Differential expression: Chunk vs dissociated

Ariel Hippen

2022-07-25

Contents

DESeq2 pipeline **1**
Load data 2
Differential expression 5
Plotting results 7
Transformations 9
Sample comparisons 12
Conclusions 17

DESeq2 pipeline

This project has two aims that focus on comparing the differential expression of bulk RNA-seq. This one assesses the effect of dissociation by comparing paired tumor samples that were sequenced as chunks vs. dissociated and then bulk sequenced. The second will assess the difference between dissociated cells that have been rRNA-depleted vs those that have been 3'/poly-A captured.

For preliminary assessment of the DE results, we will be following the DESeq2 tutorial: https://bioconductor.org/packages/devel/bioc/vignettes/DESeq2/inst/doc/DESeq2.html (With some references to https://bioconductor.org/packages/devel/workflows/vignettes/rnaseqGene/inst/doc/rnaseqGene.html#exploratory-analysis-and-visualization)

```
suppressPackageStartupMessages({
  library(data.table)
  library(DESeq2)
  library(vsn)
  library(pheatmap)
  library(RColorBrewer)
  library(PCAtools)
  library(testit)
  library(yaml)
})

params <- read_yaml("../../config.yml")
data_path <- params$data_path
local_data_path <- params$local_data_path
samples <- params$samples
```

Load data

```
# Get paths to STAR.counts files for all ribosomal depleted samples
# Note: data_path is a variable loaded from config.R
directory <- paste(data_path, "bulk_tumors", sep = "/")
sampleFiles <- list.files(directory, recursive = TRUE, full.names = TRUE)
sampleFiles <- grep("ReadsPerGene.out.tab", sampleFiles, value = TRUE)
sampleFiles <- grep("ribo", sampleFiles, value = TRUE)
```

DESeq expects a metadata table to pass into the colData part of a SummarizedExperiment object. We'll prep it here.

```
# Pull sample ids out of full path names. Their ids should be one directory in
# the path and the only exclusively numeric one, hence the regex.
sampleNames <- gsub(".*/(\\d+)/.*", "\\1", sampleFiles)
sampleCondition <- ifelse(1:16 %in% grep("chunk", sampleFiles),
                          "chunk", "dissociated")
sampleUnique <- paste(sampleNames, sampleCondition, sep = "_")
colData <- data.frame(id = sampleUnique,
                      sample = sampleNames,
                      fileName = sampleFiles,
                      condition = sampleCondition)
colData$condition <- factor(colData$condition)

colData
```

```
##                  id sample
## 1        2251_chunk   2251
## 2  2251_dissociated   2251
## 3        2267_chunk   2267
## 4  2267_dissociated   2267
## 5        2283_chunk   2283
## 6  2283_dissociated   2283
## 7        2293_chunk   2293
## 8  2293_dissociated   2293
## 9        2380_chunk   2380
## 10 2380_dissociated   2380
## 11       2428_chunk   2428
## 12 2428_dissociated   2428
## 13       2467_chunk   2467
## 14 2467_dissociated   2467
## 15       2497_chunk   2497
## 16 2497_dissociated   2497
##
## 1        /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2251/chunk_ribo/STAR/ReadsPerGe
## 2  /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2251/dissociated_ribo/STAR/ReadsPerGe
## 3        /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2267/chunk_ribo/STAR/ReadsPerGe
## 4  /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2267/dissociated_ribo/STAR/ReadsPerGe
## 5        /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2283/chunk_ribo/STAR/ReadsPerGe
## 6  /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2283/dissociated_ribo/STAR/ReadsPerGe
## 7        /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2293/chunk_ribo/STAR/ReadsPerGe
## 8  /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2293/dissociated_ribo/STAR/ReadsPerGe
## 9        /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2380/chunk_ribo/STAR/ReadsPerGe
## 10 /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2380/dissociated_ribo/STAR/ReadsPerGe
## 11       /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2428/chunk_ribo/STAR/ReadsPerGe
## 12 /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2428/dissociated_ribo/STAR/ReadsPerGe
## 13       /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2467/chunk_ribo/STAR/ReadsPerGe
## 14 /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2467/dissociated_ribo/STAR/ReadsPerGe
## 15       /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2497/chunk_ribo/STAR/ReadsPerGe
## 16 /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2497/dissociated_ribo/STAR/ReadsPerGe
##      condition
## 1        chunk
## 2  dissociated
## 3        chunk
## 4  dissociated
## 5        chunk
## 6  dissociated
## 7        chunk
## 8  dissociated
## 9        chunk
## 10 dissociated
## 11       chunk
## 12 dissociated
## 13       chunk
## 14 dissociated
## 15       chunk
## 16 dissociated
```

Now we can load in the files and create a counts matrix. Note that this shouldn't be stranded data, so we'll use column 2 of the STAR counts files.

```
counts <- matrix(nrow = 36601, ncol = nrow(colData))
for (i in 1:nrow(colData)){
  newcounts <- fread(colData$fileName[i])
  newcounts <- newcounts[-c(1:4), ]
  counts[, i] <- newcounts$V2
  if (i == 1) {
    rownames(counts) <- newcounts$V1
  } else{
      assert(rownames(counts) == newcounts$V1)
  }
}
rm(newcounts); gc()
```

```
##            used  (Mb) gc trigger  (Mb) max used  (Mb)
## Ncells  7762737 414.6   14530084 776.0  9146677 488.5
## Vcells 13885579 106.0   22156564 169.1 18393948 140.4
```

This counts matrix has the genes listed by their Ensembl IDs, which is helpful for uniqueness but bad for readability in the downstream analysis. The easiest way I've found to convert ensembl IDs to gene names for this dataset is by loading in a SingleCellExperiment object from the same experiment, where this mapping is automatically stored.

```r
sce_path <- paste(local_data_path, "sce_objects", sep = "/")
sce <- readRDS(paste(sce_path, "pooled_clustered.rds", sep = "/"))
gene_map <- as.data.frame(subset(rowData(sce), select = c("ID", "Symbol")))
gene_map <- gene_map[rownames(counts) == gene_map$ID, ]
rownames(counts) <- gene_map$Symbol
rm(sce); gc()
```

```
##            used  (Mb) gc trigger  (Mb)  max used  (Mb)
## Ncells  8229063 439.5   14530084 776.0  10390976 555.0
## Vcells 14880807 113.6  106364303 811.5 114951124 877.1
```

