RNA-seq with LIMMA-VOOM

Asad

2023-05-15

Call libraries

```
library(limma)
library(edgeR)
library(Glimma)
library(AnnotationDbi)
```

```
## Loading required package: stats4
```

```
## Loading required package: BiocGenerics
```

```
##
## Attaching package: 'BiocGenerics'
```

```
## The following object is masked from 'package:limma':
##
##     plotMA
```

```
## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs
```

```
## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, basename, cbind, colnames,
##     dirname, do.call, duplicated, eval, evalq, Filter, Find, get, grep,
##     grepl, intersect, is.unsorted, lapply, Map, mapply, match, mget,
##     order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
##     rbind, Reduce, rownames, sapply, setdiff, sort, table, tapply,
##     union, unique, unsplit, which.max, which.min
```

```
## Loading required package: Biobase
```

```
## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.
## Loading required package: IRanges
```

```
## Loading required package: S4Vectors
```

```
##
## Attaching package: 'S4Vectors'
```

```
## The following object is masked from 'package:utils':
##
##     findMatches
```

```
## The following objects are masked from 'package:base':
##
##     expand.grid, I, unname
```

```
##
## Attaching package: 'IRanges'
```

```
## The following object is masked from 'package:grDevices':
##
##     windows
```

```r
library(org.Hs.eg.db)
```

```
##
```

```r
library(ggplot2)
library(Homo.sapiens)
```

```
## Loading required package: OrganismDbi
```

```
## Loading required package: GenomicFeatures
```

```
## Loading required package: GenomeInfoDb
```

```
## Loading required package: GenomicRanges
```

```
## Loading required package: GO.db
```

```
##
```

```
## Loading required package: TxDb.Hsapiens.UCSC.hg19.knownGene
```

```r
library(RColorBrewer)
library(EnhancedVolcano)
```

```
## Loading required package: ggrepel
## Registered S3 methods overwritten by ’ggalt’:
##   method                  from
##   grid.draw.absoluteGrob  ggplot2
##   grobHeight.absoluteGrob ggplot2
##   grobWidth.absoluteGrob  ggplot2
##   grobX.absoluteGrob      ggplot2
##   grobY.absoluteGrob      ggplot2
```

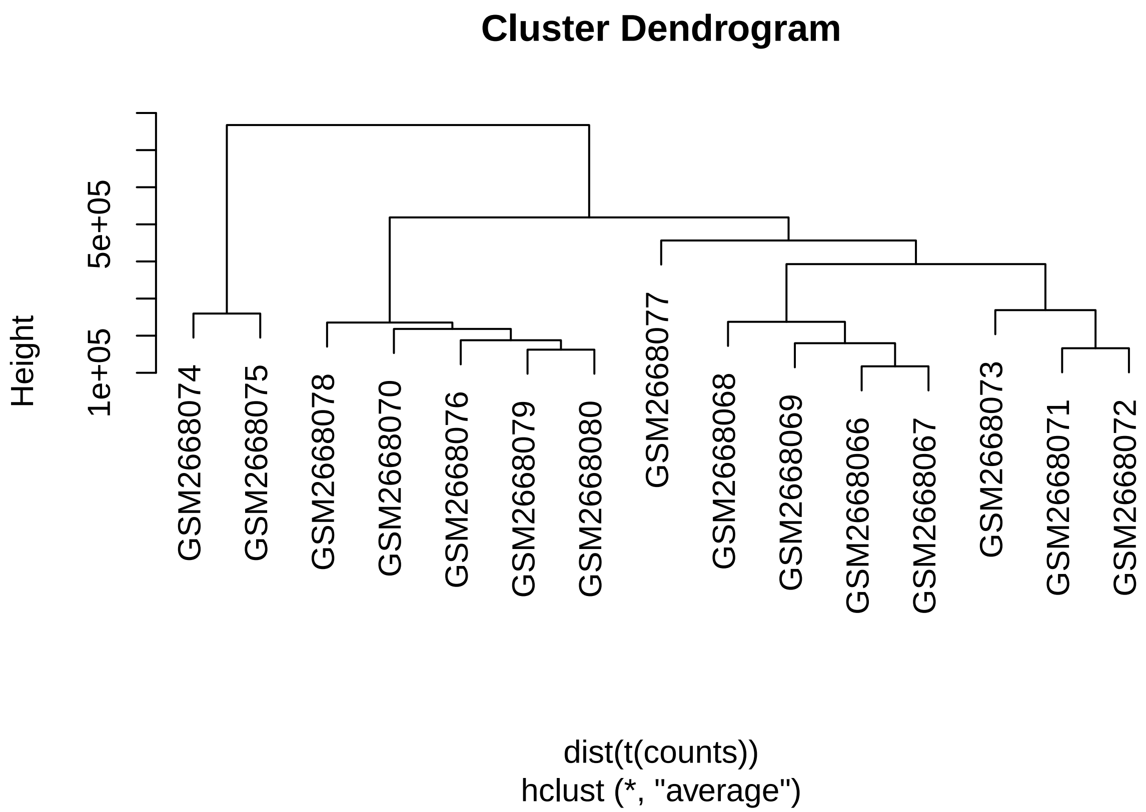
```r
library(pheatmap)
library(dplyr)
```

```
##
## Attaching package: ’dplyr’

## The following object is masked from ’package:OrganismDbi’:
##
##     select

## The following objects are masked from ’package:GenomicRanges’:
##
##     intersect, setdiff, union

## The following object is masked from ’package:GenomeInfoDb’:
##
##     intersect

## The following object is masked from ’package:AnnotationDbi’:
##
##     select

## The following objects are masked from ’package:IRanges’:
##
##     collapse, desc, intersect, setdiff, slice, union

## The following objects are masked from ’package:S4Vectors’:
##
##     first, intersect, rename, setdiff, setequal, union

## The following object is masked from ’package:Biobase’:
##
##     combine

## The following objects are masked from ’package:BiocGenerics’:
##
##     combine, intersect, setdiff, union

## The following objects are masked from ’package:stats’:
##
##     filter, lag
## The following objects are masked from ’package:base’:
##
##     intersect, setdiff, setequal, union
```

