RNAseq with edgeR

Jakob Jung

November 17, 2020

Here I perform the downstream analysis of the RNAseq experiment, in which Salmonella was challenged with PNA targeting the acpP gene and different carrier peptides: KFF, RXR and TAT (3 biological replicates of 4 combinations). RNAseq was performed to determine transcriptome changes upon exposure to PNA.

Packages

I import the packages needed for all analysis. They can all be installed from Bioconductor or CRAN if not statet otherwise:

```
library(edgeR)
library(circlize)
library(dplyr)
library(ggplot2)
library('RUVSeq')
library(RColorBrewer)
library(oligo)
library(EDASeq)
library(gplots)
library(ggrepel)
library(svglite)
library(ComplexHeatmap)
library(VennDiagram)
library(eulerr)
library(tidyverse)
library(grid)
```

Data Acquisition

I generated a tab file containing gene counts after upstream processing. Upstream processing included folowing steps (starting with fastq-files):

- BBtools for filtering, trimming and mapping
- featureCounts for generating a count matrix

Firstly, I import the gene-wise counts:

```
GenewiseCounts <- read.delim(
  "../data/rna_align/counttable.txt",sep = "\t",
  row.names = 1, header = T, comment.char = "#")

dim(GenewiseCounts)
```

```
## [1] 4915   35
```

```
head(GenewiseCounts[,1:6])
```

```
##                      Chr Start  End Strand Length
## SL1344_0001 FQ312003.1   169  255      +     87
## SL1344_0002 FQ312003.1   337 2799      +   2463
## SL1344_0003 FQ312003.1  2801 3730      +    930
## SL1344_0004 FQ312003.1  3734 5020      +   1287
## SL1344_0005 FQ312003.1  5114 5887      -    774
## SL1344_0006 FQ312003.1  5966 7396      -   1431
##             ...data.rna_align.ID.006413_S1_R1_001.bam
## SL1344_0001                                        91
## SL1344_0002                                       228
## SL1344_0003                                        83
## SL1344_0004                                       176
## SL1344_0005                                        66
## SL1344_0006                                        60
```

I have to change column names, since they include the whole path:

```
gwc <- GenewiseCounts[,5:length(GenewiseCounts[1,])]
pnapat <- "\\.\\.\\.data\\.rna_align\\..*_(S\\d\\d?)_R1_001\\.bam"
colnames (gwc) <- gsub(pnapat,"\\1", colnames(gwc))
colnames (gwc)
```

```
##  [1] "Length" "S1"     "S2"     "S3"     "S4"     "S5"     "S6"     "S7"
##  [9] "S8"     "S9"     "S10"    "S11"    "S12"    "S13"    "S14"    "S15"
## [17] "S16"    "S17"    "S18"    "S19"    "S20"    "S21"    "S22"    "S23"
## [25] "S24"    "S25"    "S26"    "S27"    "S28"    "S29"    "S30"
```

I also create a facor variable for groups of the sample data manually (from assigning sample codes to condition):

```
test <- rep(c("Water", "KFF_acpP", "KFF_acpP_scrambled", "KFF", "RXR_acpP", "RXR_acpP_scrambled",
         "RXR", "TAT_acpP", "TAT_acpP_scrambled", "TAT"), 3)
test <- as.factor(test)
test
```

```
##  [1] Water              KFF_acpP           KFF_acpP_scrambled KFF
##  [5] RXR_acpP           RXR_acpP_scrambled RXR                TAT_acpP
##  [9] TAT_acpP_scrambled TAT                Water              KFF_acpP
## [13] KFF_acpP_scrambled KFF                RXR_acpP           RXR_acpP_scrambled
## [17] RXR                TAT_acpP           TAT_acpP_scrambled TAT
## [21] Water              KFF_acpP           KFF_acpP_scrambled KFF
## [25] RXR_acpP           RXR_acpP_scrambled RXR                TAT_acpP
## [29] TAT_acpP_scrambled TAT
## 10 Levels: KFF KFF_acpP KFF_acpP_scrambled RXR RXR_acpP ... Water
```

Now that I have the read count dataframe with sample names, I import them into the edgeR environment:

```
y <- DGEList(gwc[,-1], group = test, genes = gwc[,1,drop=FALSE])
options(digits = 3)
head(y$samples)
```

```
##                  group lib.size norm.factors
## S1               Water   883257            1
## S2          KFF_acpP   842644            1
## S3 KFF_acpP_scrambled   862692            1
## S4               KFF   873619            1
## S5          RXR_acpP   772829            1
## S6 RXR_acpP_scrambled   837778            1
```

Filtering

Now I want to filter out Genes which have very low counts across all libraries. I do this by creating a cutoff

$$\frac{10}{L}$$

where L is the minimum library size in millions. We delete genes that are below the cutoff in at least 2 libraries:

```r
L <- min(y$samples$lib.size) / 1000000
cutoff <- 10/L
keep <- rowSums(cpm(y) > cutoff) >= 2
table(keep)
```

```
## keep
## FALSE  TRUE
##   519  4396
```

I retain only the unfiltered genes,and delete 519 genes below the threshold:

```r
y <- y[keep, , keep.lib.sizes=FALSE]
```

Design matrix

I create a design matrix for the samples:

```r
design <- model.matrix(~0+test)
colnames(design) <- levels(test)
rownames(design) <- colnames(y$counts)
design[1:5,]
```

```
##    KFF KFF_acpP KFF_acpP_scrambled RXR RXR_acpP RXR_acpP_scrambled TAT TAT_acpP
## S1   0        0                  0   0        0                  0   0        0
## S2   0        1                  0   0        0                  0   0        0
## S3   0        0                  1   0        0                  0   0        0
## S4   1        0                  0   0        0                  0   0        0
## S5   0        0                  0   0        1                  0   0        0
##    TAT_acpP_scrambled Water
## S1                  0     1
## S2                  0     0
## S3                  0     0
## S4                  0     0
## S5                  0     0
```

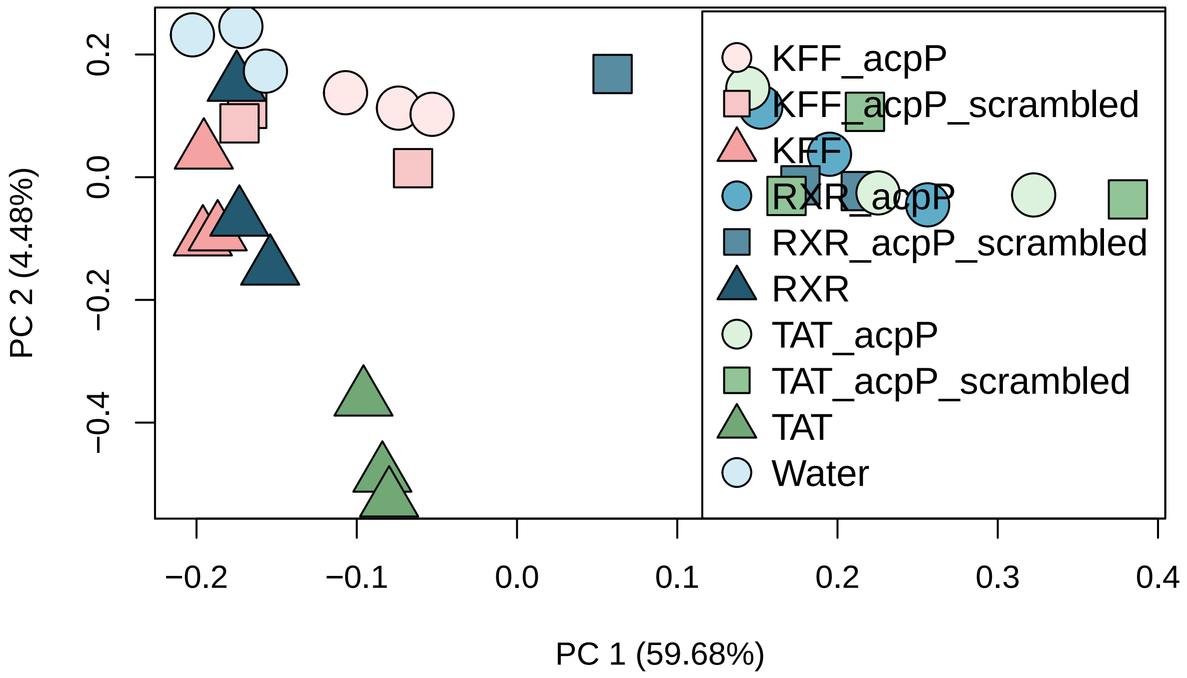
Normalization

I check how the standard TMM normalization of edgeR performs. I start with calculating normalization factors:

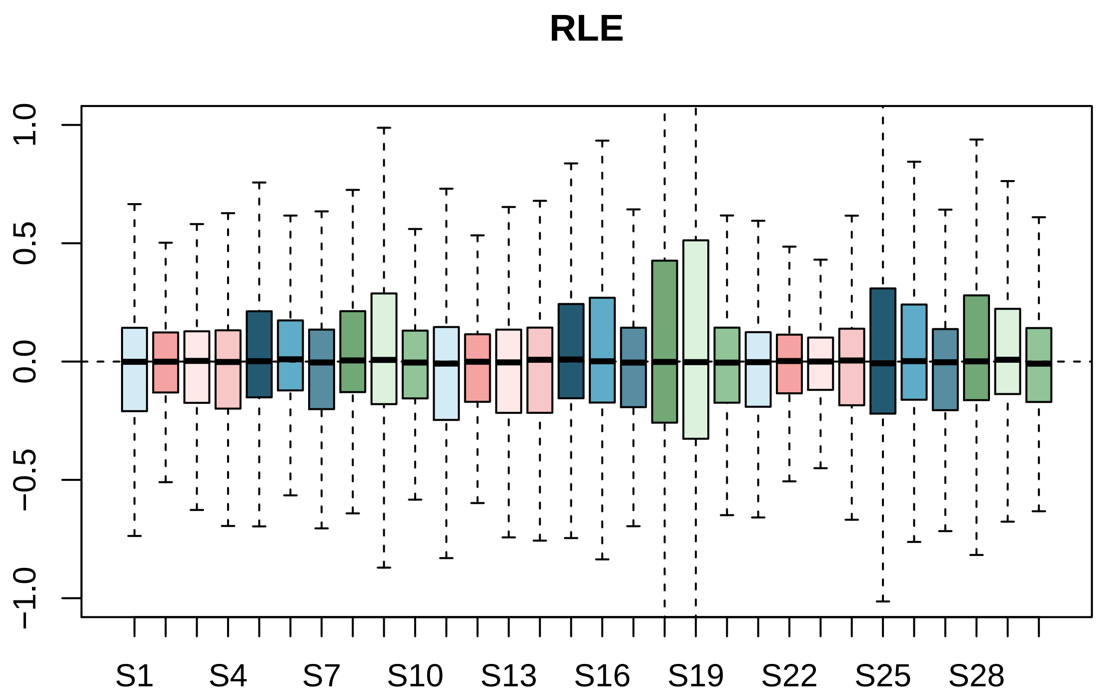
```
y <- calcNormFactors(y)
y <- estimateDisp(y, design, robust = T)
```

And now I create PCA and RLE plots:

```
colors <- c(c("#f5a2a2", "#ffe8e8", "#f7c6c6", "#235a71", "#5facc9",
              "#578ca1", "#71a876", "#dcf2dd", "#91c497", "#d3ebf5"))
pch <- c(24,21,22,24,21,22,24,21,22,21)
lt <- levels(test)[c(2,3,1,5,6,4,8,9,7,10)]
newtest <- factor(test, levels = lt)
newpch <- c(21,22,24,21,22,24,21,22,24,21)
newcols <- colors[c(2,3,1,5,6,4,8,9,7,10)]

plotPCA(cpm(y), col="black", bg=newcols[newtest],  labels=F, pch=newpch[newtest], cex=3)
legend("bottomright", legend = levels(newtest), pch=newpch,
       cex=1.2, col="black",  pt.bg = newcols, pt.cex = 2)
```



```
plotRLE(cpm(y), outline=FALSE, ylim=c(-1, 1), col=colors[as.factor(newtest)],
        main="RLE")
```



You can see that the TMM was succesful (TMM centers the RLE around 0). Also the variability is rather similar for all samples. This can also be seen in the PCA plot, where the samples separating by condition well, and no batch effects are visible.

I save the PCA as SVG:

```
svg("../analysis/PCA_TMM.svg")
plotPCA(cpm(y), col="black", bg=newcols[newtest],  labels=F, pch=newpch[newtest], cex=3)
legend("bottomright", legend = levels(newtest), pch=newpch,
       cex=1.2, col="black",  pt.bg = newcols, pt.cex = 2)
dev.off()
```

```
## pdf
##   2
```

Differential Expression analysis using TMM

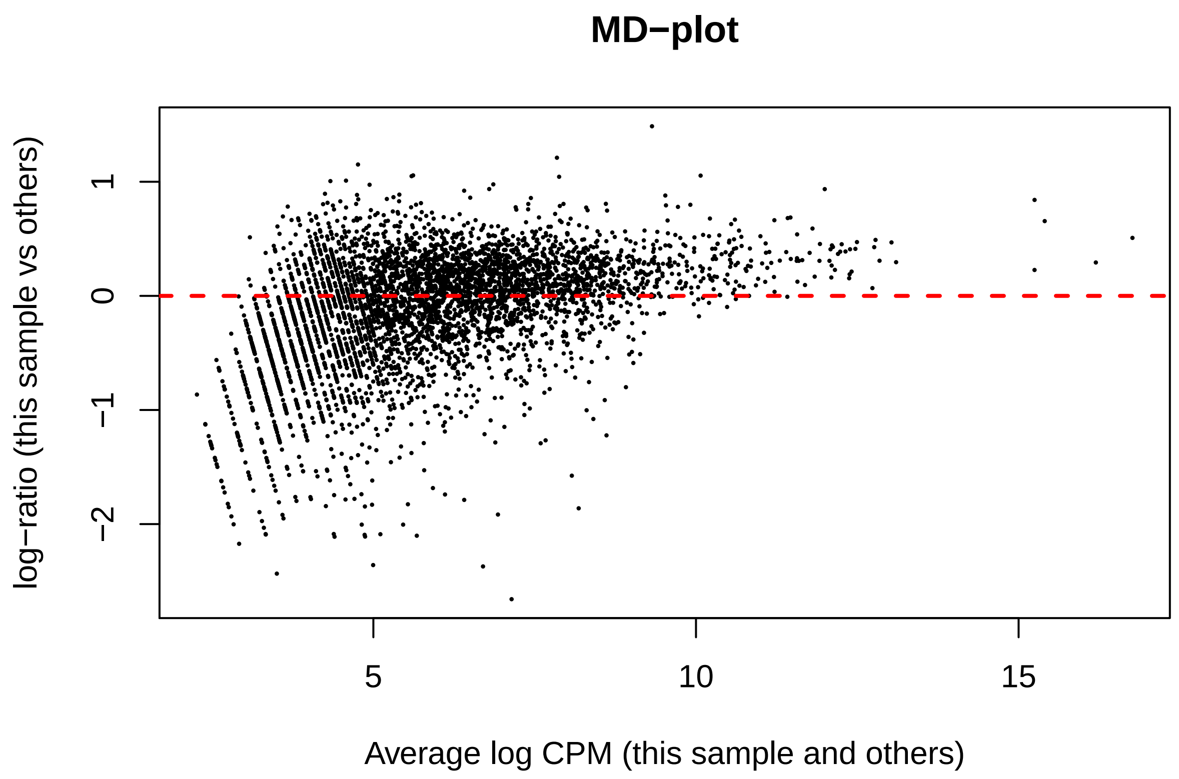
Next, I perform differential expression analysis using TMM-normalized dataset:

```
# make a contrast:
con <- makeContrasts(PNAKFF_vs_ctrl = KFF_acpP - Water,
                     PNAKFFscr_vs_ctrl = KFF_acpP_scrambled - Water,
                     KFF_vs_ctrl = KFF - Water,
                     PNARXR_vs_ctrl = RXR_acpP - Water,
                     PNARXRscr_vs_ctrl = RXR_acpP_scrambled - Water,
                     RXR_vs_ctrl = RXR - Water,
                     PNATAT_vs_ctrl = TAT_acpP - Water,
                     PNATATscr_vs_ctrl = TAT_acpP_scrambled - Water,
                     TAT_vs_ctrl = TAT - Water,
                     levels = design)

fit <- glmQLFit(y, design, robust = TRUE)
```

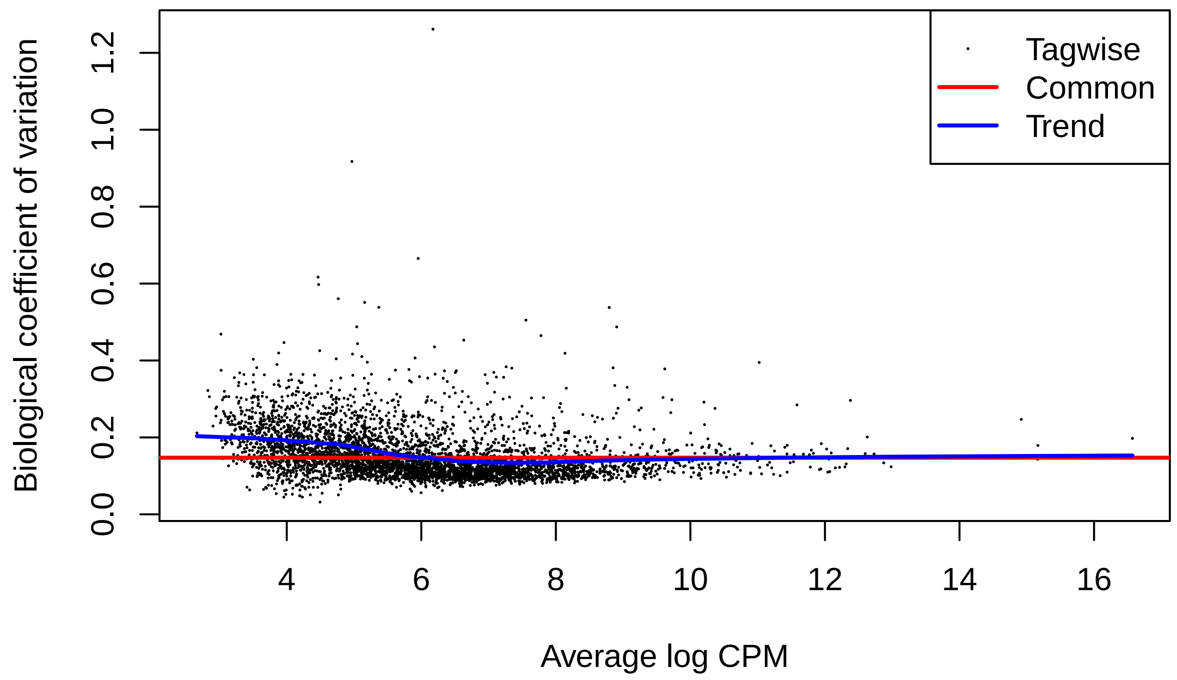
```r
res_KFF <- list(PNAKFF = glmQLFTest(fit, contrast = con[,1]),
            PNAKFFscr = glmQLFTest(fit, contrast = con[,2]),
            KFF = glmQLFTest(fit, contrast = con[,3]))

res_RXR <- list(PNARXR = glmQLFTest(fit, contrast = con[,4]),
            PNARXRscr = glmQLFTest(fit, contrast = con[,5]),
            RXR = glmQLFTest(fit, contrast = con[,6]))

res_TAT <- list(PNATAT = glmQLFTest(fit, contrast = con[,7]),
            PNATATscr = glmQLFTest(fit, contrast = con[,8]),
            TAT = glmQLFTest(fit, contrast = con[,9]))


all_res <- list(KFF = res_KFF,RXR = res_RXR, TAT = res_TAT)
```

We now create MD, BCV and QLDisp plots to access qualiy of data:

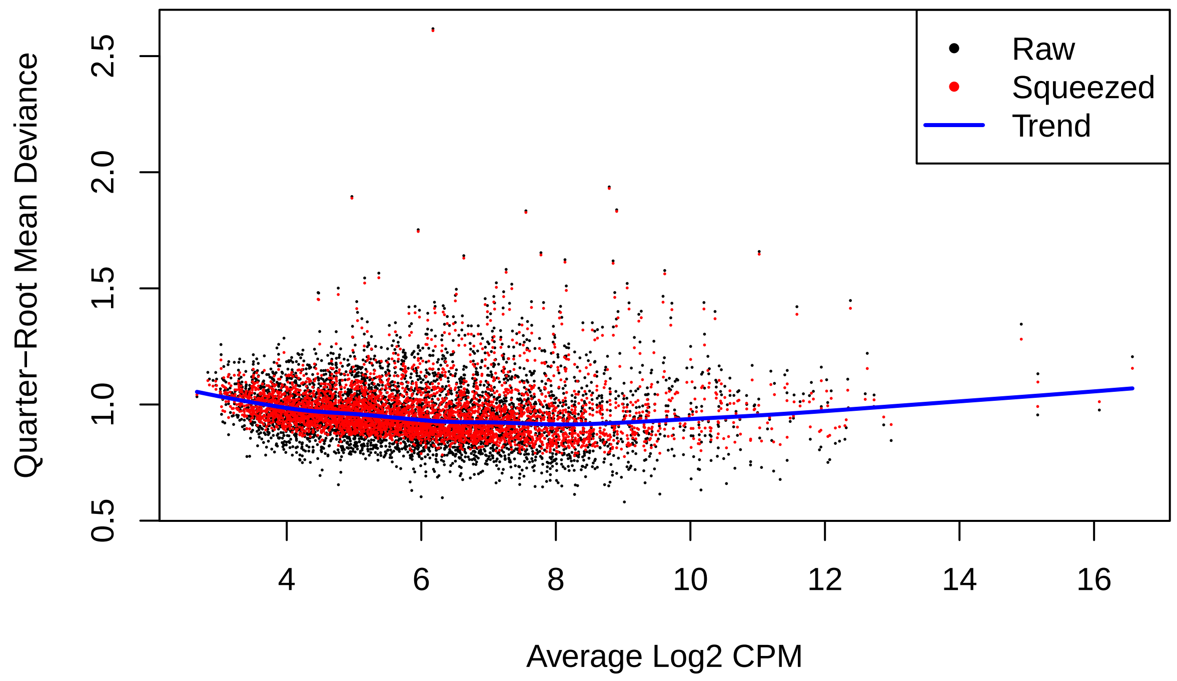
```r
plotMD(y, main = "MD-plot")
abline(h=0, col="red", lty=2, lwd=2)
```



```r
plotBCV(y)
```



```
plotQLDisp(fit)
```



The quality looks decent.

