RNA seq analysis for Clinical Neuro samples: v0

Ephifania Geza

27/09/2022

Best practices for differentail expression analyses: DESeq2 and edgeR

This is a pipeline for differential expression analysis based on two most popular tools:

- the DESeq2 and
- the edgeR

We aligned the raw sequence reads to the reference genome **H37** using the *STAR* tool and counted the number of reads that mapped to each gene using a pseudo-count method: *SALMON*. All these were done using the https://github.com/nf-core/rnaseq (https://github.com/nf-core/rnaseq) pipeline. Herein, we provide step by step differential expression analysis given the count data.

Both *DESeq2* and *edgeR* are accept the **tximport** object, as such the first part is conducted regardless of the differential expression analysis.

We set our working directory. A directory that contains the count data for samples

All count data directories have the contain the pattern "_L001"

```
# List all directories containing data
samples <- list.files(path = data, full.names = F, pattern="_L001$")
# Obtain a vector of all filenames including the path for quant files
files <- file.path(paste(data,samples, sep = ""), "quant.sf")
```

Since all count data (quant) files have the same name it is useful to have names for each element.

```
names(files) <-  samples
```

We use the tximport package to import our transcript-level estimates from *SALMON*

```
txi <- tximport(files, type="salmon", txIn = TRUE, txOut = FALSE,
       tx2gene=tx2gene, ignoreTxVersion=TRUE, ignoreAfterBar=TRUE)
```

```
## reading in files with read_tsv
```

```
## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 3
9 40 41 42 43
## transcripts missing from tx2gene: 1
## summarizing abundance
## summarizing counts
## summarizing length
```

```
#txOut = TRUE is for differential transcript usage analyses
#attributes(txi) #Same as names(txi)
```

Using the provided table for clinical details we extracted relevant metadata

```
meta <- read.csv(paste(main_dir,"metadata.txt",sep=""),  row.names = 1)
```

We change the possible conditions of interest to factors and also rename *Possible* to *PO*, *Unlikely* to *UN*, *Yes* to *YES* and *No* to *NO*.

Let's see the structure of meta

```
str(meta)
```

```
## 'data.frame':    43 obs. of  3 variables:
##  $ immunosuppressed: Factor w/ 2 levels "NO","YES": 1 2 1 1 2 2 1 2 2 2 ...
##  $ COVID19_NEUR_Sx : Factor w/ 2 levels "UN","PO": 1 2 2 2 2 2 2 2 2 2 ...
##  $ Encephalopathy  : Factor w/ 2 levels "NO","YES": 2 2 2 1 1 1 2 1 2 2 ...
```

Check if rows of colData are same as cols of counts

```
all(rownames(meta) %in% colnames(txi$counts))
```

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

Let's fix it, we want colnames of txi$counts to be rownames of meta

```
colnames(txi$counts) <- rownames(meta)
#rownames(meta) <- colnames(txi$counts)
```

Check if they are the same

```
all(rownames(meta) %in% colnames(txi$counts))
```

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

We now divide our analysis into two subsections. It is important to note that each of these sections is further subdivided into 2:

- Quality control (normalization and unsupervised clustering)
- Differential expression analysis (involves modelling the raw counts for each gene, shrinking log2 fold changes and testing for differential analysis between conditions).

Analysis based on DESeq2

In this section we give details of the analysis based on the fact that are using *DESeq2* to determine the differential expression between conditions.

Interactions

- Immunosuppressed vs COVID-19

```
group_imm_vs_cov <- paste(meta$immunosuppressed, meta$COVID19_NEUR_Sx, sep=".")
# Convert to factor
group_imm_vs_cov <- factor(group_imm_vs_cov)
levels(group_imm_vs_cov)
```

```
## [1] "NO.PO"  "NO.UN"  "YES.PO" "YES.UN"
```

```
meta$imm_cov <- group_imm_vs_cov
```

- Immunosuppressed vs Encephalopathy

```
  group_imm_vs_enc <- paste(meta$immunosuppressed, meta$Encephalopathy, sep=".")
  # Convert to factor
  group_imm_vs_enc <- factor(group_imm_vs_enc)
  levels(group_imm_vs_enc)
```

```
## [1] "NO.NO"   "NO.YES"  "YES.NO"  "YES.YES"
```

```
  meta$imm_enc <- group_imm_vs_enc
```

- Encephalopathy vs COVID-19

```
  group_enc_vs_cov <- paste(meta$Encephalopathy, meta$COVID19_NEUR_Sx, sep=".")
  # Convert to factor
  group_enc_vs_cov <- factor(group_enc_vs_cov)
  levels(group_enc_vs_cov)
```

```
## [1] "NO.PO"  "NO.UN"  "YES.PO" "YES.UN"
```

```
  meta$enc_cov <- group_enc_vs_cov
```

- Immunosuppressed vs COVID-19 vs Encephalopathy

```
group_imm_cov_enc <- paste(meta$immunosuppressed, meta$Encephalopathy,
  meta$COVID19_NEUR_Sx, sep=".")
# Convert to factor
group_imm_cov_enc <- factor(group_imm_cov_enc)
meta$imm_cov_enc <- group_imm_cov_enc
levels(meta$imm_cov_enc)
```

```
## [1] "NO.NO.PO"   "NO.NO.UN"   "NO.YES.PO"  "NO.YES.UN"  "YES.NO.PO"
## [6] "YES.NO.UN"  "YES.YES.PO" "YES.YES.UN"
```

Create a data.frame often called colData

```
sampleTable <- data.frame(meta, row.names = colnames(txi$counts))
```

NB Make sure row names of the sampleTable aligns with column names of txi$counts

```
rownames(sampleTable) <- colnames(txi$counts)
```

Let's see our metadata, so that we know if we have the right comparisons

```
library(knitr)
library(kableExtra)
```

```
##
## Attaching package: 'kableExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     group_rows
```

```
knitr::kable(head(meta, n = 2 , tidy=TRUE)) %>%
  kable_styling(full_width = F, bootstrap_options = c("striped", "hover", "condensed"),
  font_size = 8) %>% row_spec(0, font_size=7)
```

	immunosuppressed	COVID19_NEUR_Sx	Encephalopathy	imm_cov	imm_enc	enc_cov	imm_cov_enc
COVC01	NO	UN	YES	NO.UN	NO.YES	YES.UN	NO.YES.UN
COVC02	YES	PO	YES	YES.PO	YES.YES	YES.PO	YES.YES.PO

1. Quality Checking

a. Creating a DESeqDataSet object

b. Exploratory Data Analysis (PCA & hierarchical clustering).

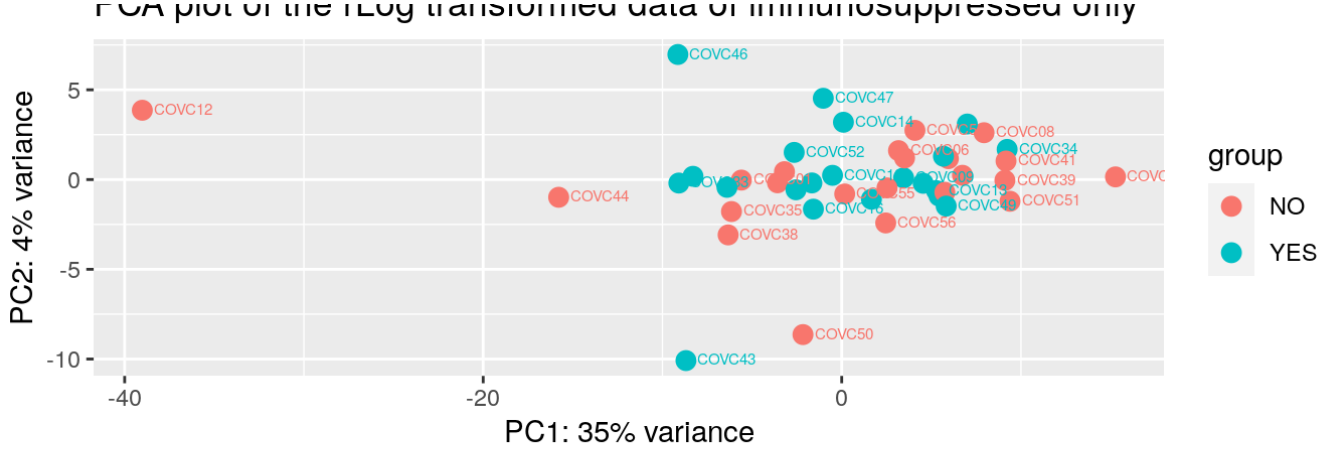
We first transform counts for data visualization. This section is based on https://hbctraining.github.io/DGE_workshop/lessons/03_DGE_QC_analysis.html (https://hbctraining.github.io/DGE_workshop/lessons/03_DGE_QC_analysis.html)

The PCAplot

PCA plot of immunosuppressed

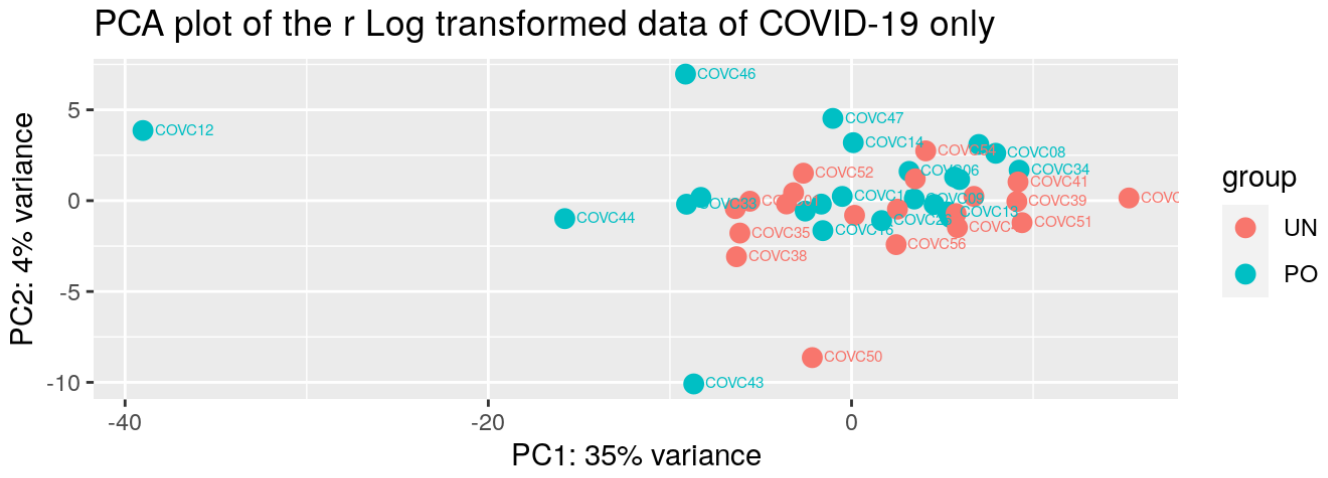
```
plotPCA(rld, intgroup=c("immunosuppressed"), ntop = 500) +
  ggtitle("PCA plot of the rLog transformed data of immunosuppressed only") +
  geom_text(aes(label=name), size=2, hjust=-0.2, vjust=0.4, check_overlap = T,
  fontface=0.1)
```