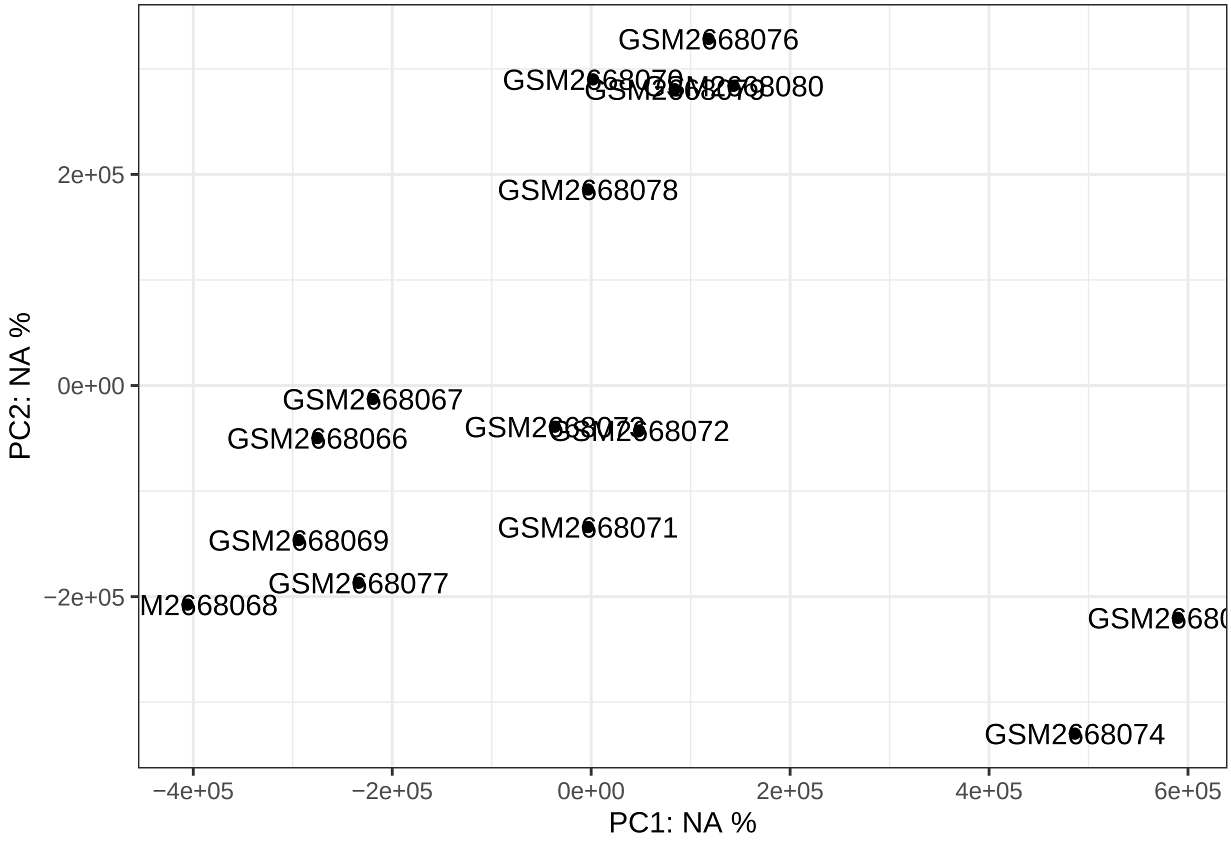
```
setwd('E:/Differential Gene Expression/LIMMA Voom')
counts<- read.csv('Raw read count.csv', row.names = 1)
```

Initial assessment

```
#Clustering
htree<- hclust(dist(t(counts)), method = 'average')
plot(htree)
```



```
#PCA Plot
pca <- prcomp(t(counts))
pca.dat <- pca$x
pca.var <- pca$sdev^2
pca.var.percent <- round(pca.var/sum(pca.var)*100, digits = 2)
pca.dat <- as.data.frame(pca.dat)
ggplot(pca.dat, aes(PC1, PC2)) +
  geom_point() +
  geom_text(label = rownames(pca.dat)) +
  labs(x = paste0('PC1: ', pca.var.percent[1], ' %'),
       y = paste0('PC2: ', pca.var.percent[2], ' %')) + theme_bw()
```



```
#Remove samples based on requirement
```

