

Report Day 1: Analysis of RNA-seq data

Anel Lopez Gonzalez

Contents

Comparing the results from Brauns et al	1
GSE214282	8
Code	9

Comparing the results from Brauns et al

In the recent study titled “Functional reprogramming of monocytes in patients with acute and convalescent severe COVID-19” published by the team of Brauns et al (2022) (Ref. 1), they tested the TLR-induced reaction of blood cells from coming from patients presenting either mild to acute response and in their recovering phases.

A Principal Component Analysis (PCA) showed that the recovering patients from COVID clustered according to their stage, early stages separate from the healthy control (Fig. 1a). Late recovery stage patients clustered together with the healthy controls. Acute infections samples formed their own sample (Fig. 1a). This pattern was recreated after analyzing the raw counts with our own pipelines (Fig. 1b and c) with both the NOISeq and the DESeq2 package. Please note, how the variance captured by the PC1 increases from 27% to 42% after the DESeq2 normalization.

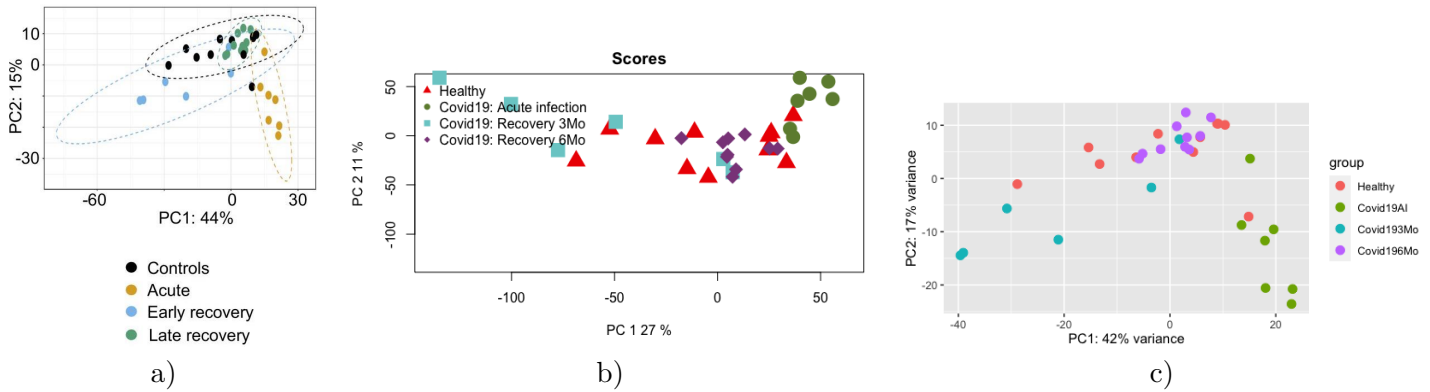


Figure 1: a) PCA of RNA seq samples taken from paper published by Brauns et al, b) PCA made with NOISeq, and c) PCA made with DESeq2.

The research team used DESeq2 to perform the differential expression analysis. Nevertheless, different parameters were used when filtering out the lowly expressed genes. In the article, genes with counts lowered or equal to 20 in at least one sample were removed, meanwhile within our pipeline we filtered out genes which do not have at least 10 counts in at least 6 samples (size of each group). Afterwards, we called DESeq2 and performed DGEA for three comparisons: acute infection (AI) against healthy (H), and the two stages of recovery against healthy three and six months, 3Mo and 6Mo, respectively. We kept the genes that passed the thresholds mentioned in the article, $FDR < 0.05$ and $|FC| > 2$.

