Step6 – DESeq2 – Clustering

Erin Osborne Nishimura

November 21, 2017

Date: November 21, 2017

Author: Erin Osborne Nishimura

Script: step6_171121_DESeq2_Clusteringanalysis.Rmd

Project: To analyze Aidan's RNA-seq datset from FACS sorted and filtered worms of the genotypes N2, elt-2D, elt-7D, and elt-2Delt-7D

Requires: + 3.3.2 + Bioconductor v. 3.2. for some reason 3.3 won't work. + RColorBrewer (1.1.2)
+ dplyr + GenomicRanges + gplots + ComplexHeatmap + circlize

Load required libraries:

- DESeq2

- RColorBrewer

- dplyr
- GenomicRanges
- gplots
- ComplexHeatmap

```
# #First time loading of DESeq2:
# source("http://bioconductor.org/biocLite.R")
# biocValid()
# biocLite("BiocUpgrade")
# biocLite("dplyr")
# biocLite("ComplexHeatmap")


library(ComplexHeatmap)
library(RColorBrewer)
library(dplyr)
library(GenomicRanges)
library(circlize)
```

Upload datasets from previous steps

I'll upload: + The list of changing genes + The annotated list of rld log counts

Get the annotated list of r-stabilized log transformed counts

```
# Get r-stabilized log transformed counts
projectdir <- "~/Dropbox/labwork/2016_ELT2_PROJECT/07_DESeq2_Analysis_EOP215"
```

```
setwd(paste(projectdir, "03_output", "step5_datasets", sep = "/"))
rlog_annot <- read.table(file="2017-07-12_step5_rlog_counts.txt", sep = "\t", na.strings = c(""), heade
dim(rlog_annot)
```

```
## [1] 16708    33
```

```
# Get list of changing genes
setwd(paste(projectdir, "03_output", "step5_datasets", sep = "/"))
changing_genes <- read.table(file = "2017-07-12_all_changing_genes_0.1alpha_0.8lfc.txt", header = FALSE
dim(changing_genes)
```

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

Incorporate in the Mann & Weisenfahrt ChIP-seq data

Produce bed files.

```
rlog_annot_bed <- rlog_annot[,c("chr_ce10", "start_min_ce10", "stop_max_ce10", "Row.names", "strand_ce1
names(rlog_annot_bed) <- c("chr", "start", "end", "id", "strand")
dim(rlog_annot_bed)
```

```
## [1] 16708     5
```

```
#Remove a few entries that have "+,-" strands
rlog_annot_bed <- rlog_annot_bed[setdiff(1:dim(rlog_annot_bed)[1], grep(",", rlog_annot_bed$strand )),]
rlog_annot_bed <- rlog_annot_bed[setdiff(1:dim(rlog_annot_bed)[1], grep(",", rlog_annot_bed$chr )),]

#Remove entries that have no start or end
rlog_annot_bed <- rlog_annot_bed[setdiff(1:dim(rlog_annot_bed)[1], grep("NA", rlog_annot_bed$end )),]
rlog_annot_bed <- rlog_annot_bed[setdiff(1:dim(rlog_annot_bed)[1], grep("NA", rlog_annot_bed$start )),]
dim(rlog_annot_bed)
```

```
## [1] 15897     5
```

```
#Change character and numeric classes
rlog_annot_bed$strand <- as.character(rlog_annot_bed$strand)
rlog_annot_bed$chr <- as.character(rlog_annot_bed$chr)
rlog_annot_bed$end <- as.numeric(as.character(rlog_annot_bed$end))
rlog_annot_bed$start <- as.numeric(as.character(rlog_annot_bed$start))

#Make Granges
norm_counts_annot_gr <- with(rlog_annot_bed, GRanges(chr, IRanges(start+1, end), strand=strand, id=id))

length(norm_counts_annot_gr)
```

```
## [1] 15897
```

```
head(norm_counts_annot_gr)
```

```
## GRanges object with 6 ranges and 1 metadata column:
##       seqnames               ranges strand |             id
##          <Rle>            <IRanges>  <Rle> |       <factor>
##   [1]     chrI [ 5107848,  5109950]      + | WBGene00000001
##   [2]    chrIV [ 9599418,  9601677]      - | WBGene00000002
##   [3]     chrV [ 9244658,  9246329]      - | WBGene00000003
##   [4]     chrX [ 2554232,  2557468]      + | WBGene00000004
##   [5]    chrIV [ 6272734,  6275702]      - | WBGene00000005
##   [6]     chrV [11466953, 11469146]      - | WBGene00000007
##   -------
##   seqinfo: 6 sequences from an unspecified genome; no seqlengths
```

Import ChIP-seq dataset from Weisenfahrt et al.

This information was downloaded from the paper: The function and regulation of the GATA factor ELT-2 in the C. elegans endoderm
Tobias Wiesenfahrt, Janette Y. Berg, Erin Osborne Nishimura, Adam G. Robinson, Barbara Goszczynski, Jason D. Lieb, James D. McGhee
http://dev.biologists.org/content/143/3/483.long

Supplemental datasets: 14_ELT2_IggBlackHOTMinus_gt30_summits.bed.

They will be imported into a GRagnes Object called peaks_bed:

```
#Import the ChIP-seq peaks from Weisenfahrt et al., 2016
setwd(paste(projectdir, "01_input", "02_from_weisenfahrt", sep = "/"))
peaks_bed <- read.table(file = "14_ELT2_IggBlackHOTMinus_gt30_summits.bed", sep = "\t", header = FALSE)
#Format the weisenfahrt peaks into a peaks_bed object that is GRanges compatible
colnames(peaks_bed) <- c("chr", "start", "end", "peakname", "MACS2_score")
peaks_bed <- with(peaks_bed, GRanges(chr, IRanges(start+1, end), id=peakname, score=MACS2_score))
head(peaks_bed)
```

```
## GRanges object with 6 ranges and 2 metadata columns:
##       seqnames               ranges strand |             id     score
##          <Rle>            <IRanges>  <Rle> |       <factor> <numeric>
##   [1]     chrI [   11374,    11374]      * |     MACS_peak_1     37.99
##   [2]     chrV [17616979, 17616979]      * | MACS_peak_10025     38.70
##   [3]     chrV [17659406, 17659406]      * | MACS_peak_10026     56.43
##   [4]     chrV [17662403, 17662403]      * | MACS_peak_10029     58.05
##   [5]     chrV [17889083, 17889083]      * | MACS_peak_10082     39.64
##   [6]     chrI [ 7287888,  7287888]      * |  MACS_peak_1010     51.89
##   -------
##   seqinfo: 6 sequences from an unspecified genome; no seqlengths
```

There are 624 peaks in the Weisenfahrt et al. dataset.

Import ChIP-seq dataset from Mann et al.

