a seed for generating 100 random numbers with a mean of 0 and standard deviation of 1.

Hide

```
set.seed(1986)
mat <- matrix(rnorm(100), ncol = 5)
```

Now, I'm generating arbitrary row names each with 3 small case random letters and column names each with 4 upper case random letters. The steps are commented out for convenience -

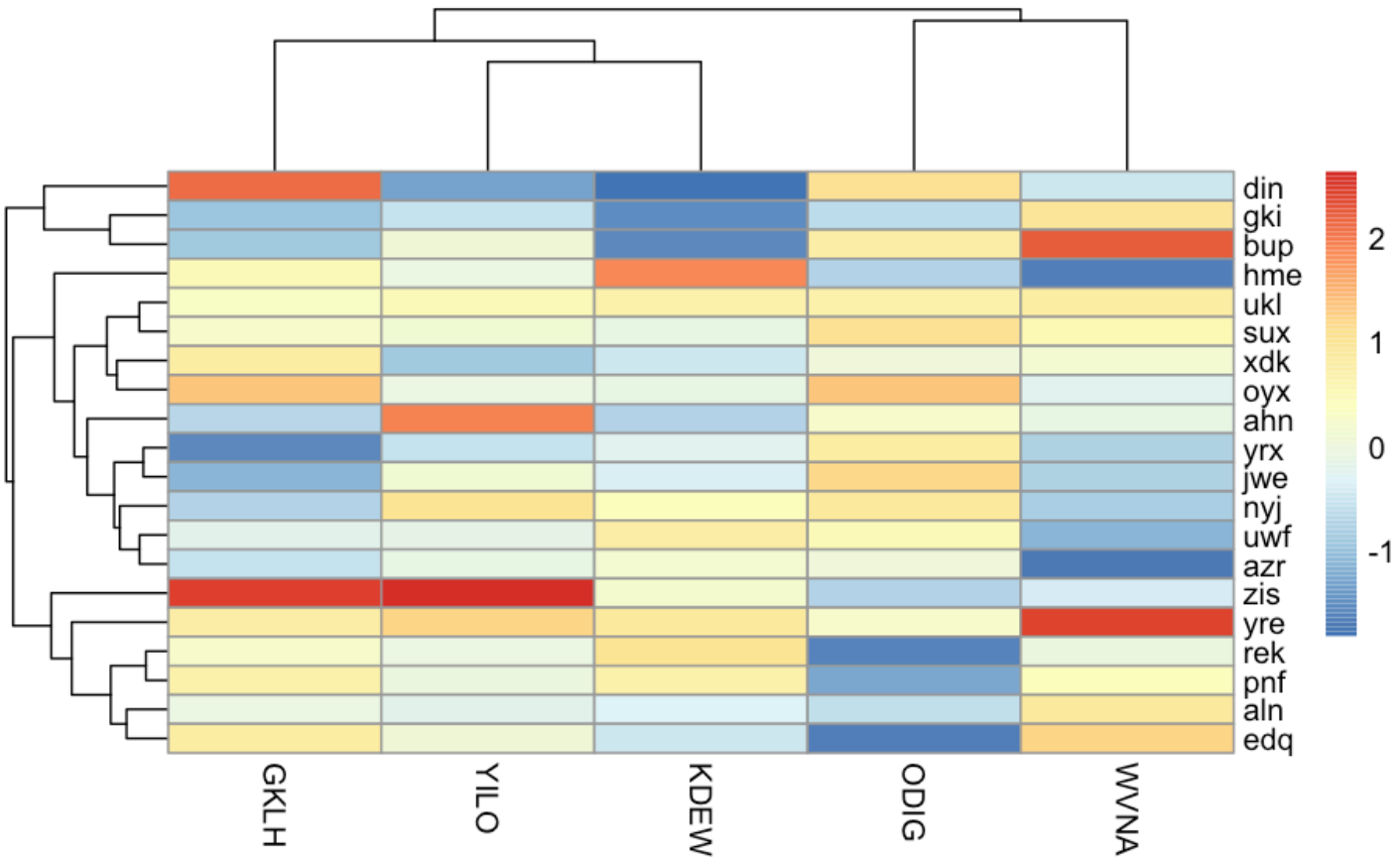
Hide

```
# sample(letters) # "s" "b" "p" "y" "n" "d" "l" "i" "a" "t" "r" "j" "h" "o" "f" "q
" "u" "z" "k" "w" "c" "v" "x" "m" "g" "e"
# paste(sample(letters), collapse = "") # "dijehquczxagoflnbswtvmkpyr"
# substr("dijehquczxagoflnbswtvmkpyr", start = 1, stop = 3) # "dij"
# substr(paste(sample(letters), collapse = ""), start = 1, stop = 3) # "obv"
# replicate(10, substr(paste(sample(letters), collapse = ""), start = 1, stop = 3)
) # "nke" "dot" "owy" "rnm" "yug" "tak" "cdu" "iac" "rpz" "biy"

rownames(mat) <- replicate(
  n = nrow(mat),
  expr = substr(
    x = paste(sample(letters), collapse = ""),
    start = 1,
    stop = 3
  )
)

colnames(mat) <- replicate(
  n = ncol(mat),
  expr = substr(
    x = paste(sample(LETTERS), collapse = ""),
    start = 1,
    stop = 4
  )
)
```