Prepare DGElist and assign groups

```
DGE<- DGEList(counts)
group<- as.factor(rep(c('Healthy', 'COVID-19'), c(5,10)))
severity<-as.factor(rep(c('HT', 'CP', 'BC'), c(5,5,5)))
DGE$samples$group<-group
DGE$samples$severity<-severity
```

Removing low expressed genes

```
table(rowSums(DGE$counts==0)==15)
```

```
##
## FALSE  TRUE
## 22987 12426
```

```
# 15 samples in our datasets. We can see aroun 12500 genes habe a count of zero.
# Let's remove those
keep <- filterByExpr(DGE, group=group)
```

```
DGE_filtered<- DGE[keep,, keep.lib.sizes=FALSE]
dim(DGE_filtered) #Around 14000 genes remain after filtering
```

```
## [1] 14295    15
```

Transforming data from the raw scale

```
cpm <- cpm(DGE)
lcpm <- cpm(DGE, log=TRUE)

L <- mean(DGE$samples$lib.size) * 1e-6
M <- median(DGE$samples$lib.size) * 1e-6
c(L, M)
```

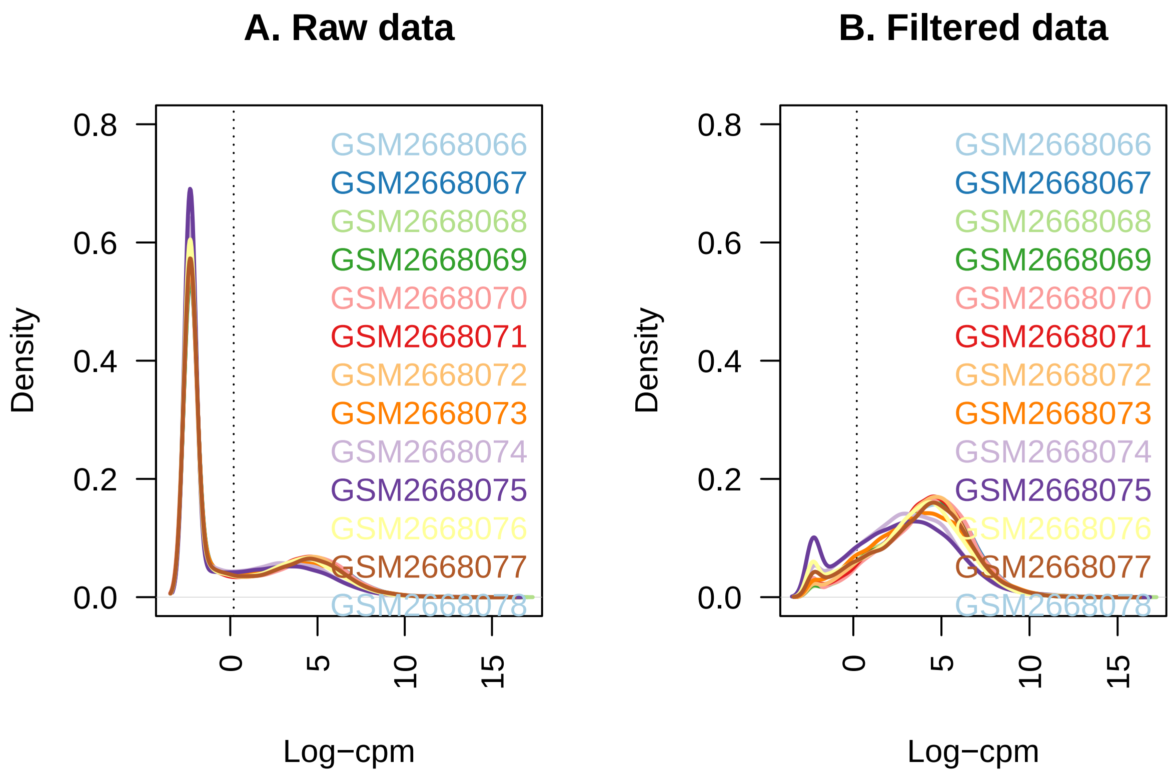
```
## [1]  9.85861 10.60113
```

Preparing density plot

```
par(mfrow=c(1,2))
lcpm.cutoff <- log2(10/M + 2/L)
nsamples <- ncol(DGE)
col <- brewer.pal(nsamples, "Paired")
```

```
## Warning in brewer.pal(nsamples, "Paired"): n too large, allowed maximum for palette Paired is 12
## Returning the palette you asked for with that many colors
```

```
samplenames<- as.character(colnames(DGE))

#Before fitering
lcpm<- lcpm #using lcpm counted before filtering
plot(density(lcpm[,1]), col=col[1], lwd=2,
ylim=c(0,0.8), las=2, main="", xlab="")
title(main="A. Raw data", xlab="Log-cpm")
abline(v=lcpm.cutoff, lty=3)
