

RWC23_ELT2_Regulated_Genes

RTPW

11/10/2020

Decide if plots should be saved to files:

Change `plot` to `TRUE` if you want to write plots to a file change `plot` to `FALSE` if you do not want to write plots to a file

```
plot <- TRUE
plotdir <- "./03_plots/"
```

Install Packages

```
# 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

```
library(biomaRt)
library(tidyverse)
```

```
## -- Attaching packages ----- tidyverse 1.3.0
## v ggplot2 3.3.0      v purrr  0.3.4
## v tibble  3.0.1      v dplyr  0.8.5
## v tidyr   1.0.3      v stringr 1.4.0
## v readr   1.3.1      v forcats 0.5.0

## -- Conflicts ----- tidyverse_conflict_
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x dplyr::select() masks biomaRt::select()
```

```

library(readxl)
library(ComplexHeatmap)

## Loading required package: grid
## =====
## ComplexHeatmap version 2.2.0
## Bioconductor page: http://bioconductor.org/packages/ComplexHeatmap/
## Github page: https://github.com/jokergoo/ComplexHeatmap
## Documentation: http://jokergoo.github.io/ComplexHeatmap-reference
##
## If you use it in published research, please cite:
## Gu, Z. Complex heatmaps reveal patterns and correlations in multidimensional
##   genomic data. Bioinformatics 2016.
## =====
library(matrixStats)

##
## Attaching package: 'matrixStats'
##
## The following object is masked from 'package:dplyr':
##
##   count
library(pheatmap)
library(RVAideMemoire)

## *** Package RVAideMemoire v 0.9-75 ***
library(dendextend)

##
## -----
## Welcome to dendextend version 1.13.4
## Type citation('dendextend') for how to cite the package.
##
## Type browseVignettes(package = 'dendextend') for the package vignette.
## The github page is: https://github.com/talgalili/dendextend/
##
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendextend/issues
## Or contact: <tal.galili@gmail.com>
##
## To suppress this message use: suppressPackageStartupMessages(library(dendextend))
## -----
##
## Attaching package: 'dendextend'
##
## The following object is masked from 'package:stats':
##
##   cutree
library(binom)
library(circlize)

## =====
## circlize version 0.4.9
## CRAN page: https://cran.r-project.org/package=circlize

```

```
## Github page: https://github.com/jokergoo/circlize
## Documentation: https://jokergoo.github.io/circlize\_book/book/
##
## If you use it in published research, please cite:
## Gu, Z. circlize implements and enhances circular visualization
## in R. Bioinformatics 2014.
##
## This message can be suppressed by:
## suppressPackageStartupMessages(library(circlize))
## =====
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
##
## The following objects are masked from 'package:dplyr':
##
## intersect, setdiff, union
##
## The following objects are masked from 'package:base':
##
## date, intersect, setdiff, union
```

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 document 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?
 - For all genes
 - For only genes bound by ELT-2
- 3) Which expression pattern clusters associate with intestine expression? (MA plot for each expression set)
 - For all genes
 - 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
 - for all TFs
 - For only TFs bound by ELT-2
- 3) which transcription factor clusters associate with intestine expression?
 - for all
 - for only ELT-2 bound

Description of Data

I will integrate a RNA-seq experiment, a microarray experiment 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 dataset is from a 2011 paper using FACS sorting of Late Embryo (LE) and Larval Stage 2 (L2) intestine cells, measured with microarray. See Spencer et. al, (2011).

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>
- 3) Spencer, W. C., Zeller, G., Watson, J. D., Henz, S. R., Watkins, K. L., McWhirter, R. D., Petersen, S., Sreedharan, V. T., Widmer, C., Jo, J., Reinke, V., Petrella, L., Strome, S., Von Stetina, S. E., Katz, M., Shaham, S., Räscher, G., & Miller, D. M. (2011). A spatial and temporal map of *C. elegans* gene expression. *Genome Research*, 21(2), 325–341. <https://doi.org/10.1101/gr.114595.110>
- 4) Boeck, M. E., Huynh, C., Gevirtzman, L., Thompson, O. A., Wang, G., Kasper, D. M., Reinke, V., Hillier, L. W., & Waterston, R. H. (2016). The time-resolved transcriptome of *C. elegans*. *Genome Research*, 26(10), 1441–1450. <https://doi.org/10.1101/gr.202663.115>

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  
## WBGene000000001      8.957161      8.858238      8.841623      8.923111      8.505028  
## WBGene000000002      7.489159      7.382905      7.518631      7.492399      7.378168  
## WBGene000000003      9.061810      8.748589      9.295497      9.286834      9.480361  
## WBGene000000004     10.916559     10.786200     11.010430     10.826657     10.836827  
## WBGene000000005      2.990777      2.864044      3.116144      2.715502      2.584081  
## WBGene000000007      5.799066      6.026780      5.831420      6.072836      5.699261  
##           elt7D_sorted_2 elt7D_sorted_3 elt2D_sorted_1 elt2D_sorted_2  
## WBGene000000001      8.568569      8.517438      9.172904      9.249496  
## WBGene000000002      7.582425      7.512668      7.503760      7.289884  
## WBGene000000003      9.451384      9.008938      8.669299      8.593847  
## WBGene000000004     10.806534     10.819497     10.303062     10.296768  
## WBGene000000005      2.881642      2.827526      2.953325      2.835451  
## WBGene000000007      5.492677      5.220378      4.683237      4.797660  
##           elt2D_sorted_3 elt2D_sorted_4 elt2Delt7D_sorted_1  
## WBGene000000001      9.211660      9.346959      9.379698  
## WBGene000000002      7.386127      7.262063      7.904008  
## WBGene000000003      8.753835      8.781267      8.791018  
## WBGene000000004     10.356820     10.366512     10.332489  
## WBGene000000005      2.886842      2.979650      2.499412  
## WBGene000000007      4.495252      4.593047      4.602235  
##           elt2Delt7D_sorted_2 elt2Delt7D_sorted_3  
## WBGene000000001      9.217403      9.101997  
## WBGene000000002      7.870852      7.762023  
## WBGene000000003      8.795191      8.936724  
## WBGene000000004     10.223675     10.597407  
## WBGene000000005      2.763405      2.428255  
## WBGene000000007      4.641832      4.476899  
  
list of all dynamically expressed genes  
  
dynamic_regulated_genes <-  
  read.table(file = "./01_input/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
```

Load differential expression clusters from Dineen and Nishimura et al (2018).

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

dineen_nishimura_sets <-
  dineen_nishimura_clusters %>% select(WBGeneID, set)
dineen_nishimura_sets_ascend <-
  arrange(dineen_nishimura_sets, WBGeneID)
dineen_nishimura_sets_ascend$set <-
  toupper(dineen_nishimura_sets_ascend$set)
dineen_nishimura_sets_ascend %>% 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

Make sure the column names are correct here and that the factor levels match the cluster description names

```
# temp_peaks <- mcols(readRDS("/Users/rtpw/Dropbox/01_GITHUBREPO/RWC23_elt2_regulated_genes/01_ChIPseq_
# write.csv(temp_peaks, file = "./01_input/201019_annotatedPeaks.csv")
library(GenomicRanges)
```

```
## 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:lubridate':
##
##   intersect, setdiff, union
## The following objects are masked from 'package:dplyr':
##
##   combine, intersect, setdiff, union
```

```

## 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
## Loading required package: S4Vectors
##
## Attaching package: 'S4Vectors'
## The following objects are masked from 'package:lubridate':
##
##   second, second<-
## The following objects are masked from 'package:dplyr':
##
##   first, rename
## The following object is masked from 'package:tidyr':
##
##   expand
## The following object is masked from 'package:base':
##
##   expand.grid
## Loading required package: IRanges
##
## Attaching package: 'IRanges'
## The following object is masked from 'package:lubridate':
##
##   %within%
## The following objects are masked from 'package:dplyr':
##
##   collapse, desc, slice
## The following object is masked from 'package:purrr':
##
##   reduce
## Loading required package: GenomeInfoDb
elt2_peaks <- as.data.frame(mcols(readRDS("/Users/rtpw/Dropbox/01_GITHUBREPO/RWC23_elt2_regulated_genes.
elt2_peaks <- elt2_peaks %>% rename(k4labels = 'cluster.description', feature = "WBGeneID")

elt2_cluster_names <- c("Embryo_Specific",
                        "Larval",
                        "Increasing",
                        "L3_High",

```

```

                                "Not_Changing")

elt2_peaks$cluster.description <-
  factor(
    elt2_peaks$cluster.description,
    levels = c(
      "LE-specific",
      "Post-embryonic",
      "Increasing",
      "L3-high",
      "Not-changing or not IDR-passing"
    ),
    labels = elt2_cluster_names
  )

elt2_peaks %>% head

```

```

##           LE_1    LE_2    L1_1    L1_2    L3_1    L3_2 LE_IDR
## ELT2peak00002 1.683526 1.347923 2.673772 2.861924 4.944366 5.051280      0
## ELT2peak00003 1.879706 1.724893 2.865070 3.237039 4.966392 5.921998      0
## ELT2peak00004 0.540568 0.554968 1.584962 1.543142 2.828888 3.314315      0
## ELT2peak00005 1.273018 0.778786 2.362570 2.140178 3.876194 4.516944      0
## ELT2peak00006 2.000000 1.972414 2.596804 2.417018 4.078502 4.157214      1
## ELT2peak00007 1.955606 2.023084 3.429138 2.929611 4.141166 4.544049      1
##           L1_IDR L3_IDR k4cluster k6cluster k4weights k6weights
## ELT2peak00002      1      1          4          6      0.990      1.000
## ELT2peak00003      1      1          4          6      0.998      1.000
## ELT2peak00004      0      1          4          6      0.942      1.000
## ELT2peak00005      0      1          4          6      0.965      1.000
## ELT2peak00006      1      1          4          6      0.904      1.000
## ELT2peak00007      1      1          4          5      0.666      1.000
##           LE_nonNormed L1_nonNormed L3_nonNormed LE_std L1_std L3_std
## ELT2peak00002      1.516      2.768      4.998 -0.895 -0.185 1.080
## ELT2peak00003      1.802      3.051      5.444 -0.881 -0.206 1.087
## ELT2peak00004      0.548      1.564      3.072 -0.929 -0.129 1.058
## ELT2peak00005      1.026      2.251      4.197 -0.916 -0.150 1.067
## ELT2peak00006      1.986      2.507      4.118 -0.796 -0.327 1.123
## ELT2peak00007      1.989      3.179      4.343 -1.004 0.008 0.996
##           name cluster.description variance      peak
## ELT2peak00002 ELT2peak00002      Increasing 3.110935 ELT2peak00002
## ELT2peak00003 ELT2peak00003      Increasing 3.424986 ELT2peak00003
## ELT2peak00004 ELT2peak00004      Increasing 1.612546 ELT2peak00004
## ELT2peak00005 ELT2peak00005      Increasing 2.556449 ELT2peak00005
## ELT2peak00006 ELT2peak00006      Increasing 1.235036 ELT2peak00006
## ELT2peak00007 ELT2peak00007      Increasing 1.384521 ELT2peak00007
##           WBGeneID start_position end_position feature_strand
## ELT2peak00002 WBGene00022277      4116      10230      -
## ELT2peak00003 WBGene00022276      11495      16837      +
## ELT2peak00004 WBGene00022276      11495      16837      +
## ELT2peak00005 WBGene00022276      11495      16837      +
## ELT2peak00006 WBGene00022278      17487      26781      -
## ELT2peak00007 WBGene00022278      17487      26781      -
##           insideFeature distancetoFeature shortestDistance
## ELT2peak00002      overlapEnd      3097      217

```



```
## ELT2peak00003 overlapStart -2760 267
## ELT2peak00004 inside 30 1965
## ELT2peak00005 inside 1250 1180
## ELT2peak00006 overlapEnd 5030 42
## ELT2peak00007 inside 53 3734
## fromOverlappingOrNearest
## ELT2peak00002 Overlapping
## ELT2peak00003 Overlapping
## ELT2peak00004 Overlapping
## ELT2peak00005 Overlapping
## ELT2peak00006 Overlapping
## ELT2peak00007 Overlapping
```

Output the number of ELT-2 peaks within each binding class.

```
table(mcols(readRDS(file = "./01_input/201109_annotatedPeaks.rds"))$k4labels)
```

```
##
##           Increasing           L3-high
##           5428           2797
##           LE-specific Not-changing or not IDR-passing
##           273           551
##           Post-embryonic
##           1963
```

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

```
elt2_detected_in_L1 <-
  elt2_peaks %>% select(WBGeneID, L1_IDR) %>% filter(L1_IDR == 1) %>% select(WBGeneID) %>% unique()

elt2_detected_in_L1 %>% head
```

```
##           WBGeneID
## 1 WBGene00022277
## 2 WBGene00022276
## 3 WBGene00022278
## 6 WBGene00021681
## 7 WBGene00002077
## 8 WBGene00004143
```

```
elt2_detected_in_L1 %>% dim
```

```
## [1] 3791 1
```

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

```
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()
```

```
##           Embryo_Specific Larval Increasing L3_High Not_Changing
## WBGene000000004           0           0           1           0           0
## WBGene000000007           0           0           2           0           0
## WBGene000000008           0           0           1           0           0
## WBGene000000009           0           0           0           0           1
## WBGene000000018           0           0           3           0           0
## WBGene000000022           0           0           1           0           0
```

Load Spencer et. al. intestine expression

This data is from a 2011 paper using FACS sorting of Late Embryo (LE) and Larval Stage 2 (L2) intestine cells, measured with microarray. See Spencer et. al, (2011).

```
spencerLEgenes <-  
  read.table(  
    "/01_input/Spencer_et_al_2010_FACS_and_pulldown_tilling_array/LE-intestine_enr_vs_ref.WS200.txt",  
    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(  
    "/01_input/Spencer_et_al_2010_FACS_and_pulldown_tilling_array/L2-intestine_enr_vs_ref.WS200.txt",  
    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.

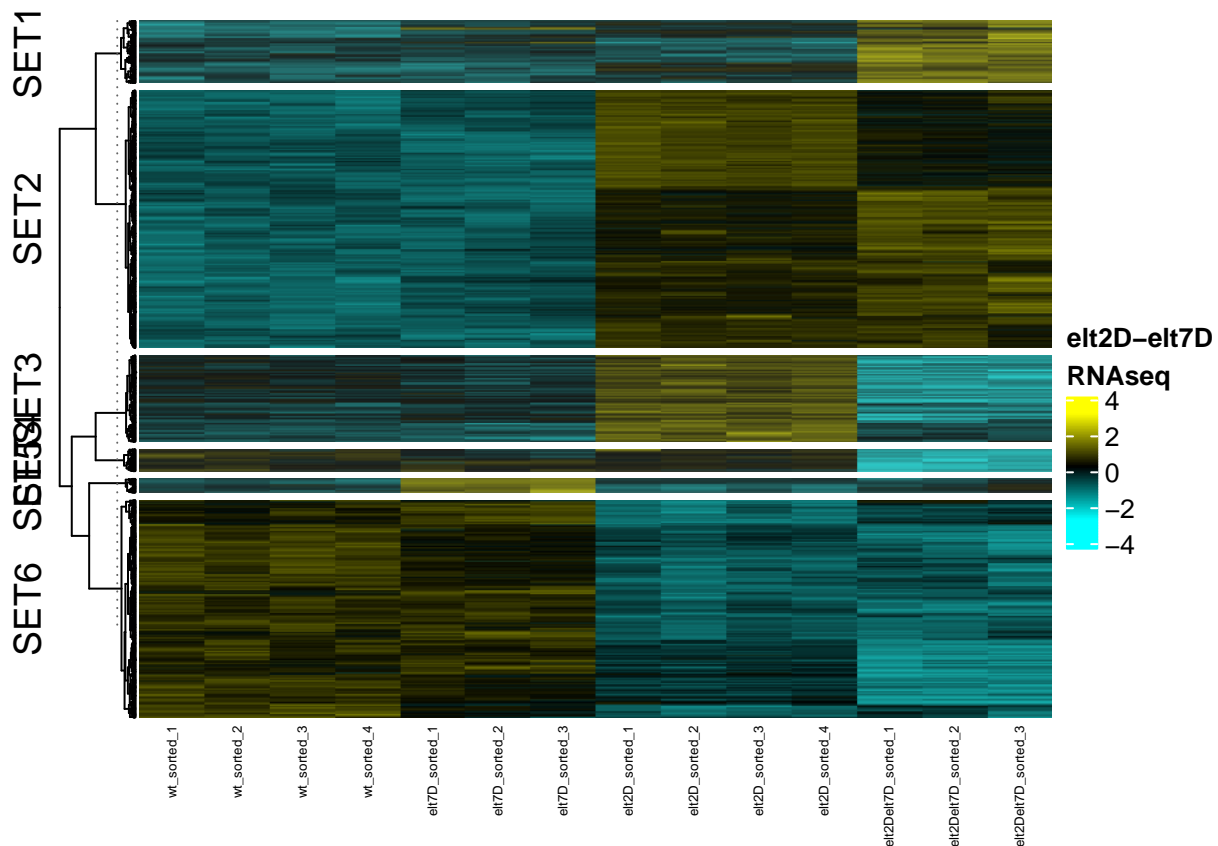
```
dynamic_counts_matrix <-  
  matrix_select(dineen_nishimura_counts_matrix,  
                dynamic_regulated_genes$WBGeneID)  
  
dynamic_counts_matrix_scaled <-  
  t(apply(unlist(dynamic_counts_matrix), 1, scale))  
  
rownames(dynamic_counts_matrix_scaled) <-  
  rownames(dynamic_counts_matrix)  
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  
## WBGene000000007    1.0068329  1.37348252   1.0589277   1.4476397    0.84613352  
## WBGene000000008    2.2632093  1.13063525   1.1251278   1.0262925   -0.03607787  
## WBGene000000009    0.1468716 -0.09556483  -0.3465276  -0.8378633    0.07003147  
## WBGene000000013   -1.0765042  0.04628523  -1.0478603  -0.4296435   -0.61401384  
## WBGene000000016   -0.1629274  0.14035593  -0.8318355  -0.2209018   -0.52814604  
## WBGene000000017    0.1344074  0.43209491  -0.4453539   0.5202470   -0.19720767  
##           elt7D_sorted_2 elt7D_sorted_3 elt2D_sorted_1 elt2D_sorted_2  
## WBGene000000007    0.51350637   0.07506888   -0.7898010   -0.6055647  
## WBGene000000008   -0.39030667   0.02722321   -0.4521136   -1.0292850  
## WBGene000000009   -0.11586861   0.42221560    0.8406016    1.2349599  
## WBGene000000013   -0.58009755  -0.38693983   -0.4767996    0.3851813  
## WBGene000000016   -0.50445577  -0.16186256   -0.5681545   -0.6137809  
## WBGene000000017    0.05519157   0.37152702   -0.9790560   -1.0378885  
##           elt2D_sorted_3 elt2D_sorted_4 elt2Delt7D_sorted_1  
## WBGene000000007   -1.09248186  -0.9350192    -0.9202246  
## WBGene000000008   -0.46498937  -0.8771172    -0.9402531  
## WBGene000000009    0.98161197   1.7266509    -1.7004545  
## WBGene000000013    0.09286966  -0.5163112    2.5457794  
## WBGene000000016   -0.75209134  -1.0136068    1.7015008  
## WBGene000000017   -1.16996644  -1.7376299    1.4066491  
##           elt2Delt7D_sorted_2 elt2Delt7D_sorted_3  
## WBGene000000007    -0.8564679    -1.1220323  
## WBGene000000008    -0.5550156    -0.8273297  
## WBGene000000009    -0.8668929    -1.4597714  
## WBGene000000013     1.4999051     0.5581492  
## WBGene000000016     2.1353949     1.3805110  
## WBGene000000017     1.6701858     0.9767996  
  
dynamic_counts_matrix_scaled_ascend <-  
  dynamic_counts_matrix_scaled[order(rownames(dynamic_counts_matrix_scaled)),]
```

Must use arrange to sort genes in descending order to ensure row order is preserved

Recreate Supplementary Figure S4a from Dineen and Nishimura et al.

Use expression clusters from Dineen and Nishimura et al to split the clusters.

```
Heatmap(
  dynamic_counts_matrix_scaled_ascend,
  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"),
  row_split = dineen_nishimura_sets_ascend$set
)
```



Add expression set and column labels.