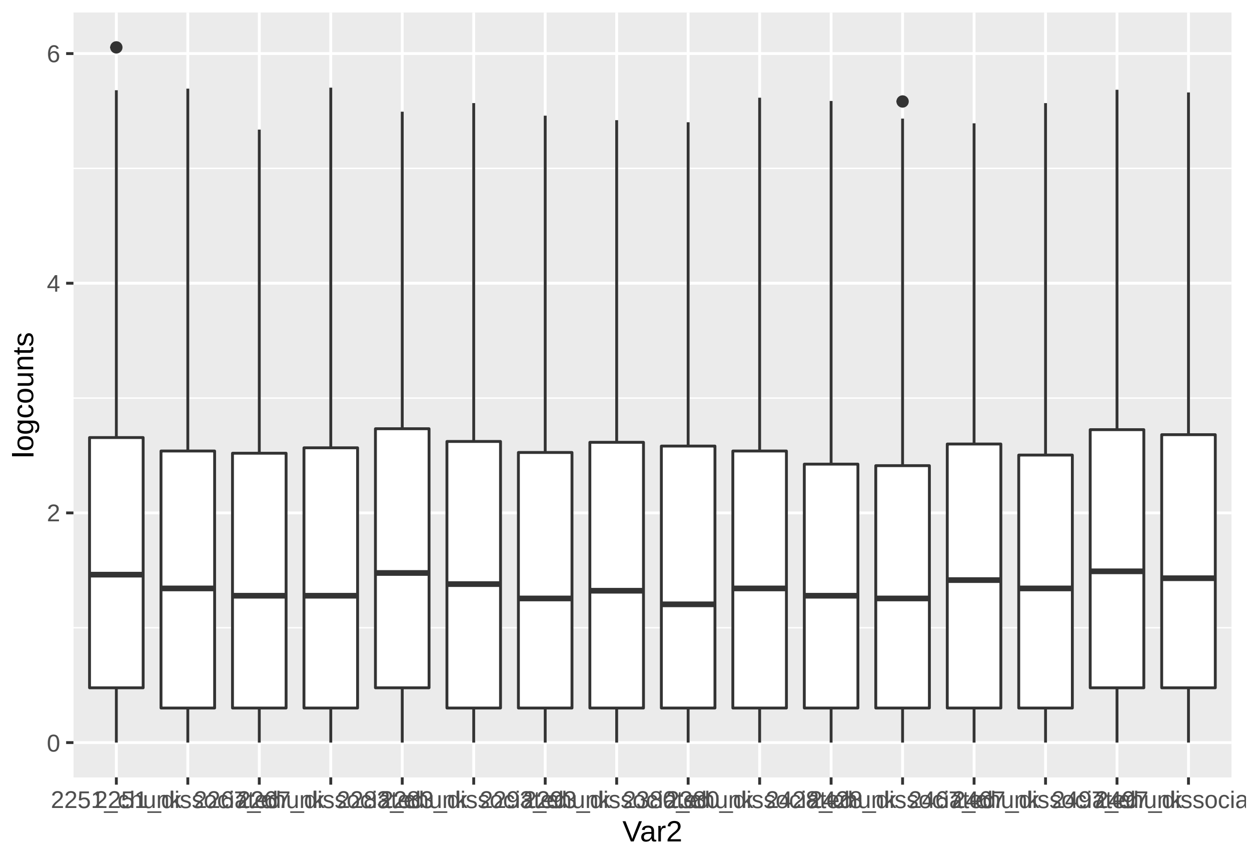
```r
# Create DESeq object
dds <- DESeqDataSetFromMatrix(countData = counts,
                              colData = colData,
                              #design = ~condition)
                              design = ~ sample + condition)
dds
```

```
## class: DESeqDataSet
## dim: 36601 16
## metadata(1): version
## assays(1): counts
## rownames(36601): MIR1302-2HG FAM138A ... AC007325.4 AC007325.2
## rowData names(0):
## colnames: NULL
## colData names(4): id sample fileName condition
```

The tutorial recommends doing some basic pre-filtering of non- or low-expressed genes to speed up computation, estimate the library depth correction factor, and to clean up later visualizations.

```r
colnames(counts) <- colData$id
melted_counts <- melt(counts)
melted_counts$logcounts <- log10(melted_counts$value + 1)

ggplot(melted_counts, aes(x = Var2, y = logcounts)) + geom_boxplot()
```



Okay, <25% of the genes have 0 reads. We'll keep it fairly broad and only remove any gene that has fewer than 10 reads total.

```
keep <- rowSums(counts(dds)) >= 10
dds <- dds[keep, ]
```

Differential expression

If you don't set the condition factor specifically, it can be hard to tell if A is upregulated compared to B or vice versa. We'll set "chunk_ribo" as the reference and look at how dissociated_ribo is upregulated or downregulated compared to that.

```
dds$condition <- relevel(dds$condition, ref = "chunk")
```

```
# Run differential expression
dds <- DESeq(dds)
res <- results(dds)
res
```

```
## log2 fold change (MLE): condition dissociated vs chunk
## Wald test p-value: condition dissociated vs chunk
## DataFrame with 30932 rows and 6 columns
##              baseMean log2FoldChange     lfcSE      stat    pvalue      padj
##             <numeric>      <numeric> <numeric> <numeric> <numeric> <numeric>
## MIR1302-2HG  2.361868       1.528813  1.073110  1.424657 0.1542564  0.328945
## AL627309.1   3.704196      -1.305210  0.649232 -2.010391 0.0443898  0.141594
## AL627309.5  40.129043      -0.607503  0.291200 -2.086204 0.0369602  0.124386
## AP006222.2   5.167057      -1.080821  0.684456 -1.579096 0.1143141  0.268501
## AL669831.2   0.809037      -1.083771  1.602071 -0.676481 0.4987350        NA
## ...                ...            ...       ...       ...       ...       ...
## AC136616.1  17.268668      0.4167940  0.636821  0.654492  0.512795  0.704535
## AC136616.2   0.748387     -0.3435435  3.016457 -0.113890  0.909325        NA
## AC141272.1   2.840799      0.8021821  1.173140  0.683791  0.494107  0.689417
## AC007325.4  21.323278      0.1325569  0.371193  0.357111  0.721009  0.850345
## AC007325.2  15.689291     -0.0560206  0.366899 -0.152687  0.878645  0.941287
```

