RWC23_ELT2_Regulated_Genes

RTPW

4/13/2020

Install Packages

```r
# if (!requireNamespace("BiocManager", quietly = TRUE))
#     install.packages("BiocManager")
# BiocManager::install()
# BiocManager::install("biomaRt")
# install.packages("tidyverse")
# install.packages("readxl")
# BiocManager::install("ComplexHeatmap")
# install.packages("matrixStats")
# install.packages("pheatmap")
# install.packages("RVAideMemoire")
# install.packages("dendextend")
# install.packages("binom")
```

Load Package Libraries

```r
library(biomaRt)
library(tidyverse)
library(readxl)
library(ComplexHeatmap)
library(matrixStats)
library(pheatmap)
library(RVAideMemoire)
library(dendextend)
library(binom)
library(circlize)
```

Background and Rationale

ELT-2 is the C. elegans intestine master regulator. Deletion of ELT-2 leads to a larval lethal phenotype, and expression of ELT-2 in non-intestine tissue induces an intestine fate.

This documet will generate plots to address the questions outlined below.

For genes differentially expressed during elt-2 (-) and/or elt-7(-):

1) which expression pattern clusters associate with ELT-2 binding?
2) which expression pattern clusters associate with ELT-2 binding categories?

a) For all genes
b) For only genes bound by ELT-2

3) Which expression pattern clusters associate with intestine expression? (MA plot for each expression set)

a) For all genes
b) For genes only bound by ELT-2

For clusters of transcription factors (TFs) differentially expressed during elt-2 (-) and/or elt-7(-):

1) which transcription factor clusters associate with ELT-2 binding?
2) which transcription factor clusters associate with ELT-2 binding categories

a) for all TFs
b) For only TFs bound by ELT-2

3) which transcription factor clusters associate with intestine expression?

a) for all
b) for only ELT-2 bound

Data

I will integrate two RNA-seq experiments and a ChIP-seq experiments.

The first is a set of RNA-seq experiments in L1 stage worms (Dineen and Nishimura, 2018). They were collected from the following genotypes, all in the L1 stage:

- wildtype (wt)
- elt-7 deleted (elt7D)
- elt-2 deleted (elt2D)
- combination fo elt-7 and elt-2 deleted (elt2Delt7D)

The purpose of including elt-7 and elt-2/elt-7 double deletion is because these two transcription factors have overlapping functionality. Deletion of elt-7 alone does not have a phenotype, but deletion of elt-7 in combination with elt-2 has an enhanced lethal phenotype of just elt-2 alone.

The second RNA-seq experiment is from FACS sorted L1 stage intestine cells. This data is unpublished.

The ChIP-seq experiments are performed against ELT-2 and are from the following developmental stages:

- late embryo (LE)
- L1
- L3

They were collected as part of the modENCODE consortium and were processed by David King. He has provided gene mapping of ELT-2 targets and categories of ELT-2 binding. The ELT-2 binding categories are as follows:

- Not changing
- Larval
- L3 high
- Embryonic
- Increasing

Citations

1) Dineen, A., Osborne Nishimura, E., Goszczynski, B., Rothman, J. H., & McGhee, J. D. (2018). Quantitating transcription factor redundancy: The relative roles of the ELT-2 and ELT-7 GATA factors in the C. elegans endoderm. Developmental Biology, 435(2), 150–161. https://doi.org/10.1016/J.YDBIO.2017.12.023

2) Kudron, M. M., Victorsen, A., Gevirtzman, L., Hillier, L. W., Fisher, W. W., Vafeados, D., … Waterston, R. H. (2018). The modern resource: genome-wide binding profiles for hundreds of Drosophila

and Caenorhabditis elegans transcription factors. Genetics, 208(3), 937–949. https://doi.org/10.1534/genetics.117.300657

Code

Source functions

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

Load and Process Datasets

Load Dineen and Osborne Nishimura et. al. Data

```
dineen_nishimura_counts <-
  read_xlsx(path = "./01_input/Table_S2_rlog_Stabilized_Read_Counts.xlsx",
            sheet = "Sheet1")

dineen_nishimura_counts_matrix <- dineen_nishimura_counts %>%
  column_to_rownames(var = "WBGeneID")  %>%
  data.matrix()

dineen_nishimura_counts_matrix %>% head
```

```
##                 wt_sorted_1 wt_sorted_2 wt_sorted_3 wt_sorted_4 elt7D_sorted_1
## WBGene00000001    8.957161    8.858238    8.841623    8.923111       8.505028
## WBGene00000002    7.489159    7.382905    7.518631    7.492399       7.378168
## WBGene00000003    9.061810    8.748589    9.295497    9.286834       9.480361
## WBGene00000004   10.916559   10.786200   11.010430   10.826657      10.836827
## WBGene00000005    2.990777    2.864044    3.116144    2.715502       2.584081
## WBGene00000007    5.799066    6.026780    5.831420    6.072836       5.699261
##                 elt7D_sorted_2 elt7D_sorted_3 elt2D_sorted_1 elt2D_sorted_2
## WBGene00000001       8.568569       8.517438       9.172904       9.249496
## WBGene00000002       7.582425       7.512668       7.503760       7.289884
## WBGene00000003       9.451384       9.008938       8.669299       8.593847
## WBGene00000004      10.806534      10.819497      10.303062      10.296768
## WBGene00000005       2.881642       2.827526       2.953325       2.835451
## WBGene00000007       5.492677       5.220378       4.683237       4.797660
##                 elt2D_sorted_3 elt2D_sorted_4 elt2Delt7D_sorted_1
## WBGene00000001       9.211660       9.346959            9.379698
## WBGene00000002       7.386127       7.262063            7.904008
## WBGene00000003       8.753835       8.781267            8.791018
## WBGene00000004      10.356820      10.366512           10.332489
## WBGene00000005       2.886842       2.979650            2.499412
## WBGene00000007       4.495252       4.593047            4.602235
##                 elt2Delt7D_sorted_2 elt2Delt7D_sorted_3
## WBGene00000001            9.217403            9.101997
## WBGene00000002            7.870852            7.762023
## WBGene00000003            8.795191            8.936724
## WBGene00000004           10.223675           10.597407
## WBGene00000005            2.763405            2.428255
## WBGene00000007            4.641832            4.476899
```

list of all dynamically expressed genes

```
dynamic_regulated_genes <-
  read.table(file = "./05_fromErin/2017-11-20_all_changing_genes_0.1alpha_0.8lfc.txt",
             quote = "",
             header = FALSE)
colnames(dynamic_regulated_genes) <- "WBGeneID"

dynamic_regulated_genes %>% head
```

```
##         WBGeneID
## 1 WBGene00004020
## 2 WBGene00015956
## 3 WBGene00000216
## 4 WBGene00001795
## 5 WBGene00008167
## 6 WBGene00010049
```

```
dineen_nishimura_clusters <-
  read_xlsx(path = "./01_input/Table_S6_All_Dynamically_Expressed_Genes_Clusters.xlsx",
            sheet = "dataset")

dineen_nishimura_clusters %>% select(WBGeneID, set) %>% head
```

```
## # A tibble: 6 x 2
##   WBGeneID       set
##   <chr>          <chr>
## 1 WBGene00000007 set6
## 2 WBGene00000008 set6
## 3 WBGene00000009 set3
## 4 WBGene00000013 set1
## 5 WBGene00000016 set1
## 6 WBGene00000017 set1
```

Load ELT-2 ChIP-seq binding annotations

```
elt2_peaks <-
  read_xlsx("./01_input/200410_peaksForBigBed.xlsx", sheet = "full cluster assignment")
```

```
## New names:
## * `` -> ...12
```

```
colnames(elt2_peaks) <-
  c(
    "chrom",
    "start",
    "end",
    "peak.name",
    "WBGeneID",
    "mapping",
    "cluster",
    "cluster.description",
    "kweight",
    "LE",
    "L1",
    "L3",
    "peak.summit.agreement"
  )

elt2_peaks$cluster.description <-
  factor(
    elt2_peaks$cluster.description,
    levels = c(
      "Not-changing or not IDR-passing",
      "L3-high",
      "Increasing",
      "Post-embryonic",
      "LE-specific"
    ),
    labels = c(
      "NotChanging",
      "L3high",
      "Increasing",
      "PostEmbryonic",
      "LEspecific"
    )
  )

elt2_peaks %>% head
```

```
## # A tibble: 6 x 13
##   chrom start   end peak.name WBGeneID mapping cluster cluster.descrip~ kweight
##   <chr> <dbl> <dbl> <chr>     <chr>    <chr>     <dbl> <fct>              <dbl>
## 1 chrI   3691  4222 ELT2peak~ WBGene0~ overla~       4 Increasing         0.818
## 2 chrI  11044 11533 ELT2peak~ WBGene0~ overla~       4 Increasing         0.913
## 3 chrI  13560 14890 ELT2peak~ WBGene0~ inside        2 PostEmbryonic      0.876
## 4 chrI  15179 15647 ELT2peak~ WBGene0~ inside        4 Increasing         0.993
## 5 chrI  16706 17483 ELT2peak~ WBGene0~ overla~       3 L3high             0.989
## 6 chrI  26789 27576 ELT2peak~ WBGene0~ downst~       1 LEspecific         0.92
## # ... with 4 more variables: LE <dbl>, L1 <dbl>, L3 <dbl>,
## #   peak.summit.agreement <dbl>
```

Make a set of genes with ELT-2 binding detected in the L1 stage.

```
elt2_detected_in_L1 <-
  elt2_peaks %>% select(WBGeneID, L1) %>% filter(L1 == 1) %>% select(WBGeneID) %>% unique()

elt2_detected_in_L1 %>% head
```

```
## # A tibble: 6 x 1
##   WBGeneID
##   <chr>
## 1 WBGene00022277
## 2 WBGene00022276
## 3 WBGene00021026
## 4 WBGene00022038
## 5 WBGene00022043
## 6 WBGene00022042
```

```
elt2_detected_in_L1 %>% dim
```

```
## [1] 2430    1
```

Make a dataframe that records the number of peaks per gene that fall in a particular binding catagory.

```
binding_cluster_gene_counts <-
  table(elt2_peaks$WBGeneID, elt2_peaks$cluster.description)
binding_cluster_gene_counts <-
  as.data.frame.matrix(binding_cluster_gene_counts)
binding_cluster_gene_counts %>% head()
```

```
##                  NotChanging L3high Increasing PostEmbryonic LEspecific
## WBGene00000003           0      0          1             0          0
## WBGene00000004           0      0          0             2          0
## WBGene00000007           0      0          1             0          0
## WBGene00000008           0      0          1             0          0
## WBGene00000009           0      0          1             1          0
## WBGene00000010           0      1          0             0          0
```

Load Spencer et. al. intestine expression

```
spencerLEgenes <-
  read.table(
    "../TF_Team/02_Data/6_Spencer_et_al_2010_FACS_and_pulldown_tilling_array/LE-intestine_enr_vs_ref.WS
    quote = "\"",
    comment.char = "",
    header = TRUE
  )
colnames(spencerLEgenes) <-
  str_c("spencer_LE_", colnames(spencerLEgenes))
spencer_LE_subset <-
  spencerLEgenes %>% select(spencer_LE_ID,
                            spencer_LE_AveExpr,
                            spencer_LE_adj_P_Val,
                            spencer_LE_FC)

spencer_LE_subset %>% head
```

```
##    spencer_LE_ID spencer_LE_AveExpr spencer_LE_adj_P_Val spencer_LE_FC
## 1 WBGene00008163               7.57                    0         13.86
## 2 WBGene00021252               8.21                    0          7.30
## 3 WBGene00019986               9.29                    0         10.67
## 4 WBGene00007904               8.16                    0          6.89
## 5 WBGene00012018              10.14                    0          6.25
## 6 WBGene00010540               8.43                    0          4.15
```

```
spencerL2genes <-
  read.table(
    "../TF_Team/02_Data/6_Spencer_et_al_2010_FACS_and_pulldown_tilling_array/L2-intestine_enr_vs_ref.WS
    quote = "\"",
    comment.char = "",
    header = TRUE
  )
colnames(spencerL2genes) <-
  str_c("spencer_L2_", colnames(spencerL2genes))
spencer_L2_subset <- spencerL2genes %>%
  select(spencer_L2_ID,
         spencer_L2_AveExpr,
         spencer_L2_adj_P_Val,
         spencer_L2_FC)

spencer_L2_subset %>% head
```

```
##    spencer_L2_ID spencer_L2_AveExpr spencer_L2_adj_P_Val spencer_L2_FC
## 1 WBGene00020352               7.52                    0          7.51
## 2 WBGene00017225               7.28                    0          5.32
## 3 WBGene00007973               7.91                    0          5.93
## 4 WBGene00018683               8.27                    0          5.10
## 5 WBGene00003696               7.95                    0          3.73
## 6 WBGene00044776               7.77                    0          6.65
```