```
RNA_column_order <-
  factor(c(
    rep("WT", 4),
    rep("elt7D", 3),
    rep("elt2D", 4),
    rep("elt7Delt2D", 3)
  ))
```

```

),
  levels = c("WT", "elt7D", "elt2D", "elt7Delt2D"))
RNA_column_order

## [1] WT      WT      WT      WT      elt7D    elt7D
## [7] elt7D    elt2D    elt2D    elt2D    elt2D    elt7Delt2D
## [13] elt7Delt2D elt7Delt2D
## Levels: WT elt7D elt2D elt7Delt2D

```

```

column_labels <-
  structure(
    c(
      "rep1",
      "rep2",
      "rep3",
      "rep4",
      "rep1",
      "rep2",
      "rep3",
      "rep1",
      "rep2",
      "rep3",
      "rep4",
      "rep1",
      "rep2",
      "rep3"
    ),
    names = colnames(dynamic_counts_matrix_scaled_ascend)
  )

```

```
column_labels
```

```

##      wt_sorted_1      wt_sorted_2      wt_sorted_3      wt_sorted_4
##      "rep1"          "rep2"          "rep3"          "rep4"
##      elt7D_sorted_1  elt7D_sorted_2  elt7D_sorted_3  elt2D_sorted_1
##      "rep1"          "rep2"          "rep3"          "rep1"
##      elt2D_sorted_2  elt2D_sorted_3  elt2D_sorted_4  elt2Delt7D_sorted_1
##      "rep2"          "rep3"          "rep4"          "rep1"
##      elt2Delt7D_sorted_2  elt2Delt7D_sorted_3
##      "rep2"          "rep3"

```

```

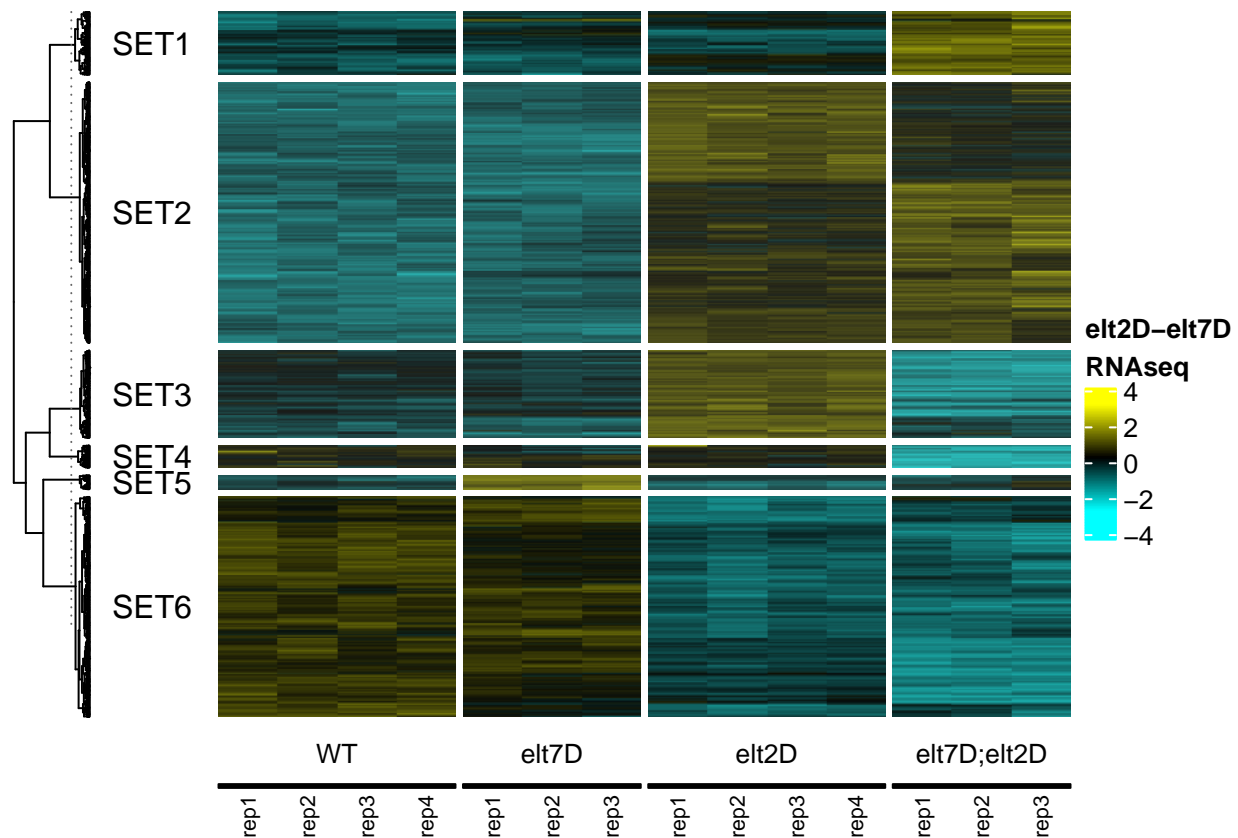
Ha <- Heatmap(
  dynamic_counts_matrix_scaled_ascend,
  name = "elt2D-elt7D\nRNAseq",
  col = colorRampPalette(c("cyan", "black", "yellow"))(1000),
  cluster_columns = FALSE,
  clustering_distance_rows = "spearman",
  clustering_method_rows = "complete",
  show_row_names = FALSE,
  show_column_names = TRUE,
  column_labels = column_labels[colnames(dynamic_counts_matrix_scaled_ascend)],
  column_names_gp = gpar(cex = 0.7),
  heatmap_legend_param = list(color_bar = "continuous"),
  row_split = dineen_nishimura_sets_ascend$set,
  row_title = NULL,

```

```

column_title = NULL,
column_split = RNA_column_order,
bottom_annotation = HeatmapAnnotation(
  foo = anno_block(
    labels = c("WT", "elt7D", "elt2D", "elt7D;elt2D"),
    labels_gp = gpar(cex = .8),
    gp = gpar(border = NA, lty = "blank")
  ),
  foo2 = anno_block(gp = gpar(fill = "black"), height = unit(0.5, "mm"))
),
left_annotation = rowAnnotation(foo = anno_block(
  labels = c("SET1", "SET2", "SET3", "SET4", "SET5", "SET6"),
  labels_rot = 0,
  gp = gpar(border = NA, lty = "blank", cex = 0.4)
))
)
Ha

```

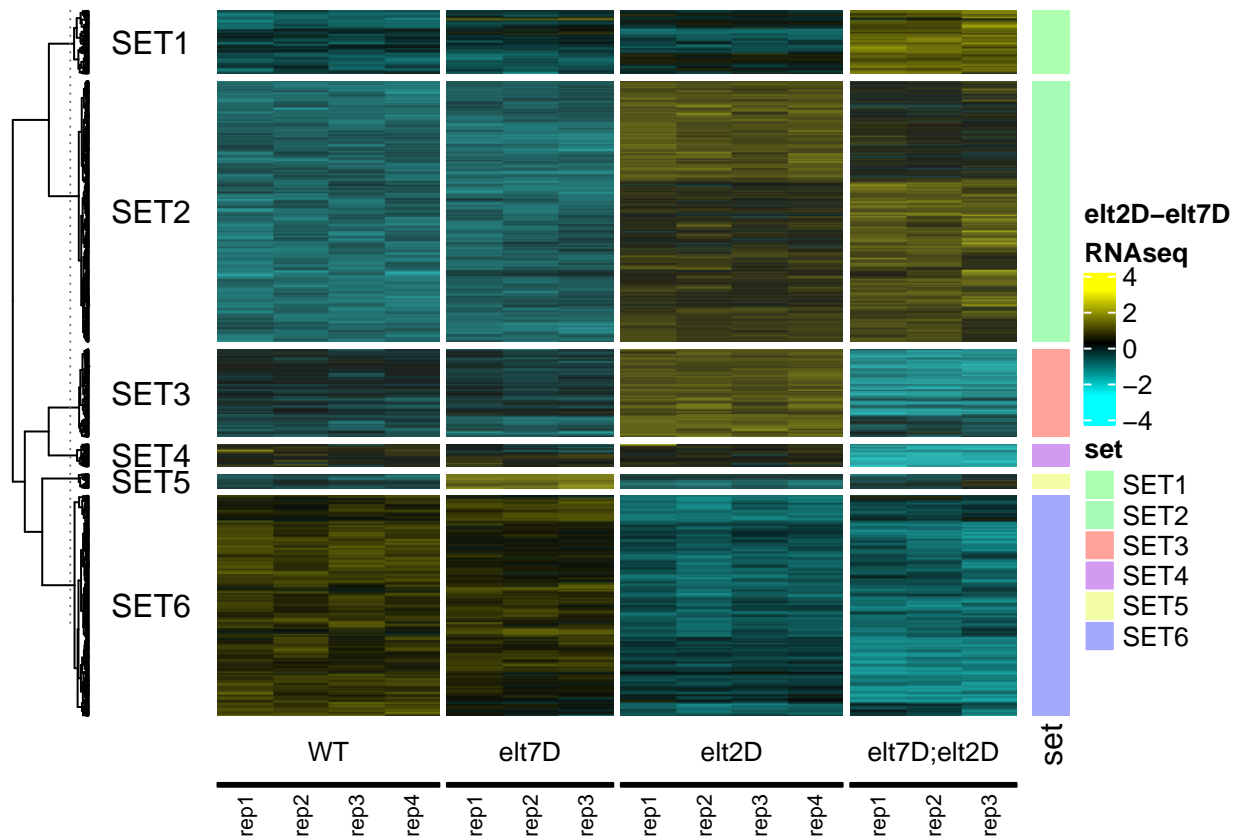


Sanity check to ensure that cluster splitting is occurring correctly. Remap the Set assignments back to the heatmap as a row annotation.

```

Ha + rowAnnotation(set = dineen_nishimura_sets_ascend$set)

```



Add L1 stage ELT-2 binding

This section will add annotation to the rows of the elt2/elt7 differential expression heatmap with ELT-2 ChIP-seq binding during the L1 stage. This will determine what differential expression sets associate with ELT-2 binding during the L1 stage. The reason L1 stage ChIP-seq eaks are being used is because the elt2/elt7 RNA-seq experiment was conducted in the L1 stage.

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

```
elt2_detected_in_L1 %>% dim
```

```
## [1] 3791 1
```

```
elt2_L1_anno <-
```

```
data.frame(
  WBGeneID = rownames(dynamic_counts_matrix_scaled_ascend),
  elt2_detected_in_L1 = ifelse(
    test = rownames(dynamic_counts_matrix_scaled_ascend) %in% elt2_detected_in_L1$WBGeneID,
    yes = "bound",
    no = "not.bound"
  ),
  stringsAsFactors = FALSE
)
```

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

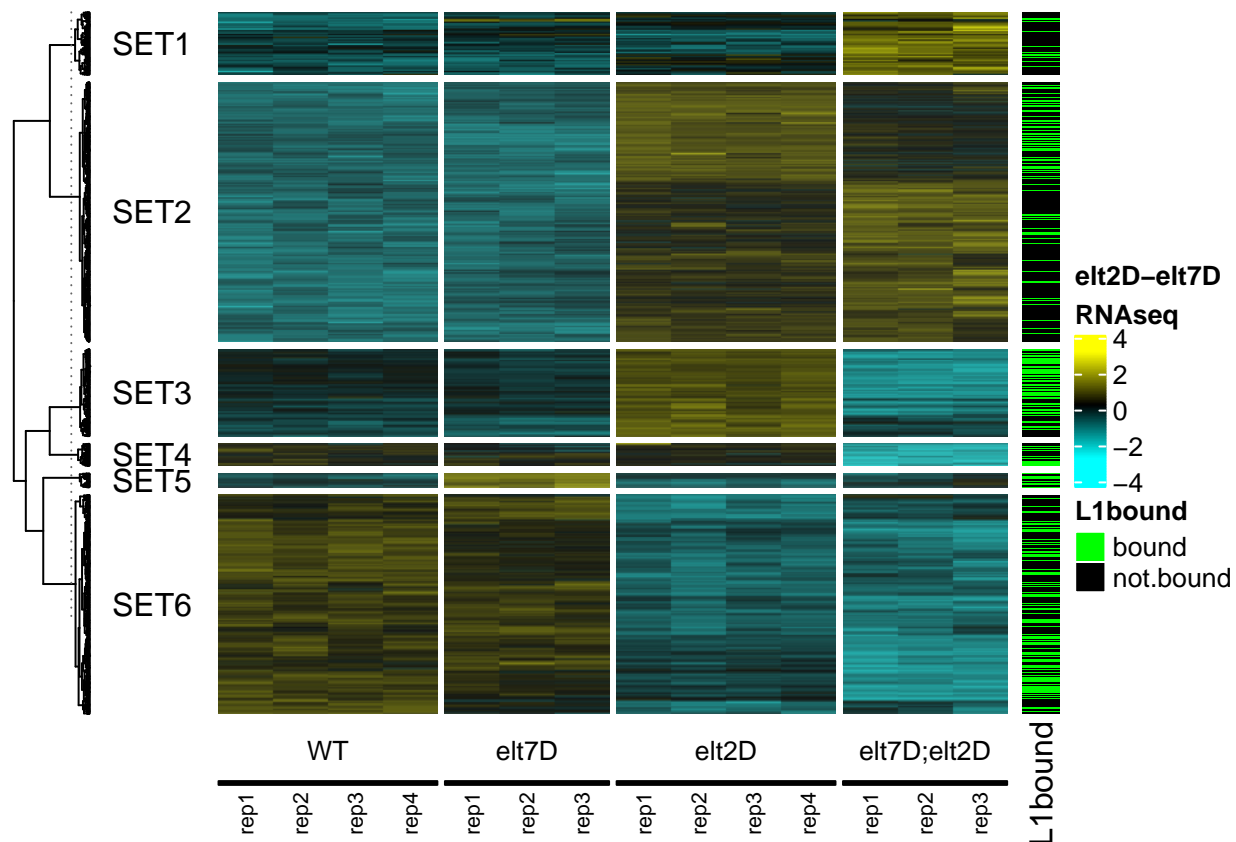
```
##           WBGeneID elt2_detected_in_L1
## 1 WBGene00000007          bound
```

```
## 2 WBGene00000008      bound
## 3 WBGene00000009      not.bound
## 4 WBGene00000013      not.bound
## 5 WBGene00000016      not.bound
## 6 WBGene00000017      not.bound
```

Incorporate this into a heatmap annotation

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

Ha_L1chip



```
if (plot == TRUE){
  myPDFplot(Ha_L1chip, "01a_DE_Heatmap_elt2elt7DERNAseq_L1elt2bound", 4, 4.5, plotdir)
}
```

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

Add Spencer intestine data

```
spencer_rna_anno <- data.frame(
  spencerLE = ifelse(
    test = rownames(dynamic_counts_matrix_scaled_ascend) %in% spencer_LE_subset$spencer_LE_ID,
    yes = "enriched",
    no = "depleted"
```



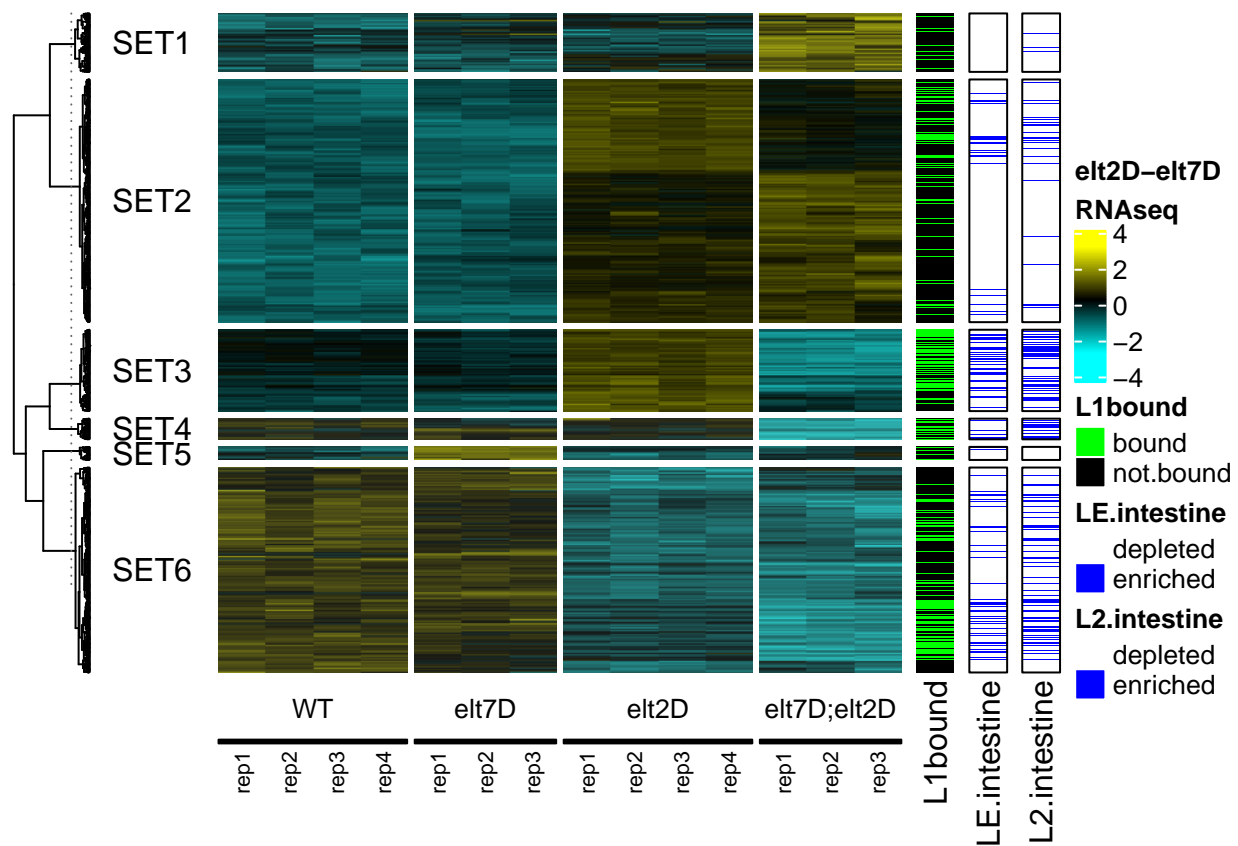
```

),
spencerL2 = ifelse(
  test = rownames(dynamic_counts_matrix_scaled_ascend) %in% spencer_L2_subset$spencer_L2_ID,
  yes = "enriched",
  no = "depleted"
)
)

Ha_L1chip_spencer <- Ha_L1chip +
  rowAnnotation(
    LE.intestine = spencer_rna_anno$spencerLE,
    col = list(LE.intestine = c(
      "enriched" = "blue", "depleted" = "white"
    )),
    border = TRUE
  ) +
  rowAnnotation(
    L2.intestine = spencer_rna_anno$spencerL2,
    col = list(L2.intestine = c(
      "enriched" = "blue", "depleted" = "white"
    )),
    border = TRUE
  )
)

Ha_L1chip_spencer

```



```
if (plot == TRUE) {
  myPDFplot(Ha_L1chip_spencer, "01b_DE_Heatmap_elt2elt7DERNAseq_L1elt2bound_spencerRNA", height = 6.5, w
}
```

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

Visually it appears that some elt2/elt7 differential expression clusters have more or less ELT-2 binding associated with the sets. I would like to be more quantitative with this assesment.

Determine enrichment of ELT-2 binding during L1 stage. I will calculate the percentage of genes with an ELT-2 ChIP-seq peak detected during the L1 stage.

First use `merge` to combine the ELT-2 binding status and expression set for each gene.

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

```
##           WBGeneID elt2_detected_in_L1  set
## 1 WBGene00000007              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
```

Next use `table` to tally the number of bound and not.bound genes per expression set.

```
clust_L1bound_counts <-
  table(expression_L1_binding$set,
        expression_L1_binding$elt2_detected_in_L1)
clust_L1bound_counts
```

```
##
##      bound not.bound
## SET1    48      243
## SET2   255      953
## SET3   216      189
## SET4    51       52
## SET5    26       39
## SET6   366     654
```

Use `prop.table` to convert these values to percentages within each set.

```
clust_L1bound_prop <- prop.table(clust_L1bound_counts, 1)
clust_L1bound_prop
```

```
##
##      bound not.bound
## SET1 0.1649485 0.8350515
## SET2 0.2110927 0.7889073
## SET3 0.5333333 0.4666667
## SET4 0.4951456 0.5048544
## SET5 0.4000000 0.6000000
## SET6 0.3588235 0.6411765
```

Adjust the percentages object into a dataframe that `ggplot2` can use.