Just to look at the initial view of the heatmap -

Hide

```
pheatmap(mat = mat)
```



Now I want to add two random levels (1 and 2) for each of the column variables, as if they belong to either group (howerver, may not form 2 different clusters in the end) -

Hide

```
# sample(c(1,2),5,replace = T)
# col_annotation <- data.frame(group=sample(c(1,2),5,replace = T))
col_annotation <- data.frame(
  sample = as.factor(sample(x = c(1, 2), size = ncol(mat), replace = T))
  )

rownames(col_annotation) <- colnames(mat) # the row names here should be the same
as the col name of the mat
```

Adding random levels 1 to 5 for each of the rows (bythe name “group1”) -

Hide

```
row_annotation <- data.frame(
  group1 = as.factor(sample(x = seq(1:5),
                            size = nrow(mat),
                            replace = T))
) # without the as.factor, the group value will be considered as continuous on the
plot

row_annotation$group2 <- as.factor(sample(x = seq(1:10),
                                          size = nrow(mat),
                                          replace = T))

rownames(row_annotation) <- rownames(mat) # the row names hree should be the same
as the row name of the mat
```

I want the heatmap sorted by the levels of group1 (of row_annotation dataframe). Therefore, I have to order the row_annotation dataframe according to group1 values. I will use this ordered row names to arrange the mat dataset in the pheatmap() function.

Hide

```
# row_annotation <- row_annotation %>% tibble::rownames_to_column("x") %>%  dplyr:
:arrange(by=group1)  %>% tibble::column_to_rownames("x") # with dplyr, the row nam
e is not retained, so I had to use tibble::rownames_to_column("a new col name") to
keep the row names and then give it back with tibble::column_to_rownames("the same
col name")
row_annotation <- row_annotation[order(row_annotation$group1), ] # same task can b
e done by order()
```

Let’s set the colour for annotations -

Hide

```
colour_list <- list(
  group1 = brewer.pal(5, "Set1"),
  group2 = brewer.pal(10, "Set3"),
  sample = c("salmon", "cadetblue")
)
# make sure that the names of the dataframe in this list matches the levels of cor
responding factors, like col_annotation$sample has only two levels (1,2) and those
are numerical.
names(colour_list$group1) <- seq(1:5)
names(colour_list$group2) <- seq(1:10)
names(colour_list$sample) <- c(1,2)
```

Now plot the heatmap -

Hide

```
options(repr.plot.width=1, repr.plot.height=20)
pheatmap(
  cluster_rows = F, # turning this off to avoid the effect of dendogram and sort t
he annotation according to group1 of annotation dataframe
  cluster_cols = T,
  mat = mat[rownames(row_annotation), ],
  color = inferno(n = 10),
  annotation_row = row_annotation,
  annotation_col = col_annotation,
  annotation_colors = colour_list,
  cutree_cols = 2 #cutree_rows won't work as the cluster_rows is set to FALSE here
)
```