Differential expression: rRNA depletion vs poly-A capture

Ariel Hippen

2022-07-28

Contents

DESeq2 pipeline

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

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

```
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 dissociated samples
# Note: data_path is 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("dissociated", 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("ribo", sampleFiles),
                          "ribo", "polyA")
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_polyA   2251
## 2   2251_ribo   2251
## 3  2267_polyA   2267
## 4   2267_ribo   2267
## 5  2283_polyA   2283
## 6   2283_ribo   2283
## 7  2293_polyA   2293
## 8   2293_ribo   2293
## 9  2380_polyA   2380
## 10  2380_ribo   2380
## 11 2428_polyA   2428
## 12  2428_ribo   2428
## 13 2467_polyA   2467
## 14  2467_ribo   2467
## 15 2497_polyA   2497
## 16  2497_ribo   2497
##
## 1  /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2251/dissociated_polyA/STAR/ReadsPerG
## 2   /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2251/dissociated_ribo/STAR/ReadsPerG
## 3  /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2267/dissociated_polyA/STAR/ReadsPerG
## 4   /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2267/dissociated_ribo/STAR/ReadsPerG
## 5  /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2283/dissociated_polyA/STAR/ReadsPerG
## 6   /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2283/dissociated_ribo/STAR/ReadsPerG
## 7  /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2293/dissociated_polyA/STAR/ReadsPerG
## 8   /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2293/dissociated_ribo/STAR/ReadsPerG
```

```
## 9  /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2380/dissociated_polyA/STAR/ReadsPerG
## 10  /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2380/dissociated_ribo/STAR/ReadsPerG
## 11 /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2428/dissociated_polyA/STAR/ReadsPerG
## 12  /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2428/dissociated_ribo/STAR/ReadsPerG
## 13 /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2467/dissociated_polyA/STAR/ReadsPerG
## 14  /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2467/dissociated_ribo/STAR/ReadsPerG
## 15 /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2497/dissociated_polyA/STAR/ReadsPerG
## 16  /home/ariel/Documents/scRNA/sc-cancer-hgsc/data/bulk_tumors/2497/dissociated_ribo/STAR/ReadsPerG
##    condition
## 1      polyA
## 2       ribo
## 3      polyA
## 4       ribo
## 5      polyA
## 6       ribo
## 7      polyA
## 8       ribo
## 9      polyA
## 10      ribo
## 11     polyA
## 12      ribo
## 13     polyA
## 14      ribo
## 15     polyA
## 16      ribo
```

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  7762738 414.6   14530197 776.0  9146667 488.5
## Vcells 13885648 106.0   22156184 169.1 18392076 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.

```
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  8229064 439.5   14530197 776.0  10390977 555.0
## Vcells 14880871 113.6  106364364 811.5 114951188 877.1
```

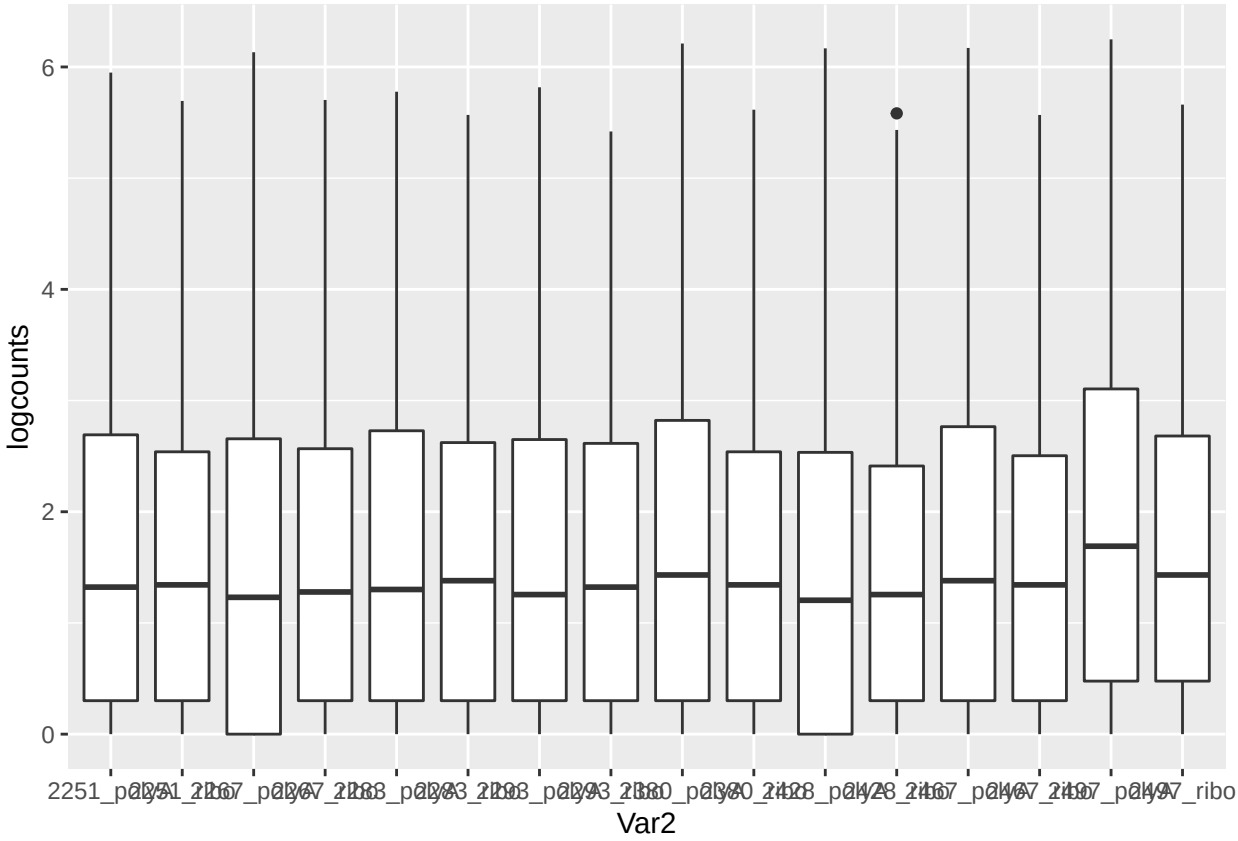
```
# Create DESeq object
dds <- DESeqDataSetFromMatrix(countData = counts,
                              colData = colData,
                              #design = ~condition)
                              design = ~ sample + condition)
dds
```

```
## 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.