This dataset is ELT-2 ChIP-seq data from the paper:
Deactivation of the GATA Transcription Factor ELT-2 Is a Major Driver of Normal Aging in C. elegans
Frederick G. Mann, Eric L. Van Nostrand, Ari E. Friedland, Xiao Liu, Stuart K. Kim
http://journals.plos.org/plosgenetics/article?id=10.1371%2Fjournal.pgen.1005956

Supplemental Dataset:
S1 Table. Low-complexity ELT-2 ChIP-seq targets.
doi:10.1371/journal.pgen.1005956.s011 (XLSX)
mann_etal_journal.pgen.1005956.s011.txt

```
#Import the ChIP-seq peaks from Mann et al., 2016
setwd(paste(projectdir, "01_input", "03_from_mann", sep = "/"))
```

```
mannPeaks <- read.table(file = "mann_etal_journal.pgen.1005956.s011.txt", sep = "\t", header = TRUE, ski

#Format Mann peaks into a GRanges compatible object called mannPeaks_bed
mannPeaks <- mutate(mannPeaks, chr = sapply(strsplit(mannPeaks$Genome.Position, ":"), "[", 1),
       range = sapply(strsplit(mannPeaks$Genome.Position, ":"), "[", 2),
       score = -log(q.value, base = 10))
```

```
## Warning: package 'bindrcpp' was built under R version 3.2.5
```

```
mannPeaks <- mutate(mannPeaks,
       start = sapply(strsplit(mannPeaks$range, "-"), "[", 1),
       stop = sapply(strsplit(mannPeaks$range, "-"), "[", 2)
       )

mannPeaks$chr <- paste("chr", mannPeaks$chr, sep = "")

#Convert mannPeaks into a bed and Granges object:
mannPeaks_bed <- mannPeaks[,c(5,8,9,1,7,4)]
colnames(mannPeaks_bed) <- c("chr", "start", "end", "id", "score", "npeaks")
mannPeaks_bed$start <- as.integer(mannPeaks_bed$start)
mannPeaks_bed$end <- as.integer(mannPeaks_bed$end)
mannPeaks_bed <- with(mannPeaks_bed, GRanges(chr, IRanges(start+1, end), id=id, score=score, npeaks=npea
```

There are 2484 peaks in the Mann et al. dataset.

Use GRanges to find overlaps between the promoters of annotated genes and the peaks in both the Weisenfahrt et al. and Mann et al. datasets.

Look for peaks that fall within 3000 bp upstream of the transcription start site and within 1000 bp downstream of the transcription start site.

Let's do the Weisenfahrt dataset first:

```
#Line up norm_counts_annot_bed with peaks and mann_peaks

#Find overlaps
#Make promoter regions; find overlaps between promoters and peaks
all_promoters <- promoters(norm_counts_annot_gr, upstream=3000, downstream=1000, use.names=TRUE)
#The above code didn't work on a more recent installation. try again
all_promoters <- promoters(norm_counts_annot_gr, upstream=3000, downstream=1000)

hits <- findOverlaps(all_promoters, peaks_bed)
head(hits)
```

```
## Hits object with 6 hits and 0 metadata columns:
##       queryHits subjectHits
##       <integer>   <integer>
##   [1]         6         570
##   [2]        23         466
##   [3]        47         566
##   [4]        49         375
##   [5]        49         376
##   [6]       100          57
##   -------
##   queryLength: 15897
##   subjectLength: 624
```

```
#length(peaks_bed)
#length(all_promoters)
#str(hits)
length(queryHits(hits))
```

```
## [1] 505
```

There are 505 of 624 ELT-2 ChIP-seq peaks (from Weisenfahrt et al) fall within 3 kb upstream and 1 kb downstream of an annotated transcriptional start site.

Save this in an object called counts_annot_peaks1

```
#Pull out the genes that match with these peaks, preserve peak scores from Weisenfahrt et al.
match_peaks <- cbind.data.frame(query_id = as.vector(all_promoters$id[queryHits(hits)]), score = as.vec
weisenfahrt_match_peaks_annot <- cbind.data.frame(query_id = as.vector(all_promoters$id[queryHits(hits)
#Compress these down into individual genes (some genes may have multiple peaks in their promoters)
unique_match_peaks <- match_peaks %>%
  group_by(query_id) %>%
  summarize(score_sum = sum(score),
            peaks_num = n()) %>%
  arrange(desc(score_sum))

#How many do we have now
dim(unique_match_peaks)
```

```
## [1] 482   3
```

```
#head(unique_match_peaks)
#as.data.frame(unique_match_peaks)[1:100,]

#OK, now go back and re-merge this with the list of genes:
#head(rlog_annot)
#head(unique_match_peaks)
#dim(rlog_annot)
#dim(unique_match_peaks)
counts_annot_peaks1 <- merge(rlog_annot, unique_match_peaks, by.x = "Row.names", by.y = "query_id", all
colnames(counts_annot_peaks1)[which(colnames(counts_annot_peaks1) == "score_sum")] <- "ELT2chip_scoresu
colnames(counts_annot_peaks1)[which(colnames(counts_annot_peaks1) == "peaks_num")] <- "ELT2chip_peaksnu
dim(counts_annot_peaks1)
```

```
## [1] 16708    35
```

```
#head(counts_annot_peaks1)
```

OK, now do the same for the Mann et al peaks data:

There are 2772 of 2484 ELT-2 ChIP-seq peaks (from Mann et al) that fall within 3 kb upstream and 1 kb downstream of an annotated transcriptional start site.

Now I'll pull out the unique genes associated with these peaks. This will be saved as an object called counts_annot_peaks2

```
## [1] 2308    3
```

