Shaleigh Smith

Next Generation Sequencing Informatics

Assigned Coursework #5

This study performed a differential gene expression analysis of RNA-Seq data using two different packages, DESeq2 and edgeR. The data had two conditions: control and expression. There were six control samples, EV1-6, and three expression vector samples, mVLT1-3. All samples were paired-end and derived from stranded RNA sequencing using dUTP method. There were two separate batches, the first was EV1-3 and the second was EV4-6 and mVLT1-3.

The data was quality trimmed using Trim Galore at phred score 30. It was then aligned to the human genome using BBMap with a minimum identity score of 0.90 to increase sensitivity and an ambiguous tag of random to randomly select the top scoring read when there were multiple mapped to the same location. The sam file that was generated was sorted and converted to a bam file. This bam file was then counted using Subread featureCounts. The input was specified as reversely stranded (-s 2) and paired end (-p). The fragment length was checked (-P) and only the fragments that had both ends successfully aligned were counted (-B). Chimeric fragments were ignored (-C) as were duplicates (--ignoreDup). Finally, only primary alignments were counted (--primary) and the counts were based on gene_id (-g). These parameters insured stringent feature counting by removing duplicates, chimeric fragments, secondary alignments, and incomplete fragments. The resulting feature count files were exported and joined in R by gene_id to create one feature count data frame with all nine samples.

The differential expression analysis was done in two parts in R using both DESeq2 and edgeR. In both analyses, the controls were always used as the standard to determine what was up-regulated or down-regulated in the mVLT vector. The first part of the differential analysis focused on identifying the batch effect: all three of the groups (control 1, control 2, and expression vector) were contrasted against each other. This confirmed that there was a batch effect, more specifically it visually and statistically separated EV1-3 from EV4-6 and the expression vector samples. This was shown in both the PCA plot from DESeq2 and the MDS plot from edgeR where the EV samples were in distinctly different clusters despite both being controls (Figure 1). This was also displayed in both heatmap results where control 1 and control 2 did not cluster together (Figure 2). Furthermore, when the controls were contrasted with the expression vector directly, they had differing numbers of up-regulated and down-regulated genes (Figure 3 and Table 1).

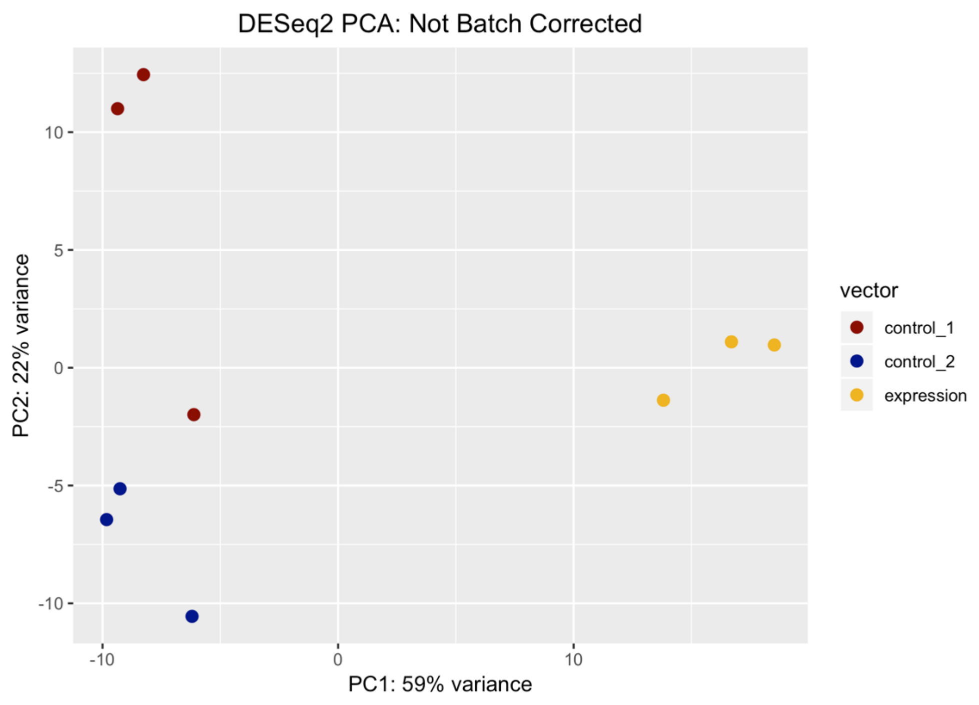
The second part of the differential analysis corrected for the batch effect and then contrasted the expression vector against all six controls. This gave the final differential gene expression results. The

DESeq2 PCA and edgeR MDS gave the same results as the first analysis because the batch effect wasn't corrected before plotting (Figures 4). The heatmaps were also comparable between DESeq2 and edgeR; the most noticeable difference was that DESeq2 rowMeans had a smaller range than edgeR (Figure 5). This was also seen in the volcano plots, where the log2 fold change for edgR had a larger range than in DESeq2 (Figure 6). When contrasting the three expression vector samples to all six controls DESeq2 had 720 up-regulated genes and 498 down-regulated genes while edgeR had 738 up-regulated genes and 691 down-regulated genes. The intersect of the two was 577 for up-regulated genes and 367 for down-regulated genes (Table 2). The intersect was lower than expected as both packages were given the same data to analyze. Additionally, the gene-specific p-values and log2 fold change noticeably differed between the two. This exemplified the fact that DESeq2 and edgeR use different methods to undergo differential gene expression analysis and therefore their discrepancies should be taken into account in future analyses.

See below for figures and code.

Figure 1. Dimensional Analysis Plots. **(a)** DESeq2 PCA plot without batch correction. Vectors (samples) included control 1 (red), control 2 (blue), and expression (yellow). **(b)** edgeR MDS plot without batch correction. Samples were visualized by sample name and group which included control 1 (red), control 2 (blue), and expression (yellow).

A



B

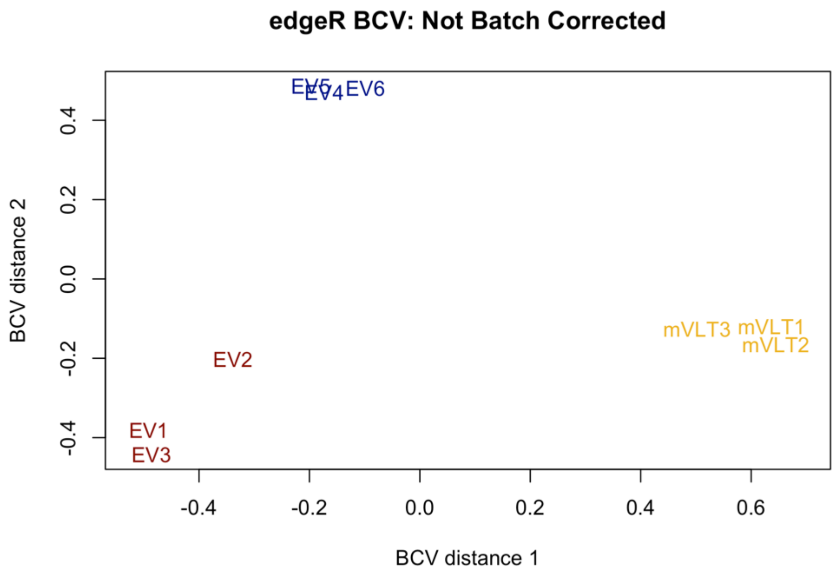
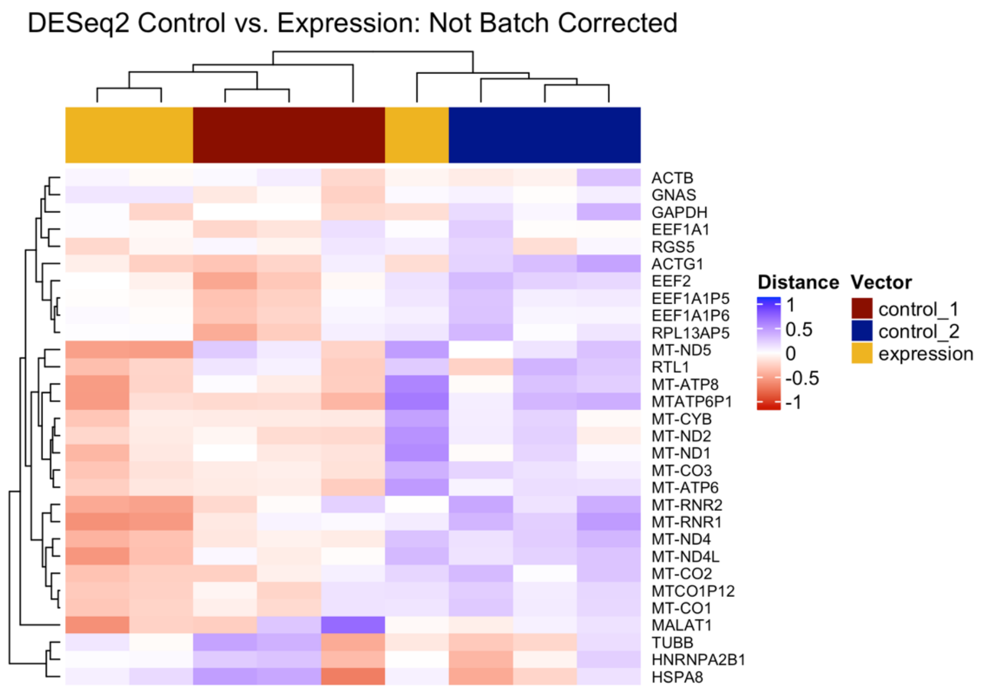


Figure 2. Variance heatmaps without batch correction. **(a-b)** DESeq2 and edgeR heatmaps of top 30 genes measure by row mean after differential analysis. Distance was equivalent to the gene value minus row mean. Vector (samples) included control 1 (red), control 2 (blue), and expression (yellow).

