# 2 Differential Gene Expression Analysis and GO Enrichment

*Stefan Siebert and Bryan Teefy*

*12/20/2019*

## Differential Gene Expression

We performed differential gene expression analysis between wildtype and *hywi* knockdown animals. As a first step we generated expression estimates using RSEM/bowtie and used reads that were filtered for adapters (TruSeq3) using trimmomatic (LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36). We made use of the RSEM functions rsem-calculate-expression (–forward-prob 0) and rsem-generate-data-matrix to generate an expression matrix (scripts: DGE_expression.sh, DGE_count_matrix.sh). The raw count matrix is available at GEO (GSE135440).

## Load the expression data / GO annotations

```
# Load the Differential Gene Expression Count Matrix

counts <- read.table("objects/Differential_Gene_Expression_Count_Matrix.txt", sep = "\t",
    check.names = FALSE, header = TRUE, row.names = 1)

# Load normalized piRNA and degradome reads and transcript characterization from
# RMD1

piRNA_Deg_counts <- read.table("objects/Annotated_piRNA_Degradome_Count_Matrix.txt",
    sep = "\t")
```

## Load Required Packages

```
library(edgeR)
library(knitr)
library(xtable)
library(ggplot2)
```

## Data exploration

We explore the data to get insights into the paired nature of the triplicate samples from wildtype and *hywi* knockdown tissue.

```
# We first calculate normalized counts for future use and to include them in the
# master dataframe.

# set gene IDs as rownames
rownames(counts) <- counts[, 1]
```

```r
# raw counts from all treatments
k <- counts[, c(2:7)]

# calculate normalization factors
nf_k <- calcNormFactors(k)

# calculate library sizes
ls_k <- colSums(k)

# effective library size using normalization factors
lse_k <- ls_k * nf_k

# normalization multiplier to use on counts
nm_k <- 1e+06/lse_k

# normalize counts using normalization multiplier
k <- k * nm_k

# round normalized counts
k <- round(k, digits = 0)

# restore ID column
k$ID <- rownames(k)

# reorder columns
k <- k[, c(7, 1:6)]

# combine rounded raw and normalized counts
k <- merge(counts[, c(1:7)], k, by = "ID")

# rename columns

colnames(k) <- c("ID", "WT1", "WT2", "WT3", "KD1", "KD2", "KD3", "nWT1", "nWT2",
    "nWT3", "nKD1", "nKD2", "nKD3")
```

```r
# explore replication

# define treatment groups for DGE
TR_k <- factor(c("k", "k", "k", "w", "w", "w"))

# set wild type as reference
TR_k <- relevel(TR_k, ref = "w")

# create experimental design data frame defining treatment type for each sample
d_k <- data.frame(Sample = colnames(counts[, c(5, 6, 7, 2, 3, 4)]), TR_k)

# generate DGEList object containing raw counts, the treatment type for each
# column, and the gene IDs
y <- DGEList(counts = counts[, c(5, 6, 7, 2, 3, 4)], group = d_k$TR_k, genes = counts[,
    1])

# label each column with the appropriate sample name
colnames(y) <- d_k$Sample
```