```
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, in some of the samples at least 25% of the genes are not expressed. Let's go a little more conservative and remove any genes that have fewer than 20 reads total.

```
keep <- rowSums(counts(dds)) >= 20
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 "ribo" as the reference and look at how dissociated_polyA is upregulated or downregulated compared to that.

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

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

```
## log2 fold change (MLE): condition polyA vs ribo
## Wald test p-value: condition polyA vs ribo
## DataFrame with 28862 rows and 6 columns
##              baseMean log2FoldChange     lfcSE       stat      pvalue
##             <numeric>      <numeric> <numeric>  <numeric>   <numeric>
```

```
## MIR1302-2HG   3.22241      -0.317498  0.765856  -0.414566 6.78460e-01
## AL627309.1    4.31123       1.265121  0.622455   2.032469 4.21062e-02
## AL627309.2    1.14983       2.086752  1.513010   1.379206 1.67831e-01
## AL627309.5   57.08226       1.029908  0.193450   5.323884 1.01575e-07
## AP006222.2    6.47429       1.425878  0.536181   2.659322 7.82980e-03
## ...               ...            ...       ...        ...         ...
## AC136352.3    0.64392     -0.1867658  3.038270 -0.0614711   0.9509840
## AC136616.1   18.57115      0.3605074  0.315927  1.1411102   0.2538241
## AC141272.1    5.90918      1.0962888  0.654842  1.6741275   0.0941056
## AC007325.4   22.38544     -0.5540068  0.293925 -1.8848601   0.0594487
## AC007325.2   17.06982     -0.0439485  0.303788 -0.1446682   0.8849729
##                    padj
##               <numeric>
## MIR1302-2HG 7.62409e-01
## AL627309.1  7.66779e-02
## AL627309.2  2.51459e-01
## AL627309.5  4.69515e-07
## AP006222.2  1.70322e-02
## ...                ...
## AC136352.3     0.966796
## AC136616.1     0.354369
## AC141272.1     0.154226
## AC007325.4     0.103731
## AC007325.2     0.921186
```

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_polyA_vs_ribo", type = "apeglm")
resLFC
```

```
## log2 fold change (MAP): condition polyA vs ribo
## Wald test p-value: condition polyA vs ribo
## DataFrame with 28862 rows and 5 columns
##              baseMean log2FoldChange     lfcSE      pvalue        padj
##             <numeric>      <numeric> <numeric>   <numeric>   <numeric>
## MIR1302-2HG   3.22241      -0.116194  0.506398 6.78460e-01 7.62409e-01
## AL627309.1    4.31123       0.816695  0.612900 4.21062e-02 7.66779e-02
## AL627309.2    1.14983       0.324770  0.722068 1.67831e-01 2.51459e-01
## AL627309.5   57.08226       0.986808  0.194622 1.01575e-07 4.69515e-07
## AP006222.2    6.47429       1.094371  0.561236 7.82980e-03 1.70322e-02
## ...               ...            ...       ...         ...         ...
## AC136352.3    0.64392      0.0057535  0.620128   0.9509840    0.966796
## AC136616.1   18.57115      0.2926305  0.290822   0.2538241    0.354369
## AC141272.1    5.90918      0.6095973  0.598326   0.0941056    0.154226
## AC007325.4   22.38544     -0.4707885  0.283017   0.0594487    0.103731
## AC007325.2   17.06982     -0.0349172  0.273604   0.8849729    0.921186
```

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

```
summary(res)
```

```
##
## out of 28862 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 8200, 28%
## LFC < 0 (down)     : 8237, 29%
## outliers [1]       : 0, 0%
## low counts [2]     : 0, 0%
## (mean count < 1)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

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