Download the genes information for NOISeq analysis:

```
#get biomaRt GC content, gene start, gene end, chr
biomaRt <- read.csv(file = "mart_export.txt", header = 1)
#filter out non canonical chr, keep only the first annotation of each gene
biomaRt_filt <- biomaRt[biomaRt$Chromosome.scaffold.name %in% c(as.character(1:22) ,"X","Y"),]
biomaRt_filt <- biomaRt_filt[!duplicated(biomaRt_filt$NCBI.gene..formerly.Entrezgene..ID),]
row.names(biomaRt_filt) <- biomaRt_filt$NCBI.gene..formerly.Entrezgene..ID

#length
len <- biomaRt_filt$Gene.end..bp. - biomaRt_filt$Gene.start..bp.
names(len) <- biomaRt_filt$NCBI.gene..formerly.Entrezgene..ID

#GC%
gc <- biomaRt_filt$Gene...GC.content
names(gc) <- biomaRt_filt$NCBI.gene..formerly.Entrezgene..ID

#biotype
type <- biomaRt_filt$Gene.type
names(type) <- biomaRt_filt$NCBI.gene..formerly.Entrezgene..ID

#chr
chr <- data.frame(Chr=biomaRt_filt$NCBI.gene..formerly.Entrezgene..ID,
                  GeneStart=biomaRt_filt$Gene.start..bp.,
                  GeneEnd=biomaRt_filt$Gene.end..bp.,
                  row.names = biomaRt_filt$NCBI.gene..formerly.Entrezgene..ID)

#filter from tbl the not annotated genes
tbl_filt <- tbl[rownames(tbl) %in% rownames(biomaRt_filt),]
tbl_count_exc <- tbl[!(rownames(tbl) %in% rownames(biomaRt_filt)),]
annotgene_ord <- biomaRt_filt[rownames(tbl_filt) ,]
#sum(rownames(annotgene_ord)==rownames(tbl_filt))

#df of exp design
myfactors <- data.frame(metadata [ colnames(tbl_filt),c("disease state:ch1")])
colnames(myfactors)[1]<- "Group"

#make data inot NOISeq object
NOISeq_obj <- readData(data = tbl, length = len, gc = gc,
                      chromosome = chr, factors = myfactors, biotype = type)

#PCA NOISeq
myPCA = dat(NOISeq_obj, type = "PCA")
par(mfrow = c(1, 1))
explo.plot(myPCA, factor = "Group")
```

Let's call DESeq2 and perform the DGEA:

```
#prepare matadata for deseq2
coldata <- pData(NOISeq_obj)
coldata[coldata=="Covid19: Acute infection"] <- "Covid19AI"
coldata[coldata=="Covid19: Recovery 3Mo"] <- "Covid193Mo"
```

```

coldata[coldata=="Covid19: Recovery 6Mo"] <- "Covid196Mo"
coldata$Group <- factor(coldata$Group, levels = unique(coldata$Group))

#create deseq2 obj
dds <- DESeqDataSetFromMatrix(countData = tbl_filt,
                              colData = coldata,
                              design = ~ Group)

#filter low counts genes
smallestGroupSize <- 6
keep <- rowSums(counts(dds) >= 10) >= smallestGroupSize
dss <- dds[keep,]

#Call DESeq2
dds <- DESeq(dds)

#PCA
vsd <- vst(dds, blind=FALSE)
plotPCA(vsd, intgroup=c("Group"), returnData=FALSE)

#retrieve results of desired comparisons
DEG_Covid19AI <- results(dds, name = "Group_Covid19AI_vs_Healthy")
DEG_Covid193Mo <- results(dds, name = "Group_Covid193Mo_vs_Healthy")
DEG_Covid196Mo <- results(dds, name = "Group_Covid196Mo_vs_Healthy")

#filter DEGs: |FC| > 2 and FDR < 0.05
DEG_Covid19AI_DF <- filter(as.data.frame(DEG_Covid19AI), padj <= 0.05 & abs(log2FoldChange)>2)
DEG_Covid193Mo_DF <- filter(as.data.frame(DEG_Covid193Mo), padj <= 0.05 & abs(log2FoldChange)>2)
DEG_Covid196Mo_DF <- filter(as.data.frame(DEG_Covid196Mo), padj <= 0.05 & abs(log2FoldChange)>2)