Process rlog counts

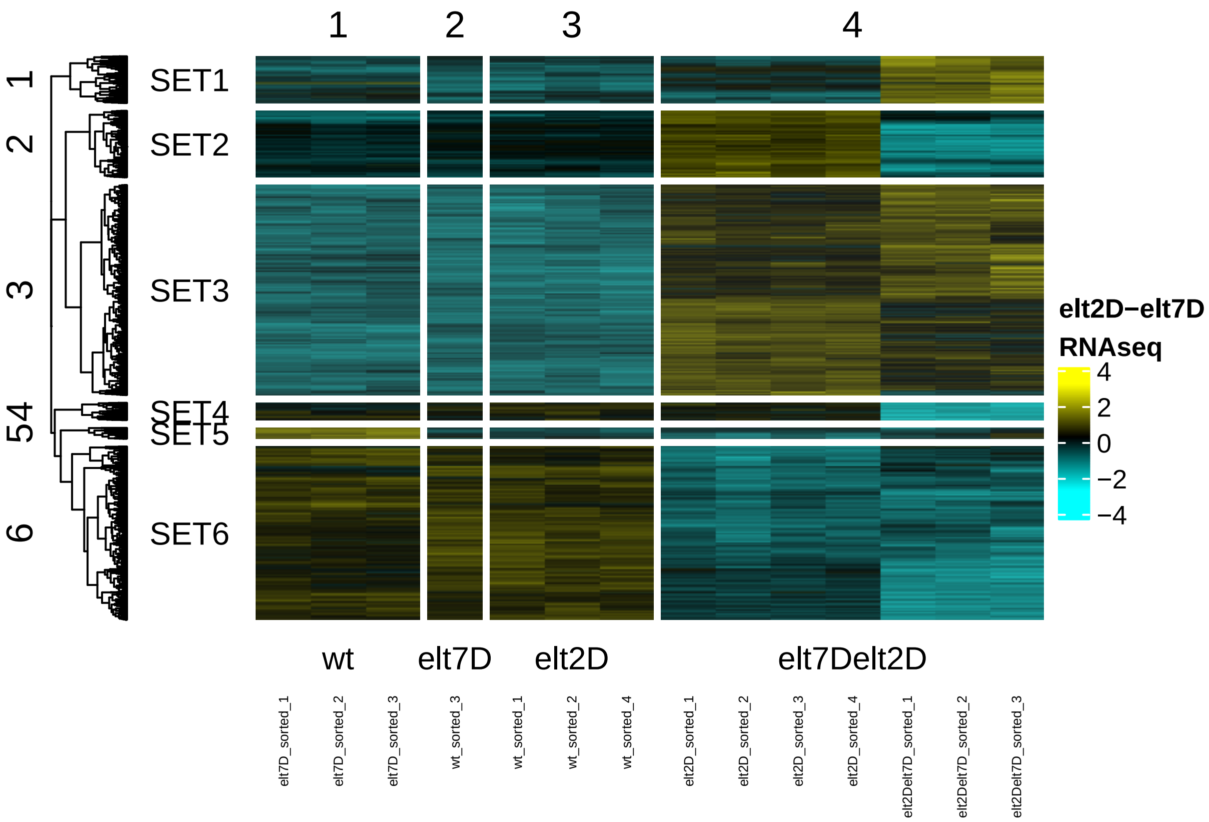
Subset rlog matrix based on presence in list `2017-11-20_all_changing_genes_0.1alpha_0.8lfc.txt`. Row scale and center the rlog counts per genes.

```
dynamic_counts_matrix <-
  matrix_select(dineen_nishimura_counts_matrix,
                dynamic_regulated_genes$WBGeneID)

dynamic_counts_matrix_scaled <-
  t(apply(unlist(dynamic_counts_matrix), 1, scale))

rownames(dynamic_counts_matrix_scaled) <-
  rownames(dynamic_counts_matrix)
colnames(dynamic_counts_matrix_scaled) <-
  colnames(dynamic_counts_matrix)
dynamic_counts_matrix_scaled %>% head
```

```
##                wt_sorted_1 wt_sorted_2 wt_sorted_3 wt_sorted_4 elt7D_sorted_1
## WBGene00000007   1.0068329  1.37348252   1.0589277   1.4476397     0.84613352
## WBGene00000008   2.2632093  1.13063525   1.1251278   1.0262925    -0.03607787
## WBGene00000009   0.1468716 -0.09556483  -0.3465276  -0.8378633     0.07003147
## WBGene00000013  -1.0765042  0.04628523  -1.0478603  -0.4296435    -0.61401384
## WBGene00000016  -0.1629274  0.14035593  -0.8318355  -0.2209018    -0.52814604
## WBGene00000017   0.1344074  0.43209491  -0.4453539   0.5202470    -0.19720767
##                elt7D_sorted_2 elt7D_sorted_3 elt2D_sorted_1 elt2D_sorted_2
## WBGene00000007     0.51350637     0.07506888     -0.7898010     -0.6055647
## WBGene00000008    -0.39030667     0.02722321     -0.4521136     -1.0292850
## WBGene00000009    -0.11586861     0.42221560      0.8406016      1.2349599
## WBGene00000013    -0.58009755    -0.38693983     -0.4767996      0.3851813
## WBGene00000016    -0.50445577    -0.16186256     -0.5681545     -0.6137809
## WBGene00000017     0.05519157     0.37152702     -0.9790560     -1.0378885
##                elt2D_sorted_3 elt2D_sorted_4 elt2Delt7D_sorted_1
## WBGene00000007    -1.09248186     -0.9350192          -0.9202246
## WBGene00000008    -0.46498937     -0.8771172          -0.9402531
## WBGene00000009     0.98161197      1.7266509          -1.7004545
## WBGene00000013     0.09286966     -0.5163112           2.5457794
## WBGene00000016    -0.75209134     -1.0136068           1.7015008
## WBGene00000017    -1.16996644     -1.7376299           1.4066491
##                elt2Delt7D_sorted_2 elt2Delt7D_sorted_3
## WBGene00000007          -0.8564679          -1.1220323
## WBGene00000008          -0.5550156          -0.8273297
## WBGene00000009          -0.8668929          -1.4597714
## WBGene00000013           1.4999051           0.5581492
## WBGene00000016           2.1353949           1.3805110
## WBGene00000017           1.6701858           0.9767996
```

Recreate Supplementary Figure S4a from Dineen and Nishimura et al.

```
Ha <- Heatmap(
  dynamic_counts_matrix_scaled,
  name = "elt2D-elt7D\nRNAseq",
  col = colorRampPalette(c("cyan", "black", "yellow"))(1000),
  cluster_columns = FALSE,
  clustering_distance_rows = "spearman",
  clustering_method_rows = "complete",
  show_row_names = FALSE,
  show_column_names = TRUE,
  row_names_gp = gpar(cex = 0.2),
  column_names_gp = gpar(cex = 0.4),
  heatmap_legend_param = list(color_bar = "continuous"),
  split = 6,
  column_km = 4,
  bottom_annotation = HeatmapAnnotation(foo = anno_block(
    labels = c("wt", "elt7D", "elt2D", "elt7Delt2D"),
    gp = gpar(border = NA, lty = "blank")
  )),
  left_annotation = rowAnnotation(foo = anno_block(
    labels = c("SET1", "SET2", "SET3", "SET4", "SET5", "SET6"),
    labels_rot = 0,
    gp = gpar(border = NA, lty = "blank")
  ))
)
Ha
```

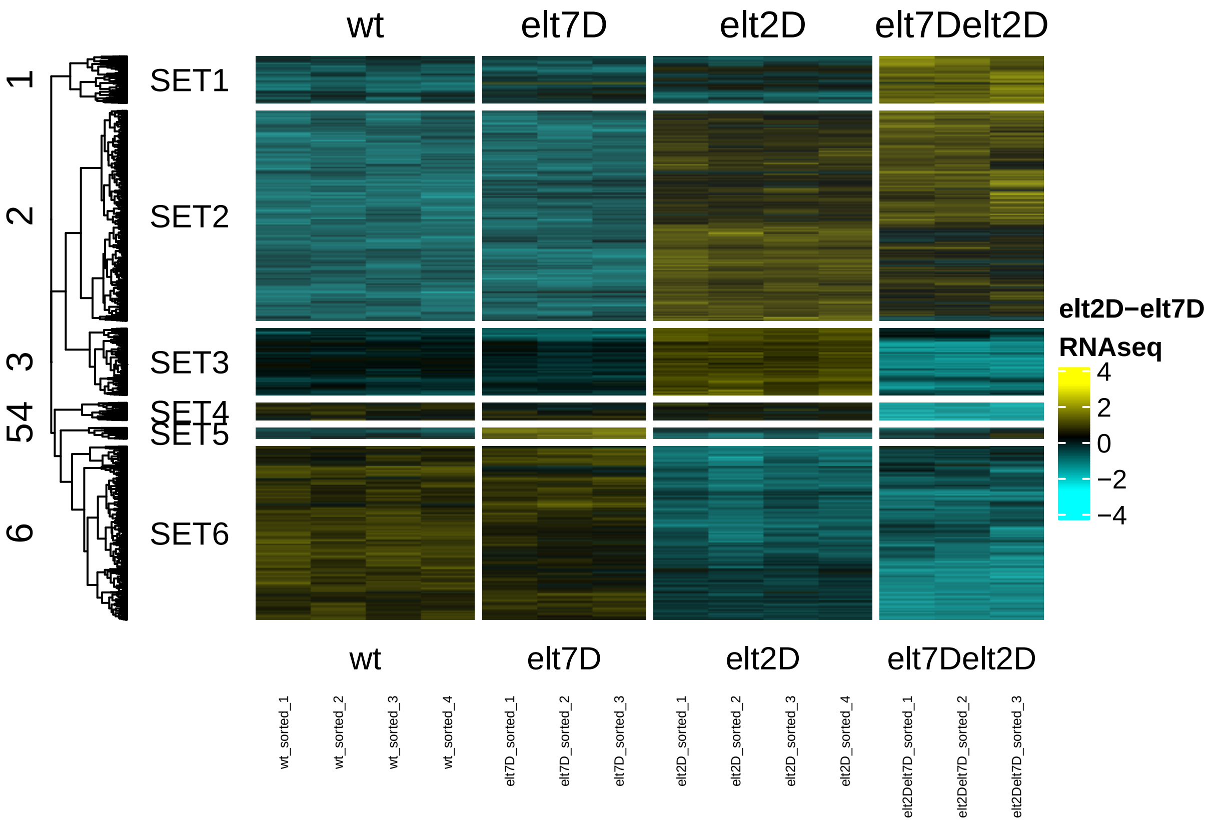


They don't really look the same. Maybe I would try doing these things to figure out how they are different. - Correlation of scaled values between the two methods - plot the row means from each analysis against eachother before and after scaling - plot the row standard deviation from each analysis per gene

Rotate the dendrogram to match Erin's order.

```
c <- cor(t(dynamic_counts_matrix_scaled), method = "spearman")
d <- as.dist(1 - c)
rowdend <- as.dendrogram(hclust(d))

SET1 = 1:276
SET2 = 277:669
SET3 = 670:1906
SET4 = 1907:2010
SET5 = 2011:2076
SET6 = 2077:3092

rowdend_rotate <- rotate(rowdend, c(SET1,
                                    SET3,
                                    SET2,
                                    SET4,
                                    SET5,
                                    SET6))

Ha_rotate <- Heatmap(
  dynamic_counts_matrix_scaled,
  name = "elt2D-elt7D\nRNAseq",
  col = colorRampPalette(c("cyan", "black", "yellow"))(1000),
  cluster_columns = FALSE,
```

```r
  column_split = factor(
    c(
      rep("wt", 4),
      rep("elt7D", 3),
      rep("elt2D", 4),
      rep("elt7Delt2D", 3)
    ),
    levels = c("wt", "elt7D", "elt2D", "elt7Delt2D")
  ),
  cluster_rows = rowdend_rotate,
  clustering_distance_rows = "spearman",
  clustering_method_rows = "complete",
  show_row_names = FALSE,
  show_column_names = TRUE,
  row_names_gp = gpar(cex = 0.2),
  column_names_gp = gpar(cex = 0.4),
  heatmap_legend_param = list(color_bar = "continuous"),
  split = 6,
  bottom_annotation = HeatmapAnnotation(foo = anno_block(
    labels = c("wt", "elt7D", "elt2D", "elt7Delt2D"),
    gp = gpar(border = NA, lty = "blank")
  )),
  left_annotation = rowAnnotation(foo = anno_block(
    labels = c("SET1", "SET2", "SET3", "SET4", "SET5", "SET6"),
    labels_rot = 0,
    gp = gpar(border = NA, lty = "blank")
  ))
)

Ha_rotate
```



Extract cluster assignments.

```
gene_expression_clusters <-
  assign_cluster_from_heatmap(mat = dynamic_counts_matrix_scaled, plot = Ha_rotate)
nrow(gene_expression_clusters) == nrow(dynamic_counts_matrix_scaled)
```

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

```
head(gene_expression_clusters)
```

```
##          WBGeneID Expression.Cluster
## 1 WBGene00012450               SET1
## 2 WBGene00018472               SET1
## 3 WBGene00004052               SET1
## 4 WBGene00007378               SET1
## 5 WBGene00018903               SET1
## 6 WBGene00007507               SET1
```

TODO: Make sure cluster assignments match Erin's

```
dineen_nishimura_sets <-
  dineen_nishimura_clusters %>% select(WBGeneID, set)
```

Add ELT-2 pattern row annotation

In ComplexHeatmap the row order of input matrix and annotation df must be identical to accurately plot data.

```r
elt2_detected_in_L1 %>% dim
```

```
## [1] 2430    1
```

```r
elt2_L1_anno <-
  data.frame(
    elt2_detected_in_L1 = ifelse(
      test = rownames(dynamic_counts_matrix_scaled) %in% elt2_detected_in_L1$WBGeneID,
      yes = "bound",
      no = "not.bound"
    )
  )

#spot check that order is preserved. bound/TRUE and not.bound/FALSE should appear
valz = 100
elt2_L1_anno[valz, 1]
```

```
## [1] not.bound
## Levels: bound not.bound
```

```r
rownames(dynamic_counts_matrix_scaled)[valz] %in%
  elt2_detected_in_L1$WBGeneID
```

```
## [1] FALSE
```

```r
elt2_L1_anno <-
  elt2_L1_anno %>% mutate(WBGeneID = rownames(dynamic_counts_matrix_scaled))

elt2_L1_anno %>% dim()
```

```
## [1] 3092    2
```

```r
elt2_L1_anno %>% class()
```

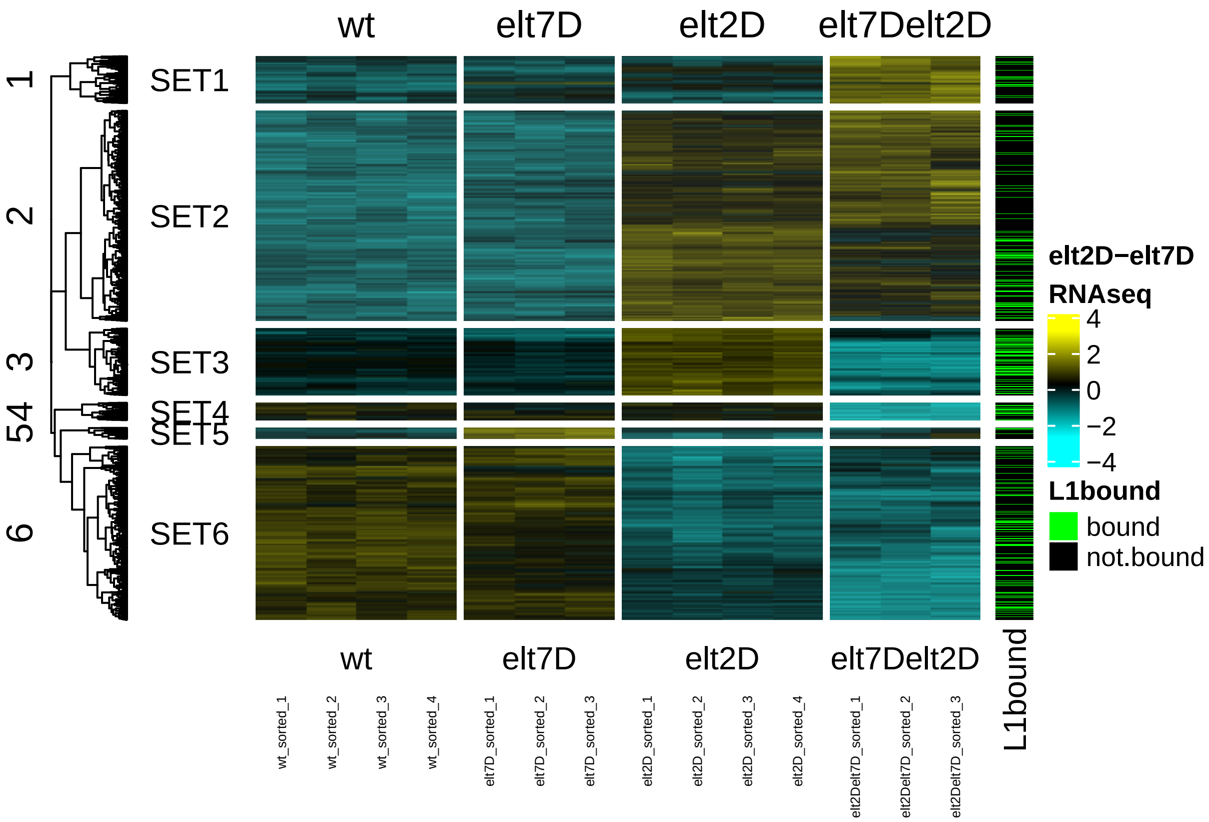
```
## [1] "data.frame"
```

```r
elt2_L1_anno %>% head()
```

```
##   elt2_detected_in_L1       WBGeneID
## 1           not.bound WBGene00000007
## 2               bound WBGene00000008
## 3           not.bound WBGene00000009
## 4           not.bound WBGene00000013
## 5           not.bound WBGene00000016
## 6           not.bound WBGene00000017
```

Incorporate this into a heatmap annotation

```r
Ha_L1chip <-
  Ha_rotate + rowAnnotation(L1bound = elt2_L1_anno$elt2_detected_in_L1,
                            col = list(L1bound = c(
                              "bound" = "green", "not.bound" = "black"
                            )))
Ha_L1chip
```



```
# pdf("./03_plots/200428_wt_elt2singledouble_L1elt2bound_heatmap.pdf", height = 5, width = 5)
# Ha_L1chip
# dev.off()
```

