# ELT-2 Regulated Genes

Rtpw

2/12/2020

## Next steps

HIGH - perform GO on up and down regulated genes LOW - elt-2 chip or promoter motifs of up and down regulated genes

## Done steps

- Do Z score of row normalization, divide by the standard deviation

## Improvements

Align RNA seq data to ce11 genome with more recent annotation.

## Libraries

```r
library(biomaRt)
library(DESeq2)
```

```
## Loading required package: S4Vectors

## Loading required package: stats4

## Loading required package: BiocGenerics

## Loading required package: parallel

##
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB

## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs
## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, basename, cbind, colnames,
##     dirname, do.call, duplicated, eval, evalq, Filter, Find, get, grep,
##     grepl, intersect, is.unsorted, lapply, Map, mapply, match, mget,
##     order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
##     rbind, Reduce, rownames, sapply, setdiff, sort, table, tapply,
##     union, unique, unsplit, which, which.max, which.min

##
## Attaching package: 'S4Vectors'

## The following object is masked from 'package:base':
##
##     expand.grid

## Loading required package: IRanges

## Loading required package: GenomicRanges

## Loading required package: GenomeInfoDb

## Loading required package: SummarizedExperiment

## Loading required package: Biobase

## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.

## Loading required package: DelayedArray

## Loading required package: matrixStats

##
## Attaching package: 'matrixStats'

## The following objects are masked from 'package:Biobase':
##
##     anyMissing, rowMedians

## Loading required package: BiocParallel

##
## Attaching package: 'DelayedArray'

## The following objects are masked from 'package:matrixStats':
##
##     colMaxs, colMins, colRanges, rowMaxs, rowMins, rowRanges

## The following objects are masked from 'package:base':
##
##     aperm, apply, rowsum
```

```r
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.0     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.5
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## -- Conflicts ---------------------------------------------------- tidyverse_conflicts() --
## x dplyr::collapse()   masks IRanges::collapse()
## x dplyr::combine()    masks Biobase::combine(), BiocGenerics::combine()
## x dplyr::count()      masks matrixStats::count()
## x dplyr::desc()       masks IRanges::desc()
## x tidyr::expand()     masks S4Vectors::expand()
## x dplyr::filter()     masks stats::filter()
## x dplyr::first()      masks S4Vectors::first()
## x dplyr::lag()        masks stats::lag()
## x ggplot2::Position() masks BiocGenerics::Position(), base::Position()
## x purrr::reduce()     masks GenomicRanges::reduce(), IRanges::reduce()
## x dplyr::rename()     masks S4Vectors::rename()
## x dplyr::select()     masks biomaRt::select()
## x purrr::simplify()   masks DelayedArray::simplify()
## x dplyr::slice()      masks IRanges::slice()
```

```
library(pheatmap)
library(readxl)
library(matrixStats)
```

Source required functions.

```
source("./RWC23_Functions.R")
```

## Differentail Expression

Load data

```
RNAcounts <- read.csv("./01_input/Table_S1_Raw_Read_Counts.csv", header=TRUE, row.names = 1)
```

This count file contains more samples than what I want to analyze. Subset the columns to just have `wt_sorted_*` and `elt2D_sorted_*`. Also select columns that correspond to ce11 genome assembly, since this is the genome used for the ChIP-seq analysis.

```
cts <- RNAcounts %>% select(wt_sorted_1, wt_sorted_2, wt_sorted_3, wt_sorted_4, elt2D_sorted_1, elt2D_se
head(cts)
```

```
##                wt_sorted_1 wt_sorted_2 wt_sorted_3 wt_sorted_4 elt2D_sorted_1
## WBGene00000001         532         462         458         525            546
## WBGene00000002         192         165         185         195            169
## WBGene00000003         577         425         649         694            371
## WBGene00000004        2111        1794        2131        1999           1158
## WBGene00000005          11           8          13           6              9
## WBGene00000007          71          82          69          92             19
##                elt2D_sorted_2 elt2D_sorted_3 elt2D_sorted_4 elt2Delt7D_sorted_1
## WBGene00000001            919            575            661                 799
## WBGene00000002            226            157            147                 291
## WBGene00000003            557            405            429                 510
## WBGene00000004           1832           1233           1288                1481
## WBGene00000005             11              8             10                   3
## WBGene00000007             36             15             18                  22
##                elt2Delt7D_sorted_2 elt2Delt7D_sorted_3
## WBGene00000001                 675                 482
## WBGene00000002                  271                  194
## WBGene00000003                  489                  425
## WBGene00000004                 1304                 1347
## WBGene00000005                    7                    1
## WBGene00000007                   22                   13
```

make `coldata`

```
coldata <- data.frame(condition = c("wt", "wt", "wt", "wt", "elt2D", "elt2D", "elt2D", "elt2D", "elt2De
coldata
```

```
##                        condition
## wt_sorted_1                   wt
## wt_sorted_2                   wt
## wt_sorted_3                   wt
## wt_sorted_4                   wt
## elt2D_sorted_1             elt2D
## elt2D_sorted_2             elt2D
## elt2D_sorted_3             elt2D
## elt2D_sorted_4             elt2D
## elt2Delt7D_sorted_1 elt2Delt7D
## elt2Delt7D_sorted_2 elt2Delt7D
## elt2Delt7D_sorted_3 elt2Delt7D
```

Check that column matrix and coldata match

```
all(rownames(coldata) == colnames(cts))
```

```
## [1] TRUE
```

Generate DESeqDataSet

```
dds <- DESeqDataSetFromMatrix(countData = cts, colData = coldata, design = ~ condition)

# add gene names
moreFeatures <- data.frame(gene_name = RNAcounts$gene_id_val, sequence_id = RNAcounts$sequence_id_list)
mcols(dds) <- DataFrame(mcols(dds), moreFeatures)
mcols(dds)
```

```
## DataFrame with 16708 rows and 2 columns
##                 gene_name sequence_id
##                  <factor>    <factor>
## WBGene00000001     aap-1  Y110A7A.10
## WBGene00000002     aat-1     F27C8.1
## WBGene00000003     aat-2     F07C3.7
## WBGene00000004     aat-3     F52H2.2
## WBGene00000005     aat-4   T13A10.10
## ...                  ...         ...
## WBGene00043705        NA          NA
## WBGene00015013        NA          NA
## WBGene00008743        NA          NA
## WBGene00235114        NA          NA
## WBGene00077643        NA          NA
```

Tell DESeq which samples are "control" and which are "control" vs "treatment". This sets up the fold change comparison manually rather than letting the alphabetical determination of factor levels.

with this step: logfoldchange(elt2D/wt)

```
dds$condition <- factor(dds$condition, levels = c("wt", "elt2D", "elt2Delt7D"))
```

Perform differential expression analysis

```
dds <- DESeq(dds)
```

```
## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing
```

```
res <- results(dds)


# Convert res to dataframe
res.df <- as.data.frame(res)

# Export the results table
#write.csv(res.df, file = "./02_DESeq2/200218_L1_wt_vs_elt2D_results.csv")

# Print results table information
head(res)
```

```
## log2 fold change (MLE): condition elt2Delt7D vs wt
## Wald test p-value: condition elt2Delt7D vs wt
## DataFrame with 6 rows and 6 columns
##                         baseMean     log2FoldChange             lfcSE
##                        <numeric>          <numeric>         <numeric>
## WBGene00000001 591.515264995531  0.374766904987348 0.100832250987597
## WBGene00000002 196.941946891564  0.431645564160724 0.122403401331081
## WBGene00000003 499.031409070873 -0.309270596061639 0.142658833176119
## WBGene00000004 1597.07886131967 -0.542369069007949 0.105519396720516
## WBGene00000005 7.82653928628189  -1.41145497345249 0.629860204886353
## WBGene00000007 41.2854757245755   -2.0673345543798 0.274722012968266
##                             stat               pvalue                 padj
##                        <numeric>            <numeric>            <numeric>
## WBGene00000001  3.71673647386337 0.000201812748989876 0.000660611216589531
## WBGene00000002  3.52641805265847 0.000421221497079934  0.00129548701082756
## WBGene00000003 -2.16790358631232   0.0301660229459714   0.0605533480446432
## WBGene00000004 -5.13999402824958 2.74747200542808e-07  1.4149353669423e-06
## WBGene00000005 -2.24090196920309   0.0250324256740502   0.0515377102412202
## WBGene00000007 -7.52518712295034 5.26448735432883e-14 5.66900376974855e-13
```

```
summary(res)
```

```
##
## out of 16707 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 4609, 28%
## LFC < 0 (down)     : 4404, 26%
## outliers [1]       : 16, 0.096%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

Perform vst and rlog transformation of read counts

```
vsd <- vst(dds)
rld <- rlog(dds)
```

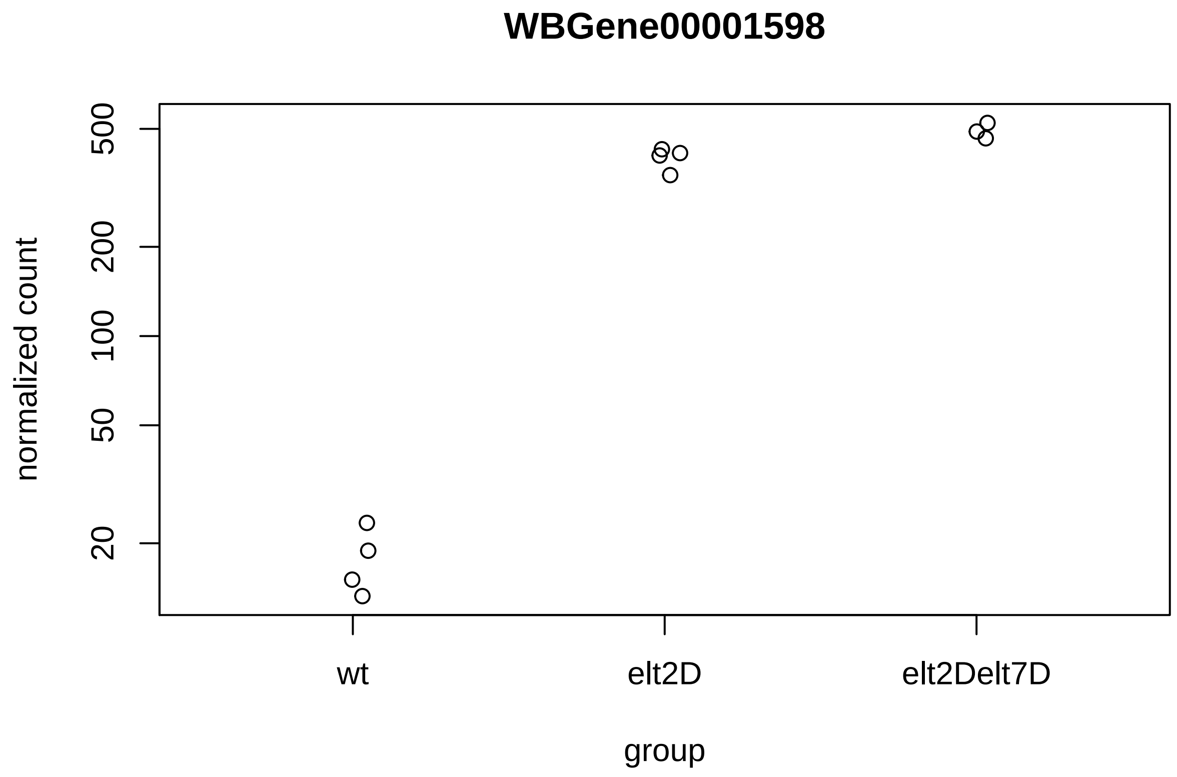
## Explore Differential Expression

Determine if glh-1 is upregulated in elt-2(-)
glh-1 = WBGene00001598
glh-1 is a germline specific gene
It is upregulated

