RNA seq analysis for Clinical Neuro samples: v1

Ephifania Geza

25/10/2022

How different is v1 from v0

In version one (v1), instead of considering the different patients conditions that is COVID-19, encephalopathy or immunosuppression, patients belong to one of the eight analysis groups (1, 2, 3, 4, 5, 6, 7, 8).

Best practices for differentail expression analyses: DESeq2

We report the results for differential expression analysis using the **DESeq2** tool. This analysis involves 43 clinical samples. We aligned the raw sequence reads (paired) to the reference genome **GRCh37** using the *STAR* alignment tool. The number of reads that mapped to each gene were counted using a pseudo-count method, **SALMON**. Quality check, read trimming, mapping and counting was done using the https://github.com/nf-core/rnaseq pipeline. While count normalization (creation of the DESeqDataSet from a matrix), exploratory data analysis (identifying outliers & sources of variation in the data), estimation of size factors (estimateSizeFactors), estimation of dispersion (estimateDispersions), Negative Binomial GLM fitting and Wald statistics, checking of the dispersion estimates (plotDispEsts), creating contrasts to perform Wald testing on the shrunken log2 foldchanges between specific conditions (where necessary), visualization of results (volcano plots, heatmaps, normalized counts plots of top genes, etc), and determining significant results was done in R.

DESeq2 can take the **tximport** object as input, as such we first convert the **SALMON** counts to this object.

All count data directories contain the pattern "_L001"

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

Since all count data (quant) files have the same name the *sample* variable should be the names of each file.

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

Using the *tximport* package we import 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 3
## transcripts missing from tx2gene: 1
## summarizing abundance
## summarizing counts
## summarizing length
```

Table 1: Metadata for the first five samples.

	immunosuppressed	COVID19_NEUR_Sx	Encephalopathy	Analysis_group
COVC01	NO	UN	YES	4
COVC02	YES	PO	YES	8
COVC06	NO	PO	YES	2
COVC08	NO	PO	NO	6
COVC09	YES	PO	NO	6

We use the provided clinical details to extract relevant metadata. The possible sample classification (conditions of interest) are given simple names: { *Possible*: *PO*, *Unlikely*: *UN*, *Yes*: *YES*, *No*: *NO*}. The structure of our metadata is as follows

```
str(meta)
```

```
## 'data.frame':    43 obs. of  4 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 ...
##  $ Analysis_group  : Factor w/ 8 levels "1","2","3","4",..: 4 8 2 6 6 7 1 8 8 1 ...
```

As highlighted earlier, when determining the genes that are differently expressed between conditions, the analysis can be split into two:

- Quality checking (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 the conditions).

Differential expression analysis given the different eight (8) analysis groups

Each of the clinical samples belongs to a specific group: *1, 2, 3, 4, 5, 6, 7, 8*. In this section we assume that the base/reference group is 4.

We provide the first 5 rows of our metadata in the given table, so that we know if we have the right comparisons.

```
knitr::kable(head(meta, n = 5, tidy=TRUE), caption = "Metadata for the first five samples.") %>%
  kable_styling(full_width = F, bootstrap_options = c("striped", "hover", "condensed"),
  font_size = 8) %>% row_spec(0, font_size=7)
```

The number of genes we have before filtering genes with low/no count is

```
nrow(dds)
```

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

Upon filtering genes with ten (10) or less counts across all samples,

```
# Number of rows after filtering
nrow(dds)
```

```
## [1] 25860
```

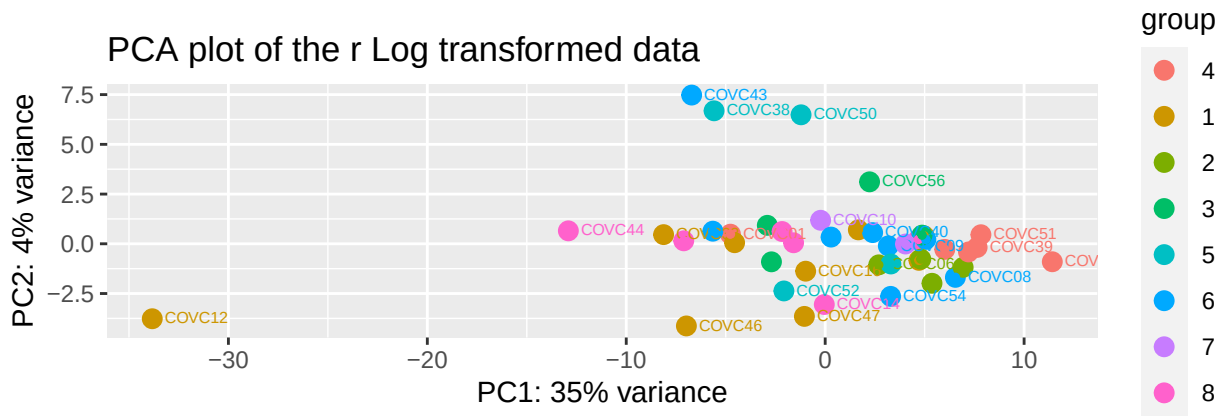
genes remained.

1. Exploratory Data Analysis

We first transform counts for data visualization. We use the *rlog* transform. We determine whether the differences between groups is greater than differences within groups using the PCA and the MDS plot.

```
plotPCA(rld, intgroup="condition", ntop = 500) +
  ggtitle("PCA plot of the r Log transformed data") +
  geom_text(aes(label=name), size=2, hjust=-0.2, vjust=0.4,
      check_overlap = T,fontface=0.5) +  geom_point(size=1)
```

a. The PCA plot



From the PCA plot, samples in same group do not cluster together, which possibly indicates that the samples in the same group vary greatly than the variations observed between samples in different groups, implying differential expression may not be greater than the variance and cannot be easily detected.

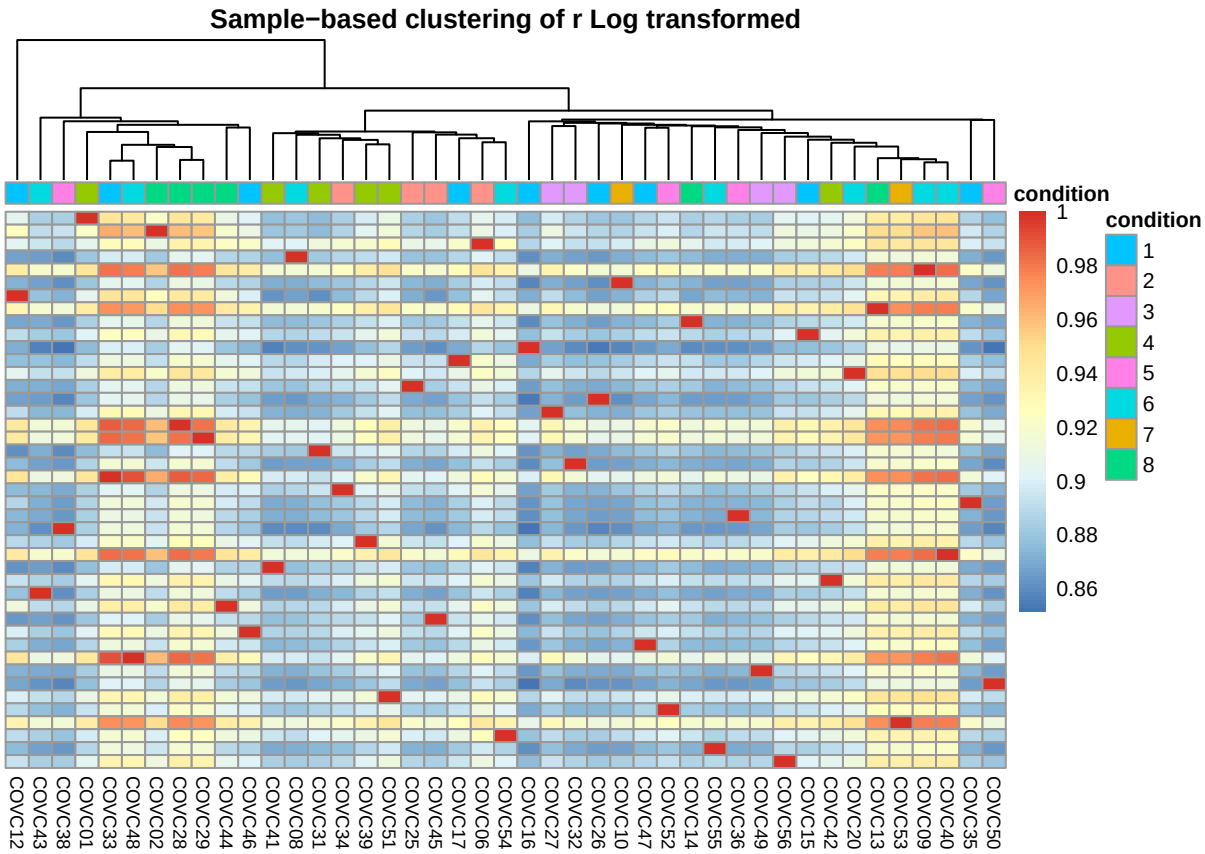
b. The MDS plot Attached to this report is interactive MDS plot in *.html* format, it was generated using the **GLIMMA** package.