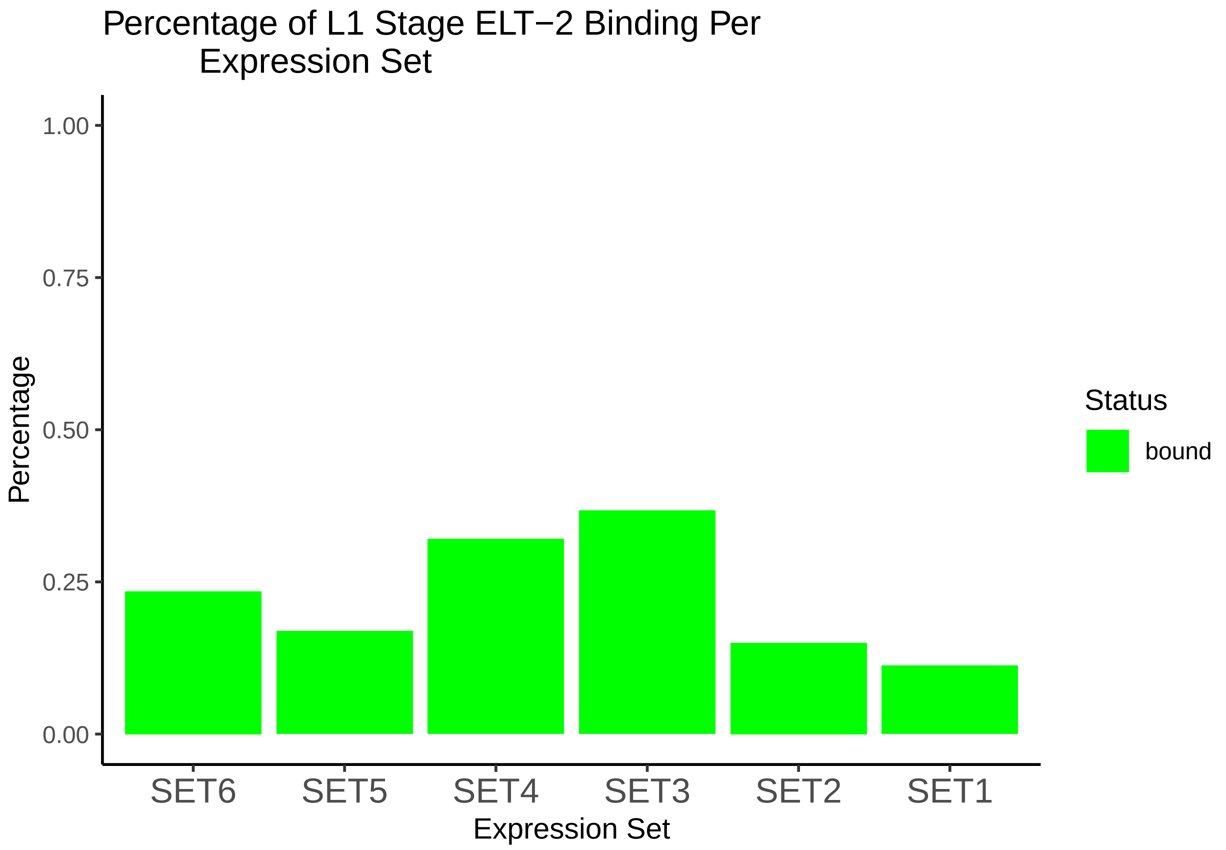
Determine enrichment of ELT-2 binding during L1 stage.

```
expression_L1_binding <-
  merge(elt2_L1_anno, gene_expression_clusters, by = "WBGeneID")
expression_L1_binding %>% head
```

```
##         WBGeneID elt2_detected_in_L1 Expression.Cluster
## 1 WBGene00000007           not.bound               SET6
## 2 WBGene00000008               bound               SET6
## 3 WBGene00000009           not.bound               SET3
## 4 WBGene00000013           not.bound               SET1
## 5 WBGene00000016           not.bound               SET1
## 6 WBGene00000017           not.bound               SET1
```

```
clust_L1bound_counts <-
  table(
    expression_L1_binding$Expression.Cluster,
    expression_L1_binding$elt2_detected_in_L1
  )

clust_L1bound_prop <- prop.table(clust_L1bound_counts, 1)

clust_L1bound_prop_ggplot <- as.data.frame(clust_L1bound_prop)

colnames(clust_L1bound_prop_ggplot) <- c("SET", "Status", "Freq")
```

```r
clust_L1bound_prop_ggplot$Status <-
  factor(clust_L1bound_prop_ggplot$Status,
         levels = c("not.bound", "bound"))

clust_L1bound_prop_ggplot$SET <-
  factor(
    clust_L1bound_prop_ggplot$SET,
    levels = c("SET6", "SET5", "SET4", "SET3", "SET2", "SET1")
  )

clust_L1bound_colors <- c("bound" = "green", "not.bound" = "black")

l1bound_percents <-
  ggplot(
    clust_L1bound_prop_ggplot %>% filter(Status == "bound"),
    aes(
      x = SET,
      y = Freq,
      fill = Status,
      order = Status
    )
  ) +
  geom_bar(stat = "identity") +
  scale_color_manual(values = clust_L1bound_colors,
                     aesthetics = c("color", "fill")) +
  ggtitle("Percentage of L1 Stage ELT-2 Binding Per
          Expression Set") +
  xlab("Expression Set") +
  ylab("Percentage") +
  theme_classic() +
  theme(axis.text.x = element_text(size = 13)) +
  ylim(0, 1)

l1bound_percents
```



```
# ggsave("./03_plots/200428_proportion_of_l1elt2_per_expression_cluster.pdf", height = 2, width = 5)
```

Plot the number of "bound" vs "not.bound" per cluster

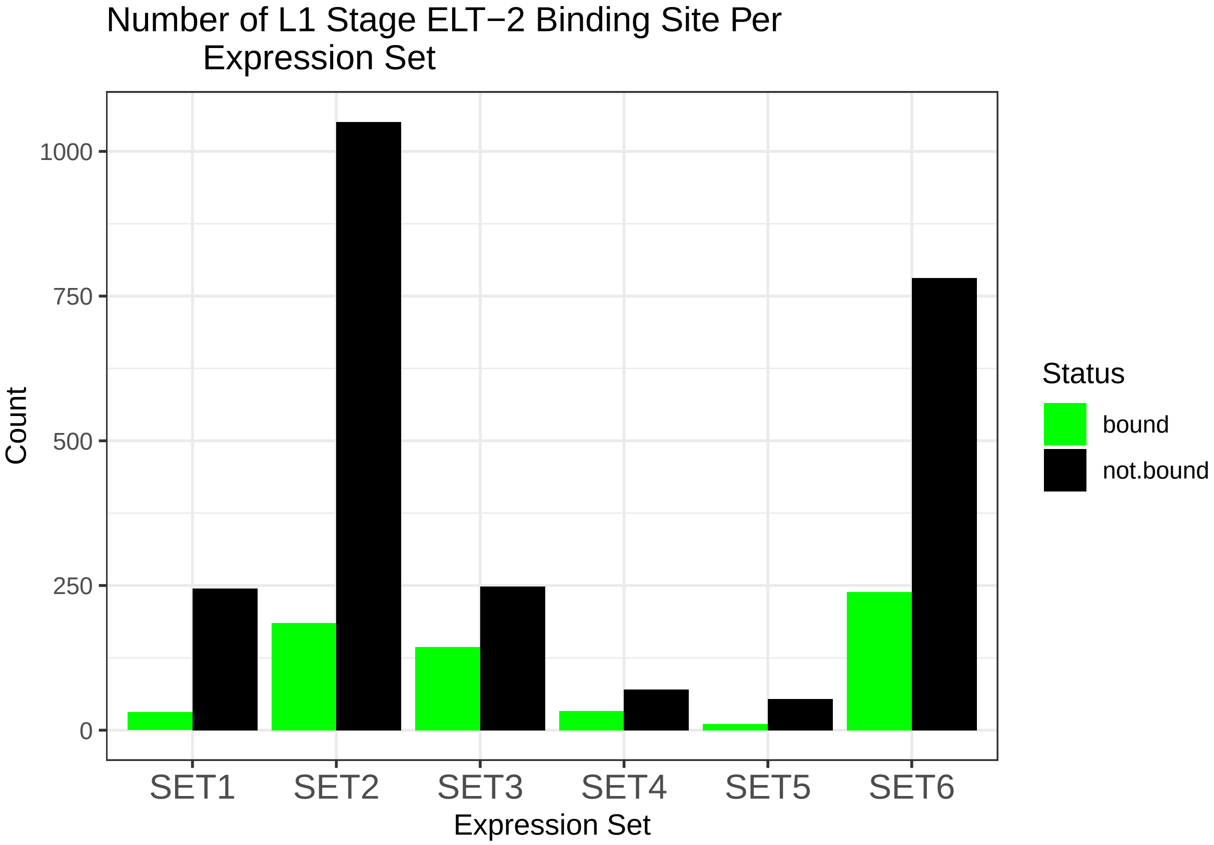
```
clust_L1bound_counts %>% head
```

```
##
##         bound not.bound
##   SET1     31       245
##   SET2    185      1051
##   SET3    144       248
##   SET4     33        70
##   SET5     11        54
##   SET6    239       781
```

```
clust_L1bound_counts_ggplot <- as.data.frame(clust_L1bound_counts)
colnames(clust_L1bound_counts_ggplot) <- c("SET", "Status", "Freq")
clust_L1bound_counts_ggplot
```

```
##     SET    Status Freq
## 1  SET1     bound   31
## 2  SET2     bound  185
## 3  SET3     bound  144
## 4  SET4     bound   33
## 5  SET5     bound   11
## 6  SET6     bound  239
## 7  SET1 not.bound  245
## 8  SET2 not.bound 1051
## 9  SET3 not.bound  248
## 10 SET4 not.bound   70
## 11 SET5 not.bound   54
## 12 SET6 not.bound  781
```

```
ggplot(clust_L1bound_counts_ggplot,
       aes(x = SET,
           y = Freq,
           fill = Status)) +
  geom_bar(stat = "identity", position = "dodge") +
  scale_color_manual(values = clust_L1bound_colors,
                     aesthetics = c("color", "fill")) +
  ggtitle("Number of L1 Stage ELT-2 Binding Site Per
          Expression Set") +
  xlab("Expression Set") +
  ylab("Count") +
  theme_bw() +
  theme(axis.text.x = element_text(size = 13))
```



```
# ggsave("./03_plots/200428_number_of_l1elt2_per_expression_cluster.pdf", height = 2, width = 5)
```

Use hypergeometric to determine enrichment of L1 stage binding per expression cluster.

Does any set have more ELT-2 bound relative to all the sets?

```
x <- 1
clust_l1bound_dhyper <- data.frame()
for (i in rownames(clust_L1bound_counts)) {
  pval <- phyper(
    q = clust_L1bound_counts[x, 1],
    m = clust_L1bound_counts[x, 1] + clust_L1bound_counts[x, 2],
    k = colSums(clust_L1bound_counts)[1],
    n = colSums(clust_L1bound_counts)[1] + colSums(clust_L1bound_counts)[2]
  )
  toappend <- data.frame(SET = i, pval = (1 - pval))
  clust_l1bound_dhyper <- bind_rows(clust_l1bound_dhyper, toappend)
  x <- x + 1
}
```

```
## Warning in bind_rows_(x, .id): Unequal factor levels: coercing to character

## Warning in bind_rows_(x, .id): binding character and factor vector, coercing
## into character vector

## Warning in bind_rows_(x, .id): binding character and factor vector, coercing
## into character vector

## Warning in bind_rows_(x, .id): binding character and factor vector, coercing
## into character vector

## Warning in bind_rows_(x, .id): binding character and factor vector, coercing
## into character vector

## Warning in bind_rows_(x, .id): binding character and factor vector, coercing
## into character vector

## Warning in bind_rows_(x, .id): binding character and factor vector, coercing
## into character vector
```

```
clust_l1bound_dhyper
```

```
##    SET         pval
## 1 SET1 9.998211e-01
## 2 SET2 4.279558e-01
## 3 SET3 0.000000e+00
## 4 SET4 1.235659e-03
## 5 SET5 6.982102e-01
## 6 SET6 8.215650e-15
```

```
clust_l1bound_dhyper %>% mutate(bool = pval < 0.05)
```

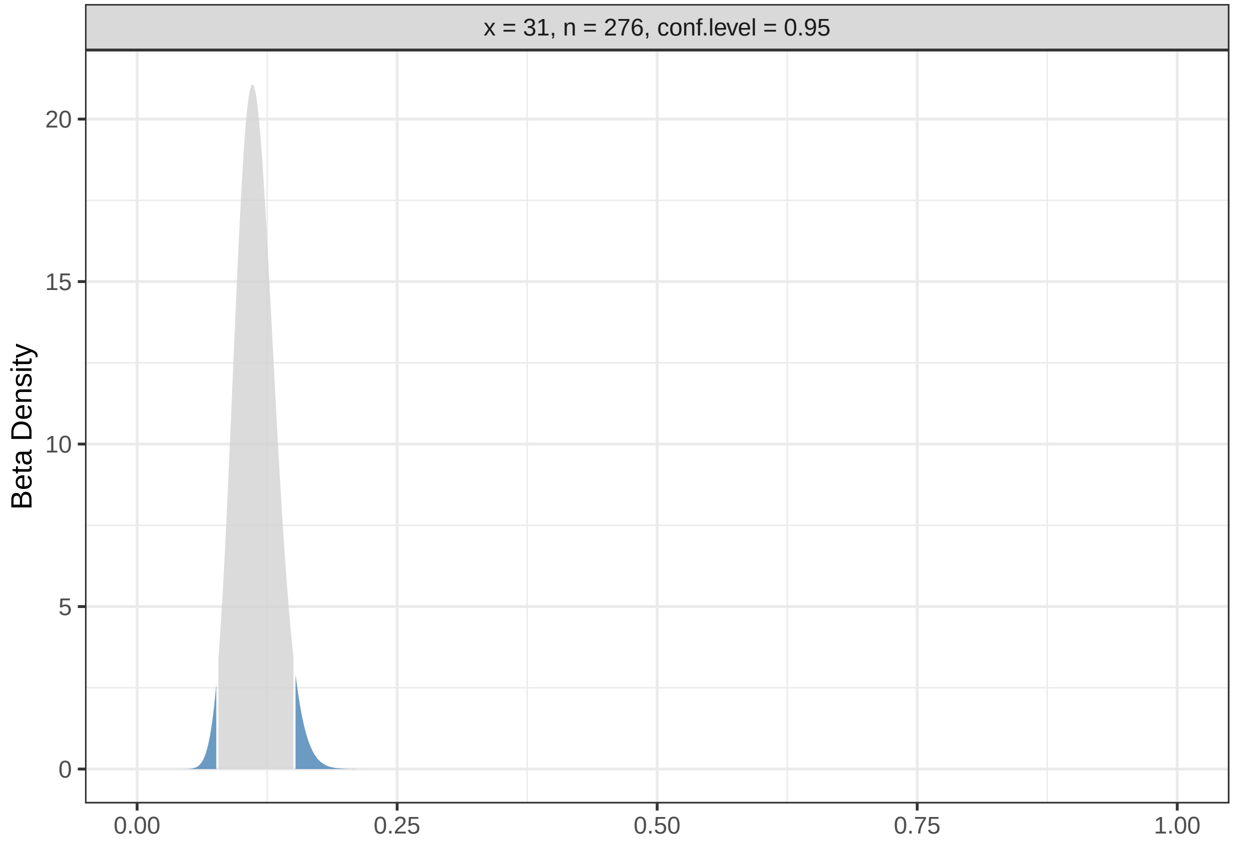
```
##    SET         pval  bool
## 1 SET1 9.998211e-01 FALSE
## 2 SET2 4.279558e-01 FALSE
## 3 SET3 0.000000e+00  TRUE
## 4 SET4 1.235659e-03  TRUE
## 5 SET5 6.982102e-01 FALSE
## 6 SET6 8.215650e-15  TRUE
```

Use binomial to determine enrichment of L1 stage binding per expression cluster.

```
xbinom = clust_L1bound_counts[1, 1]
nbinom = clust_L1bound_counts[1, 1] + clust_L1bound_counts[1, 2]
binom.bayes(xbinom, nbinom)
```

```
##   method  x   n shape1 shape2      mean      lower     upper  sig
## 1  bayes 31 276   31.5  245.5 0.1137184 0.07746624 0.1515495 0.05
```

```
binom.bayes.densityplot(binom.bayes(xbinom, nbinom))
```



```
# any percentage outside this confidence interval is "significant"
```

Use the binomial test to determine if the different expression clusters are enriched or depleted for ELT-2 binding.