PCA plot of the rLog transformed data of immunosuppressed only



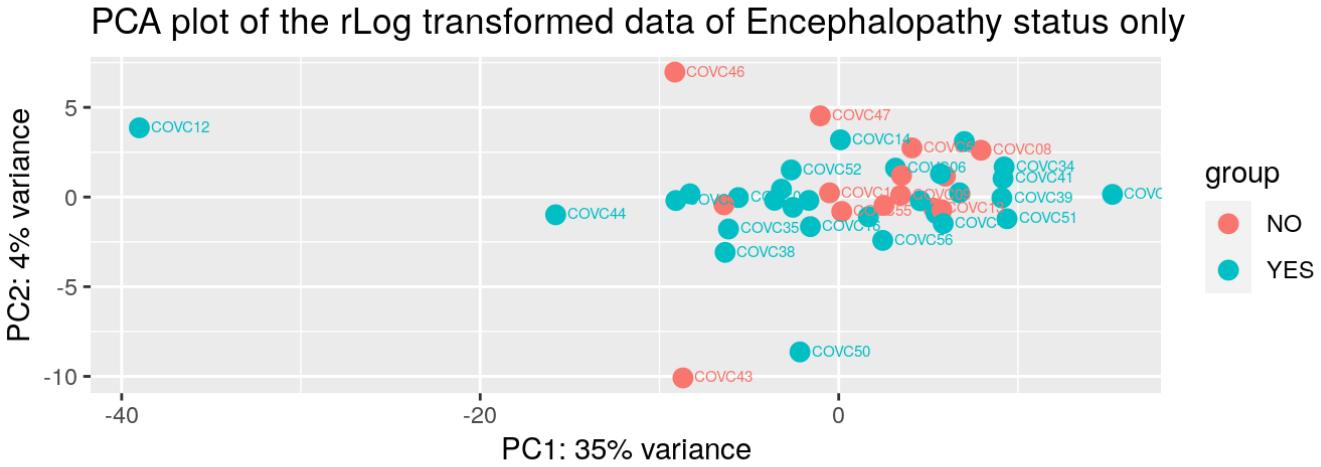
PCA plot of COVID19 only

```
plotPCA(rld, intgroup=c("COVID19_NEUR_Sx"), ntop = 500) +
  ggtitle("PCA plot of the r Log transformed data of COVID-19 only") +
  geom_text(aes(label=name), size=2, hjust=-0.2, vjust=0.4, check_overlap = T,
    fontface=0.1)
```



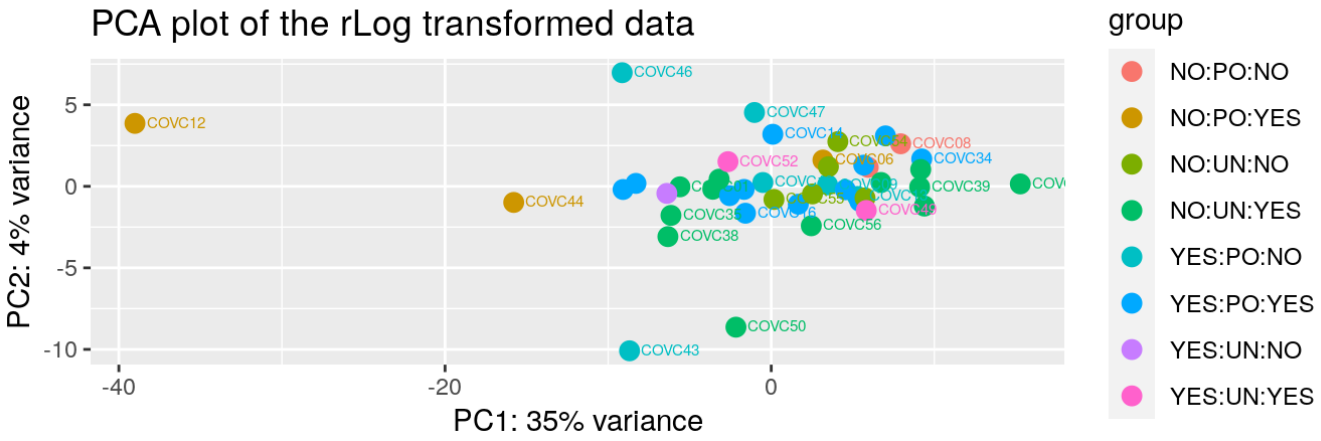
PCA plot of Encephalopathy

```
plotPCA(rld, intgroup=c("Encephalopathy"), ntop = 500) +
  ggtitle("PCA plot of the rLog transformed data of Encephalopathy status only") +
  geom_text(aes(label=name), size=2, hjust=-0.2,vjust=0.4, check_overlap = T,
    fontface=0.1)
```



PCA plot: immunosuppressed, COVID19 and Encephalopathy

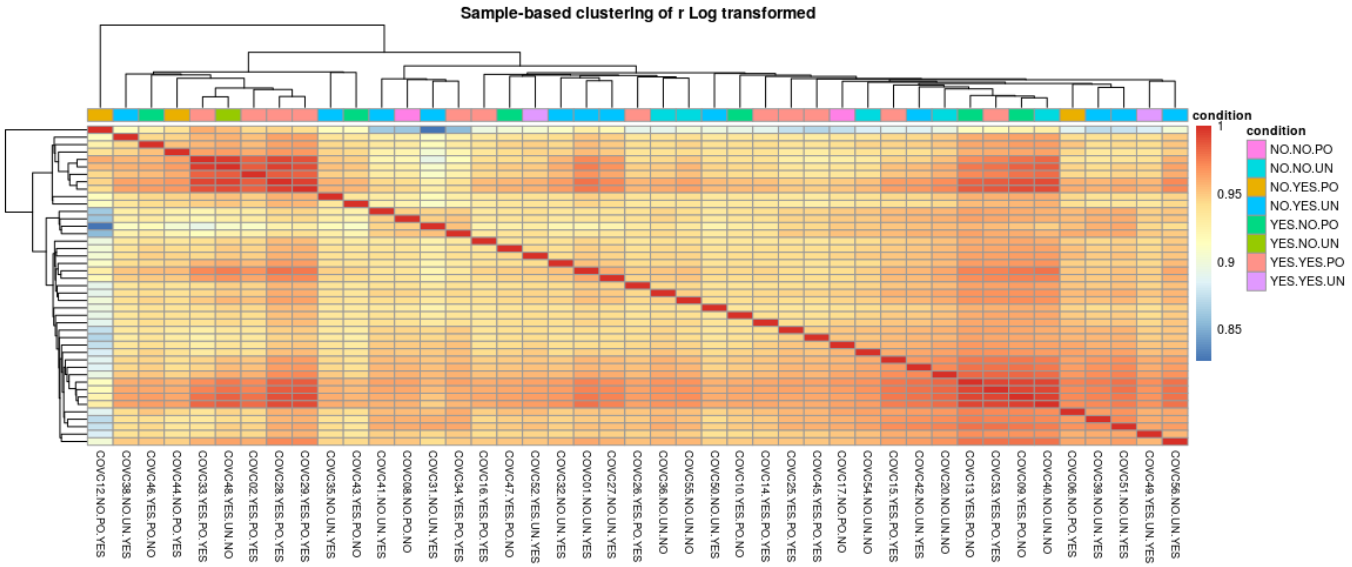
```
plotPCA(rld, intgroup = c("immunosuppressed", "COVID19_NEUR_Sx", "Encephalopathy"),
  ntop = 500) + ggtitle("PCA plot of the rLog transformed data") +
  geom_text(aes(label=name), size=2, hjust=-0.2,
  vjust=0.4, check_overlap = T, fontface=0.1)
```



Sample- and gene-based clustering

Sample-based clustering (Clustering of samples by gene expression)

```
knitr::include_graphics(paste(wkdir,"sample.png",sep = ""))
```

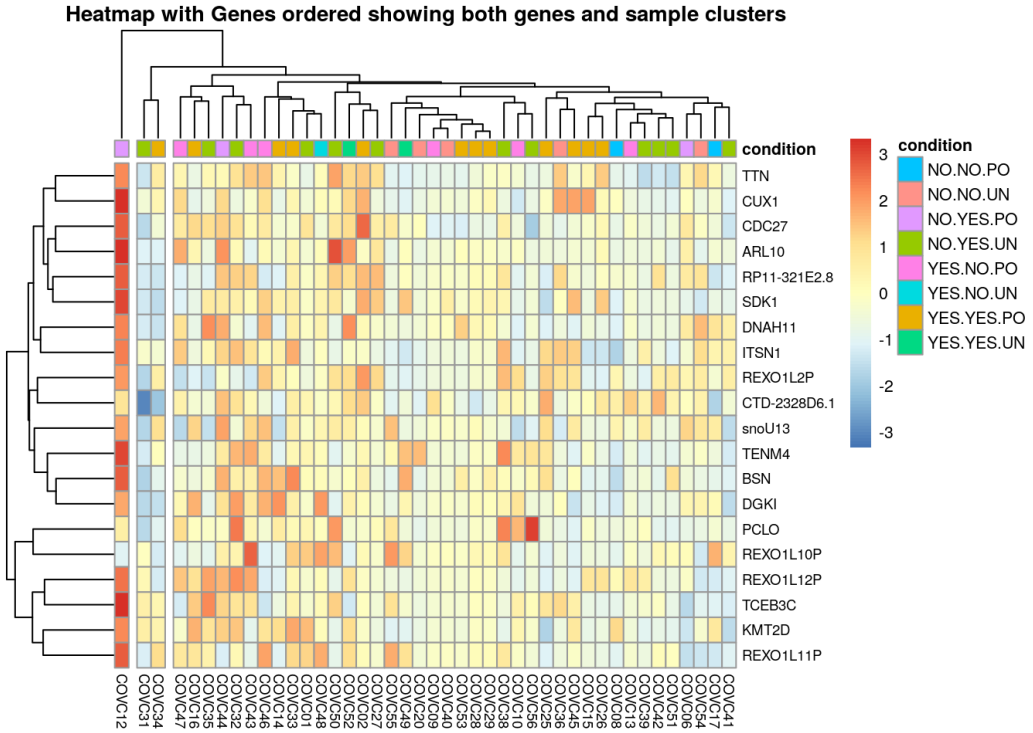


Gene-and sample-based clustering (Heatmap & dendrogram showing clustering of samples with similar gene expression & clustering of genes with similar expression patterns).