for (i in 2:nsamples){
  den <- density(lcpm[,i])
  lines(den$x, den$y, col=col[i], lwd=2)
}
legend("topright", samplenames, text.col=col, bty="n")

#After filtering
lcpm2 <- cpm(DGE_filtered, log=TRUE) #calculating new lcpm value from filtered DGE
plot(density(lcpm2[,1]), col=col[1], lwd=2,
ylim=c(0,0.8), las=2, main="", xlab="")
title(main="B. Filtered data", xlab="Log-cpm")
abline(v=lcpm.cutoff, lty=3)
for (i in 2:nsamples){
  den <- density(lcpm2[,i])
  lines(den$x, den$y, col=col[i], lwd=2)
}
legend("topright", samplenames, text.col=col, bty="n")
```

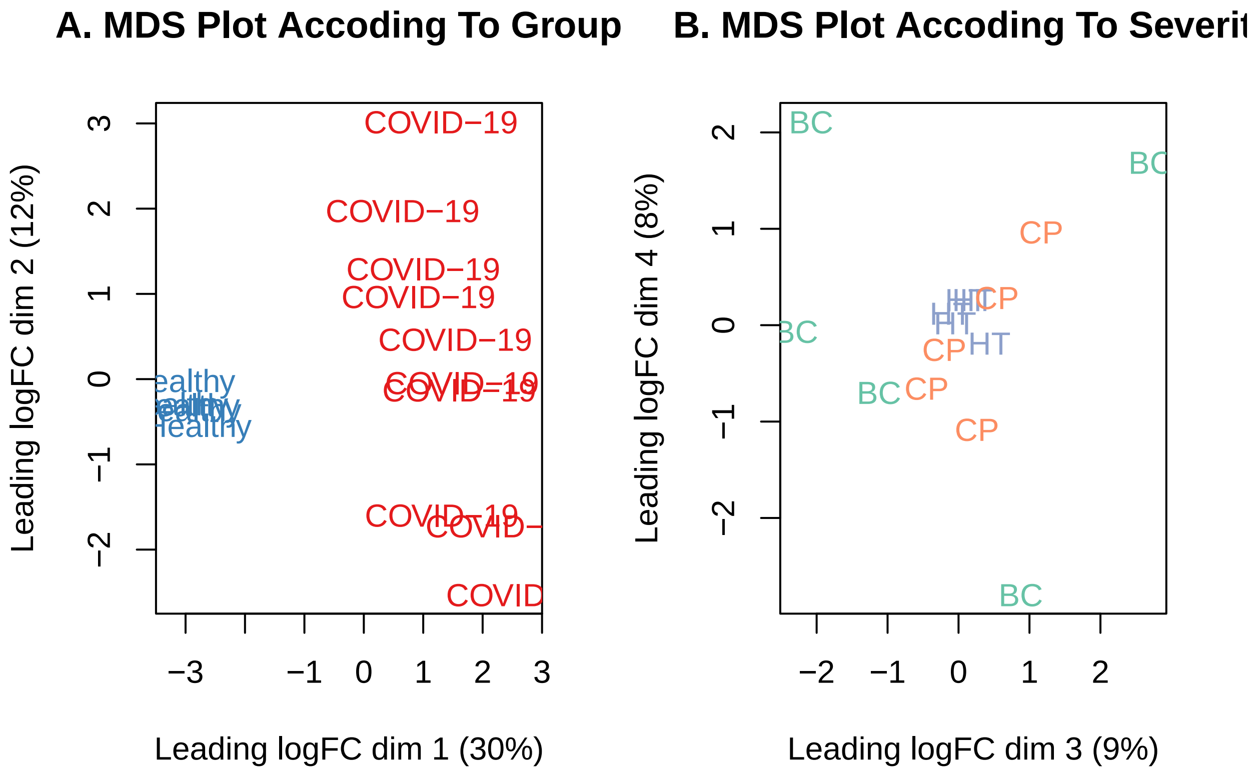


Clustering of samples

```
par(mfrow=c(1,2))
#Plotting according to group
col.group <- group
levels(col.group) <-  brewer.pal(nlevels(col.group), "Set1")
```

```
## Warning in brewer.pal(nlevels(col.group), "Set1"): minimal value for n is 3, returning requested pal
```

```
col.group <- as.character(col.group)
plotMDS(lcpm, labels=group, col=col.group)
title(main="A. MDS Plot Accoding To Groups")

#Plotting according to severity
col.severity <- severity
levels(col.severity) <-  brewer.pal(nlevels(col.severity), "Set2")
col.severity <- as.character(col.severity)
plotMDS(lcpm, labels=severity, col=col.severity, dim=c(3,4))
title(main="B. MDS Plot Accoding To Severity")
```



A. MDS Plot Accoding To Group

B. MDS Plot Accoding To Severit

```
#Online
glMDSPlot(lcpm, labels=paste(group, severity, sep="_"),
          groups=DGE$samples[,c(1,4)], launch=FALSE)
```