```
res.df[rownames(res.df) == "WBGene00001598",]
```

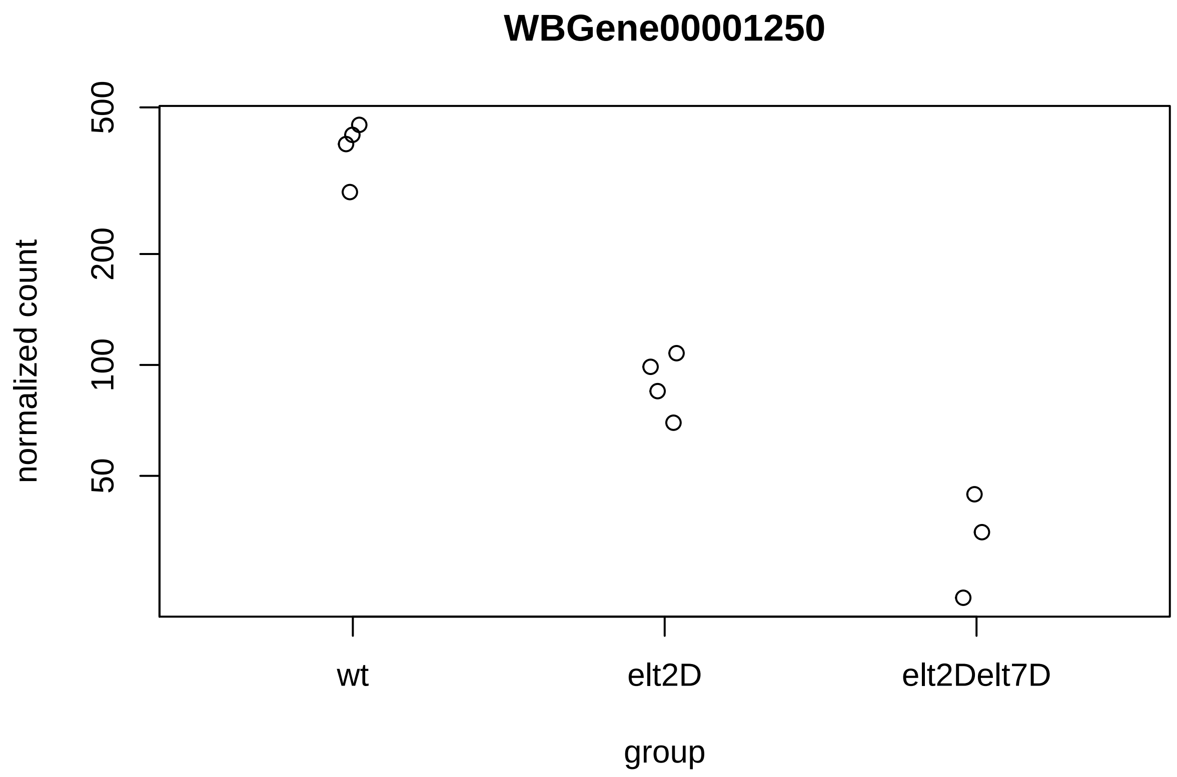
```
##                 baseMean log2FoldChange     lfcSE     stat        pvalue
## WBGene00001598 285.4151       4.843145 0.2083577 23.24438 1.621405e-119
##                          padj
## WBGene00001598 5.106203e-117
```

```
plotCounts(dds, gene = "WBGene00001598")
```



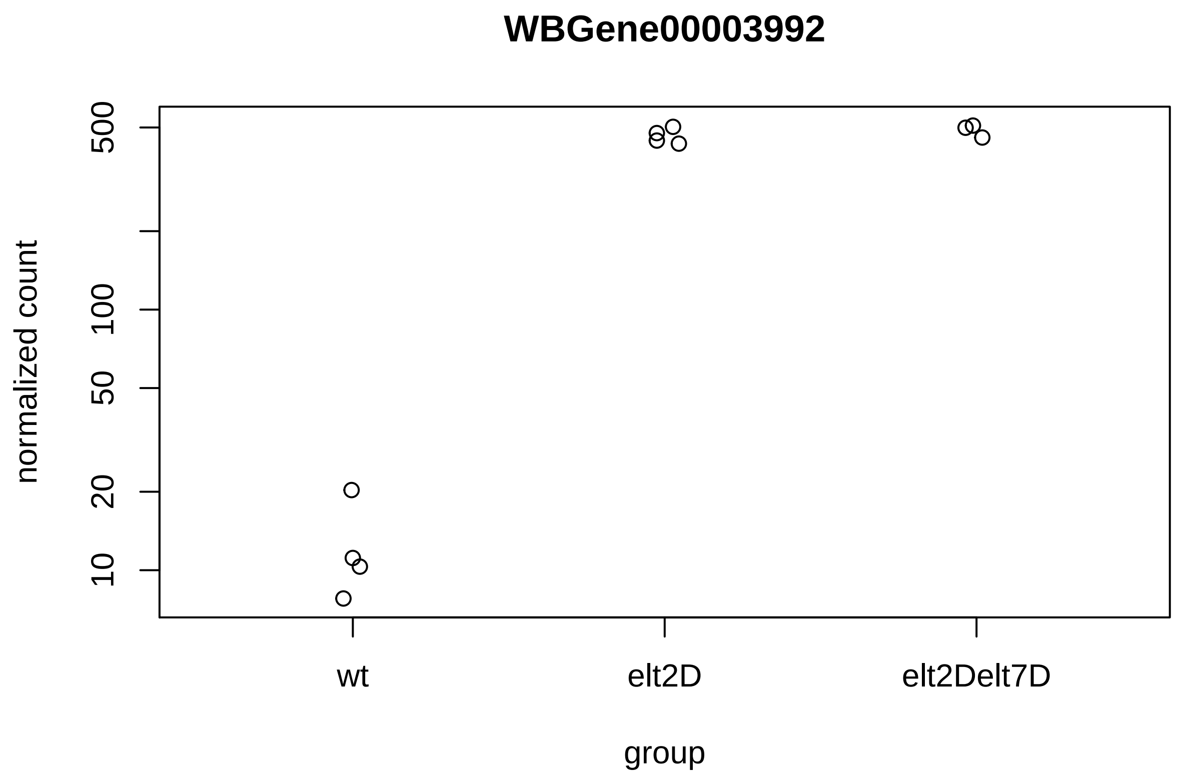
See if elt-2 is depleted
It is depleted

```
plotCounts(dds, gene = "WBGene00001250")
```



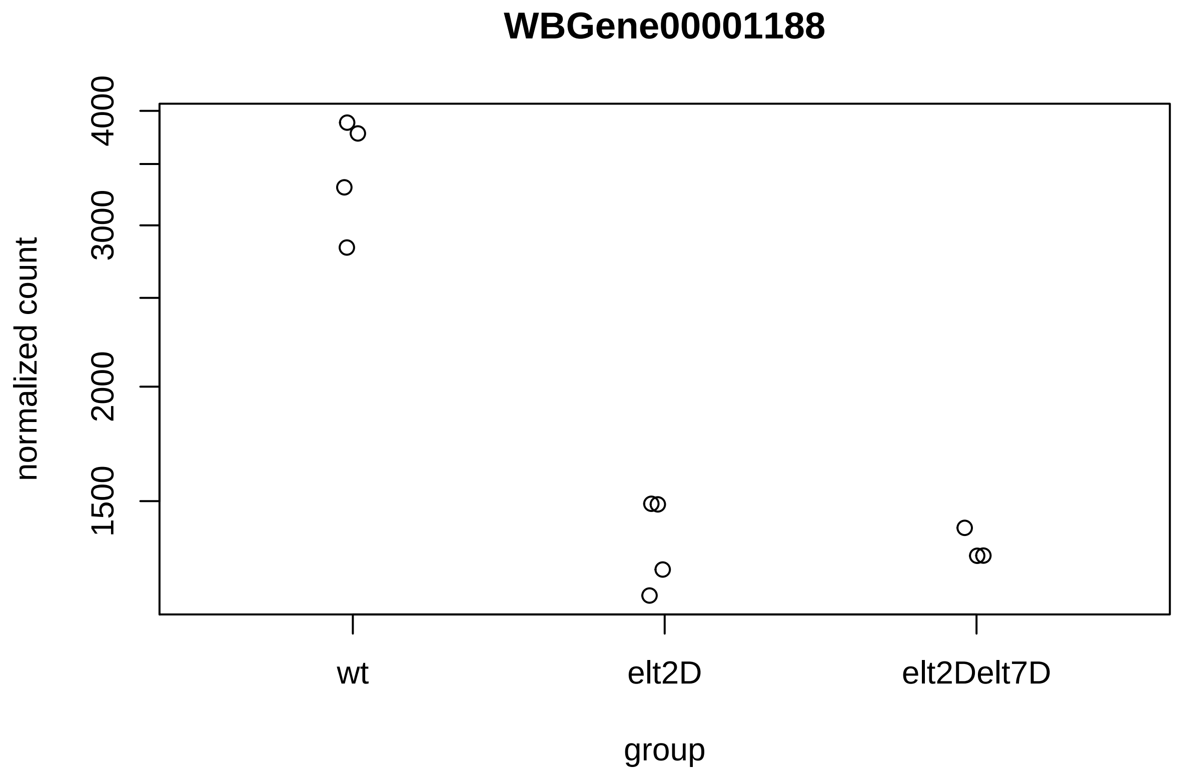
See if pgl-1 is enriched
pgl-1 = WBGene00003992
Looks like it is enriched.