Dendrogram & heatmap of scaled gene values (row)

Scaling makes it easier to observe differences in values for each of the variables. In RNA seq, we scale by row since individuals are listed accross col. Here we also customise annotation colours.

```
pheatmap(mat, fontsize = 8, fontsize_row = 7, fontsize_col = 7, scale = "row",
  annotation_col = anno, main = "Heatmap with Genes ordered showing both genes and sample clusters",
  cutree_cols=3)
```



Running DESeq2.

In this section we - (a) normalize the count data by library size by estimating the ``size factor", - (b) estimate dispersion for the negative binomial model, and - (c) fits models and getting statistics for each gene for the design specified when you imported the data.

Let's check how many genes we have

```
nrow(dds)
```

```
## [1] 55773
```

Pre-filtering low counts by removing rows of the DESeqDataSet (dds) with no counts, or only a single count across all samples.

```
dds <- dds[rowSums(counts(dds)) > 1,]
```

Now we have

```
nrow(dds)
```

```
## [1] 42445
```

after filtering

```
dds <- DESeq(dds, fitType = "local",minReplicatesForReplace = Inf)
```

```
## estimating size factors
```

```
## using 'avgTxLength' from assays(dds), correcting for library size
```

```
## estimating dispersions
```

```
## gene-wise dispersion estimates
```

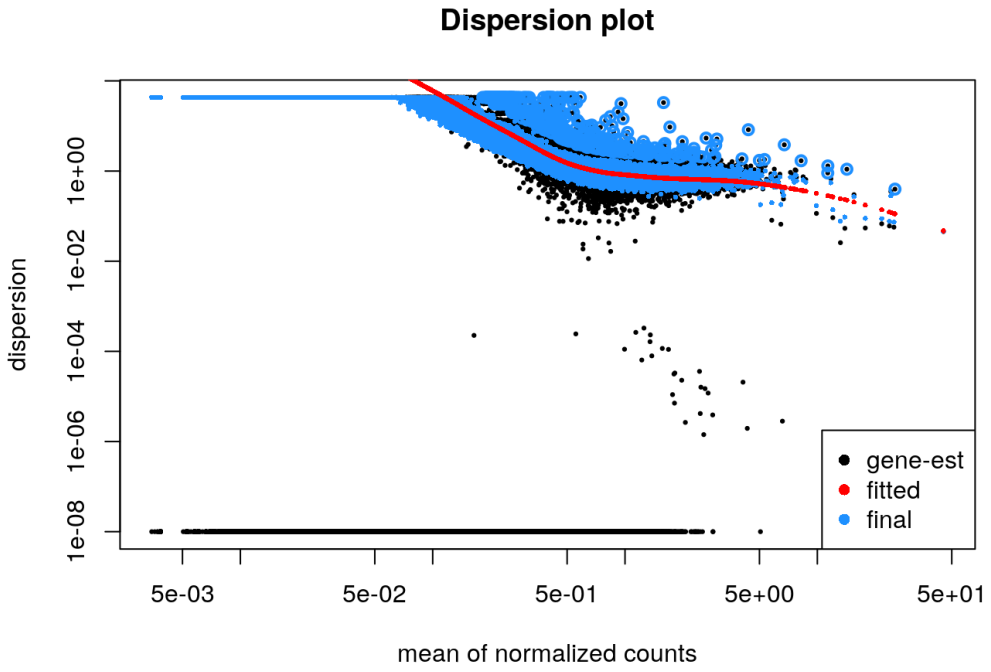
```
## mean-dispersion relationship
```

```
## final dispersion estimates
```

```
## fitting model and testing
```

Plot of dispersion estimates

```
plotDispEsts(dds, main="Dispersion plot")
```



In our model we have the following contrasts

```
resultsNames(dds)
```

```
## [1] "Intercept"                 "immunosuppressed_YES_vs_NO"
## [3] "COVID19_NEUR_Sx_PO_vs_UN"  "Encephalopathy_YES_vs_NO"
```

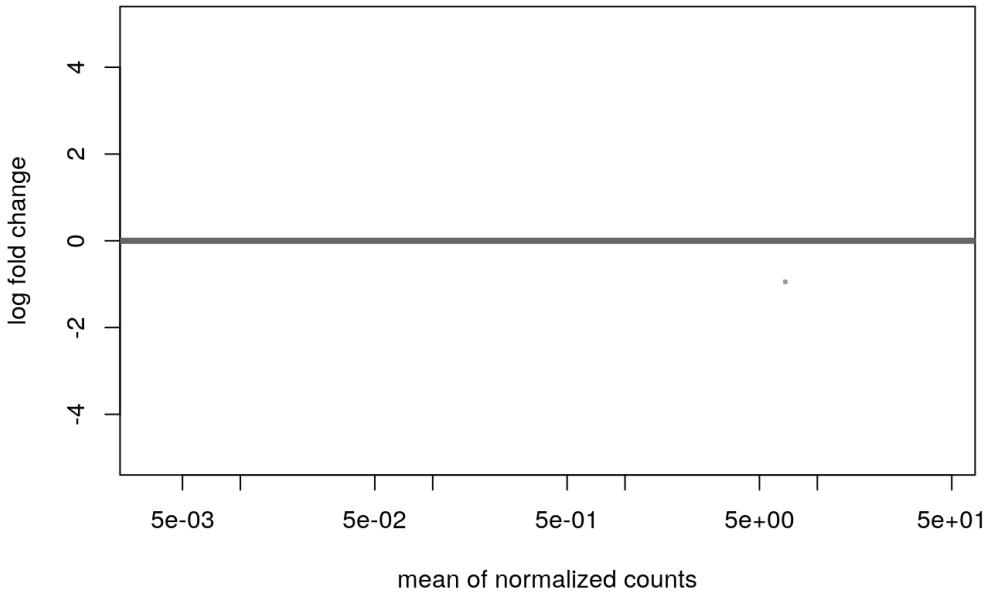
MAplot.

Shrinkage of effect size (LFC) is used 4 visualisation & ranking genes We use the “apeglm” method for effect size shrinkage as it improves the estimator when specified.

```
MAres <- lfcShrink(dds, coef="immunosuppressed_YES_vs_NO", type="apeglm")
```

```
## using 'apeglm' for LFC shrinkage. If used in published research, please cite:
##     Zhu, A., Ibrahim, J.G., Love, M.I. (2018) Heavy-tailed prior distributions for
##     sequence count data: removing the noise and preserving large differences.
##     Bioinformatics. https://doi.org/10.1093/bioinformatics/bty895
```

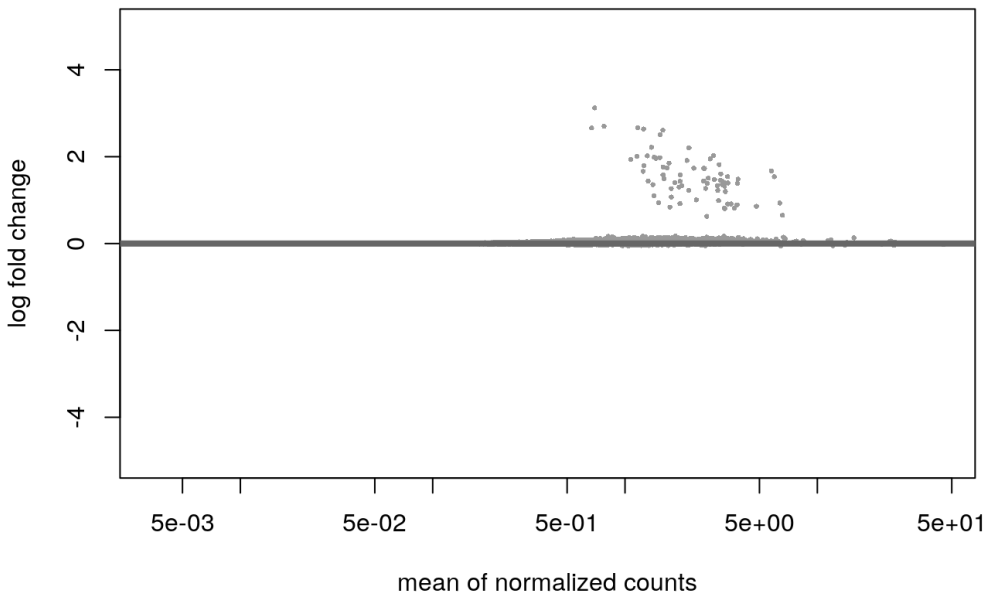
```
DESeq2::plotMA(MAres, ylim=c(-5, 5))
```




```
MAres <- lfcShrink(dds, coef="COVID19_NEUR_Sx_PO_vs_UN", type="apeglm")
```

```
## using 'apeglm' for LFC shrinkage. If used in published research, please cite:
##     Zhu, A., Ibrahim, J.G., Love, M.I. (2018) Heavy-tailed prior distributions for
##     sequence count data: removing the noise and preserving large differences.
##     Bioinformatics. https://doi.org/10.1093/bioinformatics/bty895
```

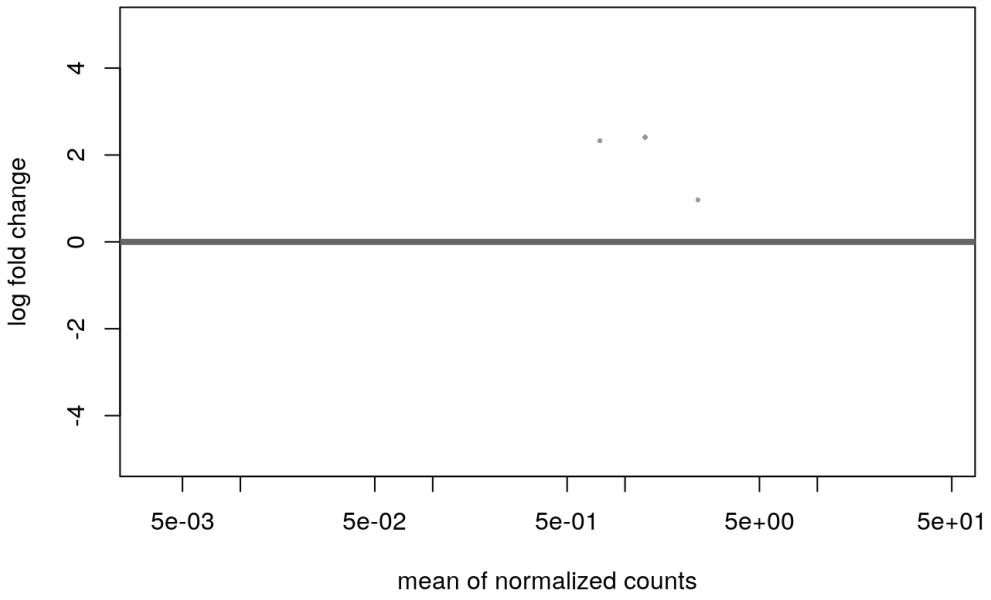
```
DESeq2::plotMA(MAres, ylim=c(-5, 5))
```




```
MAres <- lfcShrink(dds, coef="Encephalopathy_YES_vs_NO", type="apeglm")
```

```
## using 'apeglm' for LFC shrinkage. If used in published research, please cite:
##     Zhu, A., Ibrahim, J.G., Love, M.I. (2018) Heavy-tailed prior distributions for
##     sequence count data: removing the noise and preserving large differences.
##     Bioinformatics. https://doi.org/10.1093/bioinformatics/bty895
```

```
DESeq2::plotMA(MAres, ylim=c(-5, 5))
```



Testing different hypotheses

DESeq2 adjusts the p value by different methods including the “holm”, “hochberg”, “hommel”, “bonferroni”, “BY” and Benjamini and Hochberg (“BH”) which is default default. We adjusted the p-values using most of these and found no genes that were significant between the different hypothesis. Thus, the results we report herein are not p-value adjusted.

```
res_cov <- results(dds, contrast = c("COVID19_NEUR_Sx","UN","PO"), pAdjustMethod = "none")
```

Does immunosuppression independently cause differentail expression?