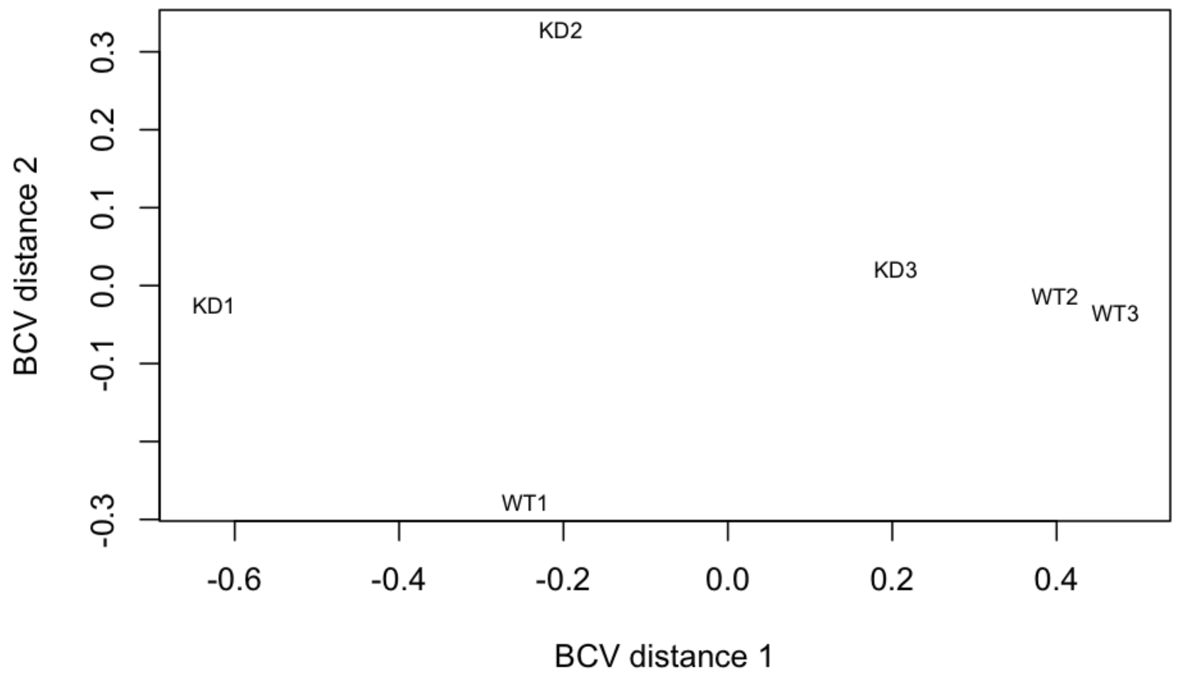
```
# calculate library size for each sample and store within DGElist
y$samples$lib.size <- colSums(y$counts)

# exclude transcripts that do not have at least two samples with more than one
# count per million
keep <- rowSums(cpm(y) > 1) >= 2
y <- y[keep, , keep.lib.sizes = FALSE]

# number of transcripts remaining after count filtering
dim(y)
```

```
## [1] 16435     6
```

```
# calculate normalization factors for each sample
y <- calcNormFactors(y)

# review library sizes and normalization factors
y$samples
```

```
##     group lib.size norm.factors
## KD1     k 27236412    1.0711124
## KD2     k 26955915    0.9927897
## KD3     k 27081381    1.0126300
## WT1     w 22165507    1.0181741
## WT2     w 27471090    0.9836625
## WT3     w 25597005    0.9272326
```

```
# create a matrix defining the control and treatment groups for the analysis
# based on what is described by the TR object
design_k <- model.matrix(~TR_k)

# Estimate dispersions
y <- estimateDisp(y, design_k)

# generate MDS Plot

plotMDS(y, method = "bcv", cex = 0.7)
```



```
# we exclude outlier replicates WT1 and KD3 from downstream analyses

# define treatment groups for DGE
TR_k <- factor(c("k", "k", "w", "w"))

# set wild type as reference
TR_k <- relevel(TR_k, ref = "w")

# create experimental design data frame defining treatment type for each sample
d_k <- data.frame(Sample = colnames(counts[, c(5, 6, 3, 4)]), TR_k)

# generate DGElist object containing raw counts, the treatment type for each
# column, and the gene annotations
y <- DGEList(counts = counts[, c(5, 6, 3, 4)], group = d_k$TR_k, genes = counts[,
    1])

# label each column with the appropriate sample name
colnames(y) <- d_k$Sample

# calculate library size for each sample and store within DGElist
y$samples$lib.size <- colSums(y$counts)

# exclude transcripts that do not have at least two samples with more than one
# count per million
keep <- rowSums(cpm(y) > 1) >= 2
y <- y[keep, , keep.lib.sizes = FALSE]
```