The tutorial says "shrinkage of effect size (LFC estimates) is useful for visualization and ranking of genes. To shrink the LFC, we pass the dds object to the function lfcShrink. We provide the dds object and the name or number of the coefficient we want to shrink."

```
resLFC <- lfcShrink(dds, coef = "condition_dissociated_vs_chunk", type = "apeglm")
resLFC
```

```
## log2 fold change (MAP): condition dissociated vs chunk
## Wald test p-value: condition dissociated vs chunk
## DataFrame with 30932 rows and 5 columns
##              baseMean log2FoldChange     lfcSE    pvalue      padj
##             <numeric>      <numeric> <numeric> <numeric> <numeric>
## MIR1302-2HG  2.361868      0.1660696  0.438301 0.1542564  0.328945
## AL627309.1   3.704196     -0.5727097  0.640803 0.0443898  0.141594
## AL627309.5  40.129043     -0.4596373  0.285304 0.0369602  0.124386
## AP006222.2   5.167057     -0.3296072  0.486707 0.1143141  0.268501
## AL669831.2   0.809037     -0.0662259  0.414310 0.4987350        NA
## ...               ...            ...       ...       ...       ...
## AC136616.1  17.268668     0.10718911  0.371424  0.512795  0.704535
## AC136616.2   0.748387    -0.00879064  0.413925  0.909325        NA
## AC141272.1   2.840799     0.06084160  0.405768  0.494107  0.689417
## AC007325.4  21.323278     0.07319792  0.282329  0.721009  0.850345
## AC007325.2  15.689291    -0.03108450  0.276522  0.878645  0.941287
```

A quick summary of our differential expression results, at both a 0.1 and 0.05 FDR.

```
summary(res)
```

```
##
## out of 30932 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 3955, 13%
## LFC < 0 (down)     : 3598, 12%
## outliers [1]       : 0, 0%
## low counts [2]     : 2999, 9.7%
## (mean count < 2)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
sum(res$padj  < 0.1, na.rm = TRUE)
```

```
## [1] 7553
```

```
res05 <- results(dds, alpha = 0.05)
summary(res05)
```

```
##
## out of 30932 with nonzero total read count
## adjusted p-value < 0.05
## LFC > 0 (up)       : 3142, 10%
## LFC < 0 (down)     : 2694, 8.7%
## outliers [1]       : 0, 0%
## low counts [2]     : 3599, 12%
## (mean count < 2)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

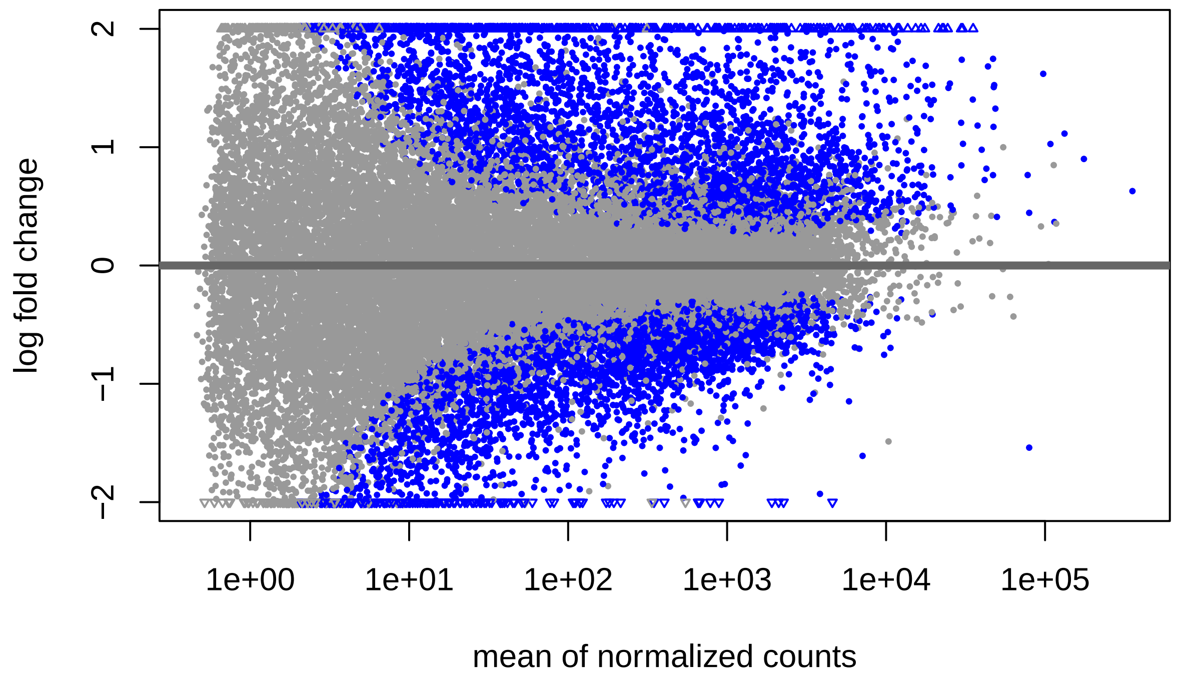
```
sum(res05$padj < 0.05, na.rm = TRUE)
```

```
## [1] 5836
```

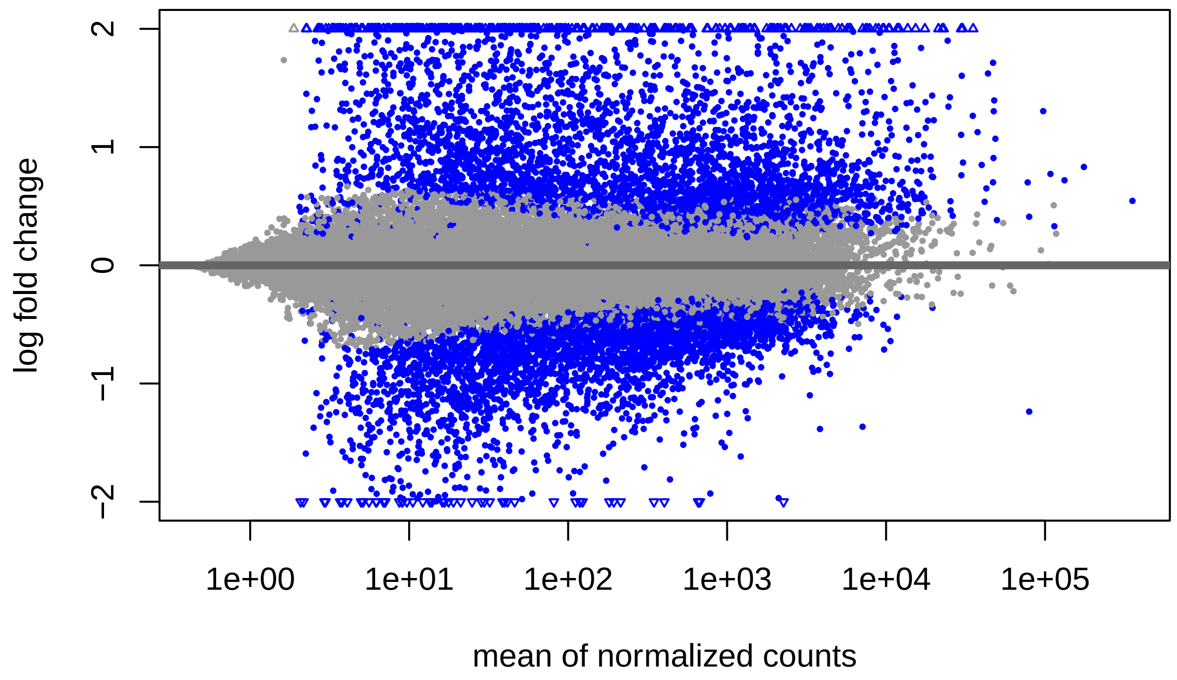
Plotting results

From tutorial: "In DESeq2, the function plotMA shows the log2 fold changes attributable to a given variable over the mean of normalized counts for all the samples in the DESeqDataSet. Points will be colored blue if the adjusted p value is less than 0.1. Points which fall out of the window are plotted as open triangles pointing either up or down."