Now I create a function which makes nice volcano-plots and run it on all the results (all PNA-samples are compared to water control for DE):

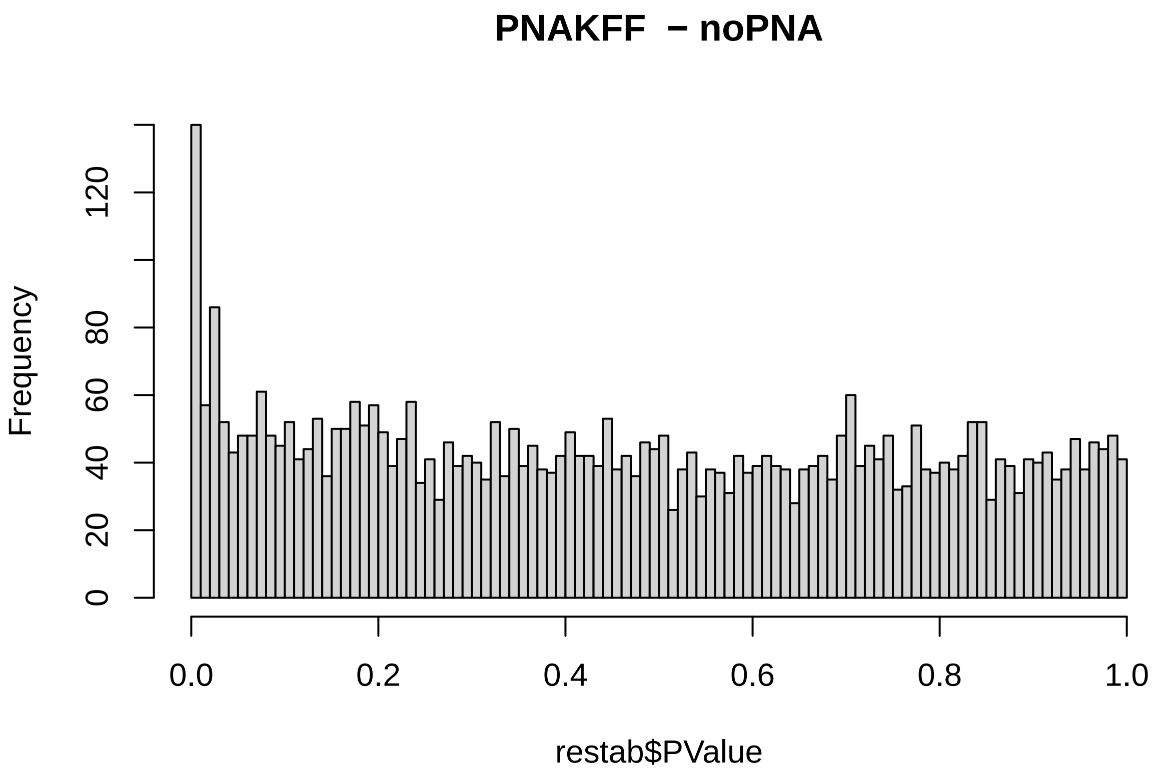
```
do_volcano <- function(restab, pointsize = 2, x_limit = F,y_limit=F, show_sig = F, alpha=0.05,
                       minlogfc=1, title = "Volcano", sRNAS = F, phopq = F) {
  rownames(restab) <- gsub("^([^A-Z].+)" , "italic('\\1')" , rownames(restab))
  g = ggplot(restab) +
  geom_point(
    data = restab,
    aes(x = logFC, y = -log10(FDR)),
    color = "darkgrey",
    cex = pointsize
  ) + theme_bw()+ # change theme to standard black&wite.
  geom_hline(yintercept = -log10(alpha),
             color = "black", linetype = 3) +
  geom_vline(xintercept = c(-minlogfc,minlogfc),
             color = "black", linetype = 3) +
  theme(axis.title.x = element_text(size=20),
        axis.title.y = element_text(size=20),
        axis.text = element_text(size=15, colour = "black"),
        panel.background = element_rect(colour = "black"),
        axis.line = element_line(colour = "black"),
        panel.grid.minor.x = element_blank(),
        panel.grid.minor.y = element_blank(),
        panel.grid.major.x =  element_blank(),#element_line(colour="lightgrey", size=0.3),
        panel.grid.major.y = element_blank(),#element_line(colour="lightgrey", size=0.3),
        plot.title = element_text(hjust = 0.5, size = 23))+
  ggtitle(title)+
  xlab(expression("Log"[2]*" fold change")) +
  ylab("- Log10 P-value (FDR)")+
  scale_x_continuous(expand = c(0,0),breaks = seq(-6,6,2), limits = c(-x_limit,x_limit)) +
  scale_y_continuous(expand = c(0, 0),breaks = seq(0,16,2), limits = c(0,y_limit))


  if (sRNAS == F) {
    g <- g +
      geom_point(
        data = restab[restab$FDR<alpha & restab$logFC < -minlogfc,],
        aes(x = logFC, y = -log10(FDR)),
        color = "blue",
        cex = pointsize) +
      geom_point(
        data = restab[restab$FDR<alpha & restab$logFC > minlogfc,],
        aes(x = logFC, y = -log10(FDR)),
        color = "red",
        cex = pointsize)
  } else{
    show <- restab[sRNAs,][which(restab[sRNAs,]$FDR < alpha),]
    g <- g +
    geom_point(
        data = restab[sRNAs,],
        aes(x = logFC, y = -log10(FDR)),
        color = "darkgreen",
        cex = pointsize) +
    geom_label_repel(
      data = show ,
      aes(x = logFC, y = -log10(FDR),
          label = rownames(show)),
      hjust = 0.1,
      vjust = 2,
      size = 4, segment.alpha = 0.5,min.segment.length=0, segment.color = "black")
```

```r
  }
  g <- g +
      geom_point(
        data = restab[restab$genes %in% c("acpP", "fabF"),],
        aes(x = logFC, y = -log10(FDR)),
        bg = "aquamarine3",
        cex = pointsize+1, pch = 21)
  if (phopq != F) {
    g <- g + geom_point(
        data = restab[restab$genes %in% phopq,],
        aes(x = logFC, y = -log10(FDR)),
        bg = "darkred",
        cex = pointsize+1, pch=21)
  }

  # show the sigficantest genes:
  if(show_sig){
    range01 <- function(x){(x-min(x))/(max(x)-min(x))}
    top_up <- restab[ which(restab$FDR < alpha & restab$logFC > minlogfc),]
    top_down <- restab[ which(restab$FDR < alpha & restab$logFC < -(minlogfc)),]

    if (length(rownames(top_up)) > 0 && (length(rownames(top_up)) > 3)){
    logFC.scaled <- range01(top_up$logFC)
    FDR.scaled <- range01(-log(top_up$FDR))
    summ <- (logFC.scaled + FDR.scaled)
    top_up <- top_up[order(-summ),][1:3,]
    }

    if (length(rownames(top_down))>0 && (length(rownames(top_down))> 3)){
      logFC.scaled <- range01(-top_down$logFC)
      FDR.scaled <- range01(-log(top_down$FDR))
      summ <- (logFC.scaled + FDR.scaled)
      top_down <- top_down[order(-summ),][1:3,]
    }

    top_peaks <- rbind(top_up, top_down)
    top_peaks <- na.omit(top_peaks)

    g <- g + geom_label_repel(
    data = top_peaks ,
    aes(x = logFC, y = -log10(FDR),
        label = rownames(top_peaks)),
    hjust = 0.1,
    vjust = 2,
    size = 5, segment.alpha = 0.5, segment.color = "black", min.segment.length=unit(0, "cm"), parse = T
  }


  g
}
```

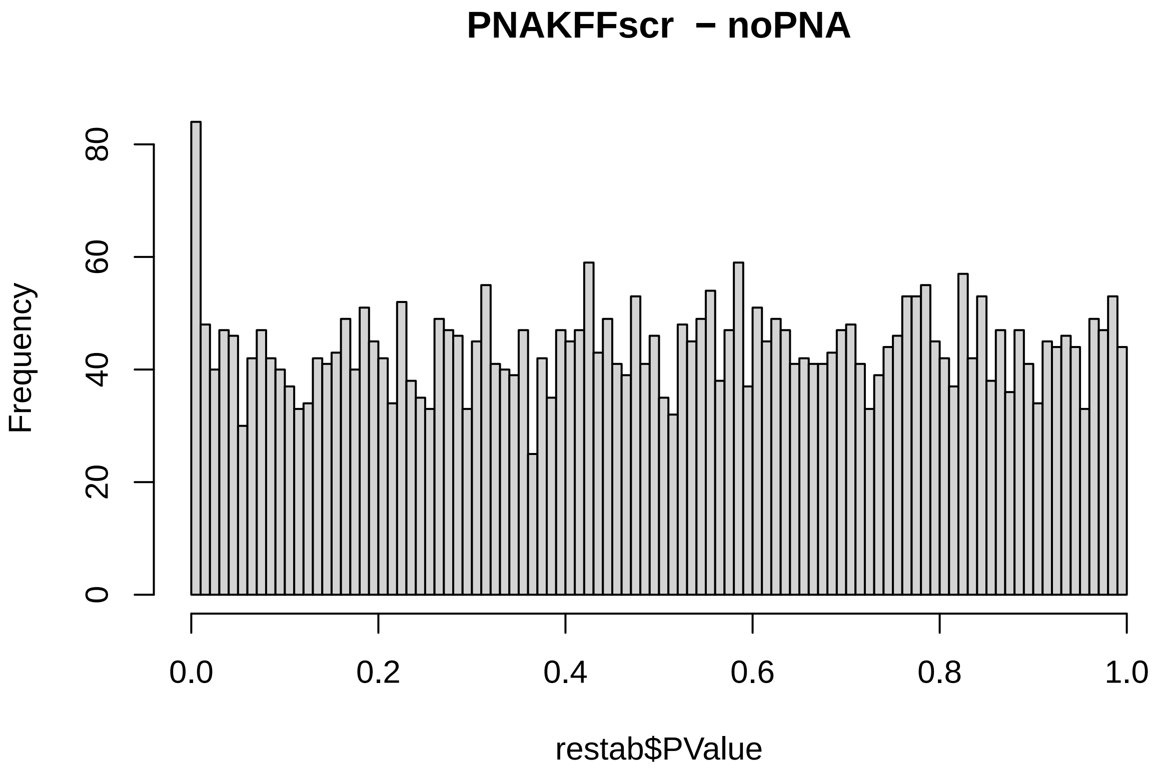
Now I adjust p-values (FDR), create volcano plots, histograms for the results (and save volcano plots as pdfs):

```r
# I create a variable containing strings of all sRNAs:
sRNAs <- c(rownames(res_KFF$PNAKFF$table)[!grepl("SL1344_", rownames(res_KFF$PNAKFF$table))], "cpxP")

# I get the links between locus tags and gene names:
pnames <- read.delim("../data/link_lt_gn.tab", header = F)
rownames(pnames) <- pnames$V2

# I also import PhoPQ related genes:
ppq_raw <- read.delim("../data/PHOPQ.tsv", header = F)
ppq <- as.character(ppq_raw$V1)
phopqvolc <- c(pnames[pnames$V1 %in% ppq,]$V2, "PinT", "SL1344_1169", "SL1344_1168")
prefname <- ifelse(phopqvolc %in% pnames$V2 ,pnames[phopqvolc,]$V1, "" )
prefname <- ifelse(isUnique(prefname), prefname, "")
phopqvolc <- ifelse(prefname != "", prefname, phopqvolc)
phopqvolc <- c(phopqvolc, "phoP", "phoQ")

for (resname in names(all_res)){
  for (name in names(all_res[[resname]])){
    # adjust p-values FDR
  all_res[[resname]][[name]]$table$FDR <- p.adjust(all_res[[resname]][[name]]$table$PValue, method = "fd
  restab <- all_res[[resname]][[name]]$table

  #add genenames (not locustags)
  prefname <- ifelse(rownames(restab) %in% pnames$V2 ,pnames[rownames(restab),]$V1, "" )
  prefname <- ifelse(isUnique(prefname), prefname, "")
  rownames(restab) <- ifelse(prefname != "", prefname, rownames(restab))

  restab$genes <- rownames(restab)

  hist(restab$PValue, breaks=100, main=paste(name," - noPNA"))

  # make volcanos:
  pdf(paste("../analysis/volcanoplots/",name, ".pdf"))
  print(do_volcano(restab, title=paste(name," - noPNA"),
                   x_limit = 7,
                   y_limit = 16,
                   alpha=0.001, pointsize = 3, show_sig = T, phopq = phopqvolc))
  dev.off()
  }
}
```

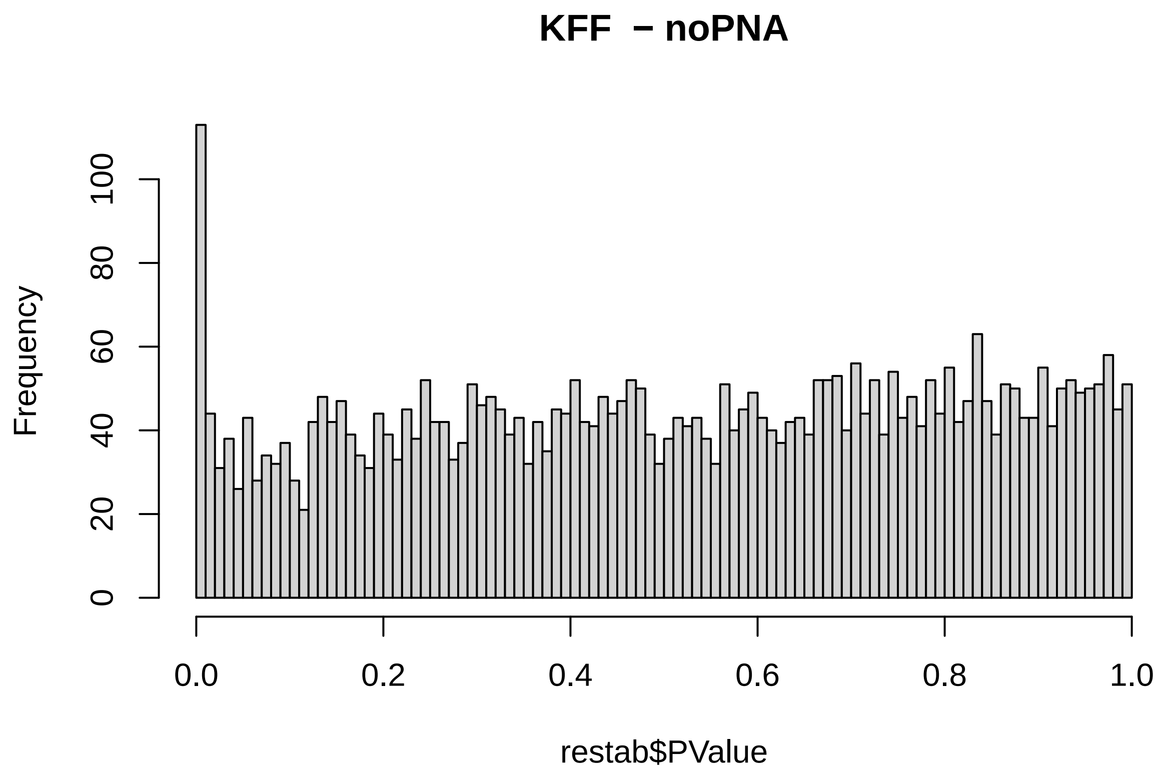
```
## Warning in if (phopq != F) {: the condition has length > 1 and only the first
## element will be used
```



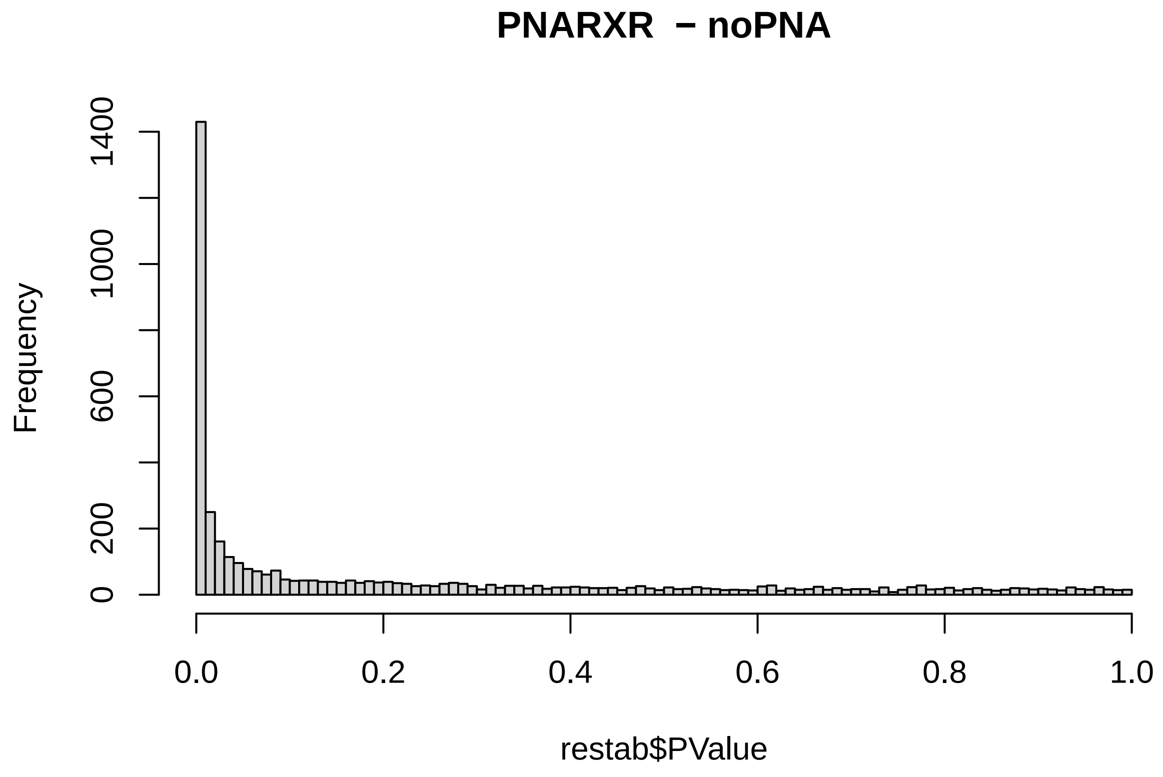
```
## Warning in if (phopq != F) {: the condition has length > 1 and only the first
## element will be used
```