```r
plotCounts(dds, gene = "WBGene00003992")
```



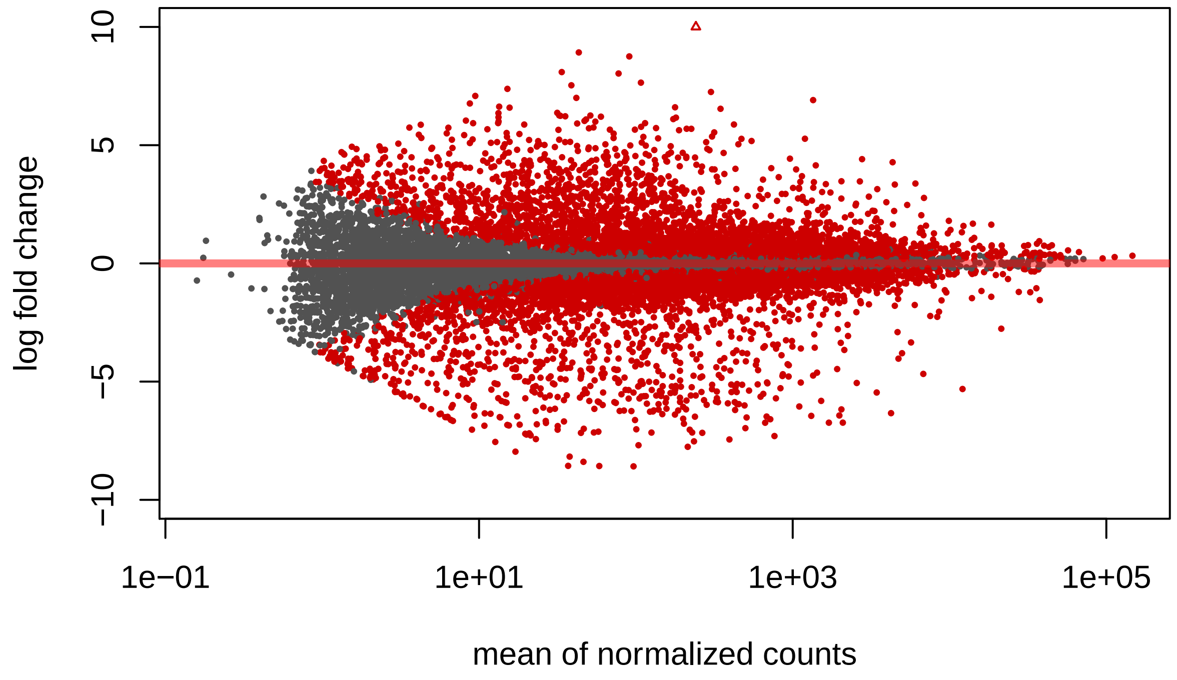
see if egl-20 (ligand of Wnt pathway) is expressed
Also is depleted

```r
plotCounts(dds, gene = "WBGene00001188")
```



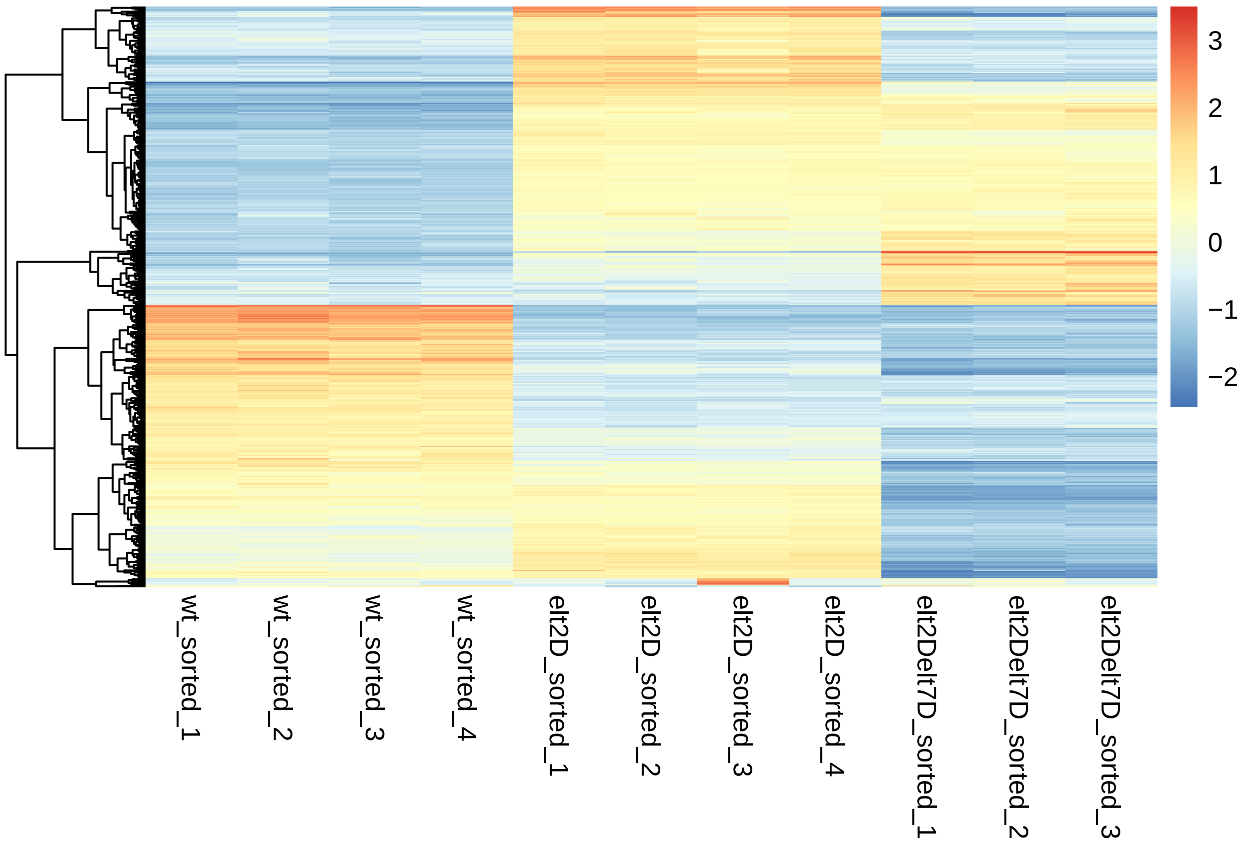
Make an MA plot for all the data.

```r
plotMA(res, ylim = c(-10, 10))
```



Make a heatmap of differentially expressed genes.
Use variance of rlog transformed read counts to filter the data set for genes that are actually changing.

```
select <- rowVars(assay(rld)) > 0.5

rowNormalized <- assay(rld)-rowMeans(assay(rld))

pheatmap(rowNormalized[select, ], cluster_cols = FALSE, cluster_rows = TRUE, show_rownames = FALSE)
```



## Determine ELT-2 regulated TFs

Load in datasets

```
res.df <- read.csv(file = "./02_DESeq2/200218_L1_wt_vs_elt2D_results.csv", header = TRUE, sep = ",", ro

res.df <- rownames_to_column(res.df, var = "WBGeneID")

wTF3.0 <- read.delim("./TF3-0_namesonly.txt", header = TRUE, sep = "\t") %>% select(Sequence.name, Publ
```

Subset elt-2(-) differentially expressed genes for transcription factors with wTF3.0 dataset.

```
elt2_responding_TF <- merge(wTF3.0, res.df, by.x = "WBGeneID", by.y = "WBGeneID")
dim(elt2_responding_TF)
```

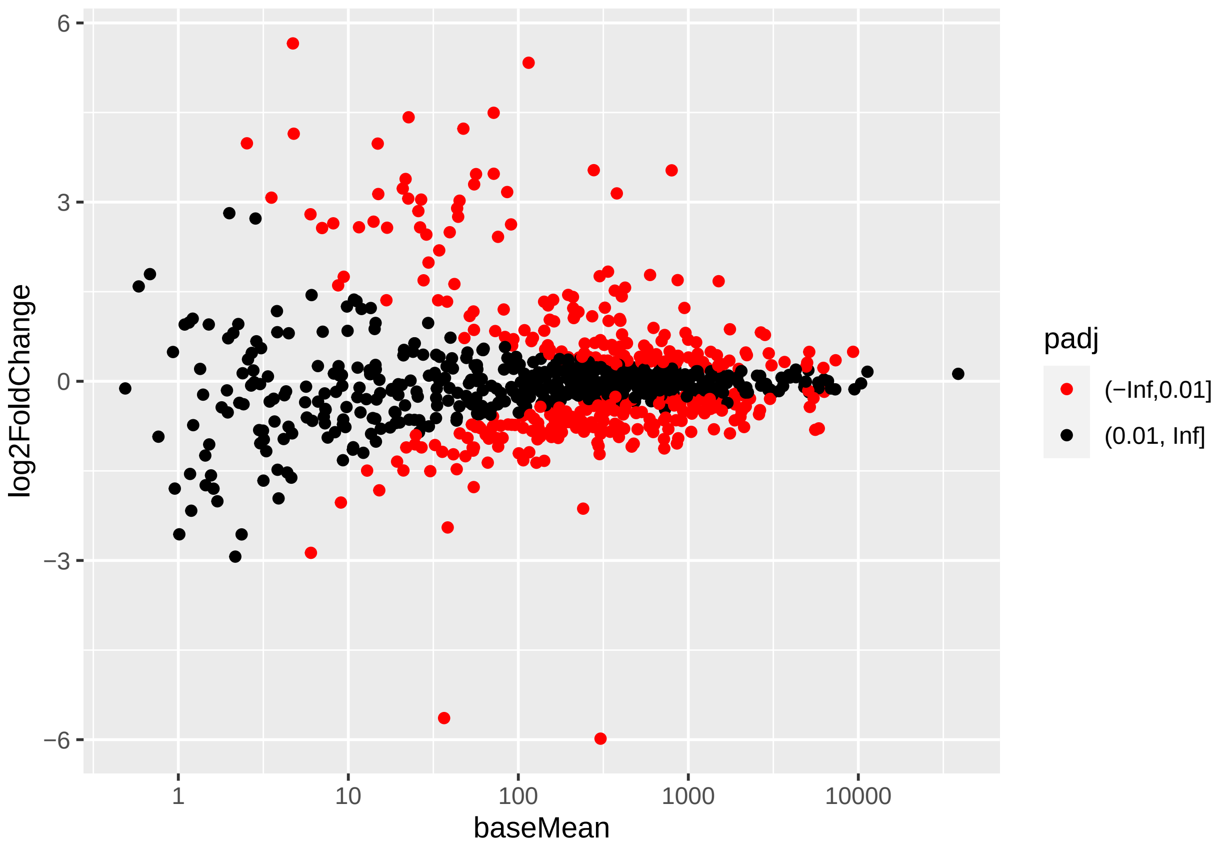
```
## [1] 854  10
```

```
head(elt2_responding_TF)
```

```
##          WBGeneID Sequence.name Public_name              DBD   baseMean
## 1 WBGene00000095       C25A1.11       aha-1             bHLH 1118.11640
## 2 WBGene00000096        C41G7.5       ahr-1             bHLH  302.50133
## 3 WBGene00000220        K08F8.2       atf-2 bZIP - 2 domains   19.68263
## 4 WBGene00000221       T04C10.4       atf-5             bZIP 2306.33631
## 5 WBGene00000222        F45E6.2       atf-6             bZIP  699.86423
## 6 WBGene00000223        C07G2.2       atf-7             bZIP 4850.15422
##   log2FoldChange      lfcSE       stat       pvalue         padj
## 1    -0.26323378 0.06830782 -3.8536404 1.163745e-04 3.407293e-04
## 2    -0.58487711 0.11438142 -5.1133927 3.164235e-07 1.276946e-06
## 3    -0.04881475 0.35069274 -0.1391952 8.892959e-01 9.258111e-01
## 4    -0.28854464 0.07890305 -3.6569519 2.552323e-04 7.061427e-04
## 5     0.07558784 0.07789255  0.9704117 3.318413e-01 4.374642e-01
## 6    -0.04463394 0.05621404 -0.7939999 4.271955e-01 5.337304e-01
```

Make an MA plot of differentially expressed transcription factors.