A



B

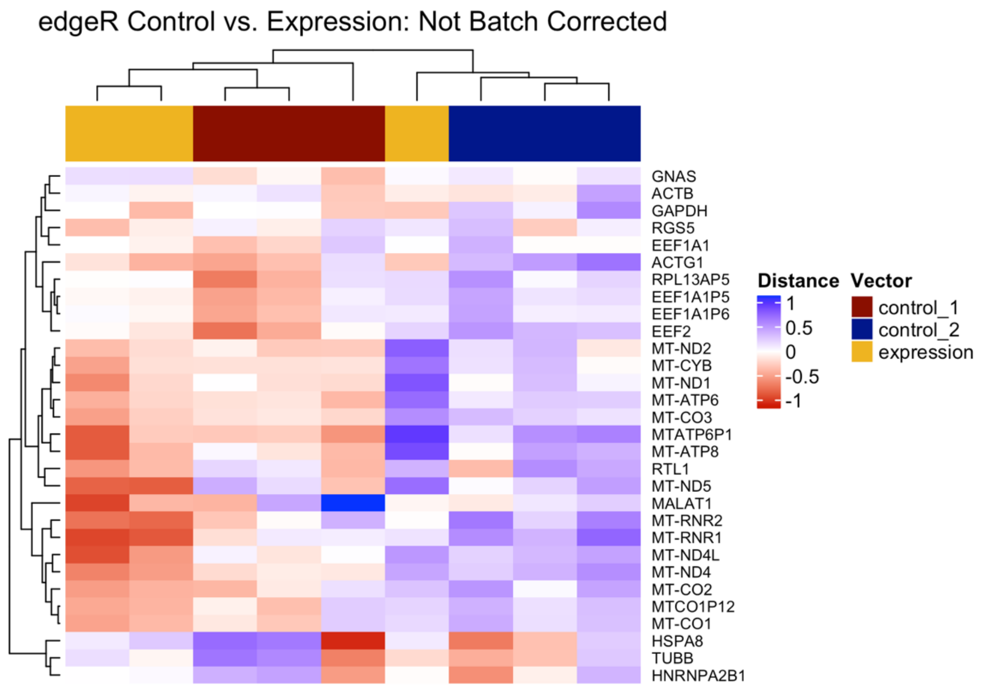
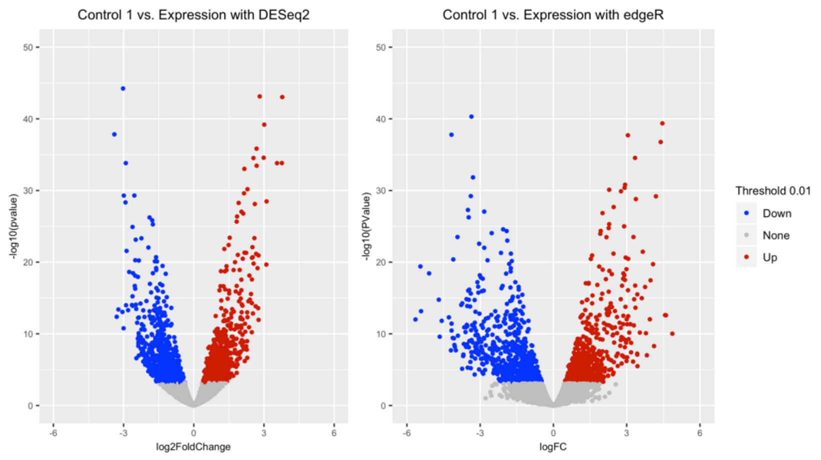
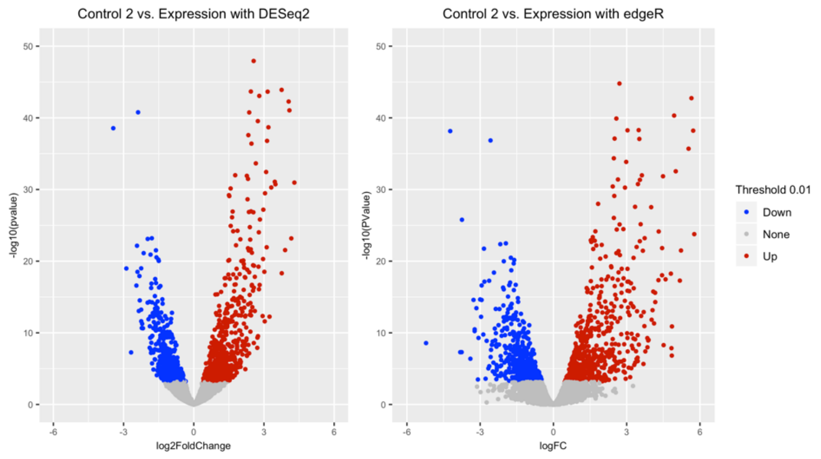


Figure 3. Volcano plots without batch correction. Color includes all genes with q value less than 0.01, down-regulated genes (blue) have a negative log2 fold change and up-regulated genes (red) have a positive log2 fold. Genes in gray were not significant and were greater than the set threshold. **(a)** Differential gene analysis results from control 1 against the expression vector. **(b)** Differential gene analysis results from control 2 against the expression vector. **(c)** Differential gene analysis results from control 1 against control 2.

A



B



C

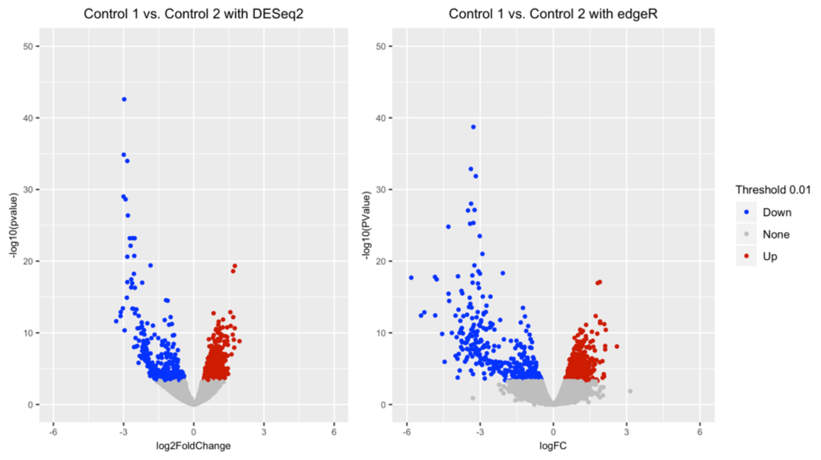
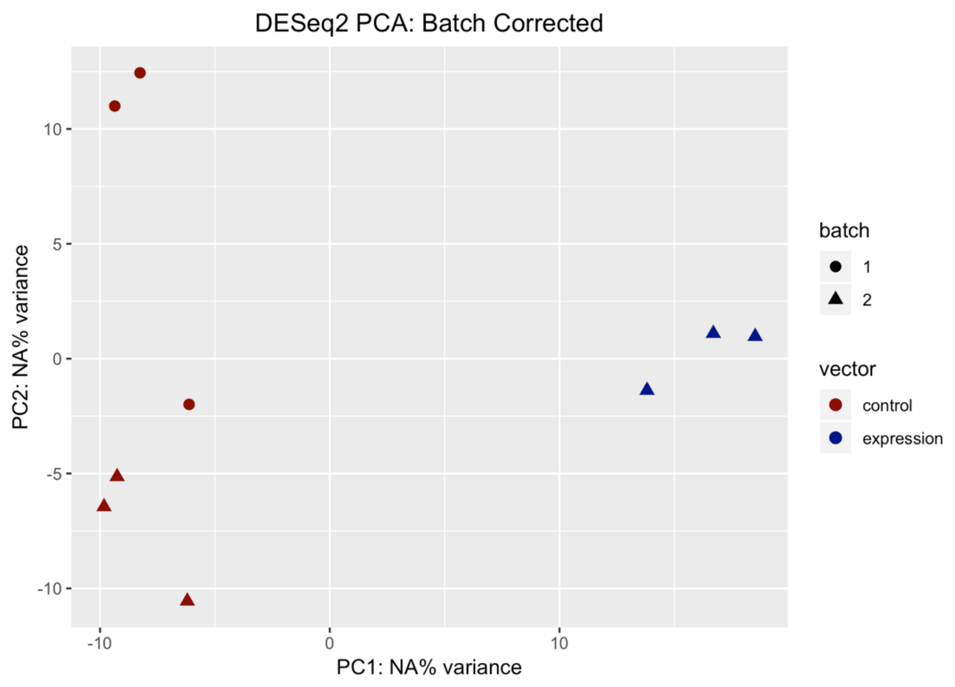


Table 1. Differential gene expression analysis results without batch correction. Includes the number of genes considered significant (below the q value threshold of 0.01) per contrast. Separated by up-regulated (positive log2 fold change) and down-regulated (negative log2 fold change). Control 1 against expression (blue), control 2 against expression (green), control 1 against control 2 (orange).

	Control 1 vs. Expression		
	DESeq2	edgeR	Intersect
Up-regulated	4726	2334	2245
Down-regulated	2091	3952	2090
Both	2091	6286	4335
	Control 2 vs. Expression		
	DESeq2	edgeR	Intersect
Up-regulated	56	66	53
Down-regulated	45	44	32
Both	101	110	84
	Control 1 vs. Control 2		
	DESeq2	edgeR	Intersect
Up-regulated	2613	2249	2191
Down-regulated	2116	4013	2116
Both	4729	6262	4307

Figure 4. Dimensional Analysis Plots. **(a)** DESeq2 PCA plot with batch correction. Vectors (samples) included controls (red) and expression (blue). Batch was grouped by shape, batch 1 (EV1-3) was a circle and batch 2 (EV4-6 and mVLT1-3) was a triangle. **(b)** edgeR MDS plot with batch correction. Samples were visualized by sample name and batch which included controls (red) and expression (blue).

A



B

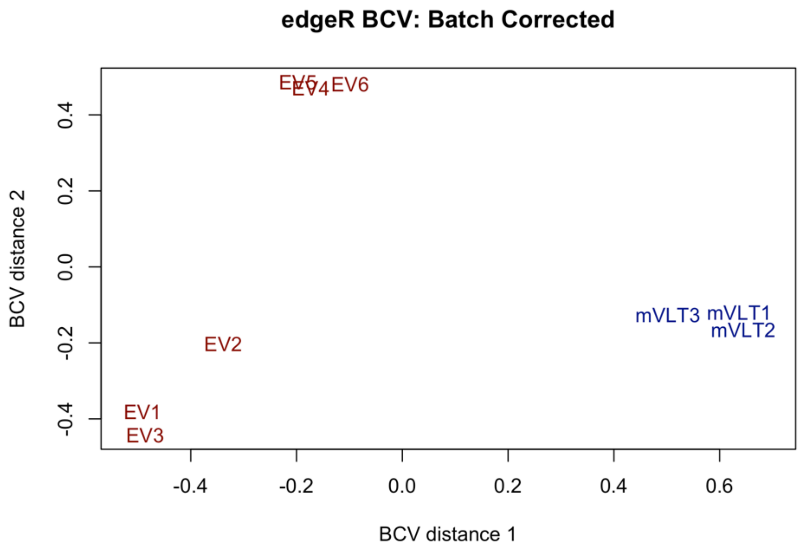
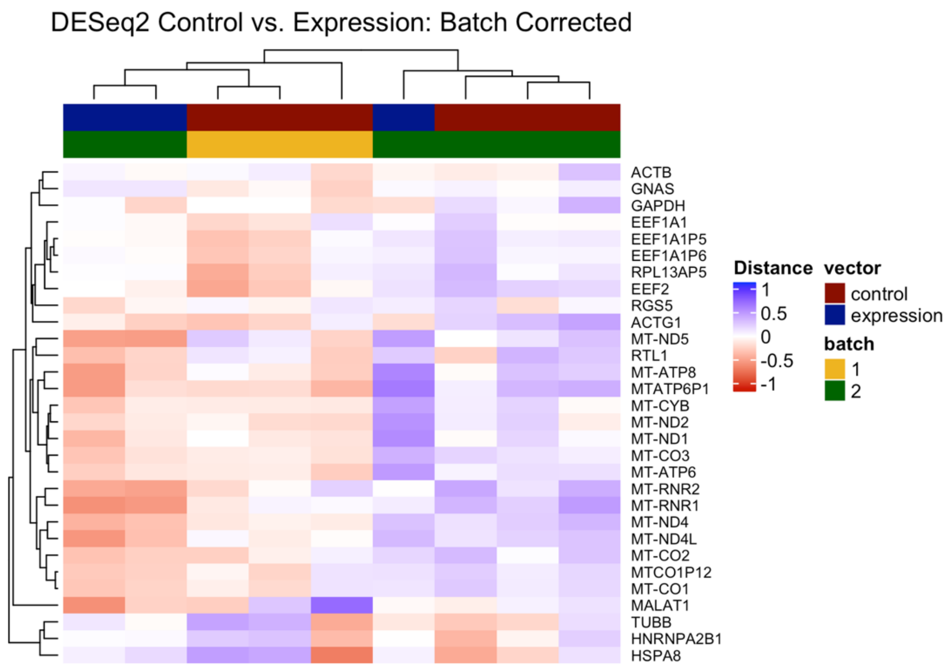


Figure 5. Variance heatmaps with batch correction. **(a-b)** DESeq2 and edgeR heatmaps of top 30 genes measure by row mean after differential analysis. Distance was equivalent to the gene value minus row mean. Vector (samples) included controls (red) and expression (blue). The batch group was visualized as batch 1, EV1-3 (yellow), and batch 2, EV4-6 and mVLT1-3 (green).