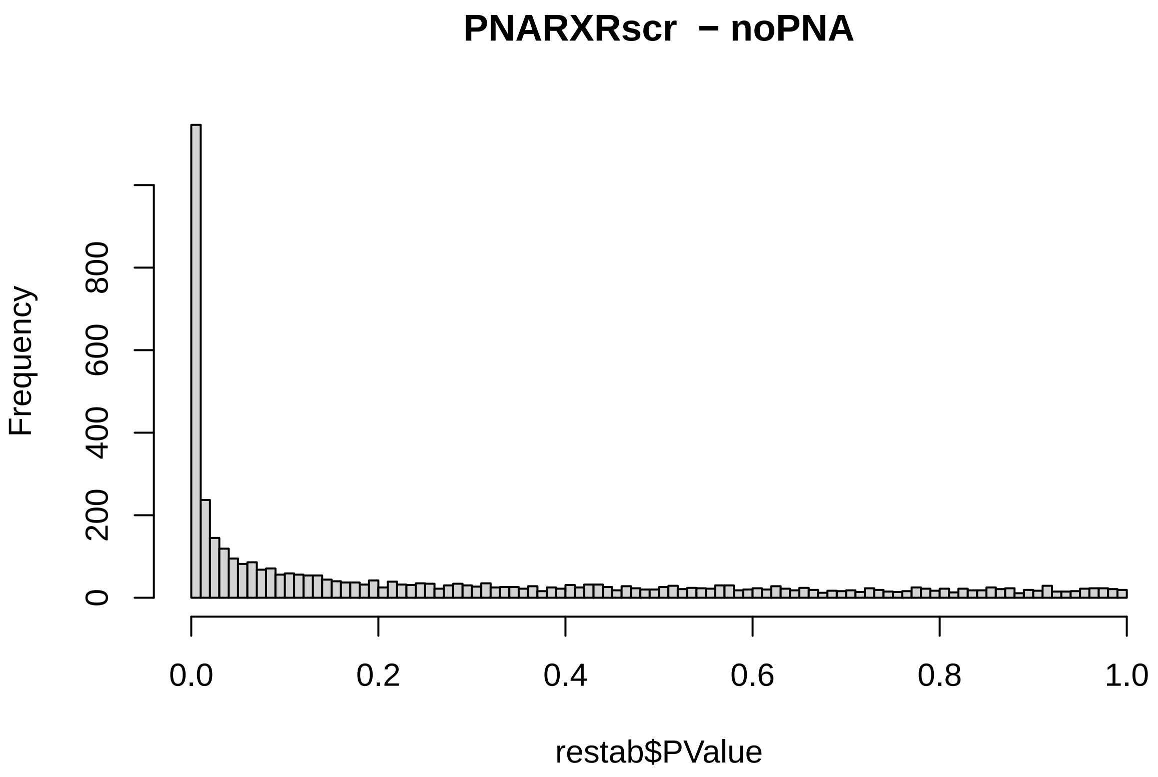
```
## Warning in if (phopq != F) {: the condition has length > 1 and only the first
## element will be used
```



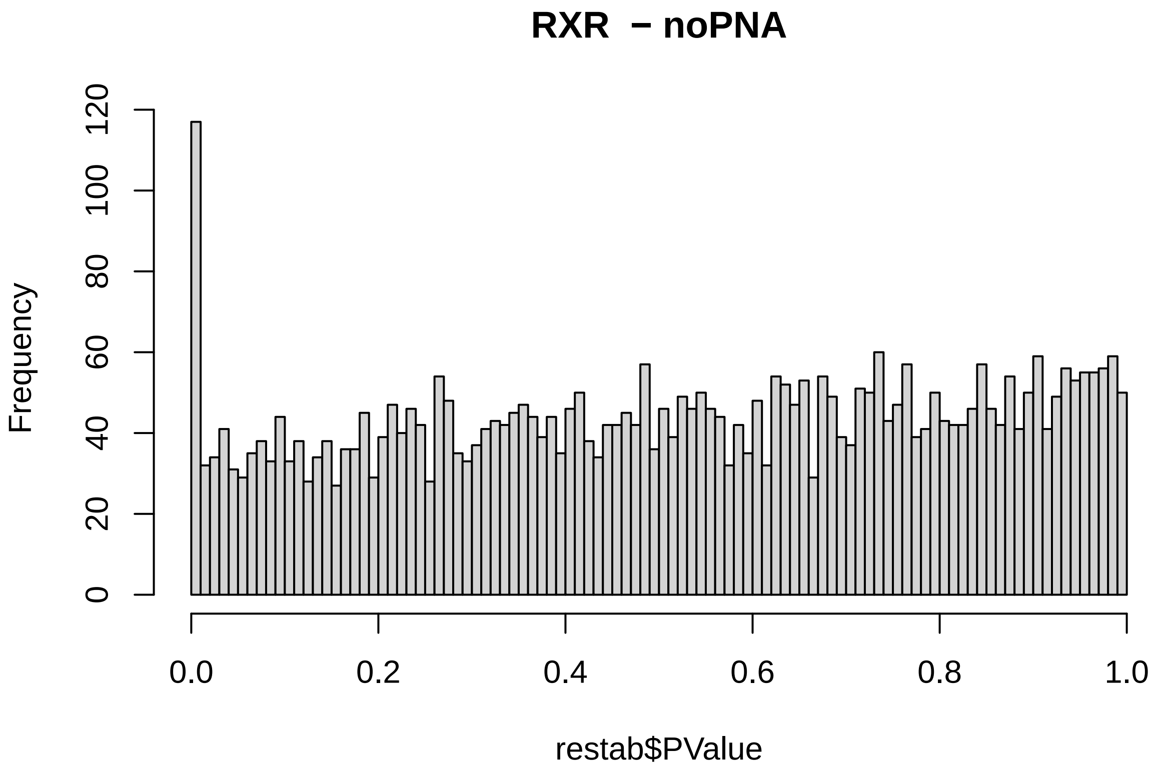
```
## Warning in if (phopq != F) {: the condition has length > 1 and only the first
## element will be used
```



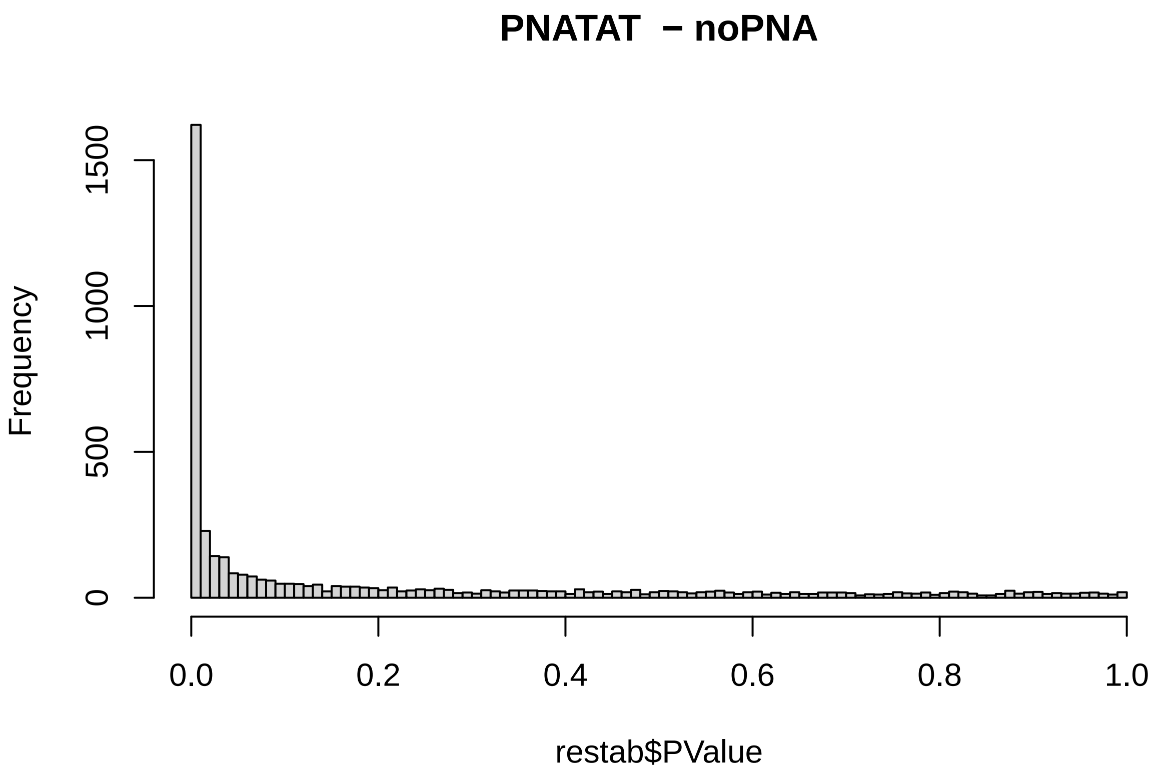
```
## Warning in if (phopq != F) {: the condition has length > 1 and only the first
## element will be used
```



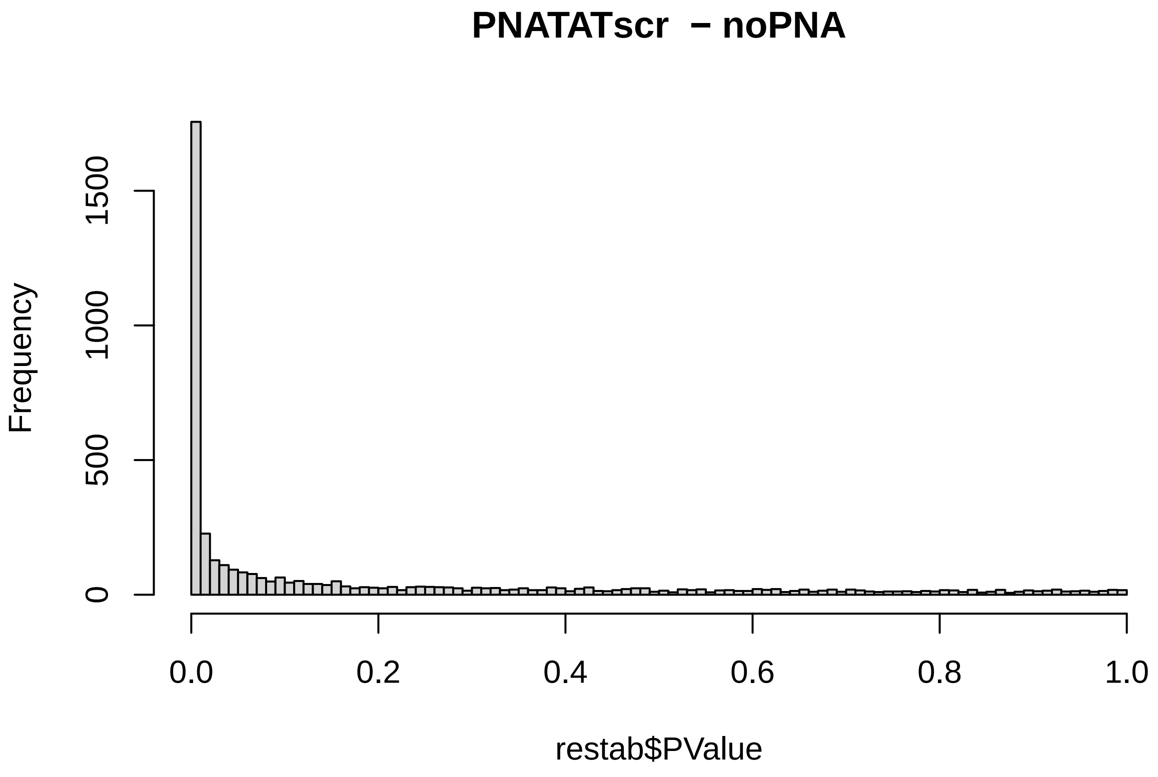
```
## Warning in if (phopq != F) {: the condition has length > 1 and only the first
## element will be used
```



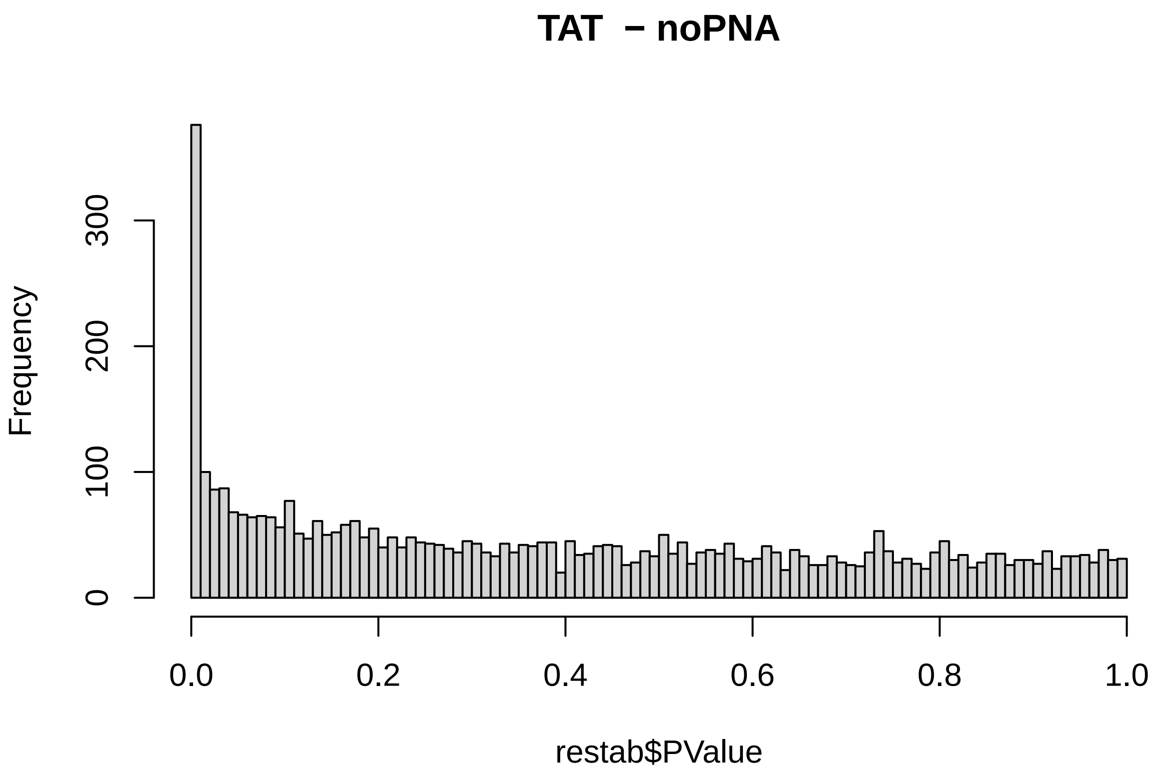
```
## Warning in if (phopq != F) {: the condition has length > 1 and only the first
## element will be used
```



```
## Warning in if (phopq != F) {: the condition has length > 1 and only the first
## element will be used
```



```
## Warning in if (phopq != F) {: the condition has length > 1 and only the first
## element will be used
```



The volcano-plots that are created look fine, and also the p-value distributions for the samples are uniform with an increment in the lowest p-values showing DE genes.

Heatmap

I create a heatmap showing the 20 highest ranking DE genesper conditon. I start with the KFF samples:

KFF

```
for (name in names(res_KFF)) {
  res_KFF[[name]]$table$p_value_FDR <- p.adjust(res_KFF[[name]]$table$PValue, method = "fdr")
  dataname <- paste("../analysis/", name, ".csv", sep = "")
  #write.csv(res_KFF[[name]]$table, dataname)
}

pttk <- res_KFF$PNAKFF$table[order(res_KFF$PNAKFF$table$PValue),]
ptscrk <- res_KFF$PNAKFFscr$table[order(res_KFF$PNAKFFscr$table$PValue),]
ttk <- res_KFF$KFF$table[order(res_KFF$KFF$table$PValue),]

topDEgenes <- c(rownames(pttk[pttk$p_value_FDR<0.001 &
                                           abs(pttk$logFC)>1,])[1:20],
                rownames(ptscrk[ptscrk$p_value_FDR<0.001 &
                                           abs(ptscrk$logFC)>1,])[1:20],
                rownames(ttk[ttk$p_value_FDR<0.001 &
                                           abs(ttk$logFC)>1,])[1:20],
                "SL1344_1133", "SL1344_1134")
topDEgenes <- unique(topDEgenes[!is.na(topDEgenes)])
```

```
logCPM <- cpm(y, prior.count = 2, log = TRUE)
logCPM <- logCPM[rownames(logCPM)%in%topDEgenes,]
logCPM <- t(scale(t(logCPM))) #centered around 0

prefname <- ifelse(rownames(logCPM) %in% pnames$V2 ,pnames[rownames(logCPM),]$V1, "" )
prefname <- ifelse(isUnique(prefname), prefname, "")
rownames(logCPM) <- ifelse(prefname != "", prefname, rownames(logCPM))

KFF_nrs <- c(1,2,3,4,11,12,13,14,21,22,23,24)
logCPM <- logCPM[,KFF_nrs]

colnames(logCPM) <- factor(c("Untreated R1 ", "on-target PPNA R1 ","scrambled PPNA R1 ","peptide alone 
                             "Untreated R2 ", "on-target PPNA R2 ","scrambled PPNA R2 ","peptide alone 
                             "Untreated R3 ", "on-target PPNA R3 ","scrambled PPNA R3 ","peptide alone 

head(logCPM)
```

```
##      Untreated R1  on-target PPNA R1  scrambled PPNA R1  peptide alone R1
## pagN       -1.763            -0.2320             0.6521             0.956
## pagP       -1.031            -0.0307             0.3011             0.736
## ybjG       -1.420            -0.3511            -0.0975             0.944
## acpP        1.014            -1.5200             0.9903             0.957
## fabF        1.022            -1.6243             0.9254             0.870
## phoP       -0.946             0.1085             0.3275             1.012
##      Untreated R2  on-target PPNA R2  scrambled PPNA R2  peptide alone R2
## pagN       -1.832              0.288            -0.4277             0.528
## pagP       -1.277              0.117            -0.0411             1.180
## ybjG       -1.438             -0.278             0.7960             1.431
## acpP        0.983             -1.491             0.8676             0.885
## fabF        0.996             -1.326             1.0927             0.741
## phoP       -0.935             -0.102             0.4655             1.166
##      Untreated R3  on-target PPNA R3  scrambled PPNA R3  peptide alone R3
## pagN       -1.821             0.6691              0.785             0.279
## pagP       -0.988             0.0401              0.449             1.036
## ybjG       -1.081            -0.0171              0.266             1.124
## acpP        0.810            -1.7105              0.733             0.859
## fabF        0.641            -1.3703              0.898             0.762
## phoP       -1.138            -0.0534              0.874             0.958
```