```

Get information on the DEGs:

```

#get annotations on DEGs
DEG_Covid19AI_DF$symbol <- mapIds(org.Hs.eg.db, keys=rownames(DEG_Covid19AI_DF),
                                  column="SYMBOL", keytype="ENTREZID", multiVals="first")
DEG_Covid19AI_DF$entrez <- mapIds(org.Hs.eg.db, keys=rownames(DEG_Covid19AI_DF),
                                  column="ENTREZID", keytype="ENTREZID", multiVals="first")
DEG_Covid19AI_DF$name <- mapIds(org.Hs.eg.db, keys=rownames(DEG_Covid19AI_DF),
                                column="GENENAME", keytype="ENTREZID", multiVals="first")
DEG_Covid19AI_DF$refseq <- mapIds(org.Hs.eg.db, keys=rownames(DEG_Covid19AI_DF),
                                  column="REFSEQ", keytype="ENTREZID", multiVals="first")
DEG_Covid19AI_DF$ensembl <- mapIds(org.Hs.eg.db, keys=rownames(DEG_Covid19AI_DF),
                                   column="ENSEMBL", keytype="ENTREZID", multiVals="first")

#get annotations on DEGs
DEG_Covid193Mo_DF$symbol <- mapIds(org.Hs.eg.db, keys=rownames(DEG_Covid193Mo_DF),
                                  column="SYMBOL", keytype="ENTREZID", multiVals="first")
DEG_Covid193Mo_DF$entrez <- mapIds(org.Hs.eg.db, keys=rownames(DEG_Covid193Mo_DF),
                                  column="ENTREZID", keytype="ENTREZID", multiVals="first")
DEG_Covid193Mo_DF$name <- mapIds(org.Hs.eg.db, keys=rownames(DEG_Covid193Mo_DF),
                                column="GENENAME", keytype="ENTREZID", multiVals="first")
DEG_Covid193Mo_DF$refseq <- mapIds(org.Hs.eg.db, keys=rownames(DEG_Covid193Mo_DF),

```

```

                                column="REFSEQ", keytype="ENTREZID", multiVals="first")
DEG_Covid193Mo_DF$ensembl <- mapIds(org.Hs.eg.db, keys=rownames(DEG_Covid193Mo_DF),
                                column="ENSEMBL", keytype="ENTREZID", multiVals="first")

```

Heatmap of DEGs across groups:

```

#make matrix of FC for all groups
DEGs_FC <- matrix(0, nrow = 510, ncol = 3)
rownames(DEGs_FC) <- rownames(DEG_Covid19AI_DF)
colnames(DEGs_FC) <- c("AI", "3Mo", "6Mo")
idx1 <- rownames(DEG_Covid19AI) %in% rownames(DEGs_FC)
DEGs_FC[,1] <- as.data.frame(DEG_Covid19AI)[idx1,2]

idx2 <- rownames(DEG_Covid193Mo) %in% rownames(DEGs_FC)
DEGs_FC[,2] <- as.data.frame(DEG_Covid193Mo)[idx2,2]

idx3 <- rownames(DEG_Covid196Mo) %in% rownames(DEGs_FC)
DEGs_FC[,3] <- as.data.frame(DEG_Covid196Mo)[idx3,2]

DEGs_FC1 <- DEGs_FC[!(row.names(DEGs_FC) == "1832"),]
pheatmap(DEGs_FC1,
          cluster_rows = TRUE,
          cluster_cols = TRUE,
          show_rownames = FALSE,
          show_colnames = TRUE)

col <- colorRampPalette(brewer.pal(10, "RdYlBu"))(256)
norm_counts <- counts(dds, normalized=TRUE)
norm_counts <- norm_counts[(row.names(norm_counts) %in% rownames(DEGs_FC)),]
colnames(norm_counts) <- coldata$Group
heatmap <- heatmap(norm_counts, col=col)

```

Perform GO analysis:

```

#GO cmd
egoAIH <- enrichGO(gene      = rownames(DEG_Covid19AI_DF),
                   OrgDb     = org.Hs.eg.db,
                   ont        = "ALL",
                   pAdjustMethod = "BH",
                   pvalueCutoff = 0.05,
                   qvalueCutoff = 0.1,
                   readable    = TRUE)

head(egoAIH)

#dotplot
dotplot(egoAIH, showCategory = 20,
        font.size = 10,
        title = "AI vs H")