```
plotMA(res, ylim = c(-2, 2))
```



```
plotMA(resLFC, ylim = c(-2, 2))
```



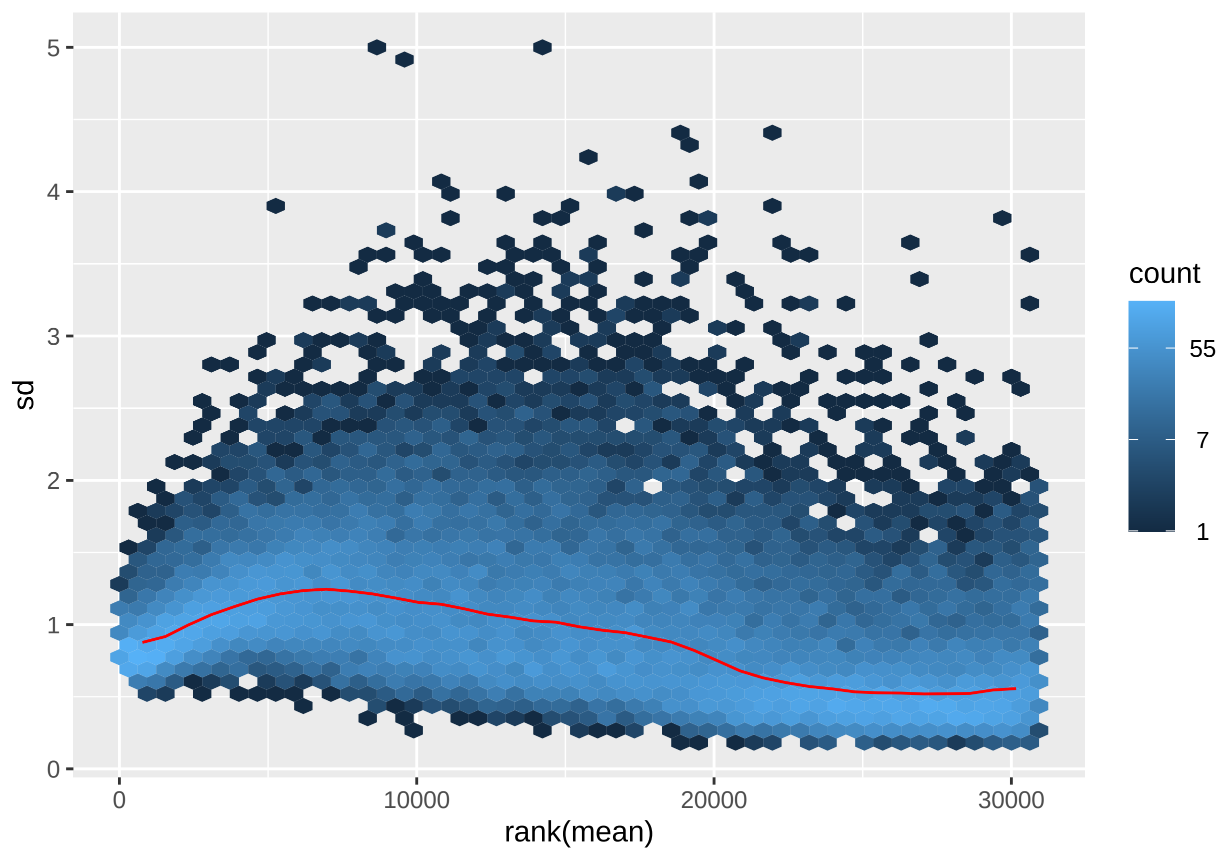
Transformations

The DESeq2 authors recommend the rlog method to adjust for heteroskedasticity in experiments with n < 30. We'll check it and the other vst method they recommend for n > 30.

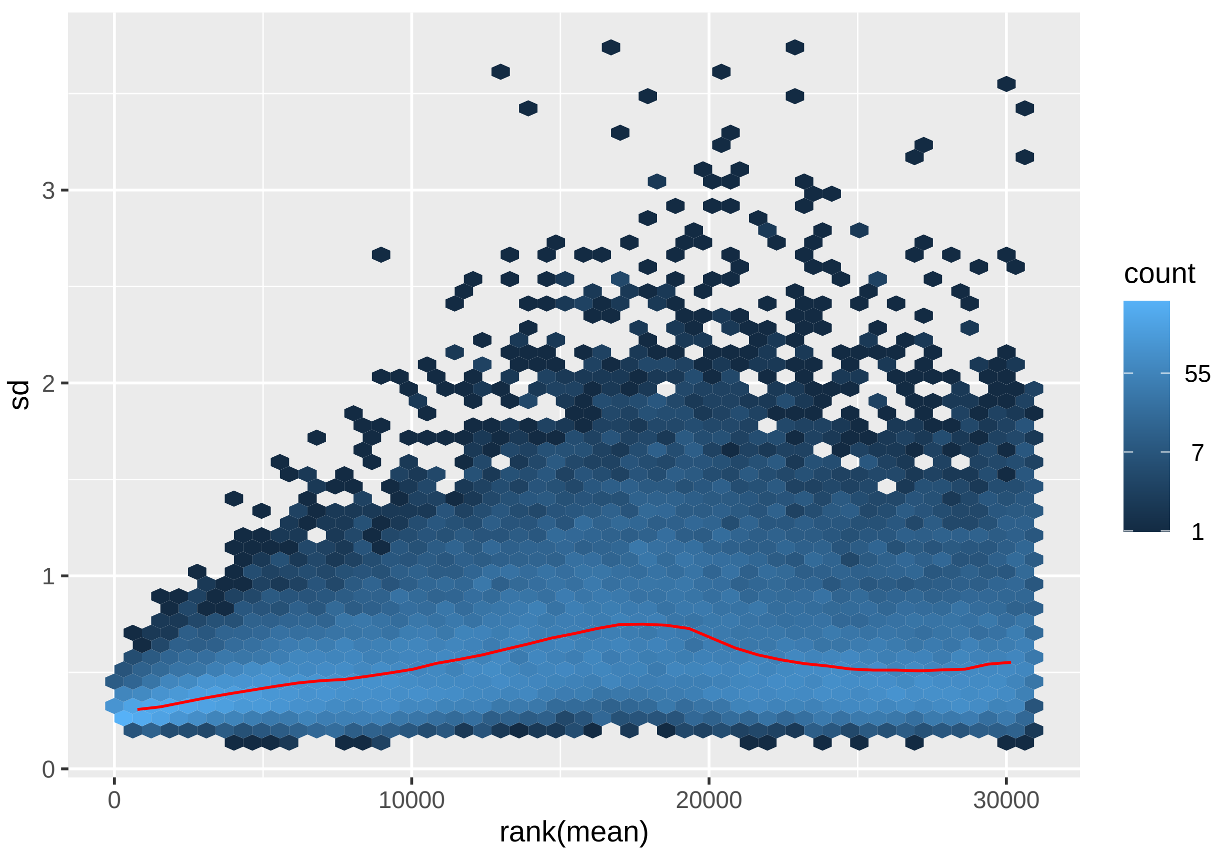
```r