```
# creates ma-plot.html in working directory
# link to it in Rmarkdown using [MA-plot](ma-plot.html)
htmlwidgets::saveWidget(glimmaMDS(dds), "mds-plot.html")
```

c. Sample- and gene-based clustering

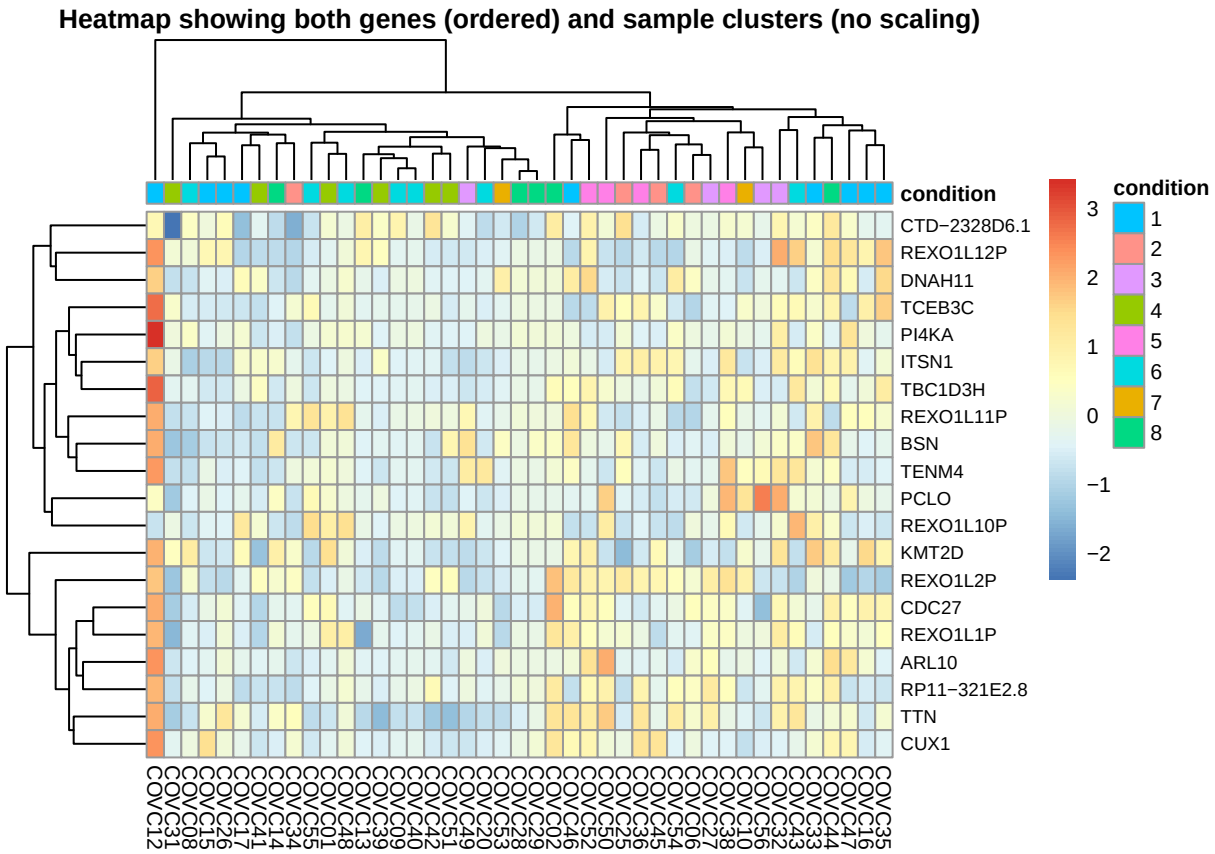
Sample-based clustering (Clustering of samples by gene expression) We cluster samples using the distance between samples based on the Pearson's correlation. Highest correlation value shows most correlated samples, those with more similar expression profiles for all transcripts.

```
# Extract the rlog matrix from the object
rld_mat <- assay(rld)
## Compute pairwise correlation values (Pearson, the default)
rld_cor <- cor(rld_mat)
pheatmap(rld_cor, cluster_rows = FALSE, clustering_distance_cols = sampleDists,
  show_rownames = F, fontsize = 8, fontsize_row = 7, fontsize_col = 7,
  annotation_col = sampleTable, main="Sample-based clustering of r Log transformed")
```



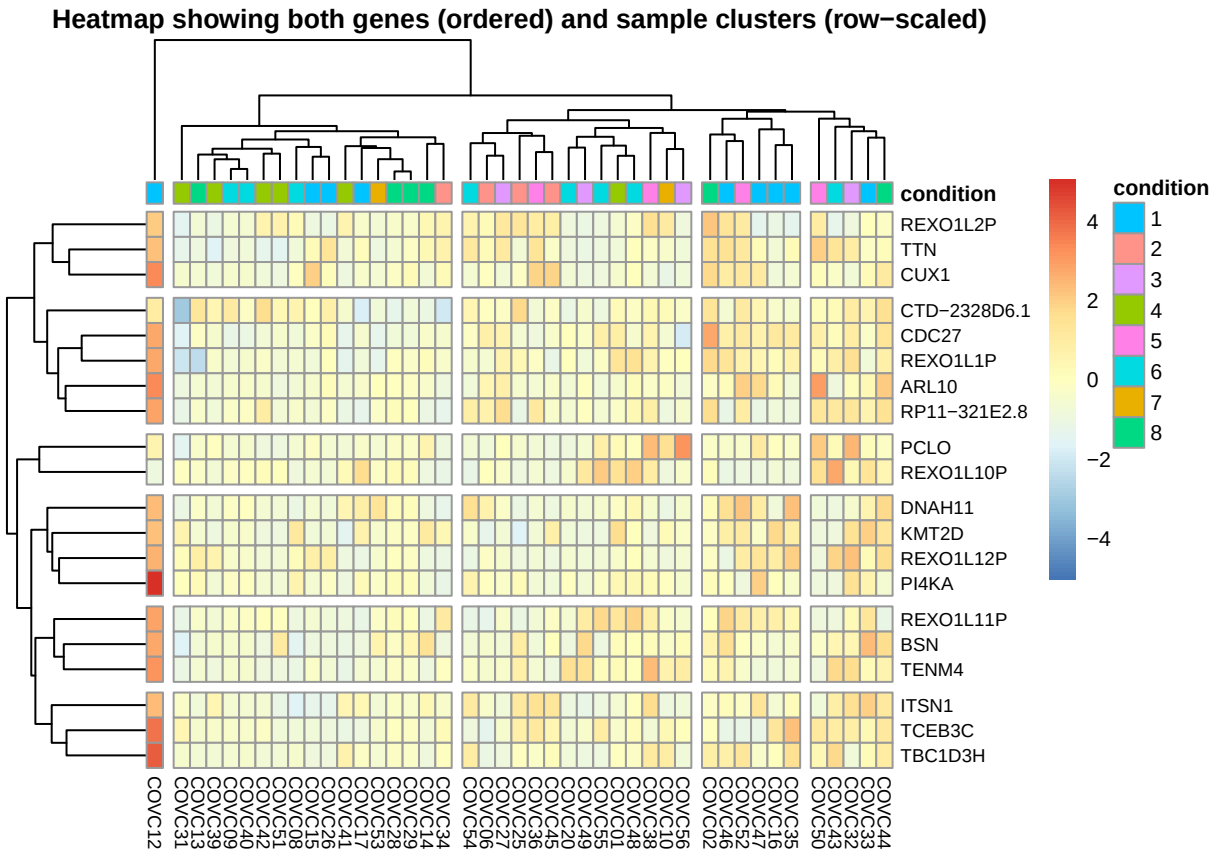
Gene- and sample-based clustering Gene-and sample-based clustering combines the heatmap (clustering of genes with similar expression patterns) & dendrogram of samples (how samples with similar gene expression cluster).

```
topVarGenes <- head(order(rowVars(assay(rld)), decreasing = T), 20)
mat  <- assay(rld)[ topVarGenes, ]
mat  <- mat - rowMeans(mat)
# Dendrogram and Heapmap
pheatmap(mat , annotation_col = sampleTable, fontsize = 8, fontsize_row = 7,
    main = "Heatmap showing both genes (ordered) and sample clusters (no scaling)")
```



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 across col. Here we also customize annotation colors.

```
pheatmap(mat, fontsize = 8, fontsize_row = 7, fontsize_col = 7, scale = "row",
    cutree_rows = 6, annotation_col = sampleTable, cutree_cols=5,
    main = "Heatmap showing both genes (ordered) and sample clusters (row-scaled)")
```



3. Library normalisation, dispersion estimation and the Wald test

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) fit models and get statistics for each gene for the design specified the data is imported.

When considering how genes are expressed between different analysis groups, we have the following contrasts

```
resultsNames(dds)
```

```
## [1] "Intercept"        "condition_1_vs_4" "condition_2_vs_4" "condition_3_vs_4"
## [5] "condition_5_vs_4" "condition_6_vs_4" "condition_7_vs_4" "condition_8_vs_4"
```

Testing different hypotheses, creating contrasts

DESeq2 adjusts the p value by different methods including the “holm”, “hochberg”, “hommel”, “bonferroni”, “BY” and Benjamini and Hochberg (“BH”) which is default. We adjusted the p-values using the “BH” method. Only groups 2 vs 4 have genes that are expressed differently (up-regulated). All other contrasts have neither up- nor down-regulated genes.

We test the following hypothesis: **H0:** Each gene in group 1 and 4 have equal expression distribution (gene is not differentially expressed vs **H1:** Genes in group 1 and 4 show significant expression distribution (they are differentially expressed).

```
res14DF <- as.data.frame(res_1vs4[order(res_1vs4$pvalue),])
table(res14DF$pvalue < 0.05)
```

```
##
```

```
## FALSE  TRUE
## 23140  2718
```

2718 genes have $p < 0.05$.

The number of the genes that are not significant, up- or down-regulated given $p < 0.1$ is given below.