```

GSEA analysis for acute and early recovery:


```

#sort list of genes according to LF value
DEG_Covid19AI_sg <- DEG_Covid19AI_DF$log2FoldChange
names(DEG_Covid19AI_sg) <- DEG_Covid19AI_DF$symbol
DEG_Covid19AI_sg <- na.omit(DEG_Covid19AI_sg)
DEG_Covid19AI_sg <- sort(DEG_Covid19AI_sg, decreasing = TRUE)

gse_AI_All <- gseGO(geneList=DEG_Covid19AI_sg,
  ont = "ALL",
  keyType = "SYMBOL",
  minGSSize = 3,
  maxGSSize = 800,
  pvalueCutoff = 0.05,
  verbose = TRUE,
  OrgDb = org.Hs.eg.db,
  pAdjustMethod = "none")
gse_AI_BP <- gseGO(geneList=DEG_Covid19AI_sg,
  ont = "BP",
  keyType = "SYMBOL",
  minGSSize = 3,
  maxGSSize = 800,
  pvalueCutoff = 0.05,
  verbose = TRUE,
  OrgDb = org.Hs.eg.db,
  pAdjustMethod = "none")
gse_AI_CC <- gseGO(geneList=DEG_Covid19AI_sg,
  ont = "CC",
  keyType = "SYMBOL",
  minGSSize = 3,
  maxGSSize = 800,
  pvalueCutoff = 0.05,
  verbose = TRUE,
  OrgDb = org.Hs.eg.db,
  pAdjustMethod = "none")

dotplot(gse_AI_All, showCategory=10, split=".sign", font.size = 10) +
  facet_grid(.~.sign) +
  scale_y_discrete(labels=function(egoBP) str_wrap(egoBP, width=70))

dotplot(gse_AI_BP, showCategory=10, split=".sign", font.size = 10) +
  facet_grid(.~.sign) +
  scale_y_discrete(labels=function(egoBP) str_wrap(egoBP, width=70))

dotplot(gse_AI_CC, showCategory=10, split=".sign", font.size = 10) +
  facet_grid(.~.sign) +
  scale_y_discrete(labels=function(egoBP) str_wrap(egoBP, width=70))

```

```

#sort list of genes according to LF value
DEG_Covid193Mo_sg <- DEG_Covid193Mo_DF$log2FoldChange
names(DEG_Covid193Mo_sg) <- DEG_Covid193Mo_DF$symbol
DEG_Covid193Mo_sg <- na.omit(DEG_Covid193Mo_sg)
DEG_Covid193Mo_sg <- sort(DEG_Covid193Mo_sg, decreasing = TRUE)

```

```

gse_3Mo_All <- gseG0(geneList=DEG_Covid193Mo_sg,
  ont = "ALL",
  keyType = "SYMBOL",
  minGSSize = 3,
  maxGSSize = 800,
  pvalueCutoff = 0.05,
  verbose = TRUE,
  OrgDb = org.Hs.eg.db,
  pAdjustMethod = "none")
gse_3Mo_BP <- gseG0(geneList=DEG_Covid193Mo_sg,
  ont = "BP",
  keyType = "SYMBOL",
  minGSSize = 3,
  maxGSSize = 800,
  pvalueCutoff = 0.05,
  verbose = TRUE,
  OrgDb = org.Hs.eg.db,
  pAdjustMethod = "none")
gse_3Mo_CC <- gseG0(geneList=DEG_Covid193Mo_sg,
  ont = "CC",
  keyType = "SYMBOL",
  minGSSize = 3,
  maxGSSize = 800,
  pvalueCutoff = 0.05,
  verbose = TRUE,
  OrgDb = org.Hs.eg.db,
  pAdjustMethod = "none")

dotplot(gse_3Mo_All, showCategory=10, split=".sign", font.size = 10) +
  facet_grid(~.sign) +
  scale_y_discrete(labels=function(egoBP) str_wrap(egoBP, width=70))

dotplot(gse_3Mo_BP, showCategory=10, split=".sign", font.size = 10) +
  facet_grid(~.sign) +
  scale_y_discrete(labels=function(egoBP) str_wrap(egoBP, width=70))

dotplot(gse_3Mo_CC, showCategory=10, split=".sign", font.size = 10) +
  facet_grid(~.sign) +
  scale_y_discrete(labels=function(egoBP) str_wrap(egoBP, width=70))

```

Reference:

1. Brauns, E., Azouz, A., Grimaldi, D., Xiao, H., Thomas, S., Nguyen, M., ... & Goriely, S. (2022). Functional reprogramming of monocytes in patients with acute and convalescent severe COVID-19. JCI insight, 7(9). JCI Insight. 2022;7(9):e154183. <https://doi.org/10.1172/jci.insight.154183>.