vsd <- vst(dds, blind = FALSE)
rld <- rlog(dds, blind = FALSE)
```

The meanSdPlot plots the mean (as ranked values) by standard deviation, if there is heteroskedasticity there should be a flat line across the values, but they say we shouldn't expect it to be perfectly straight.

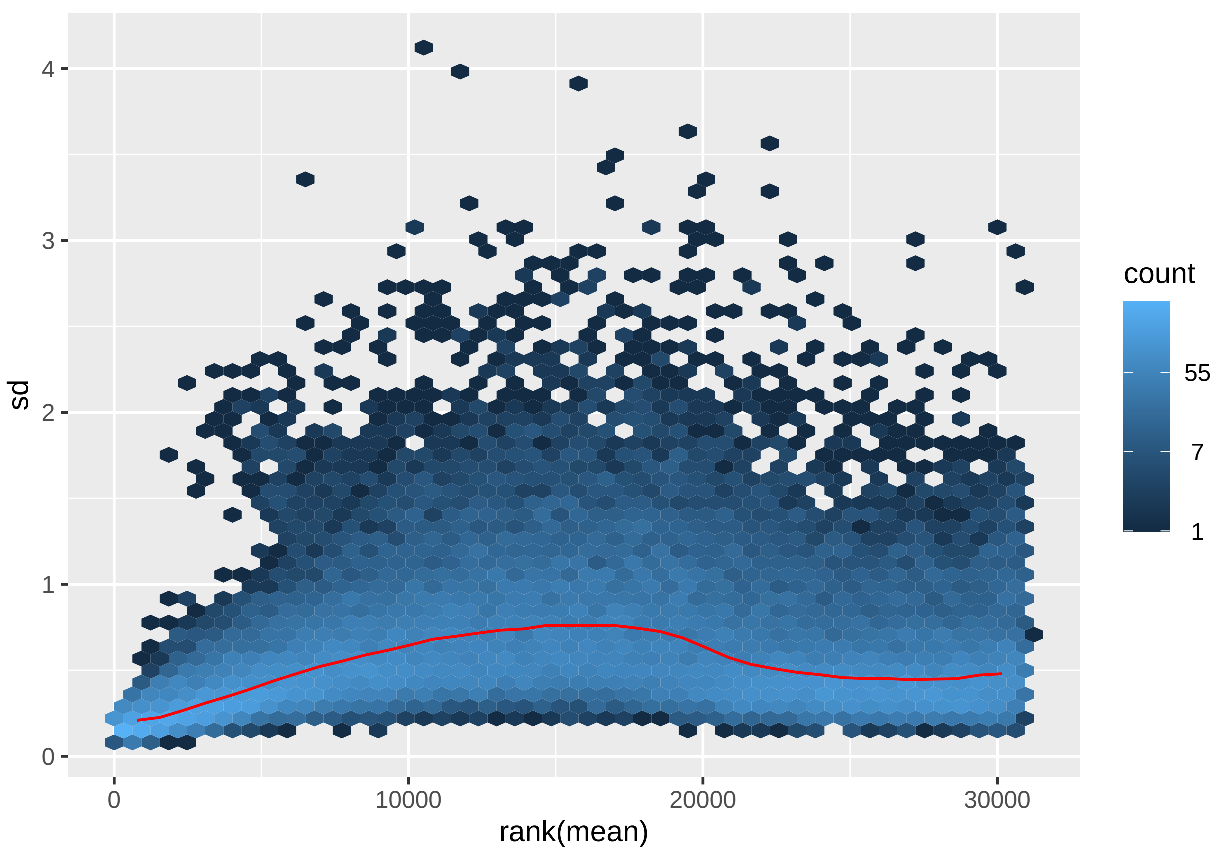
```r
ntd <- normTransform(dds)
meanSdPlot(assay(ntd))
```