A



B

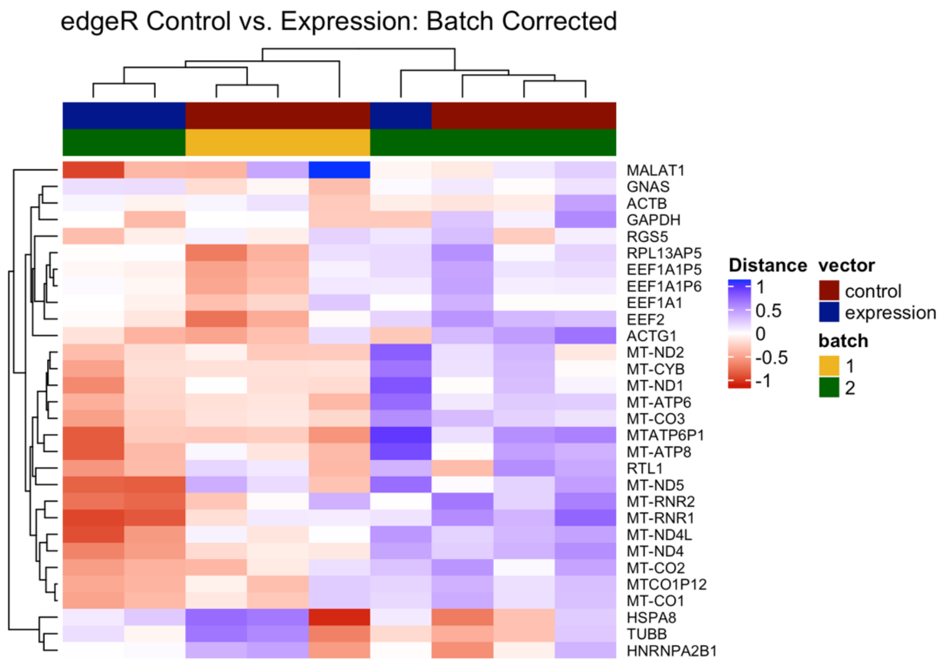


Figure 6. Volcano plot with batch correction of differential gene analysis results from control contrasted against the expression vector. Color includes all genes with q value less than 0.01, down-regulated genes (blue) have a negative log2 fold change and up-regulated genes (red) have a positive log2 fold. Genes in gray were not significant and were greater than the set threshold.

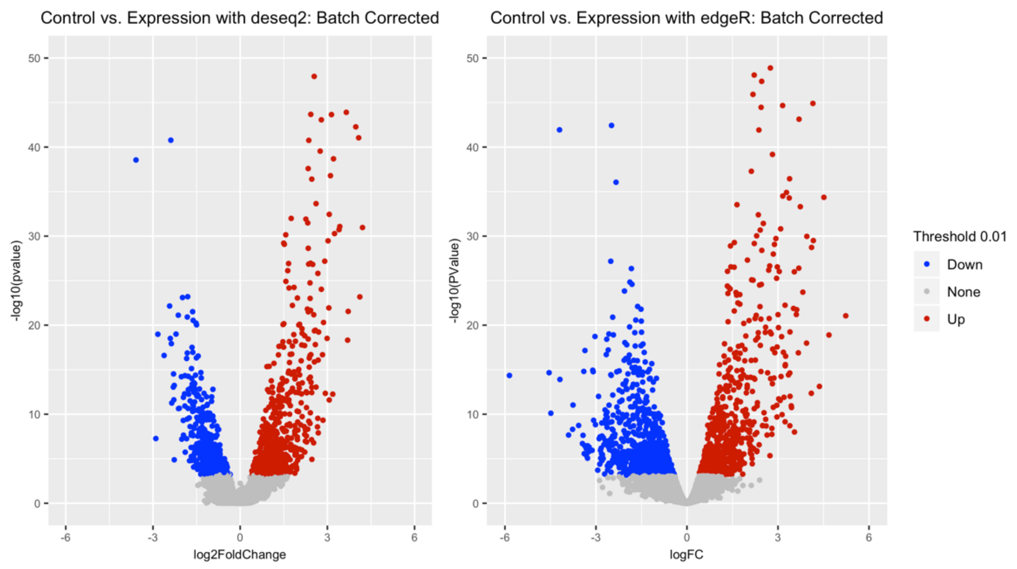


Table 2. Differential gene expression analysis results of control contrasted against expression with batch correction. Includes the number of genes considered significant (below the q value threshold of 0.01) per contrast. Separated by up-regulated (positive log2 fold change) and down-regulated (negative log2 fold change).

	Control vs. Expression		
	DESeq2	edgeR	Intersect
Up-regulated	720	738	577
Down-regulated	498	691	367
Both	1218	1429	944

Code

The code below trims, aligns, converts file format, sorts, and counts the RNA-Seq data.

```
#!/bin/bash
#SBATCH --job-name=ngs5  # Job name
#SBATCH --mail-type=END,FAIL # Mail events (NONE, BEGIN, END, FAIL, ALL)
#SBATCH --mail-user=shaleigh.smith@nyulangone.org # Where to send mail
#SBATCH --ntasks=4 # Run on a single CPU
#SBATCH --mem=32gb # Job memory request
#SBATCH --time=10:00:00 # Time limit hrs:min:sec
#SBATCH --ntasks=4
#SBATCH --output=/gpfs/scratch/sas1531/ngs5_coursework/ngs5_%j.log # Standard output and error
log
#SBATCH -p cpu_short

module load fastqc/0.11.7
module load trimgalore/0.5.0
module load python/cpu/2.7.15-ES ### CutAdapt is hidden in here
module load bbmap/38.25
module load samtools/1.9
module load subread/1.6.3


trim_galore --q 30 --phred33 --paired -o ./ --fastqc ./assignment/${1}.fastq.gz
./assignment/${2}.fastq.gz

bbmap.sh \
-Xmx26G \
ref=/gpfs/scratch/sas1531/hg38/Homo_sapiens/UCSC/hg38/Sequence/WholeGenomeFasta/genome.fa \
in=./${1}_val_1.fq.gz \
in2=./${2}_val_2.fq.gz \
outm=./${1}_R2.sam \
minid=0.90 \
ambiguous=random \
nodisk \

samtools view -S -b ${1}_R2.sam > ${1}_R2.bam
samtools sort ${1}_R2.bam -o ${1}_R2_sorted.bam

featureCounts -s 2 -p -B -C -P --ignoreDup --primary -a
/gpfs/scratch/sas1531/hg38/Homo_sapiens/UCSC/hg38/Annotation/Genes.gencode/genes.gtf -g
gene_id -o ${1}_R2_feature_counts ./first_output/${1}_R2_sorted.bam
```

Code

The code below is the submitter script for the first script:

```
#!/bin/bash
#SBATCH --job-name=job_submitter
#SBATCH --nodes=1
#SBATCH --mem=200MB
#SBATCH --time=1:00:00
#SBATCH --error=job_sub_error.txt
```

```
#SBATCH --output=job_sub_stdout.txt

sbatch --array=1 ngs5_script.sh  EV1_R1 EV1_R2
sbatch --array=1 ngs5_script.sh  EV2_R1 EV2_R2
sbatch --array=1 ngs5_script.sh  EV3_R1 EV3_R2
sbatch --array=1 ngs5_script.sh  EV4_R1 EV4_R2
sbatch --array=1 ngs5_script.sh  EV5_R1 EV5_R2
sbatch --array=1 ngs5_script.sh  EV6_R1 EV6_R2
sbatch --array=1 ngs5_script.sh  mVLT1_R1 mVLT1_R2
sbatch --array=1 ngs5_script.sh  mVLT2_R1 mVLT2_R2
sbatch --array=1 ngs5_script.sh  mVLT3_R1 mVLT3_R2
```