Merge counts_annot_peaks1 with the Mann et al datsets (unique_match_peaks2) to create the object: counts_annot_peaks2.

```
##        Row.names elt2D_sorted_1 elt2D_sorted_2 elt2D_sorted_3
## 1 WBGene00000001       9.172904       9.249496       9.211660
## 2 WBGene00000002       7.503760       7.289884       7.386127
## 3 WBGene00000003       8.669299       8.593847       8.753835
## 4 WBGene00000004      10.303062      10.296768      10.356820
## 5 WBGene00000005       2.953325       2.835451       2.886842
## 6 WBGene00000006       9.843262       9.870450      10.009631
##   elt2D_sorted_4 elt2Delt7D_sorted_1 elt2Delt7D_sorted_2
## 1       9.346959            9.379698            9.217403
## 2       7.262063            7.904008            7.870852
## 3       8.781267            8.791018            8.795191
## 4      10.366512           10.332489           10.223675
## 5       2.979650            2.499412            2.763405
## 6       9.808400            9.946104            9.855117
##   elt2Delt7D_sorted_3 wt_sorted_1 wt_sorted_2 wt_sorted_3 wt_sorted_4
## 1            9.101997    8.957161    8.858238    8.841623    8.923111
## 2            7.762023    7.489159    7.382905    7.518631    7.492399
## 3            8.936724    9.061810    8.748589    9.295497    9.286834
## 4           10.597407   10.916559   10.786200   11.010430   10.826657
## 5            2.428255    2.990777    2.864044    3.116144    2.715502
## 6            9.924009    9.650145    9.757415    9.632753    9.646993
##   elt7D_sorted_1 elt7D_sorted_2 elt7D_sorted_3 sequence_id_list
## 1       8.505028       8.568569       8.517438       Y110A7A.10
## 2       7.378168       7.582425       7.512668          F27C8.1
## 3       9.480361       9.451384       9.008938          F07C3.7
## 4      10.836827      10.806534      10.819497          F52H2.2
## 5       2.584081       2.881642       2.827526        T13A10.10
## 6       9.819234       9.750452       9.883120               NA
##   gene_id_val  transcripts chr_ce10 strand_ce10 start_min_ce10
## 1       aap-1    NM_059121     chrI           +        5107847
## 2       aat-1    NM_069306    chrIV           -        9599417
## 3       aat-2    NM_072993     chrV           -        9244657
## 4       aat-3    NM_076060     chrX           +        2554231
## 5       aat-4 NM_001028211    chrIV           -        6272733
## 6          NA           NA       NA          NA             NA
##   stop_max_ce10 chromosome_ce11 start_min_ce11 end_max_ce11 strand_ce11
## 1       5109950               I        5107844      5109947           +
## 2       9601677              IV        9599434      9601694           -
## 3       9246329               V        9244680      9246352           -
## 4       2557468               X        2554238      2557475           +
## 5       6275702              IV        6272744      6275713           -
## 6            NA              NA             NA           NA          NA
##   product_refseq symbol_refseq gene_name chr   start    stop strand
## 1    NM_059121.7         aap-1     aap-1   I 5110164 5107843      1
## 2    NM_069306.4         aat-1     aat-1  IV 9601695 9598977     -1
## 3    NM_072993.5         aat-2     aat-2   V 9246360 9244412     -1
## 4    NM_076060.4         aat-3     aat-3   X 2557725 2552436      1
## 5 NM_001028211.3         aat-4     aat-4  IV 6275713 6272591     -1
## 6             NA            NA        NA  NA      NA      NA     NA
##   ELT2chip_scoresum_Weisenfahrt ELT2chip_peaksnum_Weisenfahrt
## 1                            NA                            NA
## 2                            NA                            NA
## 3                            NA                            NA
## 4                            NA                            NA
## 5                            NA                            NA
## 6                            NA                            NA
##   ELT2chip_scoresum_Mann ELT2chip_peaksnum_Mann
## 1                     NA                     NA
## 2                     NA                     NA
## 3                     NA                     NA
## 4                     NA                     NA
## 5                     NA                     NA
## 6                     NA                     NA
```

```
## [1] 16708    37
```

Ok, now I have a large datastructure: counts_annot_peak2 with information on…
1) r-log transformed count data from the RNA-seq assay
2) annotation information for ce10 and ce11
3) ChIP-seq info from Weisenfahrt et al.
4) ChIP-seq info from Mann et al.

Import Intestine Specific and Intestine-enriched gene lists. Merge those into the dataset and incorporate the information into the clustering analysis:

```r
# Import the intestine enriched list
setwd(paste(projectdir, "01_input", "04_intestineSp", sep = "/"))
intestineEnriched <- read.table(file = "IEG_170104.txt", sep = "\t", header = FALSE)
dim(intestineEnriched)
```

```
## [1] 2202    1
```

```r
# Import the intestine specific list
setwd(paste(projectdir, "01_input", "04_intestineSp", sep = "/"))
intestineSpecific <- read.table(file = "ISG_170104.txt", sep = "\t", header = FALSE)
dim(intestineSpecific)
```

```
## [1] 137   1
```

```r
intestine_union <- union(intestineSpecific$V1, intestineEnriched$V1)

# Annotate whether a gene is intestine enriched or specific in rlog counts data frame:
all_annot_rlog_counts_peaks_int <- counts_annot_peaks2 %>%
        mutate(intestine_enr = (sequence_id_list %in% intestine_union)) %>%
        mutate(intestine_sp = (sequence_id_list %in% intestineSpecific$V1))

#dim(all_annot_rlog_counts_peaks_int)
#head(all_annot_rlog_counts_peaks_int)
#sum(all_annot_rlog_counts_peaks_int$intestine_enr)
#sum(all_annot_rlog_counts_peaks_int$intestine_sp)
```

Save the dataset

```
setwd(paste(projectdir, "03_output", "step6_cluster_analysis", sep = "/"))
filename = paste(Sys.Date(), "counts_annot_peaks_int.txt", sep = "_")
write.table(all_annot_rlog_counts_peaks_int, file = filename, sep = "\t")
colnames(all_annot_rlog_counts_peaks_int)
```

```
##  [1] "Row.names"                     "elt2D_sorted_1"
##  [3] "elt2D_sorted_2"                "elt2D_sorted_3"
##  [5] "elt2D_sorted_4"                "elt2Delt7D_sorted_1"
##  [7] "elt2Delt7D_sorted_2"           "elt2Delt7D_sorted_3"
##  [9] "wt_sorted_1"                   "wt_sorted_2"
## [11] "wt_sorted_3"                   "wt_sorted_4"
## [13] "elt7D_sorted_1"                "elt7D_sorted_2"
## [15] "elt7D_sorted_3"                "sequence_id_list"
## [17] "gene_id_val"                   "transcripts"
## [19] "chr_ce10"                      "strand_ce10"
## [21] "start_min_ce10"                "stop_max_ce10"
## [23] "chromosome_ce11"               "start_min_ce11"
## [25] "end_max_ce11"                  "strand_ce11"
## [27] "product_refseq"                "symbol_refseq"
## [29] "gene_name"                     "chr"
## [31] "start"                         "stop"
## [33] "strand"                        "ELT2chip_scoresum_Weisenfahrt"
## [35] "ELT2chip_peaksnum_Weisenfahrt" "ELT2chip_scoresum_Mann"
## [37] "ELT2chip_peaksnum_Mann"        "intestine_enr"
## [39] "intestine_sp"
```

```
colnames(all_annot_rlog_counts_peaks_int[1:100,c(1,16:39)])
```

```
##  [1] "Row.names"                     "sequence_id_list"
##  [3] "gene_id_val"                   "transcripts"
##  [5] "chr_ce10"                      "strand_ce10"
##  [7] "start_min_ce10"                "stop_max_ce10"
##  [9] "chromosome_ce11"               "start_min_ce11"
## [11] "end_max_ce11"                  "strand_ce11"
## [13] "product_refseq"                "symbol_refseq"
## [15] "gene_name"                     "chr"
## [17] "start"                         "stop"
## [19] "strand"                        "ELT2chip_scoresum_Weisenfahrt"
## [21] "ELT2chip_peaksnum_Weisenfahrt" "ELT2chip_scoresum_Mann"
## [23] "ELT2chip_peaksnum_Mann"        "intestine_enr"
## [25] "intestine_sp"
```

```
setwd(paste(projectdir, "03_output", "step6_cluster_analysis", sep = "/"))
filename = paste(Sys.Date(), "genes_ELT2peaks_intestineExpression.txt", sep = "_")

write.table(all_annot_rlog_counts_peaks_int[1:100,c(1,16:39)], file = filename, sep = "\t", quote = FALS
```