TMM Normalization and design and contrast

```
norm.counts<- calcNormFactors(DGE_filtered, method = 'TMM')
norm.counts$samples$norm.factors
```

```
##  [1] 1.1802427 1.2407390 1.1915010 1.1823618 1.2841158 1.0636689 1.0532649
##  [8] 0.9471520 0.6142830 0.5430925 0.9122115 1.0429470 1.1884506 0.9378531
## [15] 1.0055867
```

```
#save normalized read counts
TMM_Counts<- data.frame(cpm(norm.counts))
write.csv(TMM_Counts, 'normalized_counts.csv')


design <- model.matrix(~0+severity+group) #Change order to swap intercept
colnames(design) <- gsub("severity", "", colnames(design))
design
```

```
##    BC CP HT groupHealthy
## 1   0  0  1            1
## 2   0  0  1            1
## 3   0  0  1            1
## 4   0  0  1            1
## 5   0  0  1            1
## 6   0  1  0            0
## 7   0  1  0            0
## 8   0  1  0            0
## 9   0  1  0            0
## 10  0  1  0            0
## 11  1  0  0            0
## 12  1  0  0            0
## 13  1  0  0            0
## 14  1  0  0            0
## 15  1  0  0            0
## attr(,"assign")
## [1] 1 1 1 2
## attr(,"contrasts")
## attr(,"contrasts")$severity
## [1] "contr.treatment"
##
## attr(,"contrasts")$group
## [1] "contr.treatment"
```

```
contr.matrix <- makeContrasts(
  CPvsHT = CP-HT,
  BCvsHT = BC-HT,
  CPvsBC = CP-BC,
  levels = colnames(design))
contr.matrix
```

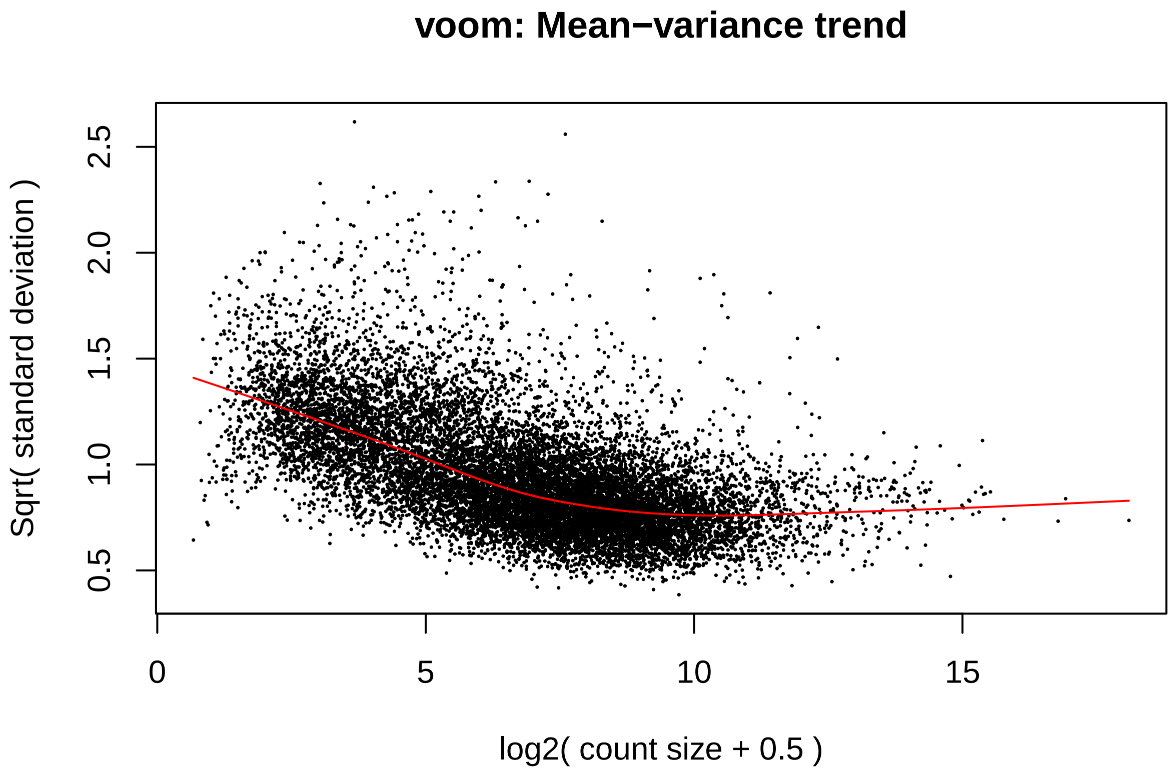
```
##                Contrasts
## Levels          CPvsHT BCvsHT CPvsBC
##   BC                 0      1     -1
##   CP                 1      0      1
##   HT                -1     -1      0
##   groupHealthy       0      0      0
```

Setting voom object and performing DEG analysis

```
voom <- voom(norm.counts, design, plot=TRUE)
```

```
## Coefficients not estimable: groupHealthy
```

```
## Warning: Partial NA coefficients for 14295 probe(s)
```