GSE214282

Transcriptome profiling of classical monocytes in ME/CFS post exercise challenge

Monocytes were collected 24 hrs after the participants performed a cardiopulmonary exercise test (CPET). Patients belonging to two different groups were included in this study: 4 myalgic encephalomyelitis, also called chronic fatigue syndrome or ME/CFS, and 4 healthy controls. The study has not been published yet. The raw counts were downloaded from the GEO database. A PCA showed poor separation between the two groups of this study (Fig. 1).

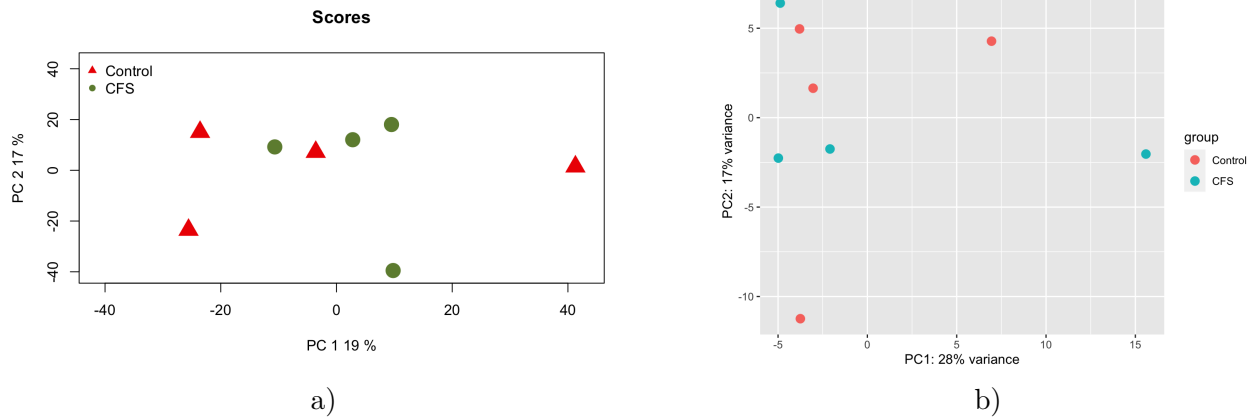


Figure 4: a) PCA of RNA seq samples done with NOISeq and b) PCA made with DESeq2.

As expected not many DEGs were found: 16 genes which became 11 after LF shrinkage. This set of genes were found with not conservative parameters: $\text{padj} < 0.01$ and $|\text{FC}| > 1$. When performed GO analysis no relevant pathways came to light. For GSEA 3 pathways were found to be significantly down-regulated.

When we plot the FC of the DEGs as a heatmap, we can see genes do not show a clear different pattern of expression between the conditions.

The response this dataset shows is very noisy. This could be because the ME/CFS does not have a huge impact on the biology of these cells. Another explanation could be that effort the CPET is not a good proxy to examine how this syndrome affects the white blood cells.

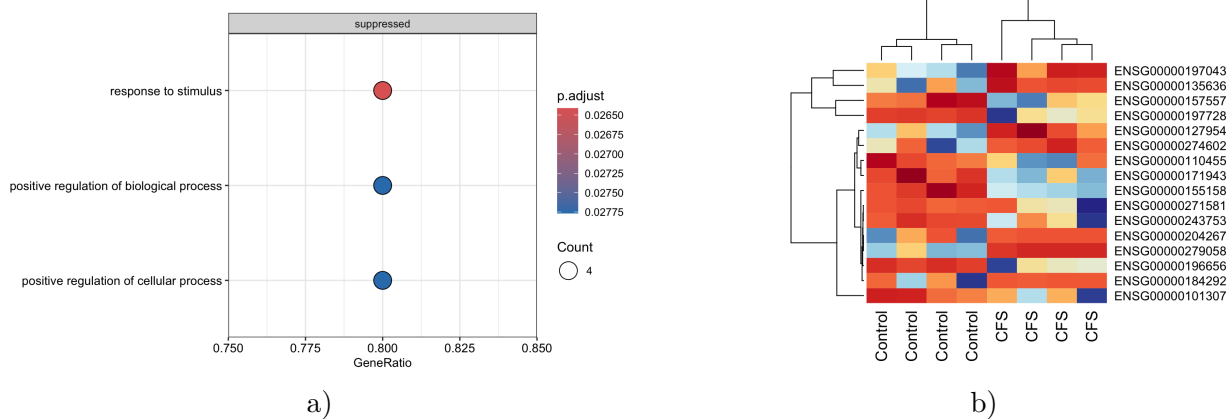


Figure 5: a) GSEA and b) Heatmap done only with the DEGs.