```
meanSdPlot(assay(vsd))
```



```
meanSdPlot(assay(rld))
```

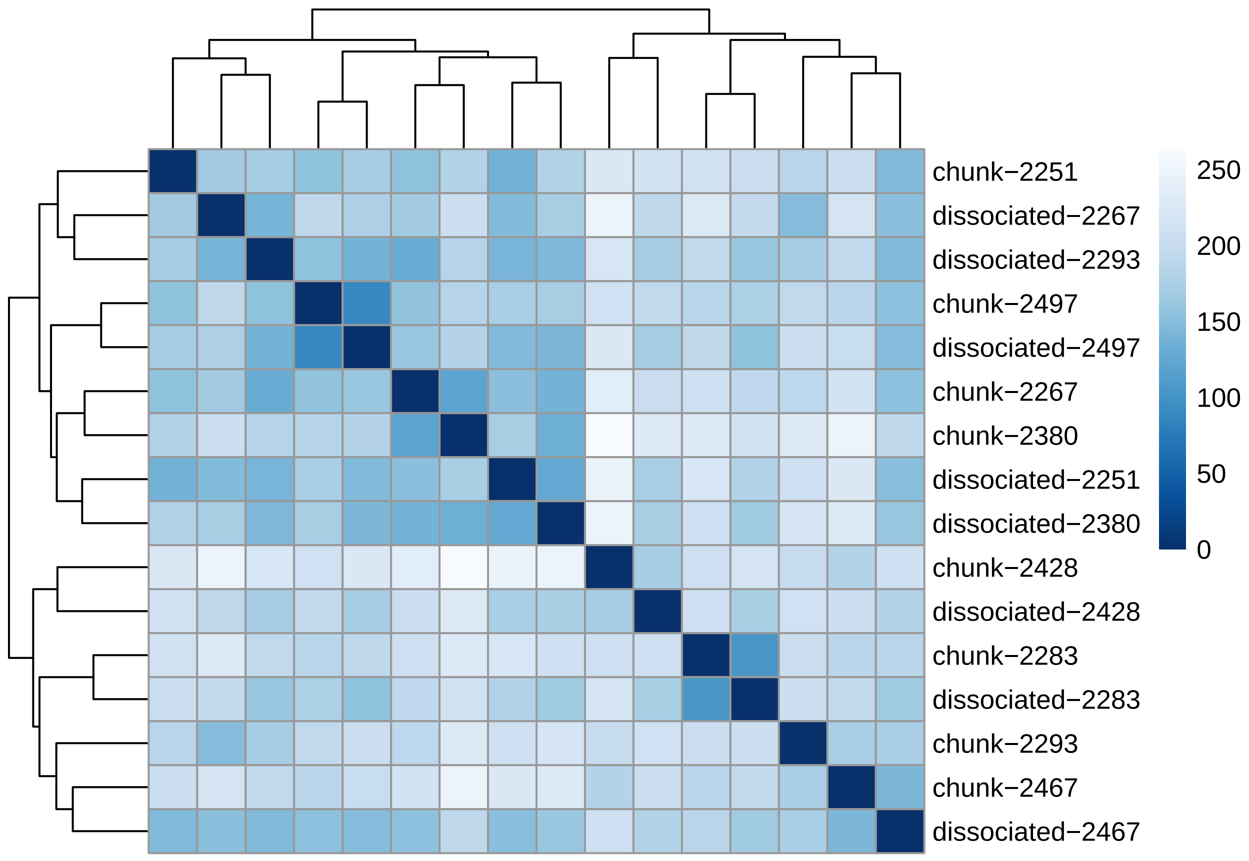


Okay, none of these are *too* heteroskedastic, but the vsd and rlog look slightly better. I'll use this transformation for the sample comparisons.

Sample comparisons

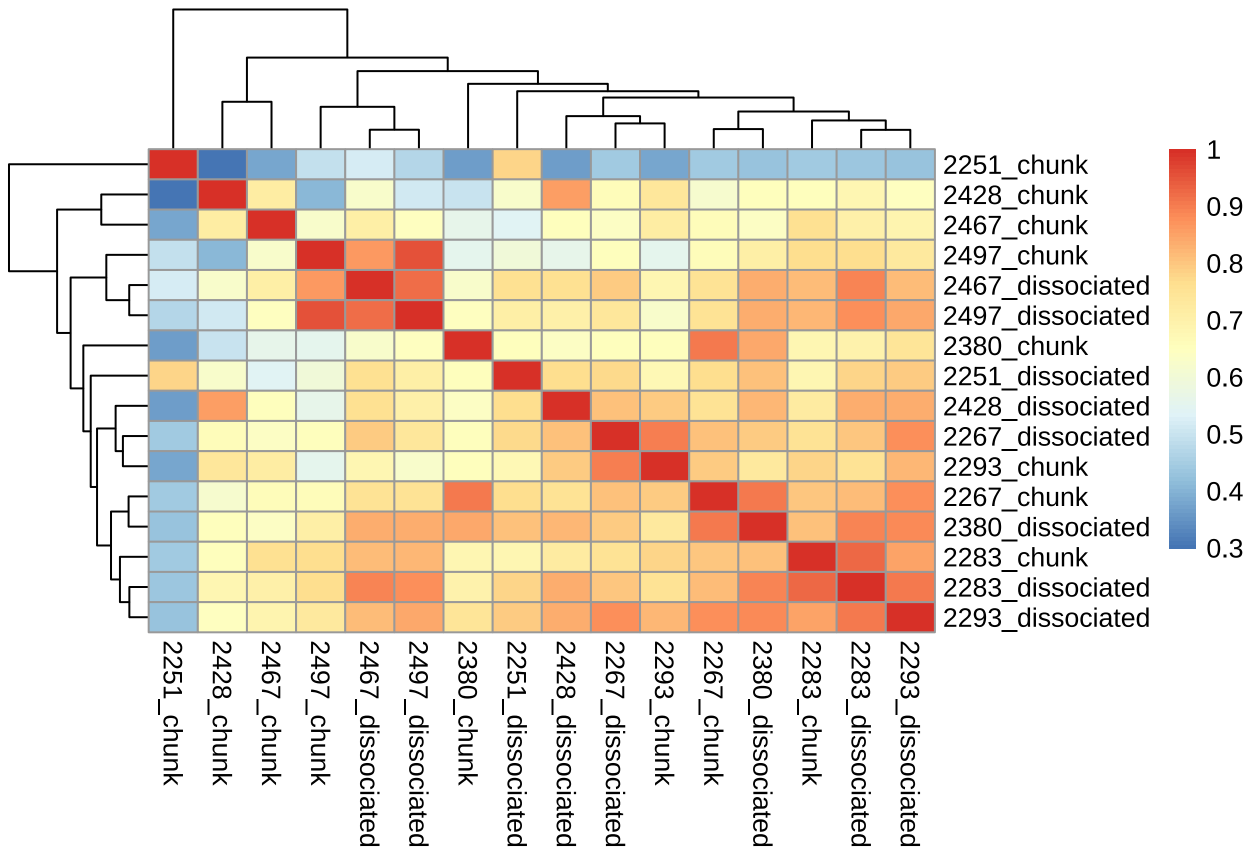
```
sampleDists <- dist(t(assay(rld)))