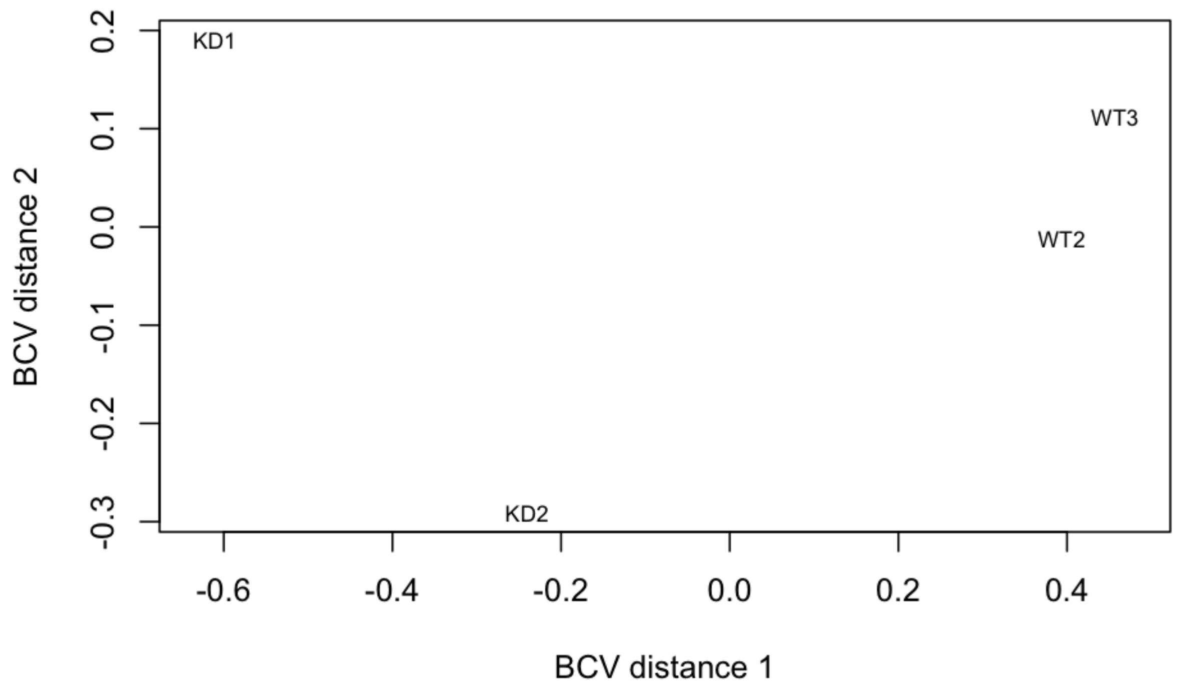
```
# number of transcripts remaining after count filtering
dim(y)
```

```
## [1] 15924     4
```

```
# calculate normalization factors for each sample
y <- calcNormFactors(y)

# review library sizes and normalization factors
y$samples
```

```
##     group lib.size norm.factors
## KD1     k 27217655    1.0849445
## KD2     k 26945252    0.9960237
## WT2     w 27460768    0.9922666
## WT3     w 25588479    0.9325978
```

```
# create a matrix defining the control and treatment groups for the analysis
# based on what is described by the TR object
design_k <- model.matrix(~TR_k)

# estimate dispersions
y <- estimateDisp(y, design_k)

# generate MDS Plot
plotMDS(y, method = "bcv", cex = 0.7)
```



## Differential Gene Expression (DGE) analysis - wildtype vs *hywi* knockdown

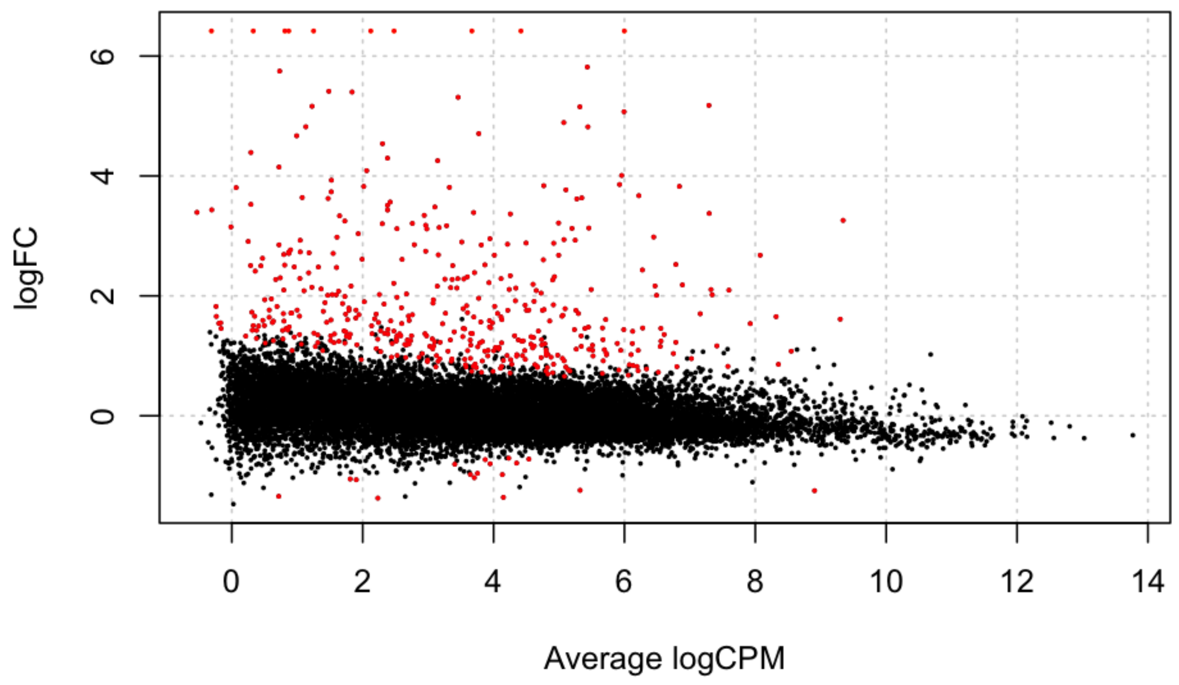
We perform DGE analysis after excluding outlier replicates WT1 and KD3.

```r
# DGE Analysis

# set p-value for cutoff
p.value = 0.05

# fit data to a negative binomial geralized log-linear model
fit_k <- glmFit(y, design_k)

# conduct a statistical test for differential gene expression based on the fit
# from
lrt_k <- glmLRT(fit_k)

# create a list of values that describes the status of each gene in the DGE test
de_k <- decideTestsDGE(lrt_k, adjust.method = "BH", p.value)

# overview of number of differentially expressed genes
summary(de_k)
```