```

clust_L1bound_prop_ggplot <- as.data.frame(clust_L1bound_prop)

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

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

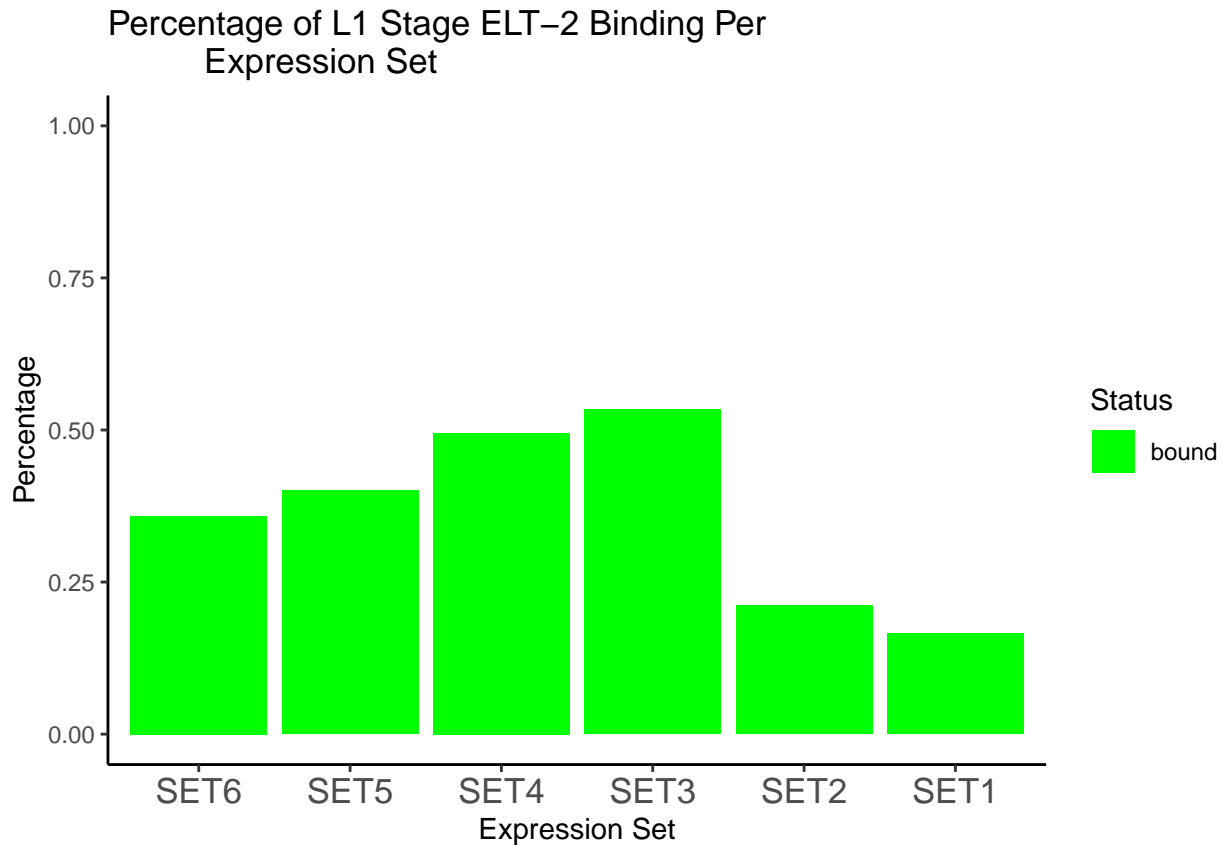
clust_L1bound_prop_ggplot$SET <-
  factor(
    clust_L1bound_prop_ggplot$SET,
    levels = c("SET6", "SET5", "SET4", "SET3", "SET2", "SET1")
  )

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

```



```
if (plot == TRUE){
myggsave(plot = l1bound_percents,
          name = "02_proportion_of_l1elt2_per_expression_cluster_200428",
          height = 2,
          width = 5,
          plotdir = plotdir)
}
```

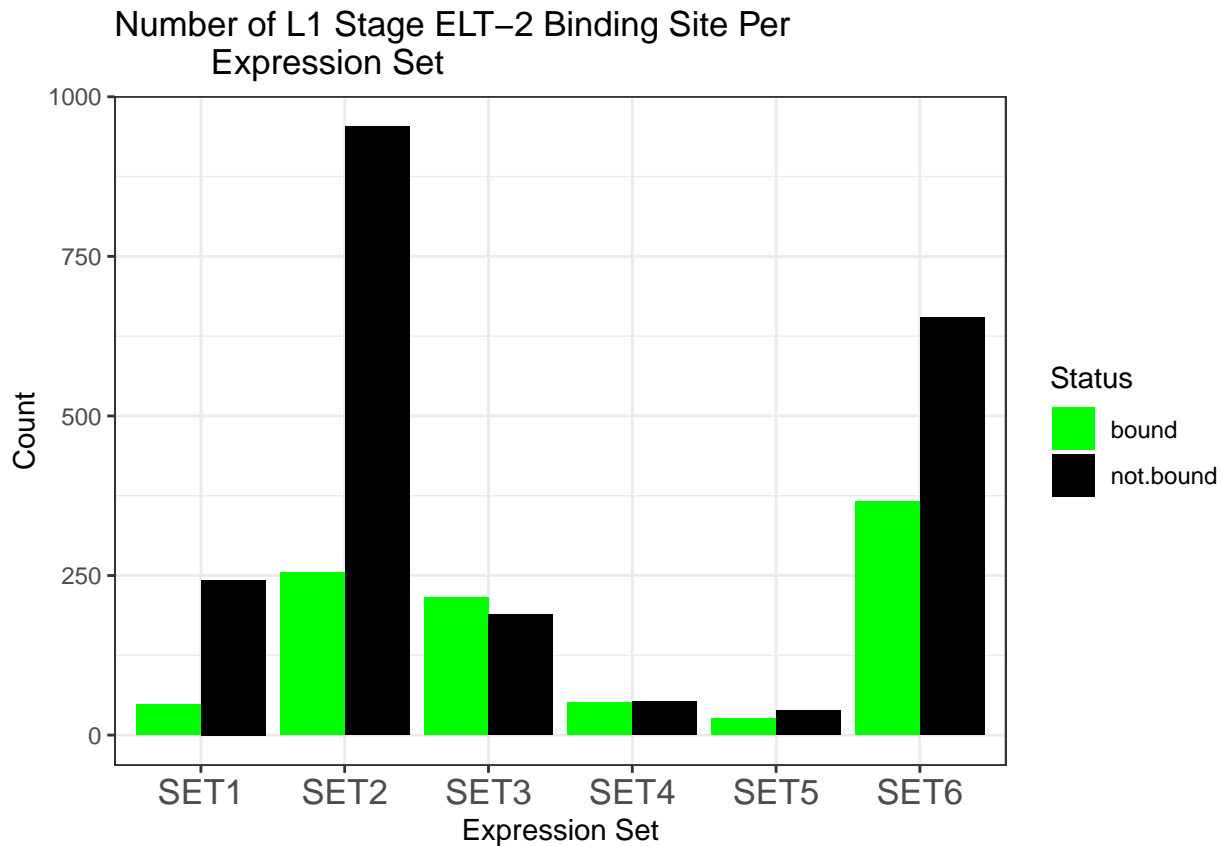
This plot shows that all of the differential expression sets have less than 50% of genes bound by ELT-2.

Rather than viewing percentages of genes bound, what is the number of “bound” vs “not.bound” per cluster?

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

bound_per_cluster <- 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))
```

```
bound_per_cluster
```



```
if (plot == TRUE){
myggsave(
  plot = bound_per_cluster,
  name = "03_number_of_l1elt2_per_expression_cluster",
  height = 2,
  width = 5,
  plotdir = plotdir
)
}
```

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

Use `binom.test` and first do 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] 0.3111255
```

Use custom function `ctable_binom()` to calculate p-value and confidence intervals for each set.

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

```
##      Set      pval conf.lower conf.upper  bool
```

```
## 1 SET1 1.823780e-08 0.1241945 0.2126797 TRUE
## 2 SET2 9.784777e-15 0.1883918 0.2352090 TRUE
## 3 SET3 2.243890e-20 0.4834195 0.5827566 TRUE
## 4 SET4 1.067986e-04 0.3951388 0.5954388 TRUE
## 5 SET5 1.399473e-01 0.2803996 0.5290211 FALSE
## 6 SET6 1.159516e-03 0.3293409 0.3891237 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.

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

```
##      Set          pval conf.lower conf.upper  bool
## 1 SET1 9.140980e-09          0 0.2049756  TRUE
## 2 SET2 4.714859e-15          0 0.2313324  TRUE
## 3 SET3 1.000000e+00          0 0.5750622 FALSE
## 4 SET4 9.999672e-01          0 0.5803138 FALSE
## 5 SET5 9.512219e-01          0 0.5095451 FALSE
## 6 SET6 9.994911e-01          0 0.3842976 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")
```

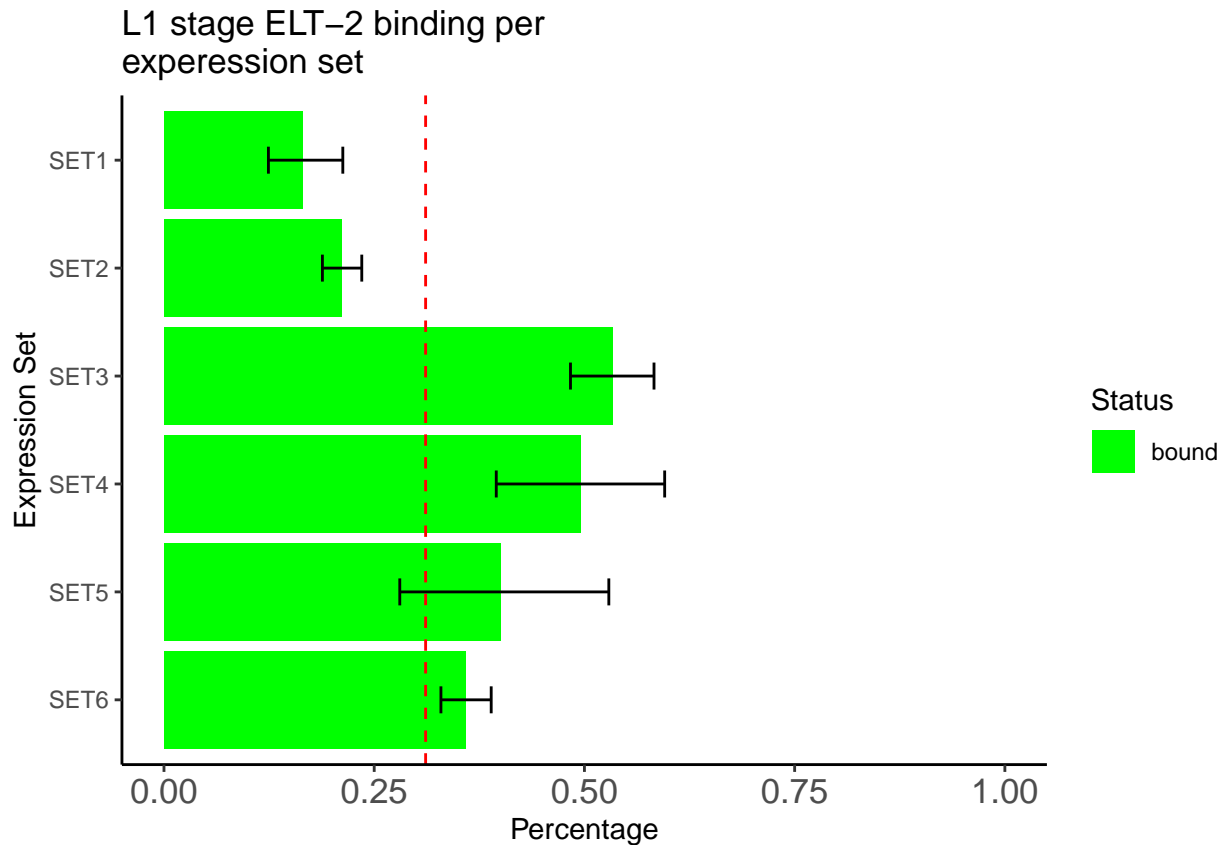
```
##      Set          pval conf.lower conf.upper  bool
## 1 SET1 1.000000e+00 0.1301165          1 FALSE
## 2 SET2 1.000000e+00 0.1919018          1 FALSE
## 3 SET3 1.670307e-20 0.4912403          1  TRUE
## 4 SET4 7.458666e-05 0.4101914          1  TRUE
## 5 SET5 8.076086e-02 0.2975268          1 FALSE
## 6 SET6 6.418951e-04 0.3339551          1  TRUE
```

This says that SET3, SET4 and SET6 have a higher percentage of genes bound compared the the “background” percent of bound genes for the entire dataset.

Make a plot that visually depicts this. Draw line on the percentage plot to indicate background percentage of L1 stage ELT-2 binding.

```
l1bound_percents_verticle <- 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() +
  ggtitle("L1 stage ELT-2 binding per\nexpression set")

l1bound_percents_verticle
```



```
if (plot == TRUE){
myggsave(
  plot = l1bound_percents_verticle,
  name = "04_percentage_l1bound_per_expression_cluster",
  width = 4,
  height = 5,
  plotdir = plotdir
)
}
```

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] 3.779117e-113
```

A very small p-value for the hypergeometric test suggests that the entire dataset is enriched for ELT-2.

The next section will compute pairwise Fisher's exact tests for the different sets. I have a difficult time interpreting these results.

```
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 5.808e-23 1.189e-31      -      -      -
## SET4 4.510e-09 2.175e-08 1.000e+00      -      -
## SET5 1.101e-03 1.521e-02 9.100e-01 1.0000      -
## SET6 1.253e-09 1.810e-13 3.057e-08 0.1112      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 TRUE  TRUE FALSE FALSE  NA
## SET6 TRUE  TRUE  TRUE FALSE FALSE
```

Row annotation of ELT-2 Binding Pattern Clusters

This section will add annotation to the rows of the elt2/elt7 differential expression heatmap with ELT-2 ChIP-seq binding pattern clusters. This will determine what differential expression sets associate with ELT-2 binding patterns.

Start by using custom function `make_cluster_annotation()`. This function takes two objects: the matrix of gene expression values and a dataframe of counts ELT-2 binding patterns per genes. It returns a dataframe with the number of ELT-2 binding categories associated with each gene.

```
chip_annotation <-
  make_cluster_annotation(dynamic_counts_matrix_scaled_ascend,
                          binding_cluster_gene_counts)

chip_annotation %>% head()
```

```
##          WBGeneID Embryo_Specific Larval Increasing L3_High Not_Changing
## 1 WBGene00000007      0      0      2      0      0
## 2 WBGene00000008      0      0      1      0      0
## 3 WBGene00000009      0      0      0      0      1
## 4 WBGene00000013      0      0      0      0      0
## 5 WBGene00000016      0      0      0      0      0
## 6 WBGene00000017      0      0      0      0      0
```

Sanity check to ensure that the order and number of rows is preserved.

```
unique(rownames(dynamic_counts_matrix_scaled_ascend) == chip_annotation$WBGeneID)
```

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

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

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

Build add row annotation for the number of ELT-2 binding clusters associated with each gene.

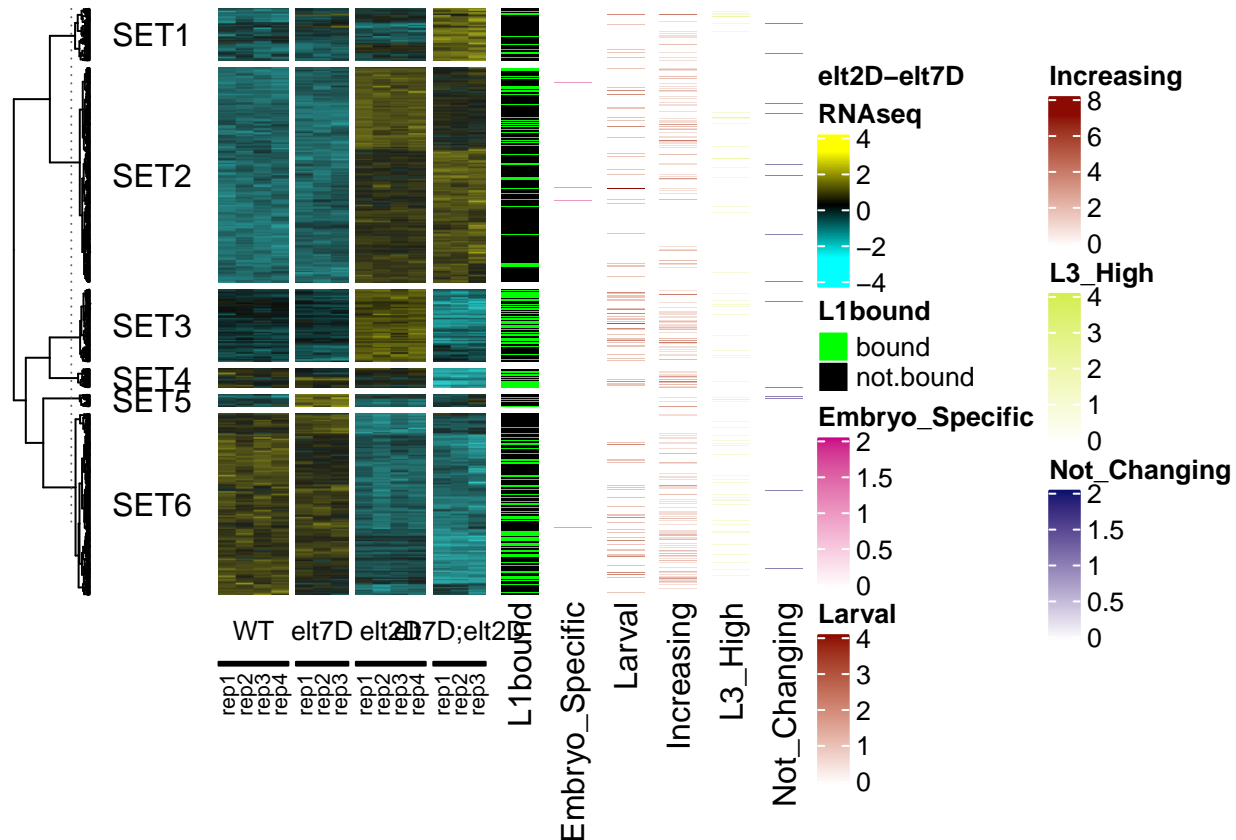
```
Ha_L1chip_bindcluster <- Ha_L1chip +
  rowAnnotation(Embryo_Specific = chip_annotation$Embryo_Specific) +
```



```

rowAnnotation(Larval = chip_annotation$Larval) +
rowAnnotation(Increasing = chip_annotation$Increasing) +
rowAnnotation(L3_High = chip_annotation$L3_High) +
rowAnnotation(Not_Changing = chip_annotation$Not_Changing)
Ha_L1chip_bindcluster

```



Have the colors match plot from David.

```

cluster_colors <-
data.frame(
  class = elt2_cluster_names,
  val = c("#7570B3", "#1B9E77", "#E7298A", "#D95F02", "#505050")
)

cluster_colors$class <-
factor(x = cluster_colors$class,
  levels = elt2_cluster_names)

```

Convert ChIP binding clusters to a present/absence list.

```

chip_annotation_present_absent <-
make_cluster_binary_annotation(chip_annotation)

```

Plot the heatmap with presence/absence.