Code

The code below includes the DESeq2, edgeR, and comparison analysis without and with batch correction, respectively, in R.

```
# Shaleigh Smith
# NGS Coursework 5

### Import Libraries
library(tidyverse)
library(plyr)
library(dplyr)
library(DESeq2)
library(edgeR)
library(purrr)
library(ComplexHeatmap)
library(circlize)
library(ggplot2)
library(ggfortify)
library(vsn)
library(AnnotationDbi)
library(org.Hs.eg.db)
library(genefilter)
library(biomaRt)
library(data.table)
library(IHW)
library(gridExtra)

### Set Working directory
setwd("/Users/sha/Desktop/NGS_Informatics/NGS_courswork/ngs_coursework5_shaleigh_smith/")

### Import & clean gene count files
count_1 <- read.table("EV1_R1_R2_feature_counts", header=TRUE)
colnames(count_1)[c(1, 7)] <- c("gene_id", "EV1")
count_1 <- dplyr::select(count_1, gene_id, EV1)
count_1$gene_id <- substring(count_1$gene_id, 1, 15)

count_2 <- read.table("EV2_R1_R2_feature_counts", header=TRUE)
colnames(count_2)[c(1, 7)] <- c("gene_id", "EV2")
count_2 <- dplyr::select(count_2, gene_id, EV2)
count_2$gene_id <- substring(count_2$gene_id, 1, 15)

count_3 <- read.table("EV3_R1_R2_feature_counts", header=TRUE)
colnames(count_3)[c(1, 7)] <- c("gene_id", "EV3")
count_3 <- dplyr::select(count_3, gene_id, EV3)
```

```
count_3$gene_id <- substring(count_3$gene_id, 1, 15)

count_4 <- read.table("EV4_R1_R2_feature_counts", header=TRUE)
colnames(count_4)[c(1, 7)] <- c("gene_id", "EV4")
count_4 <- dplyr::select(count_4, gene_id, EV4)
count_4$gene_id <- substring(count_4$gene_id, 1, 15)

count_5 <- read.table("EV5_R1_R2_feature_counts", header=TRUE)
colnames(count_5)[c(1, 7)] <- c("gene_id", "EV5")
count_5 <- dplyr::select(count_5, gene_id, EV5)
count_5$gene_id <- substring(count_5$gene_id, 1, 15)

count_6 <- read.table("EV6_R1_R2_feature_counts", header=TRUE)
colnames(count_6)[c(1, 7)] <- c("gene_id", "EV6")
count_6 <- dplyr::select(count_6, gene_id, EV6)
count_6$gene_id <- substring(count_6$gene_id, 1, 15)

count_7 <- read.table("mVLT1_R1_R2_feature_counts", header=TRUE)
colnames(count_7)[c(1, 7)] <- c("gene_id", "mVLT1")
count_7 <- dplyr::select(count_7, gene_id, mVLT1)
count_7$gene_id <- substring(count_7$gene_id, 1, 15)

count_8 <- read.table("mVLT2_R1_R2_feature_counts", header=TRUE)
colnames(count_8)[c(1, 7)] <- c("gene_id", "mVLT2")
count_8 <- dplyr::select(count_8, gene_id, mVLT2)
count_8$gene_id <- substring(count_8$gene_id, 1, 15)

count_9 <- read.table("mVLT3_R1_R2_feature_counts", header=TRUE)
colnames(count_9)[c(1, 7)] <- c("gene_id", "mVLT3")
count_9 <- dplyr::select(count_9, gene_id, mVLT3)
count_9$gene_id <- substring(count_9$gene_id, 1, 15)

### Join all gene count files into one data frame
feature_counts <- join_all(list(count_1, count_2, count_3, count_4, count_5, count_6,
                                count_7, count_8, count_9), by = "gene_id", type = "full")
# Write out table for later analysis
write.table(feature_counts, file = "feature_counts.txt", quote=FALSE, sep='\t', row.names =
FALSE)



############ Analysis WITHOUT Batch Effect Correction ############

# The first analysis was conducted without batch correction.
# The control 1, control 2 and expression vectors were compared to each other
# This displayed the clear batch effect of control 1, which was done at a
# different time than control 2 and expression

#######
### DeSeq2 Pipeline:

# Read in feature counts
deseq_counts <- read.table("feature_counts.txt", header=TRUE, row.names=1)
as.tibble(deseq_counts)

# Create DeSeq dataset
samples <- data.frame(row.names = c("EV1", "EV2", "EV3", "EV4", "EV5",
                                    "EV6", "mVLT1", "mVLT2", "mVLT3"),
                      vector = as.factor(c(rep("control_1", 3),
                                           rep("control_2", 3),
                                           rep("expression", 3))))
```

```r
deseq_df <- DESeqDataSetFromMatrix(countData = deseq_counts, colData = samples,
                                   design = ~ vector)

# Remove genes that do not have counts greater than 2 in at least 2 of the datasets (columns)
deseq_df <- deseq_df[rowSums(counts(deseq_df) >= 2) >= 2]

# Confirm that all samples are labelled correctly
as.data.frame(colData(deseq_df))

### Exploratory data analysis of DESeq matrix with transformation

# Estimate size factors
# The size factor is the median ratio of the sample over a pseudosample: for each gene, the
geometric mean of all samples
deseq_eda <- estimateSizeFactors(deseq_df)

# Apply regularized-logarithm transformation
rld <- rlog(deseq_eda, blind = FALSE)

# Apply variance stabilizing transformation
vsd <- vst(deseq_eda, blind = FALSE)

# Create new data frame three normalization methods for all samples
deseq_eda <- bind_rows(
  as_data_frame(log2(counts(deseq_eda, normalized=TRUE)[, (1:9)])) %>%
    mutate(transform = "log2(x + 1)"),
  as_data_frame(assay(vsd)[, (1:9)]) %>% mutate(transform = "vst"),
  as_data_frame(assay(rld)[, (1:9)]) %>% mutate(transform = "rlog"))

# Compare transformation visually
ggplot(deseq_eda, aes(x = EV1, y = EV2)) +
  geom_bin2d(bins = 40) +
  coord_fixed() +
  facet_grid( . ~ transform) +
  scale_fill_continuous(limits = c(0, 2000), breaks = seq(0, 2000, by = 500)) +
  ggtitle("EV1 vs. EV2") + theme(plot.title = element_text(hjust = 0.5))

ggplot(deseq_eda, aes(x = EV1, y = EV4)) +
  geom_bin2d(bins = 40) +
  coord_fixed() +
  facet_grid( . ~ transform) +
  scale_fill_continuous(limits = c(0, 2000), breaks = seq(0, 2000, by = 500)) +
  ggtitle("EV1 vs. EV4") + theme(plot.title = element_text(hjust = 0.5))

ggplot(deseq_eda, aes(x = EV1, y = mVLT1)) +
  geom_bin2d(bins = 40) +
  coord_fixed() +
  facet_grid( . ~ transform) +
  scale_fill_continuous(limits = c(0, 2000), breaks = seq(0, 2000, by = 500)) +
  ggtitle("EV1 vs. mVLT1") + theme(plot.title = element_text(hjust = 0.5))

ggplot(deseq_eda, aes(x = EV4, y = mVLT1)) +
  geom_bin2d(bins = 40) +
  coord_fixed() +
  facet_grid( . ~ transform) +
  scale_fill_continuous(limits = c(0, 2000), breaks = seq(0, 2000, by = 500)) +
  ggtitle("EV4 vs. mVLT1") + theme(plot.title = element_text(hjust = 0.5))


### Exploratory data analysis of DESeq matrix with sample comparison
# This will asses overall similarity between samples
# As shown in the above plots this similarity might not be as expected

# Calculate the euclidean distance between samples
deseq_distance <- dist(t(assay(rld)))
deseq_distance

# Create annotation for the heatmap
deseq_annotation <- as.data.frame(as.factor(c(rep("control_1", 3),
                                              rep("control_2", 3),
                                              rep("expression", 3))))
colnames(deseq_annotation)[1] <- c("Vector")
deseq_ha <- HeatmapAnnotation(df = deseq_annotation,
                              col = list(Vector = c("control_1" = "red4",
                                                    "control_2" = "darkblue",
                                                    "expression" = "goldenrod2")))

# Make dataframe a matrix for heatmpap
deseq_heatmap <- as.matrix(deseq_distance)

# Visualize with heatmap
deseq_heat <- Heatmap(deseq_heatmap,
                      top_annotation = deseq_ha,
                      show_column_names = F,
                      show_row_names = T,
                      cluster_rows = T,
                      cluster_columns = T,
                      clustering_method_columns = 'complete',
                      clustering_method_rows = "complete",
                      row_names_gp = gpar(fontsize = 8),
                      top_annotation_height = unit(1, "cm"),
                      heatmap_legend_param = list(title = "Distance"),
                      col = colorRamp2(c(0, 75), c("darkorchid4", "white")))

deseq_heat
# It looks like the second control batch is very similar to the expression vector

### Compare with PCA
# visualize screeplot and autoplotted pca for reference
deseq_pca <-prcomp(t(assay(rld)))
screeplot(deseq_pca, type='lines')
autoplot(deseq_pca)

# Create PCA object to be plotted with ggplot
plot_pca <- plotPCA(rld, intgroup = c("vector"), returnData = TRUE)

# Calculate and round variance for plotting
percentVar <- round(100 * attr(plot_pca, "percentVar"))

# Plot PCA
deseq_pca_1 <- ggplot(plot_pca, aes(x = PC1, y = PC2, color = vector)) +
  geom_point(size = 2.5) +
  xlab(paste0("PC1: ", percentVar[1], "% variance")) +
  ylab(paste0("PC2: ", percentVar[2], "% variance")) +
  coord_fixed() +
  scale_color_manual(values = c("red4", "darkblue", "goldenrod2")) +
  ggtitle("DESeq2 PCA: Not Batch Corrected") +
  theme(plot.title = element_text(hjust = 0.5))

tiff("deseq_pca_not_batch.tiff", units="in", width=7, height=5, res=300)
deseq_pca_1
dev.off()
```

```
# Make sure that contol_1 is the first level in the vector factor
# This is to ensure the log2 fold change is calculated over the control (when results are
called at random)
deseq_df$vector <- relevel(deseq_df$vector, "control_1")

# Check to insure all samples are correct
as.data.frame(colData(deseq_df))

# Run the DESeq analysis against raw counts
deseq_analysis <- DESeq(deseq_df)
results(deseq_analysis)

# Call results
deseq_results_cont1_exp <- results(deseq_analysis,
                                   contrast = c("vector", "expression", "control_1"))
deseq_results_cont1_exp

deseq_results_cont2_exp <- results(deseq_analysis,
                                   contrast = c("vector", "expression", "control_2"))
deseq_results_cont2_exp

deseq_results_cont1_cont2 <- results(deseq_analysis,
                                     contrast = c("vector", "control_2", "control_1"))
deseq_results_cont1_cont2

# Review Comparisons
mcols(deseq_results_cont1_exp, use.names = TRUE)
mcols(deseq_results_cont2_exp, use.names = TRUE)
mcols(deseq_results_cont1_cont2, use.names = TRUE)

### Review Summary

summary(deseq_results_cont1_exp, alpha = 0.01)
summary(deseq_results_cont2_exp, alpha = 0.01)
summary(deseq_results_cont1_cont2, alpha = 0.01)

### Plot p-values for distribution check

# calculate row means across vectors
mean_cont1 <-  rowMeans(counts(deseq_analysis, normalized=TRUE)[, deseq_analysis$vector ==
"control_1"])
mean_cont2 <-  rowMeans(counts(deseq_analysis, normalized=TRUE)[, deseq_analysis$vector ==
"control_2"])
mean_exp <-  rowMeans(counts(deseq_analysis, normalized=TRUE)[, deseq_analysis$vector ==
"expression"])


### Control 1 vs. expression
# Log Fold change
deseq_lfc_1 <- lfcShrink(deseq_analysis, contrast = c("vector", "expression", "control_1"))
# add row names to log fold change data frame
deseq_lfc_1 <-  cbind(as.data.frame(deseq_lfc_1), mean_exp, mean_cont1)