```
## [1] 16437
```

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

```
##
## out of 28862 with nonzero total read count
## adjusted p-value < 0.05
## LFC > 0 (up)       : 7588, 26%
## LFC < 0 (down)     : 7384, 26%
## outliers [1]       : 0, 0%
## low counts [2]     : 0, 0%
## (mean count < 1)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

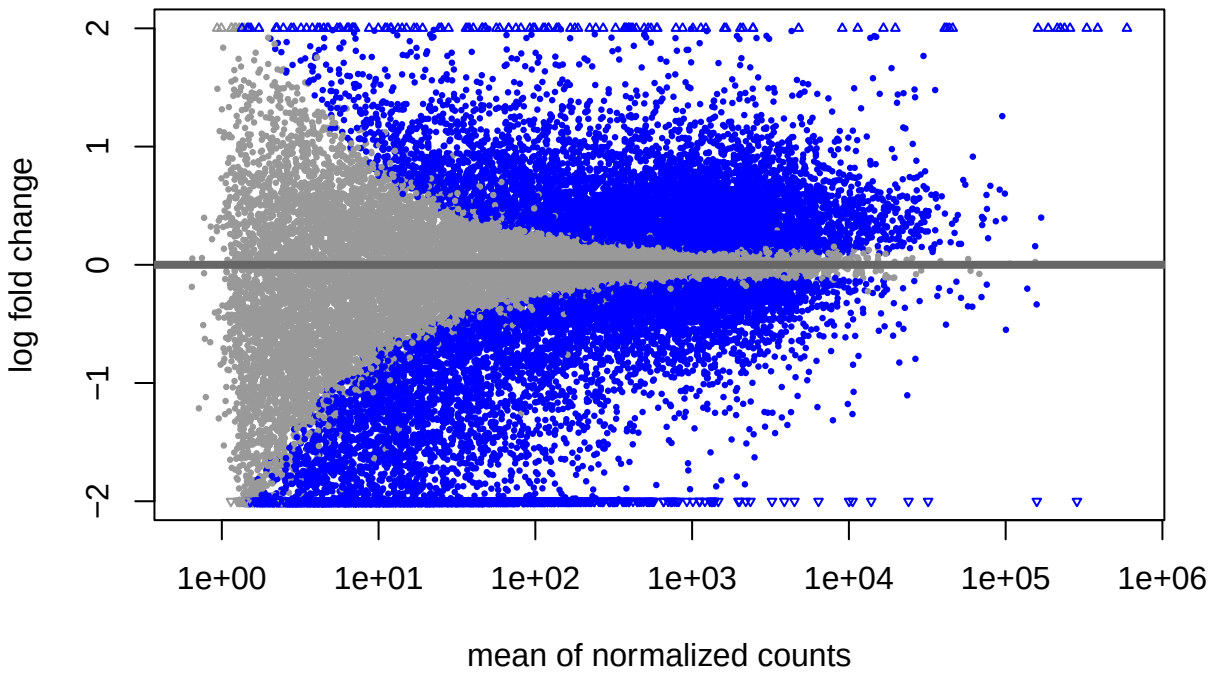
```
sum(res05$padj < 0.05, na.rm = TRUE)
```

```
## [1] 14972
```