Code

Load the needed libraries:

```
library(GEOquery)
library(NOISeq)
library(DESeq2)
library(clusterProfiler)
library(org.Hs.eg.db)
library(RColorBrewer)
library(gplots)
```

Download the counts and metadata:

```
#read countys in
counts <- read.table(file = "GSE214282_rawCounts.txt", sep = "\t", header = TRUE)
head(counts)

#geo
gset <- GEOquery::getGEO("GSE214282")[[1]]

#Get the sample metadata
metadata <- gset@phenoData@data
group <- metadata[,c("characteristics_ch1.1", "geo_accession")]
group[group=="disease state: control",1] <- "Control"
group[group=="disease state: case",1] <- "CFS"
colnames(group) <- c("Group", "SampleID")

#wirte gene names for biomart
#write.csv(counts$Id, file = "GSE214282_genes.txt", quote = FALSE, row.names = FALSE)

rownames(counts) <- counts$Id
counts <- counts[,2:9]
```

Download the genes information for NOISeq analysis:

```
#get biomart GC content, gene start, gene end, chr
biomart2 <- read.delim(file = "mart_export2.txt", header = 1, sep = "\t")
#filter out non canonical chr, keep only the first annotation of each gene
biomart_filt2 <- biomart2[biomart2$Chromosome.scaffold.name %in% c(as.character(1:22), "X", "Y"),]
biomart_filt2 <- biomart_filt2[!duplicated(biomart_filt2$NCBI.gene..formerly.Entrezgene..ID),]
biomart_filt2 <- na.omit(biomart_filt2)
rownames(biomart_filt2) <- biomart_filt2$NCBI.gene..formerly.Entrezgene..ID

#length
len <- biomart_filt2$Gene.end..bp. - biomart_filt2$Gene.start..bp.
names(len) <- biomart_filt2$NCBI.gene..formerly.Entrezgene..ID

#GC%
gc <- biomart_filt2$Gene...GC.content
names(gc) <- biomart_filt2$NCBI.gene..formerly.Entrezgene..ID

#biotype
type <- biomart_filt2$Gene.type
names(type) <- biomart_filt2$NCBI.gene..formerly.Entrezgene..ID
```

```

#chr
chr <- data.frame(Chr=biomart_filt2$NCBI.gene..formerly.Entrezgene..ID,
                  GeneStart=biomart_filt2$Gene.start..bp.,
                  GeneEnd=biomart_filt2$Gene.end..bp.,
                  row.names = biomart_filt2$NCBI.gene..formerly.Entrezgene..ID)

#filter from tbl the not annotated genes
counts_filt <- counts[rownames(counts) %in% rownames(biomart_filt2),]
counts_exc <- counts[!(rownames(counts) %in% rownames(biomart_filt2)),]
annotgene_ord <- biomart_filt2[rownames(counts_filt),]
#sum(rownames(annotgene_ord)==rownames(tbl_filt))

#df of exp design
myfactors <- data.frame(group$Group)
colnames(myfactors)[1]<- "Group"

#make data inot NOISeq object
NOISeq_obj <- readData(data = counts, length = len, gc = gc,
                      chromosome = chr, factors = myfactors, biotype = type)

#PCA NOISeq
myPCA = dat(NOISeq_obj, type = "PCA")
par(mfrow = c(1, 1))
explo.plot(myPCA, factor = "Group")

#counts
mycountsbio = dat(NOISeq_obj, factor = NULL, type = "countsbio")
par(mfrow = c(1, 1))
explo.plot(mycountsbio, toplot = 1, samples = 1, plottype = "boxplot")

```

Let's call DESeq2 and perform the DGEA:

```

#prepare matadata for deseq2
coldata2 <- pData(NOISeq_obj)
coldata2$Group <- factor(coldata2$Group, levels = unique(coldata2$Group))

#create deseq2 obj
dds2 <- DESeqDataSetFromMatrix(countData = counts,
                               colData = coldata2,
                               design = ~ Group)

#filter low counts genes
smallestGroupSize <- 2
keep <- rowSums(counts(dds2) >= 1) >= smallestGroupSize
dss2 <- dds2[keep,]

#Call DESeq2
dds2 <- DESeq(dds2)
resultsNames(dds2)

#PCA