```

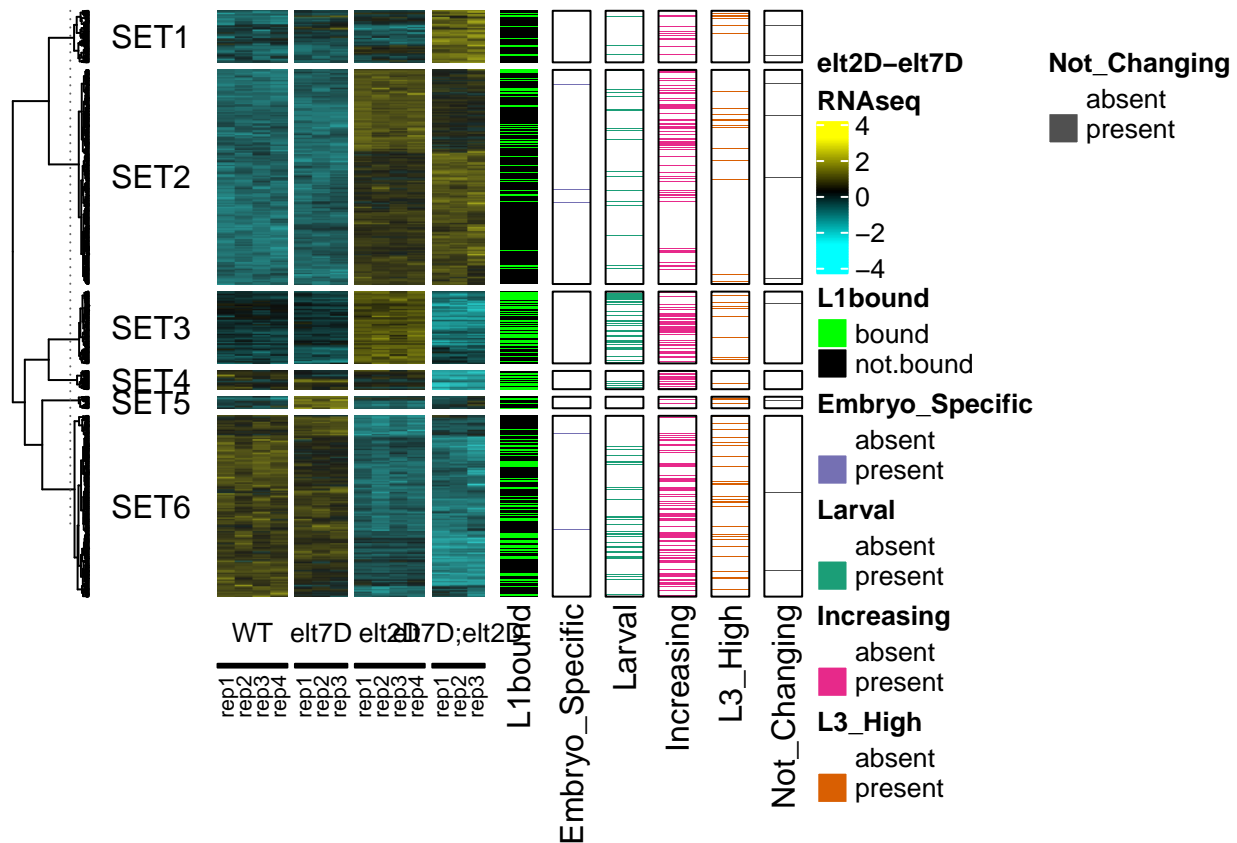
Ha_L1chip_clusterchip <-
Ha_L1chip + binding_cluster_row_annotation(chip_annotation_present_absent)

```

```

Ha_L1chip_clusterchip

```

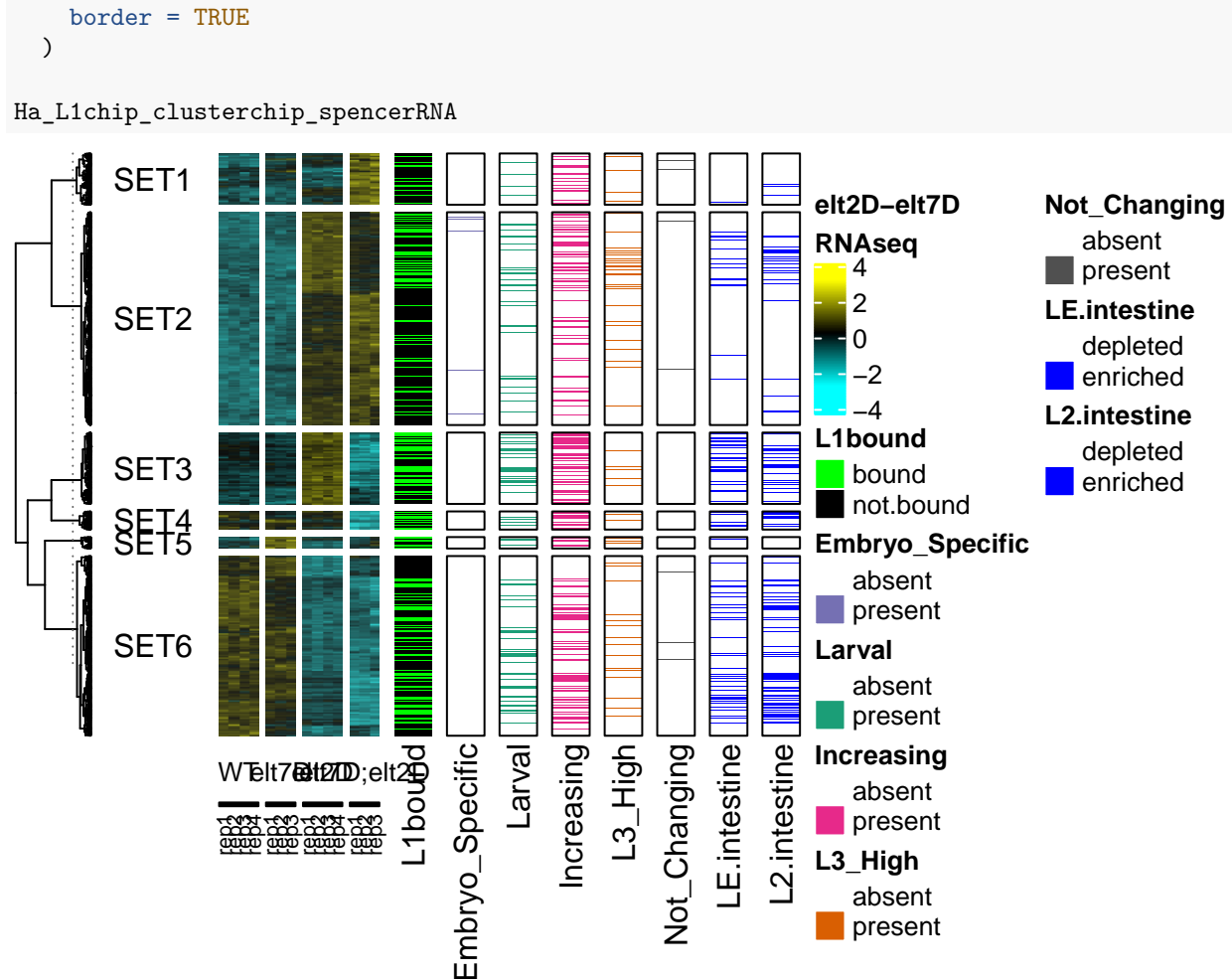


```
if(plot == TRUE){
  myPDFplot(
    plot = Ha_L1chip_clusterchip,
    name = "05a_DE_Heatmap_L1elt2bound_elt2bindclusters_anno",
    height = 6.5,
    width = 6,
    plotdir = plotdir
  )
}
```

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

Add Spencer intestine RNA row annotation

```
Ha_L1chip_clusterchip_spencerRNA <- Ha_L1chip_clusterchip +
  rowAnnotation(
    LE.intestine = spencer_rna_anno$spencerLE,
    col = list(LE.intestine = c(
      "enriched" = "blue", "depleted" = "white"
    )),
    border = TRUE
  ) +
  rowAnnotation(
    L2.intestine = spencer_rna_anno$spencerL2,
    col = list(L2.intestine = c(
      "enriched" = "blue", "depleted" = "white"
    )),
```



```
if (plot == TRUE){
  myPDFplot(
    plot = Ha_L1chip_clusterchip_spencerRNA,
    name = "05b_DE_Heatmap_L1elt2bound_elt2bindclusters_spencerRNA_anno",
    height = 6.5,
    width = 8,
    plotdir = plotdir
  )
}
```

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

Plot percentage of expression cluster group having binding pattern assignment.

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

```
exprclust_bindclust %>% head
```

```
##           WBGeneID set Embryo_Specific Larval Increasing L3_High Not_Changing
## 1 WBGene00000007 SET6          absent absent    present  absent      absent
## 2 WBGene00000008 SET6          absent absent    present  absent      absent
## 3 WBGene00000009 SET3          absent absent    absent   absent      present
## 4 WBGene00000013 SET1          absent absent    absent   absent      absent
## 5 WBGene00000016 SET1          absent absent    absent   absent      absent
## 6 WBGene00000017 SET1          absent absent    absent   absent      absent
```

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

Make a dataframe that addresses the question:

```
expressionSet_per_BindingCluster <- data.frame()
for (i in elt2_cluster_names) {
  toappend1 <-
    table(exprclust_bindclust$set,
          exprclust_bindclust[[i]]) %>%
    as.data.frame.matrix() %>%
    rownames_to_column(var = "set")
  toappend2 <- mutate(toappend1, ELT2_cluster = i,
                     percent = present / (present + absent))
  expressionSet_per_BindingCluster <-
    bind_rows(expressionSet_per_BindingCluster, toappend2)
}

expressionSet_per_BindingCluster$ELT2_cluster <-
  factor(expressionSet_per_BindingCluster$ELT2_cluster, levels = elt2_cluster_names)

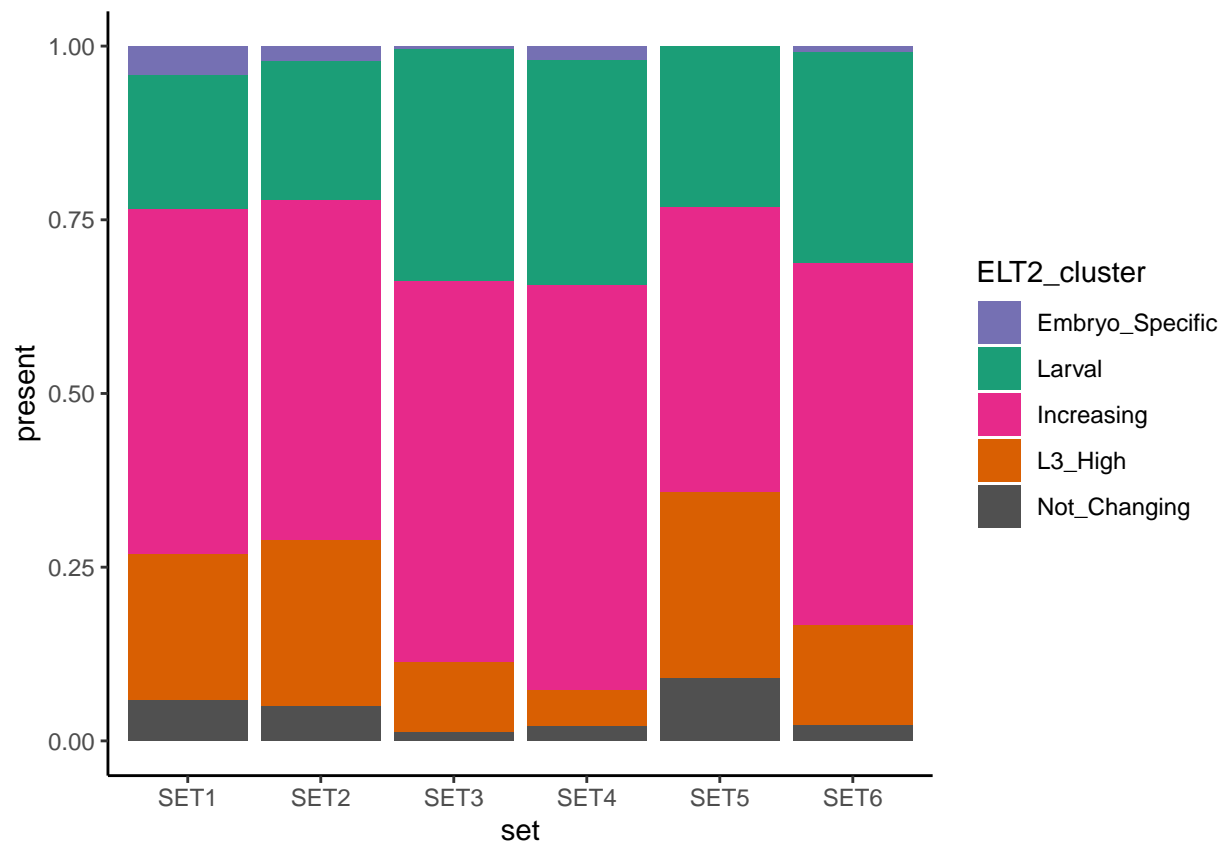
expressionSet_per_BindingCluster
```

```
##      set absent present  ELT2_cluster  percent
## 1 SET1    286      5 Embryo_Specific 0.017182131
## 2 SET2   1195     13 Embryo_Specific 0.010761589
## 3 SET3    403      2 Embryo_Specific 0.004938272
## 4 SET4    101      2 Embryo_Specific 0.019417476
## 5 SET5     65      0 Embryo_Specific 0.000000000
## 6 SET6   1014      6 Embryo_Specific 0.005882353
## 7 SET1    268     23      Larval    0.079037801
## 8 SET2   1093    115      Larval    0.095198675
## 9 SET3    270    135      Larval    0.333333333
## 10 SET4    72     31      Larval    0.300970874
## 11 SET5    52     13      Larval    0.200000000
## 12 SET6    814    206      Larval    0.201960784
## 13 SET1    232     59    Increasing 0.202749141
## 14 SET2    925    283    Increasing 0.234271523
## 15 SET3    184    221    Increasing 0.545679012
## 16 SET4     47     56    Increasing 0.543689320
## 17 SET5     42     23    Increasing 0.353846154
## 18 SET6    667    353    Increasing 0.346078431
## 19 SET1    266     25      L3_High  0.085910653
```

```
## 20 SET2    1070    138      L3_High 0.114238411
## 21 SET3     364     41      L3_High 0.101234568
## 22 SET4      98      5      L3_High 0.048543689
## 23 SET5      50     15      L3_High 0.230769231
## 24 SET6     922     98      L3_High 0.096078431
## 25 SET1     284      7    Not_Changing 0.024054983
## 26 SET2    1179     29    Not_Changing 0.024006623
## 27 SET3     400      5    Not_Changing 0.012345679
## 28 SET4     101      2    Not_Changing 0.019417476
## 29 SET5      60      5    Not_Changing 0.076923077
## 30 SET6    1005     15    Not_Changing 0.014705882
```

Make a plot that addresses the question: What is the percentage of genes with annotated ELT2 binding clusters per expression dataset?

```
percent_bind_cluster_per_DE_set <- ggplot(expressionSet_per_BindingCluster,
  aes(x = set,
    y = present,
    fill = ELT2_cluster)) +
  geom_bar(stat = "identity", position = "fill") +
  theme_classic() +
  scale_fill_manual(values = as.vector(cluster_colors$val))
percent_bind_cluster_per_DE_set
```



```
# ggsave("./03_plots/06_Cluster_percent_present_per_Set_200615.pdf")

if (plot == TRUE){
  myggsave(plot = percent_bind_cluster_per_DE_set,
```

```

    name = "06_Cluster_percent_present_per_Set",
    plotdir = plotdir,
    width = 5,
    height = 3)
}

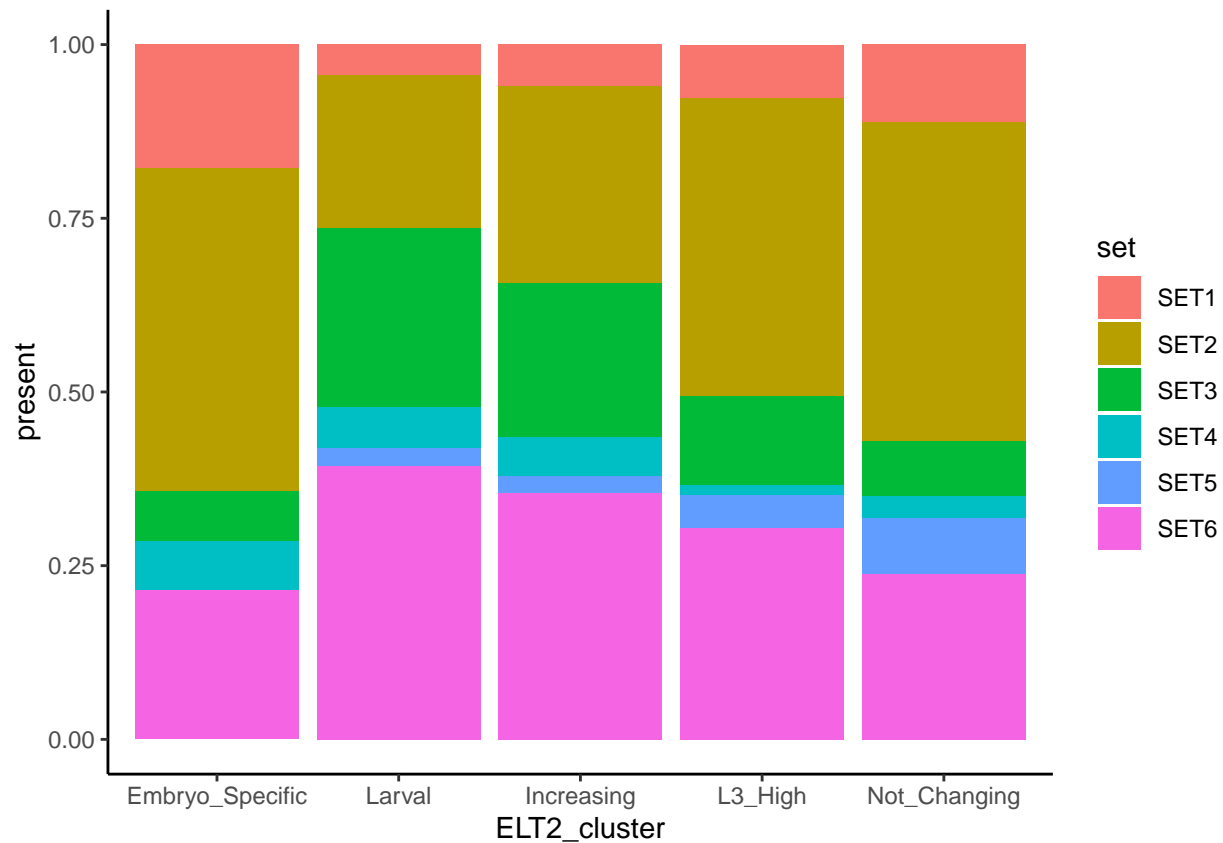
```

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

```

expressionSet_per_BindingCluster_plot <- ggplot(expressionSet_per_BindingCluster,
    aes(x = ELT2_cluster, y = present, fill = set)) +
  geom_bar(stat = "identity", position = "fill") +
  theme_classic()
expressionSet_per_BindingCluster_plot

```



```

if (plot == TRUE){
  myggsave(plot = expressionSet_per_BindingCluster_plot,
    name = "07_Set_percent_present_per_Cluster",
    plotdir = plotdir,
    width = 5,
    height = 3)
}

```

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_EL2_cluster <-
  expressionSet_per_BindingCluster %>% group_by(EL2_cluster) %>% summarise(percent = sum(present) /
                                                                    nrow(dynamic_counts_matrix))
```

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

```
expressionSet_per_BindingCluster %>% group_by(set, EL2_cluster) %>% summarise(percent = present /
                                                                    (present + absent))
```

```
## # A tibble: 30 x 3
## # Groups:   set [6]
##   set   EL2_cluster   percent
##   <chr> <fct>         <dbl>
## 1 SET1  Embryo_Specific  0.0172
## 2 SET1  Larval           0.0790
## 3 SET1  Increasing       0.203
## 4 SET1  L3_High         0.0859
## 5 SET1  Not_Changing    0.0241
## 6 SET2  Embryo_Specific  0.0108
## 7 SET2  Larval           0.0952
## 8 SET2  Increasing       0.234
## 9 SET2  L3_High         0.114
## 10 SET2 Not_Changing    0.0240
## # ... with 20 more rows
```