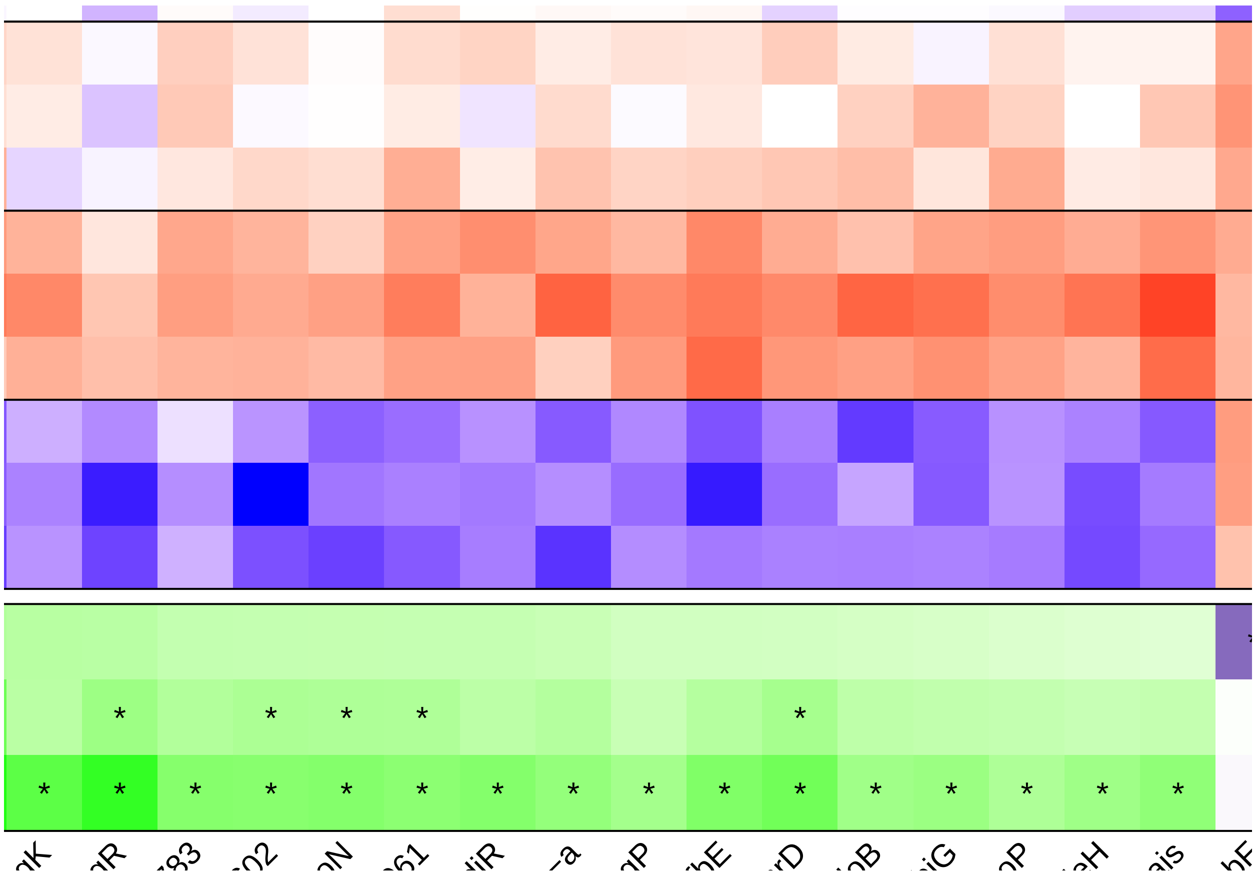
Interestingly, the patterns of the samples with scrambled PNA and the test sample (PNA11 and 10 resp.) show similar patterns of the DE genes and cluster together closely.

```
# get log2change, between those and noPNA:
logchange <- data.frame(PNAKFF = res_KFF$PNAKFF$table$logFC,PNAKFFscr =res_KFF$PNAKFFscr$table$logFC,
                             KFF =res_KFF$KFF$table$logFC, row.names = rownames(res_KFF$PNAKFF$table))

pvals <- data.frame(PNAKFF = (res_KFF$PNAKFF$table$p_value_FDR<0.001 & abs(res_KFF$PNAKFF$table$logFC)>
                        PNAKFFscr=(res_KFF$PNAKFFscr$table$p_value_FDR<0.001 &
                                     abs(res_KFF$PNAKFFscr$table$logFC)>1),
                        KFF =(res_KFF$KFF$table$p_value_FDR<0.001 & abs(res_KFF$KFF$table$logFC)>1),
                        row.names = rownames(res_KFF$PNAKFF$table))
#select only significant ones:
logchange <- logchange[rownames(logchange)%in%topDEgenes,]

#add genenames
```

```r
prefname <- ifelse(rownames(logchange) %in% pnames$V2 ,pnames[rownames(logchange),]$V1, "" )
prefname <- ifelse(isUnique(prefname), prefname, "")
rownames(logchange) <- ifelse(prefname != "", prefname, rownames(logchange))

#select only significant ones:
pvals <- pvals[rownames(pvals)%in%topDEgenes,]
#rownames(pvals) <- x$Prefname
colnames(pvals) <- factor(c("on-target PPNA ", "scrambled PPNA ", "peptide alone "))
pvals <-sapply(pvals, function(x) ifelse(x, x <- "*",x<-"") )


prefname <- ifelse(rownames(pvals) %in% pnames$V2 ,pnames[rownames(pvals),]$V1, "" )
prefname <- ifelse(isUnique(prefname), prefname, "")
rownames(pvals) <- ifelse(prefname != "", prefname, rownames(pvals))


colnames(logchange) <- factor(c("on-target PPNA ", "scrambled PPNA ", "peptide alone "))

pvals <- pvals[order(logchange$`on-target PPNA `, decreasing = T),]
logCPM <-logCPM[order(logchange$`on-target PPNA `, decreasing = T),]
logchange <- logchange[order(logchange$`on-target PPNA `, decreasing = T),]

logCPM_T <- t(logCPM[rownames(logchange),])
logchange_T <- t(logchange)
pvals_T <- t(pvals)
rownames(logchange) <- gsub("(.*)", " \\1",rownames(logchange))
```

Now I save the heatmap as pdf:

```r
ord2 <- c(rep("on-target PPNA", 3), rep("peptide alone", 3), rep("scrambled PPNA", 3),
         rep("untreated", 3))
lev2 <- c("on-target PPNA", "scrambled PPNA", "peptide alone", "untreated")
logCPM_T <- logCPM_T[sort(rownames(logCPM_T)),]

nr_degenes <- dim(logCPM_T)[2]

c1 =  circlize::colorRamp2(c(-2, 0, 2), c("blue", "white", "red"))
c2 = circlize::colorRamp2(c(-2, 0, 2), c("darkblue", "white", "green"))

ht3 <- Heatmap(logCPM_T, cluster_rows = F, name = "Log CPM",
               show_row_names = F,col=c1,
               show_heatmap_legend = F, cluster_columns = F,
               row_title_side = "right", row_title_rot = 0,
               border = TRUE,
               column_names_max_height=max_text_width(colnames(logCPM)),
               row_split = factor(ord2, levels = lev2),
               row_gap = unit(0, "cm"),
               width = unit(nr_degenes, "cm"), height = unit(10, "cm"),
               column_names_rot = 45)
               #rect_gp = gpar(col = "black", lwd = 1))


ht4 <- Heatmap(logchange_T, name = "Log2 FC",
               col = c2,
               cluster_rows = F, cluster_columns = F, show_heatmap_legend = F,
               cell_fun = function(j, i, x, y, width, height, fill) {
                 grid.text(sprintf("%.1s", pvals_T[i, j]), x, y)
               },
               border = TRUE, height = unit(3, "cm"),
               row_names_max_width = max_text_width(c(0,0),gp = gpar(fontsize = 0)),
               column_names_rot = 45)
               #rect_gp = gpar(col = "black", lwd = 1))

ht_list = ht3 %v% ht4
ht_list
```



```
lgd1 = Legend(col_fun = c1, title = expression("Log CPM"), labels_gp = gpar(fontsize = 10),
             title_gp = gpar(fontsize = 15),
             at = c(-2, 0, 2), legend_width = unit(4, "cm"), grid_width =  unit(0.8, "cm"),
             labels = c("-2", "  0", "  2"), legend_height = unit(3, "cm"),
             title_position = "leftcenter-rot")
lgd2 = Legend(col_fun = c2, title = expression("Log"[2]*" FC"), labels_gp = gpar(fontsize = 10),
             title_gp = gpar(fontsize = 15),grid_width =  unit(0.8, "cm"),
             at = c(-2, 0, 2), legend_width = unit(4, "cm"),
             labels = c("-2", "  0", "  2"), legend_height = unit(3, "cm"),
             title_position = "leftcenter-rot")

pdf("../analysis/heatmaps/heatmap_KFF_reduced.pdf", width = 20, height = 10)
draw(ht_list, ht_gap = unit(0.5, "cm"))
draw(lgd1, x = unit(5, "cm"), y = unit(14.35, "cm"), just = c("left", "bottom"))
draw(lgd2, x = unit(5, "cm"), y = unit(7.3, "cm"), just = c("left", "bottom"))
dev.off()
```

```
## pdf
##   2
```

RXR heatmap:

I do the same for RXR (redundant):

```
for (name in names(res_RXR)) {
  res_RXR[[name]]$table$p_value_FDR <- p.adjust(res_RXR[[name]]$table$PValue, method = "fdr")
  dataname <- paste("../analysis/", name, ".csv", sep = "")
  #write.csv(res_RXR[[name]]$table, dataname)
}

pttr <- res_RXR$PNARXR$table[order(res_RXR$PNARXR$table$PValue),]
ptscrr <- res_RXR$PNARXRscr$table[order(res_RXR$PNARXRscr$table$PValue),]
ttr <- res_RXR$RXR$table[order(res_RXR$RXR$table$PValue),]

topDEgenes <- c(rownames(pttr[pttr$p_value_FDR<0.001 &
                                          abs(pttr$logFC)>1,])[1:20],
                rownames(ptscrr[ptscrr$p_value_FDR<0.001 &
                                          abs(ptscrr$logFC)>1,])[1:20],
                rownames(ttr[ttr$p_value_FDR<0.001 &
                                          abs(ttr$logFC)>1,])[1:20],
                "SL1344_1133", "SL1344_1134")
topDEgenes <- unique(topDEgenes[!is.na(topDEgenes)])


logCPM <- cpm(y, prior.count = 2, log = TRUE)
logCPM <- logCPM[rownames(logCPM)%in%topDEgenes,]
logCPM <- t(scale(t(logCPM))) #centered around 0

#prefname <- ifelse(rownames(logCPM) %in% rownames(GO),GO[rownames(logCPM),]$Preferred_name, "" )
#rownames(logCPM) <- ifelse(prefname != "", prefname, rownames(logCPM))

RXR_nrs <- c(1,5,6,7,11,15,16,17,21,25,26,27)
logCPM <- logCPM[,RXR_nrs]

colnames(logCPM) <- factor(c("Untreated R1 ", "on-target PPNA R1 ","scrambled PPNA R1 ","peptide alone 
                             "Untreated R2 ", "on-target PPNA R2 ","scrambled PPNA R2 ","peptide alone 
                             "Untreated R3 ", "on-target PPNA R3 ","scrambled PPNA R3 ","peptide alone 

head(logCPM)
```

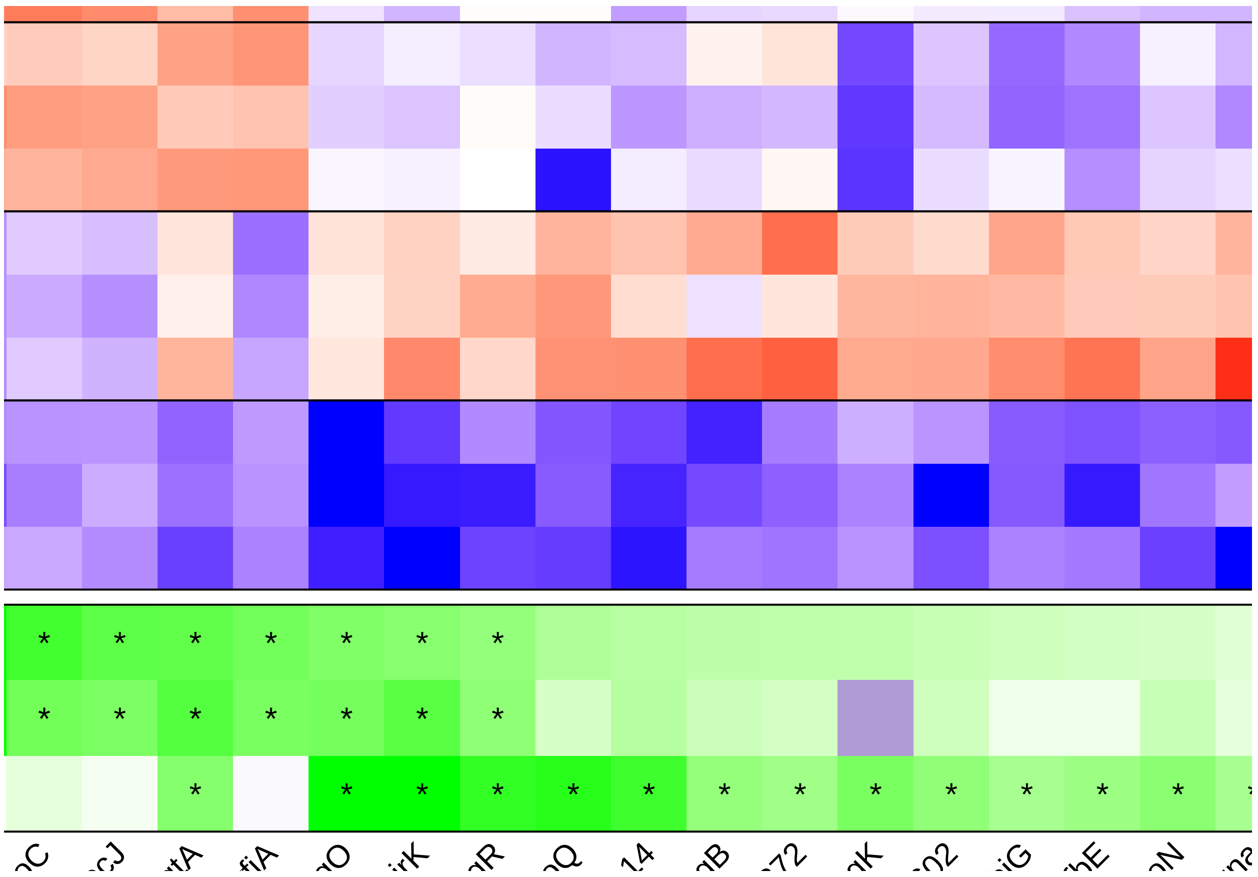
```
##               Untreated R1  on-target PPNA R1  scrambled PPNA R1
## SL1344_0066A        -1.659             1.1403             0.9549
## SL1344_0187          0.966             0.0513            -1.5362
## SL1344_0605         -1.438            -0.6104            -0.6286
## SL1344_0616         -1.031            -0.8303            -0.0336
## SL1344_0745         -0.888             0.9682             0.6395
## SL1344_0767         -1.208             0.8918             0.6903
##               peptide alone R1  Untreated R2  on-target PPNA R2
## SL1344_0066A            -0.706        -1.651              0.849
## SL1344_0187              0.403        0.900             -0.399
## SL1344_0605              0.774       -0.843             -0.174
## SL1344_0616              0.654       -1.277             -0.246
## SL1344_0745             -0.398       -1.098              0.942
## SL1344_0767             -0.886       -1.389              1.191
##               scrambled PPNA R2  peptide alone R2  Untreated R3
## SL1344_0066A               1.12            -0.997        -1.015
## SL1344_0187               -2.32            -0.183         0.339
## SL1344_0605               -1.03             0.615        -2.129
## SL1344_0616               -0.45             0.709        -0.988
## SL1344_0745                1.10            -0.553        -0.724
## SL1344_0767                1.33            -0.642        -0.932
##               on-target PPNA R3  scrambled PPNA R3  peptide alone R3
## SL1344_0066A              1.267              0.883            -1.343
## SL1344_0187              -0.991             -2.435             0.644
## SL1344_0605              -0.636             -0.283             1.882
## SL1344_0616              -0.233             -0.142             1.050
## SL1344_0745               1.390              0.936            -0.498
## SL1344_0767               1.387              1.144            -0.732
```

```r
library(ComplexHeatmap)
# get log2change, between those and noPNA:
logchange <- data.frame(PNARXR = res_RXR$PNARXR$table$logFC,PNARXRscr =res_RXR$PNARXRscr$table$logFC,
                              RXR =res_RXR$RXR$table$logFC, row.names = rownames(res_RXR$PNARXR$table))

pvals <- data.frame(PNARXR = (res_RXR$PNARXR$table$p_value_FDR<0.001 & abs(res_RXR$PNARXR$table$logFC)>
                        PNARXRscr=(res_RXR$PNARXRscr$table$p_value_FDR<0.001 & abs(res_RXR$PNARXRscr$ta
                        RXR =(res_RXR$RXR$table$p_value_FDR<0.001 & abs(res_RXR$RXR$table$logFC)>1),
                        row.names = rownames(res_RXR$PNARXR$table))
#select only significant ones:
logchange <- logchange[rownames(logchange)%in%topDEgenes,]

colnames(logchange) <- factor(c("RXR-PNA ", "RXR-PNA-scr ", "RXR "))

pnames <- read.delim("../data/link_lt_gn.tab", header = F)
rownames(pnames) <- pnames$V2

prefname <- ifelse(rownames(logchange) %in% pnames$V2 ,pnames[rownames(logchange),]$V1, "" )
prefname <- ifelse(isUnique(prefname), prefname, "")
rownames(logchange) <- ifelse(prefname != "", prefname, rownames(logchange))



#select only significant ones:
pvals <- pvals[rownames(pvals)%in%topDEgenes,]
#rownames(pvals) <- x$Prefname
colnames(pvals) <- factor(c("on-target PPNA ", "scrambled PPNA ", "peptide alone "))
pvals <-sapply(pvals, function(x) ifelse(x, x <- "*",x<-"") )

#prefname <- ifelse(rownames(pvals) %in% rownames(GO),GO[rownames(pvals),]$Preferred_name, "" )
#rownames(pvals) <- ifelse(prefname != "", prefname, rownames(pvals))


colnames(logchange) <- factor(c("on-target PPNA ", "scrambled PPNA ", "peptide alone "))
```

```
pvals <- pvals[order(logchange$`on-target PPNA `, decreasing = T),]
logCPM <-logCPM[order(logchange$`on-target PPNA `, decreasing = T),]
logchange <- logchange[order(logchange$`on-target PPNA `, decreasing = T),]

logCPM_T <- t(logCPM)
logchange_T <- t(logchange)
pvals_T <- t(pvals)

rownames(logchange) <- gsub("(.*)", " \\1",rownames(logchange))
```