```
threshold = 0.01
ggplot(data = elt2_responding_TF) +
  geom_point(aes(x = baseMean, y = log2FoldChange, colour = cut(padj, c(-Inf, threshold, +Inf)))) +
  scale_colour_manual(name = "padj", values = c("red", "black")
  ) +
  scale_x_log10()
```

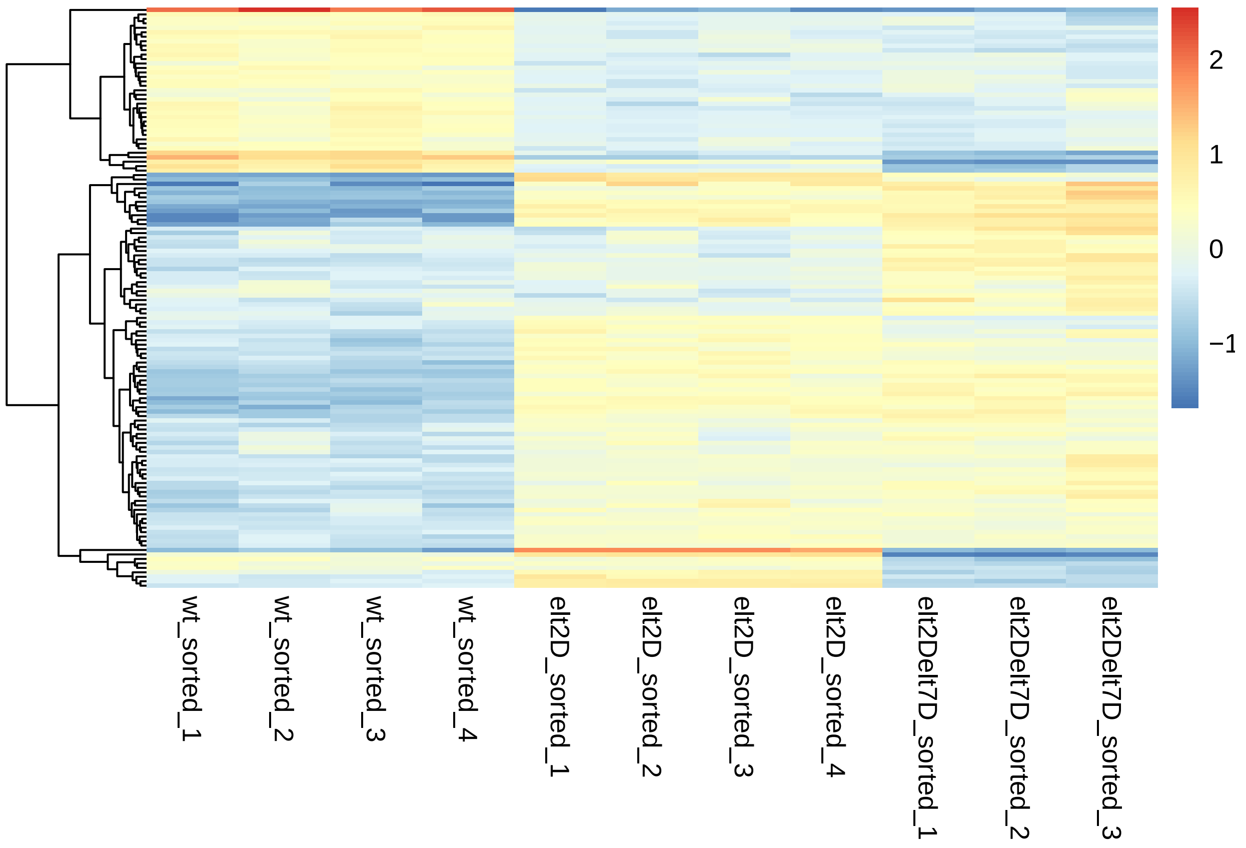


Separate dataset into activated and repressed TFs.

```
elt2_activated_TF <- elt2_responding_TF %>% filter(log2FoldChange <= 0, padj <= threshold)
elt2_repressed_TF <- elt2_responding_TF %>% filter(log2FoldChange >= 0, padj <= threshold)
```

Make heatmap of differentially expressed TFs.

```
nameselect <- rownames(assay(rld)) %in% wTF3.0$WBGeneID

nameMatrix <- assay(rld)[nameselect,]

varselect <- rowVars(nameMatrix) > 0.1

namevarMatrix <- nameMatrix[varselect, ]

namevarRowNormalized <- namevarMatrix - rowMeans(namevarMatrix)

pheatmap(namevarRowNormalized, cluster_cols = FALSE, cluster_rows = TRUE, show_rownames = FALSE, border
```



Replace WBGeneID with gene name in row name.

```
rownames(namevarRowNormalized)
```

```
##   [1] "WBGene00000220" "WBGene00000431" "WBGene00000433" "WBGene00000446"
##   [5] "WBGene00000447" "WBGene00000455" "WBGene00000458" "WBGene00000467"
##   [9] "WBGene00000473" "WBGene00000474" "WBGene00000561" "WBGene00000895"
##  [13] "WBGene00001174" "WBGene00001250" "WBGene00001252" "WBGene00001951"
##  [17] "WBGene00001952" "WBGene00001954" "WBGene00001960" "WBGene00001973"
##  [21] "WBGene00001977" "WBGene00002601" "WBGene00002987" "WBGene00003015"
##  [25] "WBGene00003017" "WBGene00003033" "WBGene00003106" "WBGene00003114"
##  [29] "WBGene00003148" "WBGene00003167" "WBGene00003228" "WBGene00003230"
##  [33] "WBGene00003231" "WBGene00003376" "WBGene00003377" "WBGene00003511"
##  [37] "WBGene00003606" "WBGene00003607" "WBGene00003633" "WBGene00003645"
##  [41] "WBGene00003653" "WBGene00003657" "WBGene00003689" "WBGene00003696"
##  [45] "WBGene00003698" "WBGene00003702" "WBGene00003711" "WBGene00003727"
##  [49] "WBGene00003847" "WBGene00003864" "WBGene00003865" "WBGene00003912"
##  [53] "WBGene00003976" "WBGene00004011" "WBGene00004027" "WBGene00004078"
##  [57] "WBGene00004096" "WBGene00004764" "WBGene00004786" "WBGene00005011"
##  [61] "WBGene00006873" "WBGene00006881" "WBGene00007048" "WBGene00007058"
##  [65] "WBGene00007242" "WBGene00007367" "WBGene00007749" "WBGene00007776"
##  [69] "WBGene00008007" "WBGene00008242" "WBGene00008417" "WBGene00008830"
##  [73] "WBGene00009014" "WBGene00009937" "WBGene00010215" "WBGene00011002"
##  [77] "WBGene00011066" "WBGene00011100" "WBGene00011130" "WBGene00011315"
##  [81] "WBGene00011376" "WBGene00011597" "WBGene00011601" "WBGene00011925"
##  [85] "WBGene00011956" "WBGene00012005" "WBGene00012101" "WBGene00012210"
##  [89] "WBGene00012435" "WBGene00012449" "WBGene00012474" "WBGene00012494"
##  [93] "WBGene00012988" "WBGene00013270" "WBGene00013380" "WBGene00013976"
##  [97] "WBGene00014253" "WBGene00015396" "WBGene00015649" "WBGene00015934"
## [101] "WBGene00016366" "WBGene00016865" "WBGene00016888" "WBGene00016930"
## [105] "WBGene00016997" "WBGene00017482" "WBGene00017651" "WBGene00017687"
## [109] "WBGene00017755" "WBGene00018099" "WBGene00018539" "WBGene00018704"
## [113] "WBGene00019327" "WBGene00019344" "WBGene00019598" "WBGene00019743"
## [117] "WBGene00019751" "WBGene00019878" "WBGene00020015" "WBGene00020555"
## [121] "WBGene00021082" "WBGene00021704" "WBGene00022060" "WBGene00022562"
## [125] "WBGene00007732" "WBGene00003688" "WBGene00011600" "WBGene00016368"
## [129] "WBGene00003648" "WBGene00004157"
```

```
paramart <- useMart("parasite_mart", dataset = "wbps_gene", host = "https://parasite.wormbase.org", por

name2id = getBM(mart = paramart,
                     filter=c("species_id_1010",
                              "biotype"),
                     value=list(species_id_1010="caelegprjna13758",
                                biotype="protein_coding"),
                     attributes = c('external_gene_id',
                                    'wbps_gene_id'))
```

```
## Cache found
```

```
head(name2id)
```

```
##   external_gene_id   wbps_gene_id
## 1            aap-1 WBGene00000001
## 2            aat-1 WBGene00000002
## 3            aat-2 WBGene00000003
## 4            aat-3 WBGene00000004
## 5            aat-4 WBGene00000005
## 6            aat-5 WBGene00000006
```

## ELT-2 Bound and Reuglated Genes

This section will integrate the L1 stage ELT-2 ChIP data analyzed by David.

Load in data.

```
elt2_peaks <- read_excel("./01_input/200331_peaksForBigBed.xlsx")

# Subset for genes bound in the L1 stage
elt2_L1_peaks <- elt2_peaks %>% select(mapped_gene, L1) %>% filter(L1 == 1) %>% select(mapped_gene)
```

Now subset the row normalized set expression set for these genes.

Use the functions to subset and row normalize the matrix.