Calculate the binomial pvalue and confidence intervals.

```
# Add a column for the background percentage of EL2 binding clusters per the whole expression dataset
expression_binding_stats <-
  expressionSet_per_BindingCluster %>% group_by(EL2_cluster) %>% mutate(background_percent = sum(present) /
                                                                    (sum(present) + sum(absent)))

# Use binom.test to calculate pvalue and confidence intervals for the percentage of EL2 binding clusters
expression_binding_stats <- expression_binding_stats %>%
  group_by(EL2_cluster, set) %>%
  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$set <-
  factor(
    expression_binding_stats$set,
    levels = c("SET6", "SET5", "SET4", "SET3", "SET2", "SET1")
  )

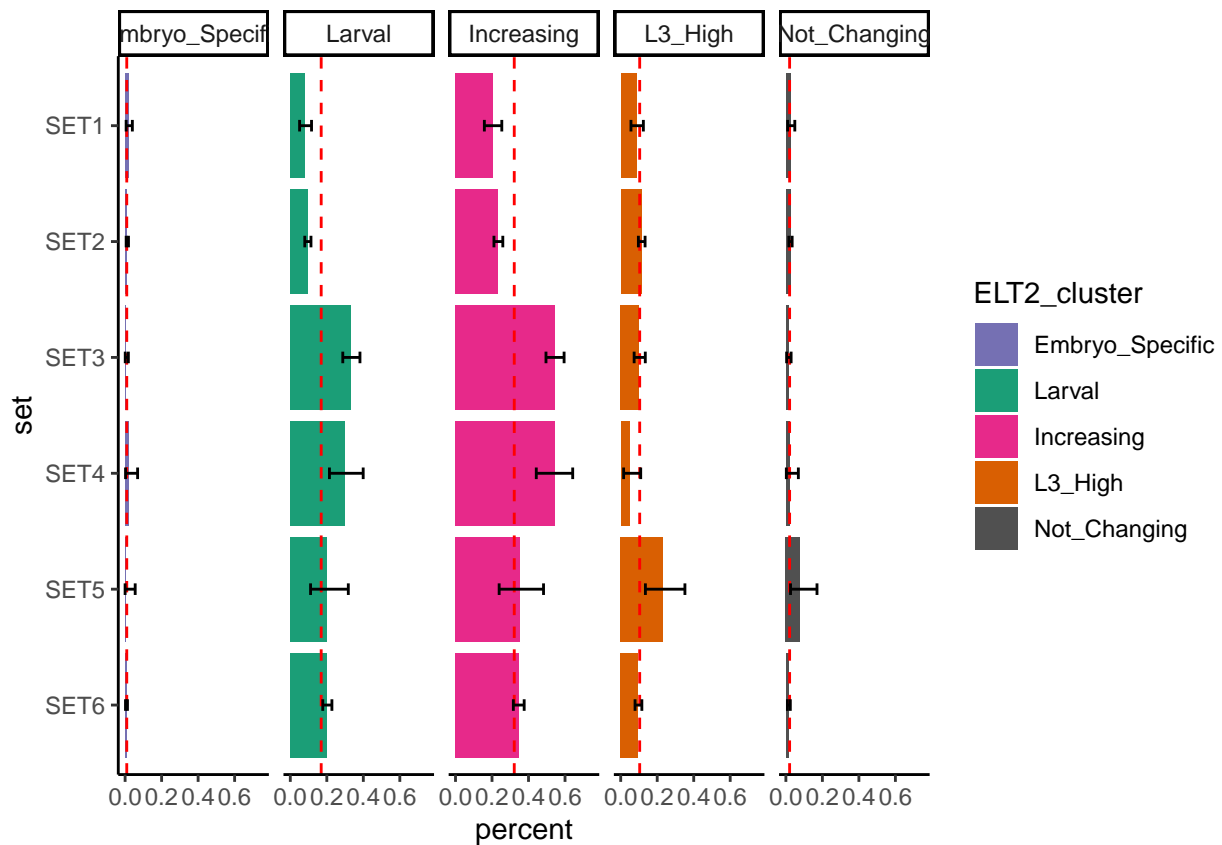
expression_binding_stats %>% head()

## # A tibble: 6 x 9
## # Groups:   ELT2_cluster, set [6]
##   set   absent present ELT2_cluster percent background_perc~ pval conf.upper
##   <fct> <int>   <int> <fct>         <dbl>         <dbl> <dbl>   <dbl>
## 1 SET1     286     5 Embryo_Spec~ 0.0172         0.00906 0.198   0.0396
## 2 SET2    1195    13 Embryo_Spec~ 0.0108         0.00906 0.540   0.0183
## 3 SET3     403     2 Embryo_Spec~ 0.00494        0.00906 0.596   0.0177
## 4 SET4     101     2 Embryo_Spec~ 0.0194         0.00906 0.239   0.0684
## 5 SET5      65     0 Embryo_Spec~ 0              0.00906 1       0.0552
## 6 SET6    1014     6 Embryo_Spec~ 0.00588        0.00906 0.405   0.0128
## # ... with 1 more variable: conf.lower <dbl>

per_cluster_pattern_percent <- ggplot(expression_binding_stats,
  aes(x = set,
      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))

per_cluster_pattern_percent

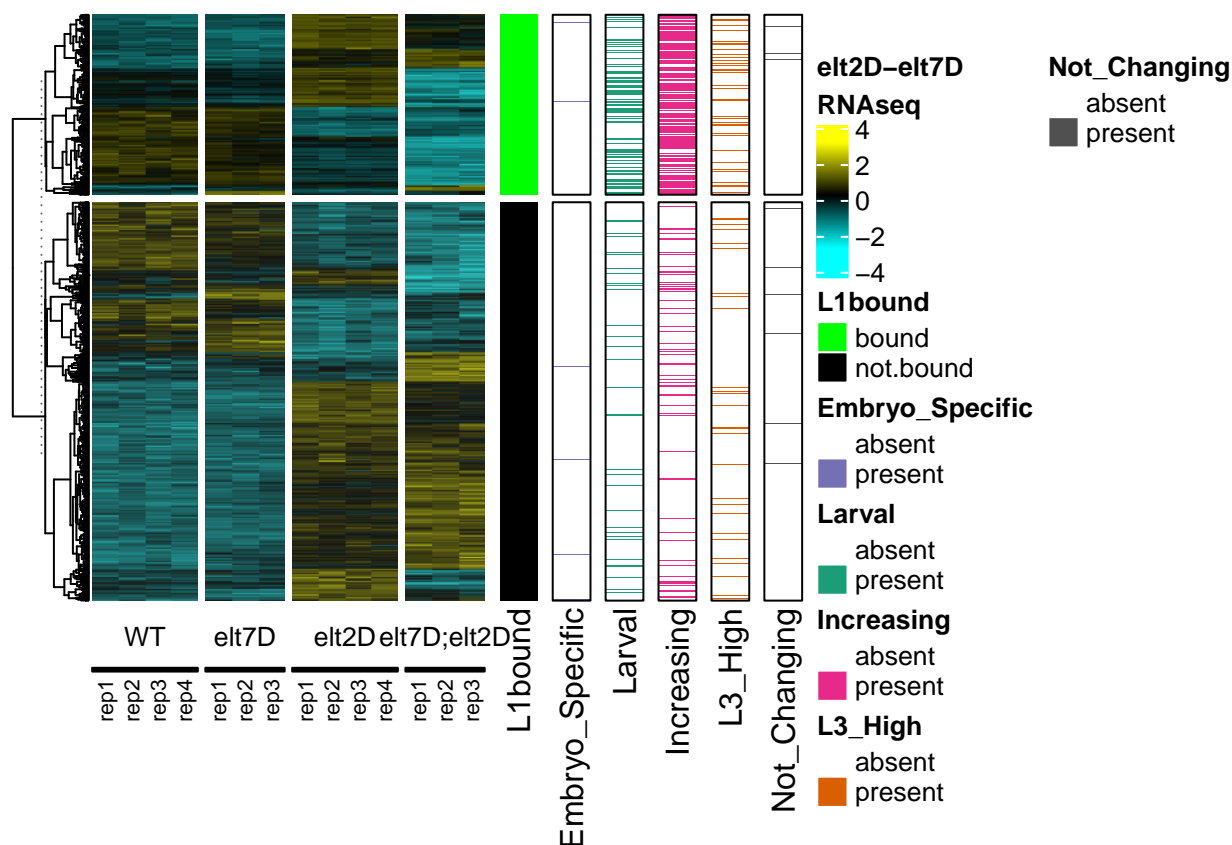
```

```
if (plot == TRUE){
  myggsave(plot = per_cluster_pattern_percent,
    name = "08_Percent_of_ELt2bindClust_per_ExpressionClustf",
    height = 5,
    width = 7,
    plotdir = plotdir)
}
```

Subset ELT-2/ELT-7 differentially expressed genes based on ELT-2 binding in L1 stage

```
RNA_heatmap2(
  dynamic_counts_matrix_scaled_ascend,
  column_split = RNA_column_order,
  row_split = elt2_L1_anno$elt2_detected_in_L1
) +
  elt2_l1_row_annotation(elt2_L1_anno) +
  binding_cluster_row_annotation(chip_annotation_present_absent)
```



```
l1_bound_list <-
  elt2_L1_anno %>% filter(elt2_detected_in_L1 == "bound") %>% select(WBGeneID) %>% arrange(WBGeneID)

dynamic_counts_matrix_scaled_bound_only <-
  matrix_select(dynamic_counts_matrix_scaled_ascend, l1_bound_list$WBGeneID)

bound_only_elt2_clust_anno <-
  make_cluster_binary_annotation(
    make_cluster_annotation(
      dynamic_counts_matrix_scaled_bound_only,
      binding_cluster_gene_counts
    )
  )

bound_only_elt2_clust_anno %>% head()
```

##	WBGeneID	Embryo_Specific	Larval	Increasing	L3_High	Not_Changing
## 1	WBGene00000007	absent	absent	present	absent	absent
## 2	WBGene00000008	absent	absent	present	absent	absent
## 3	WBGene00000064	absent	absent	absent	present	absent
## 4	WBGene00000067	absent	present	present	absent	absent
## 5	WBGene00000107	absent	absent	present	present	absent
## 6	WBGene00000136	absent	present	present	absent	absent

Assign k-means clusters for rows before plotting

```
kclus <- kmeans(dynamic_counts_matrix_scaled_bound_only, 4)
bound_only_sets <-
```

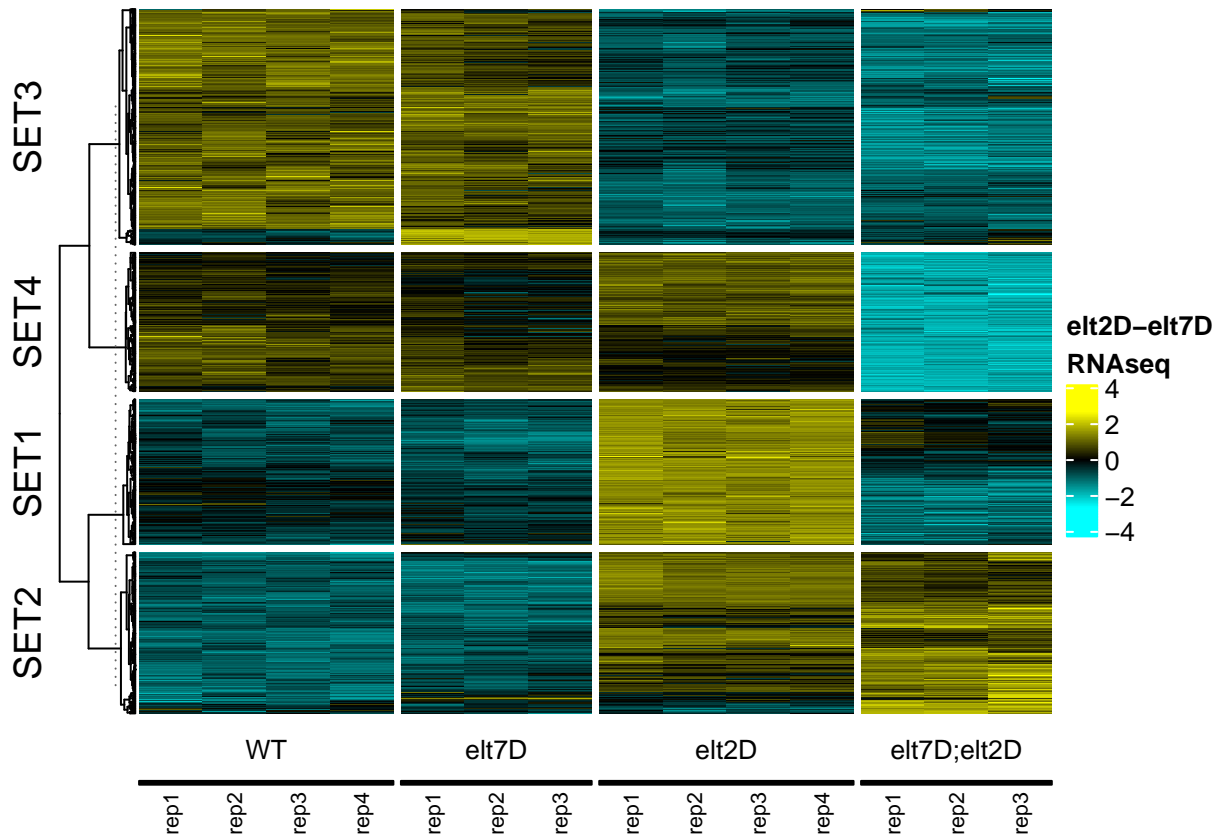
```
data.frame(
  WBGeneID = rownames(dynamic_counts_matrix_scaled_bound_only),
  set = paste("SET", kclus$cluster, sep = "")
)
head(bound_only_sets)
```

```
##      WBGeneID  set
## 1 WBGene00000007 SET2
## 2 WBGene00000008 SET2
## 3 WBGene00000064 SET4
## 4 WBGene00000067 SET1
## 5 WBGene00000107 SET2
## 6 WBGene00000136 SET2
```

Draw heatmap and check that set assignment is correct.

```
Ha_bound_only <-
  RNA_heatmap2(mat = dynamic_counts_matrix_scaled_bound_only,
    column_split = RNA_column_order,
    row_split = bound_only_sets$set,
    row_title = c("SET3", "SET4", "SET1", "SET2"))
```

Ha_bound_only



```
if (plot == TRUE){
  myPDFplot(
    plot = Ha_bound_only,
    name = "09a_DE_Heatmap_L1elt2boundOnly",

```

```

    height = 6.5, width = 6,
    plotdir = plotdir
  )
}

```

```

## pdf
## 2

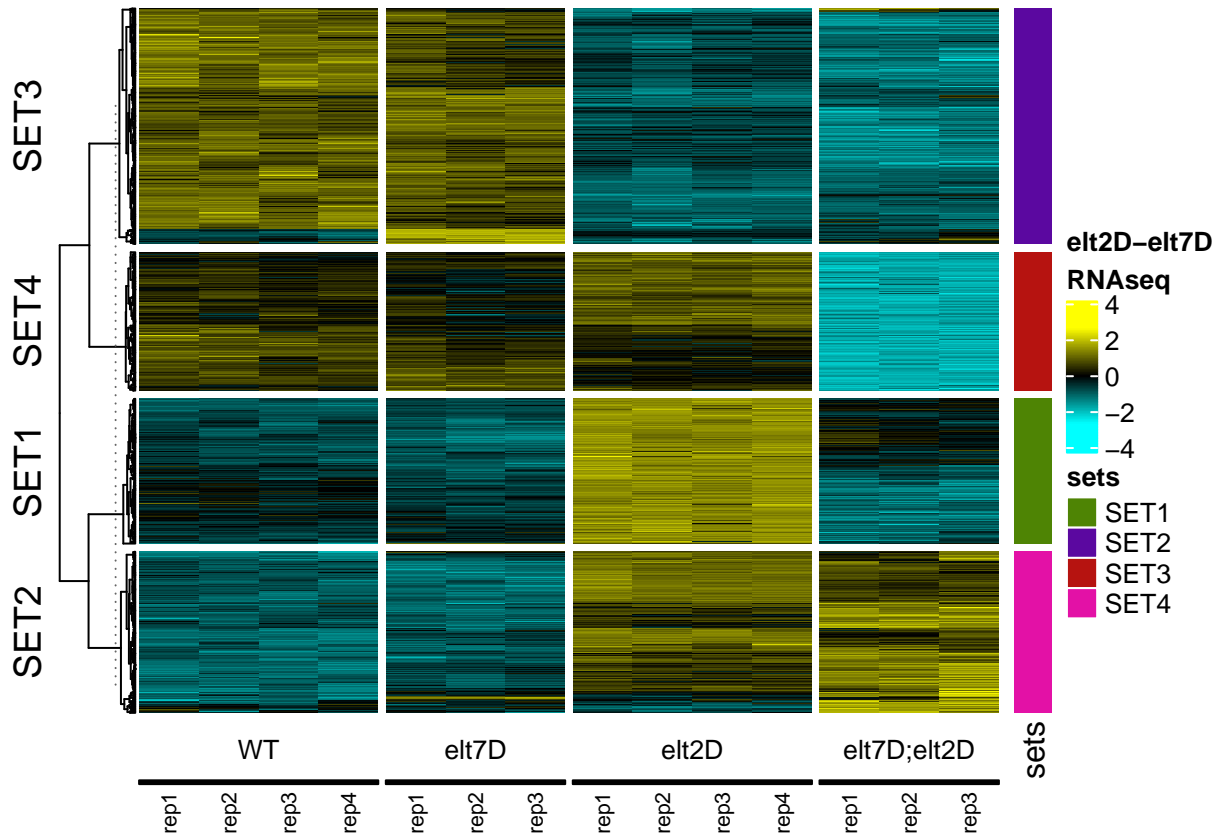
```

TODO: ROTATE CLUSTERS TO MAKE SENSE

```

Ha_bound_only +
  rowAnnotation(sets = bound_only_sets$set)

```



```

bound_only_annotation <-
  merge(bound_only_elt2_clust_anno,
    bound_only_sets,
    by.x = "WBGeneID",
    by.y = "WBGeneID")
bound_only_annotation_ascend <-
  bound_only_annotation %>% arrange(WBGeneID)
head(bound_only_annotation_ascend)

```

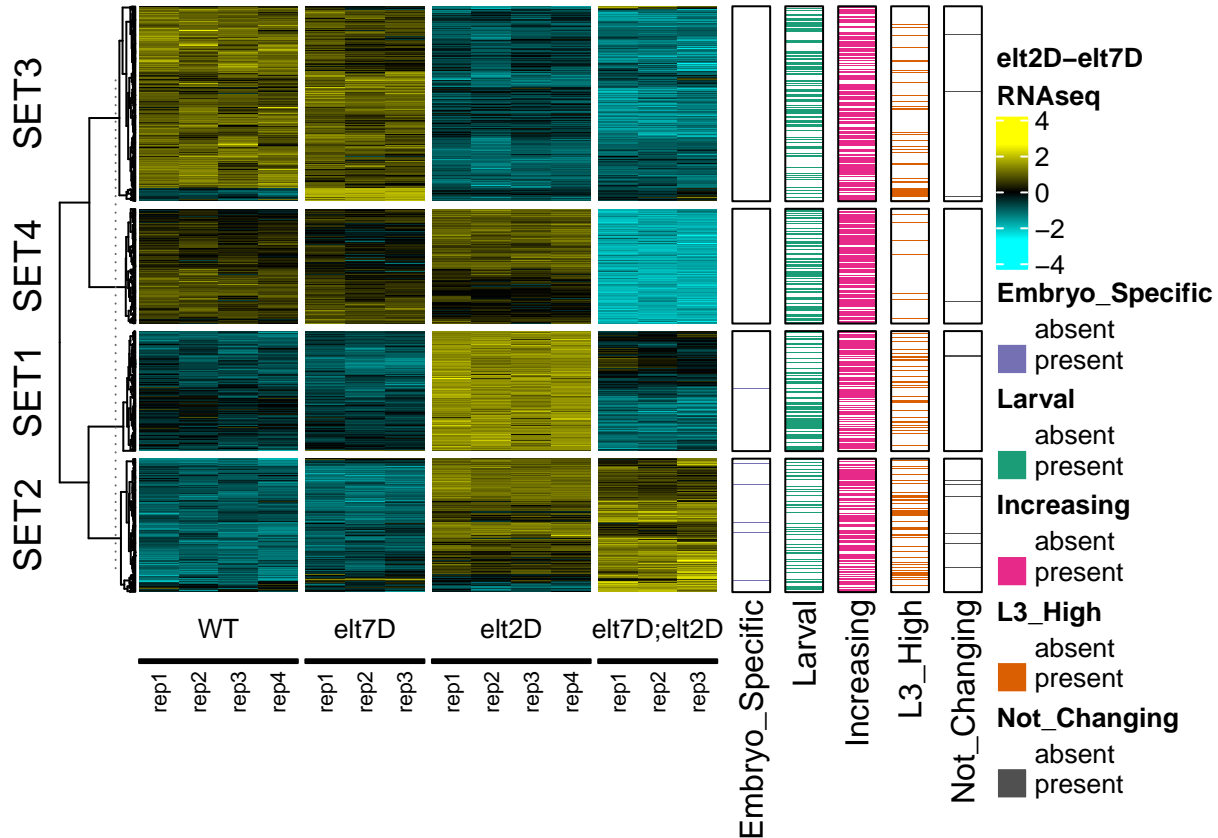
```

##           WBGeneID Embryo_Specific Larval Increasing L3_High Not_Changing set
## 1 WBGene00000007      absent absent      present absent      absent SET2
## 2 WBGene00000008      absent absent      present absent      absent SET2
## 3 WBGene00000064      absent absent      absent present      absent SET4
## 4 WBGene00000067      absent present    present absent      absent SET1