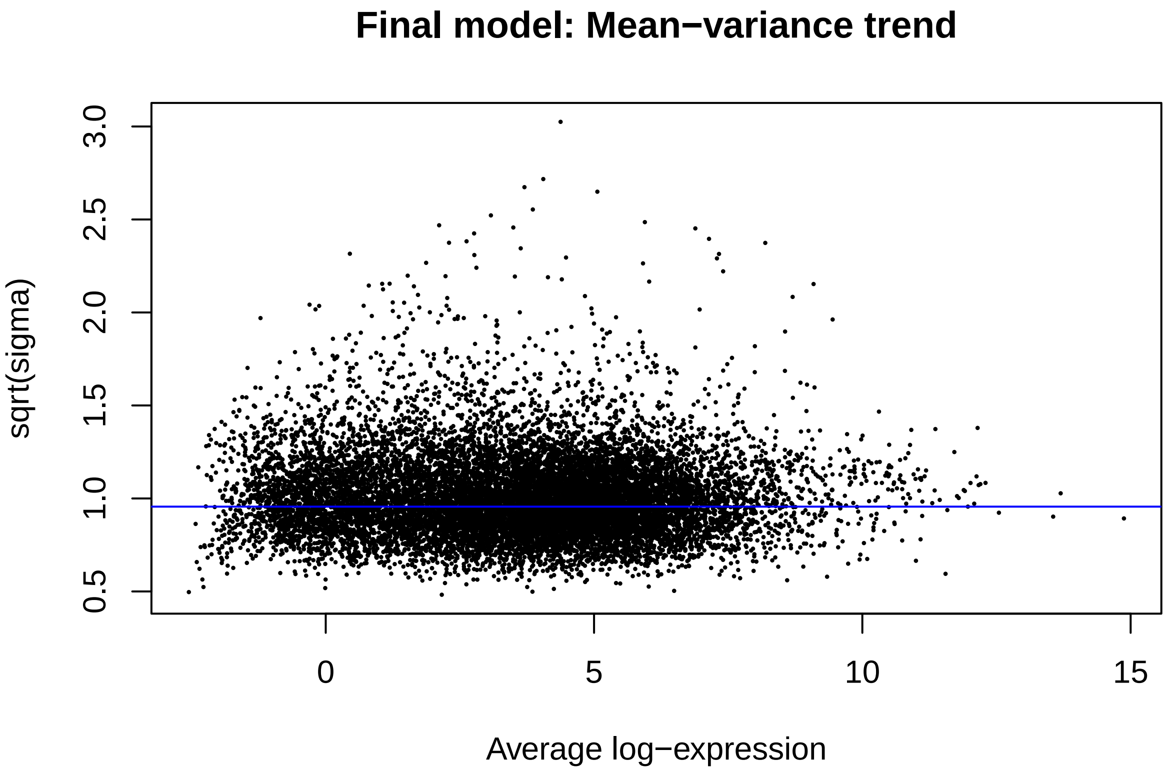
```
vfit <- lmFit(voom, design)
```

```
## Coefficients not estimable: groupHealthy
```

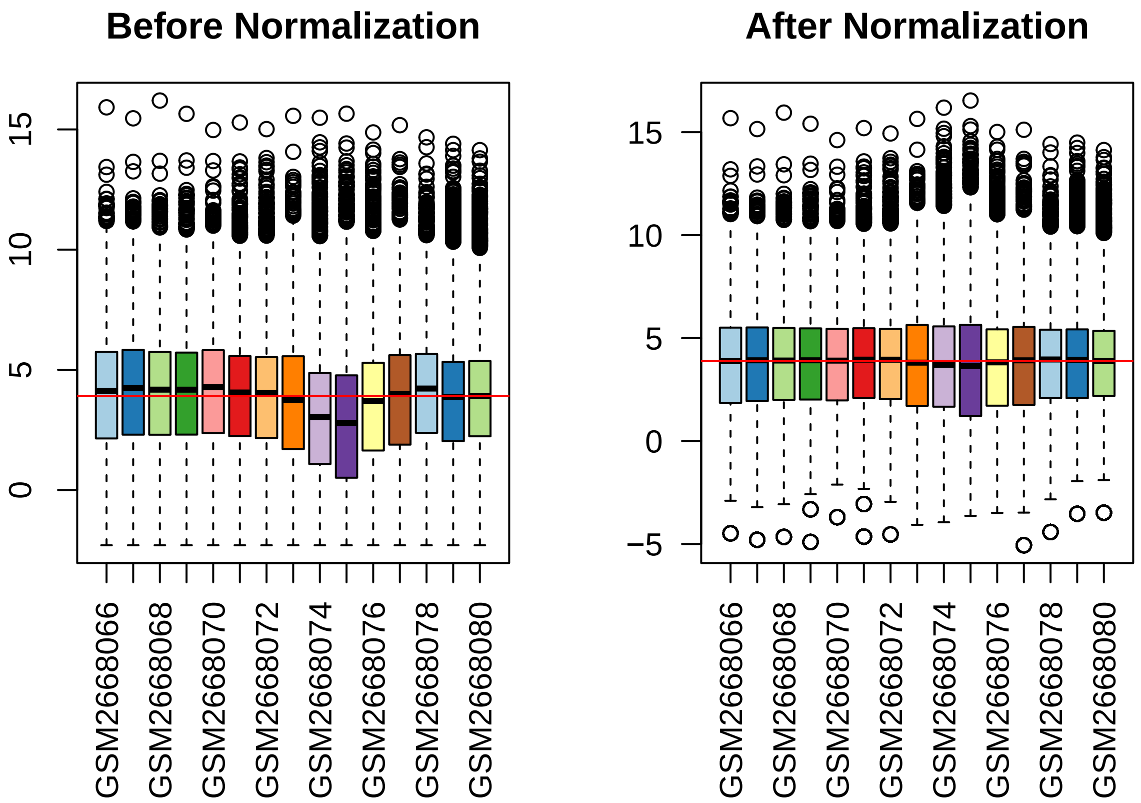
```
## Warning: Partial NA coefficients for 14295 probe(s)
```

```
vfit <- contrasts.fit(vfit, contrasts=contr.matrix)
efit <- eBayes(vfit)
```

```
plotSA(efit, main="Final model: Mean-variance trend")
```