Import the list of changing genes and filter:

```
# Get list of changing genes
setwd(paste(projectdir, "03_output", "step5_datasets", sep = "/"))
changing_genes <- read.table(file = "2017-07-12_all_changing_genes_0.1alpha_0.8lfc.txt", header = FALSE
dim(changing_genes)
```

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

```
head(changing_genes)
```

```
##               V1
## 1 WBGene00004020
## 2 WBGene00015956
## 3 WBGene00000216
## 4 WBGene00001795
## 5 WBGene00008167
## 6 WBGene00010049
```

Merge the changing genes list with the list of rlog counts

```
changing_pairwise_rlog_counts <- all_annot_rlog_counts_peaks_int[all_annot_rlog_counts_peaks_int$Row.na
dim(changing_pairwise_rlog_counts)
```

```
## [1] 3092   39
```

```
# Changing NA's to 0's:
# Changing NA's to 0
# Changing Inf to MaxFinite
changing_pairwise_rlog_counts$ELT2chip_scoresum_Weisenfahrt[which(is.na(changing_pairwise_rlog_counts$E
changing_pairwise_rlog_counts$ELT2chip_scoresum_Mann[which(is.na(changing_pairwise_rlog_counts$ELT2chip
max_finite <- max(changing_pairwise_rlog_counts$ELT2chip_scoresum_Mann[is.finite(changing_pairwise_rlog
changing_pairwise_rlog_counts$ELT2chip_scoresum_Mann[which(changing_pairwise_rlog_counts$ELT2chip_scores
#head(changing_pairwise_rlog_counts)

# setwd(paste(projectdir, "03_output", "step5_datasets", sep = "/"))
# changing_genes <- read.table(file = "2017-07-12_all_changing_genes_0.1alpha_0.8lfc.txt", header = FAL
# dim(changing_genes)
# head(changing_genes)
```

Cluster with annotation:

```
# Set up a matrix for a clustered heatmap:
rld_pairwise_matrix <- as.matrix(changing_pairwise_rlog_counts[,c(2:15)])
rld_pairwise_matrix <- rld_pairwise_matrix[,c(8:14, 1:7)]
row.names(rld_pairwise_matrix) <- changing_pairwise_rlog_counts$gene_id_val
#head(rld_pairwise_matrix)
dim(rld_pairwise_matrix)
```

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

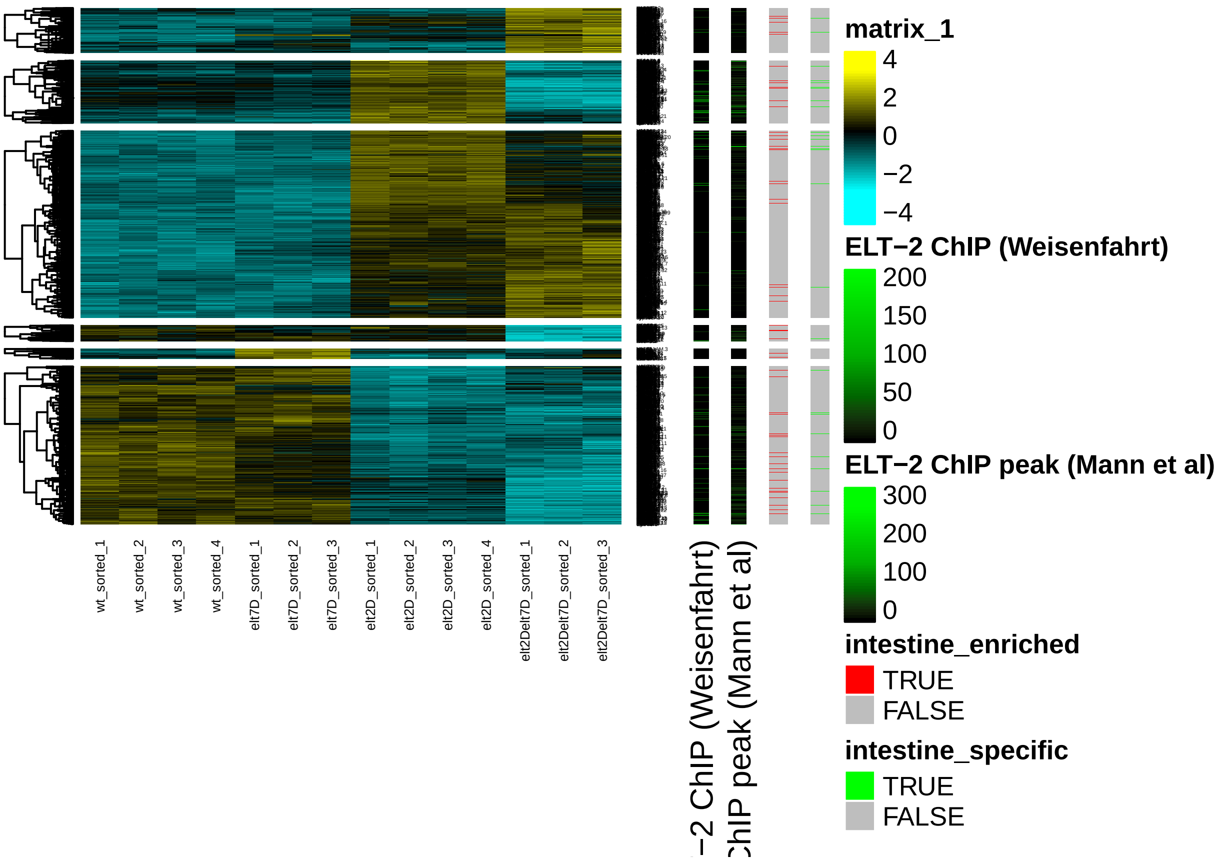
```
#scaling, no centering:
mat_scaled = t(apply(unlist(rld_pairwise_matrix), 1, scale))
```

```r
colnames(mat_scaled) <- colnames(rld_pairwise_matrix)
head(mat_scaled)
```

```
##       wt_sorted_1 wt_sorted_2 wt_sorted_3 wt_sorted_4 elt7D_sorted_1
## aat-6   1.0068329  1.37348252   1.0589277   1.4476397     0.84613352
## aat-7   2.2632093  1.13063525   1.1251278   1.0262925    -0.03607787
## aat-8   0.1468716 -0.09556483  -0.3465276  -0.8378633     0.07003147
## abf-2  -1.0765042  0.04628523  -1.0478603  -0.4296435    -0.61401384
## abf-5  -0.1629274  0.14035593  -0.8318355  -0.2209018    -0.52814604
## abf-6   0.1344074  0.43209491  -0.4453539   0.5202470    -0.19720767
##       elt7D_sorted_2 elt7D_sorted_3 elt2D_sorted_1 elt2D_sorted_2
## aat-6     0.51350637     0.07506888     -0.7898010     -0.6055647
## aat-7    -0.39030667     0.02722321     -0.4521136     -1.0292850
## aat-8    -0.11586861     0.42221560      0.8406016      1.2349599
## abf-2    -0.58009755    -0.38693983     -0.4767996      0.3851813
## abf-5    -0.50445577    -0.16186256     -0.5681545     -0.6137809
## abf-6     0.05519157     0.37152702     -0.9790560     -1.0378885
##       elt2D_sorted_3 elt2D_sorted_4 elt2Delt7D_sorted_1
## aat-6    -1.09248186     -0.9350192          -0.9202246
## aat-7    -0.46498937     -0.8771172          -0.9402531
## aat-8     0.98161197      1.7266509          -1.7004545
## abf-2     0.09286966     -0.5163112           2.5457794
## abf-5    -0.75209134     -1.0136068           1.7015008
## abf-6    -1.16996644     -1.7376299           1.4066491
##       elt2Delt7D_sorted_2 elt2Delt7D_sorted_3
## aat-6          -0.8564679          -1.1220323
## aat-7          -0.5550156          -0.8273297
## aat-8          -0.8668929          -1.4597714
## abf-2           1.4999051           0.5581492
## abf-5           2.1353949           1.3805110
## abf-6           1.6701858           0.9767996
```