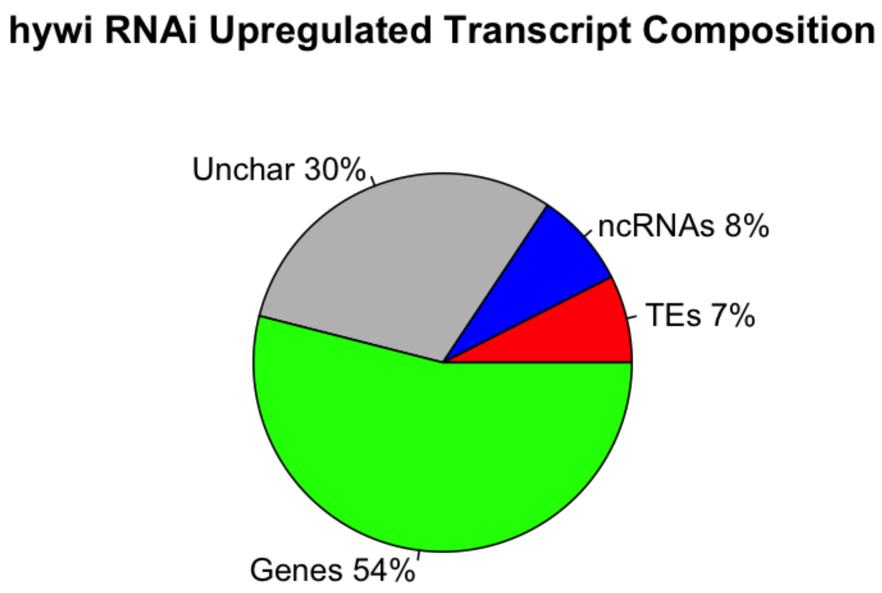
```
##         TR_kk
## Down       17
## NotSig 15466
## Up        441
```

```r
# build results table
D_k <- lrt_k$table

# restore ID column
D_k$ID <- rownames(D_k)

# add column of adjusted p values
D_k <- cbind(D_k, p.adjust(D_k$PValue, method = "BH"))
names(D_k)[names(D_k) == "p.adjust(D_k$PValue, method = \"BH\")"] = "k_Padj"

# create table summarizing rounded raw counts, normalized counts and DGE results
res_k <- merge(k, D_k, by = "ID", all = TRUE)

# call transcripts upregulated, downregulated, or unaffected

res_k$DE <- ifelse(res_k$k_Padj <= 0.05 & res_k$logFC >= 0 & !is.na(res_k$logFC &
    res_k$k_Padj), "Up", ifelse(res_k$k_Padj <= 0.05 & res_k$logFC <= 0 & !is.na(res_k$logFC &
    res_k$k_Padj), "Down", "None"))

# merge with the master dataframe

piRNA_Deg_counts <- merge(piRNA_Deg_counts, res_k, by = "ID")

# generate volcano plot

detags <- rownames(y)[as.logical(de_k)]
```

```r
plotSmear(lrt_k, de.tags = detags, cex = 0.2, cex.lab = 1, cex.axis = 1)
```



```r
# generate plot of upregulated transcripts expression fold change

Upreg_Types <- c(sum(piRNA_Deg_counts$DE == "Up" & piRNA_Deg_counts$Transcript_Class ==
    "TE"), sum(piRNA_Deg_counts$DE == "Up" & piRNA_Deg_counts$Transcript_Class ==
    "ncRNA"), sum(piRNA_Deg_counts$DE == "Up" & piRNA_Deg_counts$Transcript_Class ==
    "Unchar"), sum(piRNA_Deg_counts$DE == "Up" & piRNA_Deg_counts$Transcript_Class ==
    "Gene"))

lbls <- c("TEs", "ncRNAs", "Unchar", "Genes")
colors = c("red", "blue", "gray", "green")
pct <- round(Upreg_Types/sum(Upreg_Types) * 100)
lbls <- paste(lbls, pct)  # add percents to labels
lbls <- paste(lbls, "%", sep = "")  # ad % to labels
pie(Upreg_Types, labels = lbls, col = colors, main = "hywi RNAi Upregulated Transcript Composition")
```