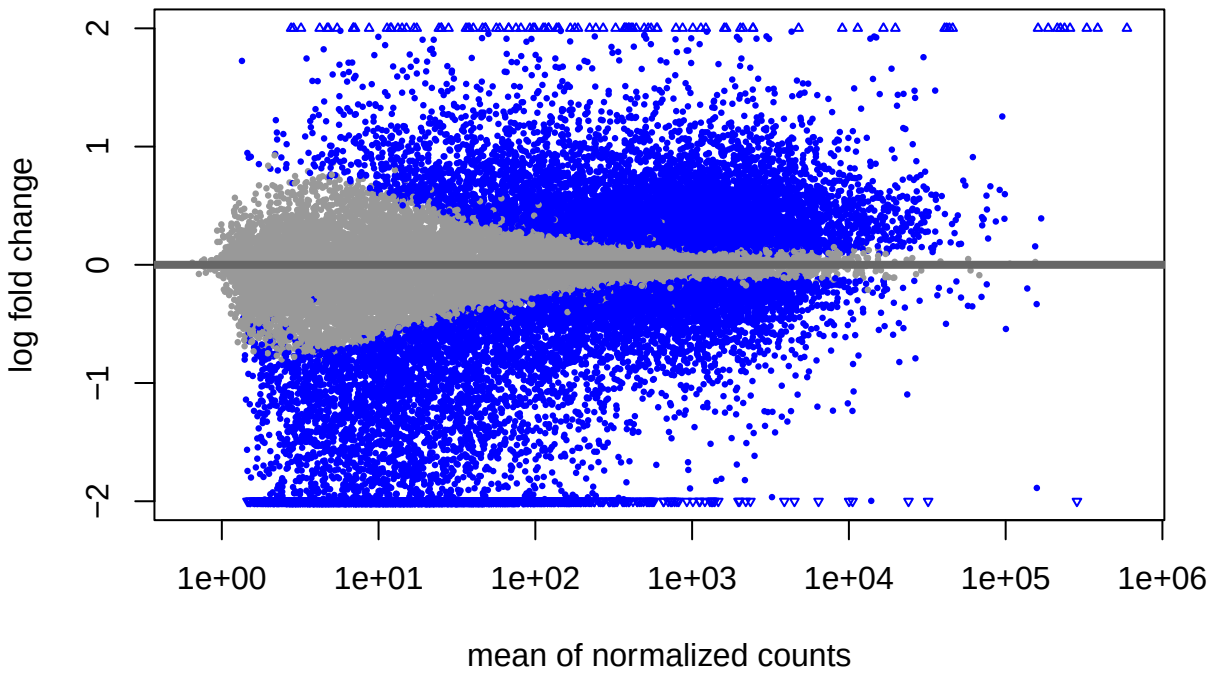
Plotting results

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))
```



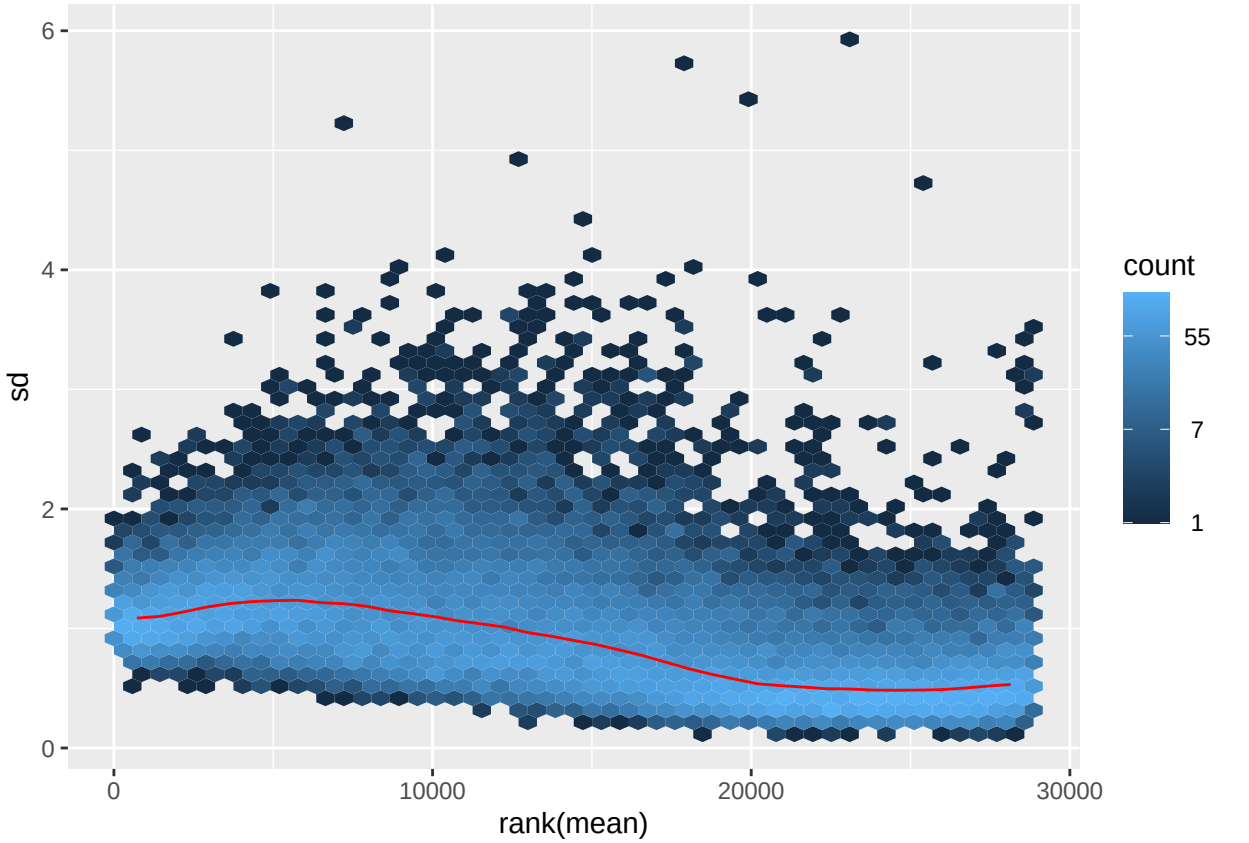
Transformations

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.