Now I save the heatmap as pdf:

```
ord2 <- c(rep("on-target PPNA", 3), rep("peptide alone", 3), rep("scrambled PPNA", 3),
         rep("untreated", 3))
lev2 <- c("on-target PPNA", "scrambled PPNA", "peptide alone", "untreated")
logCPM_T <- logCPM_T[sort(rownames(logCPM_T)),]

nr_degenes <- dim(logCPM_T)[2]


ht3 <- Heatmap(logCPM_T, cluster_rows = F, name = "Log CPM",
               show_row_names = F,col=c1,
               show_heatmap_legend = F, cluster_columns = F,
               row_title_side = "right", row_title_rot = 0,
               border = TRUE,
               column_names_max_height=max_text_width(colnames(logCPM)),
               row_split = factor(ord2, levels = lev2),
               row_gap = unit(0, "cm"),
               width = unit(nr_degenes, "cm"), height = unit(10, "cm"),
               column_names_rot = 45)
               #rect_gp = gpar(col = "black", lwd = 1))


ht4 <- Heatmap(logchange_T, name = "Log2 FC",
               col = circlize::colorRamp2(c(-2, 0, 2), c("darkblue", "white", "green")),
               cluster_rows = F, cluster_columns = F, show_heatmap_legend = F,
               cell_fun = function(j, i, x, y, width, height, fill) {
                 grid.text(sprintf("%.1s", pvals_T[i, j]), x, y)
               },
               border = TRUE, height = unit(3, "cm"),
               row_names_max_width = max_text_width(c(0,0),gp = gpar(fontsize = 0)),
               column_names_rot = 45)
               #rect_gp = gpar(col = "black", lwd = 1))

ht_list = ht3 %v% ht4
ht_list
```



```r
pdf("../analysis/heatmaps/heatmap_RXR_reduced.pdf", width = 28, height = 10)
draw(ht_list, ht_gap = unit(0.5, "cm"))
draw(lgd1, x = unit(5, "cm"), y = unit(14.3, "cm"), just = c("left", "bottom"))
draw(lgd2, x = unit(5, "cm"), y = unit(7.3, "cm"), just = c("left", "bottom"))
dev.off()
```

```
## pdf
##   2
```

TAT:

i do the same for TAT

```r
for (name in names(res_TAT)) {
  res_TAT[[name]]$table$p_value_FDR <- p.adjust(res_TAT[[name]]$table$PValue, method = "fdr")
  dataname <- paste("../analysis/", name, ".csv", sep = "")
  #write.csv(res_TAT[[name]]$table, dataname)
}


pttt <- res_TAT$PNATAT$table[order(res_TAT$PNATAT$table$PValue),]
ptscrt <- res_TAT$PNATATscr$table[order(res_TAT$PNATATscr$table$PValue),]
ttt <- res_TAT$TAT$table[order(res_TAT$TAT$table$PValue),]


topDEgenes <- c(rownames(pttt[pttt$p_value_FDR<0.001 & abs(pttt$logFC)>1,])[1:20],
                rownames(ptscrt[ptscrt$p_value_FDR<0.001 &
                                           abs(ptscrt$logFC)>1,])[1:20],
                rownames(ttt[ttt$p_value_FDR<0.001 &
                                           abs(ttt$logFC)>1,])[1:20],
                "SL1344_1133", "SL1344_1134")
topDEgenes <- unique(topDEgenes[!is.na(topDEgenes)])



logCPM <- cpm(y, prior.count = 2, log = TRUE)
logCPM <- logCPM[rownames(logCPM)%in%topDEgenes,]
logCPM <- t(scale(t(logCPM))) #centered around 0
logCPM <- logCPM[topDEgenes,]


TAT_nrs <- c(1,8,9,10,11,18,19,20,21,28,29,30)
logCPM <- logCPM[,TAT_nrs]

colnames(logCPM) <- factor(c("Untreated R1 ", "on-target PPNA R1 ","scrambled PPNA R1 ","peptide alone 
                             "Untreated R2 ", "on-target PPNA R2 ","scrambled PPNA R2 ","peptide alone 
                             "Untreated R3 ", "on-target PPNA R3 ","scrambled PPNA R3 ","peptide alone 

head(logCPM)
```

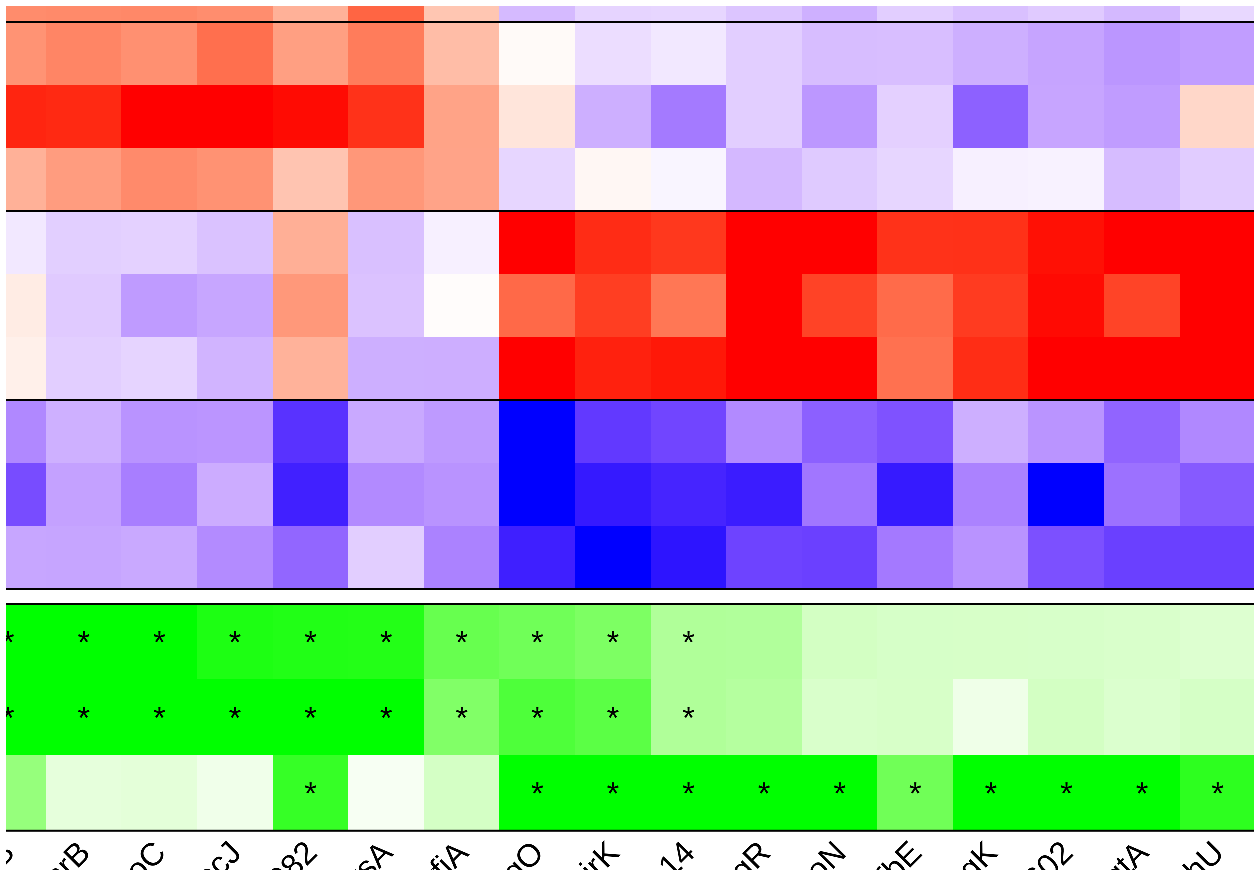
```
##              Untreated R1  on-target PPNA R1  scrambled PPNA R1
## SL1344_2742         -1.329              0.958              1.298
## SL1344_0066A        -1.659              0.859              0.995
## SL1344_1133          1.014             -0.690              0.165
## SL1344_3018         -0.913              0.967              1.434
## SL1344_1339         -1.079              1.121              1.296
## SL1344_4495         -0.852              1.016              1.246
##              peptide alone R1  Untreated R2  on-target PPNA R2
## SL1344_2742              -0.234        -1.391              1.414
## SL1344_0066A              0.116        -1.651              1.281
## SL1344_1133               0.749         0.983             -0.994
## SL1344_3018              -0.522        -0.711              1.660
## SL1344_1339              -0.599        -1.056              1.755
## SL1344_4495              -0.510        -1.051              1.609
##              scrambled PPNA R2  peptide alone R2  Untreated R3
## SL1344_2742                1.640           -0.4627        -0.746
## SL1344_0066A               1.315            0.0151        -1.015
## SL1344_1133               -0.296            0.4357         0.810
## SL1344_3018                2.103           -0.7667        -0.998
## SL1344_1339                1.889           -0.4921        -0.897
## SL1344_4495                1.813           -0.6822        -0.963
##              on-target PPNA R3  scrambled PPNA R3  peptide alone R3
## SL1344_2742                1.219              1.123           -0.4589
## SL1344_0066A               1.085              1.015            0.0187
## SL1344_1133               -0.958              0.326            0.7155
## SL1344_3018                1.163              1.118           -0.6462
## SL1344_1339                1.277              0.993           -0.6521
## SL1344_4495                1.218              0.967           -0.5328
```

```
library(ComplexHeatmap)
# get log2change, between those and noPNA:
```

```r
logchange <- data.frame(PNATAT = res_TAT$PNATAT$table$logFC,PNATATscr =res_TAT$PNATATscr$table$logFC,
                              TAT =res_TAT$TAT$table$logFC, row.names = rownames(res_TAT$PNATAT$table))

pvals <- data.frame(PNATAT = (res_TAT$PNATAT$table$p_value_FDR<0.001 & abs(res_TAT$PNATAT$table$logFC)>
                        PNATATscr=(res_TAT$PNATATscr$table$p_value_FDR<0.001 & abs(res_TAT$PNATATscr$ta
                        TAT =(res_TAT$TAT$table$p_value_FDR<0.001 & abs(res_TAT$TAT$table$logFC)>1),
                        row.names = rownames(res_TAT$PNATAT$table))
#select only significant ones:
logchange <- logchange[rownames(logchange)%in%topDEgenes,]
logchange <- logchange[topDEgenes,]

colnames(logchange) <- factor(c("TAT-PNA ", "TAT-PNA-scr ", "TAT "))


prefname <- ifelse(rownames(logchange) %in% pnames$V2 ,pnames[rownames(logchange),]$V1, "" )
prefname <- ifelse(isUnique(prefname), prefname, "")
rownames(logchange) <- ifelse(prefname != "", prefname, rownames(logchange))





#select only significant ones:
pvals <- pvals[rownames(pvals)%in%topDEgenes,]
pvals <- pvals[topDEgenes,]

#rownames(pvals) <- x$Prefname
colnames(pvals) <- factor(c("on-target PPNA ", "scrambled PPNA ", "peptide alone "))
pvals <-sapply(pvals, function(x) ifelse(x, x <- "*",x<-"") )




colnames(logchange) <- factor(c("on-target PPNA ", "scrambled PPNA ", "peptide alone "))

pvals <- pvals[order(logchange$`on-target PPNA `, decreasing = T),]
logCPM <-logCPM[order(logchange$`on-target PPNA `, decreasing = T),]
logchange <- logchange[order(logchange$`on-target PPNA `, decreasing = T),]

logCPM_T <- t(logCPM)
logchange_T <- t(logchange)
pvals_T <- t(pvals)

rownames(logchange) <- gsub("(.*)", " \\1",rownames(logchange))
```

Now I save the heatmap as pdf:

```r
ord2 <- c(rep("on-target PPNA", 3), rep("peptide alone", 3), rep("scrambled PPNA", 3),
         rep("untreated", 3))
lev2 <- c("on-target PPNA", "scrambled PPNA", "peptide alone", "untreated")
logCPM_T <- logCPM_T[sort(rownames(logCPM_T)),]

nr_degenes <- dim(logCPM_T)[2]

ht3 <- Heatmap(logCPM_T, cluster_rows = F, name = "Log CPM",
               show_row_names = F, col=c1,
               show_heatmap_legend = F, cluster_columns = F,
               row_title_side = "right", row_title_rot = 0,
               border = TRUE,
               column_names_max_height=max_text_width(colnames(logCPM)),
               row_split = factor(ord2, levels = lev2),
               row_gap = unit(0, "cm"),
               width = unit(nr_degenes, "cm"), height = unit(10, "cm"),
               column_names_rot = 45)
               #rect_gp = gpar(col = "black", lwd = 1))


ht4 <- Heatmap(logchange_T, name = "Log2 FC",
               col = c2,
               cluster_rows = F, cluster_columns = F, show_heatmap_legend = F,
               cell_fun = function(j, i, x, y, width, height, fill) {
                 grid.text(sprintf("%.1s", pvals_T[i, j]), x, y)
               },
               border = TRUE, height = unit(3, "cm"),
               row_names_max_width = max_text_width(c(0,0),gp = gpar(fontsize = 0)),
               column_names_rot = 45)
               #rect_gp = gpar(col = "black", lwd = 1))

ht_list = ht3 %v% ht4
ht_list
```



```
pdf("../analysis/heatmaps/heatmap_TAT_reduced.pdf", width = 28, height = 10)
draw(ht_list, ht_gap = unit(0.5, "cm"))
draw(lgd1, x = unit(5, "cm"), y = unit(14.3, "cm"), just = c("left", "bottom"))
draw(lgd2, x = unit(5, "cm"), y = unit(7.3, "cm"), just = c("left", "bottom"))
dev.off()
```

```
## pdf
##   2
```

sRNAs heatmap:

I also create a heatmap for DE sRNAs only to find enriched/reduced sRNAs:

```
topDE <- c(rownames(res_TAT$PNATAT$table[res_TAT$PNATAT$table$p_value_FDR<0.001 &
                                         abs(res_TAT$PNATAT$table$logFC)>1,]),
               rownames(res_TAT$PNATATscr$table[res_TAT$PNATATscr$table$p_value_FDR<0.001 &
                                         abs(res_TAT$PNATATscr$table$logFC)>1,]),
               rownames(res_TAT$TAT$table[res_TAT$TAT$table$p_value_FDR<0.001 &
                                         abs(res_TAT$TAT$table$logFC)>1,]),
               rownames(res_RXR$PNARXR$table[res_RXR$PNARXR$table$p_value_FDR<0.001 &
                                         abs(res_RXR$PNARXR$table$logFC)>1,]),
               rownames(res_RXR$PNARXRscr$table[res_RXR$PNARXRscr$table$p_value_FDR<0.001 &
                                         abs(res_RXR$PNARXRscr$table$logFC)>1,]),
               rownames(res_RXR$RXR$table[res_RXR$RXR$table$p_value_FDR<0.001 &
                                         abs(res_RXR$RXR$table$logFC)>1,]),
               rownames(res_KFF$PNAKFF$table[res_KFF$PNAKFF$table$p_value_FDR<0.001 &
                                         abs(res_KFF$PNAKFF$table$logFC)>1,]),
               rownames(res_KFF$PNAKFFscr$table[res_KFF$PNAKFFscr$table$p_value_FDR<0.001 &
                                         abs(res_KFF$PNAKFFscr$table$logFC)>1,]),
               rownames(res_KFF$KFF$table[res_KFF$KFF$table$p_value_FDR<0.001 &
                                         abs(res_KFF$KFF$table$logFC)>1,]))
topDE_srnas <- unique(topDE[topDE %in% sRNAs])

logchange_TAT <- data.frame(PNATAT = res_TAT$PNATAT$table$logFC,PNATATscr =res_TAT$PNATATscr$table$logF
                            TAT =res_TAT$TAT$table$logFC, row.names = rownames(res_TAT$PNATAT$table))

pvals_TAT <- data.frame(PNATAT = (res_TAT$PNATAT$table$p_value_FDR<0.001 & abs(res_TAT$PNATAT$table$log
                        PNATATscr=(res_TAT$PNATATscr$table$p_value_FDR<0.001 & abs(res_TAT$PNATATscr$ta
                        TAT =(res_TAT$TAT$table$p_value_FDR<0.001 & abs(res_TAT$TAT$table$logFC)>1),
                        row.names = rownames(res_TAT$PNATAT$table))
#select only significant ones:
logchange_TAT <- logchange_TAT[rownames(logchange_TAT)%in%topDE_srnas,]
logchange_TAT <- logchange_TAT[rownames(logchange_TAT),]
colnames(logchange_TAT) <- factor(c("TAT-acpP ", "TAT-acpP-scrambled ", "TAT "))

prefname <- ifelse(rownames(logchange_TAT) %in% pnames$V2 ,pnames[rownames(logchange_TAT),]$V1, "" )
prefname <- ifelse(isUnique(prefname), prefname, "")
rownames(logchange_TAT) <- ifelse(prefname != "", prefname, rownames(logchange_TAT))


#select only significant ones:
pvals_TAT <- pvals_TAT[rownames(pvals_TAT)%in%topDE_srnas,]
pvals_TAT <- pvals_TAT[rownames(pvals_TAT),]
pvals_TAT <-sapply(pvals_TAT, function(x) ifelse(x, x <- "*",x<-"") )
```

```
rownames(pvals_TAT) <- rownames(logchange_TAT)




##RXR
logchange_RXR <- data.frame(PNARXR = res_RXR$PNARXR$table$logFC,PNARXRscr =res_RXR$PNARXRscr$table$logF
                              RXR =res_RXR$RXR$table$logFC, row.names = rownames(res_RXR$PNARXR$table))

pvals_RXR <- data.frame(PNARXR = (res_RXR$PNARXR$table$p_value_FDR<0.001 & abs(res_RXR$PNARXR$table$log
                        PNARXRscr=(res_RXR$PNARXRscr$table$p_value_FDR<0.001 & abs(res_RXR$PNARXRscr$ta
                        RXR =(res_RXR$RXR$table$p_value_FDR<0.001 & abs(res_RXR$RXR$table$logFC)>1),
                        row.names = rownames(res_RXR$PNARXR$table))
#select only significant ones:
logchange_RXR <- logchange_RXR[rownames(logchange_RXR)%in%topDE_srnas,]
logchange_RXR <- logchange_RXR[rownames(logchange_RXR),]
colnames(logchange_RXR) <- factor(c("RXR-acpP ", "RXR-acpP-scrambled ", "RXR "))

prefname <- ifelse(rownames(logchange_RXR) %in% pnames$V2 ,pnames[rownames(logchange_RXR),]$V1, "" )
prefname <- ifelse(isUnique(prefname), prefname, "")
rownames(logchange_RXR) <- ifelse(prefname != "", prefname, rownames(logchange_RXR))


#select only significant ones:
pvals_RXR <- pvals_RXR[rownames(pvals_RXR)%in%topDE_srnas,]
pvals_RXR <- pvals_RXR[rownames(pvals_RXR),]
pvals_RXR <-sapply(pvals_RXR, function(x) ifelse(x, x <- "*",x<-"") )
rownames(pvals_RXR) <- rownames(logchange_RXR)




##KFF
logchange_KFF <- data.frame(PNAKFF = res_KFF$PNAKFF$table$logFC,PNAKFFscr =res_KFF$PNAKFFscr$table$logF
                              KFF =res_KFF$KFF$table$logFC, row.names = rownames(res_KFF$PNAKFF$table))

pvals_KFF <- data.frame(PNAKFF = (res_KFF$PNAKFF$table$p_value_FDR<0.001 & abs(res_KFF$PNAKFF$table$log
                        PNAKFFscr=(res_KFF$PNAKFFscr$table$p_value_FDR<0.001 & abs(res_KFF$PNAKFFscr$ta
                        KFF =(res_KFF$KFF$table$p_value_FDR<0.001 & abs(res_KFF$KFF$table$logFC)>1),
                        row.names = rownames(res_KFF$PNAKFF$table))
#select only significant ones:
logchange_KFF <- logchange_KFF[rownames(logchange_KFF)%in%topDE_srnas,]
logchange_KFF <- logchange_KFF[rownames(logchange_KFF),]
colnames(logchange_KFF) <- factor(c("KFF-acpP ", "KFF-acpP-scrambled ", "KFF "))

prefname <- ifelse(rownames(logchange_KFF) %in% pnames$V2 ,pnames[rownames(logchange_KFF),]$V1, "" )
prefname <- ifelse(isUnique(prefname), prefname, "")
rownames(logchange_KFF) <- ifelse(prefname != "", prefname, rownames(logchange_KFF))


#select only significant ones:
pvals_KFF <- pvals_KFF[rownames(pvals_KFF)%in%topDE_srnas,]
pvals_KFF <- pvals_KFF[rownames(pvals_KFF),]
```

```r
pvals_KFF <-sapply(pvals_KFF, function(x) ifelse(x, x <- "*",x<-"") )
rownames(pvals_KFF) <- rownames(logchange_KFF)


logfcsrnas <- cbind(logchange_KFF, logchange_RXR, logchange_TAT)
pvalssrnas <- cbind(pvals_KFF, pvals_RXR, pvals_TAT)
logfcsrnas <-logfcsrnas[order(logfcsrnas$`KFF-acpP `),]
pvalssrnas <- pvalssrnas[rownames(logfcsrnas),]
```