```r
elt2_bound_matrix <- matrix_select(assay(rld), elt2_L1_peaks$mapped_gene)

elt2bound_rownormMatrix <- row_normalize_matrix_cutoff(
  count_matrix = elt2_bound_matrix,
  variance_cutoff =  0.5
  )

head(elt2bound_rownormMatrix)
```

```
##                 wt_sorted_1 wt_sorted_2 wt_sorted_3 wt_sorted_4 elt2D_sorted_1
## WBGene00000022  -0.8776676  -1.0652937  -0.8943597  -1.0651520      0.9607023
## WBGene00000136   1.5086891   1.4044684   1.5699383   1.4718552     -0.2107053
## WBGene00000172   0.5423177   0.6035933   0.2775824   0.4495869      0.7846391
## WBGene00000212   0.2258180   0.6782376   0.2807804   0.4314604      0.8564958
## WBGene00000214   1.6529634   1.9191847   1.9392496   1.3315259     -0.6143219
## WBGene00000215   2.0334889   2.2653447   2.0248544   1.9586817     -1.3207025
##                 elt2D_sorted_2 elt2D_sorted_3 elt2D_sorted_4 elt2Delt7D_sorted_1
## WBGene00000022       0.7736452     0.94379989      0.7837143          0.02098377
## WBGene00000136      -0.3544595    -0.09764439     -0.2871333         -1.71537290
## WBGene00000172       0.9472956     0.66884185      0.9323174         -1.67042064
## WBGene00000212       0.8710924     0.96642706      0.9353808         -1.72805622
## WBGene00000214      -0.2609884    -0.55458218     -0.2366091         -2.04098622
## WBGene00000215      -1.1089528    -0.83159209     -0.9644632         -1.45702143
##                 elt2Delt7D_sorted_2 elt2Delt7D_sorted_3
## WBGene00000022           0.07090722           0.3487203
## WBGene00000136          -1.64382731          -1.6458084
## WBGene00000172          -1.81812604          -1.7176276
## WBGene00000212          -1.67988406          -1.8377524
## WBGene00000214          -1.81810338          -1.3173325
## WBGene00000215          -1.41765395          -1.1819836
```

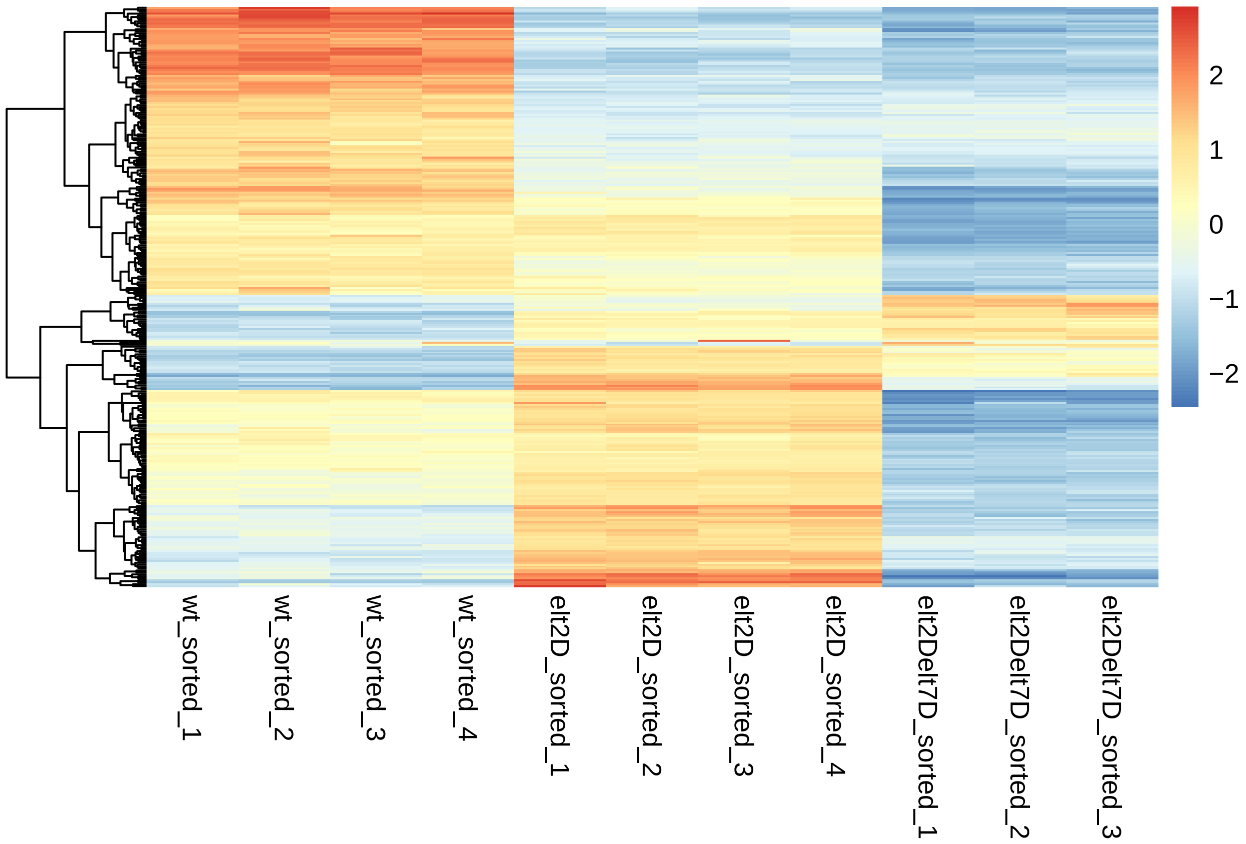
Replace the WBGeneIDs in the row name with gene names.

```r
elt2bound_rowNormGeneNameMatrix <- id2name(elt2bound_rownormMatrix)

head(elt2bound_rowNormGeneNameMatrix)
```

```
##       wt_sorted_1 wt_sorted_2 wt_sorted_3 wt_sorted_4 elt2D_sorted_1
## abt-4  -0.8776676  -1.0652937  -0.8943597  -1.0651520      0.9607023
## amt-4   1.5086891   1.4044684   1.5699383   1.4718552     -0.2107053
## aqp-4   0.5423177   0.6035933   0.2775824   0.4495869      0.7846391
## asm-2   0.2258180   0.6782376   0.2807804   0.4314604      0.8564958
## asp-1   1.6529634   1.9191847   1.9392496   1.3315259     -0.6143219
## asp-2   2.0334889   2.2653447   2.0248544   1.9586817     -1.3207025
##       elt2D_sorted_2 elt2D_sorted_3 elt2D_sorted_4 elt2Delt7D_sorted_1
## abt-4      0.7736452     0.94379989      0.7837143          0.02098377
## amt-4     -0.3544595    -0.09764439     -0.2871333         -1.71537290
## aqp-4      0.9472956     0.66884185      0.9323174         -1.67042064
## asm-2      0.8710924     0.96642706      0.9353808         -1.72805622
## asp-1     -0.2609884    -0.55458218     -0.2366091         -2.04098622
## asp-2     -1.1089528    -0.83159209     -0.9644632         -1.45702143
##       elt2Delt7D_sorted_2 elt2Delt7D_sorted_3
## abt-4          0.07090722          0.3487203
## amt-4         -1.64382731         -1.6458084
## aqp-4         -1.81812604         -1.7176276
## asm-2         -1.67988406         -1.8377524
## asp-1         -1.81810338         -1.3173325
## asp-2         -1.41765395         -1.1819836
```

Now plot a heatmap of ELT-2 regulated genes.

```
pheatmap(elt2bound_rowNormGeneNameMatrix,
         cluster_cols = FALSE,
         cluster_rows = TRUE,
         show_rownames = FALSE,
         border_color = NA)#,
```

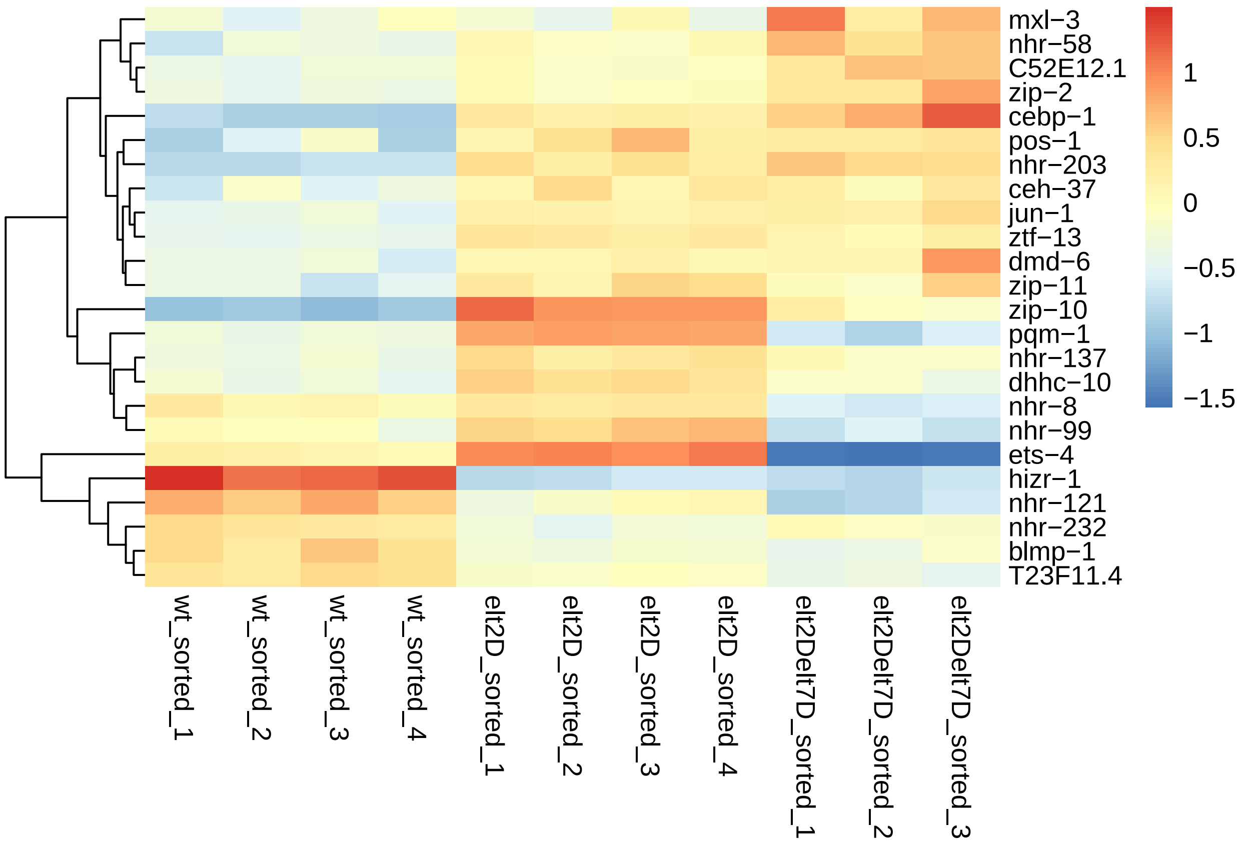


```
         #cellheight = 10,
         #filename = "./03_plots/200331_L1_Elt2_Elt7_Bound_Regulated_Genes_Row_Normalized_NoNames.pdf")
```

Do a similar analysis for TFs only.