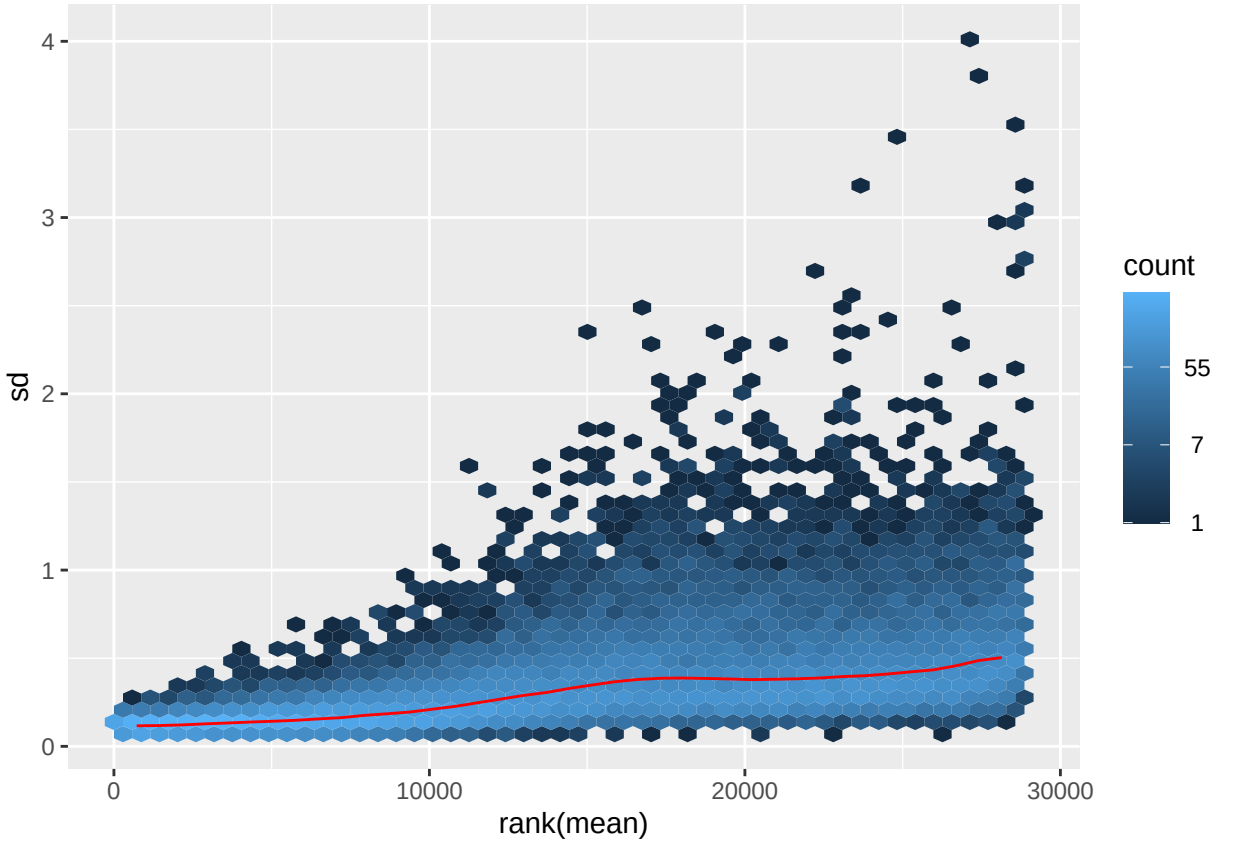
```
vsd <- vst(dds, blind = FALSE)
rld <- rlog(dds, blind = FALSE)
```

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.

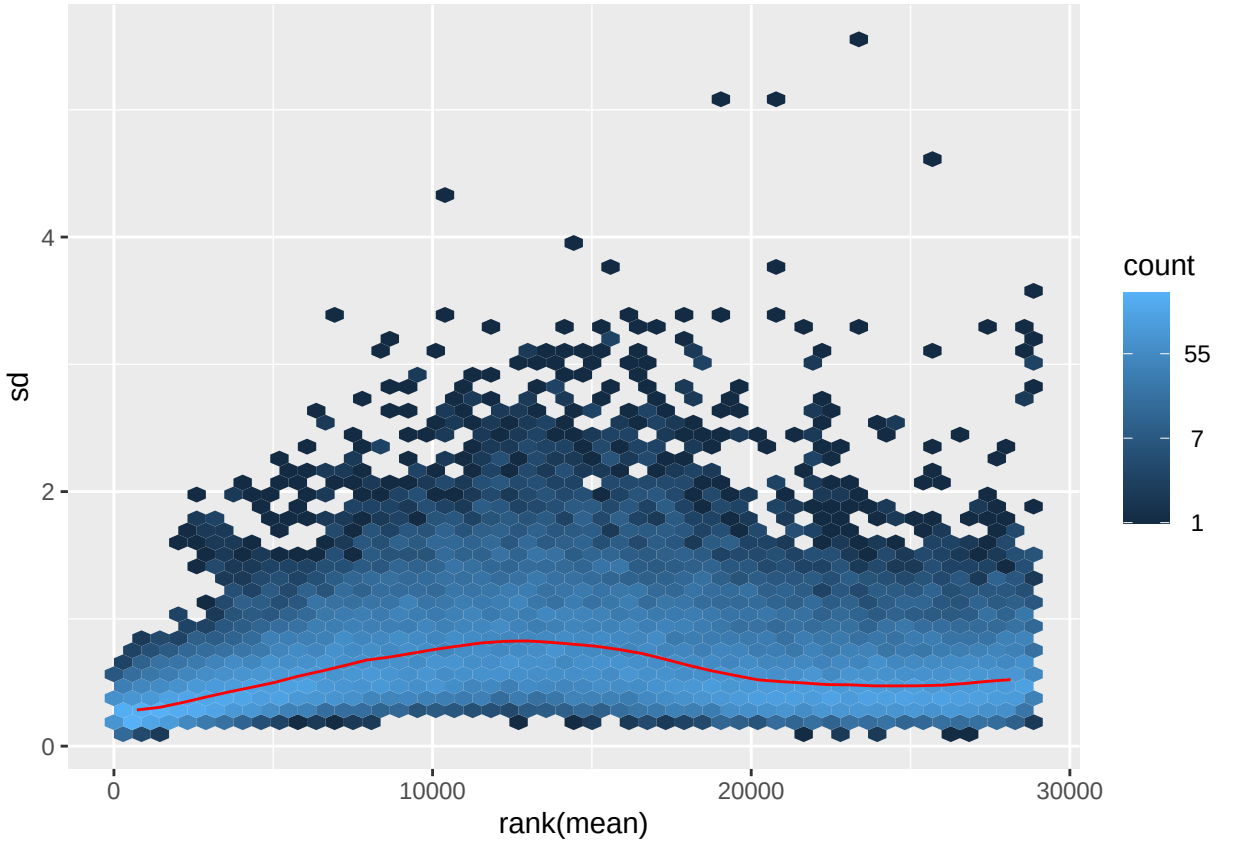