## Somatic Hywi piRNA Mapping Density Ordering

To infer direct targets of Hywi, *hywi* RNAi upregulated transcripts can be described as high and low Hywi piRNA mapping transcripts. "High-mapping"" transcripts are likely to be direct Hywi targets while "low-mapping" transcripts are unlikely to be direct Hywi targets. High Mapping transcripts were defined as those transcripts that were in the top 20% of read counts per kilobase million after combining Colch Hywi Sense and Colch Hywi Antisense reads.

To infer which transcripts were most likely to be involved in the ping-pong cycle in somatic stem cells, transcripts were ordered by Colch Hywi Antisense reads per kilobase million. Transcripts that fall within the top 20% in this category were considered to be putative ping-pong transcripts.

To infer which transcripts may be processed by Hywi phasing in somatic stem cells, transcripts were ordered by Colch Hywi Sense reads per kilobase million. Transcripts that fall within the top 20% in this category were considered to be putative phased transcripts.

The same process was performed for Hyli piRNAs.

```
# Generate percentiles for the different groups

piRNA_Deg_counts$Somatic_Hywi_Perc <- (piRNA_Deg_counts$Colch_Hywi_AS_Counts_RPK +
    piRNA_Deg_counts$Colch_Hywi_S_Counts_RPK)

piRNA_Deg_counts$Somatic_Hywi_Perc <- ecdf(piRNA_Deg_counts$Somatic_Hywi_Perc)(piRNA_Deg_counts$Somatic_

# Repeat for only Colch Hywi Antisense piRNAs
```

```r
piRNA_Deg_counts$Somatic_Hywi_Perc_AS <- (piRNA_Deg_counts$Colch_Hywi_AS_Counts_RPK)

piRNA_Deg_counts$Somatic_Hywi_Perc_AS <- ecdf(piRNA_Deg_counts$Somatic_Hywi_Perc_AS)(piRNA_Deg_counts$So

# Repeat for only Colch Hywi Sense piRNAs

piRNA_Deg_counts$Somatic_Hywi_Perc_S <- (piRNA_Deg_counts$Colch_Hywi_S_Counts_RPK)

piRNA_Deg_counts$Somatic_Hywi_Perc_S <- ecdf(piRNA_Deg_counts$Somatic_Hywi_Perc_S)(piRNA_Deg_counts$Soma

# Repeat with Hyli

piRNA_Deg_counts$Somatic_Hyli_Perc <- (piRNA_Deg_counts$Colch_Hyli_AS_Counts_RPK +
    piRNA_Deg_counts$Colch_Hyli_S_Counts_RPK)

piRNA_Deg_counts$Somatic_Hyli_Perc <- ecdf(piRNA_Deg_counts$Somatic_Hyli_Perc)(piRNA_Deg_counts$Somatic_

# Repeat for only Colch Hyli Antisense piRNAs

piRNA_Deg_counts$Somatic_Hyli_Perc_AS <- (piRNA_Deg_counts$Colch_Hyli_AS_Counts_RPK)

piRNA_Deg_counts$Somatic_Hyli_Perc_AS <- ecdf(piRNA_Deg_counts$Somatic_Hyli_Perc_AS)(piRNA_Deg_counts$So

# Repeat for only Colch Hyli Sense piRNAs

piRNA_Deg_counts$Somatic_Hyli_Perc_S <- (piRNA_Deg_counts$Colch_Hyli_S_Counts_RPK)

piRNA_Deg_counts$Somatic_Hyli_Perc_S <- ecdf(piRNA_Deg_counts$Somatic_Hyli_Perc_S)(piRNA_Deg_counts$Soma

# write table that summarizes data write.table(piRNA_Deg_counts, file =
# 'objects/Annotated_piRNA_Degradome_DGE_Count_Matrix.txt', sep = '\t')
```

## GO-Term Enrichment Analysis

GO-term enrichment analysis was performed on upregulated transcripts against the entire transcriptome to investigate a functional response to somatic *hywi* knockdown.