```
summary(res_1vs4)
```

```
##
## out of 25860 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 919, 3.6%
## LFC < 0 (down)     : 0, 0%
## outliers [1]       : 2, 0.0077%
## low counts [2]     : 20555, 79%
## (mean count < 1)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
res24DF <- as.data.frame(res_2vs4[order(res_2vs4$pvalue),])
print(table(res24DF$pvalue < 0.05))
```

```
##
## FALSE  TRUE
## 25706   152
```

152 genes have $p < 0.05$ between group 2 and 4. And, the summary of non-significant, up- and down-regulated genes is given by

```
summary(res_2vs4)
```

```
##
## out of 25860 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 0, 0%
## LFC < 0 (down)     : 0, 0%
## outliers [1]       : 2, 0.0077%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
res34DF <- as.data.frame(res_3vs4[order(res_3vs4$pvalue),])
print(table(res34DF$pvalue < 0.05))
```

```
##
## FALSE  TRUE
## 25328   530
```

530 genes have $p < 0.05$ between group 3 and 4. And, the summary of non-significant, up- and down-regulated genes is given by

```
summary(res_3vs4)
```

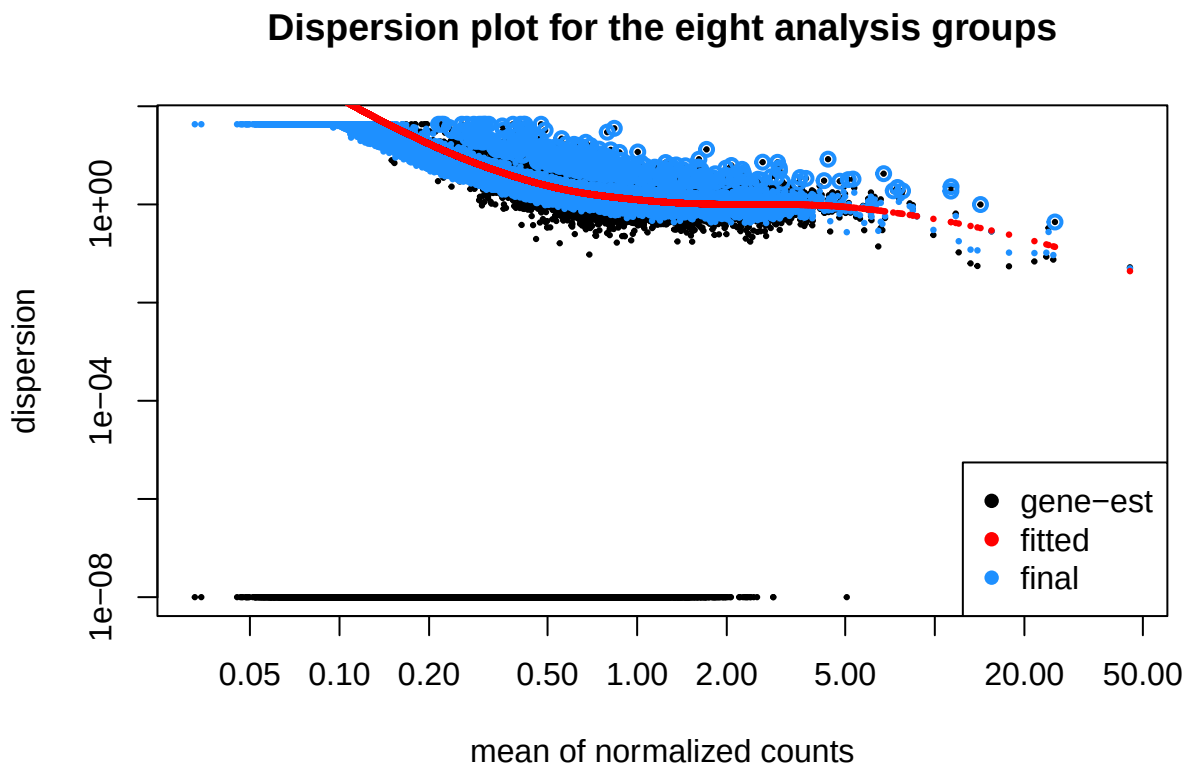
```
##
## out of 25860 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 0, 0%
## LFC < 0 (down)     : 0, 0%
```

```
## outliers [1]       : 2, 0.0077%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
summary(res_8vs4)
```

```
##
## out of 25860 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 0, 0%
## LFC < 0 (down)     : 0, 0%
## outliers [1]       : 2, 0.0077%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
plotDispEsts(dds, main="Dispersion plot for the eight analysis groups")
```

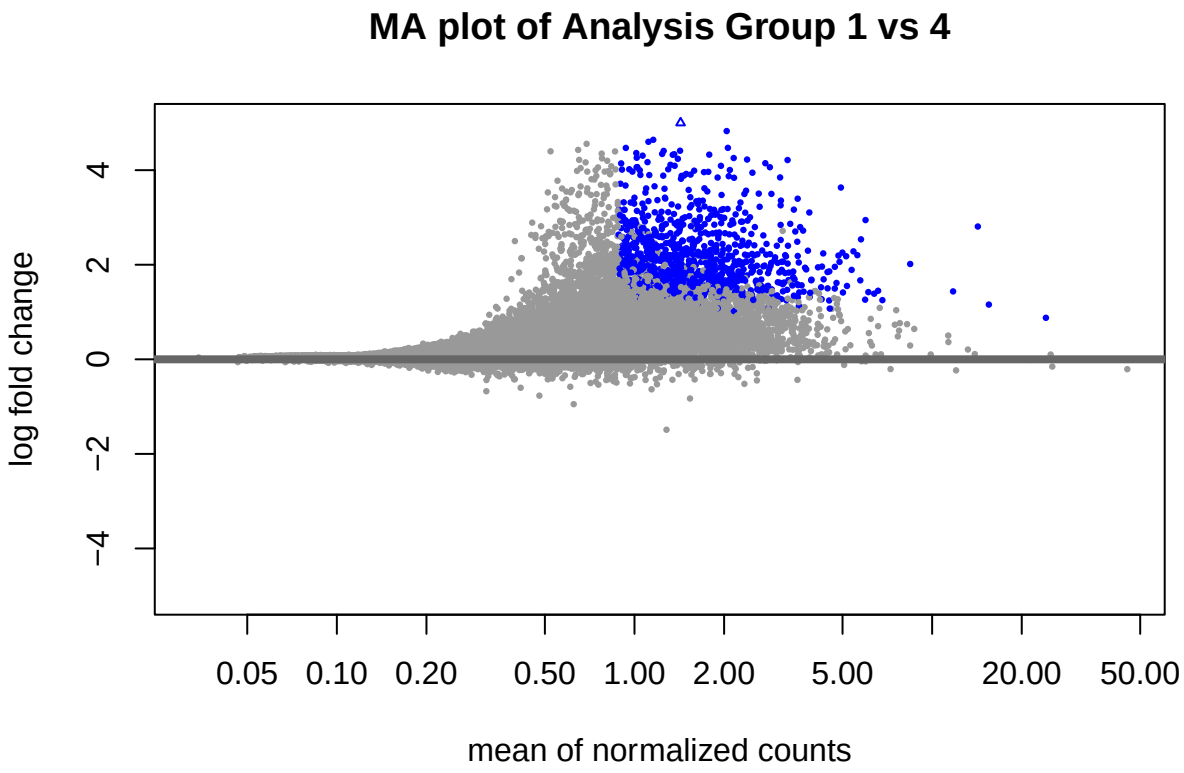


Shrinkage of effect size (LFC) is used for visualization and ranking genes We use the "apeglm" method for effect size shrinkage as it improves the estimator when specified.

```
MAres_1vs4 <- lfcShrink(dds, coef="condition_1_vs_4", 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_1vs4, ylim=c(-5, 5), main="MA plot of Analysis Group 1 vs 4")
```



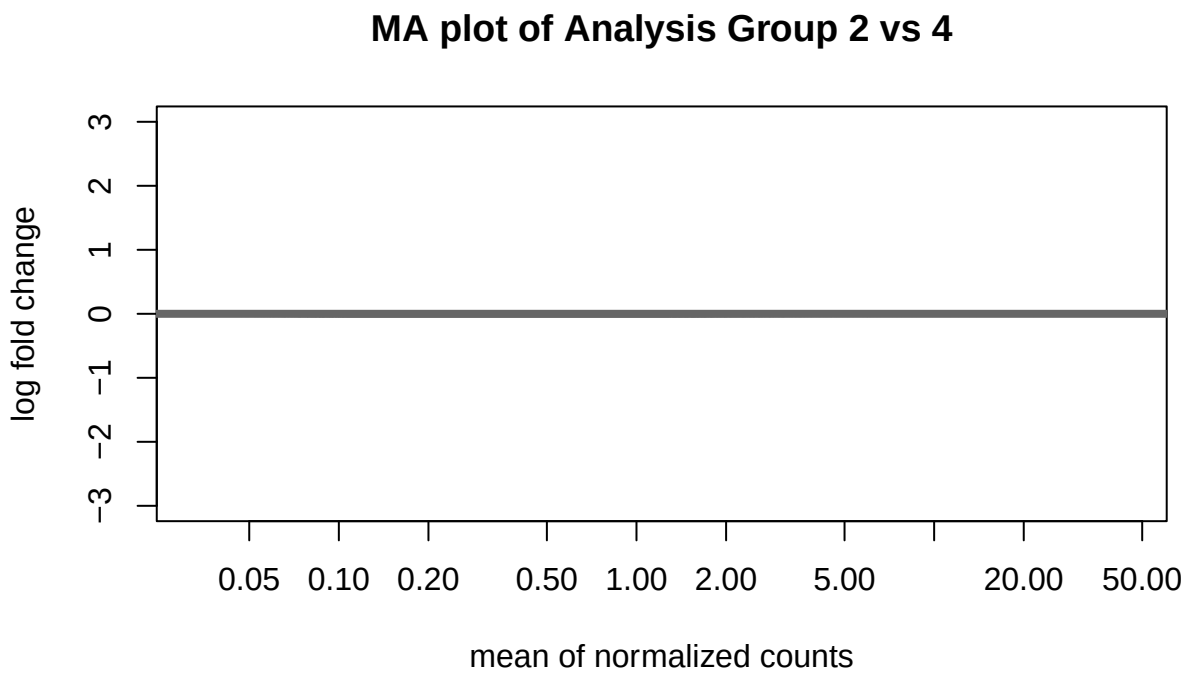

```
#htmlwidgets::saveWidget(glimmaMA(dds), "ma-plot-glimma.html")
```

From the MA plot of group 1 and 4, the points in grey represent the genes that are not significant despite having $M \neq 0$. The points in blue with $M \geq 0$ are up-regulated. We do not have down-regulated genes since there are no blue points with $M \leq 0$.