sampleDistMatrix <- as.matrix(sampleDists)
rownames(sampleDistMatrix) <- paste(rld$condition, rld$sample, sep = "-")
colnames(sampleDistMatrix) <- NULL
colors <- colorRampPalette( rev(brewer.pal(9, "Blues")) )(255)
pheatmap(sampleDistMatrix,
         clustering_distance_rows = sampleDists,
         clustering_distance_cols = sampleDists,
         col = colors)
```



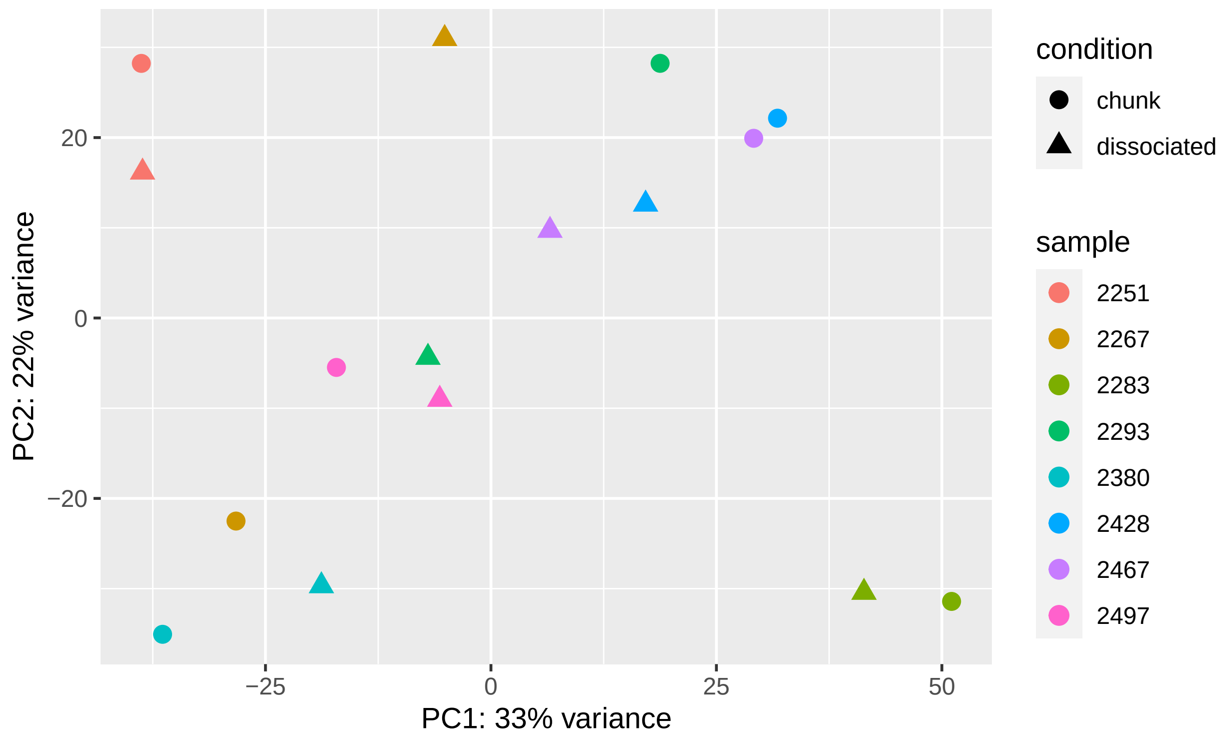
The above is based on Euclidean distances, which can be unreliable in a high-dimensional space like gene expression data. Let's plot the correlations between samples.