```
ntd <- normTransform(dds)
meanSdPlot(assay(ntd))
```




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




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

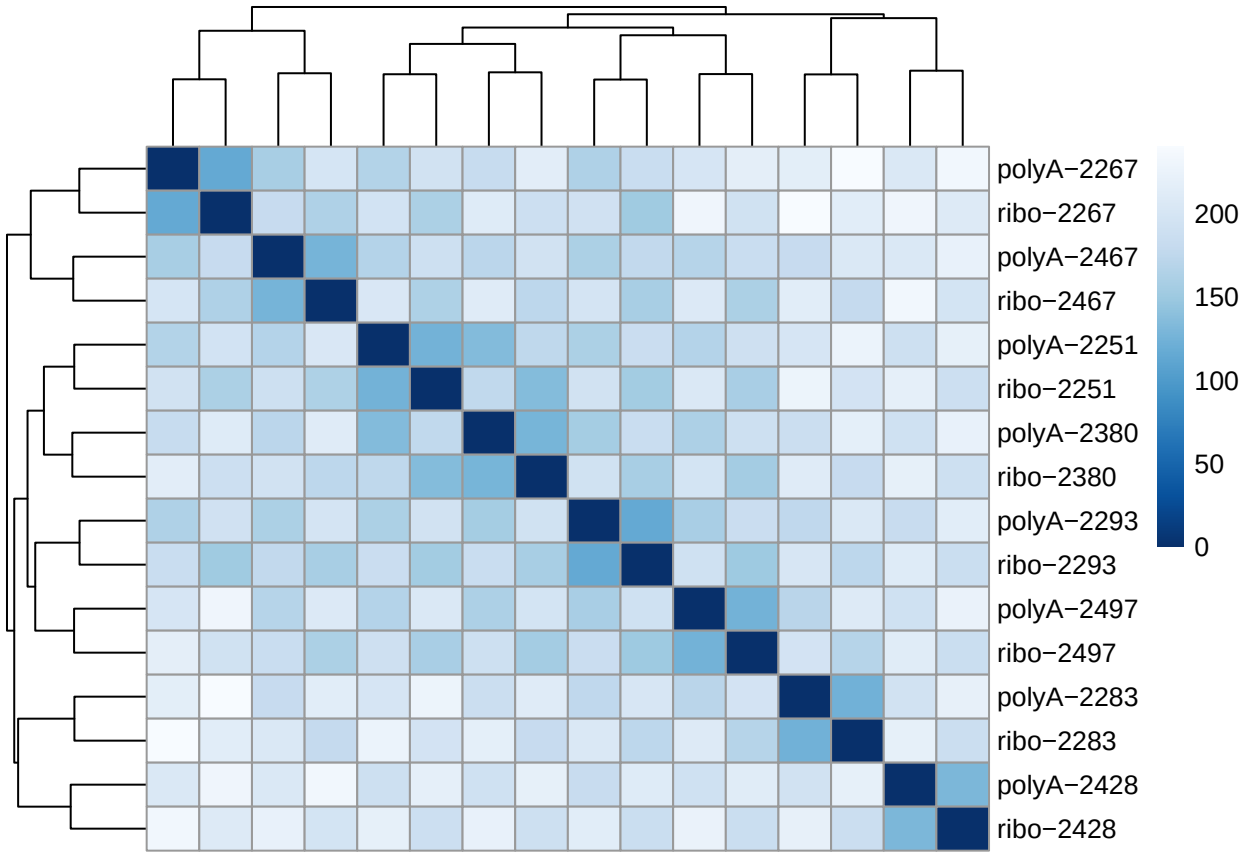