# Map gene names and ids
deseq_lfc_1$symbol <- mapIds(org.Hs.eg.db, keys=row.names(deseq_lfc_1),
                             column="SYMBOL", keytype="ENSEMBL", multiVals="first")
deseq_lfc_1$entrez <- mapIds(org.Hs.eg.db, keys=row.names(deseq_lfc_1),
                             column="ENTREZID", keytype="ENSEMBL", multiVals="first")
# Plot histogram to review distribution
ggplot(data = deseq_lfc_1, mapping = aes(x = pvalue)) + geom_histogram(binwidth = 0.01)
ggplot(data = deseq_lfc_1, mapping = aes(x = padj)) + geom_histogram(binwidth = 0.01)
```

```r
### Control 2 vs. expression
# Log Fold change
deseq_lfc_2 <- lfcShrink(deseq_analysis, contrast = c("vector", "expression", "control_2"))
# add row names to log fold change data frame
deseq_lfc_2 <-  cbind(as.data.frame(deseq_lfc_2), mean_exp, mean_cont2)
# Map gene names and ids
deseq_lfc_2$symbol <- mapIds(org.Hs.eg.db, keys=row.names(deseq_lfc_2),
                              column="SYMBOL", keytype="ENSEMBL", multiVals="first")
deseq_lfc_2$entrez <- mapIds(org.Hs.eg.db, keys=row.names(deseq_lfc_2),
                              column="ENTREZID", keytype="ENSEMBL", multiVals="first")
# Plot histogram to review distribution
ggplot(data = deseq_lfc_2, mapping = aes(x = pvalue)) + geom_histogram(binwidth = 0.01)
ggplot(data = deseq_lfc_2, mapping = aes(x = padj)) + geom_histogram(binwidth = 0.01)

### This shows that control 2 is very similar to the Expression group***


### Control 1 vs. Control 2
# Log Fold change
deseq_lfc_3 <- lfcShrink(deseq_analysis, contrast = c("vector", "control_2", "control_1"))
# add row names to log fold change data frame
deseq_lfc_3 <-  cbind(as.data.frame(deseq_lfc_3), mean_cont1, mean_cont2)
# Map gene names and ids
deseq_lfc_3$symbol <- mapIds(org.Hs.eg.db, keys=row.names(deseq_lfc_3),
                              column="SYMBOL", keytype="ENSEMBL", multiVals="first")
deseq_lfc_3$entrez <- mapIds(org.Hs.eg.db, keys=row.names(deseq_lfc_3),
                              column="ENTREZID", keytype="ENSEMBL", multiVals="first")
# Plot histogram to review distribution
ggplot(data = deseq_lfc_3, mapping = aes(x = pvalue)) + geom_histogram(binwidth = 0.01)
ggplot(data = deseq_lfc_3, mapping = aes(x = padj)) + geom_histogram(binwidth = 0.01)

### This also supports that control 2 is very similar to the expression group ***

#### Volcano Plots

# Create up, none, down labels for color then plot

# Control 1 vs. Expression
deseq_lfc_1 <- deseq_lfc_1 %>%
  mutate(threshold = ifelse((log2FoldChange >= 0 & padj < 0.01), "up_sig",
                            ifelse((log2FoldChange <= 0 & padj < 0.01) ,
                                   "down_sig", "not_sig")))
deseq_lfc_1$threshold[is.na(deseq_lfc_1$threshold)] <- "not_sig"

des_vol_1 <- ggplot(deseq_lfc_1, aes(x = log2FoldChange, y = -log10(pvalue))) +
  geom_point(aes(colour = threshold), size=1) +
  ggtitle("Control 1 vs. Expression with DESeq2") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ylim(0,50) + xlim(-6,6) +
  scale_colour_manual(values=c("blue", "grey", "red3"),
                      name="Threshold 0.01",
                      breaks=c("down_sig", "not_sig", "up_sig"),
                      labels=c("Down", "None", "Up"))

# Control 2 vs. Expression
deseq_lfc_2 <- deseq_lfc_2 %>%
  mutate(threshold = ifelse((log2FoldChange >= 0 & padj < 0.01), "up_sig",
                            ifelse((log2FoldChange <= 0 & padj < 0.01) ,
                                   "down_sig", "not_sig")))
```

```
deseq_lfc_2$threshold[is.na(deseq_lfc_2$threshold)] <- "not_sig"

des_vol_2 <- ggplot(deseq_lfc_2, aes(x = log2FoldChange, y = -log10(pvalue))) +
  geom_point(aes(colour = threshold), size=1) +
  ggtitle("Control 2 vs. Expression with DESeq2") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ylim(0,50) + xlim(-6,6) +
  scale_colour_manual(values=c("blue", "grey", "red3"),
                      name="Threshold 0.01",
                      breaks=c("down_sig", "not_sig", "up_sig"),
                      labels=c("Down", "None", "Up"))

# Control 1 vs. Control 2
deseq_lfc_3 <- deseq_lfc_3 %>%
  mutate(threshold = ifelse((log2FoldChange >= 0 & padj < 0.01), "up_sig",
                            ifelse((log2FoldChange <= 0 & padj < 0.01) ,
                                   "down_sig", "not_sig")))
deseq_lfc_3$threshold[is.na(deseq_lfc_3$threshold)] <- "not_sig"

des_vol_3 <- ggplot(deseq_lfc_3, aes(x = log2FoldChange, y = -log10(pvalue))) +
  geom_point(aes(colour = threshold), size=1) +
  ggtitle("Control 1 vs. Control 2 with DESeq2") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ylim(0,50) + xlim(-6,6) +
  scale_colour_manual(values=c("blue", "grey", "red3"),
                      name="Threshold 0.01",
                      breaks=c("down_sig", "not_sig", "up_sig"),
                      labels=c("Down", "None", "Up"))

#### Heatmaps with top variable genes

# Control 1 vs. Expression
# Transform with regularized log
rld_1 <- rlog(deseq_analysis)
#assay(rld_1)
# Order genes then filter for the top 25
top_genes_1 <- head(order(rowMeans(assay(rld_1)), decreasing = TRUE), 30)
as.tibble(top_genes_1)
deseq_genes <- assay(rld_1)[top_genes_1, ]
# Calculate distances from the mean for clearer heatmap
deseq_genes <- (deseq_genes - rowMeans(deseq_genes))

# Connect to ensemble database and label gene ids with gene symbol
mart <- useMart("ENSEMBL_MART_ENSEMBL","hsapiens_gene_ensembl")
gene_symbol <- getBM(c("hgnc_symbol","ensembl_gene_id"), "ensembl_gene_id",
row.names(deseq_genes), mart)

row.names(deseq_genes)[match(gene_symbol[,2], row.names(deseq_genes))] <- gene_symbol[,1]

as.tibble(deseq_genes)

# Create Annotation
vector_annotation <- as.data.frame(colData(rld_1)[, c("vector")])
colnames(vector_annotation)[1] <- "Vector"
vector_ha <- HeatmapAnnotation(df = vector_annotation,
                               col = list(Vector = c("control_1" = "red4",
                                                     "control_2" = "darkblue",
                                                     "expression" = "goldenrod2")))
# Create Heatmap
deseq_heat_1 <- Heatmap(deseq_genes,
                        top_annotation = vector_ha,
                        show_column_names = F,
                        show_row_names = T,
                        cluster_rows = T,
                        cluster_columns = T,
                        clustering_method_columns = 'complete',
                        clustering_method_rows = "complete",
                        row_names_gp = gpar(fontsize = 8),
                        top_annotation_height = unit(1, "cm"),
                        heatmap_legend_param = list(title = "Distance"),
                        col = colorRamp2(c(-1, 0, 1), c("red3", "white", "blue")),
                        column_title = "DESeq2 Control vs. Expression: Not Batch Corrected")

deseq_heat_1

tiff("deseq_head_not_batch.tiff", units="in", width=7, height=5, res=300)
deseq_heat_1
dev.off()

# Add column for upregulated vs. downregulated
deseq_lfc_1$expression <- NA
deseq_lfc_1$expression[deseq_lfc_1$log2FoldChange > 0 ] <- "up"
deseq_lfc_1$expression[deseq_lfc_1$log2FoldChange < 0 ] <- "down"
setDT(deseq_lfc_1, keep.rownames = TRUE)[]
colnames(deseq_lfc_1)[1] <- "gene_id"
as.tibble(deseq_lfc_1)

deseq_lfc_2$expression <- NA
deseq_lfc_2$expression[deseq_lfc_2$log2FoldChange > 0 ] <- "up"
deseq_lfc_2$expression[deseq_lfc_2$log2FoldChange < 0 ] <- "down"
setDT(deseq_lfc_2, keep.rownames = TRUE)[]
colnames(deseq_lfc_2)[1] <- "gene_id"
as.tibble(deseq_lfc_2)

deseq_lfc_3$expression <- NA
deseq_lfc_3$expression[deseq_lfc_3$log2FoldChange > 0 ] <- "up"
deseq_lfc_3$expression[deseq_lfc_3$log2FoldChange < 0 ] <- "down"
setDT(deseq_lfc_3, keep.rownames = TRUE)[]
colnames(deseq_lfc_3)[1] <- "gene_id"
as.tibble(deseq_lfc_3)

# Filter out columns for significantly upregulated and downregulated genes
deseq_lfc_1_final <- dplyr::select(deseq_lfc_1, gene_id, symbol, log2FoldChange, padj,
expression)
deseq_lfc_1_final <- filter(deseq_lfc_1_final, padj < 0.01)
deseq_lfc_1_final <- deseq_lfc_1_final[order(deseq_lfc_1_final$padj),]
as.tibble(deseq_lfc_1_final)

deseq_lfc_2_final <- dplyr::select(deseq_lfc_2, gene_id, symbol, log2FoldChange, padj,
expression)
deseq_lfc_2_final <- filter(deseq_lfc_2_final, padj < 0.01)
deseq_lfc_2_final <- deseq_lfc_2_final[order(deseq_lfc_2_final$padj),]
as.tibble(deseq_lfc_2_final)

deseq_lfc_3_final <- dplyr::select(deseq_lfc_3, gene_id, symbol, log2FoldChange, padj,
expression)
deseq_lfc_3_final <- filter(deseq_lfc_3_final, padj < 0.01)
deseq_lfc_3_final <- deseq_lfc_3_final[order(deseq_lfc_3_final$padj),]
as.tibble(deseq_lfc_3_final)


# Convert gene ids to gene symbol
```

```
mart <- useMart("ENSEMBL_MART_ENSEMBL","hsapiens_gene_ensembl")
gene_symbol <- getBM(c("hgnc_symbol","ensembl_gene_id"), "ensembl_gene_id",
row.names(deseq_analysis), mart)

row.names(deseq_analysis)[match(gene_symbol[,2], row.names(deseq_analysis))] <-
gene_symbol[,1]

### Filter and order dataframes
# State whether they are upregulated or downregulated
# Filter out columns for significantly upregulated and downregulated genes

# Top 30 expressed genes (by p-value)
deseq_top_30 <- as.data.frame(results(deseq_analysis))
deseq_top_30 <- deseq_top_30[order(deseq_top_30$padj),]
deseq_top_30 <- head(deseq_top_30, 30)
setDT(deseq_top_30, keep.rownames = TRUE)[]
colnames(deseq_top_30)[1] <- "genes"

as.tibble(deseq_top_30)

deseq_top_30$expression <- NA
deseq_top_30$expression[deseq_top_30$log2FoldChange > 0 ] <- "up"
deseq_top_30$expression[deseq_top_30$log2FoldChange < 0 ] <- "down"
as.tibble(deseq_top_30)


#########
### edgeR Pipeline:

# Read in feature counts
edge_counts <- read.table("feature_counts.txt", header=TRUE, row.names=1)
as.tibble(edge_counts)

### Set up edgeR matrix
group <- c(1,1,1,2,2,2,3,3,3)
edge_df <- DGEList(counts = edge_counts, group = group, genes = rownames(edge_counts))
summary(edge_df)

# filter out lowly expressed genes using count-per-million (CPM)
# Re-calculate library size
filter_rows <- rowSums(cpm(edge_df) > 1) >= 2
edge_df_filter <- edge_df[filter_rows, , keep.lib.sizes=FALSE]
summary(edge_df_filter)

# Normalize RNA composition (accounts for library size)
edge_df_norm <- calcNormFactors(edge_df_filter,  method="TMM")
edge_df_norm

### Exploratory data analysis with DGEList object

# Plot MDS?
tiff("edge_mds_not_batch.tiff", units="in", width=7, height=5, res=300)
plotMDS(edge_df_norm, method="bcv", col=as.numeric(edge_df_norm$samples$group),
        main=("edgeR BCV: Not Batch Corrected"))
dev.off()

# Estimate Dispersion (common and tagwise in one run)
edge_disp <- estimateDisp(edge_df_norm)
edge_disp