Functionalize the clustering

```r
return_cluster <- function(dataframe, num){
        #dataframe <- changing_pairwise_rlog_counts_intSp
        # Convert dataframe to a matrix
        rld_pairwise_matrix <- as.matrix(dataframe[,c(2:15)])
        rld_pairwise_matrix <- rld_pairwise_matrix[,c(8:14, 1:7)]
        row.names(rld_pairwise_matrix) <- dataframe$gene_id_val

        # scaling, no centering:
        mat_scaled = t(apply(unlist(rld_pairwise_matrix), 1, scale))
        colnames(mat_scaled) <- colnames(rld_pairwise_matrix)

        # Draw main RNA-seq heatmap
        ht1 <- Heatmap(mat_scaled,
          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 = num)

        # Draw ChIP-seq Weisenfahrt heatmap
        maxWeis <- max(dataframe$ELT2chip_scoresum_Weisenfahrt)
        maxWeis1 <- maxWeis*0.125
        maxWeis2 <- maxWeis*0.25
        maxWeis3 <- maxWeis*0.5
        ht2 <- Heatmap(dataframe$ELT2chip_scoresum_Weisenfahrt,
                       name = "ELT-2 ChIP (Weisenfahrt)",
                       show_row_names = FALSE, width = unit(2, "mm"),
                       col = colorRamp2(c(0, maxWeis1, maxWeis2, maxWeis3),
                                        c("#000000", "#006400", "#00af00", "#00fb00")),
                       heatmap_legend_param = list(color_bar = "continuous"))

        # Draw ChIP-seq Mann heatmap
        maxMann <- max(dataframe$ELT2chip_scoresum_Mann)
        maxMann1 <- maxMann*0.125
        maxMann2 <- maxMann*0.25
        maxMann3 <- maxMann*0.5
        ht3 <- Heatmap(dataframe$ELT2chip_scoresum_Mann,
                       name = "ELT-2 ChIP peak (Mann et al)",
                       show_row_names = FALSE,
                       width = unit(2, "mm"),
                       col = colorRamp2(c(0, maxMann1, maxMann2, maxMann3),
                                      c("#000000", "#006400", "#00af00", "#00fb00")),
                       heatmap_legend_param = list(color_bar = "continuous"))

        # Annotate with intestine-enriched
        ha <- rowAnnotation(intestine_enriched = dataframe$intestine_enr,
                            col = list(intestine_enriched = c("TRUE" = "red", "FALSE" = "gray")),
                            gap = unit(0, "mm"),
                            width = unit(0.25, "cm"))

        # Annotate with intestine-specific
        ha2 <- rowAnnotation(intestine_specific = dataframe$intestine_sp,
                             col = list(intestine_specific = c("TRUE" = "green", "FALSE" = "gray")),
                             gap = unit(0, "mm"),
                             width = unit(0.25, "cm"))

        ht_list2 = ht1 + ht2 + ht3 + ha + ha2
        return(ht_list2)
}

# Generate the clustered heatmap for all genes
draw(return_cluster(changing_pairwise_rlog_counts, 6))
```