Heatmap:

```r
col_fun = colorRamp2(c(-2, 0, 2), c("darkblue", "beige", "red"))
ht_vert <- Heatmap(logfcsrnas, cluster_rows = F, cluster_columns = F,
               name = "sRNA", col = col_fun,
               show_heatmap_legend = F,
               row_title_side = "left", row_title_rot = 0,
               #border = TRUE,
               cell_fun = function(j, i, x, y, width, height, fill) {
                 grid.text(sprintf("%.1s", pvalssrnas[i, j]), x, y)
               },
               column_names_gp = gpar(fontsize = 10),
               row_names_gp = gpar(fontsize = 10),
               column_split = factor(c(rep("KFF",3),rep("RXR",3),rep("TAT",3))),
               width = unit(9*0.7, "cm"), height = unit(dim(logfcsrnas)[1]/2, "cm"),
               column_names_rot = 45,  border = TRUE)
```

```
## Warning: The input is a data frame, convert it to the matrix.
```

```r
               #, rect_gp = gpar(col = "black", lwd = 0.01))

lgd = Legend(col_fun = col_fun, title = expression("Log"[2]*" FC"), labels_gp = gpar(fontsize = 8),
             title_gp = gpar(fontsize = 12),
             at = c(-2, 0, 2), legend_width = unit(4, "cm"),
             labels = c("-2", "  0", "  2"), legend_height = unit(3, "cm"),
             title_position = "leftcenter-rot")

pdf("../analysis/heatmaps/heatmap_srnas.pdf")
draw(ht_vert)
draw(lgd, x = unit(2, "cm"), y = unit(8, "cm"), just = c("left", "bottom"))
dev.off()
```

```
## pdf
##   2
```

#Venn diagrams: I create some Venn diagrams (not included in manuscript) to get overview of overlapping DE genes:

```r
list_DEgenes <- list(PNA_TAT = rownames(res_TAT$PNATAT$table[res_TAT$PNATAT$table$p_value_FDR<0.001 &
                                  abs(res_TAT$PNATAT$table$logFC)>1,]),
                PNA_TAT_scr = rownames(res_TAT$PNATATscr$table[res_TAT$PNATATscr$table$p_value_FDR<0.00
                                  abs(res_TAT$PNATATscr$table$logFC)>1,]),
                TAT = rownames(res_TAT$TAT$table[res_TAT$TAT$table$p_value_FDR<0.001 &
                                  abs(res_TAT$TAT$table$logFC)>1,]),
                PNA_RXR = rownames(res_RXR$PNARXR$table[res_RXR$PNARXR$table$p_value_FDR<0.001 &
                                  abs(res_RXR$PNARXR$table$logFC)>1,]),
                PNA_RXR_scr = rownames(res_RXR$PNARXRscr$table[res_RXR$PNARXRscr$table$p_value_FDR<0.00
                                           abs(res_RXR$PNARXRscr$table$logFC)>1,]),
                RXR = rownames(res_RXR$RXR$table[res_RXR$RXR$table$p_value_FDR<0.001 &
                                           abs(res_RXR$RXR$table$logFC)>1,]),
                PNA_KFF = rownames(res_KFF$PNAKFF$table[res_KFF$PNAKFF$table$p_value_FDR<0.001 &
                                           abs(res_KFF$PNAKFF$table$logFC)>1,]),
                PNA_KFF_scr = rownames(res_KFF$PNAKFFscr$table[res_KFF$PNAKFFscr$table$p_value_FDR<0.00
                                           abs(res_KFF$PNAKFFscr$table$logFC)>1,]),
                KFF = rownames(res_KFF$KFF$table[res_KFF$KFF$table$p_value_FDR<0.001 &
                                           abs(res_KFF$KFF$table$logFC)>1,]))

svg("../analysis/VennDiagrams/Venn_TMM_TAT",width = 10, height = 10)
plot(euler(list_DEgenes[1:3]) , quantities = T)
dev.off()
```

```
## pdf
##   2
```

```
svg("../analysis/VennDiagrams/Venn_TMM_RXR",width = 10, height = 10)
plot(euler(list_DEgenes[4:6]) , quantities = T)
dev.off()
```

```
## pdf
##   2
```

```
svg("../analysis/VennDiagrams/Venn_TMM_KFF",width = 10, height = 10)
plot(euler(list_DEgenes[7:9]) , quantities = T)
dev.off()
```

```
## pdf
##   2
```

```
svg("../analysis/VennDiagrams/Venn_TMM_peptides",width = 10, height = 10)
plot(euler(list_DEgenes[c(3,6,9)]) , quantities = T)
dev.off()
```

```
## pdf
##   2
```

```
svg("../analysis/VennDiagrams/Venn_TMM_PPNAs",width = 10, height = 10)
plot(euler(list_DEgenes[c(1,4,7)]) , quantities = T)
dev.off()
```

```
## pdf
##   2
```

KEGG pathway analysis

I perform the KEGG-analysis using the FRY gene set analysis tool from limma:

```
library(KEGGREST)
# get link and list to get kegg info:
link_kegg <- keggLink("pathway", "sey")
list_kegg <- keggList("pathway", "sey")

kegg_pw_ids <- names(list_kegg)

#rename genes, remove ones which arent in our data:
```

```
names(link_kegg) <- gsub("sey:(.*)", "\\1", names(link_kegg)) #rename genes as locus tags
link_kegg <- link_kegg[names(link_kegg) %in% rownames(res_KFF$PNAKFF$table)] #remove genes not in data


idx_kegg <- sapply(kegg_pw_ids, function(x){
  x <- unique(names(link_kegg[link_kegg == x])) # choose all genes, except duplucates
})
# add phopq pw to kegg
ppq_raw <- read.delim("../data/PHOPQ.tsv", header = F)
ppq <- as.character(ppq_raw$V1)
phopq <- pnames[pnames$V1 %in% ppq,]$V2

idx_kegg$PhoPQ <- phopq[phopq %in% rownames(y$counts)] # add PhoPQ genes


#do fry:
kegg_fry_PNAKFF <- fry(y, idx_kegg, design, contrast=con[,1])
kegg_fry_KFF <- fry(y, idx_kegg, design, contrast=con[,3])
kegg_fry_PNAKFFscr <- fry(y, idx_kegg, design, contrast=con[,2])

kegg_fry_PNARXR <- fry(y, idx_kegg, design, contrast=con[,4])
kegg_fry_RXR <- fry(y, idx_kegg, design, contrast=con[,6])
kegg_fry_PNARXRscr <- fry(y, idx_kegg, design, contrast=con[,5])

kegg_fry_PNATAT <- fry(y, idx_kegg, design, contrast=con[,7])
kegg_fry_TAT <- fry(y, idx_kegg, design, contrast=con[,9])
kegg_fry_PNATATscr <- fry(y, idx_kegg, design, contrast=con[,8])

list_kegg_fry <- list(kegg_fry_PNAKFF=kegg_fry_PNAKFF,kegg_fry_PNAKFFscr=kegg_fry_PNAKFFscr,kegg_fry_KF
                      kegg_fry_PNARXR=kegg_fry_PNARXR,kegg_fry_PNARXRscr=kegg_fry_PNARXRscr,kegg_fry_RXR=kegg
                      kegg_fry_PNATAT=kegg_fry_PNATAT,kegg_fry_PNATATscr=kegg_fry_PNATATscr,kegg_fry_TAT=kegg
```

add KEGG terms:

```
for (fryres in names(list_kegg_fry)) {
  list_kegg_fry[[fryres]][["TERM"]] <- list_kegg[rownames(list_kegg_fry[[fryres]])]
  list_kegg_fry[[fryres]][["TERM"]] <- gsub("(.*) - Salmonella enterica subsp. enterica serovar Typhimu
                                             "\\1", list_kegg_fry[[fryres]][["TERM"]])
  list_kegg_fry[[fryres]]["PhoPQ",][["TERM"]] <- "PhoPQ"
}


kegg_frysig <- lapply(list_kegg_fry, function(x) x[x[["FDR"]]<0.05 & x[["NGenes"]]>10,])
kegg_siggos <- c()


for (i in names(kegg_frysig)) {
  print(i)
  print(dim(kegg_frysig[[i]]))
  print(kegg_frysig[[i]][,c(1,2,4,7)])
  kegg_siggos <- c(kegg_siggos, rownames(kegg_frysig[[i]][1:10,]))  # can be modified
}
```

```
## [1] "kegg_fry_PNAKFF"
## [1] 2 7
##               NGenes Direction     FDR            TERM
## PhoPQ             15        Up 0.00466           PhoPQ
## path:sey03430     25      Down 0.00466 Mismatch repair
## [1] "kegg_fry_PNAKFFscr"
## [1] 3 7
##               NGenes Direction      FDR
## PhoPQ             15        Up 4.47e-05
## path:sey01503     36        Up 3.13e-02
## path:sey02020    176        Up 4.12e-02
##                                                           TERM
## PhoPQ                                                    PhoPQ
## path:sey01503 Cationic antimicrobial peptide (CAMP) resistance
## path:sey02020                             Two-component system
## [1] "kegg_fry_KFF"
## [1] 2 7
##               NGenes Direction      FDR
## PhoPQ             15        Up 3.09e-07
## path:sey01503     36        Up 1.47e-03
##                                                           TERM
## PhoPQ                                                    PhoPQ
## path:sey01503 Cationic antimicrobial peptide (CAMP) resistance
## [1] "kegg_fry_PNARXR"
## [1] 48  7
##               NGenes Direction      FDR
## path:sey02020    176        Up 1.04e-05
## path:sey02060     39        Up 2.26e-05
## path:sey00910     21        Up 8.01e-05
## path:sey00051     40        Up 1.28e-04
## path:sey00053     15        Up 1.28e-04
## path:sey03440     29      Down 1.95e-04
## path:sey00061     13      Down 2.46e-04
## path:sey00650     31        Up 2.46e-04
## path:sey02010    166        Up 2.46e-04
## path:sey03018     16      Down 2.96e-04
## path:sey00052     28        Up 2.96e-04
## path:sey03430     25      Down 2.96e-04
## path:sey00920     35        Up 5.13e-04
## path:sey00900     12      Down 5.64e-04
## path:sey00220     17        Up 8.86e-04
## path:sey00071     13        Up 9.28e-04
## path:sey00640     35        Up 9.79e-04
## path:sey00860     31        Up 1.26e-03
## path:sey01120    253        Up 1.26e-03
## path:sey03060     18      Down 1.72e-03
## path:sey00230     68      Down 2.12e-03
## path:sey00480     24      Down 2.12e-03
## path:sey00790     24      Down 2.25e-03
## path:sey00040     29        Up 2.25e-03
## path:sey00500     35        Up 2.25e-03
## path:sey01210     29        Up 2.60e-03
## path:sey00310     14        Up 2.94e-03
## path:sey01212     20      Down 3.42e-03
## path:sey01501     20        Up 3.97e-03
## path:sey00970     26      Down 5.29e-03
## path:sey01100    860        Up 5.45e-03
## path:sey00030     36      Down 6.31e-03
## PhoPQ             15        Up 6.39e-03
## path:sey00780     14      Down 8.31e-03
## path:sey00630     30        Up 1.00e-02
## path:sey04122     17      Down 1.10e-02
## path:sey00350     15        Up 1.16e-02
## path:sey00290     20        Up 1.33e-02
## path:sey03010     56      Down 1.64e-02
## path:sey00400     23        Up 2.02e-02
## path:sey01240    152      Down 2.82e-02
## path:sey00660     16        Up 2.91e-02
## path:sey03030     19      Down 3.22e-02
## path:sey00340     12      Down 3.48e-02
## path:sey00680     29      Down 3.95e-02
## path:sey00270     35      Down 4.52e-02
## path:sey00450     15        Up 4.62e-02
## path:sey00770     25        Up 4.67e-02
##                                                          TERM
## path:sey02020                             Two-component system
## path:sey02060                      Phosphotransferase system (PTS)
## path:sey00910                              Nitrogen metabolism
## path:sey00051                      Fructose and mannose metabolism
## path:sey00053                     Ascorbate and aldarate metabolism
## path:sey03440                           Homologous recombination
## path:sey00061                            Fatty acid biosynthesis
## path:sey00650                             Butanoate metabolism
## path:sey02010                                 ABC transporters
## path:sey03018                                  RNA degradation
## path:sey00052                             Galactose metabolism
## path:sey03430                                  Mismatch repair
## path:sey00920                                Sulfur metabolism
## path:sey00900                      Terpenoid backbone biosynthesis
## path:sey00220                             Arginine biosynthesis
## path:sey00071                             Fatty acid degradation
## path:sey00640                             Propanoate metabolism
## path:sey00860                 Porphyrin and chlorophyll metabolism
## path:sey01120           Microbial metabolism in diverse environments
## path:sey03060                                   Protein export
## path:sey00230                                Purine metabolism
## path:sey00480                            Glutathione metabolism
## path:sey00790                              Folate biosynthesis
## path:sey00040               Pentose and glucuronate interconversions
## path:sey00500                        Starch and sucrose metabolism
## path:sey01210                      2-Oxocarboxylic acid metabolism
## path:sey00310                               Lysine degradation
## path:sey01212                            Fatty acid metabolism
## path:sey01501                            beta-Lactam resistance
## path:sey00970                        Aminoacyl-tRNA biosynthesis
## path:sey01100                               Metabolic pathways
## path:sey00030                          Pentose phosphate pathway
## PhoPQ                                                        PhoPQ
## path:sey00780                                Biotin metabolism
## path:sey00630            Glyoxylate and dicarboxylate metabolism
## path:sey04122                                Sulfur relay system
## path:sey00350                                Tyrosine metabolism
## path:sey00290        Valine, leucine and isoleucine biosynthesis
## path:sey03010                                           Ribosome
## path:sey00400 Phenylalanine, tyrosine and tryptophan biosynthesis
## path:sey01240                          Biosynthesis of cofactors
## path:sey00660                C5-Branched dibasic acid metabolism
## path:sey03030                                    DNA replication
## path:sey00340                               Histidine metabolism
## path:sey00680                                 Methane metabolism
## path:sey00270               Cysteine and methionine metabolism
## path:sey00450                          Selenocompound metabolism
## path:sey00770                  Pantothenate and CoA biosynthesis
## [1] "kegg_fry_PNARXRscr"
## [1] 41  7
##               NGenes Direction      FDR
## path:sey02020    176        Up 4.17e-05
## path:sey02060     39        Up 1.42e-04
## path:sey00053     15        Up 7.43e-04
## path:sey01501     20        Up 7.43e-04
## path:sey00910     21        Up 7.43e-04
## path:sey00640     35        Up 7.43e-04
## path:sey02010    166        Up 1.20e-03
## path:sey00650     31        Up 1.35e-03
## path:sey00970     26      Down 1.50e-03
## path:sey03430     25      Down 1.50e-03
## path:sey03060     18      Down 2.07e-03
## path:sey01210     29        Up 2.07e-03
## path:sey00051     40        Up 2.07e-03
## path:sey00920     35        Up 2.17e-03
## path:sey03440     29      Down 2.65e-03
## path:sey00071     13        Up 2.65e-03
## path:sey00310     14        Up 2.77e-03
## path:sey00061     13      Down 3.38e-03
## path:sey01120    253        Up 3.61e-03
## path:sey00860     31        Up 3.61e-03
## path:sey00230     68      Down 4.99e-03
## path:sey00790     24      Down 5.71e-03
## path:sey00040     29        Up 7.39e-03
## path:sey00500     35        Up 7.39e-03
## path:sey00290     20        Up 7.39e-03
## path:sey00052     28        Up 7.39e-03
## path:sey00900     12      Down 7.39e-03
## PhoPQ             15        Up 7.88e-03
## path:sey00220     17        Up 7.88e-03
## path:sey00350     15        Up 9.32e-03
## path:sey01100    860        Up 1.06e-02
## path:sey00770     25        Up 1.06e-02
## path:sey00550     26      Down 1.11e-02
## path:sey03018     16      Down 1.97e-02
## path:sey05132     33        Up 2.25e-02
## path:sey00680     29      Down 2.98e-02
## path:sey01240    152      Down 3.17e-02
## path:sey03030     19      Down 3.46e-02
## path:sey00780     14      Down 3.69e-02
## path:sey00660     16        Up 3.69e-02
## path:sey03010     56      Down 4.68e-02
##                                                 TERM
## path:sey02020                    Two-component system
## path:sey02060               Phosphotransferase system (PTS)
## path:sey00053             Ascorbate and aldarate metabolism
## path:sey01501                      beta-Lactam resistance
## path:sey00910                        Nitrogen metabolism
## path:sey00640                       Propanoate metabolism
## path:sey02010                           ABC transporters
## path:sey00650                        Butanoate metabolism
## path:sey00970                  Aminoacyl-tRNA biosynthesis
## path:sey03430                              Mismatch repair
## path:sey03060                               Protein export
## path:sey01210               2-Oxocarboxylic acid metabolism
## path:sey00051               Fructose and mannose metabolism
## path:sey00920                           Sulfur metabolism
## path:sey03440                     Homologous recombination
## path:sey00071                      Fatty acid degradation
## path:sey00310                          Lysine degradation
## path:sey00061                     Fatty acid biosynthesis
## path:sey01120 Microbial metabolism in diverse environments
## path:sey00860          Porphyrin and chlorophyll metabolism
## path:sey00230                           Purine metabolism
## path:sey00790                          Folate biosynthesis
## path:sey00040     Pentose and glucuronate interconversions
## path:sey00500                 Starch and sucrose metabolism
## path:sey00290  Valine, leucine and isoleucine biosynthesis
## path:sey00052                        Galactose metabolism
## path:sey00900               Terpenoid backbone biosynthesis
## PhoPQ                                               PhoPQ
## path:sey00220                       Arginine biosynthesis
## path:sey00350                          Tyrosine metabolism
## path:sey01100                           Metabolic pathways
## path:sey00770            Pantothenate and CoA biosynthesis
## path:sey00550                   Peptidoglycan biosynthesis
## path:sey03018                              RNA degradation
## path:sey05132                          Salmonella infection
## path:sey00680                           Methane metabolism
## path:sey01240                    Biosynthesis of cofactors
## path:sey03030                              DNA replication
## path:sey00780                            Biotin metabolism
## path:sey00660           C5-Branched dibasic acid metabolism
## path:sey03010                                     Ribosome
## [1] "kegg_fry_RXR"
## [1] 3 7
##               NGenes Direction      FDR
## PhoPQ             15        Up 9.54e-07
## path:sey01503     36        Up 2.78e-03
## path:sey02020    176        Up 3.53e-02
##                                                     TERM
## PhoPQ                                               PhoPQ
## path:sey01503 Cationic antimicrobial peptide (CAMP) resistance
## path:sey02020                              Two-component system
## [1] "kegg_fry_PNATAT"
## [1] 48  7
##                NGenes Direction      FDR
## path:sey02020    176        Up 7.03e-07
## path:sey00640     35        Up 7.03e-07
## path:sey02010    166        Up 3.12e-06
## path:sey02060     39        Up 3.74e-06
## path:sey00910     21        Up 1.05e-05
## path:sey00051     40        Up 3.04e-05
## path:sey00053     15        Up 3.88e-05
## path:sey01210     29        Up 7.81e-05
## path:sey01501     20        Up 7.81e-05
## path:sey01120    253        Up 7.81e-05
## path:sey00650     31        Up 7.81e-05
## path:sey00920     35        Up 9.89e-05
## path:sey00310     14        Up 9.89e-05
## path:sey00860     31        Up 1.12e-04
## path:sey00220     17        Up 1.16e-04
## path:sey01100    860        Up 1.16e-04
## path:sey00071     13        Up 1.16e-04
## path:sey03018     16      Down 1.33e-04
## path:sey00052     28        Up 2.00e-04
## path:sey00500     35        Up 2.37e-04
## path:sey00061     13      Down 3.36e-04
## path:sey03060     18      Down 3.36e-04
## path:sey00790     24      Down 4.71e-04
## path:sey03430     25      Down 4.98e-04
## path:sey00290     20        Up 6.80e-04
## path:sey03440     29      Down 6.80e-04
## path:sey00900     12      Down 8.08e-04
## path:sey00970     26      Down 8.98e-04
## path:sey00620     55        Up 9.09e-04
## path:sey00450     15        Up 9.63e-04
## path:sey00350     15        Up 1.07e-03
## path:sey00630     30        Up 1.30e-03
## path:sey00020     27        Up 1.60e-03
## path:sey00040     29        Up 2.09e-03
## path:sey03030     19      Down 2.09e-03
## path:sey00330     23        Up 3.85e-03
## path:sey00400     23        Up 3.85e-03
## path:sey03010     56      Down 4.35e-03
## path:sey01110    348        Up 4.54e-03
## path:sey04122     17      Down 8.83e-03
## path:sey01230    127        Up 2.48e-02
## path:sey02024     60        Up 2.51e-02
## path:sey00770     25        Up 2.71e-02
## path:sey01212     20      Down 3.01e-02
## path:sey00230     68      Down 3.02e-02
## path:sey00300     13        Up 3.80e-02
## path:sey00780     14      Down 3.80e-02
## path:sey00660     16        Up 3.85e-02
##                                                                 TERM
## path:sey02020                                Two-component system
## path:sey00640                               Propanoate metabolism
## path:sey02010                                    ABC transporters
## path:sey02060                     Phosphotransferase system (PTS)
## path:sey00910                                 Nitrogen metabolism
## path:sey00051                     Fructose and mannose metabolism
## path:sey00053                   Ascorbate and aldarate metabolism
## path:sey01210                     2-Oxocarboxylic acid metabolism
## path:sey01501                              beta-Lactam resistance
## path:sey01120        Microbial metabolism in diverse environments
## path:sey00650                                Butanoate metabolism
## path:sey00920                                   Sulfur metabolism
## path:sey00310                                  Lysine degradation
## path:sey00860                Porphyrin and chlorophyll metabolism
## path:sey00220                               Arginine biosynthesis
## path:sey01100                                  Metabolic pathways
## path:sey00071                              Fatty acid degradation
## path:sey03018                                     RNA degradation
## path:sey00052                                Galactose metabolism
## path:sey00500                       Starch and sucrose metabolism
## path:sey00061                             Fatty acid biosynthesis
## path:sey03060                                      Protein export
## path:sey00790                                 Folate biosynthesis
## path:sey03430                                     Mismatch repair
## path:sey00290         Valine, leucine and isoleucine biosynthesis
## path:sey03440                            Homologous recombination
## path:sey00900                     Terpenoid backbone biosynthesis
## path:sey00970                         Aminoacyl-tRNA biosynthesis
## path:sey00620                                 Pyruvate metabolism
## path:sey00450                           Selenocompound metabolism
## path:sey00350                                 Tyrosine metabolism
## path:sey00630             Glyoxylate and dicarboxylate metabolism
## path:sey00020                           Citrate cycle (TCA cycle)
## path:sey00040            Pentose and glucuronate interconversions
## path:sey03030                                     DNA replication
## path:sey00330                     Arginine and proline metabolism
## path:sey00400 Phenylalanine, tyrosine and tryptophan biosynthesis
## path:sey03010                                            Ribosome
## path:sey01110               Biosynthesis of secondary metabolites
## path:sey04122                                 Sulfur relay system
## path:sey01230                         Biosynthesis of amino acids
## path:sey02024                                      Quorum sensing
## path:sey00770                   Pantothenate and CoA biosynthesis
## path:sey01212                               Fatty acid metabolism
## path:sey00230                                   Purine metabolism
## path:sey00300                                 Lysine biosynthesis
## path:sey00780                                   Biotin metabolism
## path:sey00660                 C5-Branched dibasic acid metabolism
## [1] "kegg_fry_PNATATscr"
## [1] 52  7
##               NGenes Direction      FDR
## path:sey02020    176        Up 7.64e-07
## path:sey00640     35        Up 7.64e-07
## path:sey02060     39        Up 1.93e-06
## path:sey00910     21         Up 2.44e-06
## path:sey02010    166         Up 7.00e-06
## path:sey00051     40         Up 7.00e-06
## path:sey00790     24       Down 2.39e-05
## path:sey01120    253         Up 2.66e-05
## path:sey01501     20         Up 2.84e-05
## path:sey00053     15         Up 3.35e-05
## path:sey00650     31         Up 3.48e-05
## path:sey00220     17         Up 3.48e-05
## path:sey00920     35         Up 3.48e-05
## path:sey01210     29         Up 3.48e-05
## path:sey00860     31         Up 3.85e-05
## path:sey00052     28         Up 4.18e-05
## path:sey03430     25       Down 6.62e-05
## path:sey00500     35         Up 7.55e-05
## path:sey01100    860         Up 8.99e-05
## path:sey03440     29       Down 9.03e-05
## path:sey00310     14         Up 9.75e-05
## path:sey00071     13         Up 1.12e-04
## path:sey03018     16       Down 1.24e-04
## path:sey03060     18       Down 1.36e-04
## path:sey00350     15         Up 2.01e-04
## path:sey00620     55         Up 2.73e-04
## path:sey03030     19       Down 3.23e-04
## path:sey00970     26       Down 3.32e-04
## path:sey00630     30         Up 3.84e-04
## path:sey00330     23         Up 3.85e-04
## path:sey00900     12       Down 6.53e-04
## path:sey00061     13       Down 6.80e-04
## path:sey00020     27         Up 7.43e-04
## path:sey00040     29         Up 8.47e-04
## path:sey00290     20         Up 9.03e-04
## path:sey04122     17       Down 1.09e-03
## path:sey00450     15         Up 1.18e-03
## path:sey01110    348         Up 3.61e-03
## path:sey03010     56       Down 3.89e-03
## path:sey01240    152       Down 6.07e-03
## path:sey00400     23         Up 6.86e-03
## path:sey00660     16         Up 9.78e-03
## path:sey02024     60         Up 1.26e-02
## path:sey00010     43         Up 1.63e-02
## path:sey01230    127         Up 1.63e-02
## path:sey00230     68       Down 1.86e-02
## path:sey00260     36         Up 2.07e-02
## path:sey05132     33         Up 2.73e-02
## path:sey01212     20       Down 3.93e-02
## path:sey00300     13         Up 4.33e-02
## path:sey00770     25         Up 4.47e-02
## path:sey00540     39       Down 4.85e-02
##                                                             TERM
## path:sey02020                                Two-component system
## path:sey00640                               Propanoate metabolism
## path:sey02060                     Phosphotransferase system (PTS)
## path:sey00910                                 Nitrogen metabolism
## path:sey02010                                    ABC transporters
## path:sey00051                     Fructose and mannose metabolism
## path:sey00790                                Folate biosynthesis
## path:sey01120         Microbial metabolism in diverse environments
## path:sey01501                             beta-Lactam resistance
## path:sey00053                   Ascorbate and aldarate metabolism
## path:sey00650                               Butanoate metabolism
## path:sey00220                              Arginine biosynthesis
## path:sey00920                                  Sulfur metabolism
## path:sey01210                     2-Oxocarboxylic acid metabolism
## path:sey00860                Porphyrin and chlorophyll metabolism
## path:sey00052                               Galactose metabolism
## path:sey03430                                    Mismatch repair
## path:sey00500                       Starch and sucrose metabolism
## path:sey01100                                 Metabolic pathways
## path:sey03440                             Homologous recombination
## path:sey00310                                 Lysine degradation
## path:sey00071                             Fatty acid degradation
## path:sey03018                                    RNA degradation
## path:sey03060                                     Protein export
## path:sey00350                                Tyrosine metabolism
## path:sey00620                                Pyruvate metabolism
## path:sey03030                                    DNA replication
## path:sey00970                         Aminoacyl-tRNA biosynthesis
## path:sey00630             Glyoxylate and dicarboxylate metabolism
## path:sey00330                   Arginine and proline metabolism
## path:sey00900                     Terpenoid backbone biosynthesis
## path:sey00061                            Fatty acid biosynthesis
## path:sey00020                            Citrate cycle (TCA cycle)
## path:sey00040            Pentose and glucuronate interconversions
## path:sey00290         Valine, leucine and isoleucine biosynthesis
## path:sey04122                                Sulfur relay system
## path:sey00450                            Selenocompound metabolism
## path:sey01110               Biosynthesis of secondary metabolites
## path:sey03010                                             Ribosome
## path:sey01240                            Biosynthesis of cofactors
## path:sey00400 Phenylalanine, tyrosine and tryptophan biosynthesis
## path:sey00660                    C5-Branched dibasic acid metabolism
## path:sey02024                                       Quorum sensing
## path:sey00010                          Glycolysis / Gluconeogenesis
## path:sey01230                           Biosynthesis of amino acids
## path:sey00230                                   Purine metabolism
## path:sey00260             Glycine, serine and threonine metabolism
## path:sey05132                                  Salmonella infection
## path:sey01212                                Fatty acid metabolism
## path:sey00300                                  Lysine biosynthesis
## path:sey00770                       Pantothenate and CoA biosynthesis
## path:sey00540                      Lipopolysaccharide biosynthesis
## [1] "kegg_fry_TAT"
## [1] 13  7
##               NGenes Direction      FDR
## PhoPQ             15        Up 4.84e-10
## path:sey02020    176        Up 1.78e-05
## path:sey01503     36        Up 2.63e-04
## path:sey00250     33      Down 5.03e-04
## path:sey03440     29      Down 5.93e-03
## path:sey03430     25      Down 7.24e-03
## path:sey03030     19      Down 1.97e-02
## path:sey01240    152      Down 2.95e-02
## path:sey00230     68      Down 2.95e-02
## path:sey02060     39        Up 2.98e-02
## path:sey00970     26      Down 4.54e-02
## path:sey00240     43      Down 4.54e-02
## path:sey00900     12      Down 4.54e-02
##                                                        TERM
## PhoPQ                                                 PhoPQ
## path:sey02020                          Two-component system
## path:sey01503 Cationic antimicrobial peptide (CAMP) resistance
## path:sey00250      Alanine, aspartate and glutamate metabolism
## path:sey03440                        Homologous recombination
## path:sey03430                                 Mismatch repair
## path:sey03030                                 DNA replication
## path:sey01240                       Biosynthesis of cofactors
## path:sey00230                               Purine metabolism
## path:sey02060                  Phosphotransferase system (PTS)
## path:sey00970                      Aminoacyl-tRNA biosynthesis
## path:sey00240                            Pyrimidine metabolism
## path:sey00900                  Terpenoid backbone biosynthesis
```

```
kegg_siggos <- unique(kegg_siggos[!grepl("NA", kegg_siggos)])
```