Okay, regular and rlog look okay, vst looks more heteroskedastic now. I'll use rlog for the sample comparisons.

Sample comparisons

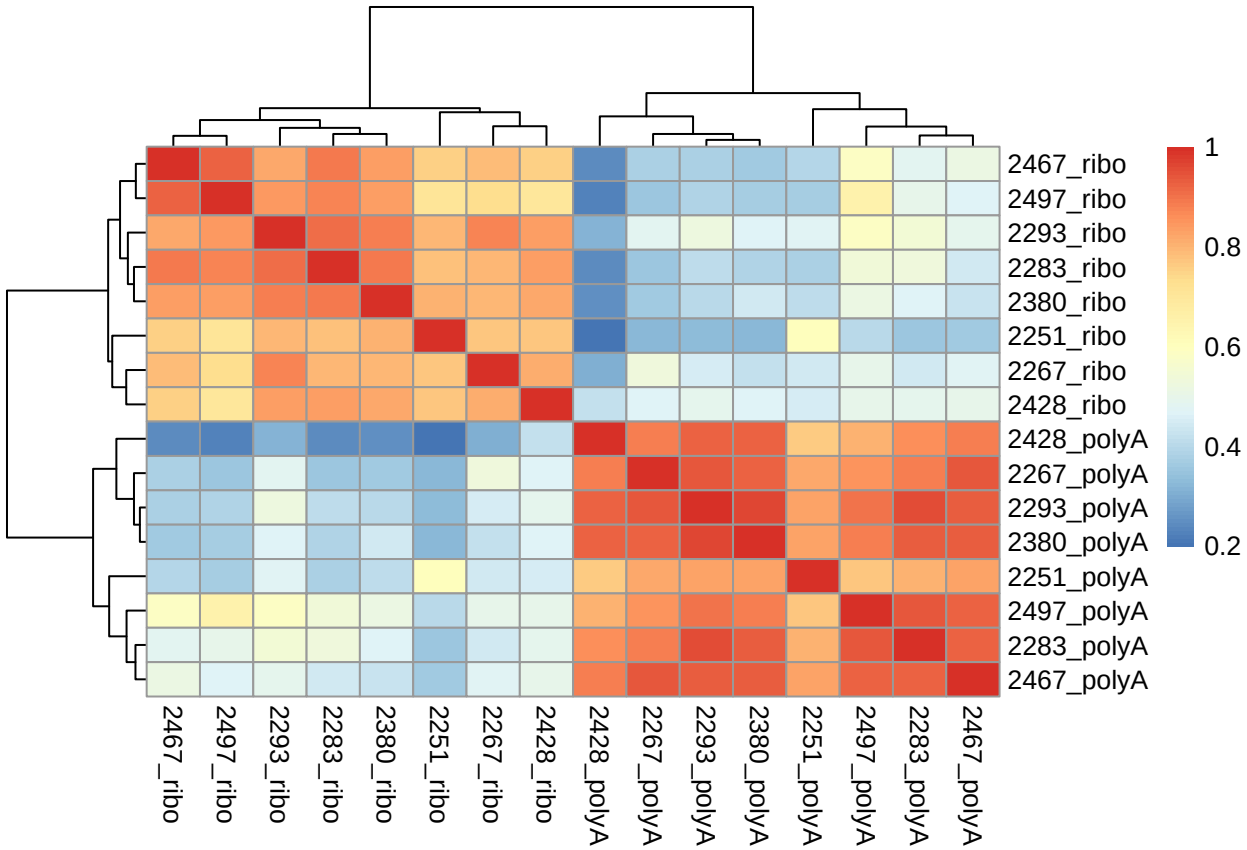
```
sampleDists <- dist(t(assay(rld)))