# Plot BCV
```

```
plotBCV(edge_disp)

### Heatmap with top genes
logcpm <- cpm(edge_df_norm, prior.count=2, log=TRUE)
logcpm

top_genes_2 <- head(order(rowMeans(logcpm), decreasing = TRUE), 30)
as.tibble(top_genes_2)
edge_genes <- logcpm[top_genes_2, ]
edge_genes
# Calculate distances from the mean for clearer heatmap
edge_genes <- (edge_genes - rowMeans(edge_genes))

# Connect to ensemble database and label gene ids with gene symbol
mart <- useMart("ENSEMBL_MART_ENSEMBL","hsapiens_gene_ensembl")
gene_symbol <- getBM(c("hgnc_symbol","ensembl_gene_id"), "ensembl_gene_id",
row.names(edge_genes), mart)
row.names(edge_genes)[match(gene_symbol[,2], row.names(edge_genes))] <- gene_symbol[,1]

# Creat annotation
edge_annotation <- as.data.frame(as.factor(c(rep("control_1", 3),
                                             rep("control_2", 3),
                                             rep("expression", 3))))
colnames(edge_annotation)[1] <- c("Vector")
edge_ha <- HeatmapAnnotation(df = edge_annotation, col = list(Vector = c("control_1" = "red4",
"control_2" = "darkblue","expression" = "goldenrod2")))

# Create heatmap
edge_heat_1 <- Heatmap(edge_genes,
                       top_annotation = edge_ha,
                       show_column_names = F,
                       show_row_names = T,
                       cluster_rows = T,
                       cluster_columns = T,
                       clustering_method_columns = 'complete',
                       clustering_method_rows = "complete",
                       row_names_gp = gpar(fontsize = 8),
                       top_annotation_height = unit(1, "cm"),
                       heatmap_legend_param = list(title = "Distance"),
                       col = colorRamp2(c(-1, 0, 1), c("red3", "white", "blue")),
                       column_title = "edgeR Control vs. Expression: Not Batch Corrected")

edge_heat_1

tiff("edge_heat_not_batch.tiff", units="in", width=7, height=5, res=300)
edge_heat_1
dev.off()

### Differential gene expression Classic Approach

# Between all
edge_exact <- exactTest(edge_disp)
edge_top_30 <- as.data.frame(topTags(edge_exact, n = 30))
edge_top_30
edge_top <- topTags(edge_exact, n = nrow(edge_exact))
edge_top

# Control 1 vs. Expression
edge_exact_1 <- exactTest(edge_disp, pair = c("1", "3"))
edge_top_1 <- topTags(edge_exact_1, n = nrow(edge_exact_1))
edge_top_1
```

```r
# Control 2 vs. Expression
edge_exact_2 <- exactTest(edge_disp, pair = c("2", "3"))
edge_top_2 <- topTags(edge_exact_2, n = nrow(edge_exact_2))
edge_top_2

# Control 1 vs. Control 2
edge_exact_3 <- exactTest(edge_disp, pair = c("1", "2"))
edge_top_3 <- topTags(edge_exact_3, n = nrow(edge_exact_3))
edge_top_3

#### Volcano Plots

# Create up, none, down labels for color then plot

# Control 1 vs. Expression
edge_top_1_volcano <- as.data.frame(edge_top_1) %>%
  mutate(threshold = ifelse((logFC >= 0 & FDR < 0.01), "up_sig",
                            ifelse((logFC <= 0 & FDR < 0.01) ,
                                   "down_sig", "not_sig")))
edge_top_1_volcano$threshold[is.na(edge_top_1_volcano$threshold)] <- "not_sig"

edge_vol_1 <- ggplot(edge_top_1_volcano, aes(x = logFC, y = -log10(PValue))) +
  geom_point(aes(colour = threshold), size=1) +
  ggtitle("Control 1 vs. Expression with edgeR") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ylim(0,50) + xlim(-6,6) +
  scale_colour_manual(values=c("blue", "grey", "red3"),
                      name="Threshold 0.01",
                      breaks=c("down_sig", "not_sig", "up_sig"),
                      labels=c("Down", "None", "Up"))
edge_vol_1

# Control 2 vs. Expression
edge_top_2_volcano <- as.data.frame(edge_top_2) %>%
  mutate(threshold = ifelse((logFC >= 0 & FDR < 0.01), "up_sig",
                            ifelse((logFC <= 0 & FDR < 0.01) ,
                                   "down_sig", "not_sig")))
edge_top_2_volcano$threshold[is.na(edge_top_2_volcano$threshold)] <- "not_sig"

edge_vol_2 <- ggplot(edge_top_2_volcano, aes(x = logFC, y = -log10(PValue))) +
  geom_point(aes(colour = threshold), size=1) +
  ggtitle("Control 2 vs. Expression with edgeR") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ylim(0,50) + xlim(-6,6) +
  scale_colour_manual(values=c("blue", "grey", "red3"),
                      name="Threshold 0.01",
                      breaks=c("down_sig", "not_sig", "up_sig"),
                      labels=c("Down", "None", "Up"))

# Control 1 vs. Control 2
edge_top_3_volcano <- as.data.frame(edge_top_3) %>%
  mutate(threshold = ifelse((logFC >= 0 & FDR < 0.01), "up_sig",
                            ifelse((logFC <= 0 & FDR < 0.01) ,
                                   "down_sig", "not_sig")))
edge_top_3_volcano$threshold[is.na(edge_top_3_volcano$threshold)] <- "not_sig"

edge_vol_3 <- ggplot(edge_top_3_volcano, aes(x = logFC, y = -log10(PValue))) +
  geom_point(aes(colour = threshold), size=1) +
  ggtitle("Control 1 vs. Control 2 with edgeR") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ylim(0,50) + xlim(-6,6) +
  scale_colour_manual(values=c("blue", "grey", "red3"),
                      name="Threshold 0.01",
                      breaks=c("down_sig", "not_sig", "up_sig"),
                      labels=c("Down", "None", "Up"))

# Convert gene ids to gene symbol
mart <- useMart("ENSEMBL_MART_ENSEMBL","hsapiens_gene_ensembl")
gene_symbol_edge <- getBM(c("hgnc_symbol","ensembl_gene_id"), "ensembl_gene_id",
row.names(edge_top_1), mart)
colnames(gene_symbol_edge)[1:2] <- c("gene_symbol", "genes")

### Filter and order dataframes
# State whether they are upregulated or downregulated
# Filter out columns for significantly upregulated and downregulated genes

# Control 1 vs. Expression
edge_top_1_gene <- as.data.frame(edge_top_1)
edge_top_1_gene <- join(gene_symbol_edge, edge_top_1_gene, type = "full", by = "genes")
edge_top_1_gene$expression <- NA
edge_top_1_gene$expression[edge_top_1_gene$logFC > 0 ] <- "up"
edge_top_1_gene$expression[edge_top_1_gene$logFC < 0 ] <- "down"
edge_top_1_gene <- filter(edge_top_1_gene, FDR < 0.01)
edge_top_1_gene <- edge_top_1_gene[order(edge_top_1_gene$FDR),]
edge_top_1_gene

# Control 2 vs. Expression
edge_top_2_gene <- as.data.frame(edge_top_2)
edge_top_2_gene <- join(gene_symbol_edge, edge_top_2_gene, type = "full", by = "genes")
edge_top_2_gene$expression <- NA
edge_top_2_gene$expression[edge_top_2_gene$logFC > 0 ] <- "up"
edge_top_2_gene$expression[edge_top_2_gene$logFC < 0 ] <- "down"
edge_top_2_gene <- filter(edge_top_2_gene, FDR < 0.01)
edge_top_2_gene <- edge_top_2_gene[order(edge_top_2_gene$FDR),]
edge_top_2_gene

# Control 3 vs. Control 2
edge_top_3_gene <- as.data.frame(edge_top_3)
edge_top_3_gene <- join(gene_symbol_edge, edge_top_3_gene, type = "full", by = "genes")
edge_top_3_gene$expression <- NA
edge_top_3_gene$expression[edge_top_3_gene$logFC > 0 ] <- "up"
edge_top_3_gene$expression[edge_top_3_gene$logFC < 0 ] <- "down"
edge_top_3_gene <- filter(edge_top_3_gene, FDR < 0.01)
edge_top_3_gene <- edge_top_3_gene[order(edge_top_3_gene$FDR),]
edge_top_3_gene

# Top 30 EdgeR

edge_top_30 <- as.data.frame(edge_top_30)
edge_top_30 <- join(gene_symbol_edge, edge_top_30, type = "full", by = "genes")
edge_top_30$expression <- NA
edge_top_30$expression[edge_top_30$logFC > 0 ] <- "up"
edge_top_30$expression[edge_top_30$logFC < 0 ] <- "down"
edge_top_30 <- filter(edge_top_30, FDR < 0.01)
edge_top_30 <- edge_top_30[order(edge_top_30$FDR),]
edge_top_30 <- head(edge_top_30, 30)
#edge_top_30 <- dplyr::select(edge_top_30, -rn)
edge_top_30


#######
```

```
### DESeq2 and edgeR Comparison

# Compare top 30 from all
edge_30_list <- as.vector(edge_top_30$genes)
deseq_30_list <- as.vector(deseq_top_30$genes)
length(intersect(edge_30_list, deseq_30_list)) #3

# Compare Control 1 vs. Expression
# All significant
edge_top_1_list <- as.vector(edge_top_1_gene$genes) #6286
deseq_lfc_1_list <- as.vector(deseq_lfc_1_final$gene_id) #4726
length(intersect(edge_top_1_list, deseq_lfc_1_list)) #4335
# Upregulated
edge_up_1_list <- edge_top_1_gene[edge_top_1_gene$expression == "up",] # 2334
nrow(edge_up_1_list)
edge_up_1_list <- as.vector(edge_up_1_list$genes)
deseq_up_1_list <- deseq_lfc_1_final[deseq_lfc_1_final$expression == "up",] # 2635
nrow(deseq_up_1_list)
deseq_up_1_list <- as.vector(deseq_up_1_list$gene_id)
length(intersect(edge_up_1_list, deseq_up_1_list)) # 2245
# Downregulated
edge_down_1_list <- edge_top_1_gene[edge_top_1_gene$expression == "down",] # 3952
nrow(edge_down_1_list)
edge_down_1_list <- as.vector(edge_down_1_list$genes)
deseq_down_1_list <- deseq_lfc_1_final[deseq_lfc_1_final$expression == "down",] # 2091
nrow(deseq_down_1_list)
deseq_down_1_list <- as.vector(deseq_down_1_list$gene_id)
length(intersect(edge_down_1_list, deseq_down_1_list)) # 2090

# Compare Control 2 vs. Expression
# All significant
edge_top_2_list <- as.vector(edge_top_2_gene$genes) #110
deseq_lfc_2_list <- as.vector(deseq_lfc_2_final$gene_id) #101
length(intersect(edge_top_2_list, deseq_lfc_2_list)) #84
# Upregulated
edge_up_2_list <- edge_top_2_gene[edge_top_2_gene$expression == "up",] # 66
nrow(edge_up_2_list)
edge_up_2_list <- as.vector(edge_up_2_list$genes)
deseq_up_2_list <- deseq_lfc_2_final[deseq_lfc_2_final$expression == "up",] # 56
nrow(deseq_up_2_list)
deseq_up_2_list <- as.vector(deseq_up_2_list$gene_id)
length(intersect(edge_up_2_list, deseq_up_2_list)) # 53
# Downregulated
edge_down_2_list <- edge_top_2_gene[edge_top_2_gene$expression == "down",] # 44
nrow(edge_down_2_list)
edge_down_2_list <- as.vector(edge_down_2_list$genes)
deseq_down_2_list <- deseq_lfc_2_final[deseq_lfc_2_final$expression == "down",] # 45
nrow(deseq_down_2_list)
deseq_down_2_list <- as.vector(deseq_down_2_list$gene_id)
length(intersect(edge_down_2_list, deseq_down_2_list)) # 32


# Compare Control 1 vs. Control 2
# All significant
edge_top_3_list <- as.vector(edge_top_3_gene$genes) #6262
deseq_lfc_3_list <- as.vector(deseq_lfc_3_final$gene_id) #4729
length(intersect(edge_top_3_list, deseq_lfc_3_list)) #4307
# Upregulated
edge_up_3_list <- edge_top_3_gene[edge_top_3_gene$expression == "up",] # 2249
nrow(edge_up_3_list)
edge_up_3_list <- as.vector(edge_up_3_list$genes)
```

```
deseq_up_3_list <- deseq_lfc_3_final[deseq_lfc_3_final$expression == "up",] # 2613
nrow(deseq_up_3_list)
deseq_up_3_list <- as.vector(deseq_up_3_list$gene_id)
length(intersect(edge_up_3_list, deseq_up_3_list)) # 2191
# Downregulated
edge_down_3_list <- edge_top_3_gene[edge_top_3_gene$expression == "down",] # 4013
nrow(edge_down_3_list)
edge_down_3_list <- as.vector(edge_down_3_list$genes)
deseq_down_3_list <- deseq_lfc_3_final[deseq_lfc_3_final$expression == "down",] # 2116