```
elt2_bound_TF_matrix <- matrix_select(count_matrix = elt2_bound_matrix, gene_subset_vector = wTF3.0$WBG

elt2_bound_TF_rowNorm_matrix <- row_normalize_matrix_cutoff(
  count_matrix = elt2_bound_TF_matrix,
  variance_cutoff =  0.1
  )


elt2_bound_TF_rowNorm_matrix <- id2name(elt2_bound_TF_rowNorm_matrix)
```

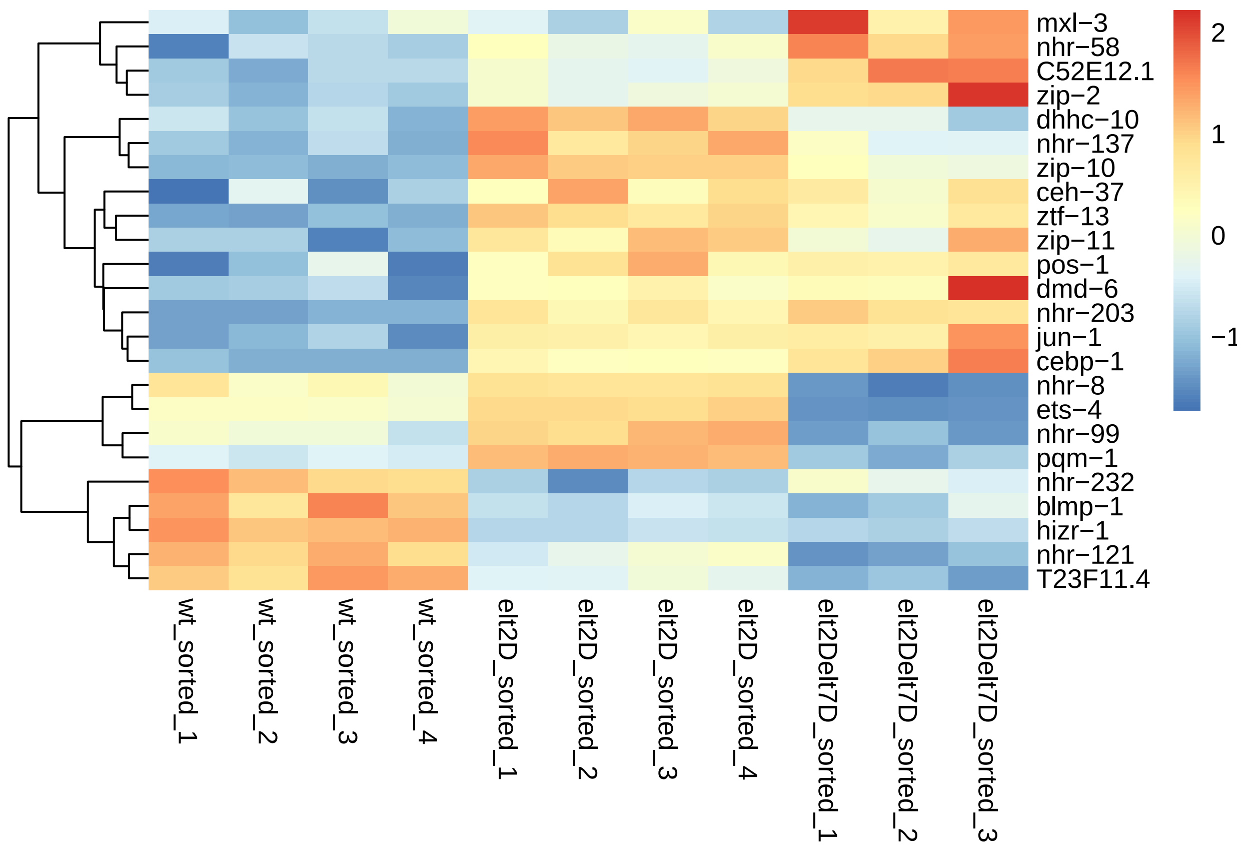
```
pheatmap(elt2_bound_TF_rowNorm_matrix,
         cluster_cols = FALSE,
         cluster_rows = TRUE,
         show_rownames = TRUE,
         border_color = NA)#,
```



```
         #cellheight = 10,
         #filename = "./03_plots/200331_L1_Elt2_Elt7_Bound_Regulated_TFs_Row_Normalized_Heatmap.pdf")
```

Do it for transcription factors.

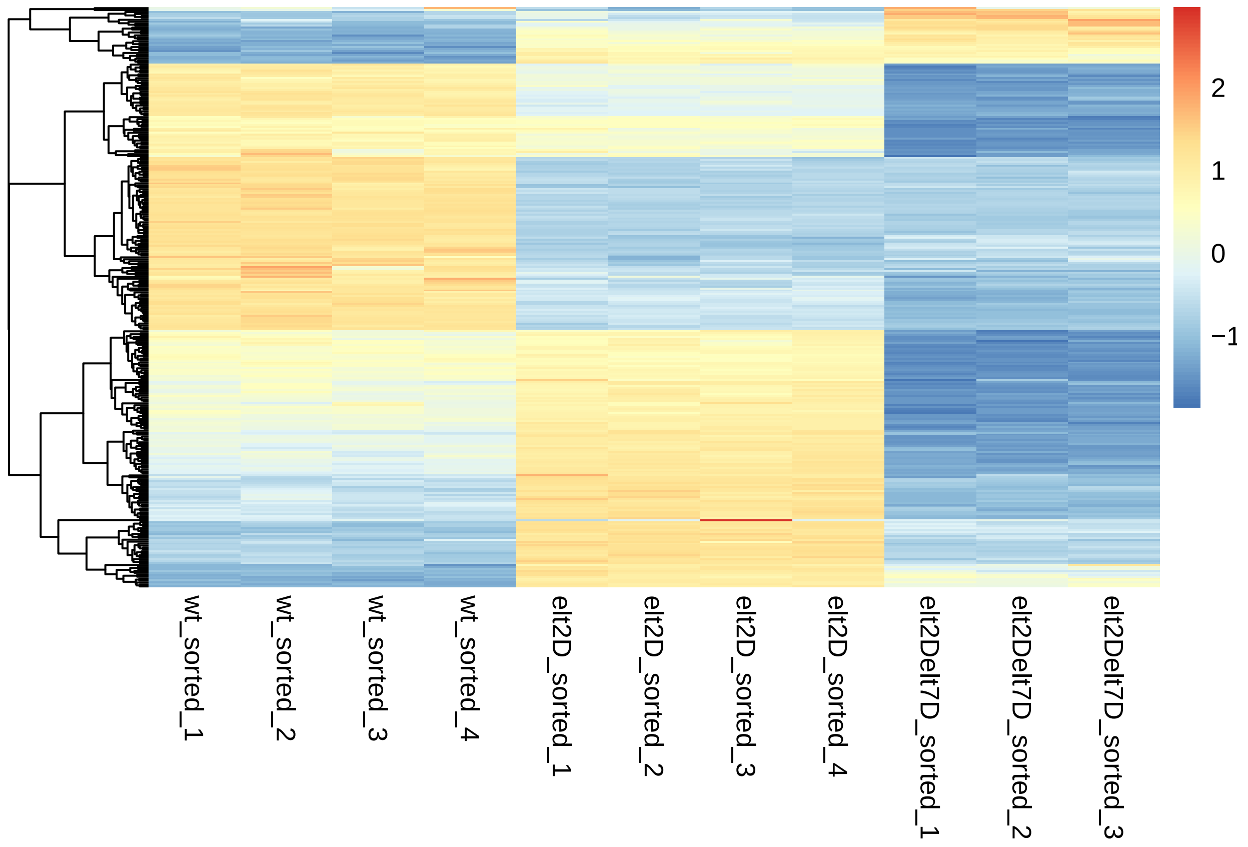
```
elt2_bound_TF_zscore_matrix <- row_zscore_matrix_cutoff(
  count_matrix = elt2_bound_TF_matrix,
  variance_cutoff =  0.1
  )

elt2_bound_TF_zscore_matrix <- id2name(elt2_bound_TF_zscore_matrix)

pheatmap(elt2_bound_TF_zscore_matrix,
         cluster_cols = FALSE,
         cluster_rows = TRUE,
         show_rownames = TRUE,
         border_color = NA)#,
```



```
         #cellheight = 10,
         #filename = "./03_plots/200328_L1_Elt2_Bound_Regulated_TFs_Zscore_Heatmap.pdf")
```

Do it for all genes.

```
elt2bound_zscore_matrix <- row_zscore_matrix_cutoff(
  count_matrix = elt2_bound_matrix,
  variance_cutoff =  0.5
  )

elt2bound_zscore_matrix <- id2name(elt2bound_zscore_matrix)

pheatmap(elt2bound_zscore_matrix,
         cluster_cols = FALSE,
         cluster_rows = TRUE,
         show_rownames = FALSE,
         border_color = NA)#,
```



```
         #cellheight = 10,    #420!!
         #filename = "./03_plots/200327_L1_Elt2_Bound_Regulated_Genes_Zscore_Heatmap.pdf")
```

## Use pairwise differential expression as regulated gene filter

Load in data.