```

```
## 5 WBGene00000107      absent absent      present present      absent SET2
## 6 WBGene00000136      absent present     present absent      absent SET2
```

```
Ha_bound_only_chipClust <-
  Ha_bound_only + binding_cluster_row_annotation(bound_only_elt2_clust_anno)
Ha_bound_only_chipClust
```



```
if (plot == TRUE){
  myPDFplot(
    plot = Ha_bound_only_chipClust,
    name = "09b_DE_Heatmap_L1elt2boundOnly_elt2bindclusters_anno",
    height = 6.5,
    width = 6,
    plotdir = plotdir
  )
}
```

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

Add Spencer intestine expression row annotation

```
bound_only_spencer_rna_anno <- data.frame(
  spencerLE = ifelse(
    test = rownames(dynamic_counts_matrix_scaled_bound_only) %in% spencer_LE_subset$spencer_LE_ID,
    yes = "enriched",
    no = "depleted"
  ),
```

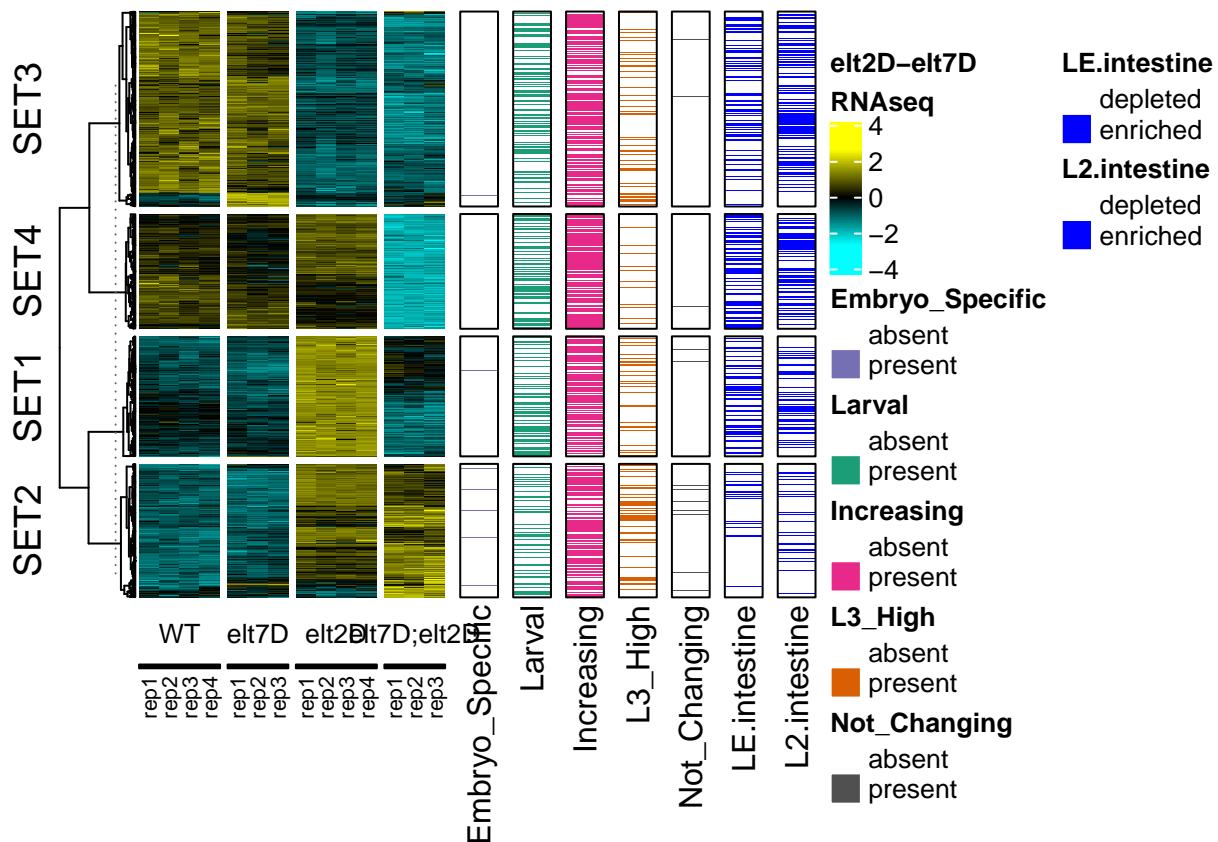
```

spencerL2 = ifelse(
  test = rownames(dynamic_counts_matrix_scaled_bound_only) %in% spencer_L2_subset$spencer_L2_ID,
  yes = "enriched",
  no = "depleted"
)
)

Ha_bound_only_chipClust_spencer <- Ha_bound_only_chipClust +
  rowAnnotation(
    LE.intestine = bound_only_spencer_rna_anno$spencerLE,
    col = list(LE.intestine = c(
      "enriched" = "blue", "depleted" = "white"
    )),
    border = TRUE
  ) +
  rowAnnotation(
    L2.intestine = bound_only_spencer_rna_anno$spencerL2,
    col = list(L2.intestine = c(
      "enriched" = "blue", "depleted" = "white"
    )),
    border = TRUE
  )
)

Ha_bound_only_chipClust_spencer

```



```

if (plot == TRUE){
  myPDFplot(

```

```

plot = Ha_bound_only_chipClust_spencer,
name = "09c_DE_Heatmap_L1elt2boundOnly_elt2bindclusters_spencerRNA",
height = 6.5,
width = 6,
plotdir = plotdir
)
}

```

```

## pdf
## 2

```

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

```

bound_only_exprclust_bindclust <-
  merge(bound_only_sets,
        chip_annotation_present_absent,
        by.x = "WBGeneID",
        by.y = "WBGeneID")

```

```
bound_only_exprclust_bindclust %>% head
```

```

##           WBGeneID set Embryo_Specific Larval Increasing L3_High Not_Changing
## 1 WBGene00000007 SET2             absent absent      present absent      absent
## 2 WBGene00000008 SET2             absent absent      present absent      absent
## 3 WBGene00000064 SET4             absent absent      absent present      absent
## 4 WBGene00000067 SET1             absent present    present absent      absent
## 5 WBGene00000107 SET2             absent absent      present present      absent
## 6 WBGene00000136 SET2             absent present    present absent      absent

```

Make a dataframe that addresses the question:

```

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

```

```

bound_only_expressionSet_per_BindingCluster$ELT2_cluster <-
  factor(bound_only_expressionSet_per_BindingCluster$ELT2_cluster,
        levels = elt2_cluster_names)

```

```
bound_only_expressionSet_per_BindingCluster
```

```

##      set absent present  ELT2_cluster    percent
## 1 SET1    202      3 Embryo_Specific 0.014634146
## 2 SET2    332      1 Embryo_Specific 0.003003003
## 3 SET3    196      0 Embryo_Specific 0.000000000

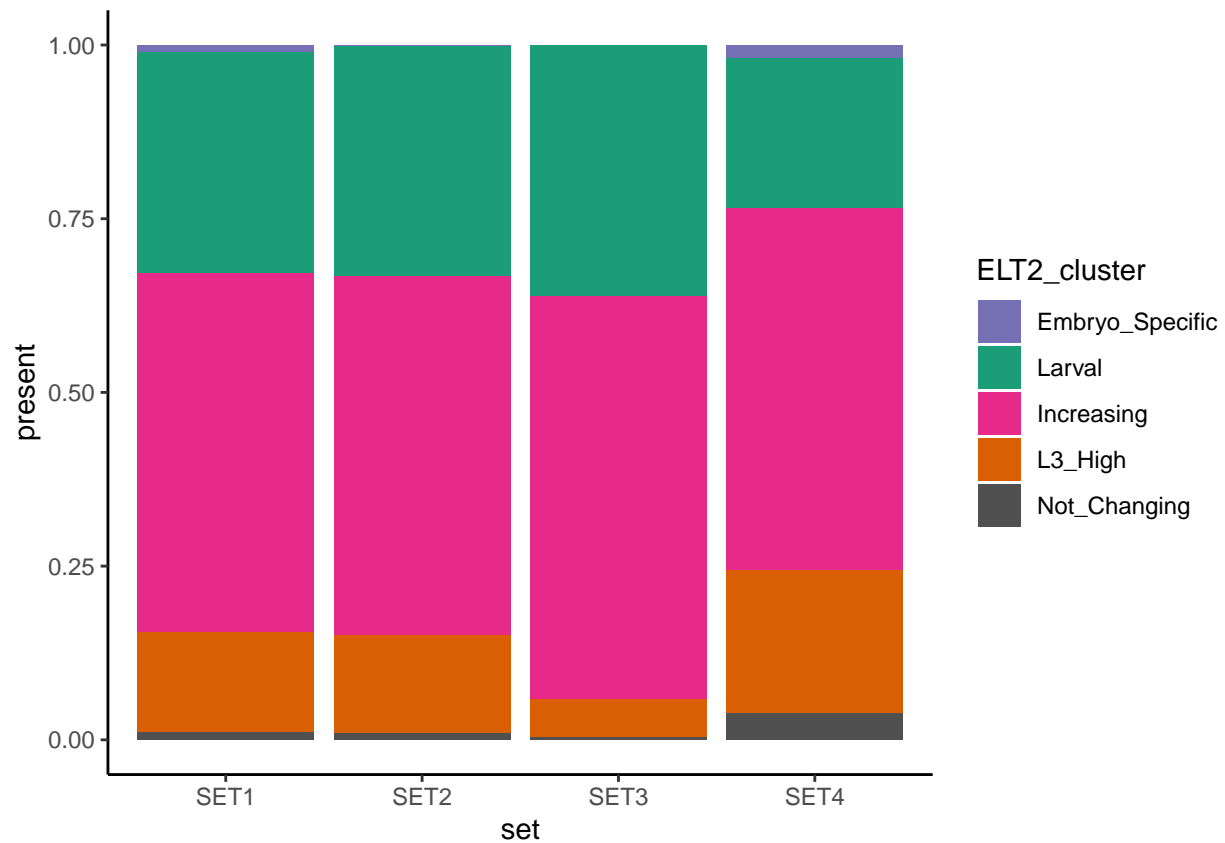
```

## 4	SET4	222	6	Embryo_Specific	0.026315789
## 5	SET1	115	90	Larval	0.439024390
## 6	SET2	191	142	Larval	0.426426426
## 7	SET3	97	99	Larval	0.505102041
## 8	SET4	160	68	Larval	0.298245614
## 9	SET1	59	146	Increasing	0.712195122
## 10	SET2	111	222	Increasing	0.666666667
## 11	SET3	37	159	Increasing	0.811224490
## 12	SET4	63	165	Increasing	0.723684211
## 13	SET1	164	41	L3_High	0.200000000
## 14	SET2	272	61	L3_High	0.183183183
## 15	SET3	181	15	L3_High	0.076530612
## 16	SET4	163	65	L3_High	0.285087719
## 17	SET1	202	3	Not_Changing	0.014634146
## 18	SET2	329	4	Not_Changing	0.012012012
## 19	SET3	195	1	Not_Changing	0.005102041
## 20	SET4	216	12	Not_Changing	0.052631579

Make a plot that addresses the question: What is the percentage of genes with annotated ELT2 binding clusters per expression dataset?

```
bound_percent_plot <- ggplot(
  bound_only_expressionSet_per_BindingCluster,
  aes(x = set,
      y = present,
      fill = ELT2_cluster)
) +
  geom_bar(stat = "identity", position = "fill") +
  theme_classic() +
  scale_fill_manual(values = as.vector(cluster_colors$val))

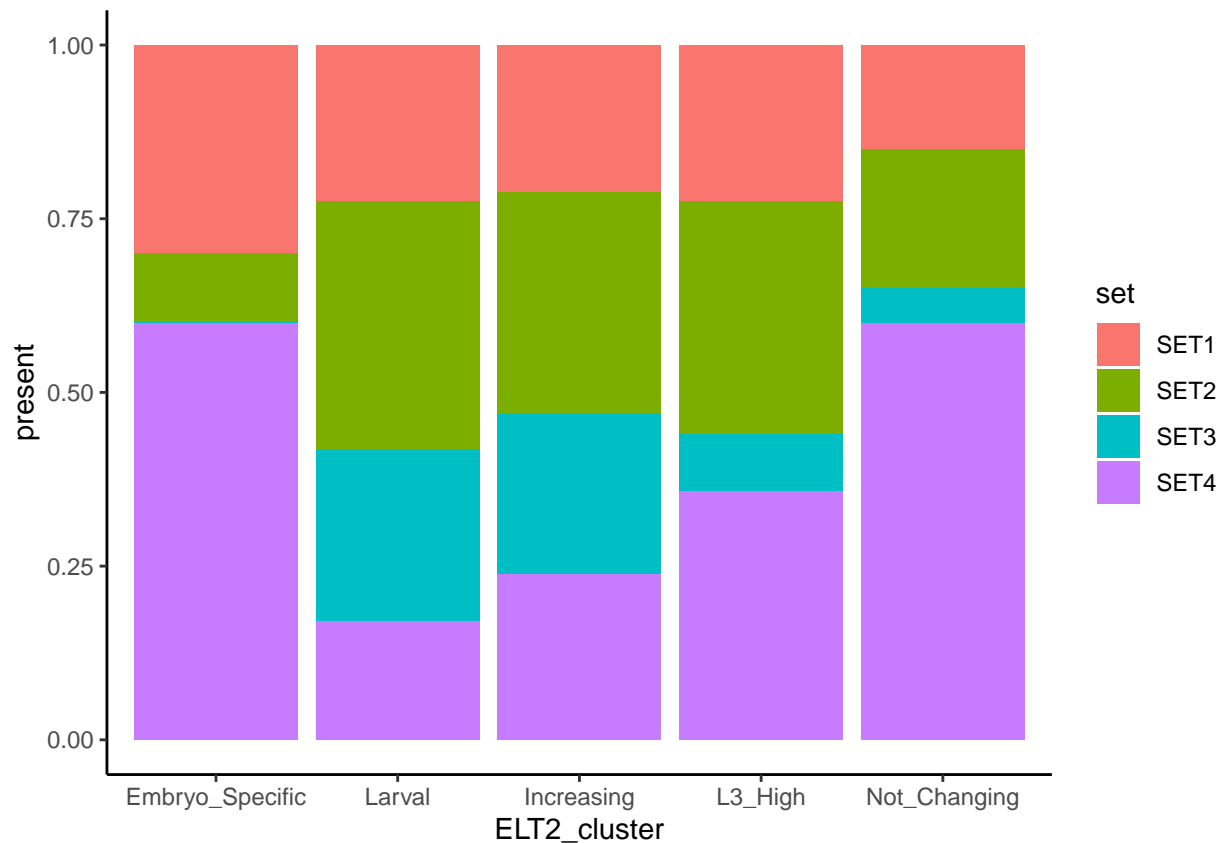
bound_percent_plot
```

```
if (plot == TRUE){
  myggsave(plot = bound_percent_plot,
    name = "10a_Bound_Only_Cluster_percent_present_per_Set",
    height = 3,
    width = 5,
    plotdir = plotdir)
}
```

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

```
bound_percent_plot_inverse <- ggplot(
  bound_only_expressionSet_per_BindingCluster,
  aes(x = ELT2_cluster, y = present, fill = set)
) +
  geom_bar(stat = "identity", position = "fill") +
  theme_classic()
bound_percent_plot_inverse
```



```
if (plot == TRUE){
  myggsave(plot = bound_percent_plot_inverse,
    name = "10b_Bound_Only_Set_percent_present_per_Cluster",
    height = 3,
    width = 5,
    plotdir = plotdir)
}
```

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.

```
bound_only_percent_bound_per_ELt2_cluster <-
  bound_only_expressionSet_per_BindingCluster %>% group_by(ELT2_cluster) %>% summarise(percent = sum(pr
    nrow(dynamic_counts_matrix))
```

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

```
bound_only_expressionSet_per_BindingCluster %>% group_by(set, ELT2_cluster) %>% summarise(percent = pre
  (present + absent))
```

```
## # A tibble: 20 x 3
## # Groups:   set [4]
##   set  ELT2_cluster  percent
##   <chr> <fct>         <dbl>
## 1 SET1 Embryo_Specific 0.0146
## 2 SET1 Larval         0.439
## 3 SET1 Increasing     0.712
## 4 SET1 L3_High        0.2
```

```
## 5 SET1 Not_Changing 0.0146
## 6 SET2 Embryo_Specific 0.00300
## 7 SET2 Larval 0.426
## 8 SET2 Increasing 0.667
## 9 SET2 L3_High 0.183
## 10 SET2 Not_Changing 0.0120
## 11 SET3 Embryo_Specific 0
## 12 SET3 Larval 0.505
## 13 SET3 Increasing 0.811
## 14 SET3 L3_High 0.0765
## 15 SET3 Not_Changing 0.00510
## 16 SET4 Embryo_Specific 0.0263
## 17 SET4 Larval 0.298
## 18 SET4 Increasing 0.724
## 19 SET4 L3_High 0.285
## 20 SET4 Not_Changing 0.0526
```

Calculate the binomial pvalue and confidence intervals.