```
MAres_2vs4 <- lfcShrink(dds, coef="condition_2_vs_4", 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_2vs4, ylim=c(-3, 3), main="MA plot of Analysis Group 2 vs 4")
```

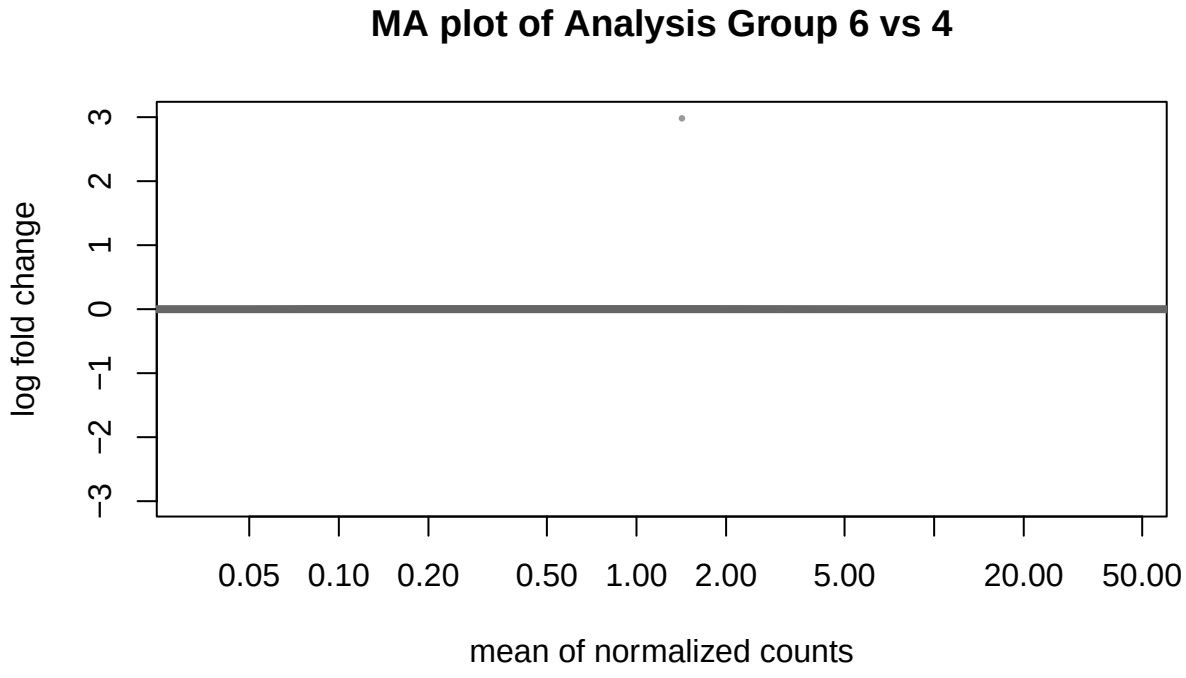


All genes have no significant difference between group 2 and 4.

```
MAres_6vs4 <- lfcShrink(dds, coef="condition_6_vs_4", 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_6vs4, ylim=c(-3, 3), main="MA plot of Analysis Group 6 vs 4")
```

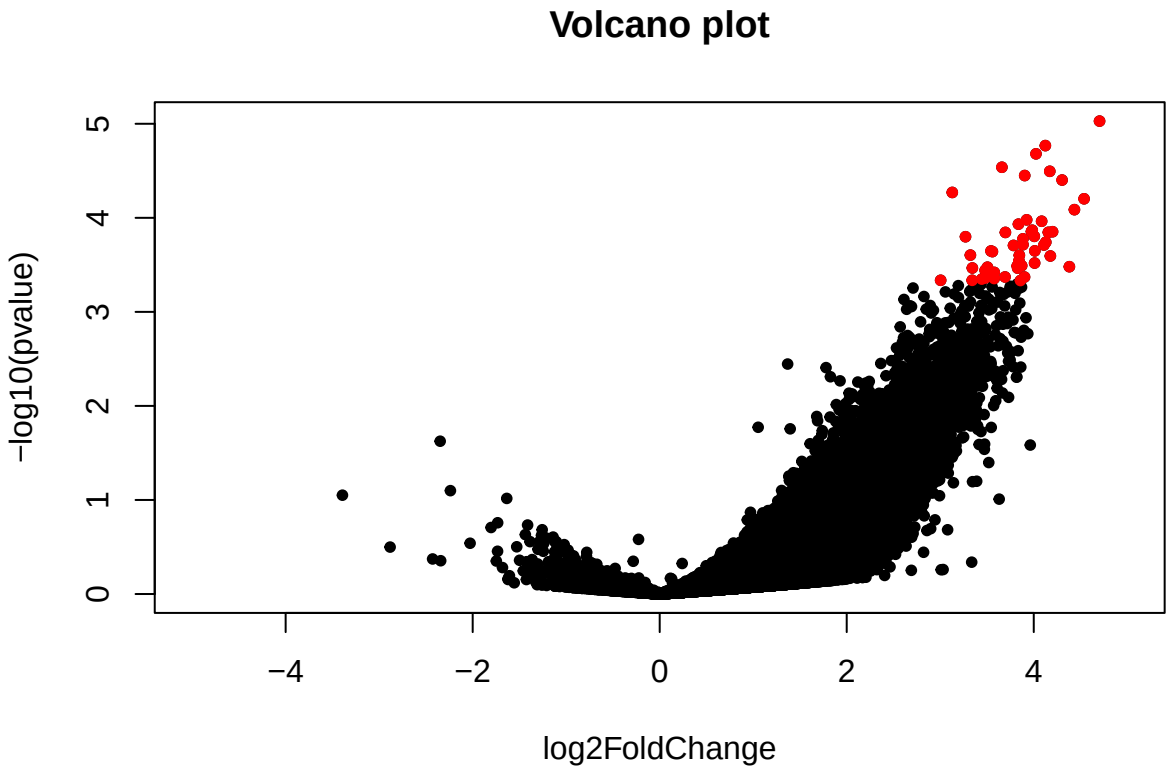


We checked all the other groups and just as between group 2 and 4, all genes have no significant difference

between the tested groups.

```
with(res_1vs4_DF, plot(log2FoldChange, -log10(pvalue), pch=20, main="Volcano plot", xlim=c(-5,5)))

with(subset(res_1vs4_DF, padj<.05 ), points(log2FoldChange, -log10(pvalue), pch=20, col="red"))
```

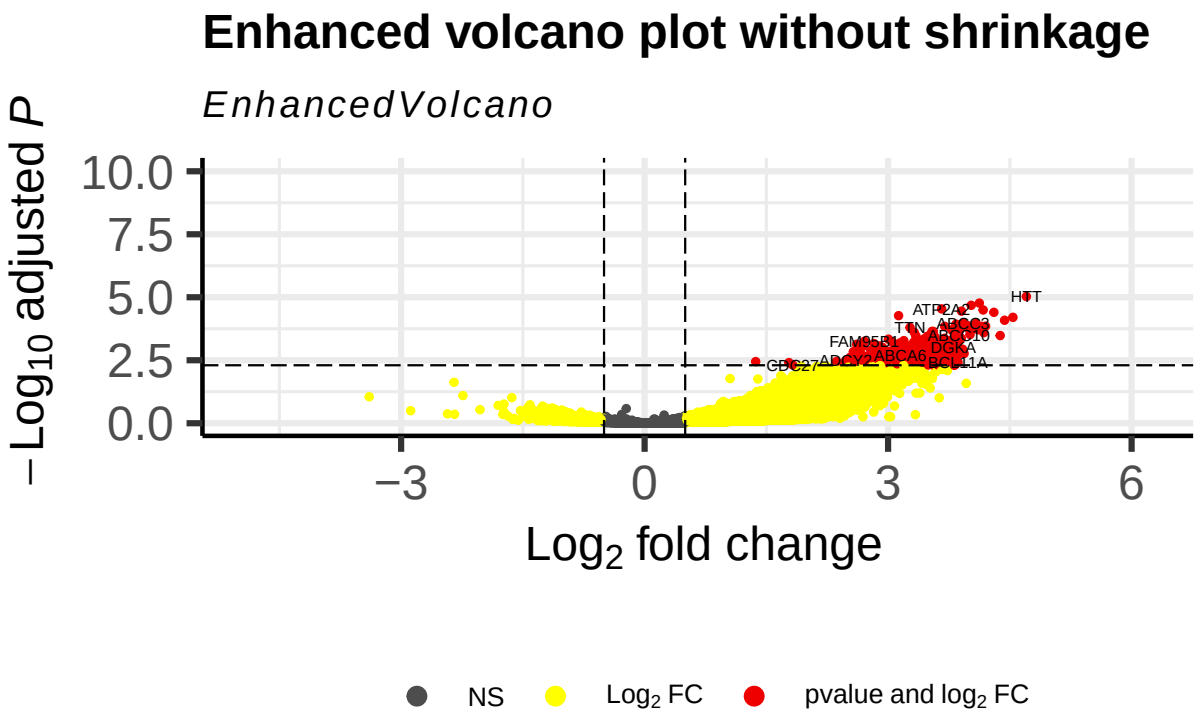


Volcano plot