Use binom.test and first two a two-tailed test.

First calculate the proportion of bound genes over the total number of genes in the analysis.

```
proportion = as.numeric(colSums(clust_L1bound_counts)[1]) /
  as.numeric(colSums(clust_L1bound_counts)[1] + colSums(clust_L1bound_counts)[2])
proportion
```

```
## [1] 0.207956
```

Use custom function ctable_binom() to calculate p-vaule for each set.

```
l1bound_binom <- ctable_binom(clust_L1bound_counts, "two.sided")
```

```
##    Set         pval conf.lower conf.upper  bool
## 1 SET1 4.024887e-05 0.07760327  0.1556278  TRUE
## 2 SET2 1.999465e-07 0.13023469  0.1708005  TRUE
## 3 SET3 3.905529e-13 0.31950900  0.4172047  TRUE
## 4 SET4 7.240238e-03 0.23184100  0.4195741  TRUE
## 5 SET5 5.413473e-01 0.08762605  0.2826562 FALSE
## 6 SET6 4.082629e-02 0.20862677  0.2615436  TRUE
```

This says that all sets but SET5 have a significant difference in genes bound compared to the entire dataset.

Now use the less or greater argument of binom.test to see if there is more or less binding.

First two less

```
ctable_binom(ctable = clust_L1bound_counts, alt = "less")
```

```
##    Set         pval conf.lower conf.upper  bool
## 1 SET1 2.069242e-05          0  0.1485670  TRUE
## 2 SET2 9.704061e-08          0  0.1673893  TRUE
## 3 SET3 1.000000e+00          0  0.4093251 FALSE
## 4 SET4 9.973903e-01          0  0.4041263 FALSE
## 5 SET5 2.752156e-01          0  0.2645358 FALSE
## 6 SET6 9.816208e-01          0  0.2571740 FALSE
```

This says that set 1 and 2 have less ELT-2 binding compared to the entire dataset.

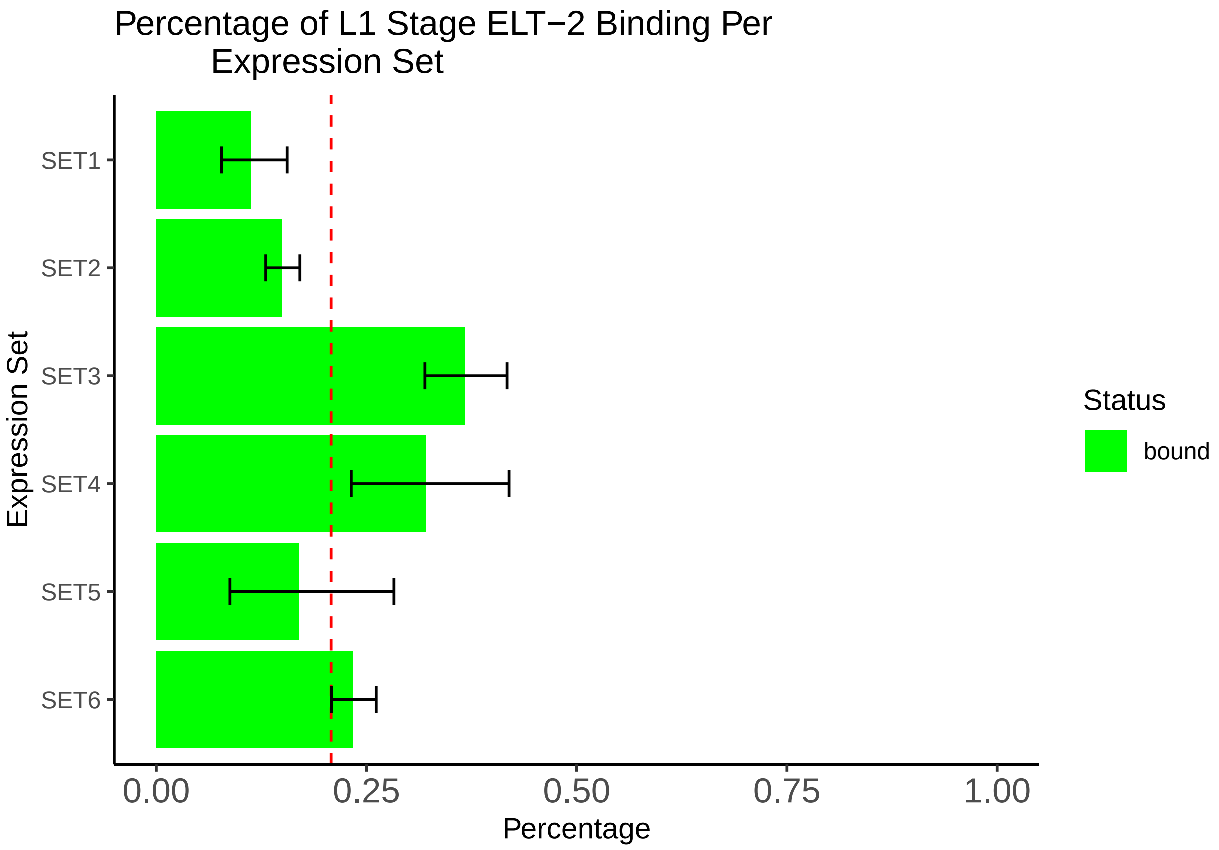
Now try greater.

```
ctable_binom(clust_L1bound_counts, "greater")
```

```
##    Set         pval conf.lower conf.upper  bool
## 1 SET1 9.999904e-01 0.08247951          1 FALSE
## 2 SET2 9.999999e-01 0.13321068          1 FALSE
## 3 SET3 3.121652e-13 0.32686852          1  TRUE
## 4 SET4 5.031706e-03 0.24460667          1  TRUE
## 5 SET5 8.207255e-01 0.09790359          1 FALSE
## 6 SET6 2.203126e-02 0.21259691          1  TRUE
```

Draw line on percentage plots to indicate background percentage of L1 binding.

```
l1bound_percents +
  geom_hline(yintercept = proportion,
             color = "red",
             linetype = "dashed") +
  geom_errorbar(
    ymax = l1bound_binom$conf.upper,
    ymin = l1bound_binom$conf.lower,
    width = 0.25
  ) +
  coord_flip()
```



```
ggsave(
  "./03_plots/200504_percentage_l1bound_per_expression_cluster.pdf",
  width = 4,
  height = 5
)
```

Use the hypergeometric test to determine: Are changing genes (all sets) enriched for L1 binding?

```
N <- 20470
k <- nrow(elt2_detected_in_L1)
x3 <- as.numeric(colSums(clust_L1bound_counts)[1])
m <-
  as.numeric(colSums(clust_L1bound_counts)[1] + colSums(clust_L1bound_counts)[2])
dhyper(x3, m, N, k)
```

```
## [1] 1.05078e-78
```

Compute pairwise fisher’s exact tests

```
fisher.multcomp(clust_L1bound_counts, p.method = "bonferroni")
```

```
##
##          Pairwise comparisons using Fisher's exact test for count data
##
## data:  clust_L1bound_counts
##
##            SET1      SET2      SET3   SET4 SET5
## SET2 1.000e+00         -         -      -    -
## SET3 3.796e-13 7.087e-18         -      -    -
## SET4 7.412e-05 5.569e-04 1.000e+00      -    -
## SET5 1.000e+00 1.000e+00 2.535e-02 0.4808    -
## SET6 5.820e-05 5.075e-06 1.433e-05 0.8164    1
##
## P value adjustment method: bonferroni
```

```
fisher.multcomp(clust_L1bound_counts, p.method = "bonferroni")$p.value < 0.05
```

```
##       SET1  SET2  SET3  SET4  SET5
## SET2 FALSE    NA    NA    NA    NA
## SET3  TRUE  TRUE    NA    NA    NA
## SET4  TRUE  TRUE FALSE    NA    NA
## SET5 FALSE FALSE  TRUE FALSE    NA
## SET6  TRUE  TRUE  TRUE FALSE FALSE
```

Row annotation of ELT-2 Binding Pattern Clusters

```
dynamic_counts_matrix_scaled %>% dim
```

```
## [1] 3092   14
```

```
chip_annotation <- dynamic_counts_matrix_scaled %>%
  as.data.frame.matrix() %>%
  rownames_to_column() %>%
  left_join(rownames_to_column(binding_cluster_gene_counts), by = "rowname") %>%
  select(rowname,
         NotChanging,
         L3high,
         Increasing,
         PostEmbryonic,
         LEspecific) %>%
  replace_na(list(
    NotChanging = 0,
    L3high = 0,
    Increasing = 0,
    PostEmbryonic = 0,
    LEspecific = 0
  ))

unique(rownames(dynamic_counts_matrix_scaled) == chip_annotation$rowname)
```

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

```
nrow(dynamic_counts_matrix_scaled) == nrow(chip_annotation)
```

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

```
Ha_L1chip_bindcluster <- Ha_L1chip +
  rowAnnotation(NotChanging = chip_annotation$NotChanging) +
  rowAnnotation(LESpecific = chip_annotation$LEspecific) +
  rowAnnotation(Increasing = chip_annotation$Increasing) +
  rowAnnotation(PostEmbryonic = chip_annotation$PostEmbryonic) +
  rowAnnotation(L3high = chip_annotation$L3high)
```

Have the colors match plot from David.

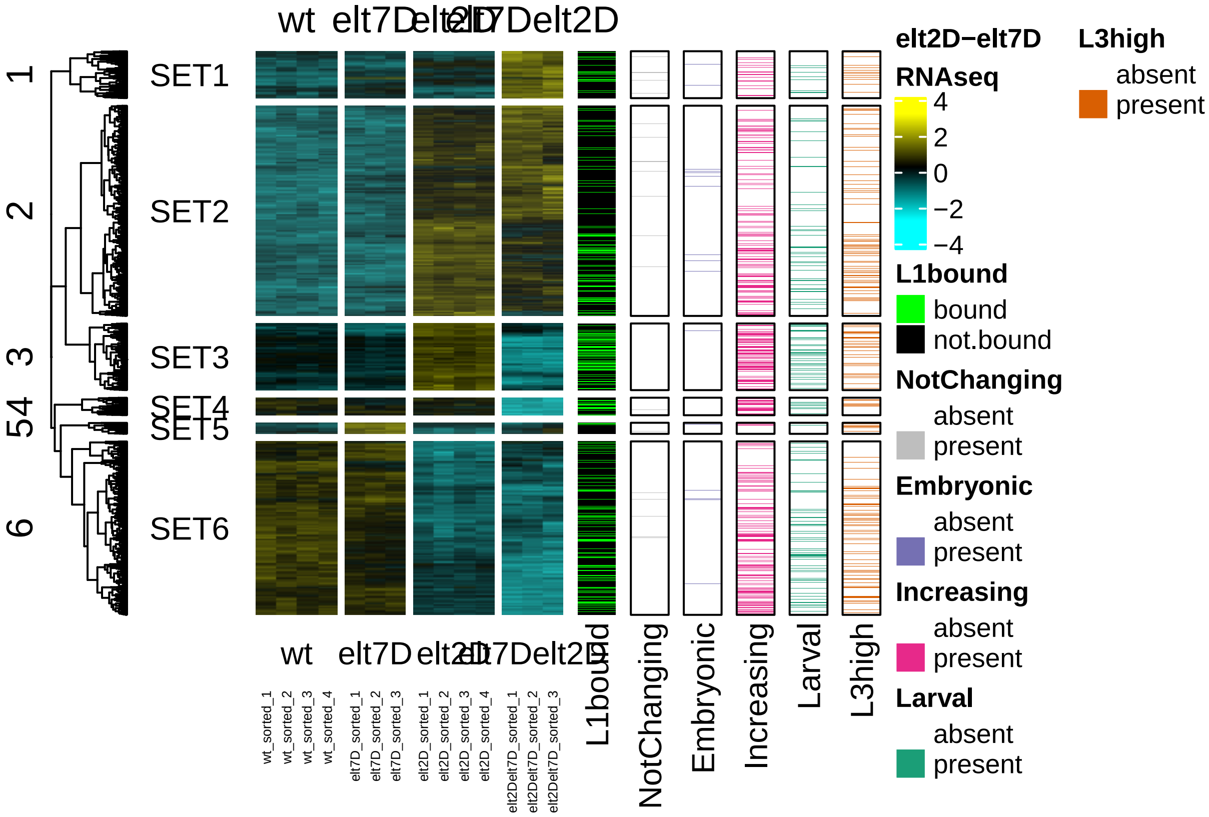
```r
cluster_colors <-
  data.frame(
    class = c(
      "LEspecific",
      "PostEmbryonic",
      "Increasing",
      "L3high",
      "NotChanging"
    ),
    val = c("#7570B3", "#1B9E77", "#E7298A", "#D95F02", "grey")
  )
cluster_colors$class <-
  factor(
    x = cluster_colors$class,
    levels = c(
      "LEspecific",
      "PostEmbryonic",
      "Increasing",
      "L3high",
      "NotChanging"
    )
  )
```

Convert ChIP binding clusters to a present/absence list.

```r
chip_annotation_present_absent <- chip_annotation %>%
  mutate(
    NotChanging = if_else(
      condition = chip_annotation$NotChanging == 0,
      true = "absent",
      false = "present"
    )
  ) %>%
  mutate(L3high = if_else(
    condition = chip_annotation$L3high == 0,
    true = "absent",
    false = "present"
  )) %>%
  mutate(
    Increasing = if_else(
      condition = chip_annotation$Increasing == 0,
      true = "absent",
      false = "present"
    )
  ) %>%
  mutate(
    PostEmbryonic = if_else(
      condition = chip_annotation$PostEmbryonic == 0,
      true = "absent",
      false = "present"
    )
  ) %>%
  mutate(
    LEspecific = if_else(
      condition = chip_annotation$LEspecific == 0,
      true = "absent",
      false = "present"
    )
  )
```

Plot the heatmap with presence/absence.

```
Ha_L1chip_clusterchip <- Ha_L1chip +
  rowAnnotation(
    NotChanging = chip_annotation_present_absent$NotChanging,
    col = list(NotChanging = c(
      "absent" = "white", "present" = "grey"
    )),
    border = TRUE
  ) +
  rowAnnotation(
    Embryonic = chip_annotation_present_absent$LEspecific,
    col = list(Embryonic = c(
      "absent" = "white", "present" = "#7570B3"
    )),
    border = TRUE
  ) +
  rowAnnotation(
    Increasing = chip_annotation_present_absent$Increasing,
    col = list(Increasing = c(
      "absent" = "white", "present" = "#E7298A"
    )),
    border = TRUE
  ) +
  rowAnnotation(
    Larval = chip_annotation_present_absent$PostEmbryonic,
    col = list(Larval = c(
      "absent" = "white", "present" = "#1B9E77"
    )),
    border = TRUE
  ) +
  rowAnnotation(L3high = chip_annotation_present_absent$L3high,
                col = list(L3high = c(
                  "absent" = "white", "present" = "#D95F02"
                )),
                border = TRUE)

Ha_L1chip_clusterchip
```



```
# pdf("./03_plots/200504_wt_elt2singledouble_L1elt2bound_elt2bindclusters_heatmap.pdf", height = 5, wid
# Ha_L1chip_clusterchip
# dev.off()
```

Plot percentage of expression cluster group having binding pattern assignment.

```
exprclust_bindclust <-
  merge(
    gene_expression_clusters,
    chip_annotation_present_absent,
    by.x = "WBGeneID",
    by.y = "rowname"
  )

exprclust_bindclust %>% head
```

```
##         WBGeneID Expression.Cluster NotChanging L3high Increasing PostEmbryonic
## 1 WBGene00000007               SET6      absent absent    present        absent
## 2 WBGene00000008               SET6      absent absent    present        absent
## 3 WBGene00000009               SET3      absent absent    present       present
## 4 WBGene00000013               SET1      absent absent     absent        absent
## 5 WBGene00000016               SET1      absent absent     absent        absent
## 6 WBGene00000017               SET1      absent absent     absent        absent
##   LEspecific
## 1     absent
## 2     absent
## 3     absent
## 4     absent
## 5     absent
## 6     absent
```

What is the percentage of genes with annotated ELT2 binding clusters per expression dataset?