Upon conducting the test for the significance of difference between the patients with/without immuno-suppression, the summary of genes that are not significant, up-and down-regulated is as follows

```
summary(res_imm)
```

```
##
## out of 42445 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 202, 0.48%
## LFC < 0 (down)     : 428, 1%
## outliers [1]       : 57, 0.13%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

Does COVID-19 as a possible aetiology independently cause differential expression?

Upon conducting the test for the significance of difference between the patients who are most probable and unlikely to have COVID-19, the summary of genes that are not significant, up-and down-regulated is as follows

```
summary(res_cov)
```

```
##
## out of 42445 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 35, 0.082%
## LFC < 0 (down)     : 1700, 4%
## outliers [1]       : 57, 0.13%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

Does encephalopathy status independently cause differential expression?

Upon conducting the test for the significance of difference between the patients with/without encephalopathy status, the summary of genes that are not significant, up-and down-regulated is as follows

```
summary(res_enc)
```

```
##
## out of 42445 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 400, 0.94%
## LFC < 0 (down)     : 66, 0.16%
## outliers [1]       : 57, 0.13%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

Interaction of conditions

```
resultsNames(dds_ic)
```

```
## [1] "Intercept"               "imm_cov_NO.UN_vs_NO.PO"
## [3] "imm_cov_YES.PO_vs_NO.PO" "imm_cov_YES.UN_vs_NO.PO"
```

```
resultsNames(dds_ie)
```

```
## [1] "Intercept"                "imm_enc_NO.YES_vs_NO.NO"
## [3] "imm_enc_YES.NO_vs_NO.NO"  "imm_enc_YES.YES_vs_NO.NO"
```

```
resultsNames(dds_ec)
```

```
## [1] "Intercept"               "enc_cov_NO.UN_vs_NO.PO"
## [3] "enc_cov_YES.PO_vs_NO.PO" "enc_cov_YES.UN_vs_NO.PO"
```

```
res_imm_cov_noun_nopo <- results(dds_ic, name = "imm_cov_NO.UN_vs_NO.PO",
  pAdjustMethod = "none" )
res_imm_cov_yespo_nopo <- results(dds_ic, name = "imm_cov_YES.PO_vs_NO.PO",
  pAdjustMethod = "none" )
res_imm_cov_yesun_nopo <- results(dds_ic, name = "imm_cov_YES.UN_vs_NO.PO",
  pAdjustMethod = "none" )
summary(res_imm_cov_noun_nopo)
```

```
##
## out of 42445 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 20, 0.047%
## LFC < 0 (down)     : 2476, 5.8%
## outliers [1]       : 95, 0.22%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
summary(res_imm_cov_yespo_nopo)
```

```
##
## out of 42445 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 82, 0.19%
## LFC < 0 (down)     : 902, 2.1%
## outliers [1]       : 95, 0.22%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

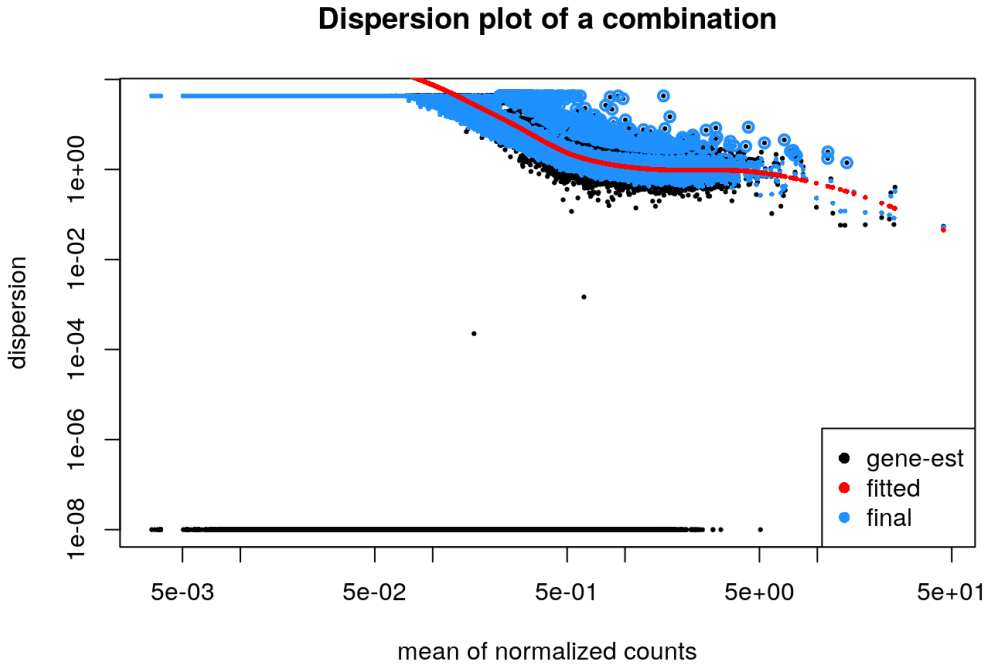
```
summary(res_imm_cov_yesun_nopo)
```

```
##
## out of 42445 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 77, 0.18%
## LFC < 0 (down)     : 269, 0.63%
## outliers [1]       : 95, 0.22%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

How do the three conditions affect the expression of genes

Plot dispersion estimates

```
plotDispEsts(dds1, main="Dispersion plot of a combination")
```



Let's get the coef first

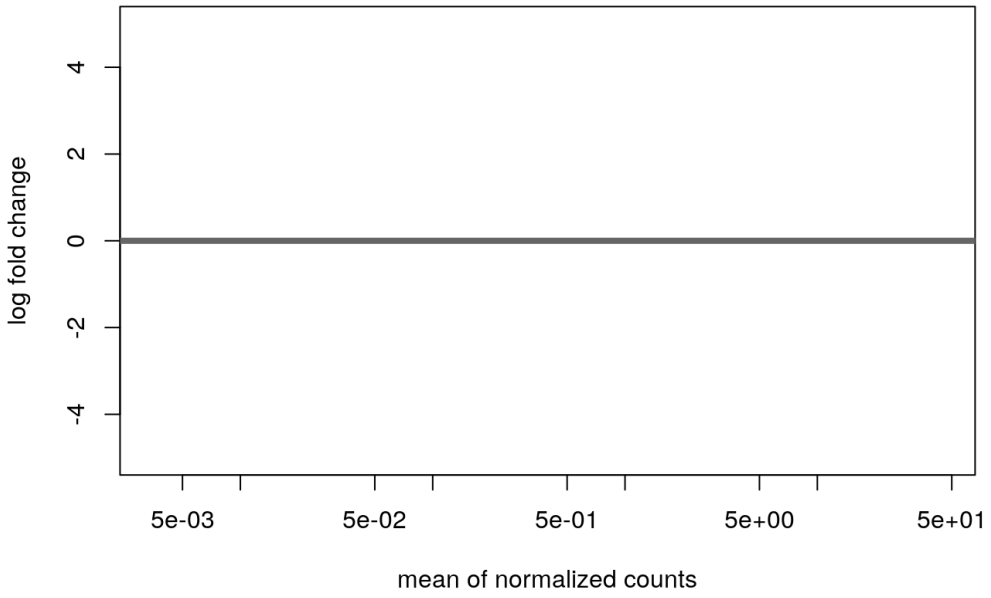
```
resultsNames(dds1)
```

```
## [1] "Intercept"                           "imm_cov_enc_NO.NO.UN_vs_NO.NO.PO"
## [3] "imm_cov_enc_NO.YES.PO_vs_NO.NO.PO"  "imm_cov_enc_NO.YES.UN_vs_NO.NO.PO"
## [5] "imm_cov_enc_YES.NO.PO_vs_NO.NO.PO"  "imm_cov_enc_YES.NO.UN_vs_NO.NO.PO"
## [7] "imm_cov_enc_YES.YES.PO_vs_NO.NO.PO" "imm_cov_enc_YES.YES.UN_vs_NO.NO.PO"
```

```
MAres <- lfcShrink(dds1, coef="imm_cov_enc_NO.NO.UN_vs_NO.NO.PO", type="apeglm")
```

```
## using 'apeglm' for LFC shrinkage. If used in published research, please cite:
##     Zhu, A., Ibrahim, J.G., Love, M.I. (2018) Heavy-tailed prior distributions for
##     sequence count data: removing the noise and preserving large differences.
##     Bioinformatics. https://doi.org/10.1093/bioinformatics/bty895
```

```
DESeq2::plotMA(MAres, ylim=c(-5, 5))
```




```
summary(res_YESYESUN_NONOPO)
```

```
##
## out of 42445 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 427, 1%
## LFC < 0 (down)     : 51, 0.12%
## outliers [1]       : 3, 0.0071%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
summary(res_YESYESPO_NONOPO)
```

```
##
## out of 42445 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 938, 2.2%
## LFC < 0 (down)     : 23, 0.054%
## outliers [1]       : 3, 0.0071%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
summary(res_YESNOUN_NONOPO)
```

```
##
## out of 42445 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 500, 1.2%
## LFC < 0 (down)     : 0, 0%
## outliers [1]       : 3, 0.0071%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
summary(res_YESNOPO_NONOPO)
```