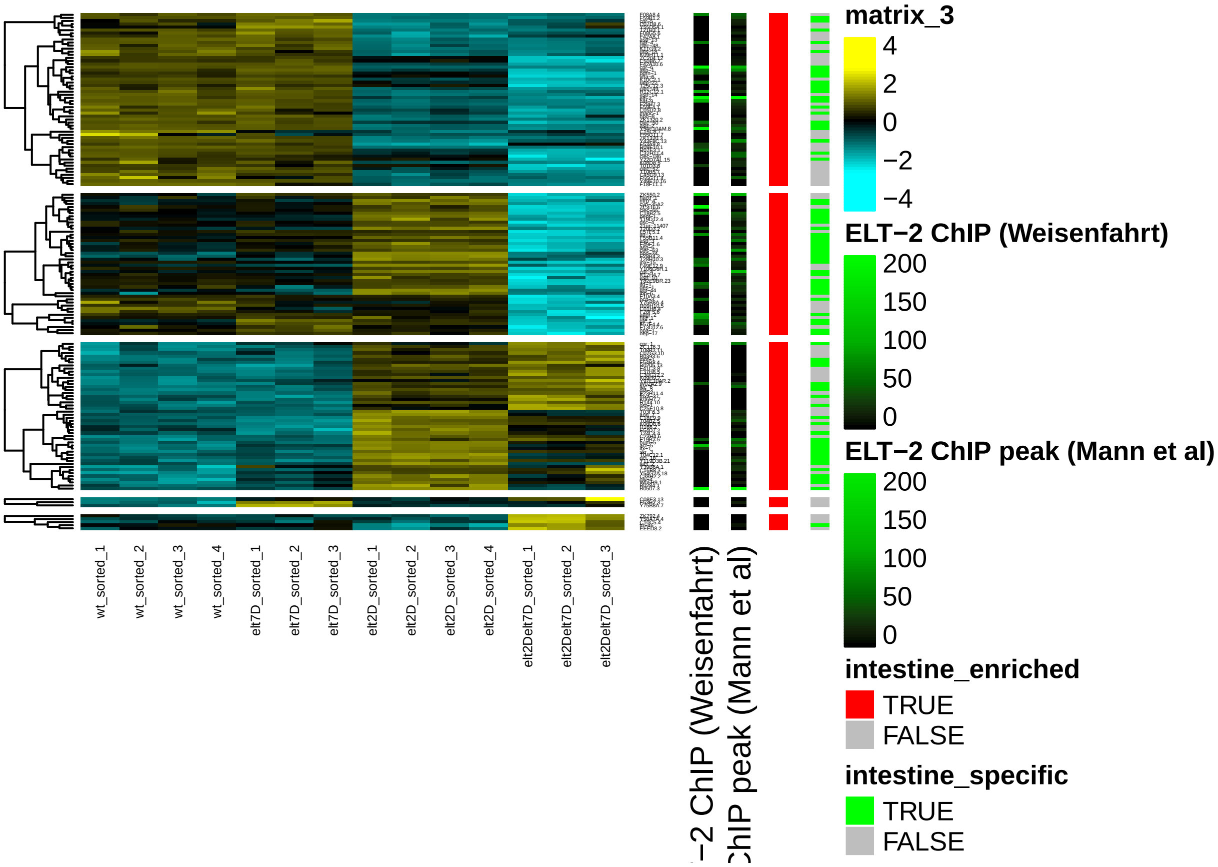
```
changing_cluster_list <- return_cluster(changing_pairwise_rlog_counts, 5)

# str(changing_cluster_list)
# as.vector(changing_pairwise_rlog_counts_intSp[row_order(ht1)[[1]],]$WBGene)
# row_order(changing_cluster_list)[[1]]

#Save it as a figure
setwd(paste(projectdir, "03_output", sep = "/"))
dir.create(paste(projectdir, "03_output", "step6_cluster_analysis", sep = "/"), showWarnings = FALSE)
setwd(paste(projectdir, "03_output", "step6_cluster_analysis", sep = "/"))
pdf(paste(Sys.Date(), "changing_genes_cluster_annotated_all.pdf", sep = "_" ), width=5, height=8)
draw(changing_cluster_list)
dev.off()
```

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

```
# Generate the clustered heatmap for intestine-enriched genes
changing_pairwise_rlog_counts_intEnr <- changing_pairwise_rlog_counts %>%
         filter(intestine_enr == TRUE)

dim(changing_pairwise_rlog_counts_intEnr)
```

```
## [1] 158  39
```

```
#head(changing_pairwise_rlog_counts_intEnr)

draw(return_cluster(changing_pairwise_rlog_counts_intEnr, 5))
```



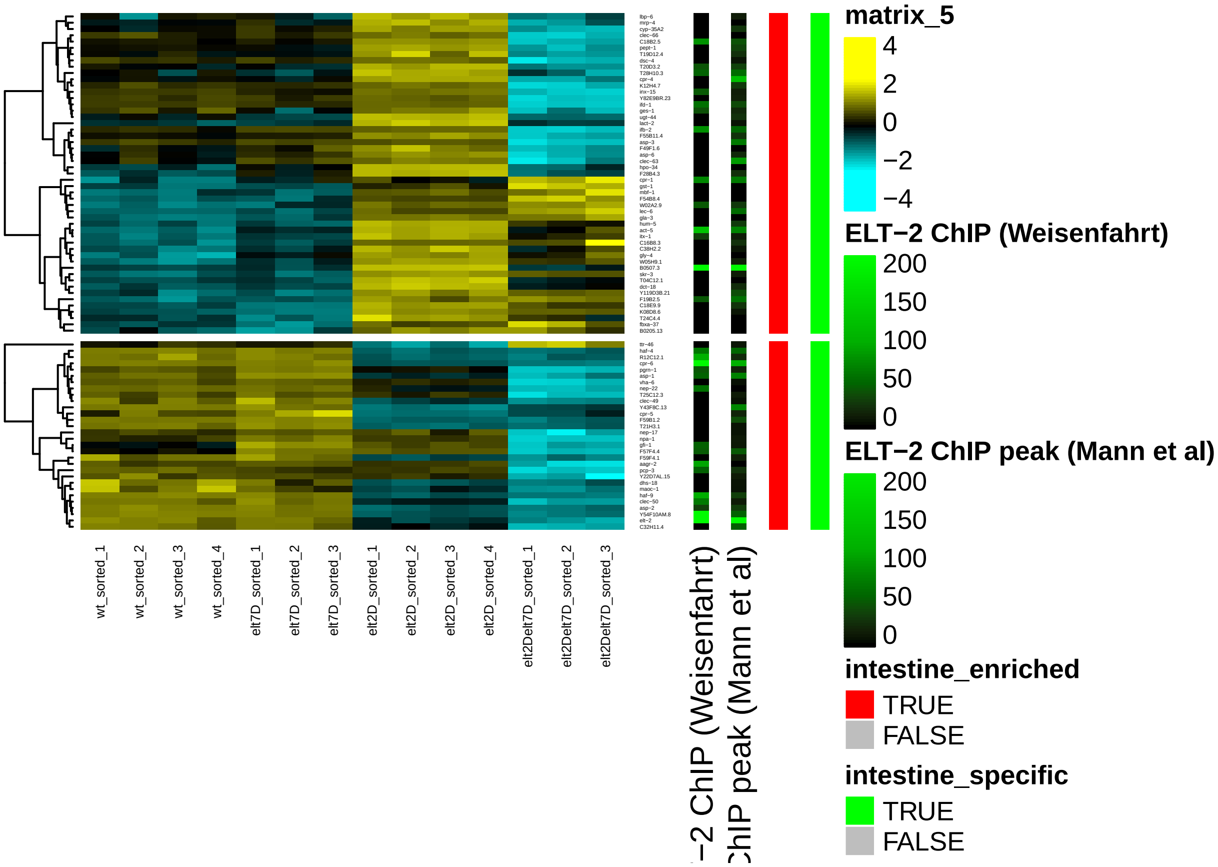
```
enriched_changing_cluster_list <- return_cluster(changing_pairwise_rlog_counts_intEnr, 5)

setwd(paste(projectdir, "03_output", sep = "/"))
dir.create(paste(projectdir, "03_output", "step6_cluster_analysis", sep = "/"), showWarnings = FALSE)
setwd(paste(projectdir, "03_output", "step6_cluster_analysis", sep = "/"))
pdf(paste(Sys.Date(), "all_changing_genes_cluster_annotated_enriched.pdf", sep = "_" ), width=5, height=
draw(enriched_changing_cluster_list)
dev.off()
```