```
up_in_wt_v_elt2 <- read_excel("01_input/Table_S4_Pairwise_Diff_Expression.xlsx",
    sheet = "1_up_in_wt_v_elt2", col_names = FALSE)
```

```
## New names:
## * `` -> ...1
```

```
down_in_wt_v_elt2 <- read_excel("01_input/Table_S4_Pairwise_Diff_Expression.xlsx",
    sheet = "2_down_in_wt_v_elt2", col_names = FALSE)
```

```
## New names:
## * `` -> ...1
```

```
up_in_wt_v_elt7 <- read_excel("01_input/Table_S4_Pairwise_Diff_Expression.xlsx",
    sheet = "3_up_in_wt_v_elt7", col_names = FALSE)
```

```
## New names:
## * `` -> ...1
```

```
up_in_wt_v_elt7elt2 <- read_excel("01_input/Table_S4_Pairwise_Diff_Expression.xlsx",
    sheet = "5_up_in_wt_v_elt7elt2", col_names = FALSE)
```

```
## New names:
## * `` -> ...1
```

```
down_in_wt_v_elt7elt2 <- read_excel("01_input/Table_S4_Pairwise_Diff_Expression.xlsx",
    sheet = "6_down_in_wt_v_elt7elt2", col_names = FALSE)
```

```
## New names:
## * `` -> ...1
```

```
up_in_elt2_v_elt7elt2 <- read_excel("01_input/Table_S4_Pairwise_Diff_Expression.xlsx",
    sheet = "7_up_in_elt2_v_elt7elt2", col_names = FALSE)
```

```
## New names:
## * `` -> ...1
```

```
down_in_elt2_v_elt7elt2 <- read_excel("01_input/Table_S4_Pairwise_Diff_Expression.xlsx",
    sheet = "8_down_in_elt2_v_elt7elt2", col_names = FALSE)
```

```
## New names:
## * `` -> ...1
```

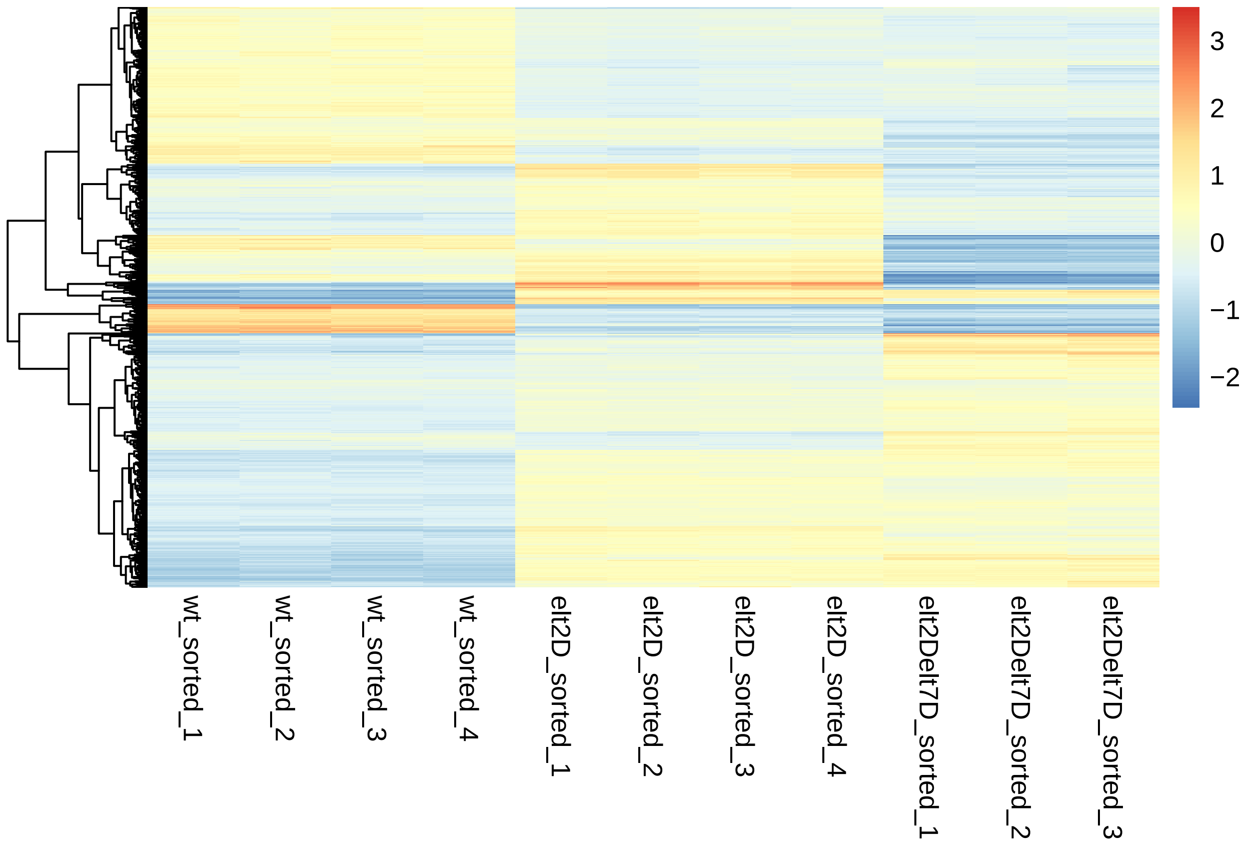
```
colnames(up_in_wt_v_elt2) <- c("WBGeneID")
colnames(down_in_wt_v_elt2) <- c("WBGeneID")
colnames(up_in_wt_v_elt7) <- c("WBGeneID")
colnames(up_in_wt_v_elt7elt2) <- c("WBGeneID")
colnames(down_in_wt_v_elt7elt2) <- c("WBGeneID")
colnames(up_in_elt2_v_elt7elt2) <- c("WBGeneID")
colnames(down_in_elt2_v_elt7elt2) <- c("WBGeneID")
```

Make a union of these lists with unique WBGeneIDs.

```
union_elt2elt7_DE <- data.frame(WBGeneID = c(up_in_wt_v_elt2$WBGeneID,
                        down_in_wt_v_elt2$WBGeneID,
                        up_in_wt_v_elt7$WBGeneID,
                        up_in_wt_v_elt7elt2$WBGeneID,
                        down_in_wt_v_elt7elt2$WBGeneID,
                        up_in_elt2_v_elt7elt2$WBGeneID,
                        down_in_elt2_v_elt7elt2$WBGeneID
                        )) %>% unique()
```

Subset count matrix for presence in union of all pairwise comparisons.

```
all_pairwise_subset <- matrix_select(assay(rld), union_elt2elt7_DE$WBGeneID)

row_normalize_matrix <- function(count_matrix){
  namevarRowNormalized <- count_matrix - rowMeans(count_matrix)
  return(namevarRowNormalized)
}

all_pairwise_subset_rownorm <- row_normalize_matrix(all_pairwise_subset)

myPheatmap(all_pairwise_subset_rownorm)
```



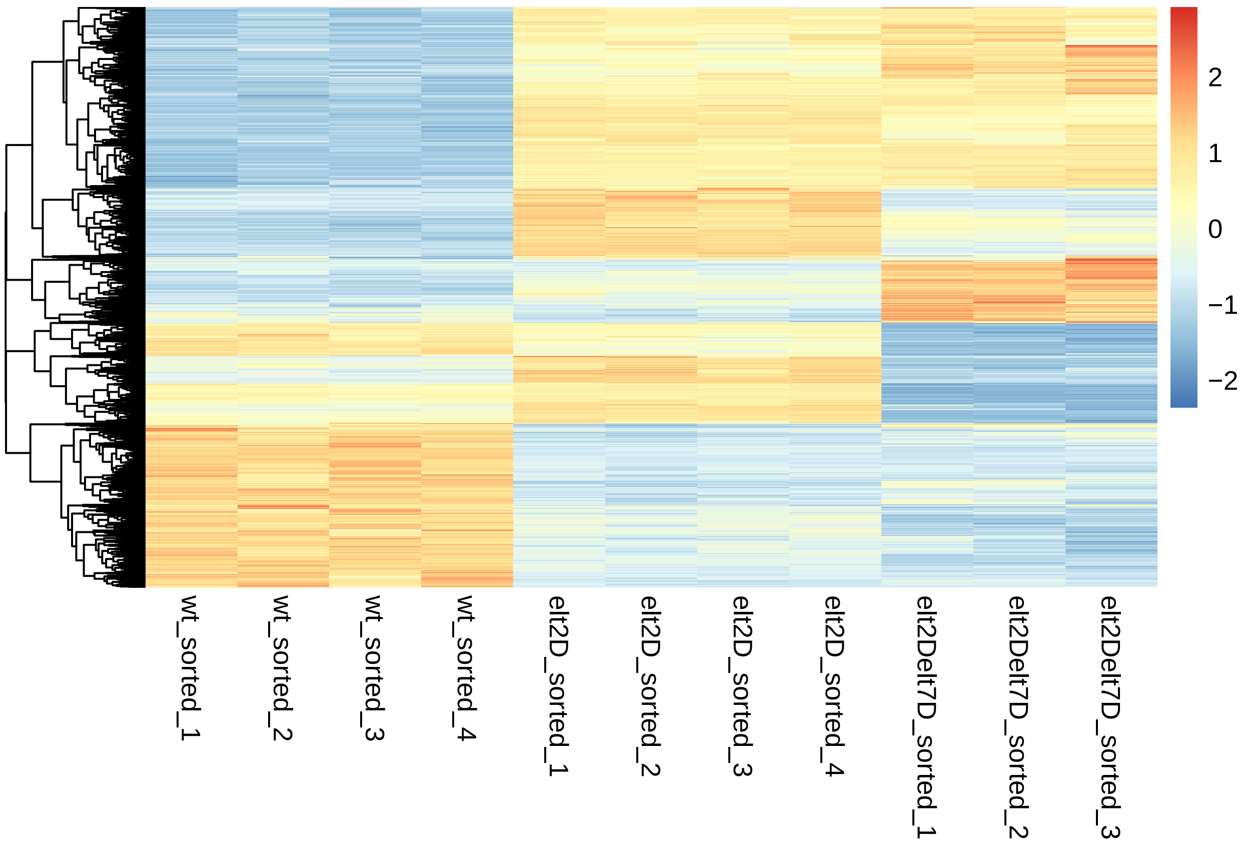
Hard to see anything useful with this.

Do the same thing but use Z score. Maybe there will be more detail.