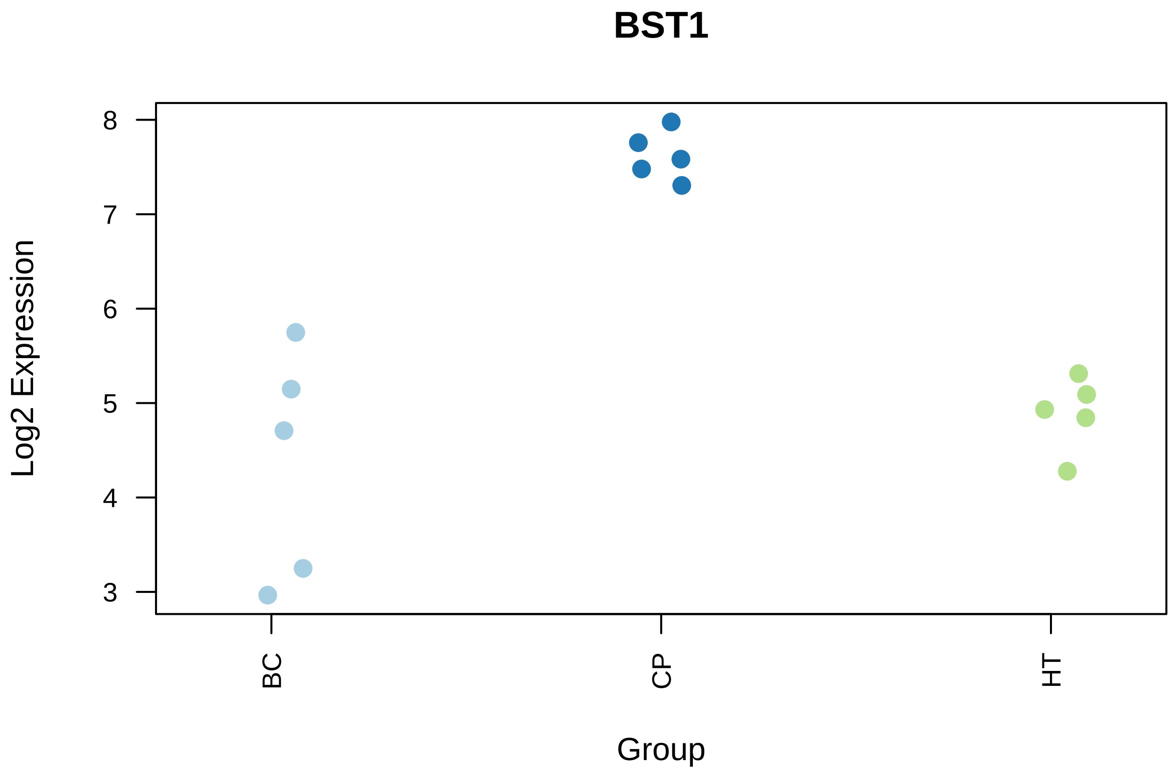
```
par(mfrow=c(1,2))
par(mar=c(7,3,3,2))
boxplot(lcpm2, xlab="", ylab="Log2 counts per million",
        las=3,main="Before Normalization", col=col)
abline(h=median(lcpm2),col="red")

boxplot(voom$E, xlab="", ylab="Log2 counts per million",
        las=2,main="After Normalization", col=col)
abline(h=median(voom$E),col="red")
```



```
graphics.off()
```

Check expression of single gene in all groups

```
stripchart(voom$E["ENSG00000109743",]~severity,vertical=TRUE,las=2,
           cex.axis=0.8,pch=16,cex=1.3,col=col,method="jitter",xlab='Group',
           ylab= 'Log2 Expression',main="BST1")
```