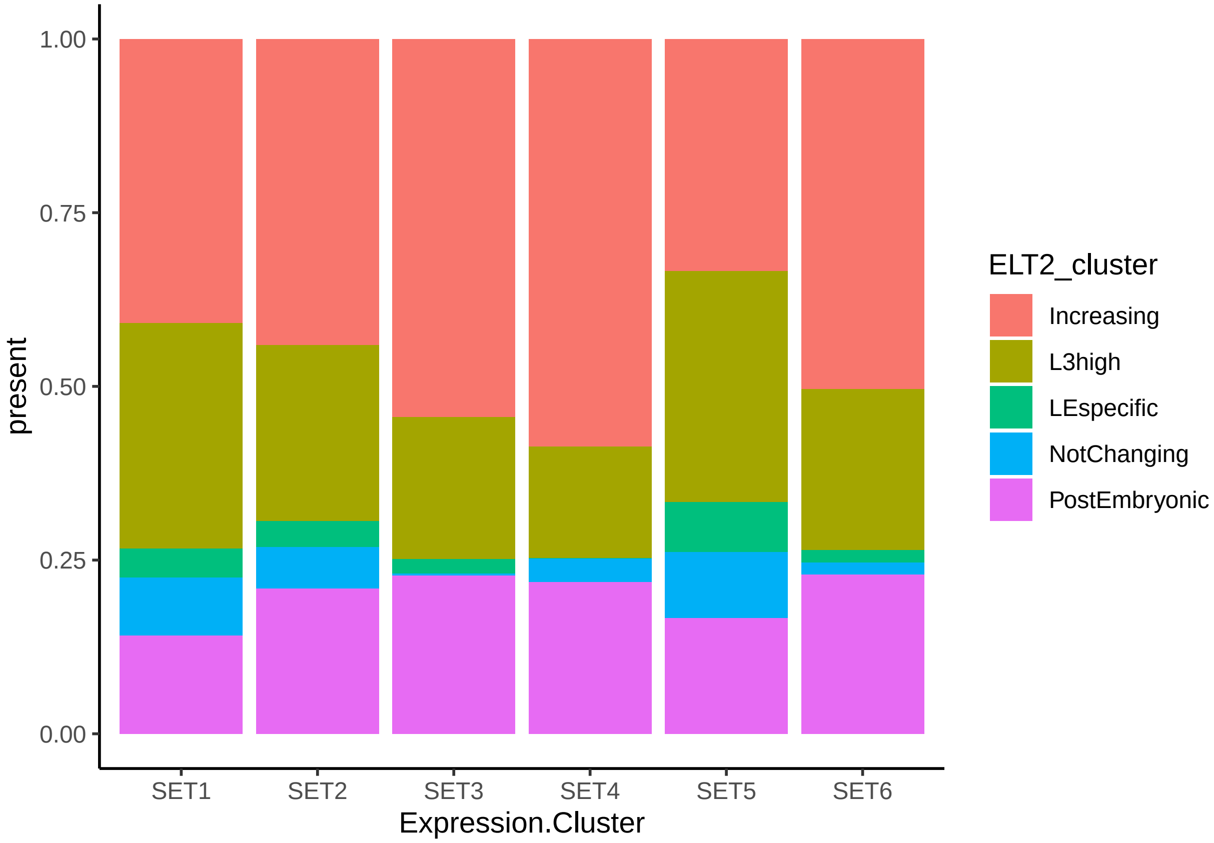
Make a dataframe that addresses the question:

```
elt2_cluster_names <-
  c("NotChanging",
    "L3high",
    "Increasing",
    "PostEmbryonic",
    "LEspecific")

expressionSet_per_BindingCluster <- data.frame()
for (i in elt2_cluster_names) {
  toappend <-
    table(exprclust_bindclust$Expression.Cluster,
          exprclust_bindclust[[i]]) %>%
    as.data.frame.matrix() %>%
    rownames_to_column(var = "Expression.Cluster") %>%
    mutate(ELT2_cluster = i,
           percent = present / (present + absent))
  expressionSet_per_BindingCluster <-
    bind_rows(expressionSet_per_BindingCluster, toappend)
}
expressionSet_per_BindingCluster
```

```
##    Expression.Cluster absent present  ELT2_cluster    percent
## 1                SET1    266      10   NotChanging 0.03623188
## 2                SET2   1199      37   NotChanging 0.02993528
## 3                SET3    391       1   NotChanging 0.00255102
## 4                SET4    100       3   NotChanging 0.02912621
## 5                SET5     61       4   NotChanging 0.06153846
## 6                SET6   1009      11   NotChanging 0.01078431
## 7                SET1    237      39        L3high 0.14130435
## 8                SET2   1079     157        L3high 0.12702265
## 9                SET3    322      70        L3high 0.17857143
## 10               SET4     89      14        L3high 0.13592233
## 11               SET5     51      14        L3high 0.21538462
## 12               SET6    872     148        L3high 0.14509804
## 13               SET1    227      49    Increasing 0.17753623
## 14               SET2    964     272    Increasing 0.22006472
## 15               SET3    206     186    Increasing 0.47448980
## 16               SET4     52      51    Increasing 0.49514563
## 17               SET5     51      14    Increasing 0.21538462
## 18               SET6    700     320    Increasing 0.31372549
## 19               SET1    259      17 PostEmbryonic 0.06159420
## 20               SET2   1107     129 PostEmbryonic 0.10436893
## 21               SET3    314      78 PostEmbryonic 0.19897959
## 22               SET4     84      19 PostEmbryonic 0.18446602
## 23               SET5     58       7 PostEmbryonic 0.10769231
## 24               SET6    874     146 PostEmbryonic 0.14313725
## 25               SET1    271       5    LEspecific 0.01811594
## 26                   SET2    1213       23     LEspecific 0.01860841
## 27                   SET3     385        7     LEspecific 0.01785714
## 28                   SET4     103        0     LEspecific 0.00000000
## 29                   SET5      62        3     LEspecific 0.04615385
## 30                   SET6    1009       11     LEspecific 0.01078431
```

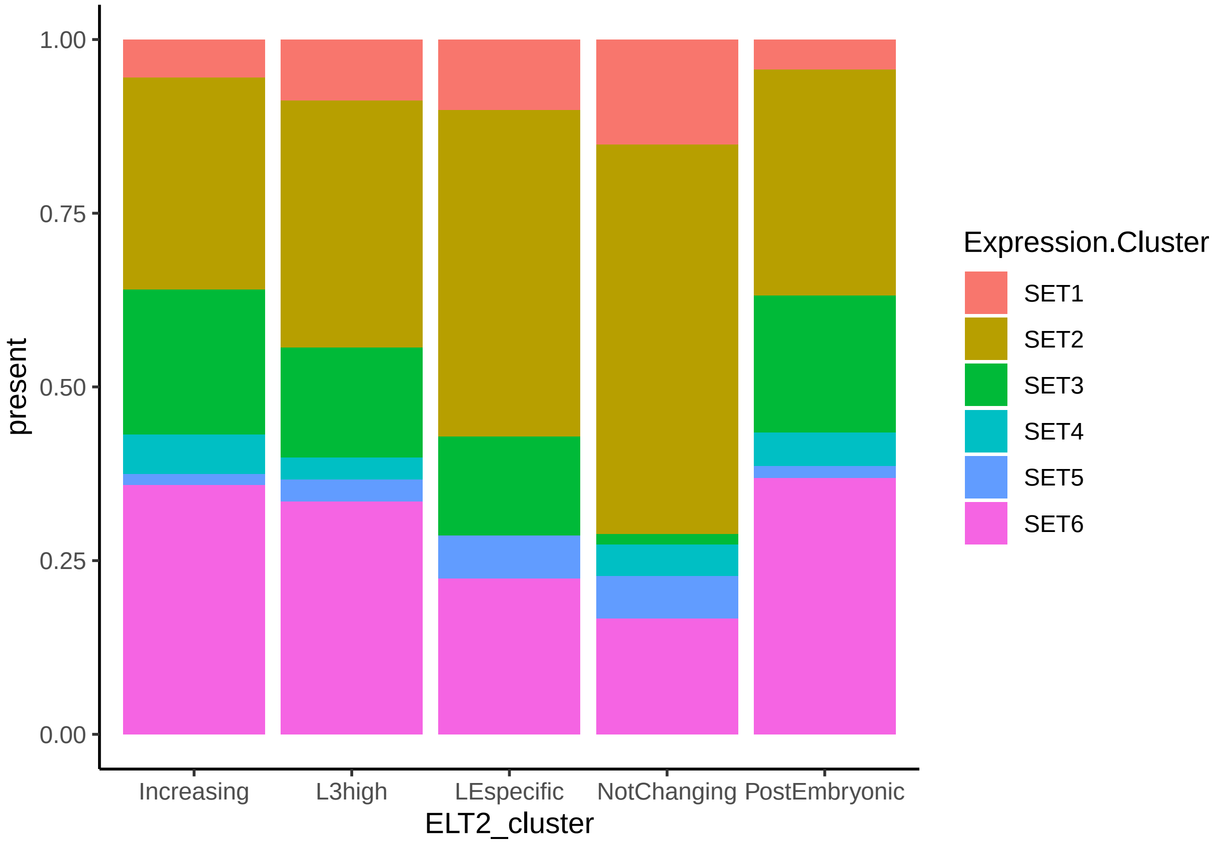
Make a plot that addresses the question:

```
ggplot(
  expressionSet_per_BindingCluster,
  aes(x = Expression.Cluster, y = present, fill = ELT2_cluster)
) +
  geom_bar(stat = "identity", position = "fill") +
  theme_classic()
```



What is the percentage of genes within each Expression Set that are associated with an ELT-2 binding cluster?

```
ggplot(
  expressionSet_per_BindingCluster,
  aes(x = ELT2_cluster, y = present, fill = Expression.Cluster)
) +
  geom_bar(stat = "identity", position = "fill") +
  theme_classic()
```



Make a series of horizontal barplots with percentage of ELT-2 binding cluster per expression cluster.

First, calculate the percentage of each ELT-2 binding category against the total dataset.

```
percent_bound_per_ELT2_cluster <-
  expressionSet_per_BindingCluster %>% group_by(ELT2_cluster) %>% summarise(percent = sum(present) /
                                                                              nrow(dynamic_counts_matri
```

Next calculate the the 95% Confidence Interval with the Bionomial Test.

```
expressionSet_per_BindingCluster %>% group_by(Expression.Cluster, ELT2_cluster) %>% summarise(percent =
                                                                                                (presen
```

```
## # A tibble: 30 x 3
## # Groups:   Expression.Cluster [6]
##    Expression.Cluster ELT2_cluster  percent
##    <chr>              <chr>           <dbl>
##  1 SET1               Increasing     0.178
##  2 SET1               L3high         0.141
##  3 SET1               LEspecific     0.0181
##  4 SET1               NotChanging    0.0362
##  5 SET1               PostEmbryonic  0.0616
##  6 SET2               Increasing     0.220
##  7 SET2               L3high         0.127
##  8 SET2               LEspecific     0.0186
##  9 SET2               NotChanging    0.0299
## 10 SET2               PostEmbryonic  0.104
## # ... with 20 more rows
```