```
##
## out of 42445 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 933, 2.2%
## LFC < 0 (down)     : 13, 0.031%
## outliers [1]       : 3, 0.0071%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
summary(res_YESYESPO_NONOPO)
```

```
##
## out of 42445 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 938, 2.2%
## LFC < 0 (down)     : 23, 0.054%
## outliers [1]       : 3, 0.0071%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
summary(res_NOYESUN_NONOPO)
```

```
##
## out of 42445 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 488, 1.1%
## LFC < 0 (down)     : 57, 0.13%
## outliers [1]       : 3, 0.0071%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
summary(res_NONOUN_NONOPO)
```

```
##
## out of 42445 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 303, 0.71%
## LFC < 0 (down)     : 56, 0.13%
## outliers [1]       : 3, 0.0071%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
summary(res_NOYESPO_NONOPO)
```

```
##
## out of 42445 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 2955, 7%
## LFC < 0 (down)     : 4, 0.0094%
## outliers [1]       : 3, 0.0071%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

Significant genes based on DESeq2

Number of genes that are significant given nonoun vs nonopo

```
nrow(sig_genes_NONOUN_NONOPO)
```

```
## [1] 0
```

```
# Write out significant genes into a file if any.
write_csv(sig_genes_NONOUN_NONOPO,paste(wkdir,"sigresults_nonoun_nonopo.csv",sep = ""))
```

Number of genes that are significant given noyesun vs nonopo

```
nrow(sig_genes_NOYESUN_NONOPO)
```

```
## [1] 0
```

```
# Write out significant genes into a file if any.
write_csv(sig_genes_NOYESUN_NONOPO,paste(wkdir,"sigresults_noyespo_nonopo.csv",sep = ""))
```

Number of genes that are significant given noyespo vs nonopo

```
nrow(sig_genes_NOYESPO_NONOPO)
```

```
## [1] 0
```

```
# Write out significant genes into a file if any.
write_csv(sig_genes_NOYESPO_NONOPO,paste(wkdir,"sigresults_noyespo_nonopo.csv",sep = ""))
```

Number of genes that are significant given yesyespo vs nonopo

```
nrow(sig_genes_YESYESPO_NONOPO)
```

```
## [1] 0
```

```
# Write out significant genes into a file if any.
write_csv(sig_genes_YESYESPO_NONOPO,paste(wkdir,"sigresults_yesyespo_nonopo.csv",sep = ""))
```

Number of genes that are significant given yesyesun vs nonopo

```
nrow(sig_genes_YESYESUN_NONOPO)
```

```
## [1] 0
```

```
# Write out significant genes into a file if any.
write_csv(sig_genes_YESYESUN_NONOPO,paste(wkdir,"sigresults_yesyesun_nonopo.csv",sep = ""))
```

Number of genes that are significant given yesnopo vs nonopo

```
nrow(sig_genes_YESNOPO_NONOPO)
```

```
## [1] 0
```

```
# Write out significant genes into a file if any.
write_csv(sig_genes_YESNOPO_NONOPO,paste(wkdir,"sigresults_yesnopo_nonopo.csv",sep = ""))
```

Number of genes that are significant given yesnoun vs nonopo

```
nrow(sig_genes_YESNOUN_NONOPO)
```

```
## [1] 0
```

```
# Write out significant genes into a file if any.
write_csv(sig_genes_YESNOUN_NONOPO,paste(wkdir,"sigresults_yesnoun_nonopo.csv",sep = ""))
```

Reports

We have saved the genes that showed differential expression between the tested conditions in the following accompanying files

EdgeR

Creating a DGEList object for use in edgeR.

```
y <- DGEList(cts)

y <- scaleOffset(y, normMat)
```

Filtering. A gene is only retained if it is expressed at a minimum level:

```
keep <- filterByExpr(y)
```

```
## Warning in filterByExpr.DGEList(y): All samples appear to belong to the same
## group.
```

```
# Lets check the number of genes that are retained
summary(keep) # Only 8 genes are retained
```

```
##    Mode   FALSE    TRUE
## logical   55765       8
```

```
dim(y)
```

```
## [1]  8 43
```

Our y is now ready to be used for dispersion estimation.

We create a matrix of CPMs within edgeR

As compared to the median of ration's (DESeq2 normalization), in **edgeR**, TMM normalization is performed to eliminate composition biases between libraries.

```
y <- calcNormFactors(y)
```

```
## Warning in calcNormFactors.DGEList(y): object contains offsets, which take precedence over library
## sizes and norm factors (and which will not be recomputed).
```

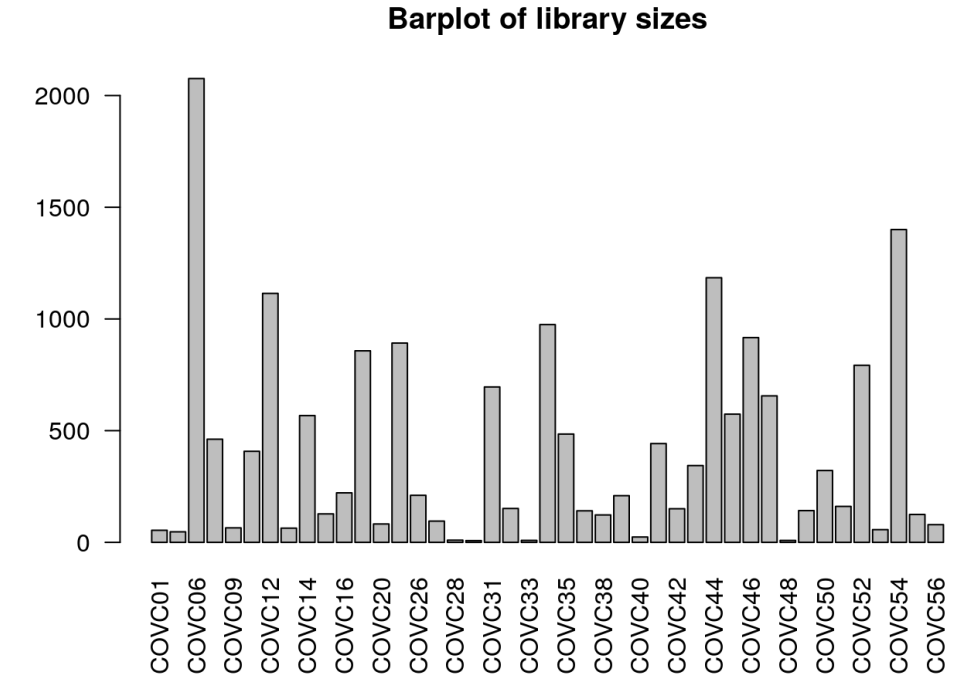
```
# store the groups for the samples in the DGEList object
group <- paste(meta$immunosuppressed,meta$COVID19_NEUR_Sx,sep=".")
# Convert to factor
group <- factor(group)
levels(group)
```

```
## [1] "NO.PO"  "NO.UN"  "YES.PO" "YES.UN"
```

```
y$samples$group <- group
```

Barplot of library sizes

```
barplot(y$samples$lib.size,names=colnames(y),las=2,main = "Barplot of library sizes")
```



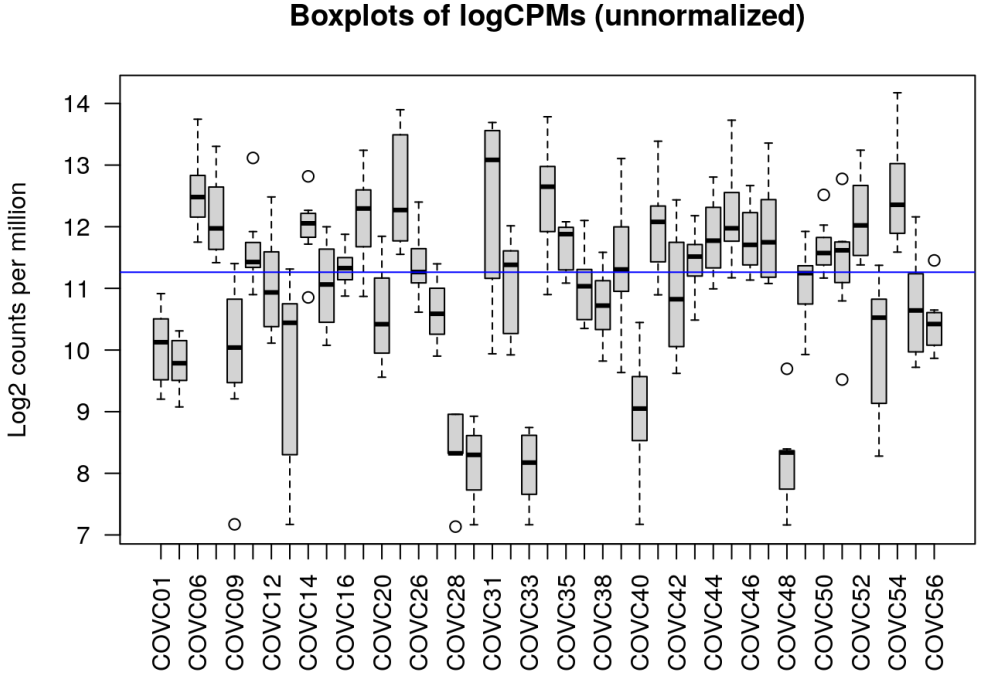
Distribution of raw counts

We use the cpm function to get log2 counts per million, which are corrected for the different library sizes. The cpm function also adds a small offset to avoid taking log of zero.

```
logcounts <- cpm(y,log = TRUE)
```