nrow(deseq_down_3_list)
deseq_down_3_list <- as.vector(deseq_down_3_list$gene_id)
length(intersect(edge_down_3_list, deseq_down_3_list)) # 2116


### Compare Volcano Plots
vol_lay <- rbind(c(1,1,1,2,2,2,2),
                 c(1,1,1,2,2,2,2))

tiff("final_volcano_cont1_exp_not_batch.tiff", units="in", width=9, height=5, res=300)
grid.arrange(des_vol_1 + theme(legend.position = 'none',
                               title = element_text(size = 9),
                               axis.title = element_text(size =8),
                               axis.text = element_text(size = 7)),
             edge_vol_1+ theme(title = element_text(size = 9),
                               axis.title = element_text(size =8),
                               axis.text = element_text(size = 7)),
             ncol = 2, layout_matrix = vol_lay)
dev.off()

tiff("final_volcano_cont2_exp_not_batch.tiff", units="in", width=9, height=5, res=300)
grid.arrange(des_vol_2 + theme(legend.position = 'none',
                               title = element_text(size = 9),
                               axis.title = element_text(size =8),
                               axis.text = element_text(size = 7)),
             edge_vol_2+ theme(title = element_text(size = 9),
                               axis.title = element_text(size =8),
                               axis.text = element_text(size = 7)),
             ncol = 2, layout_matrix = vol_lay)
dev.off()

tiff("final_volcano_cont1_cont2_not_batch.tiff", units="in", width=9, height=5, res=300)
grid.arrange(des_vol_3 + theme(legend.position = 'none',
                               title = element_text(size = 9),
                               axis.title = element_text(size =8),
                               axis.text = element_text(size = 7)),
             edge_vol_3+ theme(title = element_text(size = 9),
                               axis.title = element_text(size =8),
                               axis.text = element_text(size = 7)),
             ncol = 2, layout_matrix = vol_lay)
dev.off()

### Compare Heat
deseq_heat_1
edge_heat_1
```

```r
############ Analysis WITH Batch Effect Correction ############

# This is the same analysis as above with the exception of batch correction.
# As a result the contrast is between all control and the expressoin vector.
# The batches are corrected.

#######
### Deseq with Combined Controls (batch effect Corrected) Pipeline

# Read in feature counts
deseq_batch_counts <- read.table("feature_counts.txt", header=TRUE, row.names=1)
as.tibble(deseq_batch_counts)

# Create DeSeq dataset
samples_batch <- data.frame(row.names = c("EV1", "EV2", "EV3", "EV4", "EV5",
                                          "EV6", "mVLT1", "mVLT2", "mVLT3"),
                            vector = as.factor(c(rep("control", 3),
                                                 rep("control", 3),
                                                 rep("expression", 3))),
                            batch = as.factor(c(rep("1", 3),
                                                rep("2", 3),
                                                rep("2", 3))))

deseq_batch_df <- DESeqDataSetFromMatrix(countData = deseq_batch_counts, colData =
samples_batch,
                                         design = ~ vector + batch)

# Remove genes that do not have counts greater than 2 in at least 2 of the datasets (columns)
deseq_batch_df <- deseq_batch_df[rowSums(counts(deseq_batch_df) >= 2) >= 2]

# Confirm that all samples are labelled correctly
as.data.frame(colData(deseq_batch_df))

### Exploratory data analysis of DESeq matrix with sample comparison
# This will asses overall similarity between samples
# As shown in the above plots this similarity might not be as expected

# Estimate size factors
# The size factor is the median ratio of the sample over a pseudosample: for each gene, the
geometric mean of all samples
deseq_batch_eda <- estimateSizeFactors(deseq_batch_df)

# Apply regularized-logarithm transformation
rld_batch <- rlog(deseq_batch_eda, blind = FALSE)

# Calculate the euclidean distance between samples
deseq_batch_distance <- dist(t(assay(rld_batch)))
deseq_batch_distance

# Create annotation for the heatmap
deseq_batch_annotation <-  samples_batch

deseq_batch_ha <- HeatmapAnnotation(df = deseq_batch_annotation,
                                    col = list(vector = c("control" = "red4",
                                                          "expression" = "darkblue"),
                                               batch = c("1" = "goldenrod2",
                                                         "2" = "darkgreen")))


# Make dataframe a matrix for heatmpap
deseq_batch_heatmap <- as.matrix(deseq_batch_distance)
```

```
# Visualize with heatmap
deseq_batch_heat <- Heatmap(deseq_batch_heatmap,
                            top_annotation = deseq_batch_ha,
                            show_column_names = F,
                            show_row_names = T,
                            cluster_rows = T,
                            cluster_columns = T,
                            clustering_method_columns = 'complete',
                            clustering_method_rows = "complete",
                            row_names_gp = gpar(fontsize = 8),
                            top_annotation_height = unit(1, "cm"),
                            heatmap_legend_param = list(title = "Distance"),
                            col = colorRamp2(c(0, 75), c("darkorchid4", "white")))

deseq_batch_heat
# It looks like the second control batch is contaminated with the expression vector


### Compare with PCA
# visualize screeplot and autoplotted pca for reference
deseq_batch_pca <-prcomp(t(assay(rld_batch)))
screeplot(deseq_batch_pca, type='lines')
autoplot(deseq_batch_pca)

# Create PCA object to be plotted with ggplot
plot_batch_pca <- plotPCA(rld_batch, intgroup = c("vector", "batch"), returnData = TRUE)

# Calculate and round variance for plotting
percentVar_batch <- round(100 * attr(plot_batch_pca, "percentVar_batch"))

# Plot PCA
deseq_pca_2 <- ggplot(plot_batch_pca, aes(x = PC1, y = PC2, color = vector, shape = batch)) +
  geom_point(size = 2.5) +
  xlab(paste0("PC1: ", percentVar_batch[1], "% variance")) +
  ylab(paste0("PC2: ", percentVar_batch[2], "% variance")) +
  coord_fixed() +
  scale_color_manual(values = c("red4", "darkblue", "goldenrod2")) +
  ggtitle("DESeq2 PCA: Batch Corrected") +
  theme(plot.title = element_text(hjust = 0.5))

tiff("deseq_pca_batch.tiff", units="in", width=7, height=5, res=300)
deseq_pca_2
dev.off()

# Make sure that contol is the first level in the vector factor
# This is to ensure the log2 fold change is calculated over the control (when results are
called at random)
deseq_batch_df$vector <- relevel(deseq_batch_df$vector, "control")

# Check to insure all samples are correct
as.data.frame(colData(deseq_batch_df))

# Run the DESeq analysis against raw counts
deseq_batch_analysis <- DESeq(deseq_batch_df)
results(deseq_batch_analysis)

# Call results
deseq_results_cont_exp_batch <- results(deseq_batch_analysis, contrast = c("vector",
"expression", "control"))
deseq_results_cont_exp_batch
```

```r
# Review Comparisons
mcols(deseq_results_cont_exp_batch, use.names = TRUE)

### Review Summary
summary(deseq_results_cont_exp_batch, alpha = 0.01)

# c=Calculate row means across vectors
mean_cont_batch <-  rowMeans(counts(deseq_analysis, normalized=TRUE)[,
deseq_batch_analysis$vector == "control"])
mean_exp_batch <-  rowMeans(counts(deseq_analysis, normalized=TRUE)[,
deseq_batch_analysis$vector == "expression"])

### Control 1 vs. expression
# Log Fold change
deseq_batch_lfc_1 <- lfcShrink(deseq_batch_analysis, contrast = c("vector", "expression",
"control"))
# add row names to log fold change data frame
deseq_batch_lfc_1 <-  cbind(as.data.frame(deseq_batch_lfc_1), mean_exp_batch, mean_cont_batch)
# Map gene names and ids
deseq_batch_lfc_1$symbol <- mapIds(org.Hs.eg.db, keys=row.names(deseq_batch_lfc_1),
                                   column="SYMBOL", keytype="ENSEMBL", multiVals="first")
deseq_batch_lfc_1$entrez <- mapIds(org.Hs.eg.db, keys=row.names(deseq_batch_lfc_1),
                                   column="ENTREZID", keytype="ENSEMBL", multiVals="first")
# Plot histogram to review distribution
ggplot(data = deseq_batch_lfc_1, mapping = aes(x = pvalue)) + geom_histogram(binwidth = 0.01)
ggplot(data = deseq_batch_lfc_1, mapping = aes(x = padj)) + geom_histogram(binwidth = 0.01)

#### Volcano Plots

# Create up, none, down labels for color then plot

# Control 1 vs. Expression
deseq_batch_lfc_1 <- deseq_batch_lfc_1 %>%
  mutate(threshold = ifelse((log2FoldChange >= 0 & padj < 0.01), "up_sig",
                            ifelse((log2FoldChange <= 0 & padj < 0.01) ,
                                   "down_sig", "not_sig")))
deseq_batch_lfc_1$threshold[is.na(deseq_batch_lfc_1$threshold)] <- "not_sig"

des_batch_vol_1 <- ggplot(deseq_batch_lfc_1, aes(x = log2FoldChange, y = -log10(pvalue))) +
  geom_point(aes(colour = threshold), size=1) +
  ggtitle("Control vs. Expression with deseq2: Batch Corrected") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ylim(0,50) + xlim(-6,6) +
  scale_colour_manual(values=c("blue", "grey", "red3"),
                      name="Threshold 0.01",
                      breaks=c("down_sig", "not_sig", "up_sig"),
                      labels=c("Down", "None", "Up"))
des_batch_vol_1

#### Heatmaps with top variable genes

# Control 1 vs. Expression
# Transform with regularized log
rld_1_batch <- rlog(deseq_batch_analysis)

# Order genes then filter for the top 25
top_genes_1_batch <- head(order(rowMeans(assay(rld_1_batch)), decreasing = TRUE), 30)
as.tibble(top_genes_1_batch)
deseq_batch_genes <- assay(rld_1_batch)[top_genes_1_batch, ]

# Calculate distances from the mean for clearer heatmap
```

```
deseq_batch_genes <- (deseq_batch_genes - rowMeans(deseq_batch_genes))

# Connect to ensemble database and label gene ids with gene symbol
mart <- useMart("ENSEMBL_MART_ENSEMBL","hsapiens_gene_ensembl")
gene_symbol <- getBM(c("hgnc_symbol","ensembl_gene_id"), "ensembl_gene_id",
row.names(deseq_batch_genes), mart)

row.names(deseq_batch_genes)[match(gene_symbol[,2], row.names(deseq_batch_genes))] <-
gene_symbol[,1]

# Create Annotation
# deseq_batch_ha

# Create Heatmap
deseq_batch_heat_1 <- Heatmap(deseq_batch_genes,
                              top_annotation = deseq_batch_ha,
                              show_column_names = F,
                              show_row_names = T,
                              cluster_rows = T,
                              cluster_columns = T,