```
all_pairwise_subset_Zscore <- row_zscore_matrix(all_pairwise_subset)

# remove columns with NA
all_pairwise_subset_Zscore <- all_pairwise_subset_Zscore[complete.cases(all_pairwise_subset_Zscore), ]

unique(is.na(all_pairwise_subset_Zscore))
```

```
##                 wt_sorted_1 wt_sorted_2 wt_sorted_3 wt_sorted_4 elt2D_sorted_1
## WBGene00000007       FALSE       FALSE       FALSE       FALSE          FALSE
##                 elt2D_sorted_2 elt2D_sorted_3 elt2D_sorted_4 elt2Delt7D_sorted_1
## WBGene00000007          FALSE          FALSE          FALSE               FALSE
##                 elt2Delt7D_sorted_2 elt2Delt7D_sorted_3
## WBGene00000007               FALSE               FALSE
```

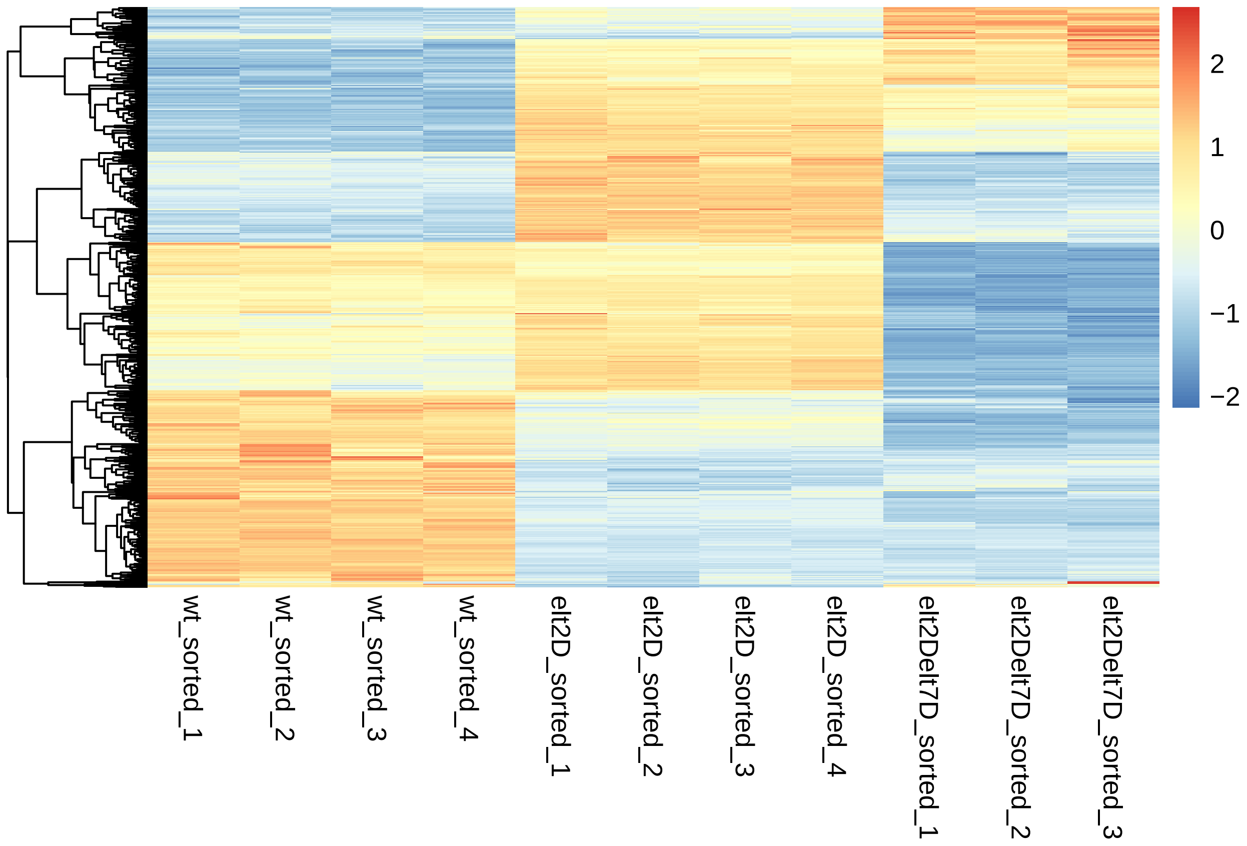
```
myPheatmap(all_pairwise_subset_Zscore)
```



Clusters are a little more obvious.

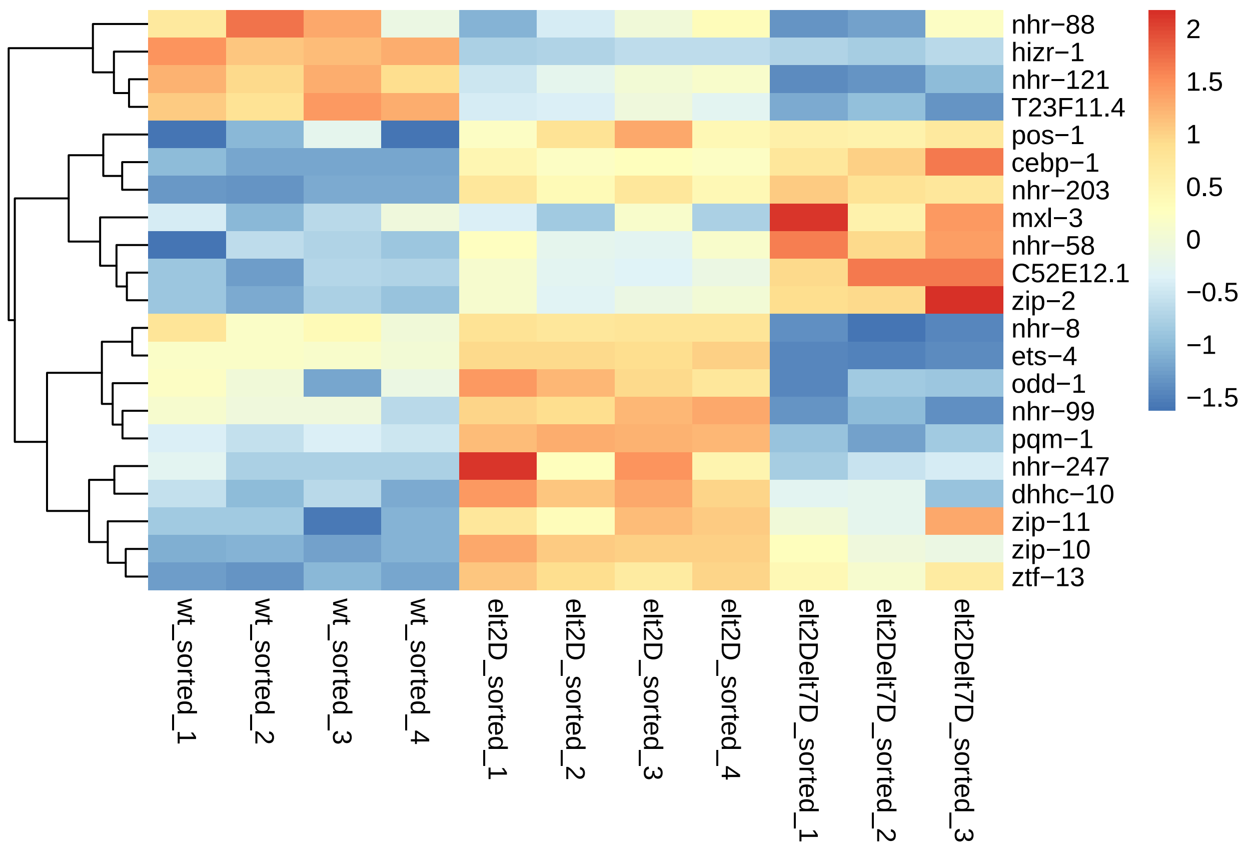
Now subset the plot for ELT-2 binding in the L1 stage.

```
elt2bound_all_pairwise_subset_Zscore <- matrix_select(all_pairwise_subset_Zscore, elt2_L1_peaks$mapped_g

myPheatmap(elt2bound_all_pairwise_subset_Zscore)
```



Now subset for genes that are transcription factors.

```
elt2bound_all_pairwise_subset_Zscore_TF <- matrix_select(elt2bound_all_pairwise_subset_Zscore, wTF3.0$WI

elt2bound_all_pairwise_subset_Zscore_TF <- id2name(elt2bound_all_pairwise_subset_Zscore_TF)

mysmallPheatmap(elt2bound_all_pairwise_subset_Zscore_TF)
```



## Session info

Document session info.

```
sessionInfo()
```

```
## R version 3.6.3 (2020-02-29)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS Sierra 10.12.5
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] parallel  stats4    stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
##  [1] readxl_1.3.1               pheatmap_1.0.12
##  [3] forcats_0.5.0              stringr_1.4.0
##  [5] dplyr_0.8.5                purrr_0.3.3
##  [7] readr_1.3.1                tidyr_1.0.2
##  [9] tibble_2.1.3                ggplot2_3.3.0
## [11] tidyverse_1.3.0             DESeq2_1.26.0
## [13] SummarizedExperiment_1.16.1 DelayedArray_0.12.2
## [15] BiocParallel_1.20.1         matrixStats_0.56.0
## [17] Biobase_2.46.0              GenomicRanges_1.38.0
## [19] GenomeInfoDb_1.22.0         IRanges_2.20.2
## [21] S4Vectors_0.24.3            BiocGenerics_0.32.0
## [23] biomaRt_2.42.0
##
## loaded via a namespace (and not attached):
##  [1] colorspace_1.4-1       htmlTable_1.13.3       XVector_0.26.0
##  [4] base64enc_0.1-3        fs_1.3.2               rstudioapi_0.11
##  [7] farver_2.0.3           bit64_0.9-7            fansi_0.4.1
## [10] AnnotationDbi_1.48.0   lubridate_1.7.4        xml2_1.2.5
## [13] splines_3.6.3          geneplotter_1.64.0     knitr_1.28
## [16] Formula_1.2-3          jsonlite_1.6.1         broom_0.5.5
## [19] annotate_1.64.0        cluster_2.1.0          dbplyr_1.4.2
## [22] png_0.1-7              compiler_3.6.3         httr_1.4.1
## [25] backports_1.1.5        assertthat_0.2.1       Matrix_1.2-18
## [28] cli_2.0.2              acepack_1.4.1          htmltools_0.4.0
## [31] prettyunits_1.1.1      tools_3.6.3            gtable_0.3.0
## [34] glue_1.3.2             GenomeInfoDbData_1.2.2 rappdirs_0.3.1
## [37] Rcpp_1.0.4             cellranger_1.1.0       vctrs_0.2.4
## [40] nlme_3.1-145           xfun_0.12              rvest_0.3.5
## [43] lifecycle_0.2.0        XML_3.99-0.3           zlibbioc_1.32.0
## [46] scales_1.1.0           hms_0.5.3              RColorBrewer_1.1-2
## [49] yaml_2.2.1             curl_4.3               memoise_1.1.0
## [52] gridExtra_2.3          rpart_4.1-15           latticeExtra_0.6-29
## [55] stringi_1.4.6          RSQLite_2.2.0          genefilter_1.68.0
## [58] checkmate_2.0.0        rlang_0.4.5            pkgconfig_2.0.3
## [61] bitops_1.0-6           evaluate_0.14          lattice_0.20-40
## [64] labeling_0.3           htmlwidgets_1.5.1      bit_1.1-15.2
## [67] tidyselect_1.0.0       magrittr_1.5           R6_2.4.1
## [70] generics_0.0.2         Hmisc_4.3-1            DBI_1.1.0
## [73] withr_2.1.2            pillar_1.4.3           haven_2.2.0
## [76] foreign_0.8-76         survival_3.1-11        RCurl_1.98-1.1
## [79] nnet_7.3-13            modelr_0.1.6           crayon_1.3.4
## [82] BiocFileCache_1.10.2   rmarkdown_2.1          jpeg_0.1-8.1
## [85] progress_1.2.2         locfit_1.5-9.1         grid_3.6.3
## [88] data.table_1.12.8      blob_1.2.1             reprex_0.3.0
## [91] digest_0.6.25          xtable_1.8-4           openssl_1.4.1
## [94] munsell_0.5.0          askpass_1.1
```