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

# Use binom.test to calculate pvalue and confidence intervals for the percentage of ELT2 binding clusters
bound_only_expression_binding_stats <-
  bound_only_expression_binding_stats %>%
  group_by(ELT2_cluster, set) %>%
  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]
  )

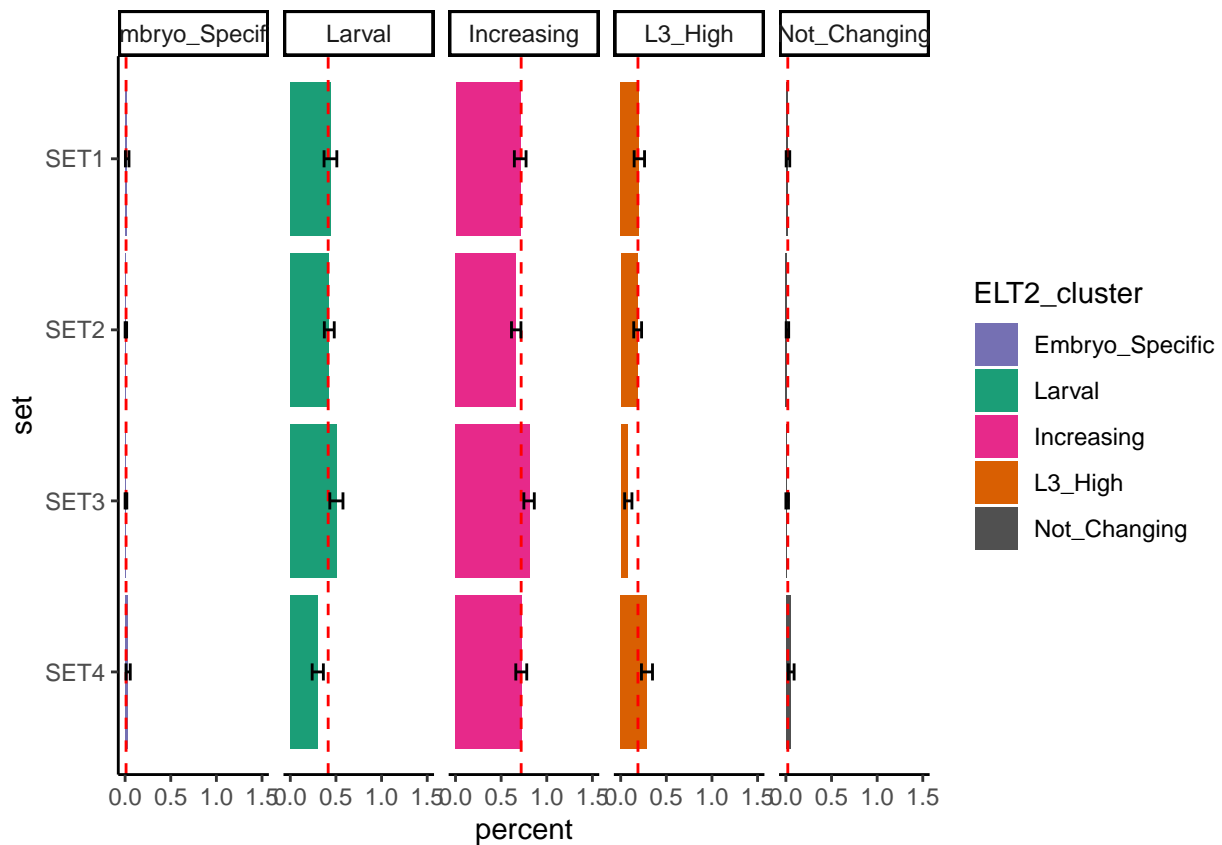
bound_only_expression_binding_stats$set <-
  factor(bound_only_expression_binding_stats$set,
    levels = c("SET4", "SET3", "SET2", "SET1"))

bound_only_expression_binding_stats %>% head()
```

```
## # A tibble: 6 x 9
## # Groups:   ELT2_cluster, set [6]
##   set   absent present ELT2_cluster percent background_perc~   pval conf.upper
##   <fct> <int>   <int> <fct>         <dbl>         <dbl> <dbl>   <dbl>
## 1 SET1     202       3 Embryo_Spec~ 0.0146         0.0104 0.476   0.0422
## 2 SET2     332       1 Embryo_Spec~ 0.00300        0.0104 0.275   0.0166
## 3 SET3     196       0 Embryo_Spec~ 0          0.0104 0.278   0.0186
## 4 SET4     222       6 Embryo_Spec~ 0.0263         0.0104 0.0331  0.0564
## 5 SET1     115     90 Larval      0.439          0.415 0.479   0.510
## 6 SET2     191    142 Larval      0.426          0.415 0.697   0.482
## # ... with 1 more variable: conf.lower <dbl>
```

```
bound_percent_multi <- ggplot(bound_only_expression_binding_stats,
  aes(x = set,
    y = percent, fill = ELT2_cluster)) +
  geom_bar(stat = "identity") +
  scale_y_continuous(limits = c(0, 1.5)) +
  theme_classic() +
  geom_hline(
    data = bound_only_expression_binding_stats %>% ungroup() %>% select(ELT2_cluster, background_percent),
    color = "red",
    linetype = "dashed",
    aes(yintercept = background_percent)
  ) +
  geom_errorbar(
    ymax = bound_only_expression_binding_stats$conf.upper,
    ymin = bound_only_expression_binding_stats$conf.lower,
    width = 0.1
  ) +
  coord_flip() +
  facet_grid(. ~ ELT2_cluster) +
  scale_fill_manual(values = as.character(cluster_colors$val))

bound_percent_multi
```



```
if (plot == TRUE){
  myggsave(plot = bound_percent_multi,
    name = "11_Bound_Only_Percent_of_ELT2bindClust_per_ExpressionClust",
    height = 5,
    width = 8,
    plotdir = plotdir)
}
```

Make a TF subset heatmap

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

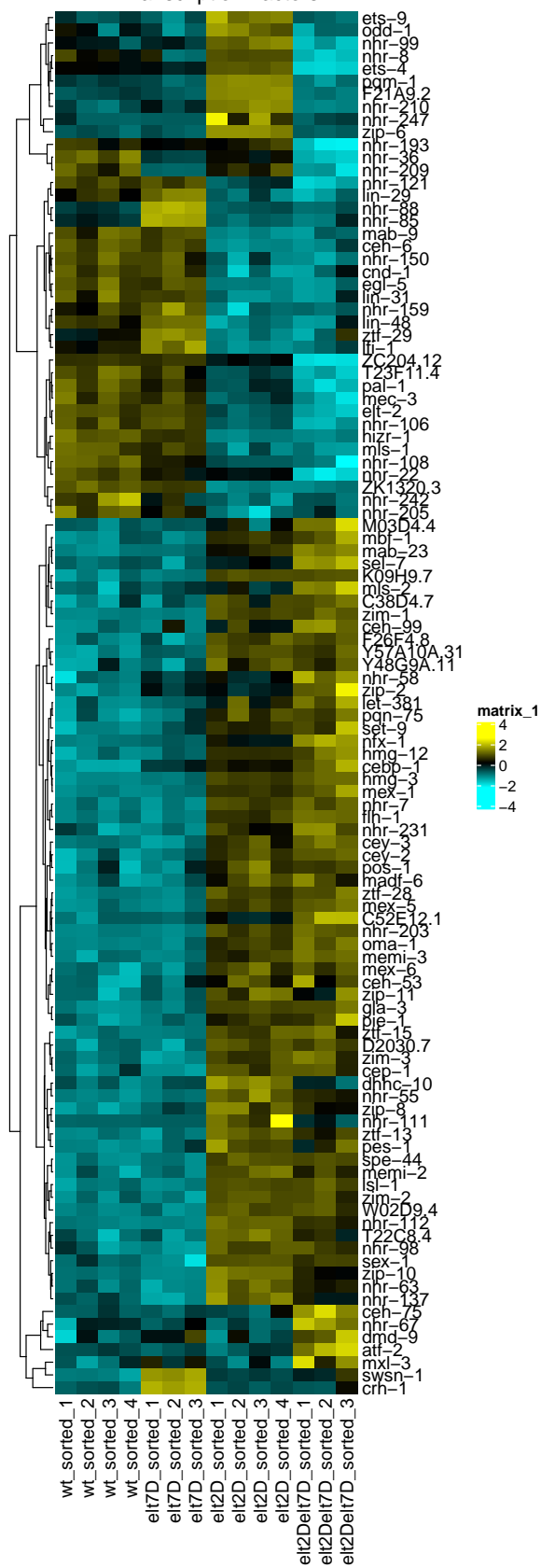
dynamic_counts_matrix_scaled_TFs <-
  matrix_select(dynamic_counts_matrix_scaled_ascend, wTF3.0$WBGeneID)

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",
```

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

Differential Expression of All Transcription Factors



```

if (plot == TRUE){
  myPDFplot(
    plot = tf_heatmap,
    name = "12_Differential_Expression_of_All_TFs",
    height = 20,
    width = 4,
    plotdir = plotdir
  )
}

```

```

## pdf
## 2

```

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

```

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

```

```

##          WBGeneID
## 1 WBGene00022562
## 2 WBGene00003621
## 3 WBGene00004096
## 4 WBGene00019327
## 5 WBGene00003711
## 6 WBGene00000793
## 7 WBGene00021082
## 8 WBGene00003607
## 9 WBGene00019743
## 10 WBGene00003689
## 11 WBGene00003648
## 12 WBGene00012101
## 13 WBGene00014193
## 14 WBGene00003698
## 15 WBGene00010215
## 16 WBGene00016997
## 17 WBGene00018704
## 18 WBGene00016865
## 19 WBGene00019344
## 20 WBGene00017687
## 21 WBGene00003727
## 22 WBGene00013976
## 23 WBGene00003511
## 24 WBGene00017651
## 25 WBGene00003106
## 26 WBGene00003678
## 27 WBGene00016888
## 28 WBGene00003845

```

```

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

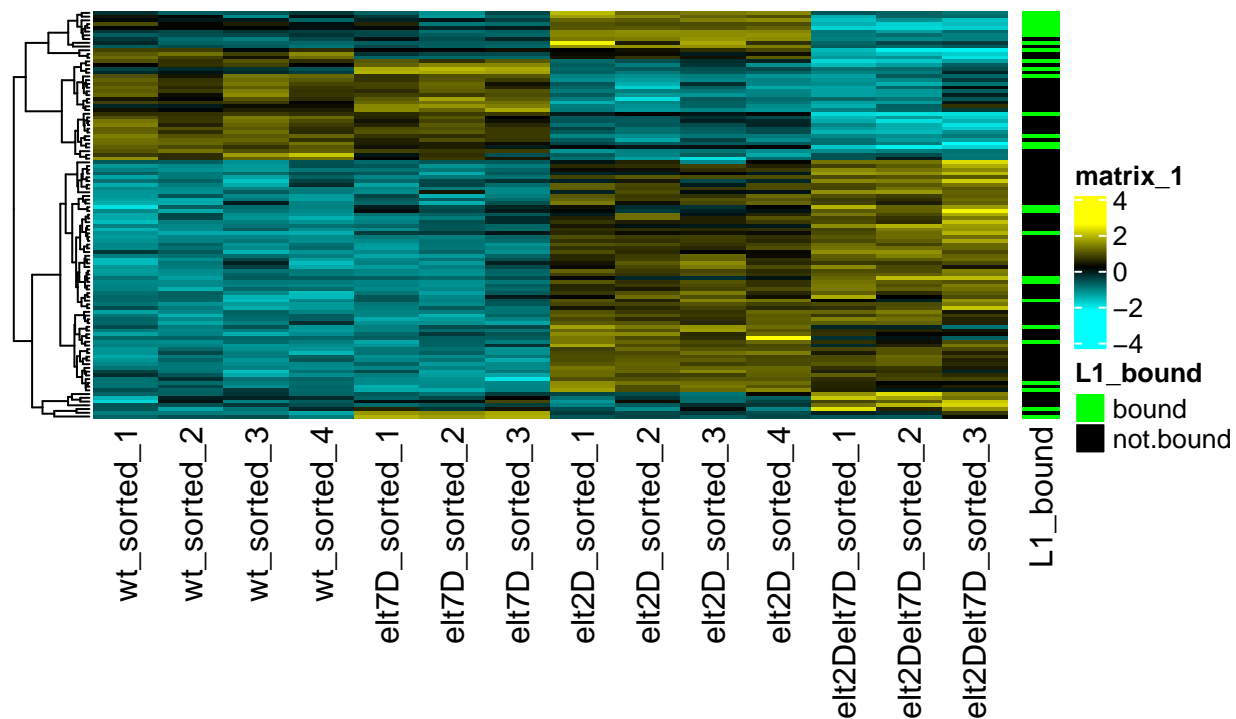
```



```
)

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
```

Differential Expression of All Transcription Factors



```
# pdf("./03_plots/13a_Differential_Expression_of_All_TFs_L1elt2bound_anno.pdf", height = 5, width = 5.5)
# tf_heatmap_L1bound
# dev.off()
```

```
if (plot == TRUE){
  myPDFplot(
    plot = tf_heatmap_L1bound,
    name = "13a_Differential_Expression_of_All_TFs_L1elt2bound_anno",
    height = 5,
    width = 5.5,
    plotdir = plotdir
  )
}
```

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

Add row annotation of intestine expression from Spencer intestine RNA data

```

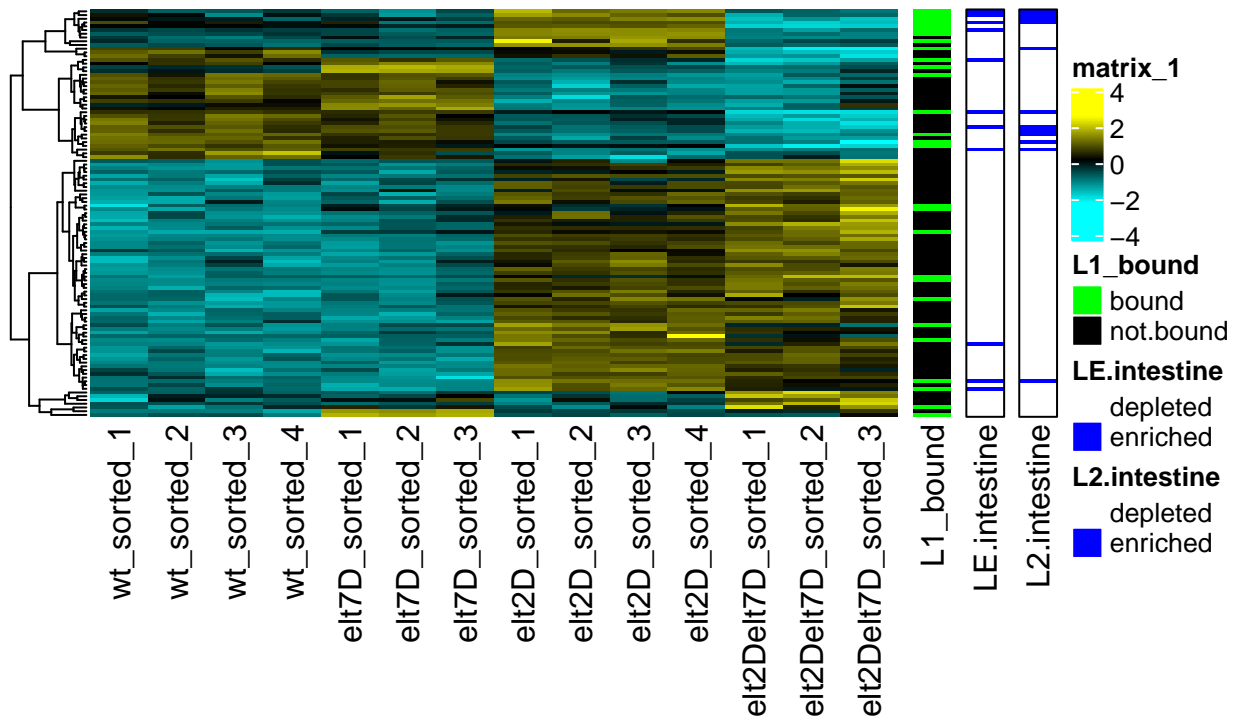
tf_spencer_rna_anno <- data.frame(
  spencerLE = ifelse(
    test = rownames(dynamic_counts_matrix_scaled_TFs) %in% spencer_LE_subset$spencer_LE_ID,
    yes = "enriched",
    no = "depleted"
  ),
  spencerL2 = ifelse(
    test = rownames(dynamic_counts_matrix_scaled_TFs) %in% spencer_L2_subset$spencer_L2_ID,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

tf_heatmap_L1bound_spencerRNA <- tf_heatmap_L1bound + rowAnnotation(
  LE.intestine = tf_spencer_rna_anno$spencerLE,
  col = list(LE.intestine = c(
    "enriched" = "blue", "depleted" = "white"
  )),
  border = TRUE
) +
rowAnnotation(
  L2.intestine = tf_spencer_rna_anno$spencerL2,
  col = list(L2.intestine = c(
    "enriched" = "blue", "depleted" = "white"
  )),
  border = TRUE
)

tf_heatmap_L1bound_spencerRNA

```

Differential Expression of All Transcription Factors

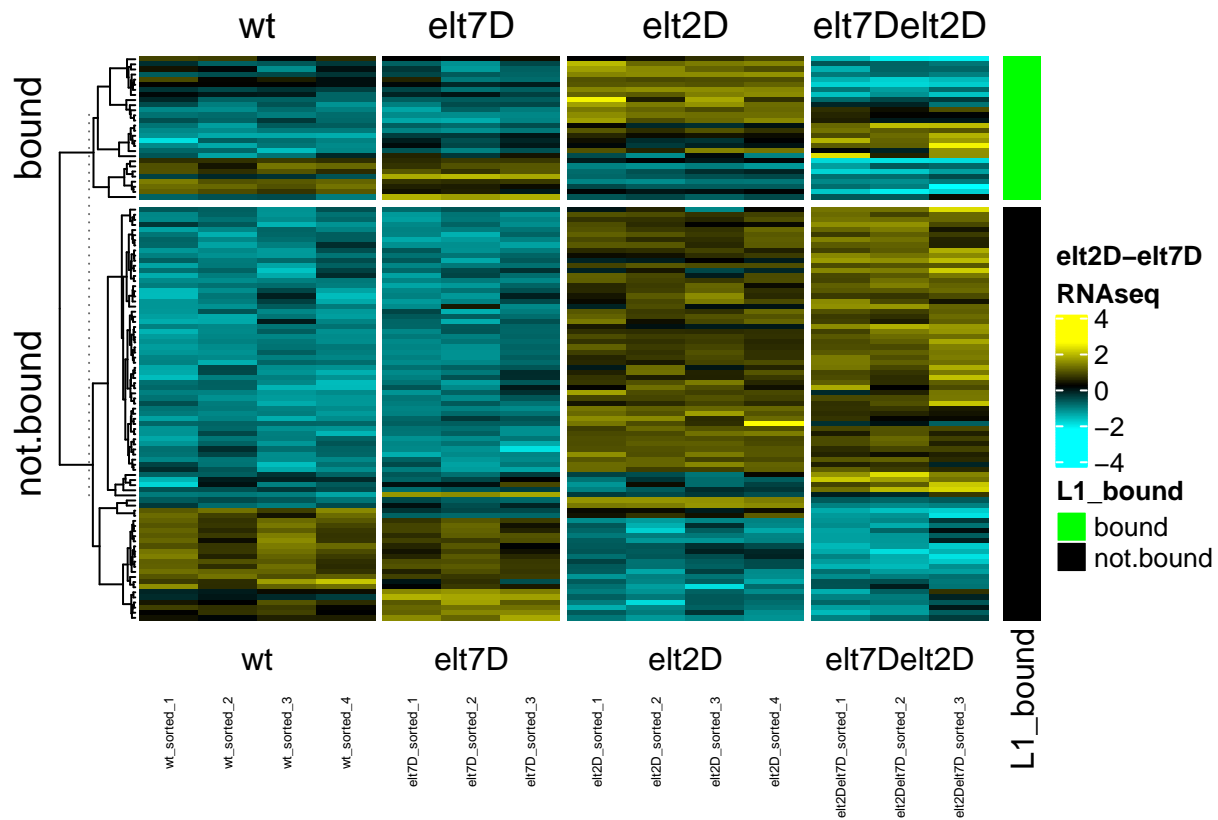


```
if (plot == TRUE){
  myPDFplot(
    plot = tf_heatmap_L1bound_spencerRNA,
    name = "13b_Differential_Expression_of_All_TFs_L1elt2bound_anno",
    height = 5,
    width = 5.5,
    plotdir = plotdir
  )
}
```

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

Split heatmap based on L1 binding

```
tf_heatmap_L1bound_split <- RNA_heatmap(dynamic_counts_matrix_scaled_TFs_names,
  split = tf_bound_anno$elt2_detected_in_L1) +
  rowAnnotation(L1_bound = tf_bound_anno$elt2_detected_in_L1,
    col = list(L1_bound = c(
      "bound" = "green", "not.bound" = "black"
    )))
tf_heatmap_L1bound_split
```



```
if (plot == TRUE){
  myPDFplot(
    plot = tf_heatmap_L1bound_split,
    name = "14a_Differential_Expression_of_All_TFs_L1elt2bound_split",
    height = 5,
    width = 5.5,
    plotdir = plotdir
  )
}
```