Create a heatmap-df for KEGG:

```
idx_kegg_char <- lapply(idx_kegg, as.character)
# I create a dataframe with mean logFC values for each significant GO-term:
hm_kegg_fry_logfc <- t(as.data.frame(lapply(idx_kegg_char[kegg_siggos], function(x){
  PNAKFF <- median(res_KFF$PNAKFF$table[x,]$logFC)
  PNAKFFscr <- median(res_KFF$PNAKFFscr$table[x,]$logFC)
  KFF <- median(res_KFF$KFF$table[x,]$logFC)

  PNARXR <- median(res_RXR$PNARXR$table[x,]$logFC)
  PNARXRscr <- median(res_RXR$PNARXRscr$table[x,]$logFC)
  RXR <- median(res_RXR$RXR$table[x,]$logFC)

  PNATAT <- median(res_TAT$PNATAT$table[x,]$logFC)
  PNATATscr <- median(res_TAT$PNATATscr$table[x,]$logFC)
  TAT <- median(res_TAT$TAT$table[x,]$logFC)
  c(PNAKFF, PNAKFFscr, KFF, PNARXR, PNARXRscr, RXR, PNATAT, PNATATscr, TAT)
})))

colnames(hm_kegg_fry_logfc) <- c("KFF-acpP ", "KFF-acpP-scrambled ", "KFF-only ","RXR-acpP ", "RXR-acpP-
                            "TAT-acpP ", "TAT-acpP-scrambled ", "TAT-only ")
hm_kegg_fry_logfc <- as.data.frame(hm_kegg_fry_logfc)
rownames(hm_kegg_fry_logfc) <- gsub("\\.", "\\:", rownames(hm_kegg_fry_logfc))
```

make heatmap:

```
hm_kegg_fry_logfc <- hm_kegg_fry_logfc[order(hm_kegg_fry_logfc[,1], decreasing = T),]

pvals <- data.frame(sapply(names(list_kegg_fry), function(x) list_kegg_fry[[x]][rownames(hm_kegg_fry_log
```

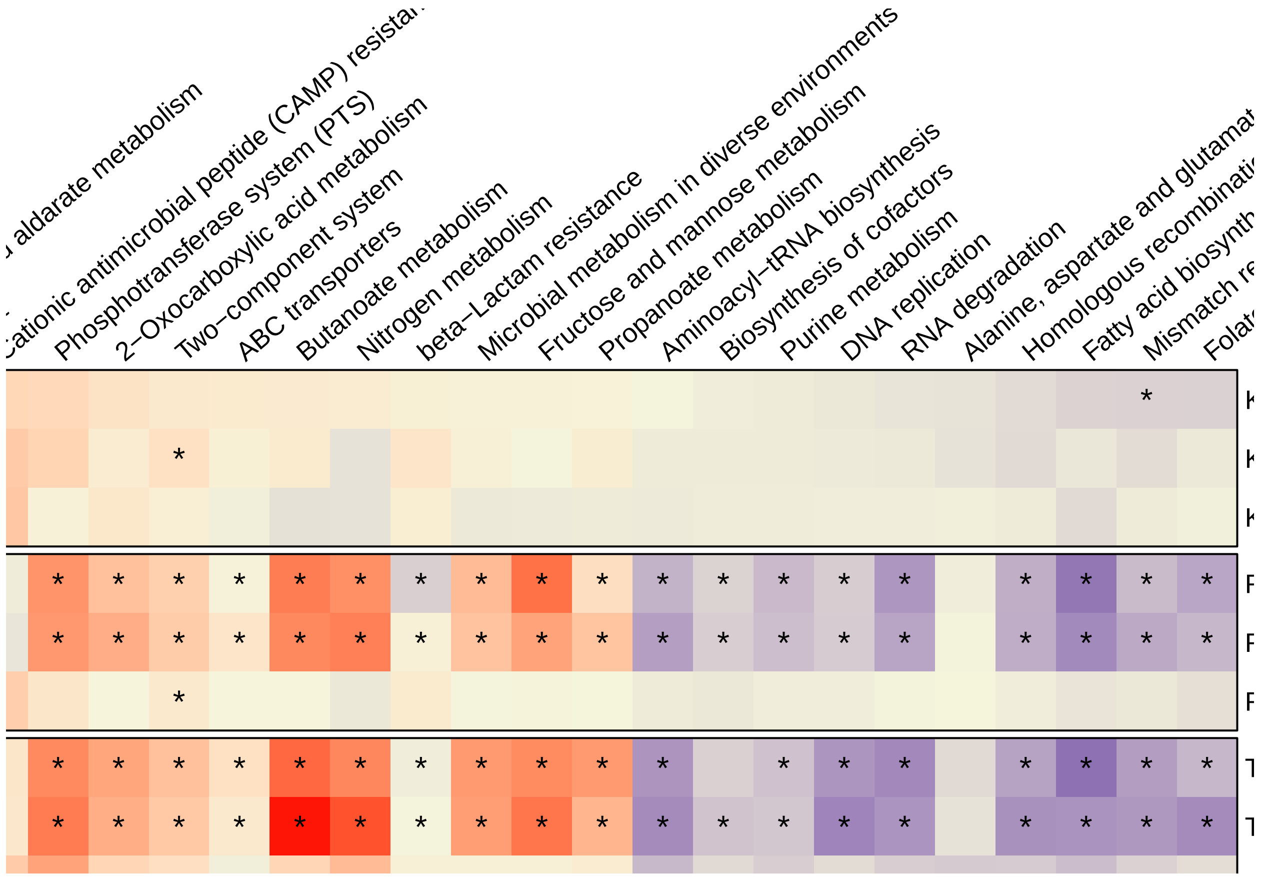
```r
                    row.names = rownames(hm_kegg_fry_logfc))

#select only significant ones:
pvals <-sapply(pvals, function(x) ifelse(x<0.05, x <- "*",x<-"") )


keggpws <- list_kegg_fry$kegg_fry_PNAKFF[rownames(hm_kegg_fry_logfc),] [["TERM"]]
rownames(hm_kegg_fry_logfc) <- ifelse(!is.na(keggpws),keggpws, rownames(hm_kegg_fry_logfc) )


ord <- c(rep("KFF", 3), rep("RXR", 3), rep("TAT", 3))
lev <- c("KFF", "RXR", "TAT")
```

plot hm (save as pdf):

```r
library(circlize)
col_fun = colorRamp2(c(-1, 0, 1), c("darkblue", "beige", "red"))

w <- length(hm_kegg_fry_logfc$`KFF-acpP `) # width of plot (nr pws)
h <- 9   # height of plot

ht_vert <- Heatmap(hm_kegg_fry_logfc, cluster_rows = F, cluster_columns = F,
               name = "GO-analysis", col = col_fun,
               show_heatmap_legend = F,
               row_title_side = "right", row_title_rot = 0,
               border = TRUE,
               cell_fun = function(j, i, x, y, width, height, fill) {
                 grid.text(sprintf("%.1s", pvals[i, j]), x, y)
               },
               column_names_gp = gpar(fontsize = 10),
               row_names_gp = gpar(fontsize = 10),
               column_split = factor(ord, levels = lev),
               row_title = NULL,
               row_gap = unit(0.1, "cm"),
               width = unit(5, "cm"), height = unit(20, "cm"),
               column_names_rot = 45)
```

```
## Warning: The input is a data frame, convert it to the matrix.
```

```r
ht_hor <- Heatmap(t(hm_kegg_fry_logfc), cluster_rows = F, cluster_columns = F,
        name = "GO-analysis", col = col_fun,
        show_heatmap_legend = F,
        row_title_side = "left", row_title_rot = 0,
        column_names_side = "top",
        border = TRUE,
        cell_fun = function(j, i, x, y, width, height, fill) {
          grid.text(sprintf("%.1s", t(pvals)[i, j]), x, y)
        },
        row_names_gp = gpar(fontsize = 10),
        column_names_gp = gpar(fontsize = 10),
        row_split = factor(ord, levels = lev),
        column_title = NULL,
        column_gap = unit(0.1, "cm"),
        width = unit(w*0.8, "cm"), height = unit(h*0.8, "cm"),
        column_names_rot = 40)
        #rect_gp = gpar(col = "black", lwd = 0.1))

ht_hor
```



```
lgd = Legend(col_fun = col_fun, title = expression("Median logFC"), direction = "horizontal",
             title_gp = gpar(fontsize = 12),
             at = c(-1, 0, 1), legend_width = unit(4, "cm"))

lgd1 = Legend(col_fun = c1, title = expression("Log CPM"), labels_gp = gpar(fontsize = 10),
             title_gp = gpar(fontsize = 15),
             at = c(-2, 0, 2), legend_width = unit(4, "cm"), grid_width =  unit(0.8, "cm"),
             labels = c("-2", "  0", "  2"), legend_height = unit(3, "cm"),
             title_position = "leftcenter-rot")

pdf("../analysis/gene_set_analysis/hm_KEGG_collapsed.pdf", width = unit(w*0.45, "cm"))
draw(ht_hor)
draw(lgd, x = unit(w*0.45, "cm"), y = unit(0.8, "cm"), just = c("left", "bottom"))
dev.off()
```

```
## pdf
##   2
```

SPI2 analysis:

Here I perform a SPI2 analysis as described in the manuscript.