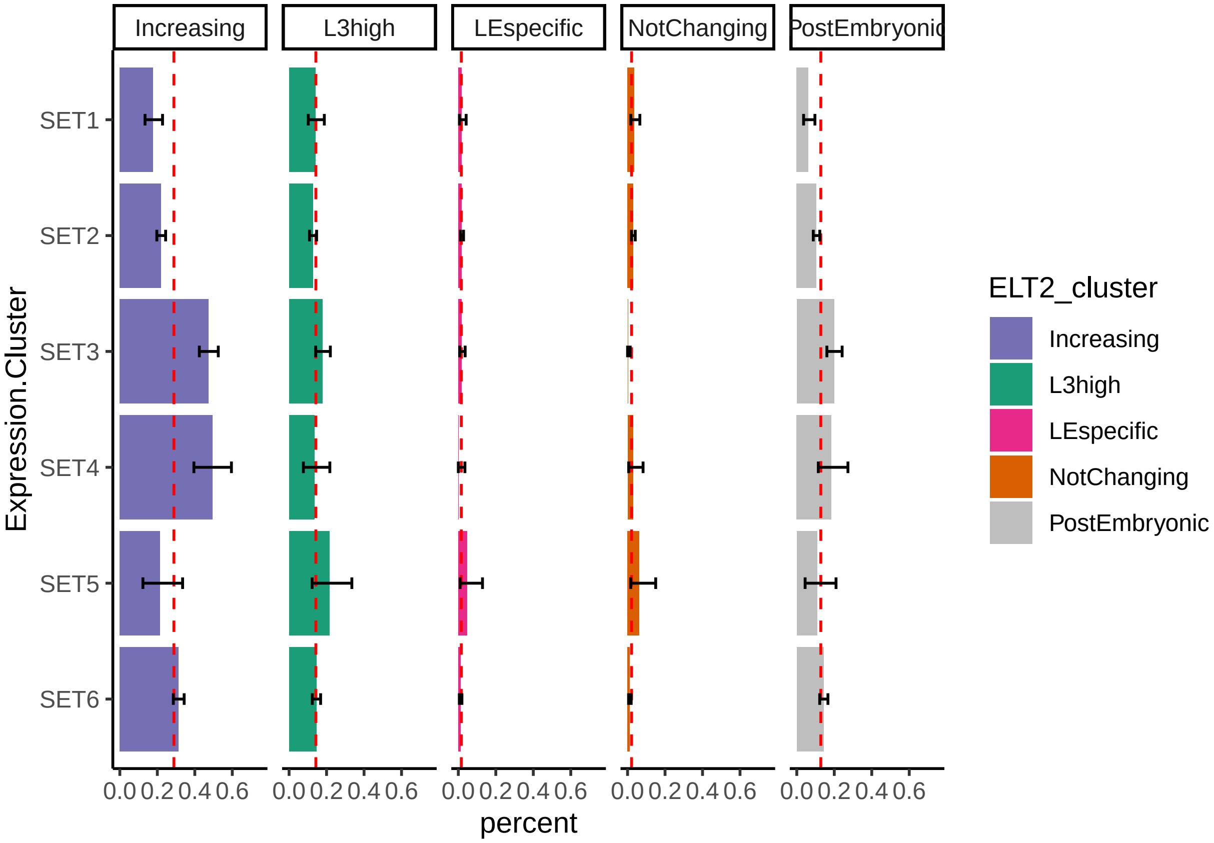
```
? binom.test
```

Calculate the binomial pvalue and confidence intervals.

```
# Add a column for the background percentage of ELT2 binding clusters per the whole expression dataset
expression_binding_stats <-
  expressionSet_per_BindingCluster %>% group_by(ELT2_cluster) %>% mutate(background_percent = sum(prese
                                                                                  (sum(present) + sum(absent))

# Use binomia.test to calculate pvalue and confidence intervales for the percentage of ELT2 binding clu
expression_binding_stats <- expression_binding_stats %>%
  group_by(ELT2_cluster, Expression.Cluster) %>%
  mutate(
    pval = binom.test(
      x = c(present, absent),
      n = present + absent,
      p = background_percent,
      alternative = "two.sided"
    )$p.value,
    conf.upper = binom.test(
      x = c(present, absent),
      n = present + absent,
      p = background_percent,
      alternative = "two.sided"
    )$conf.int[2],
    conf.lower = binom.test(
      x = c(present, absent),
      n = present + absent,
      p = background_percent,
      alternative = "two.sided"
    )$conf.int[1]
  )

expression_binding_stats$Expression.Cluster <-
  factor(
    expression_binding_stats$Expression.Cluster,
    levels = c("SET6", "SET5", "SET4", "SET3", "SET2", "SET1")
  )

expression_binding_stats %>% head()
```

```
## # A tibble: 6 x 9
## # Groups:   ELT2_cluster, Expression.Cluster [6]
##   Expression.Clus~ absent present ELT2_cluster percent background_perc~    pval
##   <fct>             <int>   <int> <chr>          <dbl>            <dbl>   <dbl>
## 1 SET1                266      10 NotChanging  0.0362           0.0213 0.0927
## 2 SET2               1199      37 NotChanging  0.0299           0.0213 0.0479
## 3 SET3                391       1 NotChanging  0.00255          0.0213 0.00430
## 4 SET4                100       3 NotChanging  0.0291           0.0213 0.486
## 5 SET5                 61       4 NotChanging  0.0615           0.0213 0.0504
## 6 SET6               1009      11 NotChanging  0.0108           0.0213 0.0166
## # ... with 2 more variables: conf.upper <dbl>, conf.lower <dbl>
```

```
ggplot(expression_binding_stats,
       aes(x = Expression.Cluster,
           y = percent, fill = ELT2_cluster)) +
  geom_bar(stat = "identity") +
  scale_y_continuous(limits = c(0, 0.75)) +
  theme_classic() +
  geom_hline(
    data = percent_bound_per_ELT2_cluster,
    color = "red",
    linetype = "dashed",
    aes(yintercept = percent)
  ) +
  geom_errorbar(
    ymax = expression_binding_stats$conf.upper,
    ymin = expression_binding_stats$conf.lower,
    width = 0.1
  ) +
  coord_flip() +
  facet_grid(. ~ ELT2_cluster) +
  scale_fill_manual(values = as.character(cluster_colors$val))
```



```
# ggsave(filename = "./03_plots/200511_Percent_of_ELT2bindClust_per_ExpressionClust.pdf")
```

L1 Intestine expression row annotation

Import RWC24 L1 Intestine results table

```
RWC24_res <-
  read.csv(
    "../RWC24_L1_Intestine_RNAseq/04_DEseq2/200511_L1_intestine_FACS_gut_vs_gutless.csv",
    row.names = 1
  )
RWC24_res <- RWC24_res %>% rownames_to_column(var = "WBGeneID")
head(RWC24_res)
```

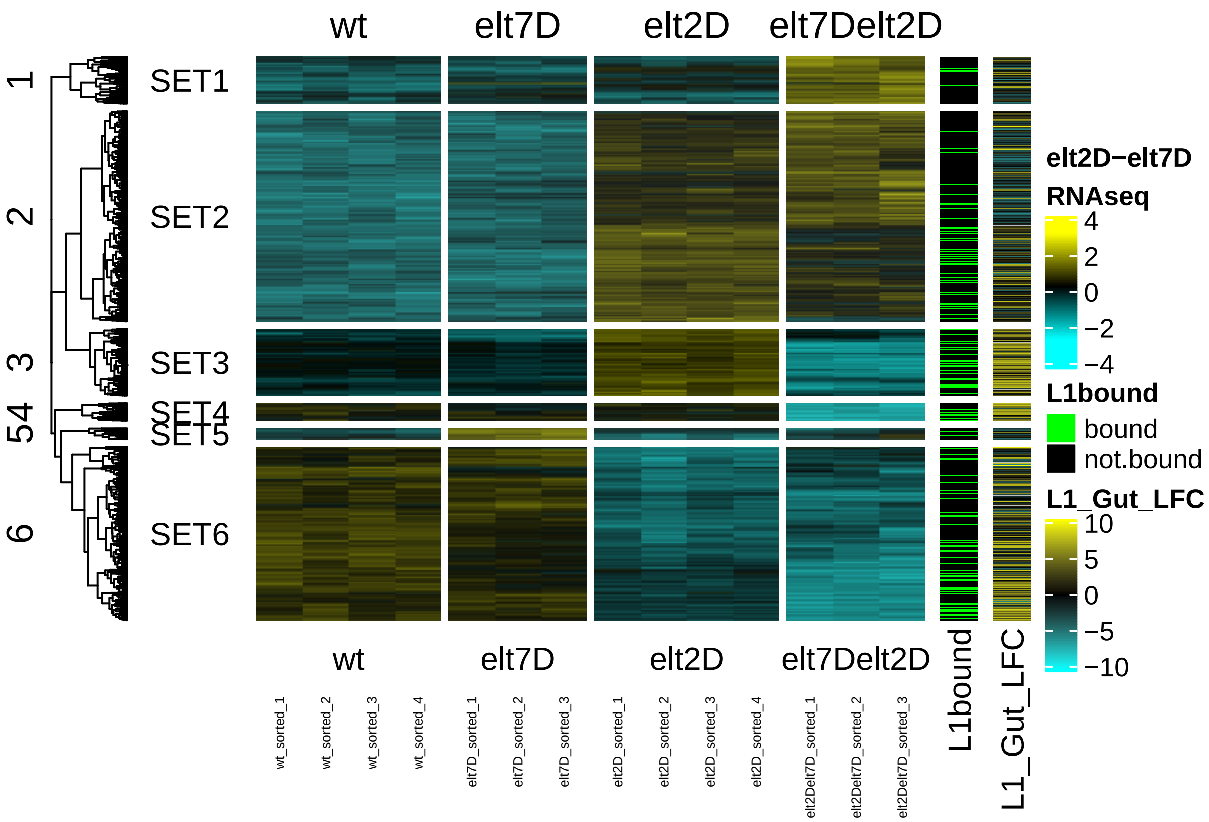
```
##         WBGeneID baseMean log2FoldChange lfcSE stat pvalue padj
## 1 WBGene00014451        0             NA    NA   NA     NA   NA
## 2 WBGene00010957        0             NA    NA   NA     NA   NA
## 3 WBGene00010958        0             NA    NA   NA     NA   NA
## 4 WBGene00014452        0             NA    NA   NA     NA   NA
## 5 WBGene00014453        0             NA    NA   NA     NA   NA
## 6 WBGene00014454        0             NA    NA   NA     NA   NA
```

Select log fold change data for row annotation.

```
L1_gut_lfc <-
  dynamic_counts_matrix_scaled %>% as.data.frame.matrix() %>% rownames_to_column(var = "WBGeneID") %>%
```

Add to heatmap as a row annotation.

```
col_fun = colorRamp2(c(-10, 0, 10), c("cyan", "black", "yellow"))

Ha_L1chip_L1gutLFC <-
  Ha_L1chip + rowAnnotation(L1_Gut_LFC = L1_gut_lfc$log2FoldChange,
                            col = list(L1_Gut_LFC = col_fun))
Ha_L1chip_L1gutLFC
```



```
# pdf(file = "./03_plots/200511_AnyDE_Genes_ELT2_ELT7_L1gutLFC.pdf", width = 5, height = 5)
# Ha_L1chip_L1gutLFC
# dev.off()
```

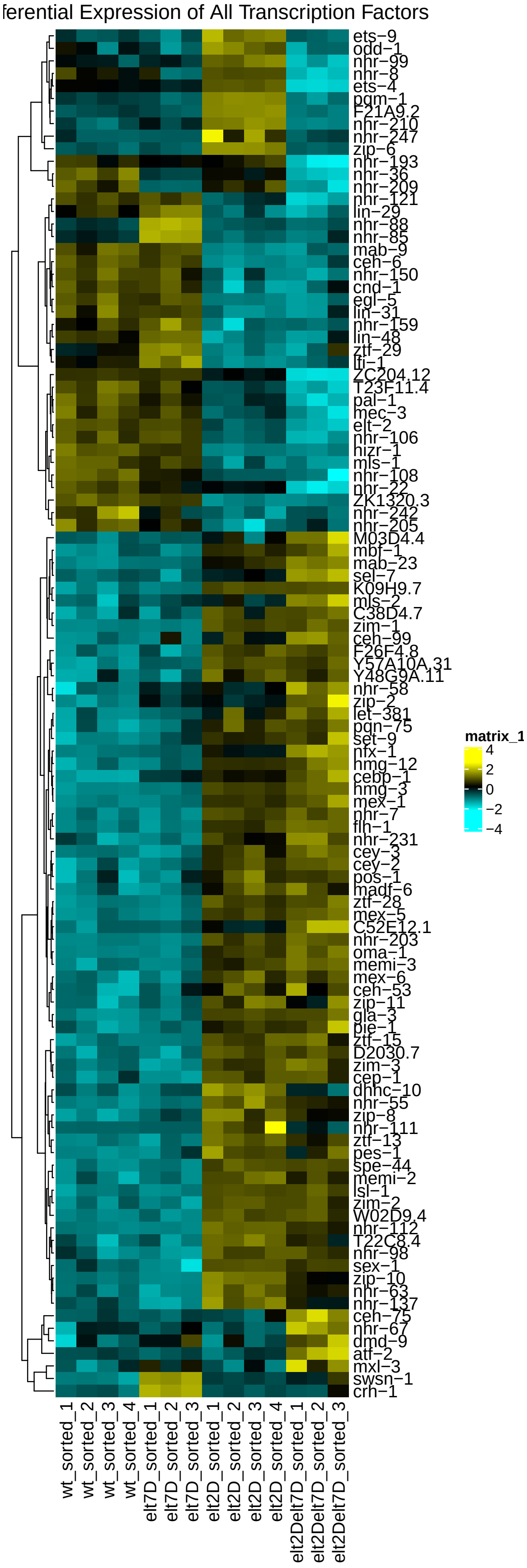
Make a TF subset heatmap

```
head(dynamic_counts_matrix_scaled)
```

```
##                 wt_sorted_1 wt_sorted_2 wt_sorted_3 wt_sorted_4 elt7D_sorted_1
## WBGene00000007   1.0068329  1.37348252   1.0589277   1.4476397     0.84613352
## WBGene00000008   2.2632093  1.13063525   1.1251278   1.0262925    -0.03607787
## WBGene00000009   0.1468716 -0.09556483  -0.3465276  -0.8378633     0.07003147
## WBGene00000013  -1.0765042  0.04628523  -1.0478603  -0.4296435    -0.61401384
## WBGene00000016  -0.1629274  0.14035593  -0.8318355  -0.2209018    -0.52814604
## WBGene00000017   0.1344074  0.43209491  -0.4453539   0.5202470    -0.19720767
##                 elt7D_sorted_2 elt7D_sorted_3 elt2D_sorted_1 elt2D_sorted_2
## WBGene00000007     0.51350637     0.07506888     -0.7898010     -0.6055647
## WBGene00000008    -0.39030667     0.02722321     -0.4521136     -1.0292850
## WBGene00000009    -0.11586861     0.42221560      0.8406016      1.2349599
## WBGene00000013    -0.58009755    -0.38693983     -0.4767996      0.3851813
## WBGene00000016    -0.50445577    -0.16186256     -0.5681545     -0.6137809
## WBGene00000017     0.05519157     0.37152702     -0.9790560     -1.0378885
##                 elt2D_sorted_3 elt2D_sorted_4 elt2Delt7D_sorted_1
## WBGene00000007    -1.09248186     -0.9350192          -0.9202246
## WBGene00000008    -0.46498937     -0.8771172          -0.9402531
## WBGene00000009     0.98161197      1.7266509          -1.7004545
## WBGene00000013     0.09286966     -0.5163112           2.5457794
## WBGene00000016    -0.75209134     -1.0136068           1.7015008
## WBGene00000017    -1.16996644     -1.7376299           1.4066491
##                 elt2Delt7D_sorted_2 elt2Delt7D_sorted_3
## WBGene00000007           -0.8564679          -1.1220323
## WBGene00000008           -0.5550156          -0.8273297
## WBGene00000009           -0.8668929          -1.4597714
## WBGene00000013            1.4999051           0.5581492
## WBGene00000016            2.1353949           1.3805110
## WBGene00000017            1.6701858           0.9767996
```

```
wTF3.0 <-
  read.csv("./01_input/TF3-0_namesonly.txt",
           sep = "\t",
           header = TRUE) %>% select(WBGeneID)

dynamic_counts_matrix_scaled_TFs <-
  matrix_select(dynamic_counts_matrix_scaled, wTF3.0$WBGeneID)

dynamic_counts_matrix_scaled_TFs_names <-
  id2name(dynamic_counts_matrix_scaled_TFs)

tf_heatmap <- Heatmap(
  dynamic_counts_matrix_scaled_TFs_names,
  col = colorRampPalette(c("cyan", "black", "yellow"))(1000),
  cluster_columns = FALSE,
  clustering_distance_rows = "spearman",
```

```r
  clustering_method_rows = "complete",
  show_row_names = TRUE,
  show_column_names = TRUE,
  column_title = "Differential Expression of All Transcription Factors"
)
tf_heatmap
```