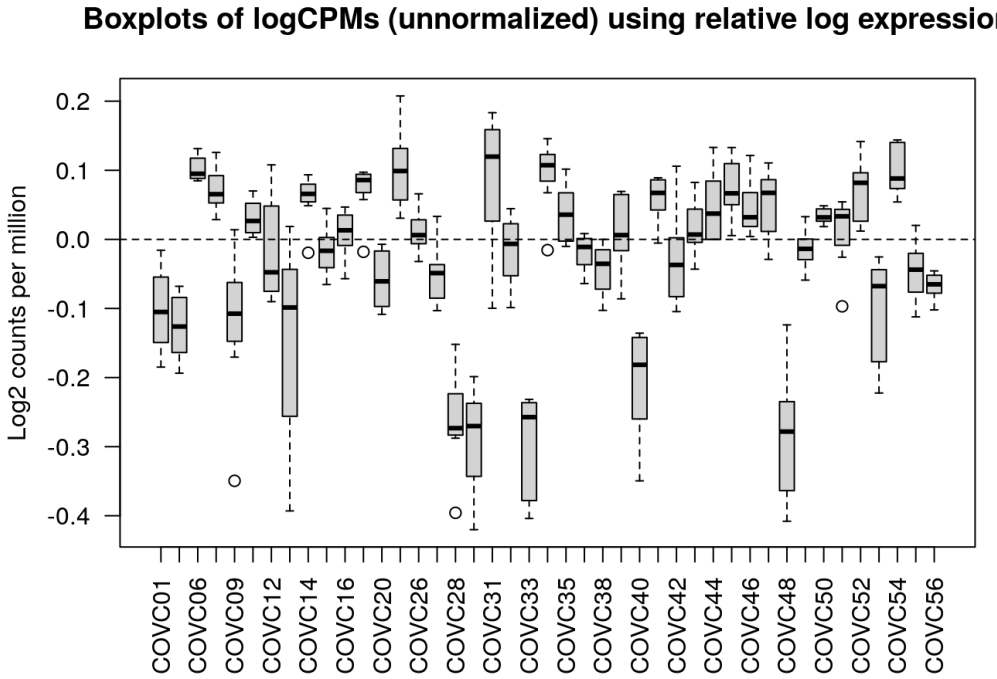
```
## Warning in cpm.DGEList(y, log = TRUE): Offset may not reflect library sizes.
## Scaling offset may be required.
```

```
boxplot(logcounts, xlab="", ylab="Log2 counts per million", las=2, main="Boxplots of logCPMs (unnormalized)")
abline(h=median(logcounts),col="blue")
```



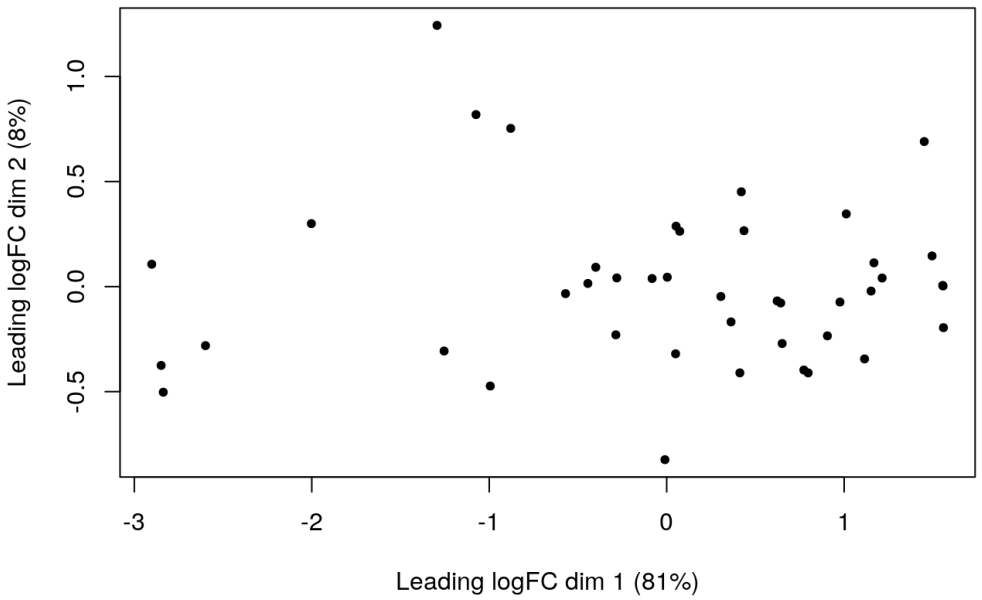
Most of the samples are far above or below the blue horizontal line, thus we need to investigate that samples further. Another kind of QC plot that is helpful in checking for dodgy samples is a relative log expression (RLE) plot (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5798764/ (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5798764/)), which is generated with plotRLE from the EDASeq package as follows.

```
plotRLE(logcounts, xlab="", ylab="Log2 counts per million", las=2, main="Boxplots of logCPMs (unnormalized) u
sing relative log expression")
```




```
# MDS plot
plotMDS(y, pch = 20)
```

```
## Warning in cpm.DGEList(x, log = TRUE, prior.count = prior.count): Offset may not
## reflect library sizes. Scaling offset may be required.
```



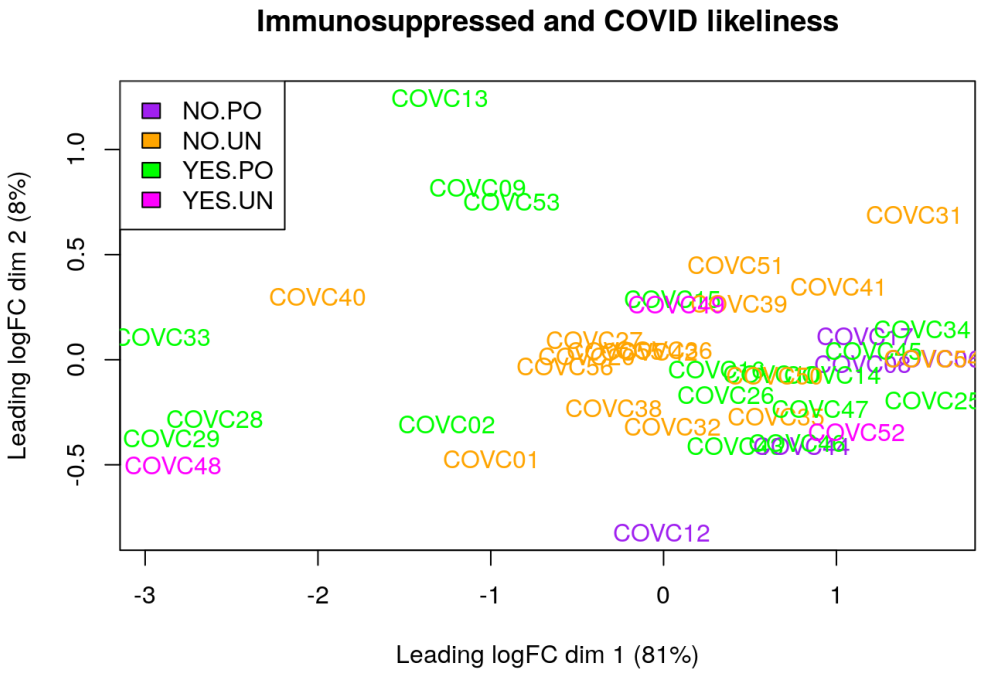

```
# Add a new column that combines immunosuppressed and covid
meta$imm_cov <- group

col.group <- c("purple","orange","green", "magenta")[meta$imm_cov]

# Redo the MDS with sample colouring
plotMDS(y,col=col.group)
```

```
## Warning in cpm.DGEList(x, log = TRUE, prior.count = prior.count): Offset may not
## reflect library sizes. Scaling offset may be required.
```

```
# Let's add a legend to the plot so we know which colours correspond to which samples
legend("topleft",fill=c("purple","orange","green","magenta"),legend=levels(meta$imm_cov))
# Add a title
title("Immunosuppressed and COVID likeliness")
```



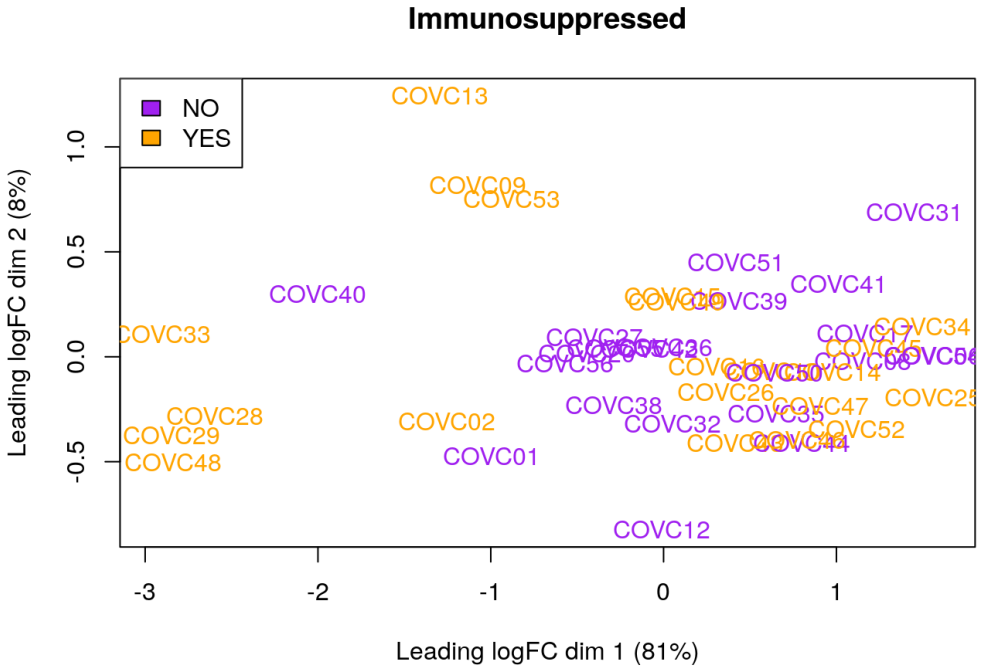

```
# Similarly for immunosuppressed only
levels(meta$immunosuppressed)
```

```
## [1] "NO"  "YES"
```

```
col.group <- c("purple","orange")[meta$immunosuppressed]
plotMDS(y,col=col.group)
```

```
## Warning in cpm.DGEList(x, log = TRUE, prior.count = prior.count): Offset may not
## reflect library sizes. Scaling offset may be required.
```

```
# Let's add a legend to the plot so we know which colours correspond to which samples
legend("topleft",fill=c("purple","orange"),legend=levels(meta$immunosuppressed))
# Add a title
title("Immunosuppressed")
```