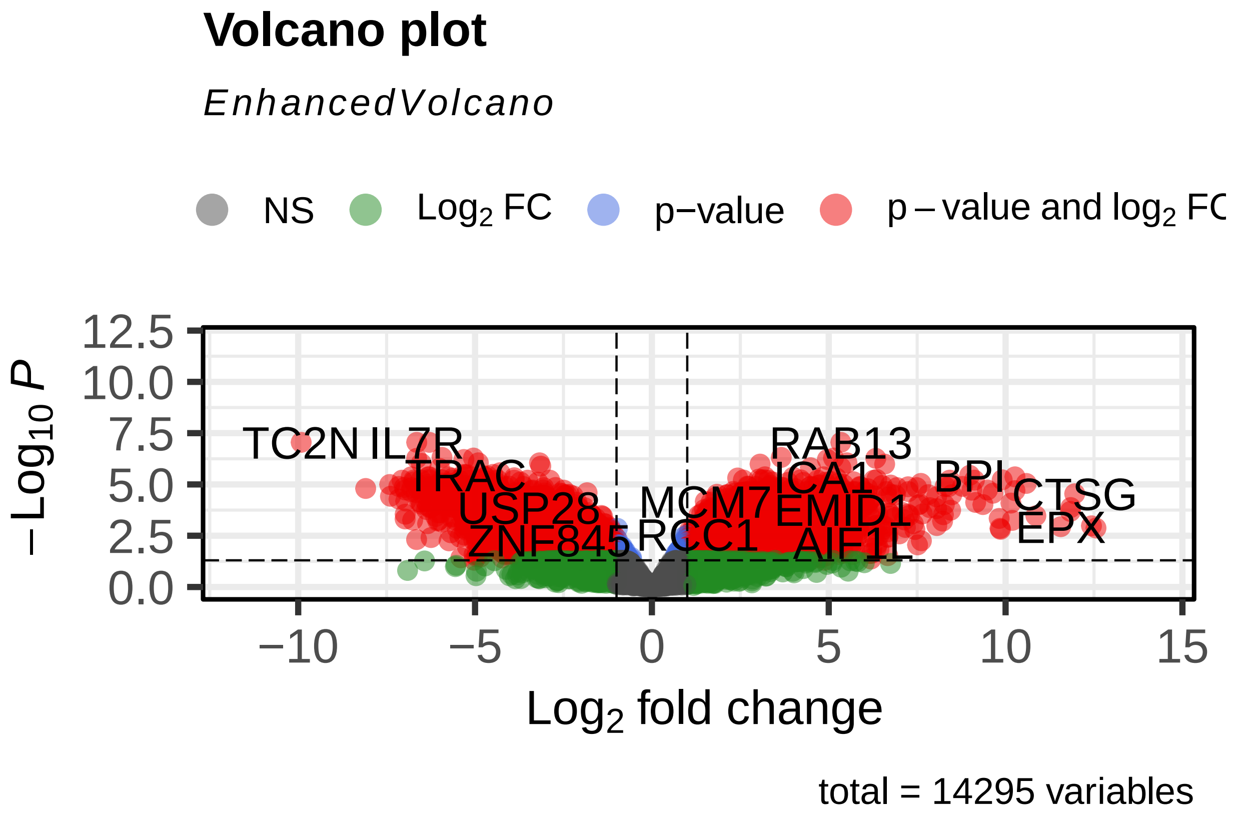


Volcano plot

```
DEGs<- topTreat(efit, n=Inf)
DEGs$symbol<- mapIds(org.Hs.eg.db, keys=rownames(DEGs),
                       keytype = "ENSEMBL", column = "SYMBOL")
```

```
## 'select()' returned 1:many mapping between keys and columns
```

```
par(mar=c(2,2,2,2))
EnhancedVolcano(DEGs,
                lab = DEGs$symbol,
                x =   'logFC',
                y = 'adj.P.Val',
                pCutoff = 0.05,
                FCcutoff = 1,
                pointSize = 3.0,
                labSize = 6.0,
                border = 'full')
```



##
## Analyze

```
sum.fit<- decideTests(efit, lfc = 1)
summary(sum.fit)
```

```
##         CPvsHT BCvsHT CPvsBC
## Down      2564   2345    727
## NotSig    9357   9801  12329
## Up        2374   2149   1239
```

Check individual group

```
CPvsHT <- topTable(efit, coef=1, n=Inf, lfc = 1, p.value = 0.05)
BCvsHT <- topTable(efit, coef=2, n=Inf, lfc = 1, p.value = 0.05)
CPvsBC <- topTable(efit, coef=3, n=Inf, lfc = 1, p.value = 0.05)
#Use topTreat in case topTable doesn't work
```

Annotation

```
CPvsHT$symbol<- mapIds(org.Hs.eg.db, keys=rownames(CPvsHT),
                keytype = "ENSEMBL", column = "SYMBOL")
```

```
## ’select()’ returned 1:many mapping between keys and columns
```

```
BCvsHT$symbol<- mapIds(org.Hs.eg.db, keys=rownames(BCvsHT),
                       keytype = "ENSEMBL", column = "SYMBOL")
```

```
## ’select()’ returned 1:many mapping between keys and columns
```

```
CPvsBC$symbol<- mapIds(org.Hs.eg.db, keys=rownames(CPvsBC),
                       keytype = "ENSEMBL", column = "SYMBOL")
```

```
## ’select()’ returned 1:many mapping between keys and columns
```