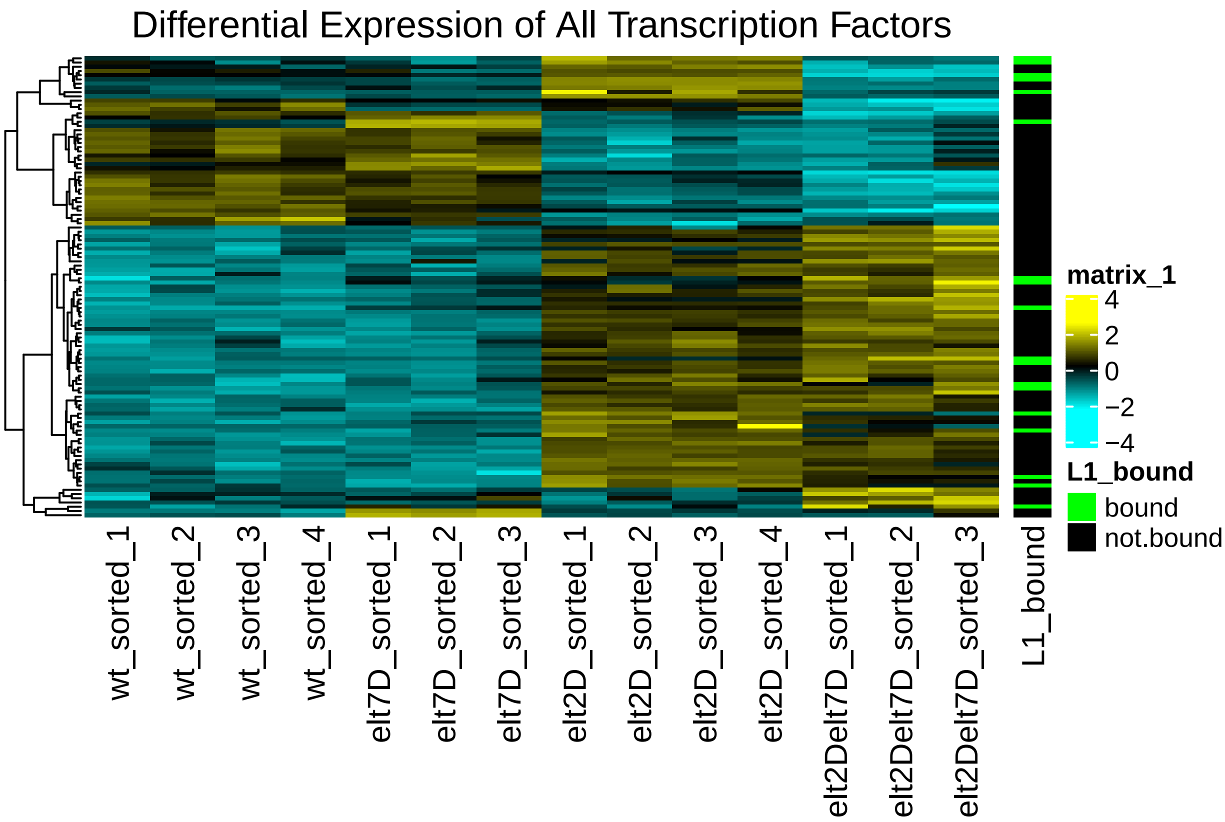

ferential Expression of All Transcription Factors
ets−9
odd−1
nhr−99
nhr−8
ets−4
pqm−1
F21A9.2
nhr−210
nhr−247
zip−6
nhr−193
nhr−36
nhr−209
nhr−121
lin−29
nhr−88
nhr−85
mab−9
ceh−6
nhr−150
cnd−1
egl−5
lin−31
nhr−159
lin−48
ztf−29
lfi−1
ZC204.12
T23F11.4
pal−1
mec−3
elt−2
nhr−106
hizr−1
mls−1
nhr−108
nhr−22
ZK1320.3
nhr−242
nhr−205
M03D4.4
mbf−1
mab−23
sel−7
K09H9.7
mls−2
C38D4.7
zim−1
ceh−99
F26F4.8
Y57A10A.31
Y48G9A.11
nhr−58
zip−2
let−381
pqn−75
set−9
nfx−1
hmg−12
cebp−1
hmg−3
mex−1
nhr−7
flh−1
nhr−231
cey−3
cey−2
pos−1
madf−6
ztf−28
mex−5
C52E12.1
nhr−203
oma−1
memi−3
mex−6
ceh−53
zip−11
gla−3
pie−1
ztf−15
D2030.7
zim−3
cep−1
dhhc−10
nhr−55
zip−8
nhr−111
ztf−13
pes−1
spe−44
memi−2
lsl−1
zim−2
W02D9.4
nhr−112
T22C8.4
nhr−98
sex−1
zip−10
nhr−63
nhr−137
ceh−75
nhr−67
dmd−9
atf−2
mxl−3
swsn−1
crh−1
wt_sorted_1
wt_sorted_2
wt_sorted_3
wt_sorted_4
elt7D_sorted_1
elt7D_sorted_2
elt7D_sorted_3
elt2D_sorted_1
elt2D_sorted_2
elt2D_sorted_3
elt2D_sorted_4
elt2Delt7D_sorted_1
elt2Delt7D_sorted_2
elt2Delt7D_sorted_3
matrix_1
4
2
0
−2
−4

Add row annotation to indicate ELT-2 binding in L1 stage

```
elt2_detected_in_L1 %>% filter(WBGeneID %in% rownames(dynamic_counts_matrix_scaled_TFs))
```

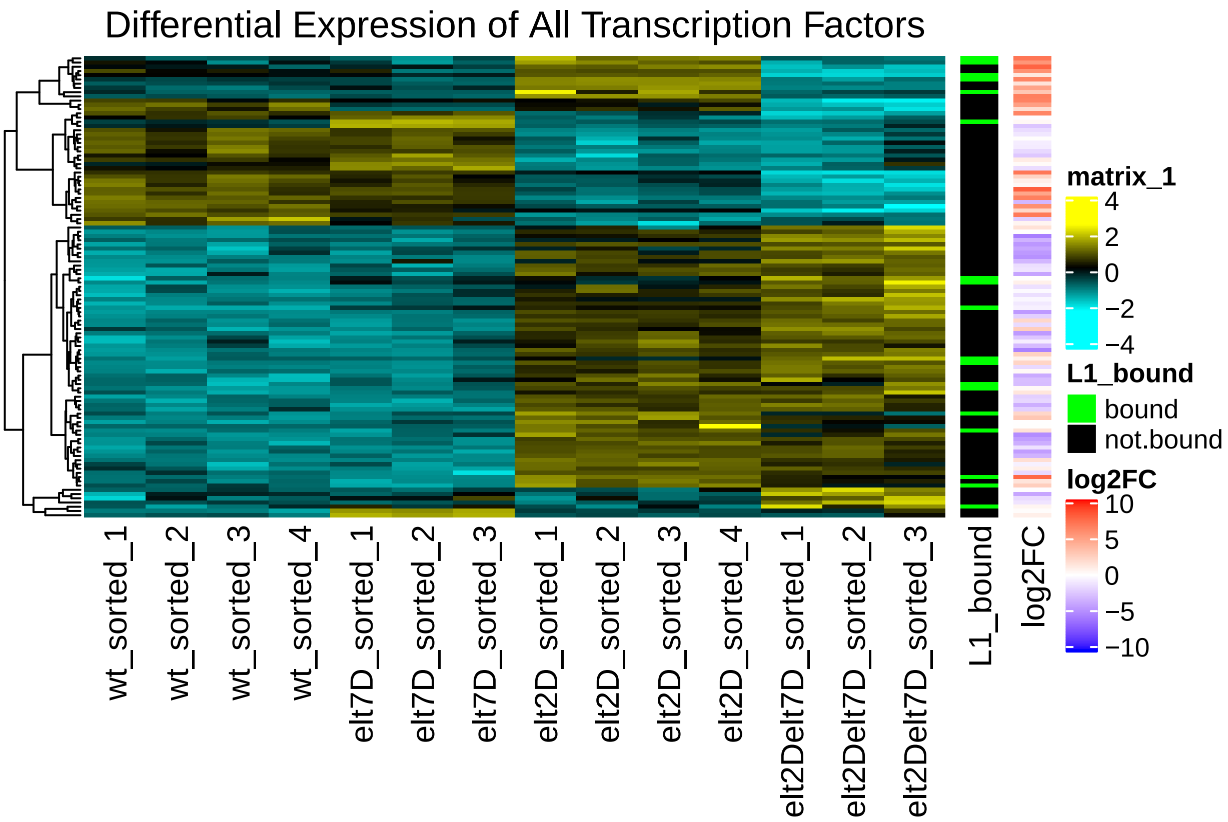
```
## # A tibble: 18 x 1
##    WBGeneID
##    <chr>
##  1 WBGene00011376
##  2 WBGene00003678
##  3 WBGene00016888
##  4 WBGene00004096
##  5 WBGene00019327
##  6 WBGene00003845
##  7 WBGene00021082
##  8 WBGene00019743
##  9 WBGene00003648
## 10 WBGene00012101
## 11 WBGene00014193
## 12 WBGene00016997
## 13 WBGene00018704
## 14 WBGene00016865
## 15 WBGene00019344
## 16 WBGene00017687
## 17 WBGene00003727
## 18 WBGene00003511
```

```
tf_bound_anno <-
  data.frame(
    elt2_detected_in_L1 = ifelse(
      test = rownames(dynamic_counts_matrix_scaled_TFs) %in% elt2_detected_in_L1$WBGeneID,
      yes = "bound",
      no = "not.bound"
    )
  )

tf_heatmap_L1bound <-
  tf_heatmap + rowAnnotation(L1_bound = tf_bound_anno$elt2_detected_in_L1,
                             col = list(L1_bound = c(
                               "bound" = "green", "not.bound" = "black"
                             )))
tf_heatmap_L1bound
```



Add row annotation of intestine expression

```
tf_lfc <-
  L1_gut_lfc %>% filter(WBGeneID %in% rownames(dynamic_counts_matrix_scaled_TFs)) %>% select(log2FoldCha

tf_heatmap_L1bound +
  rowAnnotation(log2FC = tf_lfc$log2FoldChange,
                col = list(log2FC = colorRamp2(c(-10, 0, 10), c(
                  "blue", "white", "red"
                ))))
```

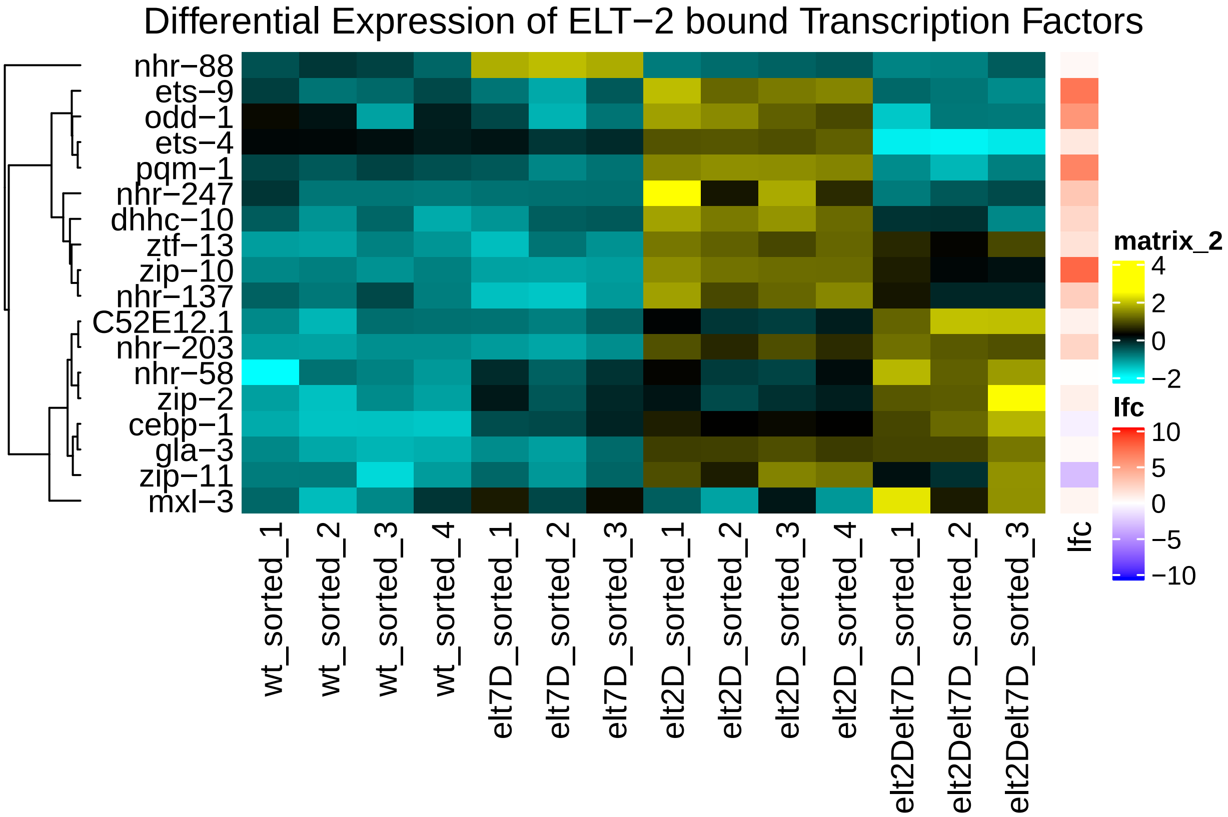


Zoom in on only bound TFs

```
dynamic_counts_matrix_scaled_TFs_bound <-
  matrix_select(dynamic_counts_matrix_scaled_TFs,
                elt2_detected_in_L1$WBGeneID)

dynamic_counts_matrix_scaled_TFs_bound_names <-
  id2name(dynamic_counts_matrix_scaled_TFs_bound)

bound_tf_lfc <-
  L1_gut_lfc %>% filter(WBGeneID %in% rownames(dynamic_counts_matrix_scaled_TFs_bound)) %>% select(log2F

bound_tf_lfc
```

```
##    log2FoldChange
## 1      0.53748793
## 2      0.34541162
## 3      2.53692129
## 4      5.37850676
## 5      6.24020573
## 6      0.30796206
## 7      7.50528790
## 8      2.93252241
## 9      6.87647779
## 10     0.71362284
## 11    -0.62100287
## 12     1.22273953
## 13     1.52846896
## 14     0.80503709
## 15     2.09004400
## 16     2.20399953
## 17    -2.83834024
## 18     0.07431616
```

```
Heatmap(
  dynamic_counts_matrix_scaled_TFs_bound_names,
  col = colorRampPalette(c("cyan", "black", "yellow"))(1000),
  cluster_columns = FALSE,
  clustering_distance_rows = "spearman",
  clustering_method_rows = "complete",
  show_row_names = TRUE,
  row_names_side = "left",
  show_column_names = TRUE,
  column_title = "Differential Expression of ELT-2 bound Transcription Factors"
) +
  rowAnnotation(lfc = bound_tf_lfc$log2FoldChange,
                col = list(lfc = colorRamp2(c(-10, 0, 10), c(
                  "blue", "white", "red"
                ))))
```



This plot suggests that transcription factors bound by ELT-2 are typically upregulated in the absence of ELT-2.

Additionally, TFs that are expressed in the L1 intestine are upregulated in absence of ELT-2 alone, but downregulated in the absence of both ELT-2 and ELT-7.

Futhermore, TFs that are not expressed in the L1 intestine are upregulated only in the absence of both ELT-2 and ELT-7.