GO-term enrichment analysis was performed using goatools v0.6.10 using the script: goatools_GO_enrichment.pl

```r
# Load GO annotation results

GO_table <- read.table("objects/GO_upreg_trans_full_ref.txt", header = T, sep = "\t")

# Take enriched biological processes

GO_table_sub <- subset(GO_table, p_bonferroni <= 0.05 & NS == "BP")

# Include GO accession number, GO term, ratio in study, ratio in population,
# bonferroni corrected p-value, and transcript IDs

GO_table_sub <- GO_table_sub[, c(1, 4:6, 10, 14)]
colnames(GO_table_sub) <- c("GO", "GO_Term", "Study", "Pop", "p_val", "ID")
```

```r
# Print table

kable(GO_table_sub, format = "markdown", padding = 100)
```

| GO | GO_Term | Study | Pop | p_val | ID |
|---|---|---|---|---|---|
| ...GO: 0006952 | defense response | 26/441 | 753/38747 | 0.0142 | t11117aep, t12198aep, t16424aep, t17178aep, t17750aep, t21013aep, t21682aep, t22133aep, t24687aep, t32280aep, t33020aep, t34385aep, t34424aep, t34475aep, t35573aep, t35608aep, t35837aep, t38672aep, t38673aep, t5914aep, t6387aep, t7326aep, t7388aep, t8582aep, t8645aep, t8946aep |
| ....GO: 0051899 | membrane depolarization | 7/441 | 46/38747 | 0.0170 | t17803aep, t18338aep, t24938aep, t25020aep, t526aep, t527aep, t9936aep |
| .......GO: 2000051 | negative regulation of non-canonical Wnt signaling pathway | 4/441 | 8/38747 | 0.0221 | t29674aep, t30176aep, t33022aep, t8142aep |
| ....GO: 0048440 | carpel development | 3/441 | 3/38747 | 0.0290 | t35573aep, t38672aep, t38673aep |
| ...GO: 0045087 | innate immune response | 14/441 | 258/38747 | 0.0370 | t11117aep, t17178aep, t22133aep, t24687aep, t34385aep, t34424aep, t35573aep, t35608aep, t35837aep, t38672aep, t38673aep, t6387aep, t7326aep, t8946aep |
| .......GO: 0031050 | dsRNA processing | 5/441 | 19/38747 | 0.0377 | t30770aep, t35488aep, t38672aep, t38673aep, t5914aep |
| ......GO: 0042108 | positive regulation of cytokine biosynthetic process | 5/441 | 20/38747 | 0.0498 | t21682aep, t22133aep, t24687aep, t34424aep, t8645aep |

### Software versions

This document was computed on Fri Jan 10 10:35:54 2020 with the following R package versions.

```
R version 3.5.3 (2019-03-11)
Platform: x86_64-apple-darwin15.6.0 (64-bit)
Running under: macOS Mojave 10.14.5

Matrix products: default
BLAS: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/3.5/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] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] ggplot2_3.2.0 xtable_1.8-3  edgeR_3.22.5  limma_3.38.3  knitr_1.22

loaded via a namespace (and not attached):
 [1] Rcpp_1.0.1       magrittr_1.5     splines_3.5.3    tidyselect_0.2.5
 [5] munsell_0.5.0    colorspace_1.4-1 lattice_0.20-38  R6_2.4.0
 [9] rlang_0.4.0      highr_0.7        dplyr_0.8.3      stringr_1.4.0
[13] tools_3.5.3      grid_3.5.3       gtable_0.3.0     xfun_0.5
[17] withr_2.1.2      htmltools_0.3.6  assertthat_0.2.1 yaml_2.2.0
[21] lazyeval_0.2.2   digest_0.6.20    tibble_2.1.3     crayon_1.3.4
[25] purrr_0.3.2      formatR_1.7      glue_1.3.1       evaluate_0.13
[29] rmarkdown_1.12   stringi_1.4.3    compiler_3.5.3   pillar_1.4.2
[33] scales_1.0.0     locfit_1.5-9.1   pkgconfig_2.0.2
```