```

```

vsd2 <- vst(dds2, blind=FALSE)
plotPCA(vsd2, intgroup=c("Group"), returnData=FALSE)

#counts dist
boxplot(log10(counts), pch=".",
        horizontal=TRUE, cex.axis=0.5,
        las=1, ylab="Samples", xlab="log2(Counts +1)")

#library(reshape2)
count_melt <- reshape2::melt(log2(counts + 1))
ggplot(data=count_melt, mapping=aes(x=value, color=variable)) + geom_density()

plotCounts(dds2, gene=which.min(CFSvsCTL$padj), intgroup="Group")

sizeFactors(dds2)

#retrieve results of desired comparisons
CFSvsCTL <- results(dds2, name = "Group_CFS_vs_Control")

CFSvsCTL_LFC <- lfcShrink(dds2, coef="Group_CFS_vs_Control", type="apeglm")

plotMA(CFSvsCTL, ylim=c(-2,2))
plotMA(CFSvsCTL_LFC, ylim=c(-2,2))

#filter DEGs: |FC| > 2 and FDR < 0.05
CFSvsCTL_DF <- filter(as.data.frame(CFSvsCTL), padj <= 0.1 & abs(log2FoldChange)>0.5)
CFSvsCTL_LFC_DF <- filter(as.data.frame(CFSvsCTL_LFC), padj <= 0.1 & abs(log2FoldChange)>0.5)

```

Get information on the DEGs:

```

#get annotations on DEGs
CFSvsCTL_DF$symbol <- mapIds(org.Hs.eg.db, keys=rownames(CFSvsCTL_DF),
                           column="SYMBOL", keytype="ENSEMBL", multiVals="first")
CFSvsCTL_DF$entrez <- mapIds(org.Hs.eg.db, keys=rownames(CFSvsCTL_DF),
                           column="ENTREZID", keytype="ENSEMBL", multiVals="first")
CFSvsCTL_DF$name <- mapIds(org.Hs.eg.db, keys=rownames(CFSvsCTL_DF),
                          column="GENENAME", keytype="ENSEMBL", multiVals="first")

```

Heatmap of DEGs across groups:

```

#make matrix of FC for all groups
col <- colorRampPalette(brewer.pal(10, "RdYlBu"))(256)
norm_counts <- counts(dds2, normalized=TRUE)
norm_counts <- norm_counts[(row.names(norm_counts) %in% rownames(CFSvsCTL_DF)),]
colnames(norm_counts) <- coldata2$Group
heatmap <- heatmap(norm_counts, col=col)

```

Perform GO analysis:

```

#GO cmd
egoCFS <- enrichGO(gene      = rownames(CFSvsCTL_DF),
                  OrgDb      = org.Hs.eg.db,
                  ont        = "ALL",
                  pAdjustMethod = "BH",
                  pvalueCutoff = 0.05,
                  qvalueCutoff = 0.1,
                  readable    = TRUE)

head(egoCFS)

#dotplot
dotplot(egoCFS, showCategory = 20,
        font.size = 10,
        title = "AI vs H")

```

GSEA analysis for acute and early recovery:

```

#sort list of genes according to LF value
CFSvsCTL_sg <- CFSvsCTL_DF$log2FoldChange
names(CFSvsCTL_sg) <- rownames(CFSvsCTL_DF)
CFSvsCTL_sg <- na.omit(CFSvsCTL_sg)
CFSvsCTL_sg <- sort(CFSvsCTL_sg, decreasing = TRUE)

gse_CFS_All <- gseGO(geneList=CFSvsCTL_sg,
                    ont = "ALL",
                    keyType = "ENSEMBL",
                    minGSSize = 3,
                    maxGSSize = 800,
                    pvalueCutoff = 0.05,
                    verbose = TRUE,
                    OrgDb = org.Hs.eg.db,
                    pAdjustMethod = "none")

head(gse_CFS_All)

dotplot(gse_CFS_All, showCategory=10, split=".sign", font.size = 10) +
  facet_grid(~.sign) +
  scale_y_discrete(labels=function(egoBP) str_wrap(egoBP, width=70))

```