```
##WE CAN USE res in place of DGE.results
dev.new(width=15, height=15)
vp1 <- EnhancedVolcano(res_1vs4, lab = rownames(res_1vs4), x = 'log2FoldChange',
  y = 'pvalue', pointSize = 1.0, labSize = 2.0, pCutoff = 0.005,
  xlab = bquote(~Log[2] ~ "fold change"), ylab = bquote(~-Log[10] ~adjusted~italic(P)),
  pCutoffCol = 'pvalue', FCcutoff = 0.5, cutoffLineType = "longdash",
  cutoffLineCol = "black", cutoffLineWidth = 0.4, col = c("grey30", "yellow",
    "royalblue", "red2"), colAlpha=1, legendLabels = c("NS", expression(Log[2] ~ FC),
  "pvalue",   expression(pvalue ~ and ~ log[2] ~ FC)), legendPosition = 'bottom',
  legendLabSize = 10, legendIconSize = 3.0, title = 'Enhanced volcano plot without shrinkage')

vp2 <- EnhancedVolcano(MAres_1vs4, lab = rownames(MAres_1vs4),
      x = 'log2FoldChange', y = 'pvalue', pCutoff = 0.005,
      legendPosition = 'bottom', pointSize = 1.0, labSize = 1.75,
      legendLabSize = 10, legendIconSize = 3.0, FCcutoff = 0.5,
      title = "Enhanced volcano plot with logFC shrinkage")
```

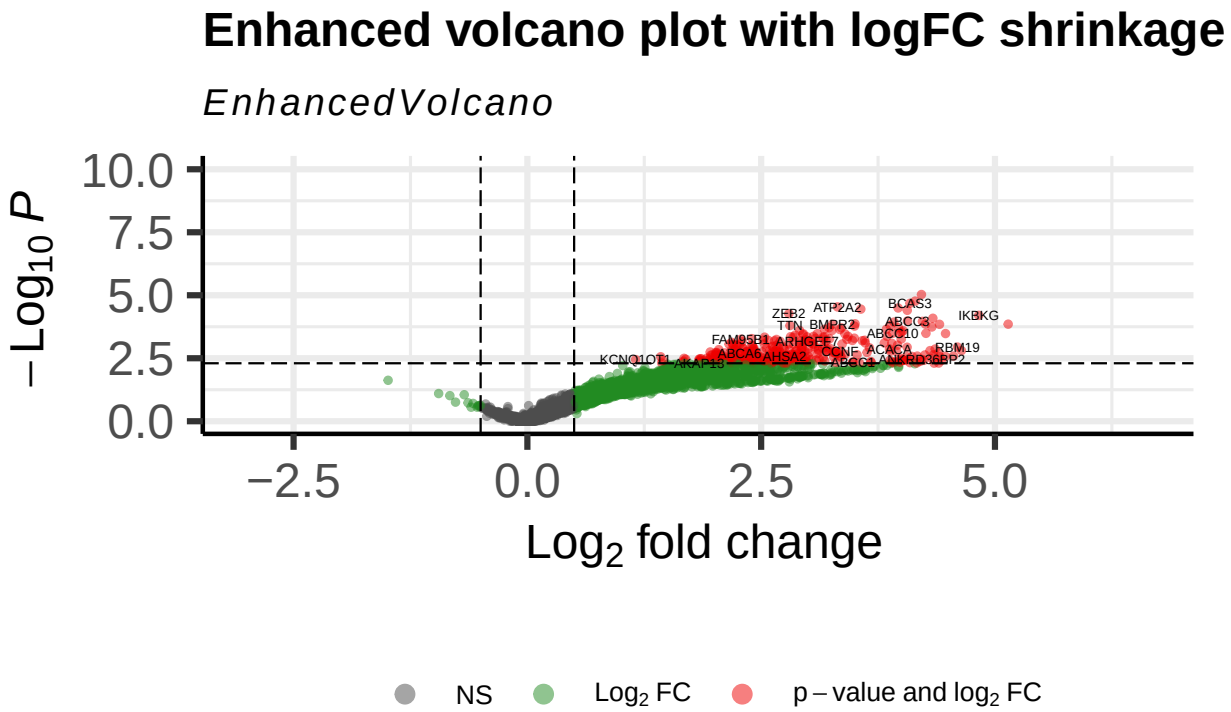
```
print(vp1)
```



total = 25860 variables

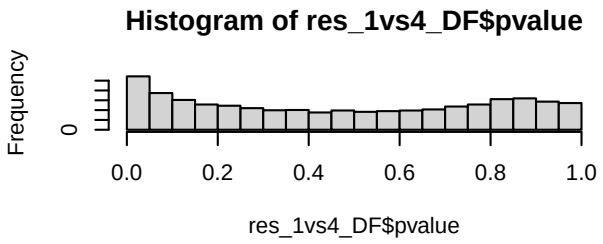
```
#knitr::include_graphics(paste(wkdir,'enhancedvolcano.png',sep=""))
#htmlwidgets::saveWidget(glimmaVolcano(dds),"volcano-glimma.html")
```

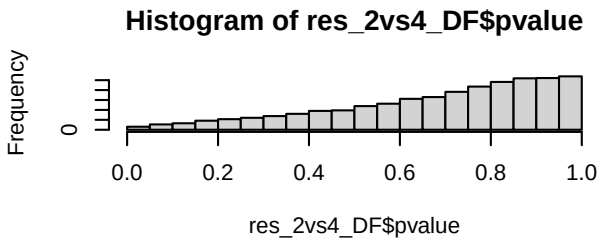
```
print(vp2)
```

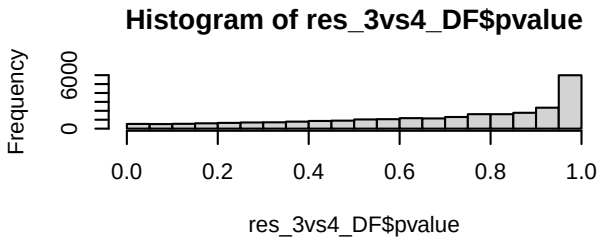


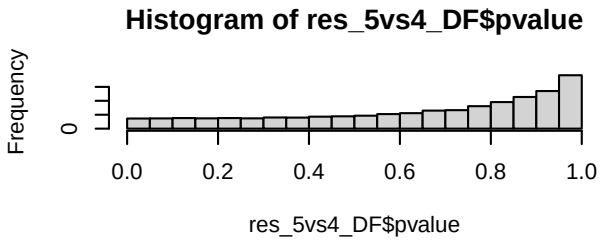
total = 25860 variables

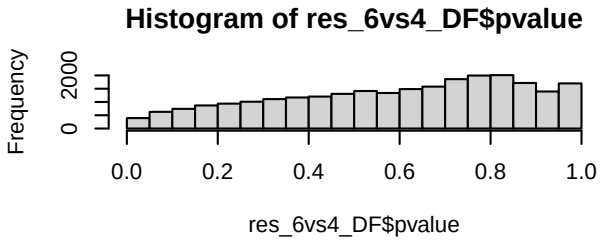
```
par(mfrow=c(3,2))
hist(res_1vs4_DF$pvalue)
hist(res_2vs4_DF$pvalue)
hist(res_3vs4_DF$pvalue)
hist(res_5vs4_DF$pvalue)
hist(res_6vs4_DF$pvalue)
hist(res_7vs4_DF$pvalue)
```

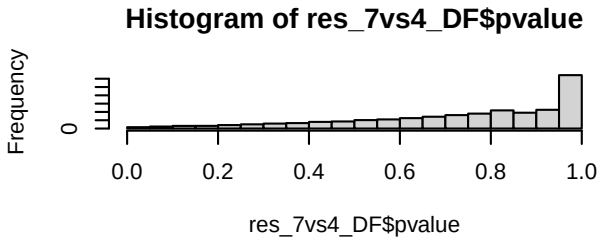






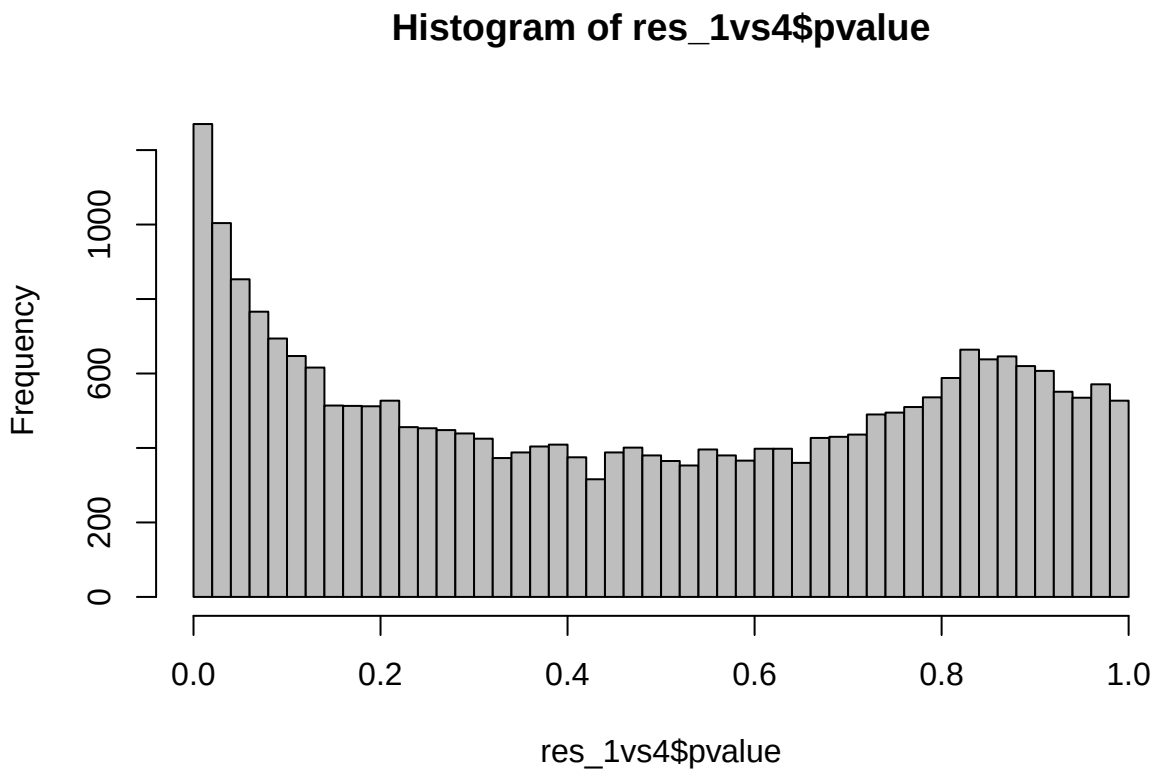