```
x<-cor(counts)
pheatmap(x)
```



There's not a ton of clear structure in either plot. It's not exclusively clustering by sample or by dissociation status but by some combination of both. Wonder how this will look on a PCA.

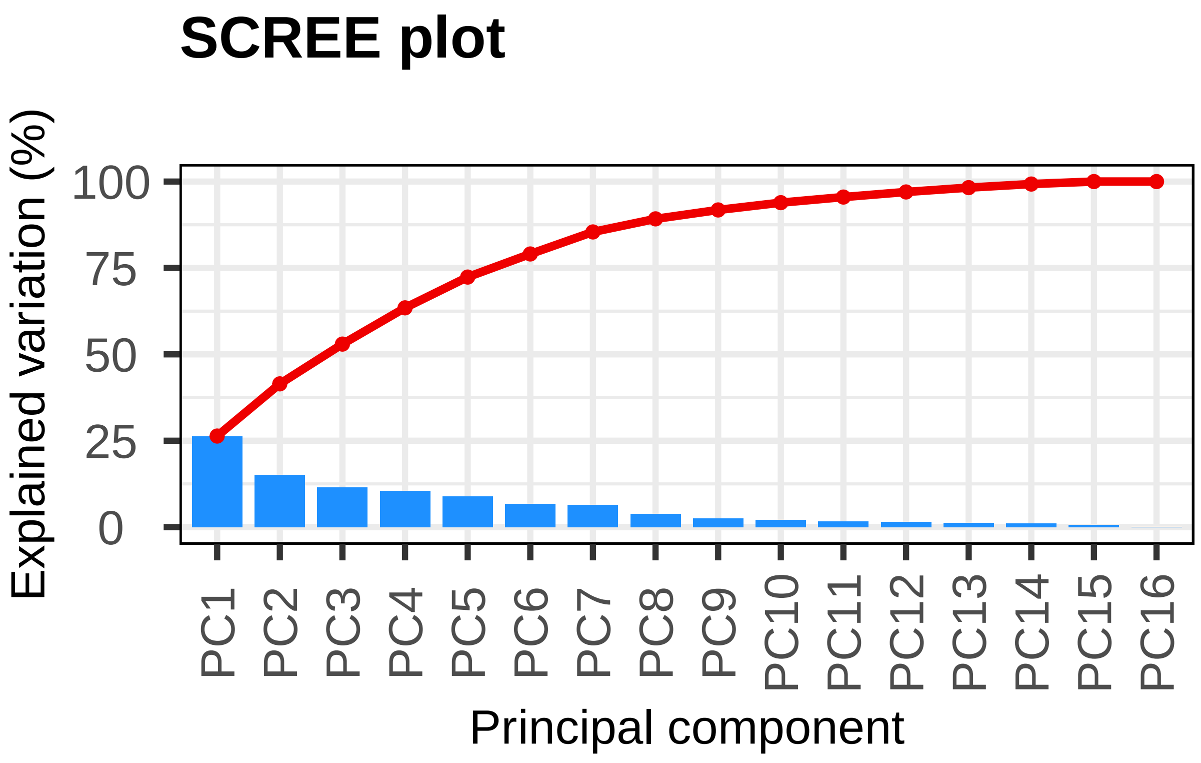
```
pcaData <- plotPCA(rld, intgroup = c("condition", "sample"), returnData = TRUE)
percentVar <- round(100 * attr(pcaData, "percentVar"))
ggplot(pcaData, aes(PC1, PC2, color = sample, shape = condition)) +
  geom_point(size = 3) +
  xlab(paste0("PC1: ", percentVar[1], "% variance")) +
  ylab(paste0("PC2: ", percentVar[2], "% variance")) +
  coord_fixed()
```



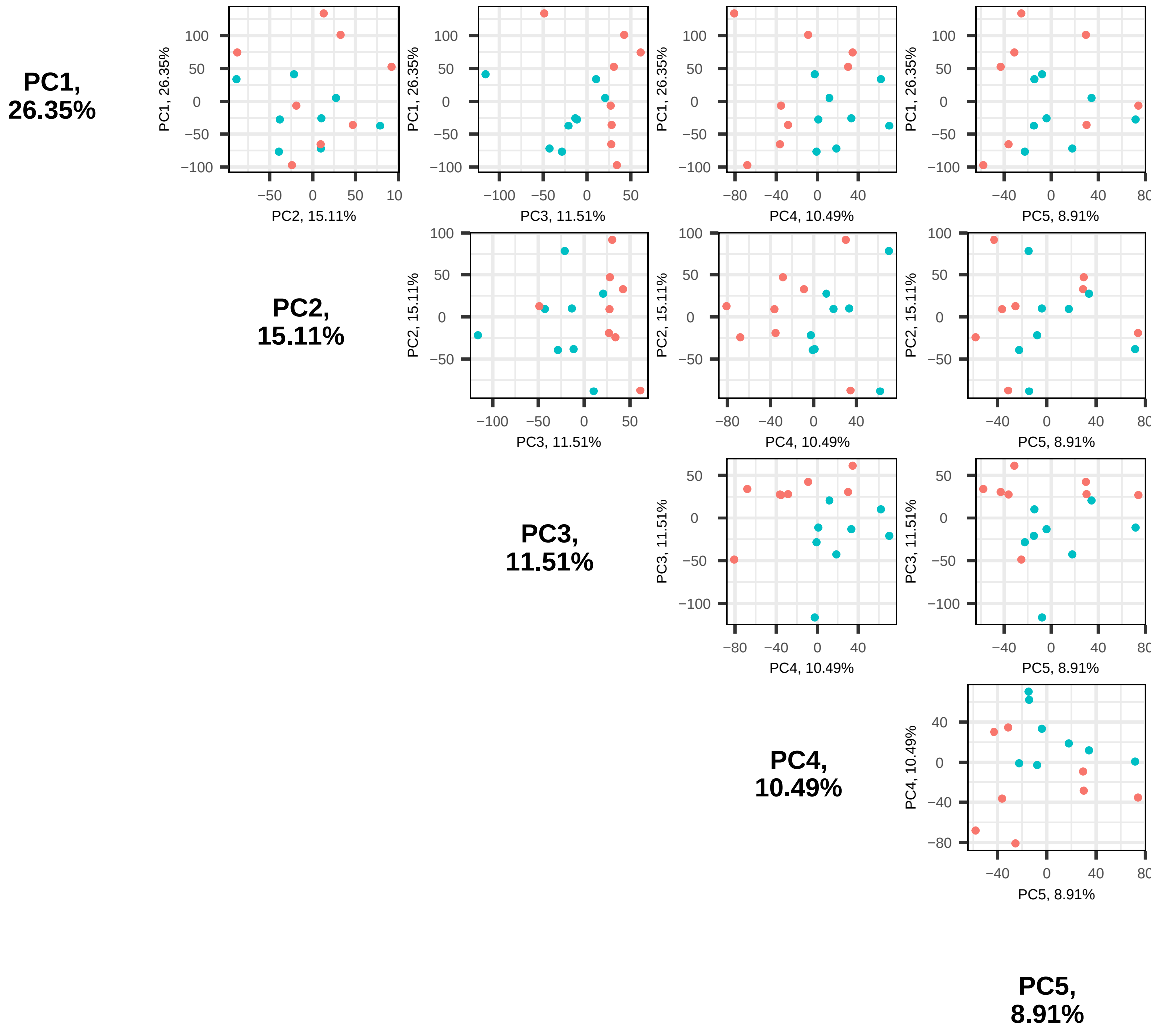
With a couple of exceptions, the samples seem to be largely clustering together, not as much by dissociation status. But it's also worth noting that the purple and blue samples, which do seem to be clustering by dissociation status, had relatively low RIN scores (<7). Maybe a coincidence but maybe not.

For thoroughness, I'm going to check and see if dissociation status is represented in PCs 3 or 4. I'm going to use the PCAtools tutorial (https://bioconductor.org/packages/release/bioc/vignettes/PCAtools/inst/doc/PCAtools.html) for this.

```
p <- pca(assay(rld), metadata = colData(rld), removeVar = 0.1)
screeplot(p, axisLabSize = 18, titleLabSize = 22)
```



```
pairsplot(p, colby = "condition", pointSize = 2)
```



It looks like PC3 is dissociation status.

Conclusions

- There is definitely some differential expression happening between the chunk samples and the dissociated samples, with slightly more differentially upregulated genes in the dissociated cells.
- Our data may benefit from rlog transformation but it doesn't make a huge difference to downstream results.
- At a high level, our samples are clustering more by patient than by dissociation status, but not perfectly, particularly in the lower quality data (the ones with lower RIN scores).
- Dissociation status is definitely informing samples' distance from one another, but it's not the predominant factor (it's roughly the third principal component).

With this information in hand, we'll try to analyze our differential expression results with GSEA.

```r
# Save data
deseq_path <- paste(local_data_path, "deseq2_output", sep = "/")
saveRDS(dds, file = paste(deseq_path, "chunk_vs_dissociated_data.rds", sep = "/"))

# Save results files
saveRDS(res, file = paste(deseq_path, "chunk_vs_dissociated_FDR_0.1.rds", sep = "/"))
saveRDS(res05, file = paste(deseq_path, "chunk_vs_dissociated_FDR_0.05.rds", sep = "/"))
```