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

```
# Generate the clustered heatmap for intestine-specific genes
changing_pairwise_rlog_counts_intSp <- changing_pairwise_rlog_counts %>%
         filter(intestine_sp == TRUE)

dim(changing_pairwise_rlog_counts_intSp)
```

```
## [1] 81 39
```

```
#head(changing_pairwise_rlog_counts_intSp)

draw(return_cluster(changing_pairwise_rlog_counts_intSp, 2))
```



```
specific_changing_cluster_list <- return_cluster(changing_pairwise_rlog_counts_intSp, 2)

setwd(paste(projectdir, "03_output", sep = "/"))
dir.create(paste(projectdir, "03_output", "step6_cluster_analysis", sep = "/"), showWarnings = FALSE)
setwd(paste(projectdir, "03_output", "step6_cluster_analysis", sep = "/"))
pdf(paste(Sys.Date(), "ungrouped_changing_genes_cluster_annotated_specific.pdf", sep = "_" ), width=5,
draw(specific_changing_cluster_list)
dev.off()
```

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

Save genelists over each subcluster

```
# Write a script to automate the heatmap generation:

return_clusterlist <- function(dataframe, num){
        #dataframe <- changing_pairwise_rlog_counts_intSp
        # Convert dataframe to a matrix
        rld_pairwise_matrix <- as.matrix(dataframe[,c(2:15)])
        rld_pairwise_matrix <- rld_pairwise_matrix[,c(8:14, 1:7)]
        row.names(rld_pairwise_matrix) <- dataframe$gene_id_val

        # scaling, no centering:
        mat_scaled = t(apply(unlist(rld_pairwise_matrix), 1, scale))
        colnames(mat_scaled) <- colnames(rld_pairwise_matrix)

        # Draw main RNA-seq heatmap
        ht1 <- Heatmap(mat_scaled,
          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 = num)

        return(ht1)
}

# Generate the clustered heatmap list for all genes
dim(changing_pairwise_rlog_counts)
```

```
## [1] 3092   39
```

```
changing_cluster_list <- return_clusterlist(changing_pairwise_rlog_counts, 6)

#Extract WBGene identifiers for each cluster:
set1 <- as.vector(changing_pairwise_rlog_counts[row_order(changing_cluster_list)[[1]],]$Row.names)
set3 <- as.vector(changing_pairwise_rlog_counts[row_order(changing_cluster_list)[[2]],]$Row.names)
set2 <- as.vector(changing_pairwise_rlog_counts[row_order(changing_cluster_list)[[3]],]$Row.names)
set4 <- as.vector(changing_pairwise_rlog_counts[row_order(changing_cluster_list)[[4]],]$Row.names)
set5 <- as.vector(changing_pairwise_rlog_counts[row_order(changing_cluster_list)[[5]],]$Row.names)
set6 <- as.vector(changing_pairwise_rlog_counts[row_order(changing_cluster_list)[[6]],]$Row.names)
length(set1)
```

```
## [1] 291
```

```
length(set2)
```

```
## [1] 1208
```

```
length(set3)
```

```
## [1] 405
```

```
length(set4)
```

```
## [1] 103
```

```
length(set5)
```

```
## [1] 65
```

Save a Supplemental Data Table containing all changing genes, their annotations, and their set category. . . .

```
# Merge changing_pairwise_r_log_counts with the set ID lists...
changing_pairwise_rlog_counts_plusSet <- changing_pairwise_rlog_counts %>%
  mutate(set = ifelse(changing_pairwise_rlog_counts$Row.names %in% set1, "set1",
                      ifelse(changing_pairwise_rlog_counts$Row.names %in% set2, "set2",
                             ifelse(changing_pairwise_rlog_counts$Row.names %in% set3, "set3",
                                    ifelse(changing_pairwise_rlog_counts$Row.names %in% set4, "set4",
                                           ifelse(changing_pairwise_rlog_counts$Row.names %in% set5, "se
                                                  ifelse(changing_pairwise_rlog_counts$Row.names %in% s

# Save to a file
setwd(paste(projectdir, "03_output", "step6_cluster_analysis", sep = "/"))
filename = paste(Sys.Date(), "Supplemental_Dataset_all_changing_genes_annotated_sets.txt", sep = "_")

write.table(changing_pairwise_rlog_counts_plusSet, file = filename, sep = "\t", quote = FALSE)
```

Save lists for GO ontology

```
#Save all clusters in a set list:
# notes, set2 and 3 were switched in illustrator, so I'll amend the code to reflect that here...
setlist <- list(set1 = set1,
                set2 = set2,
                set3 = set3,
                set4 = set4,
                set5 = set5,
                set6 = set6)


#Write set lists to file:
setwd(paste(projectdir, "03_output", sep = "/"))
dir.create(paste(projectdir, "03_output", "step6_clusters_for_GO", sep = "/"), showWarnings = FALSE)
setwd(paste(projectdir, "03_output", "step6_clusters_for_GO", sep = "/"))
getwd()
```

```
## [1] "/Users/erinnishimura/Dropbox/labwork/2016_ELT2_PROJECT/07_DESeq2_Analysis_EOP215/03_output/step
```

```
for (n in c(1:length(setlist))) {
  write(setlist[[n]], file = paste(Sys.Date(), "Geneset_cluster", n, "all.txt", sep = "_" ))
}

# Write background list:
write(as.vector(rlog_annot$Row.names), file = paste(Sys.Date(), "background", "all.txt", sep = "_" ))
```

```
# write a list for homer for clustering information:

#load annotation information
setwd(paste(projectdir, "03_output", sep = "/"))
lookup <- read.table(file = "2016-07-04_lookup_table_ce10_ce11.pdf",
                     sep = "\t", header = TRUE)
```

```r
#Some Lookup table entries have two WBGene entries:
duplicated_names <- names(which(table(lookup$WBGene) >1 ))
length(duplicated_names)
```

```
## [1] 41
```

```r
duplicated_lookup_entries <- lookup[lookup$WBGene %in% duplicated_names,]

#Resolve the lookup table entries by taking the first entry of each:
duplicated_genes <- lookup[lookup$WBGene %in% duplicated_names,]
dim(duplicated_genes)
```

```
## [1] 82 19
```

```r
non_duplicated_genes <- lookup[!(lookup$WBGene %in% duplicated_names),]
dim(non_duplicated_genes)
```

```
## [1] 43400    19
```

```r
lookup <- rbind(non_duplicated_genes, duplicated_genes[seq(1,82,2),])
dim(lookup)
```

```
## [1] 43441    19
```

```r
convert_names_homer <- function(vector) {
        newnames <- lookup[which(lookup$WBGene %in% vector),]$transcript
        return(as.vector(newnames))
        print(length(newnames))
        print(length(vector))

}

setlist_homer <- lapply(setlist, convert_names_homer)

#Write set lists to file:
setwd(paste(projectdir, "03_output", sep = "/"))
dir.create(paste(projectdir, "03_output", "step6_clusters_for_homer", sep = "/"), showWarnings = FALSE)
setwd(paste(projectdir, "03_output", "step6_clusters_for_homer", sep = "/"))
getwd()
```

```
## [1] "/Users/erinnishimura/Dropbox/labwork/2016_ELT2_PROJECT/07_DESeq2_Analysis_EOP215/03_output/step6
```

```r
for (n in c(1:length(setlist))) {
  setlist_homer[[n]]
  write(unlist(strsplit(setlist_homer[[n]], split = ",")), file = paste(Sys.Date(), "Geneset_cluster_hm
}

# Write background list:
write(unlist(strsplit(as.character(rlog_annot$transcripts), split = ",")), file = paste(Sys.Date(), "ba
```