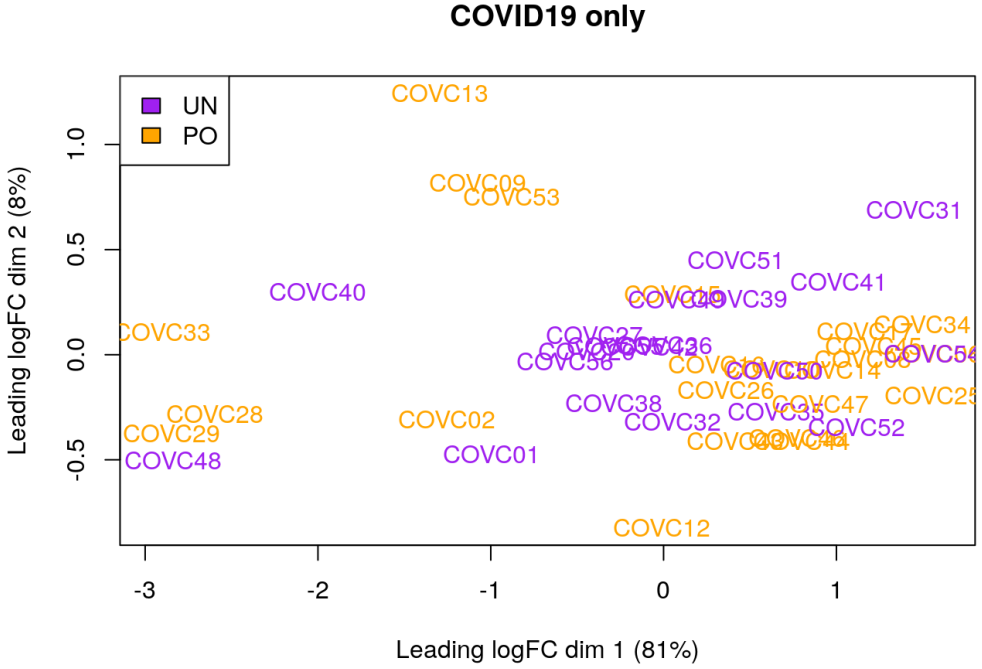

```
# Similarly for immunosuppressed only
levels(meta$COVID19_NEUR_Sx)
```

```
## [1] "UN" "PO"
```

```
col.group <- c("purple","orange")[meta$COVID19_NEUR_Sx]
plotMDS(y,col=col.group)
```

```
## Warning in cpm.DGEList(x, log = TRUE, prior.count = prior.count): Offset may not
## reflect library sizes. Scaling offset may be required.
```

```
# Let's add a legend to the plot so we know which colours correspond to which samples
legend("topleft",fill=c("purple","orange"),legend=levels(meta$COVID19_NEUR_Sx))
# Add a title
title("COVID19 only")
```



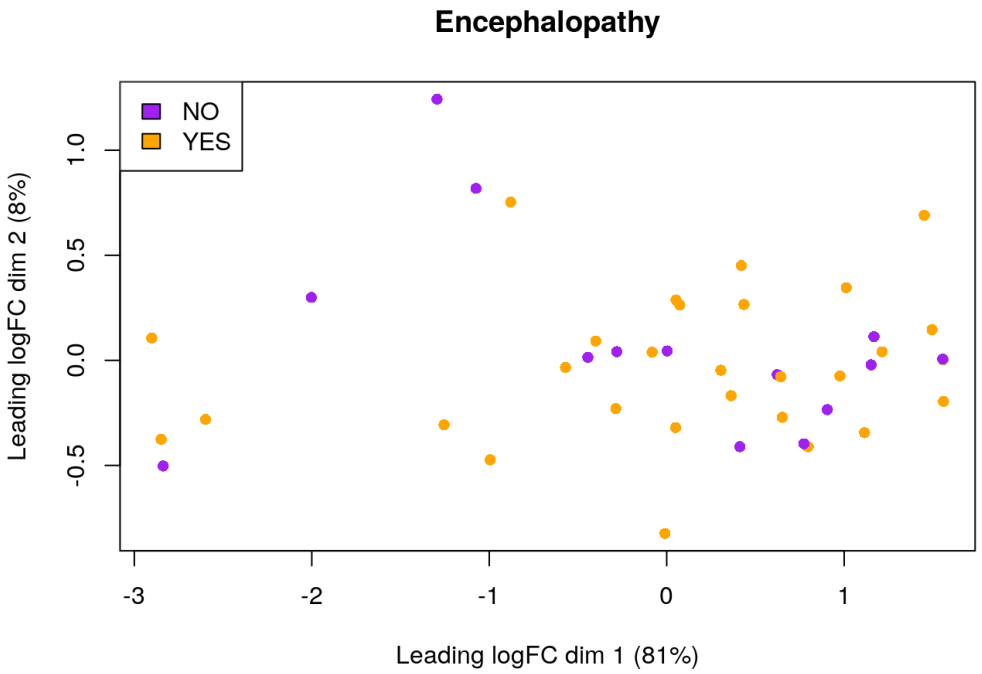
```
# Similarly for Encephalopathy status only
levels(meta$Encephalopathy)
```

```
## [1] "NO"  "YES"
```

```
col.group <- c("purple","orange")[meta$Encephalopathy]
plotMDS(y,col=col.group,pch = 16)
```

```
## Warning in cpm.DGEList(x, log = TRUE, prior.count = prior.count): Offset may not
## reflect library sizes. Scaling offset may be required.
```

```
# Let's add a legend to the plot so we know which colours correspond to which samples
legend("topleft",fill=c("purple","orange"),legend=levels(meta$Encephalopathy))
# Add a title
title("Encephalopathy")
```

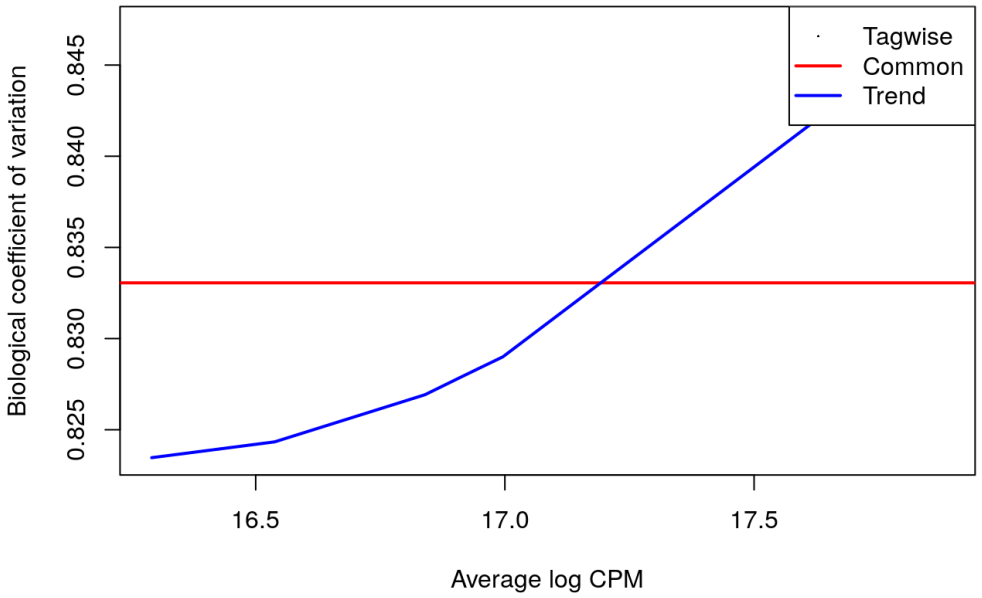



```
design <- model.matrix(~ 0+meta$imm_cov) # Hypothesis 1
# Estimate dispersions
y <- estimateDisp(y, design, robust=TRUE)

y$common.dispersion
```

```
## [1] 0.6939839
```

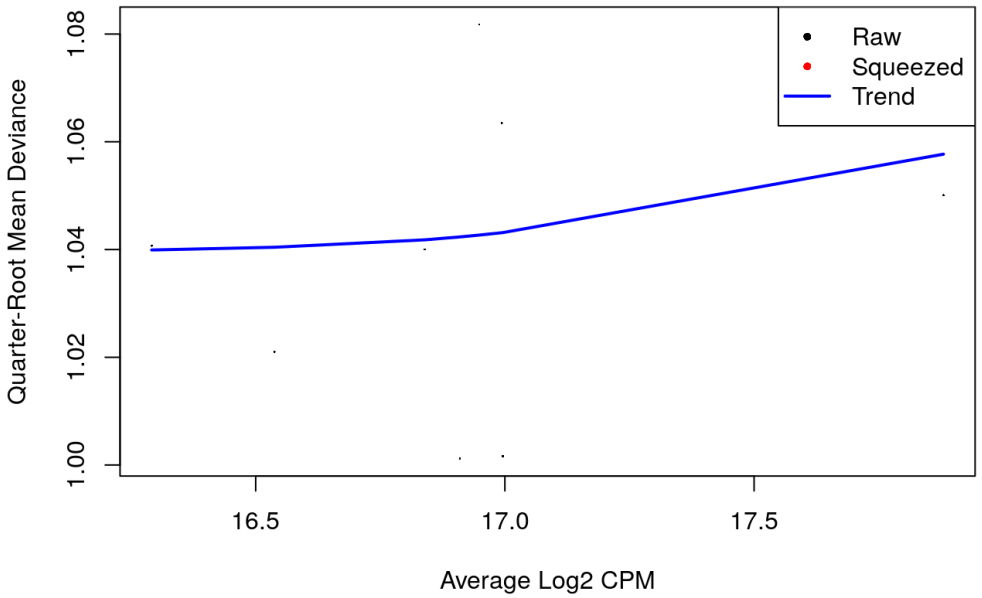
```
plotBCV(y) # only trended is useful in the QL pipeline
```




```
fit <- glmQLFit(y, design, robust=TRUE) # estimate QL dispersions
# Let's see the coefficients fitted
kable(head(fit$coefficients)) %>% kable_styling(font_size=7)
```

	meta$imm_c ovNO. PO < /th >< thstyle =" text − align : right;" > meta imm_covNO.UN	meta$imm_c ovYES. PO < /th >< thstyle =" text − align : right;" > meta imm_covYES.UN		
CTD-2328D6.1	-5.350307	-6.045358	-5.893943	-5.724303
KCNQ1OT1	-5.793206	-6.587184	-6.394515	-6.535573
MT-CO1	-5.680921	-5.847362	-6.024171	-6.352437
MT-ND4	-5.807764	-6.141381	-6.218888	-6.355850
MT-ND5	-5.470181	-5.759984	-5.989771	-6.272699
MT-RNR1	-5.322572	-5.869533	-5.771240	-5.956352

```
plotQLDisp(fit) # visualize QL dispersions
```



Differential Expression

We test for significant DE in each gene, using the QL F-test - (1) We test for DE btwn immunosupressed and no immunosuppressed

```
qlf1 <- glmQLFTest(fit,coef=1)
```

```
qlf_2vs1 <- glmQLFTest(fit,coef = 2)
```

Baseline meta$imm_covNO.PO$, meta$imm_covYES.PO,

"1*meta$imm_covNO.PO - 1 * meta$imm_covYES.PO"

```
qlf_3vs1 <- glmQLFTest(fit,contrast = c(1,0,-1,0))
```

```
# "1*meta$imm_covNO.PO -1*meta$imm_covYES.PO"
qlf_4vs1 <- glmQLFTest(fit,contrast = c(1,0,0,-1))
```

```
# "1*meta$imm_covNO.UN -1*meta$imm_covYES.PO"
qlf_2vs3 <- glmQLFTest(fit,contrast = c(0,1,-1,0))
```

The top set of most significant genes can be examined with topTags. A +ve LFC represents genes that are up in YES over NO. Multiplicity correction is performed by applying the B-H method on the p-values, to control the false discovery rate (FDR).