```
#hist(res_8vs4_DF$pvalue)
```

```
res_1vs4 <- results(dds, contrast = c("condition","1","4"))
hist(res_1vs4$pvalue, breaks=50, col="grey")
```



The number of genes that are significantly different between group 1 and 4 considering a padj < 0.05 and abs(log2FoldChange) > 0.58 cutoff are:

```
print(table(res_1vs4_DF$padj < 0.05 & abs(res_1vs4_DF$log2FoldChange) > 0.58))
```

```
##
## FALSE  TRUE
##  9667    50
```

We considered significant genes as those that have passed the following thresholds: padj < 0.05 and abs(log2FoldChange) > 0.58 and save these genes in a file under the name **res_sig_genes_1vs4.csv**.

```
summary(res_1vs6)
```

```
##
## out of 25860 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 0, 0%
## LFC < 0 (down)     : 0, 0%
## outliers [1]       : 2, 0.0077%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
summary(res_2vs6)
```

```
##
## out of 25860 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 0, 0%
## LFC < 0 (down)     : 0, 0%
```

```
## outliers [1]       : 2, 0.0077%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
summary(res_3vs6)
```

```
##
## out of 25860 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 0, 0%
## LFC < 0 (down)     : 0, 0%
## outliers [1]       : 2, 0.0077%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
summary(res_5vs6)
```

```
##
## out of 25860 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 0, 0%
## LFC < 0 (down)     : 0, 0%
## outliers [1]       : 2, 0.0077%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
summary(res_7vs6)
```

```
##
## out of 25860 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 0, 0%
## LFC < 0 (down)     : 0, 0%
## outliers [1]       : 2, 0.0077%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
summary(res_8vs6)
```

```
##
## out of 25860 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 0, 0%
## LFC < 0 (down)     : 0, 0%
## outliers [1]       : 2, 0.0077%
## 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

```
txi_cov <- tximport(files_cov, 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
## transcripts missing from tx2gene: 1
## summarizing abundance
## summarizing counts
## summarizing length
```

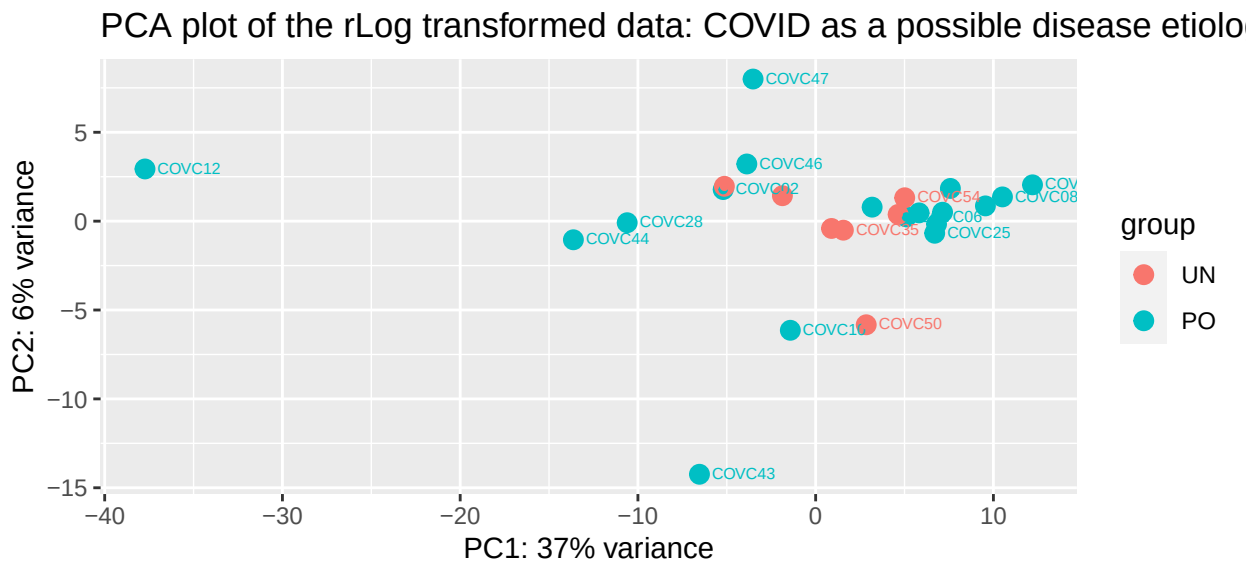
It is important to note that the clinical details are used in the extraction of the relevant metadata.

Checking if the row names of metadata are now the same as column names of the counts

```
rld_cov <- rlog(dds_cov, fitType = "local", blind=FALSE)
```

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

```
plotPCA(rld_cov, intgroup="condition_cov", ntop = 500) +
  ggtitle("PCA plot of the rLog transformed data: COVID as a possible disease etiology") +
  geom_text(aes(label=name), size=2,hjust=-0.2, vjust=0.4, check_overlap = T,
      fontface=1) +  geom_point(size=1)
```




```
# Extract the rlog matrix from the object
rld_cov_mat <- assay(rld_cov)
```

```
# creates ma-plot.html in working directory
# link to it in Rmarkdown using [MA-plot](ma-plot.html)
htmlwidgets::saveWidget(glimmaMDS(dds_cov, groups = sampleTable_cov,
            labels = rownames(sampleTable_cov)), "mds-plot_cov.html")
```

```
resultsNames(dds_cov)
```

```
## [1] "Intercept"              "condition_cov_PO_vs_UN"
```

The summary of up- and down-regulated genes for

```
summary(res_cov)
```

```
##
## out of 24561 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 0, 0%
## LFC < 0 (down)     : 0, 0%
## outliers [1]       : 14, 0.057%
## low counts [2]     : 0, 0%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

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

```
res_cov_Ordered <- res_cov[order(res_cov$padj),]
res_cov_OrderedDF <- as.data.frame(res_cov_Ordered)
write.csv(res_cov_OrderedDF, file = paste(covid,"res_cov_results.csv",sep = ""))
```

Creating a DGEList object for use in edgeR.

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