Find transcription factors in the different sets:

Don't evaluate

Draw plots of elt-2 ChIP-seq data

Don't evaluate

Save the genesets for the intestine specific cluster

```
# Perform the clustering to divide the intestine specific genes into four clusters
intspecific_cluster_list <- return_clusterlist(changing_pairwise_rlog_counts_intSp, 4)


#Extract WBGene identifiers for each cluster:
classB <- as.vector(changing_pairwise_rlog_counts_intSp[row_order(intspecific_cluster_list)[[1]],]$Row.n
classC <- as.vector(changing_pairwise_rlog_counts_intSp[row_order(intspecific_cluster_list)[[2]],]$Row.n
classD <- as.vector(changing_pairwise_rlog_counts_intSp[row_order(intspecific_cluster_list)[[3]],]$Row.n
classA <- as.vector(changing_pairwise_rlog_counts_intSp[row_order(intspecific_cluster_list)[[4]],]$Row.n

length(classA)
```

```
## [1] 29
```

```
length(classB)
```

```
## [1] 26
```

```
length(classC)
```

```
## [1] 25
```

```
length(classD)
```

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

```
#Save all clusters in a set list:
# notes, set2 and 3 were switched in illustrator, so I'll amend the code to reflect that here...
sp_setlist <- list(setA = classA,
                setB = classB,
                setC = classC,
                setD = classD)


#Write set lists to file:
setwd(paste(projectdir, "03_output", sep = "/"))
dir.create(paste(projectdir, "03_output", "step6_clusters_for_bifrucationPlot", sep = "/"), showWarnings
setwd(paste(projectdir, "03_output", "step6_clusters_for_bifrucationPlot", sep = "/"))
getwd()
```

```
## [1] "/Users/erinnishimura/Dropbox/labwork/2016_ELT2_PROJECT/07_DESeq2_Analysis_EOP215/03_output/step6
```

```
for (n in c(1:length(sp_setlist))) {
  write(sp_setlist[[n]], file = paste(Sys.Date(), "_geneset_cluster", n, "intsp.txt", sep = "_" ))
}
```

Make a new Supplemental Dataset with just the intestine specific information and the different classes...

```
# head(changing_pairwise_rlog_counts_intSp)
dim(changing_pairwise_rlog_counts_intSp)
```

```
## [1] 81 39
```

```
# Merge changing_pairwise_r_log_counts with the set ID lists...
changing_intestineSp_rlog_counts_plusSet <- changing_pairwise_rlog_counts_intSp %>%
  mutate(class = ifelse(changing_pairwise_rlog_counts_intSp$Row.names %in% classA, "classA",
                      ifelse(changing_pairwise_rlog_counts_intSp$Row.names %in% classB, "ClassB",
                             ifelse(changing_pairwise_rlog_counts_intSp$Row.names %in% classC, "ClassC"
                                    ifelse(changing_pairwise_rlog_counts_intSp$Row.names %in% classD, "


# head(changing_intestineSp_rlog_counts_plusSet)
dim(changing_intestineSp_rlog_counts_plusSet)
```

```
## [1] 81 40
```

```
# Save to a file
setwd(paste(projectdir, "03_output", "step6_cluster_analysis", sep = "/"))
filename = paste(Sys.Date(), "Supplemental_Dataset_intestineSp_changing_annotated_sets.txt", sep = "_")

write.table(changing_intestineSp_rlog_counts_plusSet, file = filename, sep = "\t", quote = FALSE)
```

```
sessionInfo()
```

```
## R version 3.2.4 Revised (2016-03-16 r70336)
## Platform: x86_64-apple-darwin13.4.0 (64-bit)
## Running under: OS X 10.11.6 (El Capitan)
##
## 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] stats4    parallel  grid      stats     graphics  grDevices utils
##  [8] datasets  methods   base
##
## other attached packages:
##  [1] bindrcpp_0.2         circlize_0.4.0       GenomicRanges_1.22.4
##  [4] GenomeInfoDb_1.6.3   IRanges_2.4.8        S4Vectors_0.8.11
##  [7] BiocGenerics_0.16.1  dplyr_0.7.1          RColorBrewer_1.1-2
## [10] ComplexHeatmap_1.6.0
##
## loaded via a namespace (and not attached):
##  [1] shape_1.4.2          modeltools_0.2-21    GetoptLong_0.1.6
##  [4] kernlab_0.9-25       lattice_0.20-35      colorspace_1.3-2
##  [7] htmltools_0.3.6      viridisLite_0.2.0    yaml_2.1.14
## [10] rlang_0.1.1          glue_1.1.1           prabclus_2.2-6
## [13] fpc_2.1-10           plyr_1.8.4           bindr_0.1
## [16] zlibbioc_1.16.0      robustbase_0.92-7    stringr_1.2.0
## [19] munsell_0.4.3        gtable_0.2.0         GlobalOptions_0.0.12
## [22] mvtnorm_1.0-6        evaluate_0.10.1      knitr_1.16
## [25] flexmix_2.3-14      class_7.3-14        DEoptimR_1.0-8
## [28] trimcluster_0.1-2   Rcpp_0.12.11        scales_0.4.1
## [31] backports_1.1.0     diptest_0.75-7      XVector_0.10.0
## [34] gridExtra_2.2.1     rjson_0.2.15        ggplot2_2.2.1
## [37] digest_0.6.12       stringi_1.1.5       rprojroot_1.2
## [40] tools_3.2.4         magrittr_1.5        lazyeval_0.2.0
## [43] tibble_1.3.3        cluster_2.0.6       whisker_0.3-2
## [46] pkgconfig_2.0.1     dendextend_1.5.2    MASS_7.3-47
## [49] Matrix_1.2-10       assertthat_0.2.0    rmarkdown_1.6
## [52] viridis_0.4.0       R6_2.2.2            mclust_5.3
## [55] nnet_7.3-12
```