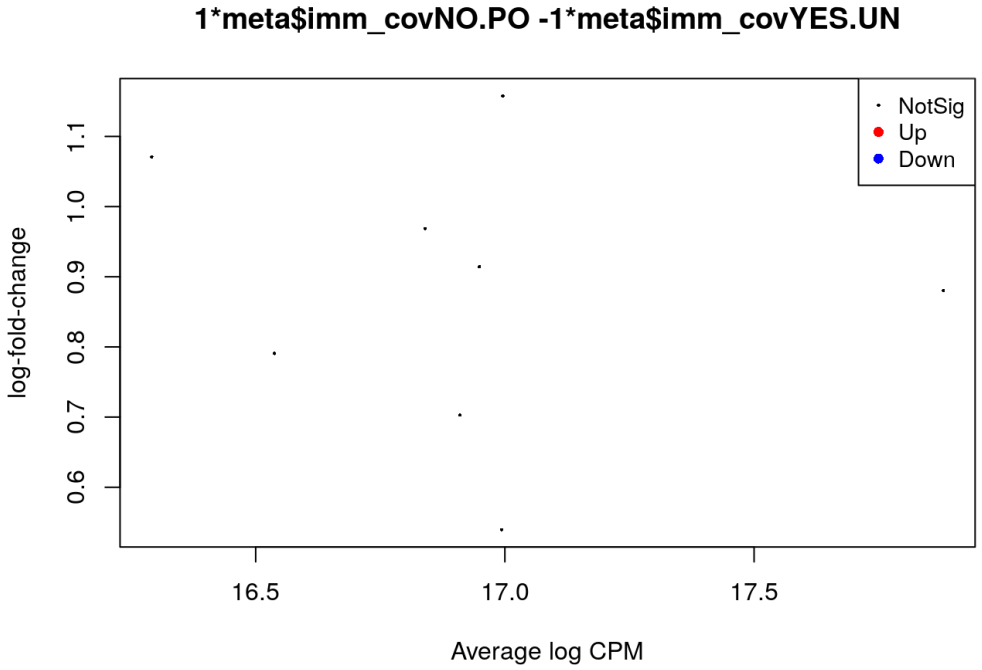
```
topTags(qlf1)
```

```
## Coefficient:  meta$imm_covNO.PO
##                  logFC   logCPM        F       PValue          FDR
## KCNQ1OT1     -8.357830 16.29143 60.40131 1.129776e-13 4.675805e-13
## MT-ND4       -8.378832 16.53798 60.32123 1.168951e-13 4.675805e-13
## MT-CO1       -8.195837 16.83954 58.07694 3.048465e-13 8.129240e-13
## REXO1L1P     -7.912438 16.90980 55.43164 9.513326e-13 1.899354e-12
## MT-ND5       -7.891803 16.99582 54.91945 1.187096e-12 1.899354e-12
## CTD-2328D6.1 -7.718862 16.99390 53.47452 2.221009e-12 2.742945e-12
## MT-RNR1      -7.678848 16.94866 53.29612 2.400077e-12 2.742945e-12
## MT-RNR2      -6.896365 17.87989 42.14168 3.338450e-10 3.338450e-10
```

```
# The total# of DE genes in each direction at a FDR of 5% can be
## examined with decideTests.
summary(decideTests(qlf_3vs4)) # Let's, up-, down-regulated & none signicant genes
```

```
##        1*meta$imm_covYES.PO -1*meta$imm_covYES.UN
## Down                                           0
## NotSig                                         8
## Up                                             0
```

```
# Visualize DE test results using an MD plot
plotMD(qlf_4vs1)
```



glmTreat is used to narrow down the list of DE genes and focus on biologically meaningful genes. Test whether the DE is significantly above a log2-FC of log 2 1.2, i.e., a FC of 1.2.

```
tr <- glmTreat(fit, contrast=c(0,1,-1,0), lfc=log2(1.2))
# Top set of most significant genes
kable(topTags(tr)) %>% kable_styling(font_size=7)
```

	logFC	unshrunk.logFC	logCPM	PValue	FDR
MT-ND5	0.3315134	0.3325423	16.99582	0.4857757	0.8875832
KCNQ1OT1	-0.2779619	-0.2796417	16.29143	0.5601572	0.8875832
MT-CO1	0.2550814	0.2559257	16.83954	0.5902769	0.8875832
CTD-2328D6.1	-0.2184450	-0.2191962	16.99390	0.6476200	0.8875832
REXO1L1P	-0.2032792	-0.2041573	16.90980	0.6700462	0.8875832
MT-RNR1	-0.1418071	-0.1422486	16.94866	0.7654899	0.8875832
MT-ND4	0.1118203	0.1122564	16.53798	0.8132941	0.8875832
MT-RNR2	0.0694620	0.0695638	17.87989	0.8875832	0.8875832

x	x	x
BH	1*meta$imm_covNO.UN -1*meta$imm_covYES.PO	glm

```
# Number of up-, down-regulated & none signicant genes
summary(decideTests(tr))
```

```
##        1*meta$imm_covNO.UN -1*meta$imm_covYES.PO
## Down                                           0
## NotSig                                         8
## Up                                             0
```

```
sessionInfo()
```

```
## R version 4.2.1 (2022-06-23)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: AIMS Desktop 2020.2
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/openblas/libblas.so.3
## LAPACK: /usr/lib/x86_64-linux-gnu/libopenblasp-r0.3.5.so
##
## locale:
##  [1] LC_CTYPE=en_ZA.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_ZA.UTF-8        LC_COLLATE=en_ZA.UTF-8
##  [5] LC_MONETARY=en_ZA.UTF-8    LC_MESSAGES=en_ZA.UTF-8
##  [7] LC_PAPER=en_ZA.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_ZA.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats4    stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] EDASeq_2.30.0               ShortRead_1.54.0
##  [3] GenomicAlignments_1.32.1    Rsamtools_2.12.0
##  [5] Biostrings_2.64.1           XVector_0.36.0
##  [7] BiocParallel_1.30.4         edgeR_3.38.4
##  [9] limma_3.52.4                kableExtra_1.3.4
## [11] knitr_1.41                  forcats_0.5.2
## [13] stringr_1.5.0               dplyr_1.0.10
## [15] purrr_1.0.1                 tidyr_1.2.1
## [17] tibble_3.1.8                tidyverse_1.3.2
## [19] pheatmap_1.0.12             RColorBrewer_1.1-3
## [21] ggrepel_0.9.2               ggplot2_3.4.0
## [23] reshape_0.8.9               biomaRt_2.52.0
## [25] DESeq2_1.36.0               SummarizedExperiment_1.26.1
## [27] Biobase_2.56.0              MatrixGenerics_1.8.1
## [29] matrixStats_0.63.0          GenomicRanges_1.48.0
## [31] GenomeInfoDb_1.32.4         IRanges_2.30.1
## [33] S4Vectors_0.34.0            BiocGenerics_0.42.0
## [35] readr_2.1.3                 tximport_1.24.0
##
## loaded via a namespace (and not attached):
##   [1] readxl_1.4.1           backports_1.4.1        aroma.light_3.26.0
##   [4] BiocFileCache_2.4.0    systemfonts_1.0.4      plyr_1.8.8
##   [7] splines_4.2.1          digest_0.6.31          htmltools_0.5.4
##  [10] fansi_1.0.4            magrittr_2.0.3         memoise_2.0.1
##  [13] googlesheets4_1.0.1    tzdb_0.3.0             annotate_1.74.0
##  [16] modelr_0.1.10          R.utils_2.12.2         vroom_1.6.1
##  [19] svglite_2.1.1          bdsmatrix_1.3-6        timechange_0.2.0
##  [22] prettyunits_1.1.1      jpeg_0.1-10            colorspace_2.0-3
##  [25] blob_1.2.3             rvest_1.0.3            rappdirs_0.3.3
##  [28] apeglm_1.18.0          haven_2.5.1            xfun_0.36
##  [31] crayon_1.5.2           RCurl_1.98-1.9         jsonlite_1.8.4
##  [34] genefilter_1.78.0      survival_3.5-0         glue_1.6.2
##  [37] gtable_0.3.1           gargle_1.2.1           zlibbioc_1.42.0
##  [40] webshot_0.5.4          DelayedArray_0.22.0    scales_1.2.1
##  [43] mvtnorm_1.1-3          DBI_1.1.3              Rcpp_1.0.10
##  [46] viridisLite_0.4.1      xtable_1.8-4           progress_1.2.2
##  [49] emdbook_1.3.12         bit_4.0.5              httr_1.4.4
##  [52] ellipsis_0.3.2         R.methodsS3_1.8.2      pkgconfig_2.0.3
##  [55] XML_3.99-0.13          farver_2.1.1           deldir_1.0-6
##  [58] sass_0.4.4             dbplyr_2.3.0           locfit_1.5-9.7
##  [61] utf8_1.2.2             tidyselect_1.2.0       labeling_0.4.2
##  [64] rlang_1.0.6            AnnotationDbi_1.58.0   munsell_0.5.0
##  [67] cellranger_1.1.0       tools_4.2.1            cachem_1.0.6
##  [70] cli_3.6.0              generics_0.1.3         RSQLite_2.2.20
##  [73] broom_1.0.2            evaluate_0.20          fastmap_1.1.0
##  [76] yaml_2.3.6             bit64_4.0.5            fs_1.5.2
##  [79] KEGGREST_1.36.3        R.oo_1.25.0            xml2_1.3.3
##  [82] compiler_4.2.1         rstudioapi_0.14        filelock_1.0.2
##  [85] curl_5.0.0             png_0.1-8              reprex_2.0.2
##  [88] geneplotter_1.74.0     bslib_0.4.2            stringi_1.7.12
```

```
##  [91] highr_0.10             GenomicFeatures_1.48.4 lattice_0.20-45
##  [94] Matrix_1.5-3           vctrs_0.5.1            pillar_1.8.1
##  [97] lifecycle_1.0.3        jquerylib_0.1.4        bitops_1.0-7
## [100] rtracklayer_1.56.1     BiocIO_1.6.0           latticeExtra_0.6-30
## [103] R6_2.5.1               hwriter_1.3.2.1        codetools_0.2-18
## [106] MASS_7.3-58.1          assertthat_0.2.1       rjson_0.2.21
## [109] withr_2.5.0            GenomeInfoDbData_1.2.8 parallel_4.2.1
## [112] hms_1.1.2              grid_4.2.1             coda_0.19-4
## [115] rmarkdown_2.20         googledrive_2.0.0      bbmle_1.0.25
## [118] numDeriv_2016.8-1.1    lubridate_1.9.0        restfulr_0.0.15
## [121] interp_1.1-3
```