```
##    Mode   FALSE    TRUE
## logical   55756      17
```

Based on **edgeR**, we only have

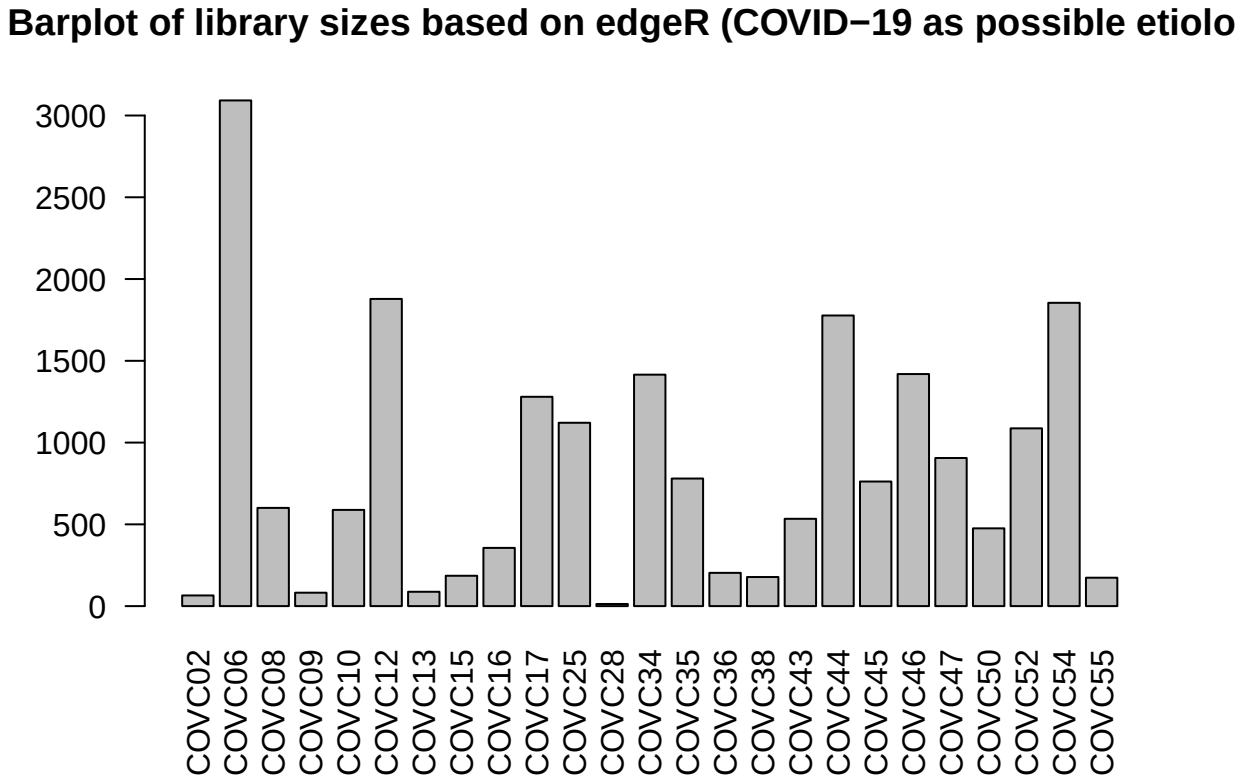
```
dim(y)[1]
```

```
## [1] 17
```

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

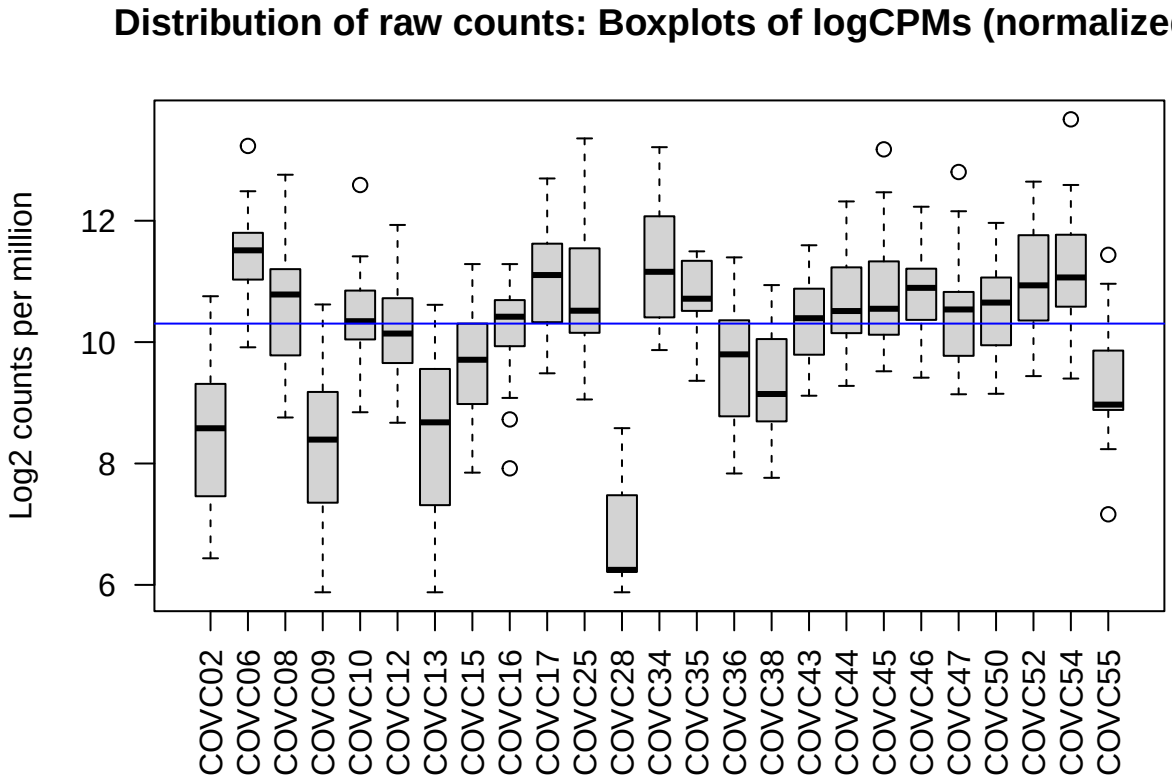
As compared to the median of ratio's (DESeq2 normalization), in **edgeR**, trimmed mean of M-values (TMM) is the default normalization. It is performed to eliminate composition biases between libraries.

```
barplot(y$samples$lib.size, names=colnames(y), las=2,
    main = "Barplot of library sizes based on edgeR (COVID-19 as possible etiology)")
```



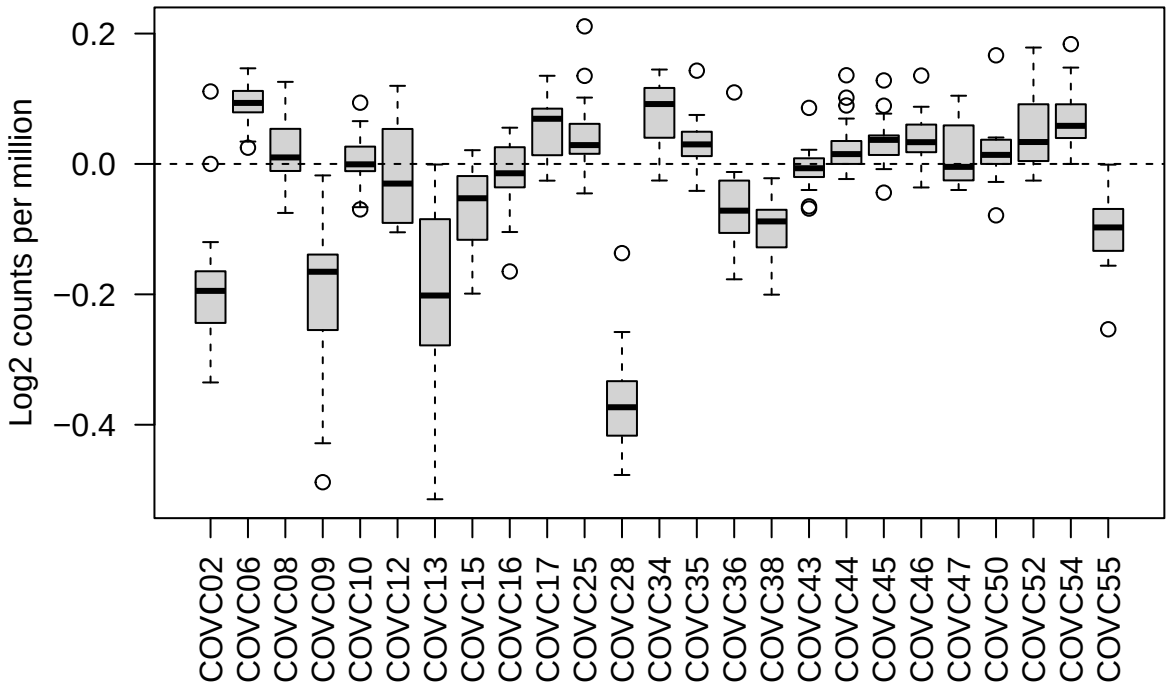
We use the cpm function to get log2 counts per million, which are corrected for the different library sizes. The cpm function also adds a small offset to avoid taking log of zero.

```
logcounts <- cpm(y,log = TRUE)
boxplot(logcounts, xlab="", ylab="Log2 counts per million", las=2, main="Distribution of raw counts: Bo
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/), which is generated with plotRLE from the EDASeq package as follows.