                              clustering_method_columns = 'complete',
                              clustering_method_rows = "average",
                              row_names_gp = gpar(fontsize = 8),
                              top_annotation_height = unit(1, "cm"),
                              heatmap_legend_param = list(title = "Distance"),
                              col = colorRamp2(c(-1, 0, 1), c("red3", "white", "blue")),
                              column_title = "DESeq2 Control vs. Expression: Batch Corrected")

deseq_batch_heat_1

tiff("deseq_heat_batch.tiff", units="in", width=7, height=5, res=300)
deseq_batch_heat_1
dev.off()

# Convert gene ids to gene symbol
mart <- useMart("ENSEMBL_MART_ENSEMBL","hsapiens_gene_ensembl")
gene_symbol_deseq_batch <- getBM(c("hgnc_symbol","ensembl_gene_id"), "ensembl_gene_id",
row.names(deseq_results_cont_exp_batch), mart)

row.names(deseq_results_cont_exp_batch)[match(gene_symbol_deseq_batch[,2],
row.names(deseq_results_cont_exp_batch))] <- gene_symbol_deseq_batch[,1]

### Filter and order dataframes
# State whether they are upregulated or downregulated
# Filter out columns for significantly upregulated and downregulated genes

# Control vs. Expression Top 30
deseq_top_30_batch <- as.data.frame(deseq_results_cont_exp_batch)
deseq_top_30_batch <- deseq_top_30_batch[order(deseq_top_30_batch$padj),]
deseq_top_30_batch
deseq_top_30_batch <- head(deseq_top_30_batch, 30)
setDT(deseq_top_30_batch, keep.rownames = TRUE)[]
colnames(deseq_top_30_batch)[1] <- "genes"

as.tibble(deseq_top_30_batch)

deseq_top_batch <- as.data.frame(deseq_results_cont_exp_batch)
deseq_top_batch <- deseq_top_batch[order(deseq_top_batch$padj),]
setDT(deseq_top_batch, keep.rownames = TRUE)[]
colnames(deseq_top_batch)[1] <- "genes"
deseq_top_batch$expression <- NA
```

```
deseq_top_batch$expression[deseq_top_batch$log2FoldChange > 0 ] <- "up"
deseq_top_batch$expression[deseq_top_batch$log2FoldChange < 0 ] <- "down"
as.tibble(deseq_top_batch)
```

```
########
# edgeR with Combined Controls (batch effect Corrected) Pipeline

# Read in feature counts
edge_batch_counts <- read.table("feature_counts.txt", header=TRUE, row.names=1)
as.tibble(edge_batch_counts)

### Set up edgeR matrix
group <- c(1,1,1,1,1,1,2,2,2)
batch <- c(1,1,1,2,2,2,2,2,2)
edge_count_df <- DGEList(counts = edge_batch_counts, group = group, genes =
rownames(edge_batch_counts))
summary(edge_count_df)

#### Design Matrix for batch effect
design <- model.matrix(~ group + batch, data = edge_count_df$samples)
design

# filter out lowly expressed genes using count-per-million (CPM)
# Re-calculate library size
filter_rows <- rowSums(cpm(edge_count_df) > 1) >= 2
edge_batch_df_filter <- edge_count_df[filter_rows, , keep.lib.sizes=FALSE]
summary(edge_batch_df_filter)

# Normalize RNA composition (accounts for library size)
edge_batch_df_norm <- calcNormFactors(edge_batch_df_filter,  method="TMM")
edge_batch_df_norm

### Exploratory data analysis with DGEList object

# Plot MDS
tiff("edge_mds_batch.tiff", units="in", width=7, height=5, res=300)
plotMDS(edge_batch_df_norm, method="bcv", col=as.numeric(edge_batch_df_norm$samples$group),
main=("edgeR BCV: Batch Corrected"))
dev.off()

# Estimate Dispersion (common and tagwise in one run)
edge_batch_disp <- estimateDisp(edge_batch_df_norm, design)
edge_batch_disp

# Plot BCV
plotBCV(edge_batch_disp)

### Heatmap
logcpm_batch <- cpm(edge_batch_df_norm, prior.count=2, log=TRUE)
logcpm_batch
```

```
top_genes_2_batch <- head(order(rowMeans(logcpm_batch), decreasing = TRUE), 30)
as.tibble(top_genes_2_batch)
edge_batch_genes <- logcpm_batch[top_genes_2_batch, ]
edge_batch_genes
# Calculate distances from the mean for clearer heatmap
edge_batch_genes <- (edge_batch_genes - rowMeans(edge_batch_genes))

# Connect to ensemble database and label gene ids with gene symbol
mart <- useMart("ENSEMBL_MART_ENSEMBL","hsapiens_gene_ensembl")
gene_symbol <- getBM(c("hgnc_symbol","ensembl_gene_id"), "ensembl_gene_id",
row.names(edge_batch_genes), mart)
row.names(edge_batch_genes)[match(gene_symbol[,2], row.names(edge_batch_genes))] <-
gene_symbol[,1]

# Creat annotation
edge_batch_annotation <-  samples_batch

edge_batch_ha <- HeatmapAnnotation(df = edge_batch_annotation,
                                   col = list(vector = c("control" = "red4",
                                                         "expression" = "darkblue"),
                                              batch = c("1" = "goldenrod2",
                                                        "2" = "darkgreen")))

# Create heatmap
edge_batch_heat_1 <- Heatmap(edge_batch_genes,
                             top_annotation = edge_batch_ha,
                             show_column_names = F,
                             show_row_names = T,
                             cluster_rows = T,
                             cluster_columns = T,
                             clustering_method_columns = 'complete',
                             clustering_method_rows = "average",
                             row_names_gp = gpar(fontsize = 8),
                             top_annotation_height = unit(1, "cm"),
                             heatmap_legend_param = list(title = "Distance"),
                             col = colorRamp2(c(-1, 0, 1), c("red3", "white", "blue")),
                             column_title = "edgeR Control vs. Expression: Batch Corrected")

edge_batch_heat_1

tiff("edge_heat_batch.tiff", units="in", width=7, height=5, res=300)
edge_batch_heat_1
dev.off()

### Differential gene expression Classic Approach
edge_batch_exact <- exactTest(edge_batch_disp)
edge_batch_top_30 <- as.data.frame(topTags(edge_batch_exact, n = 30))
edge_batch_top_30
edge_batch_top <- topTags(edge_batch_exact, n = nrow(edge_batch_exact))
edge_batch_top

# Other approach (not used in this analysis)
# Use QL F-tests instead of the more usual likelihood ratio tests (LRT) as they give stricter
error rate control by accounting for the uncertainty in dispersion estimation:
# This gives lower p-values
#fit <- glmQLFit(edge_batch_disp, design, robust=TRUE)
#plotQLDisp(fit)
#qlf <- glmQLFTest(fit)
#edge_batch_top_30 <- as.data.frame(topTags(qlf, n = 30))
#edge_batch_top_30
```

```
#edge_batch_top <- topTags(qlf, n = nrow(qlf))
#edge_batch_top

edge_batch_top_final <- as.data.frame(edge_batch_top[,-1])
edge_batch_top_final

#### Volcano Plots
# Create up, none, down labels for color then plot

# Control 1 vs. Expression
edge_batch_top_volcano <- as.data.frame(edge_batch_top) %>%
  mutate(threshold = ifelse((logFC >= 0 & FDR < 0.01), "up_sig",
                            ifelse((logFC <= 0 & FDR < 0.01) ,
                                   "down_sig", "not_sig")))
edge_batch_top_volcano$threshold[is.na(edge_batch_top_volcano$threshold)] <- "not_sig"

edge_batch_vol_1 <- ggplot(edge_batch_top_volcano, aes(x = logFC, y = -log10(PValue))) +
  geom_point(aes(colour = threshold), size=1) +
  ggtitle("Control vs. Expression with edgeR: Batch Corrected") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ylim(0,50) + xlim(-6,6) +
  scale_colour_manual(values=c("blue", "grey", "red3"),
                      name="Threshold 0.01",
                      breaks=c("down_sig", "not_sig", "up_sig"),
                      labels=c("Down", "None", "Up"))
edge_batch_vol_1

# Connect to ensemble database and label gene ids with gene symbol
mart <- useMart("ENSEMBL_MART_ENSEMBL","hsapiens_gene_ensembl")
gene_symbol <- getBM(c("hgnc_symbol","ensembl_gene_id"), "ensembl_gene_id",
row.names(edge_batch_top_final), mart)
colnames(gene_symbol)[2] <- "genes"
gene_symbol

#row.names(edge_batch_top_final)[match(gene_symbol[,2], row.names(edge_batch_top_final))] <-
gene_symbol[,1]
#row.names(edge_batch_top_final)

# Top 30 EdgeR
edge_batch_top_final <- as.data.frame(edge_batch_top_final)
setDT(edge_batch_top_final, keep.rownames = TRUE)[]
colnames(edge_batch_top_final)[1] <- "genes"
edge_batch_top_final
edge_batch_top_final <- join(gene_symbol, edge_batch_top_final, type = "full", by = "genes")
edge_batch_top_final <- edge_batch_top_final[order(edge_batch_top_final$FDR),]
edge_batch_top_final
edge_batch_top_final$expression <- NA
edge_batch_top_final$expression[edge_batch_top_final$logFC > 0 ] <- "up"
edge_batch_top_final$expression[edge_batch_top_final$logFC < 0 ] <- "down"
edge_batch_top_final <- edge_batch_top_final[order(edge_batch_top_final$FDR),]
colnames(edge_batch_top_final)[1:2] <- c("genes", "gene_ids")
edge_batch_top_final




#######
### Compare DESeq2 and edgeR with correction for batch effects

# Compare Control vs. Expression (intersect)
# Filter for threshold 0.01
```

```
deseq_top_batch <- filter(deseq_top_batch, padj < 0.01)
edge_batch_top_final <- filter(edge_batch_top_final, FDR < 0.01)

# All significant
deseq_top_list <- as.vector(deseq_top_batch$genes) # 1218
length(deseq_top_list)
edge_top_list <- as.vector(edge_batch_top_final$genes) # 1429
length(edge_top_list)
length(intersect(deseq_top_list, edge_top_list)) # 944

# Upregulated
deseq_top_list_up <- deseq_top_batch[deseq_top_batch$expression == "up",] # 720
nrow(deseq_top_list_up)
deseq_top_list_up <- as.vector(deseq_top_list_up$genes)
edge_top_list_up <- edge_batch_top_final[edge_batch_top_final$expression == "up",] # 738
nrow(edge_top_list_up)
edge_top_list_up <- as.vector(edge_top_list_up$genes)
length(intersect(deseq_top_list_up, edge_top_list_up)) # 577

# Downregulated
deseq_top_list_down <- deseq_top_batch[deseq_top_batch$expression == "down",] #498
nrow(deseq_top_list_down)
deseq_top_list_down <- as.vector(deseq_top_list_down$genes)
edge_top_list_down <- edge_batch_top_final[edge_batch_top_final$expression == "down",] # 691
nrow(edge_top_list_down)
edge_top_list_down <- as.vector(edge_top_list_down$genes)
length(intersect(deseq_top_list_down, edge_top_list_down)) # 367

### Compare heatmaps
deseq_batch_heat_1
edge_batch_heat_1

### Compare volanco plots
vol_lay <- rbind(c(1,1,1,2,2,2,2),
                 c(1,1,1,2,2,2,2))

tiff("final_volcano_batch.tiff", units="in", width=9, height=5, res=300)
grid.arrange(des_batch_vol_1 + theme(legend.position = 'none',
                                     title = element_text(size = 9),
                                     axis.title = element_text(size =8),
                                     axis.text = element_text(size = 7)),
             edge_batch_vol_1+ theme(title = element_text(size = 9),
                                     axis.title = element_text(size =8),
                                     axis.text = element_text(size = 7)),
             ncol = 2, layout_matrix = vol_lay)
dev.off()
```