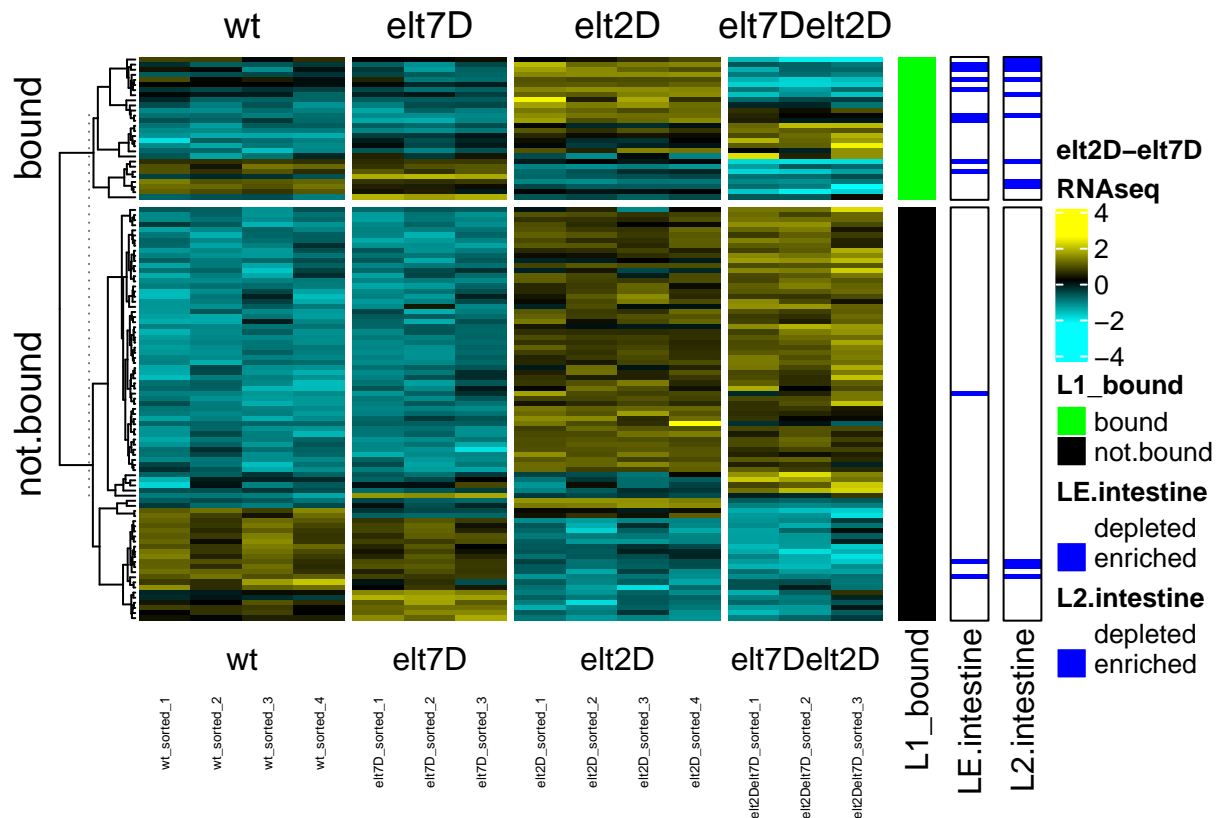
```
## pdf
## 2
```

Add row annotation of intestine expression from Spencer intestine RNA data to split heatmap

```
tf_heatmap_L1bound_split_spencerRNA <- tf_heatmap_L1bound_split +
  rowAnnotation(
    LE.intestine = tf_spencer_rna_anno$spencerLE,
    col = list(LE.intestine = c(
      "enriched" = "blue", "depleted" = "white"
    )),
    border = TRUE
  ) +
  rowAnnotation(
    L2.intestine = tf_spencer_rna_anno$spencerL2,
    col = list(L2.intestine = c(
      "enriched" = "blue", "depleted" = "white"
    )),
    border = TRUE
  )
```

```
)

tf_heatmap_L1bound_split_spencerRNA
```



```
if (plot == TRUE){
  myPDFplot(
    plot = tf_heatmap_L1bound_split_spencerRNA,
    name = "14b_Differential_Expression_of_All_TFs_L1elt2bound_split_spencerRNA",
    height = 5,
    width = 5.5,
    plotdir = plotdir
  )
}
```

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

Zoom in on only bound TFs

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

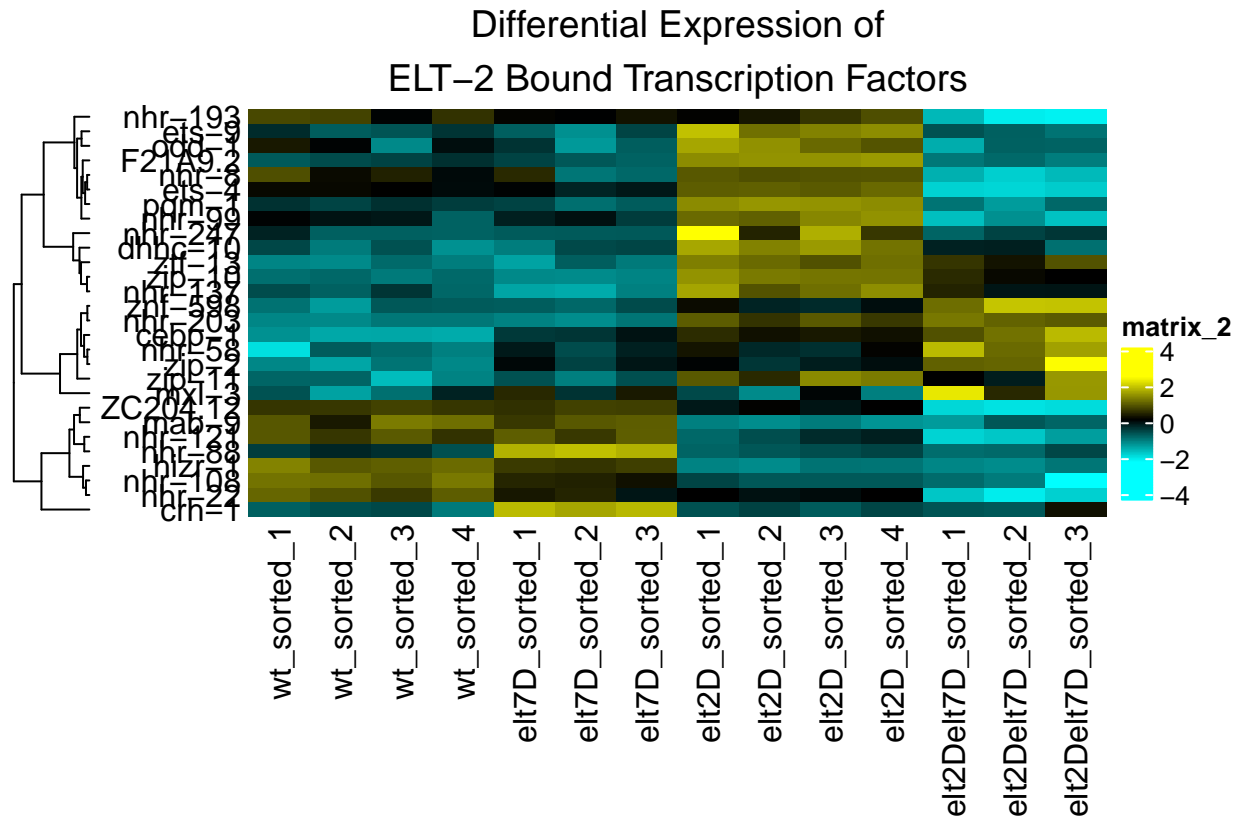
dynamic_counts_matrix_scaled_TFs_bound_names <-
  id2name(dynamic_counts_matrix_scaled_TFs_bound)

HAboundTF <- 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\nELT-2 Bound Transcription Factors"
)
HAboundTF

```



```

if (plot == TRUE){
  myPDFplot(
    plot = HAboundTF,
    name = "15a_Differential_Expression_Bound_TFs_only",
    height = 5,
    width = 5.5,
    plotdir = plotdir
  )
}

## pdf
## 2

tf_bound_spencer_rna_anno <- data.frame(
  spencerLE = ifelse(
    test = rownames(dynamic_counts_matrix_scaled_TFs_bound) %in% spencer_LE_subset$spencer_LE_ID,
    yes = "enriched",
    no = "depleted"
  )
)

```

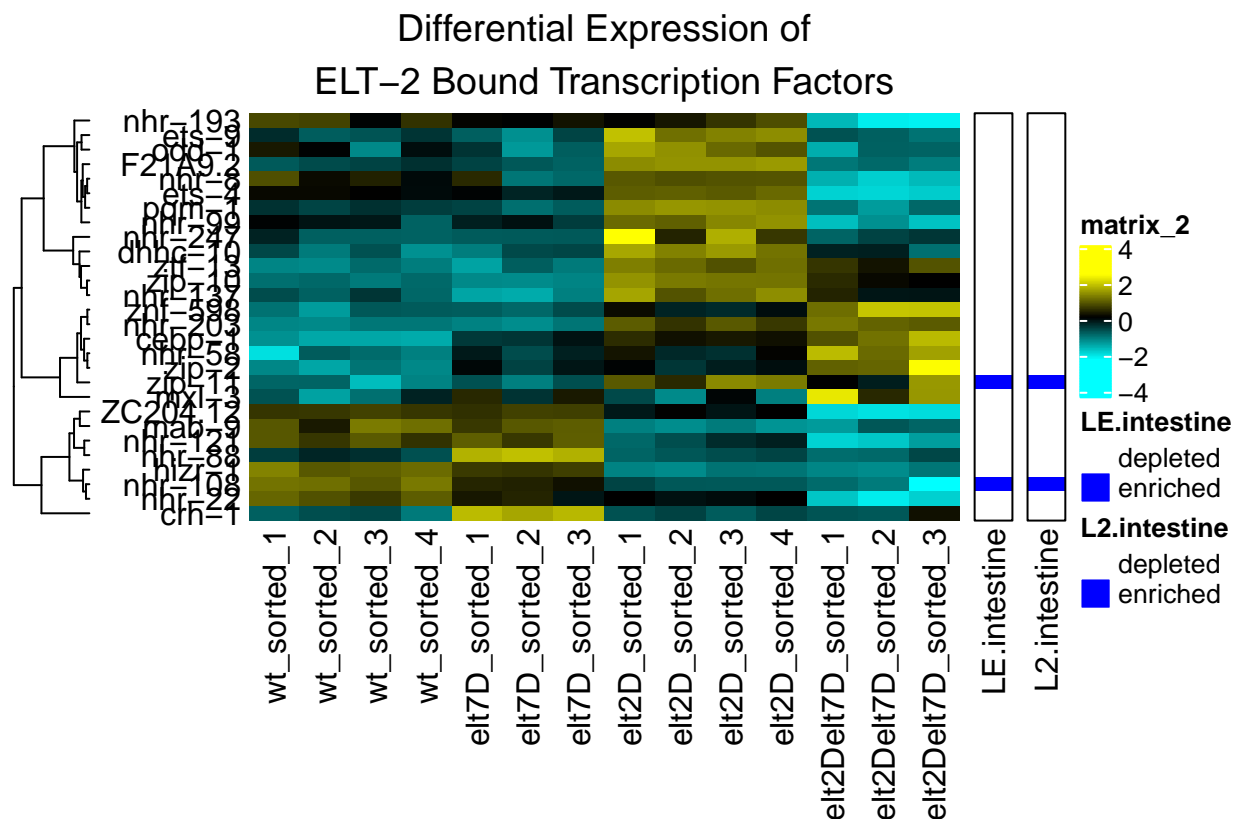
```

),
spencerL2 = ifelse(
  test = rownames(dynamic_counts_matrix_scaled_TFs_bound) %in% spencer_L2_subset$spencer_L2_ID,
  yes = "enriched",
  no = "depleted"
)
)

HAboundTF_spencerRNA <- HAboundTF + rowAnnotation(
  LE.intestine = tf_spencer_rna_anno$spencerLE,
  col = list(LE.intestine = c(
    "enriched" = "blue", "depleted" = "white"
  )),
  border = TRUE
) +
rowAnnotation(
  L2.intestine = tf_spencer_rna_anno$spencerL2,
  col = list(L2.intestine = c(
    "enriched" = "blue", "depleted" = "white"
  )),
  border = TRUE
)

HAboundTF_spencerRNA

```



```

if (plot == TRUE){
  myPDFplot(
    plot = HAboundTF_spencerRNA,

```

```

name = "15b_Differential_Expression_Bound_TFs_only_spencerRNA",
height = 5,
width = 5.5,
plotdir = plotdir
)
}

```

```

## pdf
## 2

```

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

TFs to follow up: pqm-1 (intestine), 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

Scratch

```

resWtV2 <- read_excel("./01_input/Table_S3_Pairwise_Comparison.xlsx",
  sheet = "2017-07-12_resWtV2", skip = 2)

```

```

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i = sheet, :
## Expecting numeric in G16065 / R16065C7: got 'NA'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i = sheet, :
## Expecting numeric in G16066 / R16066C7: got 'NA'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i = sheet, :
## Expecting numeric in G16067 / R16067C7: got 'NA'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i = sheet, :
## Expecting numeric in G16068 / R16068C7: got 'NA'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i = sheet, :
## Expecting numeric in G16069 / R16069C7: got 'NA'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i = sheet, :
## Expecting numeric in G16070 / R16070C7: got 'NA'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i = sheet, :
## Expecting numeric in G16071 / R16071C7: got 'NA'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i = sheet, :
## Expecting numeric in G16072 / R16072C7: got 'NA'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i = sheet, :
## Expecting numeric in G16073 / R16073C7: got 'NA'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i = sheet, :
## Expecting numeric in G16074 / R16074C7: got 'NA'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i = sheet, :
## Expecting numeric in G16075 / R16075C7: got 'NA'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i = sheet, :
## Expecting numeric in G16076 / R16076C7: got 'NA'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i = sheet, :
## Expecting numeric in G16077 / R16077C7: got 'NA'

```


[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

```
## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i = sheet, :
## Expecting numeric in G16708 / R16708C7: got 'NA'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i = sheet, :
## Expecting numeric in G16709 / R16709C7: got 'NA'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i = sheet, :
## Expecting numeric in G16710 / R16710C7: got 'NA'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i = sheet, :
## Expecting numeric in G16711 / R16711C7: got 'NA'

## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i = sheet, :
## Expecting numeric in G16712 / R16712C7: got 'NA'
```

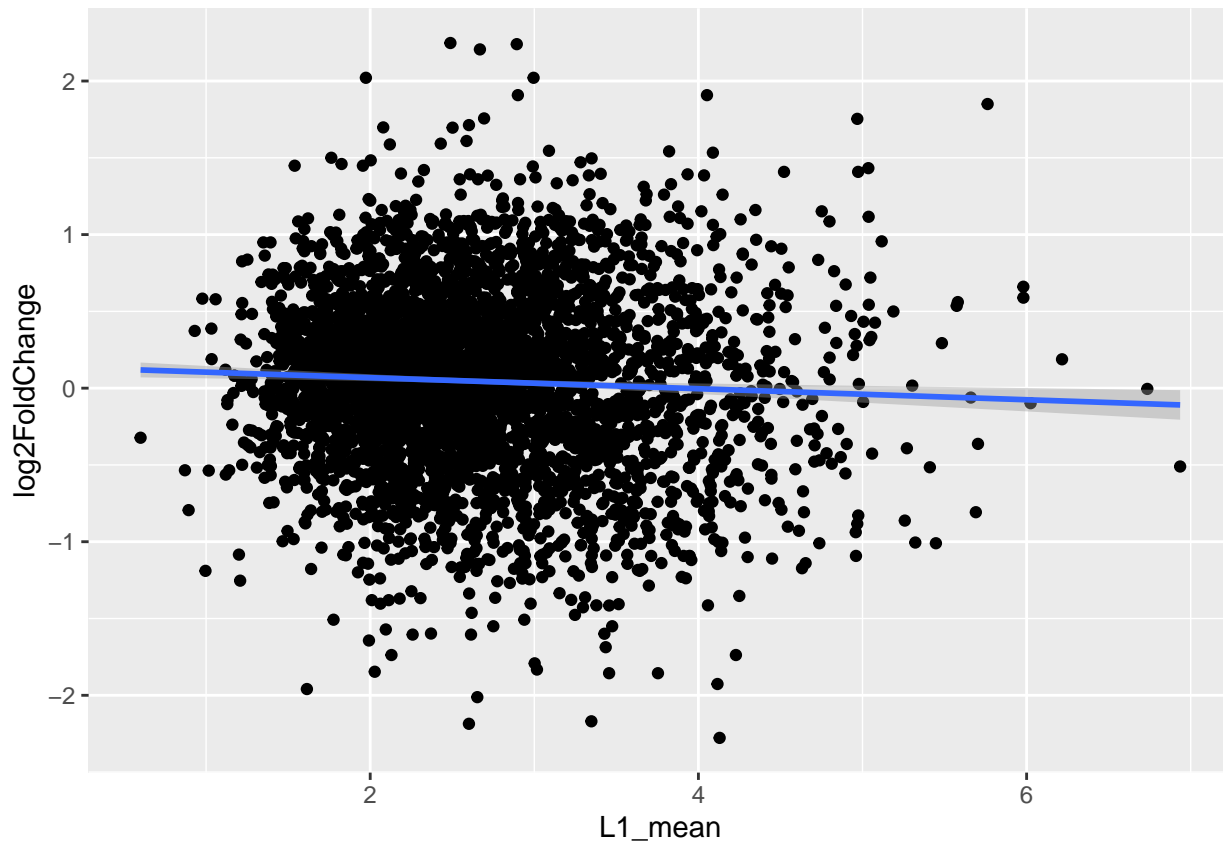
```
resWtV2
```

```
## # A tibble: 16,708 x 7
```

```
##   WBGeneID      baseMean log2FoldChange  lfcSE  stat    pvalue    padj
##   <chr>         <dbl>         <dbl>  <dbl> <dbl>    <dbl>    <dbl>
## 1 WBGene00007894    903.           6.98 0.182   34.0 3.42e-253 5.50e-249
## 2 WBGene00000783   1784.           3.80 0.0933  32.1 1.36e-226 1.09e-222
## 3 WBGene00004999   5635.          -4.70 0.122  -31.9 7.62e-224 4.08e-220
## 4 WBGene00019967   1581.           3.83 0.0950  31.9 2.33e-223 9.37e-220
## 5 WBGene00007331   2208.          -6.02 0.168  -31.1 2.72e-212 7.80e-209
## 6 WBGene00019495   2321.          -5.85 0.162  -31.1 2.91e-212 7.80e-209
## 7 WBGene00000789   2672.          -4.24 0.111  -31.0 5.66e-211 1.30e-207
## 8 WBGene00000786   1060.          -4.59 0.128  -29.5 1.36e-191 2.72e-188
## 9 WBGene00018547    508.           4.62 0.133   28.7 1.65e-181 2.94e-178
## 10 WBGene00018760    952.          -6.41 0.196  -28.7 9.04e-181 1.45e-177
## # ... with 16,698 more rows
```

```
ggplot(elt2_peaks %>% full_join(resWtV2, by = "WBGeneID") %>% filter(L1_IDR == 1, padj >= 0.05) %>% mutate(
  aes(x = L1_mean, y = log2FoldChange)
) +
  geom_point() +
  geom_smooth(method = lm)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



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] parallel stats4   grid      stats     graphics  grDevices utils
## [8] datasets  methods  base
##
## other attached packages:
## [1] GenomicRanges_1.38.0 GenomeInfoDb_1.22.1 IRanges_2.20.2
## [4] S4Vectors_0.24.4    BiocGenerics_0.32.0 lubridate_1.7.8
## [7] circlize_0.4.9      binom_1.1-1        dendextend_1.13.4
## [10] RVAideMemoire_0.9-75 pheatmap_1.0.12     matrixStats_0.56.0
## [13] ComplexHeatmap_2.2.0 readxl_1.3.1        forcats_0.5.0
## [16] stringr_1.4.0       dplyr_0.8.5         purrr_0.3.4
```



```

## [19] readr_1.3.1          tidyr_1.0.3          tibble_3.0.1
## [22] ggplot2_3.3.0        tidyverse_1.3.0      biomaRt_2.42.1
##
## loaded via a namespace (and not attached):
## [1] nlme_3.1-147          bitops_1.0-6          fs_1.4.1
## [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              mgcv_1.8-31
## [13] DBI_1.1.0             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         XVector_0.26.0
## [34] pkgconfig_2.0.3       htmltools_0.4.0       dbplyr_1.4.3
## [37] rlang_0.4.6           GlobalOptions_0.1.1   rstudioapi_0.11
## [40] RSQlite_2.2.0         farver_2.0.3          shape_1.4.4
## [43] generics_0.0.2        jsonlite_1.6.1        RCurl_1.98-1.2
## [46] magrittr_1.5          GenomeInfoDbData_1.2.2 Matrix_1.2-18
## [49] Rcpp_1.0.4.6          munsell_0.5.0         fansi_0.4.1
## [52] viridis_0.5.1         lifecycle_0.2.0       stringi_1.4.6
## [55] yaml_2.2.1            zlibbioc_1.32.0       BiocFileCache_1.10.2
## [58] blob_1.2.1            crayon_1.3.4          lattice_0.20-41
## [61] splines_3.6.3         haven_2.2.0           hms_0.5.3
## [64] knitr_1.28            pillar_1.4.4          rjson_0.2.20
## [67] reprex_0.3.0          XML_3.99-0.3          glue_1.4.0
## [70] evaluate_0.14         modelr_0.1.7          png_0.1-7
## [73] vctrs_0.2.4          cellranger_1.1.0      gtable_0.3.0
## [76] openssl_1.4.1         clue_0.3-57           assertthat_0.2.1
## [79] xfun_0.13            broom_0.5.6           viridisLite_0.3.0
## [82] AnnotationDbi_1.48.0  memoise_1.1.0         cluster_2.1.0
## [85] ellipsis_0.3.0

```