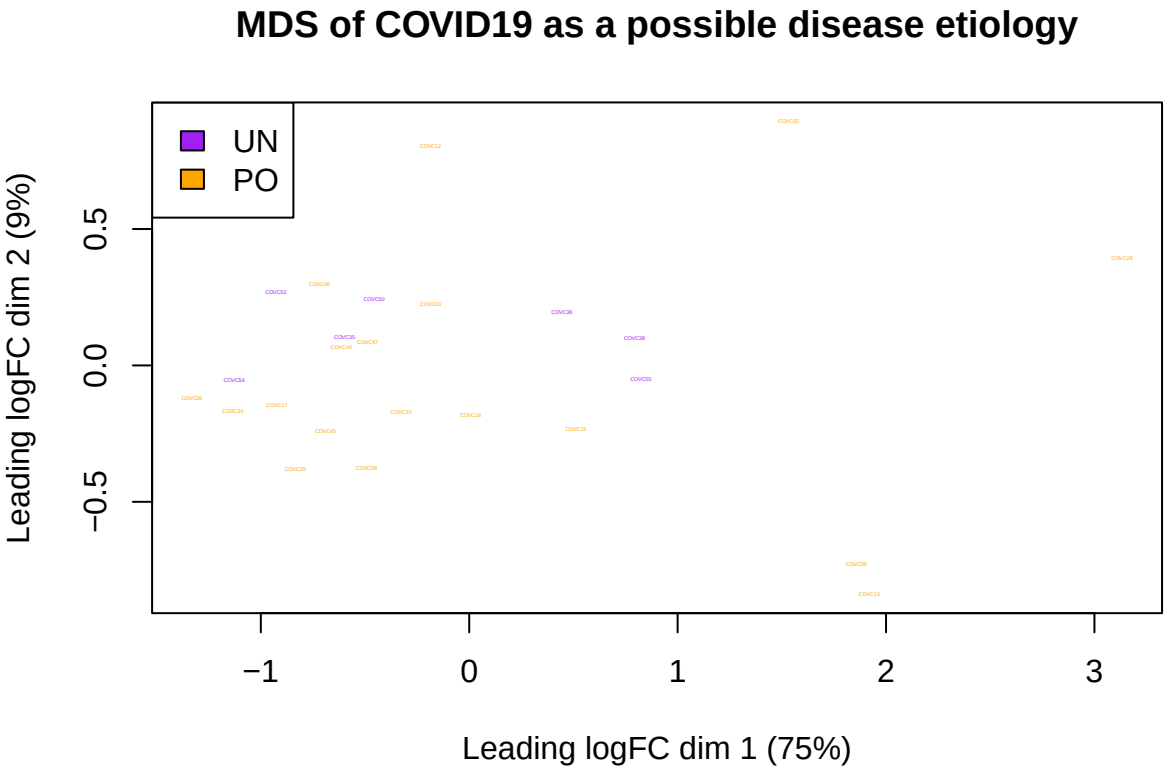
```
plotRLE(logcounts, xlab="", ylab="Log2 counts per million", las=2, main="Boxplots of logCPMs (unnormali
```




```
# # Similarly for COVID only
levels(meta_cov$COVID19_NEUR_Sx)
```

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

```
col.group <- c("purple","orange")[meta_cov$COVID19_NEUR_Sx]
plotMDS(y,col=col.group,cex = 0.2)
legend("topleft",fill=c("purple","orange"),legend=levels(meta_cov$COVID19_NEUR_Sx))
 title("MDS of COVID19 as a possible disease etiology")
```



Differential Expression

We test for significant DE in each gene, using the QL F-test - (1) We test for DE between samples whose COVID-19 are a possible or unlikely disease etiology for neurological disorder

```
y$samples$group <- group
# Estimate dispersion
dge <- estimateDisp(y)
```

```
## Using classic mode.
```

```
# Testing for dge given two conditions (single factor design)
et <- exactTest(dge)
# Get the de table
et$table
```

```
##                    logFC   logCPM    PValue
## CDC27        -0.02807759 15.52686 0.6487559
## CTD-2328D6.1  0.24160176 16.58848 0.3879437
## KCNQ1OT1     -0.05759218 15.86343 0.7708496
## MT-CO1        0.06239504 16.24328 0.7539959
## MT-CO3        0.31824442 14.87678 0.2788075
## MT-CYB       -0.10339813 15.57079 0.5938823
## MT-ND1       -0.20962749 15.23765 0.3364746
## MT-ND2        0.15022404 15.38586 0.5082435
## MT-ND4        0.14701594 15.84253 0.5666872
## MT-ND5       -0.36498271 16.38012 0.1717799
## MT-RNR1       0.11184316 16.51796 0.7421166
## MT-RNR2      -0.12371893 17.40976 0.6049921
## MUC16         0.16610568 14.84171 0.8523269
## PDE4DIP      -0.04830123 14.75322 0.5611965
```

```
## REXO1L1P      0.01772028 16.37433 1.0000000
## TRAPPC9       0.12855858 14.91599 0.9555410
## TTN           0.16019978 15.18690 0.8756068
```

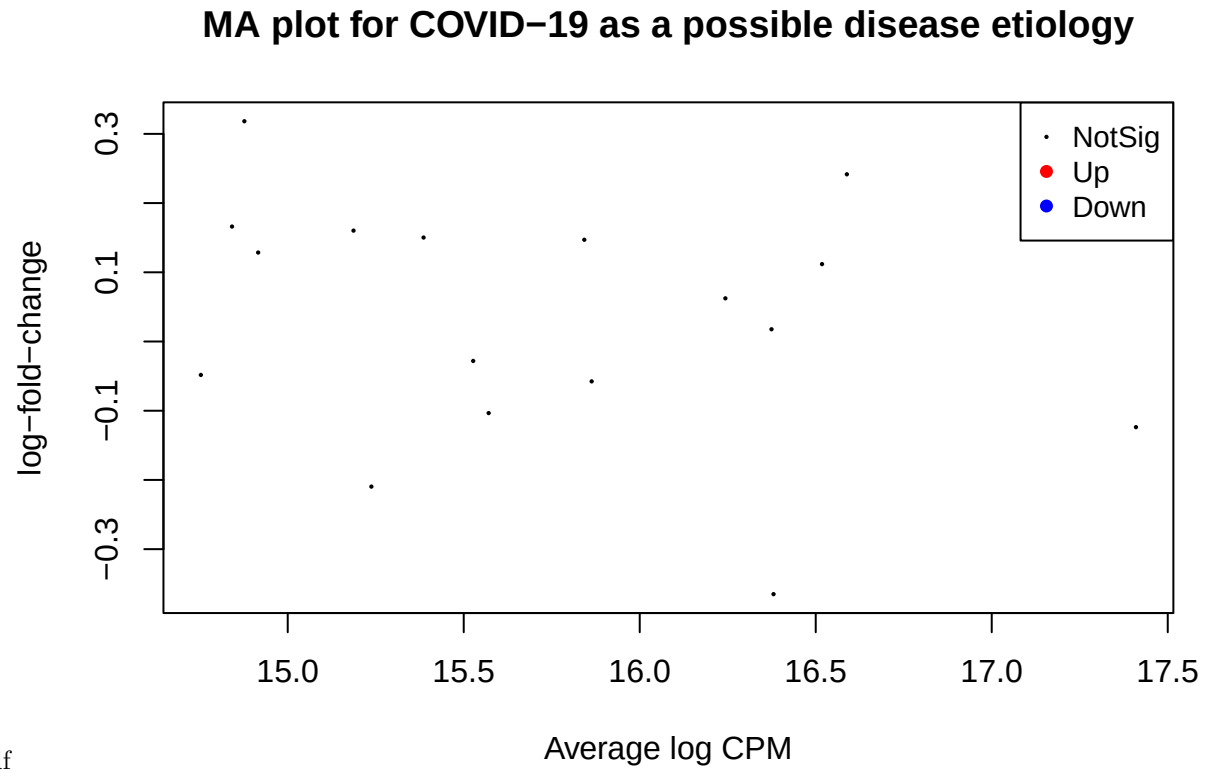
```
#Extracting table with padj (FDR)
top_degs <- topTags(et, n="Inf")
# Get number of genes Down-, up-regulated and not significant
summary(decideTests(et, lfc = 1))
```

```
##        PO-UN
## Down       0
## NotSig    17
## Up         0
```

```
# Save differential expression analysis table in a file
write.csv(as.data.frame(top_degs), file = paste(covid, "covid_possible_unlikely_dge.csv",sep=""))
```

MA plot for COVID-19 as a possible disease etiology

```
plotMD(et, main = "MA plot for COVID-19 as a possible disease etiology")
```



ma-plot-1.pdf

```
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         patchwork_1.1.2.9000
##  [9] EnhancedVolcano_1.14.0      Glimma_2.6.0
## [11] kableExtra_1.3.4            knitr_1.41
## [13] edgeR_3.38.4                limma_3.52.4
## [15] forcats_0.5.2               stringr_1.5.0
## [17] dplyr_1.0.10                purrr_1.0.1
## [19] tidyr_1.2.1                 tibble_3.1.8
## [21] tidyverse_1.3.2             pheatmap_1.0.12
## [23] RColorBrewer_1.1-3          ggrepel_0.9.2
## [25] ggplot2_3.4.0               reshape_0.8.9
## [27] biomaRt_2.52.0              DESeq2_1.36.0
## [29] SummarizedExperiment_1.26.1 Biobase_2.56.0
## [31] MatrixGenerics_1.8.1        matrixStats_0.63.0
## [33] GenomicRanges_1.48.0        GenomeInfoDb_1.32.4
## [35] IRanges_2.30.1              S4Vectors_0.34.0
## [37] BiocGenerics_0.42.0         readr_2.1.3
## [39] 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              htmlwidgets_1.6.1
##  [52] httr_1.4.4             ellipsis_0.3.2         R.methodsS3_1.8.2
##  [55] pkgconfig_2.0.3        XML_3.99-0.13          farver_2.1.1
##  [58] deldir_1.0-6           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     stringi_1.7.12         highr_0.10
##  [91] GenomicFeatures_1.48.4 lattice_0.20-45        Matrix_1.5-3
##  [94] vctrs_0.5.1            pillar_1.8.1           lifecycle_1.0.3
##  [97] bitops_1.0-7           rtracklayer_1.56.1     BiocIO_1.6.0
## [100] latticeExtra_0.6-30    hwriter_1.3.2.1        R6_2.5.1
## [103] codetools_0.2-18       MASS_7.3-58.1          assertthat_0.2.1
## [106] rjson_0.2.21           withr_2.5.0            GenomeInfoDbData_1.2.8
## [109] parallel_4.2.1         hms_1.1.2              grid_4.2.1
## [112] coda_0.19-4            rmarkdown_2.20         googledrive_2.0.0
## [115] bbmle_1.0.25           numDeriv_2016.8-1.1    lubridate_1.9.0
## [118] restfulr_0.0.15        interp_1.1-3
```