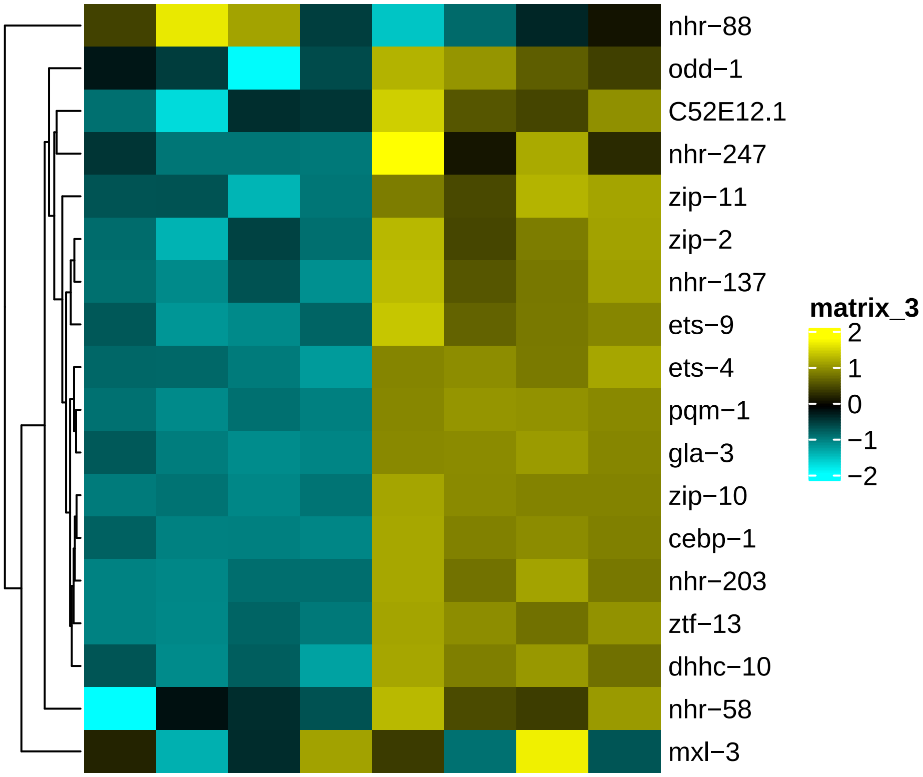
TFs to follow up: pqm-1, zip-10, odd-1 (repressed by elt-2 alone, normally gut expressed). nhr-58 (vulva), zip-2 (neuron), cebp-1 (neuron), gla-3 (germline), zip-11

old code below

Transcription factor subset plots

Load in wTF3.0 list

```r
wTF3.0 <-
  read.csv("./01_input/TF3-0_namesonly.txt",
           sep = "\t",
           header = TRUE) %>% select(WBGeneID)

dynamic_counts_matrix_TFs <-
  matrix_select(dynamic_counts_matrix, wTF3.0$WBGeneID)
dynamic_counts_matrix_TFs_bound <-
  matrix_select(dynamic_counts_matrix_scaled_TFs,
                elt2_detected_in_L1$WBGeneID)

dynamic_counts_matrix_TFs_bound_sub <-
  subset(
    dynamic_counts_matrix_TFs_bound,
    select = c(
      "wt_sorted_1",
      "wt_sorted_2",
      "wt_sorted_3",
      "wt_sorted_4",
      "elt2D_sorted_1",
      "elt2D_sorted_2",
      "elt2D_sorted_3",
      "elt2D_sorted_4"
    )
  )

dynamic_counts_matrix_TFs_bound_sub_scale <-
  t(apply(unlist(dynamic_counts_matrix_TFs_bound_sub), 1, scale))

# varselect <- rowVars(colsub) >= 0.5
# colsub <- colsub[varselect, ]

dynamic_counts_matrix_TFs_bound_sub_scale_names <-
  id2name(dynamic_counts_matrix_TFs_bound_sub_scale)

bound_TF_heatmap <-
  Heatmap(
    dynamic_counts_matrix_TFs_bound_sub_scale_names,
    col = colorRampPalette(c("cyan", "black", "yellow"))(1000),
    cluster_columns = FALSE,
    #clustering_distance_rows = "spearman",
    row_dend_reorder = TRUE,
    clustering_method_rows = "complete",
    show_row_names = TRUE,
    show_column_names = TRUE,
    row_names_gp = gpar(fontsize = 10),
    column_names_gp = gpar(cex = 0.4),
    heatmap_legend_param = list(color_bar = "continuous"),
    #split = 2,
    width = unit(3, "in"),
    height = unit(4, "in")
  )
bound_TF_heatmap
```

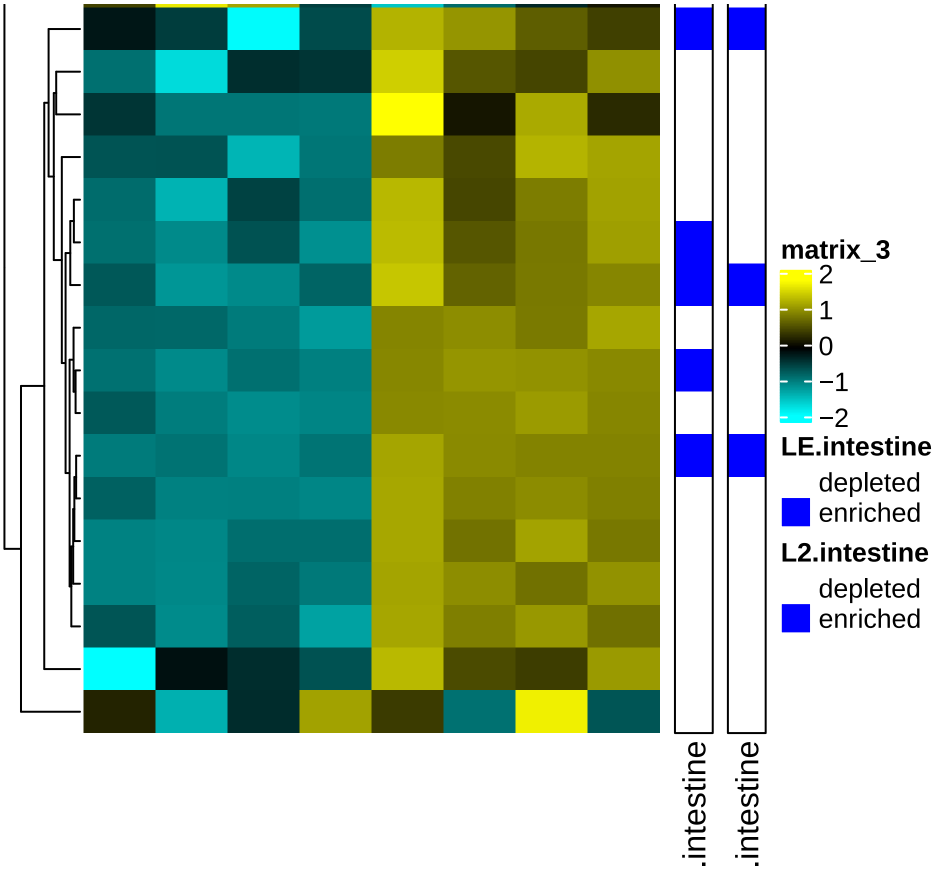


```
pdf("./03_plots/200428_TF_only_L1elt2bound",
    height = 5,
    width = 5)
bound_TF_heatmap
dev.off()
```

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

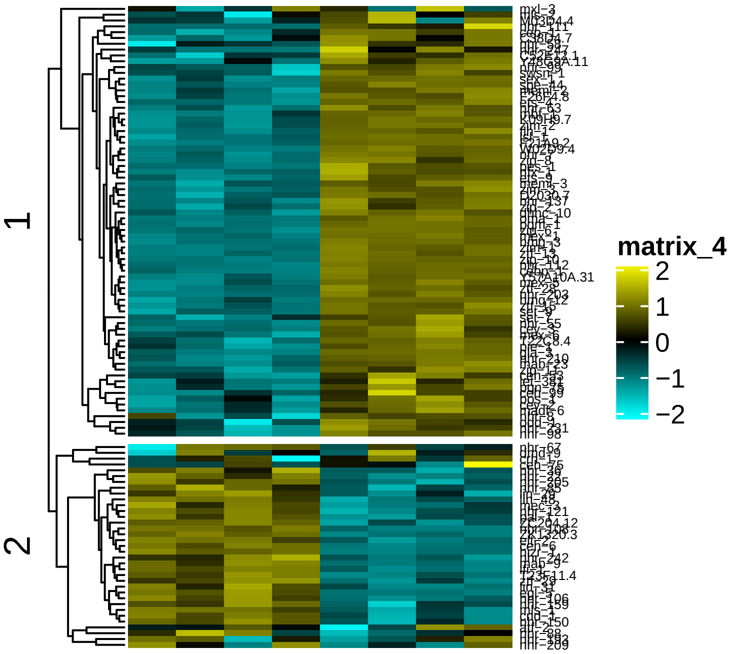
Add intestine expressed annotation.

```
bound_TF_rna_anno <- data.frame(
  spencerLE = ifelse(
    test = rownames(dynamic_counts_matrix_TFs_bound_sub_scale) %in% spencer_LE_subset$spencer_LE_ID,
    yes = "enriched",
    no = "depleted"
  ),
  spencerL2 = ifelse(
    test = rownames(dynamic_counts_matrix_TFs_bound_sub_scale) %in% spencer_L2_subset$spencer_L2_ID,
    yes = "enriched",
    no = "depleted"
  )
)

bound_TF_heatmap +
    rowAnnotation(LE.intestine = bound_TF_rna_anno$spencerLE, col = list(LE.intestine = c("enriched" =
  rowAnnotation(L2.intestine = bound_TF_rna_anno$spencerL2, col = list(L2.intestine = c("enriched" = "bl
```



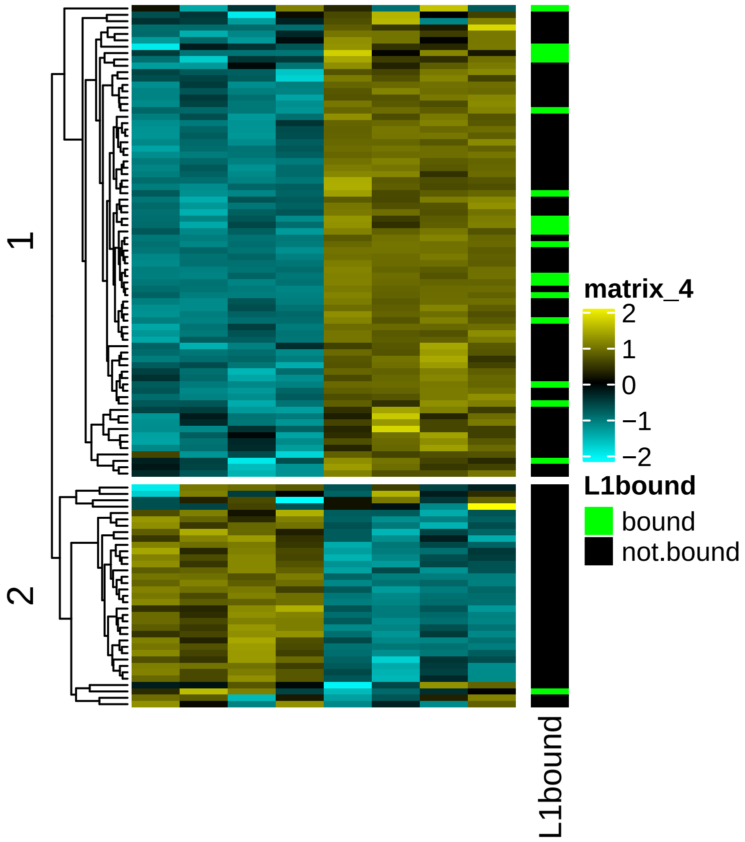
```
dynamic_counts_matrix_TFs <- matrix_select(dynamic_counts_matrix, wTF3.0$WBGeneID)
dynamic_counts_matrix_TFs_sub <- subset(dynamic_counts_matrix_TFs, select = c("wt_sorted_1", "wt_sorted_
dynamic_counts_matrix_TFs_sub_scale <- t(apply(unlist(dynamic_counts_matrix_TFs_sub), 1, scale))
dynamic_counts_matrix_TFs_sub_scale_names <- id2name(dynamic_counts_matrix_TFs_sub_scale)
Ha_TF <- Heatmap(dynamic_counts_matrix_TFs_sub_scale_names,
          col = colorRampPalette(c("cyan","black","yellow"))(1000),
          cluster_columns = FALSE,
          #clustering_distance_rows = "spearman",
          #row_dend_reorder = TRUE,
          clustering_method_rows = "complete",
          show_row_names = TRUE,
          show_column_names = TRUE,
          row_names_gp = gpar(cex = 0.4),
          column_names_gp = gpar(cex = 0.4),
          heatmap_legend_param = list(color_bar = "continuous"),
          #split = 2,
           width = unit(2, "in"),
          # height = unit(3, "in"),
          split = 2)
```

```
Ha_TF
```



```
# svg("./killme.svg", width = 8, height = 8)
# Ha_TF
# dev.off()
```

```
TF_anno <- data.frame(elt2_detected_in_L1 = ifelse(test = rownames(dynamic_counts_matrix_TFs_sub_scale)

Ha_TF + rowAnnotation(L1bound = TF_anno$elt2_detected_in_L1,
                      col = list(L1bound = c("bound" = "green", "not.bound" = "black")))
```



```
# pdf("./tf_heatmap.pdf", height = 8, width = 8)
# Ha_TF + rowAnnotation(L1bound = TF_anno$elt2_detected_in_L1,
#                    col = list(L1bound = c("bound" = "green", "not.bound" = "black")))
# dev.off()
```

Add intestine expression row annotation.

```
TF_rna_anno <- data.frame(
  spencerLE = ifelse(test = rownames(dynamic_counts_matrix_TFs_sub_scale) %in% spencer_LE_subset$spencer
  spencerL2 = ifelse(test = rownames(dynamic_counts_matrix_TFs_sub_scale) %in% spencer_L2_subset$spencer
    )

Ha_TF_spencer <- Ha_TF + rowAnnotation(L1bound = TF_anno$elt2_detected_in_L1,
                   col = list(L1bound = c("bound" = "green", "not.bound" = "black"))) +
    rowAnnotation(LE.intestine = TF_rna_anno$spencerLE, col = list(LE.intestine = c("enriched" = "blue"
  rowAnnotation(L2.intestine = TF_rna_anno$spencerL2, col = list(L2.intestine = c("enriched" = "blue",

# pdf("./03_plots/200428_DE_TFs_elt2chip_intestineExpression", height = 8, width = 8)
# Ha_TF_spencer
# dev.off()
```

Results and interpretation

Session Info

```
sessionInfo()
```

```
## R version 3.6.3 (2020-02-29)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS High Sierra 10.13.6
##
## 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] grid      stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] circlize_0.4.9       binom_1.1-1          dendextend_1.13.4
##  [4] RVAideMemoire_0.9-75 pheatmap_1.0.12      matrixStats_0.56.0
##  [7] ComplexHeatmap_2.2.0 readxl_1.3.1         forcats_0.5.0
## [10] stringr_1.4.0        dplyr_0.8.5          purrr_0.3.4
## [13] readr_1.3.1          tidyr_1.0.3          tibble_3.0.1
## [16] ggplot2_3.3.0        tidyverse_1.3.0      biomaRt_2.42.1
##
## loaded via a namespace (and not attached):
##  [1] nlme_3.1-147         fs_1.4.1             lubridate_1.7.8
##  [4] bit64_0.9-7          RColorBrewer_1.1-2   progress_1.2.2
##  [7] httr_1.4.1           tools_3.6.3          backports_1.1.6
## [10] utf8_1.1.4           R6_2.4.1             DBI_1.1.0
## [13] BiocGenerics_0.32.0  colorspace_1.4-1     GetoptLong_0.1.8
## [16] withr_2.2.0          gridExtra_2.3        tidyselect_1.0.0
## [19] prettyunits_1.1.1    bit_1.1-15.2         curl_4.3
## [22] compiler_3.6.3       cli_2.0.2            rvest_0.3.5
## [25] Biobase_2.46.0       xml2_1.3.2           labeling_0.3
## [28] scales_1.1.0         askpass_1.1          rappdirs_0.3.1
## [31] digest_0.6.25        rmarkdown_2.1        pkgconfig_2.0.3
## [34] htmltools_0.4.0      dbplyr_1.4.3         rlang_0.4.6
## [37] GlobalOptions_0.1.1  rstudioapi_0.11      RSQLite_2.2.0
## [40] farver_2.0.3         shape_1.4.4          generics_0.0.2
## [43] jsonlite_1.6.1       magrittr_1.5         Rcpp_1.0.4.6
## [46] munsell_0.5.0        S4Vectors_0.24.4     fansi_0.4.1
## [49] viridis_0.5.1        lifecycle_0.2.0      stringi_1.4.6
## [52] yaml_2.2.1           BiocFileCache_1.10.2 blob_1.2.1
## [55] parallel_3.6.3       crayon_1.3.4         lattice_0.20-41
## [58] haven_2.2.0          hms_0.5.3            knitr_1.28
## [61] pillar_1.4.4         rjson_0.2.20         stats4_3.6.3
## [64] reprex_0.3.0         XML_3.99-0.3         glue_1.4.0
## [67] evaluate_0.14        modelr_0.1.7         png_0.1-7
## [70] vctrs_0.2.4          cellranger_1.1.0     gtable_0.3.0
## [73] openssl_1.4.1        clue_0.3-57          assertthat_0.2.1
## [76] xfun_0.13            broom_0.5.6          viridisLite_0.3.0
## [79] AnnotationDbi_1.48.0 memoise_1.1.0        IRanges_2.20.2
## [82] cluster_2.1.0        ellipsis_0.3.0
```