First, I read in the dataframes:

```
topDEgenes <- c(rownames(pttt[pttt$p_value_FDR<0.001 & abs(pttt$logFC)>1,])[1:10],
                rownames(ptscrt[ptscrt$p_value_FDR<0.001 &
                                           abs(ptscrt$logFC)>1,])[1:10],
                rownames(ttt[ttt$p_value_FDR<0.001 &
                                           abs(ttt$logFC)>1,])[1:10],
                rownames(pttr[pttr$p_value_FDR<0.001 & abs(pttr$logFC)>1,])[1:10],
                rownames(ptscrr[ptscrr$p_value_FDR<0.001 &
                                           abs(ptscrr$logFC)>1,])[1:10],
                rownames(ttr[ttr$p_value_FDR<0.001 &
                                           abs(ttr$logFC)>1,])[1:10],
                rownames(pttk[pttk$p_value_FDR<0.001 & abs(pttk$logFC)>1,])[1:10],
                rownames(ptscrk[ptscrk$p_value_FDR<0.001 &
                                           abs(ptscrk$logFC)>1,])[1:10],
                rownames(ttk[ttk$p_value_FDR<0.001 &
                                           abs(ttk$logFC)>1,])[1:10],
                "SL1344_1133", "SL1344_1134")
topDEgenes <- unique(topDEgenes[!is.na(topDEgenes)])
write_delim(data.frame(topDEgenes), "../data/PhoPQ_analysis_salcom/upgenes.txt"," ")

df_spi2 <- read.delim("../data/PhoPQ_analysis_salcom/salcom_query_degenes.txt", header = T, sep="\t")
df_phopq <- read.delim("../data/PhoPQ_analysis_salcom/salcom_query.phopq.txt", header = T, sep="\t")

df_salcom_spi2 <- data.frame(Wildtype = df_spi2$WT.MEP, SPI2 = df_spi2$WT.InSPI2,
                        PhoPQ_ko = df_spi2$X.Delta.phoP.Q.InSPI2, row.names = df_spi2$SL1344.Locus.ID)

df_salcom_phopq <- data.frame(Wildtype = df_phopq$WT.MEP, SPI2 = df_phopq$WT.InSPI2,
                        PhoPQ_ko = df_phopq$X.Delta.phoP.Q.InSPI2, row.names = df_phopq$SL1344.Locus.ID

tpm_salcom_spi2 <- data.frame(sapply(df_salcom_spi2, function(x) as.integer(gsub(",","", x))),
                              row.names = rownames(df_salcom_spi2))
tpm_salcom_phopq <- data.frame(sapply(df_salcom_phopq, function(x) as.integer(gsub(",","", x))),
                              row.names = rownames(df_salcom_phopq))

tpm_salcom_spi2 <- tpm_salcom_spi2[order(tpm_salcom_spi2$SPI2),]
tpm_salcom_phopq <- tpm_salcom_phopq[order(tpm_salcom_phopq$SPI2),]
tpm_salcom_spi2 <- tpm_salcom_spi2[!(rownames(tpm_salcom_spi2)=="SL1344_1325"),]

tpm_salcom_spi2 <- tpm_salcom_spi2[!(rownames(tpm_salcom_spi2) %in% rownames(tpm_salcom_phopq)),]

tpm_salcom_spi2$condition <- "DE genes"
tpm_salcom_phopq$condition <- "PhoPQ"



rownames(tpm_salcom_phopq) <- gsub(".*MgrR", "MgrR", rownames(tpm_salcom_phopq))
tpm_salcom <- data.frame(rbind(tpm_salcom_phopq, tpm_salcom_spi2), row.names = c(rownames(tpm_salcom_pho
                                                                                  rownames(tpm_salcom_spi

str(tpm_salcom)
```

```
## 'data.frame':    52 obs. of  4 variables:
##  $ Wildtype : int  6 5 2 6 1 10 127 5 167 267 ...
##  $ SPI2     : int  5 20 33 70 91 146 274 389 389 429 ...
##  $ PhoPQ_ko : int  3 3 1 6 0 3 0 38 71 2 ...
##  $ condition: chr  "PhoPQ" "PhoPQ" "PhoPQ" "PhoPQ" ...
```

Get all genes which are included in both datasets:

```
all_spi2_genes <- rownames(tpm_salcom)[rownames(tpm_salcom) %in% rownames(y)]
str(all_spi2_genes)
```

```
##  chr [1:49] "SL1344_4387" "SL1344_1033" "SL1344_2363" "SL1344_1530" ...
```

```
tpm_salcom <- tpm_salcom[all_spi2_genes,]

prefname <- ifelse(all_spi2_genes %in% pnames$V2 ,pnames[all_spi2_genes,]$V1, "" )
prefname <- ifelse(isUnique(prefname), prefname, "")
prefname_spi2 <- ifelse(prefname != "", prefname, all_spi2_genes)

rownames(tpm_salcom) <- prefname_spi2
```

Now we have to make heatmaps with log foldchanges for all samples:

```
# make list of all results, get logchange table:
reslist <- list(KFF = res_KFF,RXR = res_RXR, TAT = res_TAT)
logchanges_spi2 <- data.frame(lapply(reslist, function(l) {
  logfc <- data.frame( sapply(names(l), function(r) {
    l[[r]]$table[all_spi2_genes,1]
  }), row.names = all_spi2_genes )
}))

colnames(logchanges_spi2) <- c("KFF-acpP ", "KFF-acpP-scrambled ", "KFF-only ","RXR-acpP ",
                               "RXR-acpP-scrambled ", "RXR-only ",
                               "TAT-acpP ", "TAT-acpP-scrambled ", "TAT-only")


# get mean cpm values of all conditions:
countscpm <- cpm(y)[all_spi2_genes,]

spi2_cpm <- sapply(levels(test), function(t) {
  rowMeans(countscpm[,t == test])
})
spi2_cpm <- spi2_cpm[,c(2,3,1,5,6,4,8,9,7,10)]


pvalues_spi2 <- data.frame(lapply(reslist, function(l) {
  logfc <- data.frame( sapply(names(l), function(r) {
    l[[r]]$table[all_spi2_genes,]$p_value_FDR<0.001 & abs(l[[r]]$table[all_spi2_genes,]$logFC)>1
  }), row.names = all_spi2_genes )
}))

pvalues_spi2 <-sapply(pvalues_spi2, function(x) ifelse(x , x <- "*",x<-""))
```

Plot Heatmap:

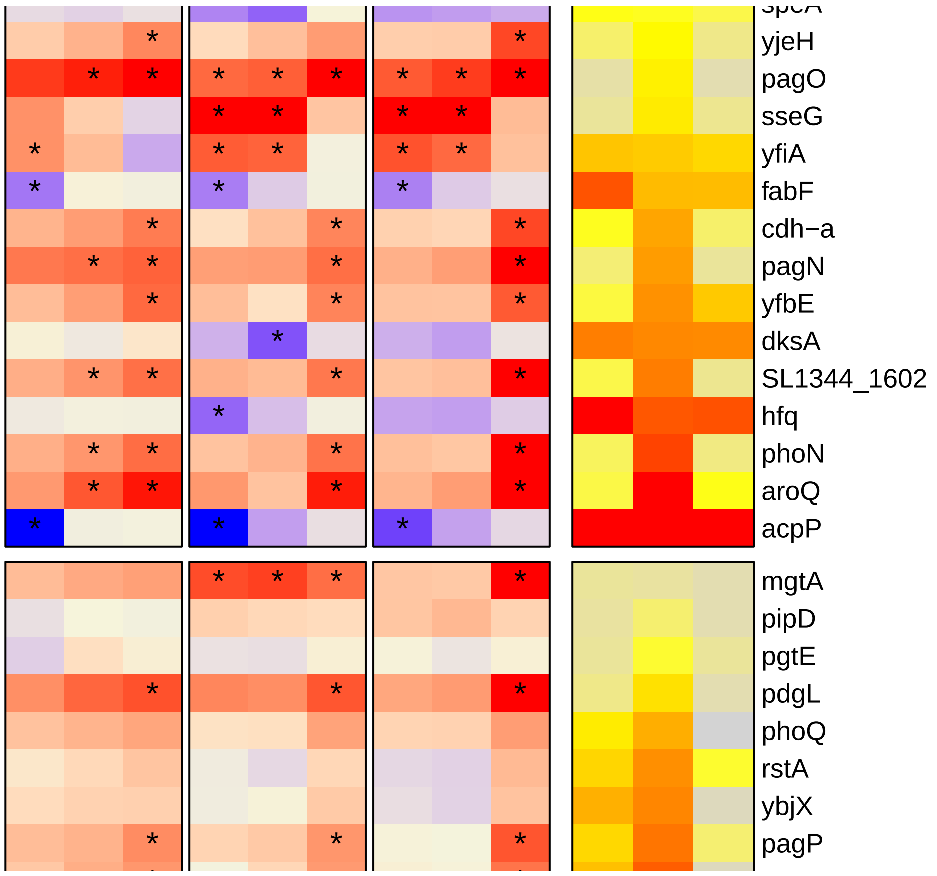
```
h <- dim(logchanges_spi2)[1] # width of plot (nr pws)

col_fun = colorRamp2(c(-2, 0, 2), c("blue", "beige", "red"))
ht_vert <- Heatmap(logchanges_spi2, cluster_rows = F, cluster_columns = F,
               name = "SPI2 analysis", col = col_fun,
               show_heatmap_legend = F,
               row_title_side = "left", row_title_rot = 0,
               border = TRUE,
               cell_fun = function(j, i, x, y, width, height, fill) {
                 grid.text(sprintf("%.1s", pvalues_spi2[i, j]), x, y)
               },
               column_names_gp = gpar(fontsize = 11),
               row_names_gp = gpar(fontsize = 10),
               column_split = factor(ord, levels = lev),
               row_split = factor(tpm_salcom$condition),
               row_gap = unit(0.2, "cm"),
               width = unit(9*0.8, "cm"), height = unit(h/2, "cm"),
               column_names_rot = 45)
```

```
## Warning: The input is a data frame, convert it to the matrix.
```

```
col_col_fun = colorRamp2(c(0, 2,3), c("lightgrey", "yellow", "red"))
ht_colgan <- Heatmap(log10(tpm_salcom[,1:3]), cluster_rows = F, cluster_columns = F,
               name = "Colgan et al., 2016", col = col_col_fun,
               show_heatmap_legend = F,
               row_title_side = "right", row_title_rot = 0,
               border = TRUE,
               row_split = factor(tpm_salcom$condition),
               column_names_gp = gpar(fontsize = 11),
               row_names_gp = gpar(fontsize = 10),
               row_gap = unit(0.2, "cm"),
               width = unit(3*0.8, "cm"), height = unit(h/2, "cm"),
               column_names_rot = 45)
```

```
## Warning: The input is a data frame, convert it to the matrix.
```

```
order = factor(c(rep("KFF", 3), rep("RXR", 3), rep("TAT", 3), "Control"))



ht_list =  ht_vert + ht_colgan
ht_list
```



```r
lgd = Legend(col_fun = col_fun, title = expression("Log"[2]*" FC"),
             at = c(-2, 0, 2), legend_width = unit(4, "cm"),
             title_gp  = gpar(fontsize(14)),
             labels = c("-2", "  0", "  2"), legend_height = unit(3, "cm"),
             title_position = "leftcenter-rot")
lgd_col = Legend(col_fun = col_col_fun, title = expression("TPM"),
             at = c(0, 1, 2, 3), labels = c(0,10,100,1000), legend_width = unit(3, "cm"),
             title_gp  = gpar(fontsize(14)),
             legend_height = unit(3, "cm"),
             title_position = "leftcenter-rot")


pdf("../analysis/PhoPQ_Salcom/hm_salcom_phopq_spi2.pdf", width = 10, height = 15)
draw(ht_list, ht_gap = unit(0.5, "cm"))
draw(lgd, x = unit(4, "cm"), y = unit(18, "cm"), just = c("left", "bottom"))
draw(lgd_col, x = unit(21, "cm"), y = unit(18, "cm"), just = c("left", "bottom"))
dev.off()
```

```
## pdf
##   2
```

save csv of all raw counts for supplementary material:

```r
counts <- y$counts
test <- c("Water1", "KFF_acpP1", "KFF_acpP_scrambled1", "KFF1", "RXR_acpP1", "RXR_acpP_scrambled1",
          "RXR1", "TAT_acpP1", "TAT_acpP_scrambled1", "TAT1",
          "Water2", "KFF_acpP2", "KFF_acpP_scrambled2", "KFF2", "RXR_acpP2", "RXR_acpP_scrambled2",
          "RXR2", "TAT_acpP2", "TAT_acpP_scrambled2", "TAT2",
          "Water3", "KFF_acpP3", "KFF_acpP_scrambled3", "KFF3", "RXR_acpP3", "RXR_acpP_scrambled3",
          "RXR3", "TAT_acpP3", "TAT_acpP_scrambled3", "TAT3")
colnames(counts) <- test

prefname <- ifelse(rownames(counts) %in% pnames$V2 ,pnames[rownames(counts),]$V1, "" )
prefname <- ifelse(isUnique(prefname), prefname, "")
gene_name <- ifelse(prefname != "", prefname, rownames(counts))

locus_tag <- rownames(counts)

counts_raw <- cbind(locus_tag, gene_name, counts)

write.csv(counts_raw, "../analysis/analysis_complete/supp_tables/raw_counts.csv")

for (i in all_res) {
  for (l in names(i)){
    tab <- i[[l]]$table[order(i[[l]]$table$FDR),]
    prefname <- ifelse(rownames(tab) %in% pnames$V2 ,pnames[rownames(tab),]$V1, "" )
    prefname <- ifelse(isUnique(prefname), prefname, "")
    genename <- ifelse(prefname != "", prefname, rownames(tab))
    tabs <- cbind(genename,tab)
    write.csv(tabs, paste("../analysis/analysis_complete/supp_tables/",l,"_vs_water",".csv", sep = ""))
  }
}


for (l in names(list_kegg_fry)){
  tabs <- list_kegg_fry[[l]]
  write.csv(tabs, paste("../analysis/analysis_complete/supp_tables/",l,"_vs_water",".csv", sep = ""))
}
```

Packages used:

```
sessionInfo()
```

```
## R version 4.0.3 (2020-10-10)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 20.04.1 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/atlas/libblas.so.3.10.3
## LAPACK: /usr/lib/x86_64-linux-gnu/atlas/liblapack.so.3.10.3
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=de_DE.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=de_DE.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=de_DE.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=de_DE.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
##  [1] grid      stats4    parallel  stats     graphics  grDevices utils
##  [8] datasets  methods   base
```

```
##
## other attached packages:
##  [1] KEGGREST_1.28.0             forcats_0.5.0
##  [3] stringr_1.4.0               purrr_0.3.4
##  [5] readr_1.4.0                 tidyr_1.1.2
##  [7] tibble_3.0.4                tidyverse_1.3.0
##  [9] eulerr_6.1.0                VennDiagram_1.6.20
## [11] futile.logger_1.4.3         ComplexHeatmap_2.4.3
## [13] svglite_1.2.3.2             ggrepel_0.8.2
## [15] gplots_3.1.0                oligo_1.52.1
## [17] oligoClasses_1.50.4         RColorBrewer_1.1-2
## [19] RUVSeq_1.22.0               EDASeq_2.22.0
## [21] ShortRead_1.46.0            GenomicAlignments_1.24.0
## [23] SummarizedExperiment_1.18.2 DelayedArray_0.14.1
## [25] matrixStats_0.57.0          Rsamtools_2.4.0
## [27] GenomicRanges_1.40.0        GenomeInfoDb_1.24.2
## [29] Biostrings_2.56.0           XVector_0.28.0
## [31] IRanges_2.22.2              S4Vectors_0.26.1
## [33] BiocParallel_1.22.0         Biobase_2.48.0
## [35] BiocGenerics_0.34.0         ggplot2_3.3.2
## [37] dplyr_1.0.2                 circlize_0.4.11
## [39] edgeR_3.30.3                limma_3.44.3
##
## loaded via a namespace (and not attached):
##   [1] readxl_1.3.1           backports_1.2.0        aroma.light_3.18.0
##   [4] BiocFileCache_1.12.1   systemfonts_0.3.2      polylabelr_0.2.0
##   [7] splines_4.0.3          digest_0.6.27          foreach_1.5.1
##  [10] htmltools_0.5.0        fansi_0.4.1            magrittr_2.0.1
##  [13] memoise_1.1.0          cluster_2.1.0          annotate_1.66.0
##  [16] modelr_0.1.8           R.utils_2.10.1         askpass_1.1
##  [19] prettyunits_1.1.1      jpeg_0.1-8.1           colorspace_2.0-0
##  [22] rvest_0.3.6            blob_1.2.1             rappdirs_0.3.1
##  [25] haven_2.3.1            xfun_0.19              jsonlite_1.7.1
##  [28] crayon_1.3.4           RCurl_1.98-1.2         genefilter_1.70.0
##  [31] survival_3.2-7         iterators_1.0.13       glue_1.4.2
##  [34] polyclip_1.10-0        gtable_0.3.0           zlibbioc_1.34.0
##  [37] GetoptLong_1.0.4       shape_1.4.5            scales_1.1.1
##  [40] DESeq_1.39.0           futile.options_1.0.1   DBI_1.1.0
##  [43] Rcpp_1.0.5             xtable_1.8-4           progress_1.2.2
##  [46] clue_0.3-57            bit_4.0.4              preprocessCore_1.50.0
##  [49] httr_1.4.2             ellipsis_0.3.1         ff_4.0.4
##  [52] farver_2.0.3           pkgconfig_2.0.3        XML_3.99-0.5
##  [55] R.methodsS3_1.8.1      dbplyr_2.0.0           locfit_1.5-9.4
##  [58] tidyselect_1.1.0       rlang_0.4.9            AnnotationDbi_1.50.3
##  [61] cellranger_1.1.0       munsell_0.5.0          tools_4.0.3
##  [64] cli_2.2.0              generics_0.1.0         RSQLite_2.2.1
##  [67] broom_0.7.2            evaluate_0.14          yaml_2.2.1
##  [70] fs_1.5.0               knitr_1.30             bit64_4.0.5
##  [73] caTools_1.18.0         formatR_1.7            R.oo_1.24.0
##  [76] xml2_1.3.2             biomaRt_2.44.4         rstudioapi_0.13
##  [79] compiler_4.0.3         curl_4.3               png_0.1-7
##  [82] affyio_1.58.0          reprex_0.3.0           statmod_1.4.35
##  [85] geneplotter_1.66.0     stringi_1.5.3          GenomicFeatures_1.40.1
##  [88] gdtools_0.2.2          lattice_0.20-41        Matrix_1.2-18
##  [91] vctrs_0.3.5           pillar_1.4.7          lifecycle_0.2.0
##  [94] BiocManager_1.30.10   GlobalOptions_0.1.2   bitops_1.0-6
##  [97] rtracklayer_1.48.0    R6_2.5.0              latticeExtra_0.6-29
## [100] hwriter_1.3.2         KernSmooth_2.23-18    affxparser_1.60.0
## [103] codetools_0.2-18      lambda.r_1.2.4        MASS_7.3-53
## [106] gtools_3.8.2          assertthat_0.2.1      openssl_1.4.3
## [109] rjson_0.2.20          withr_2.3.0           GenomeInfoDbData_1.2.3
## [112] hms_0.5.3             rmarkdown_2.5         lubridate_1.7.9.2
```