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)
```



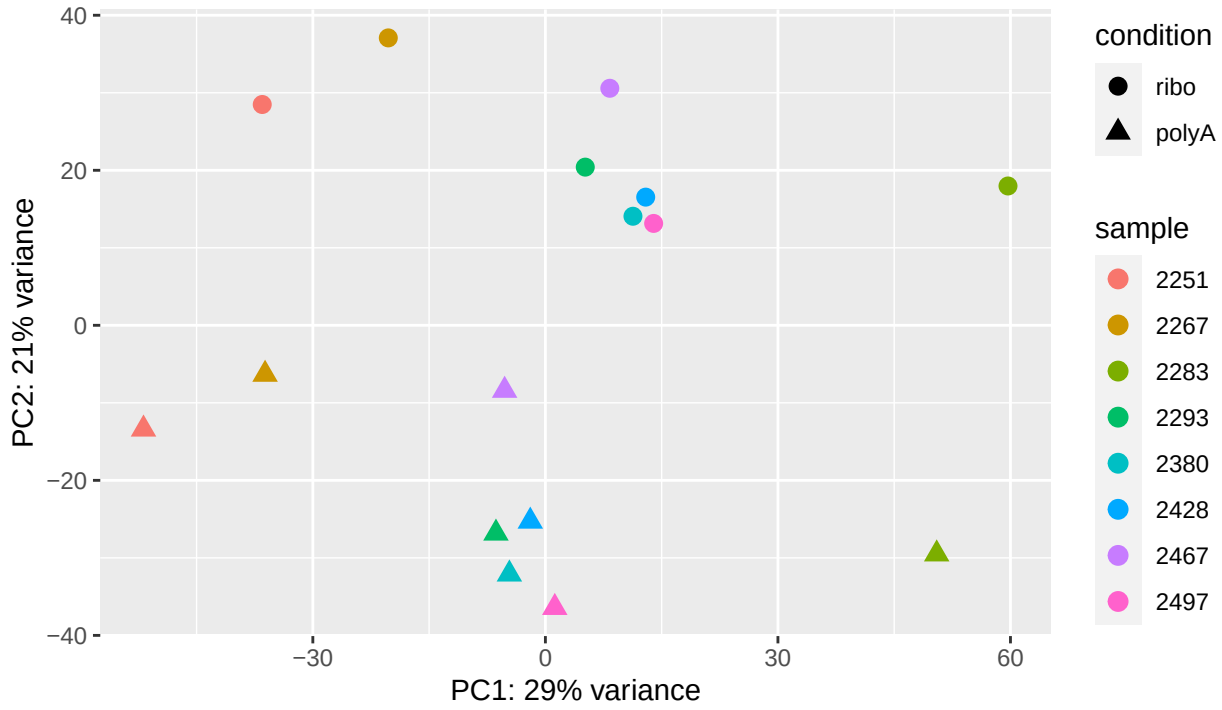
The structure looks fairly sample-specific here.

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



Wow! The correlations are highly based on condition, not sample. Let's see how the PCA looks.

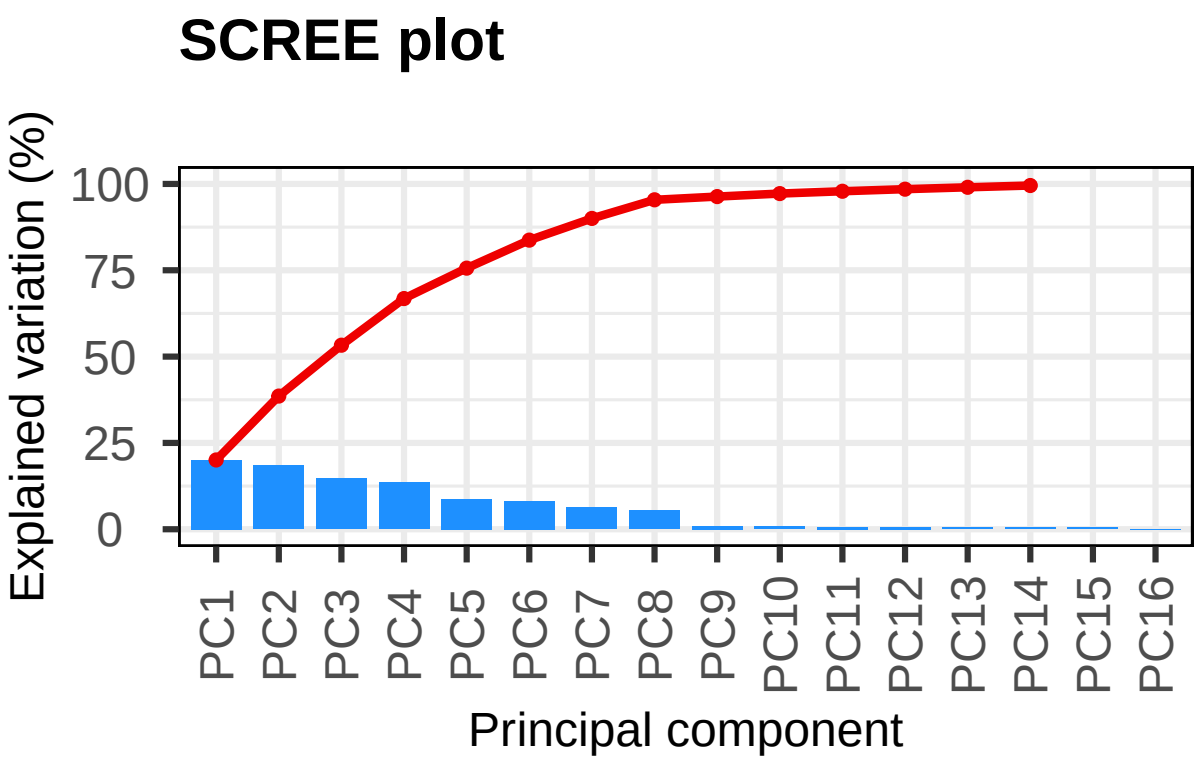
```
pcaData <- plotPCA(rld, intgroup = c("condition", "sample"), returnData = TRUE)
percentVar <- round(100 * attr(pcaData, "percentVar"))
ggplot(pcaData, aes(PC1, PC2, color = sample, shape = condition)) +
  geom_point(size = 3) +
  xlab(paste0("PC1: ", percentVar[1], "% variance")) +
  ylab(paste0("PC2: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



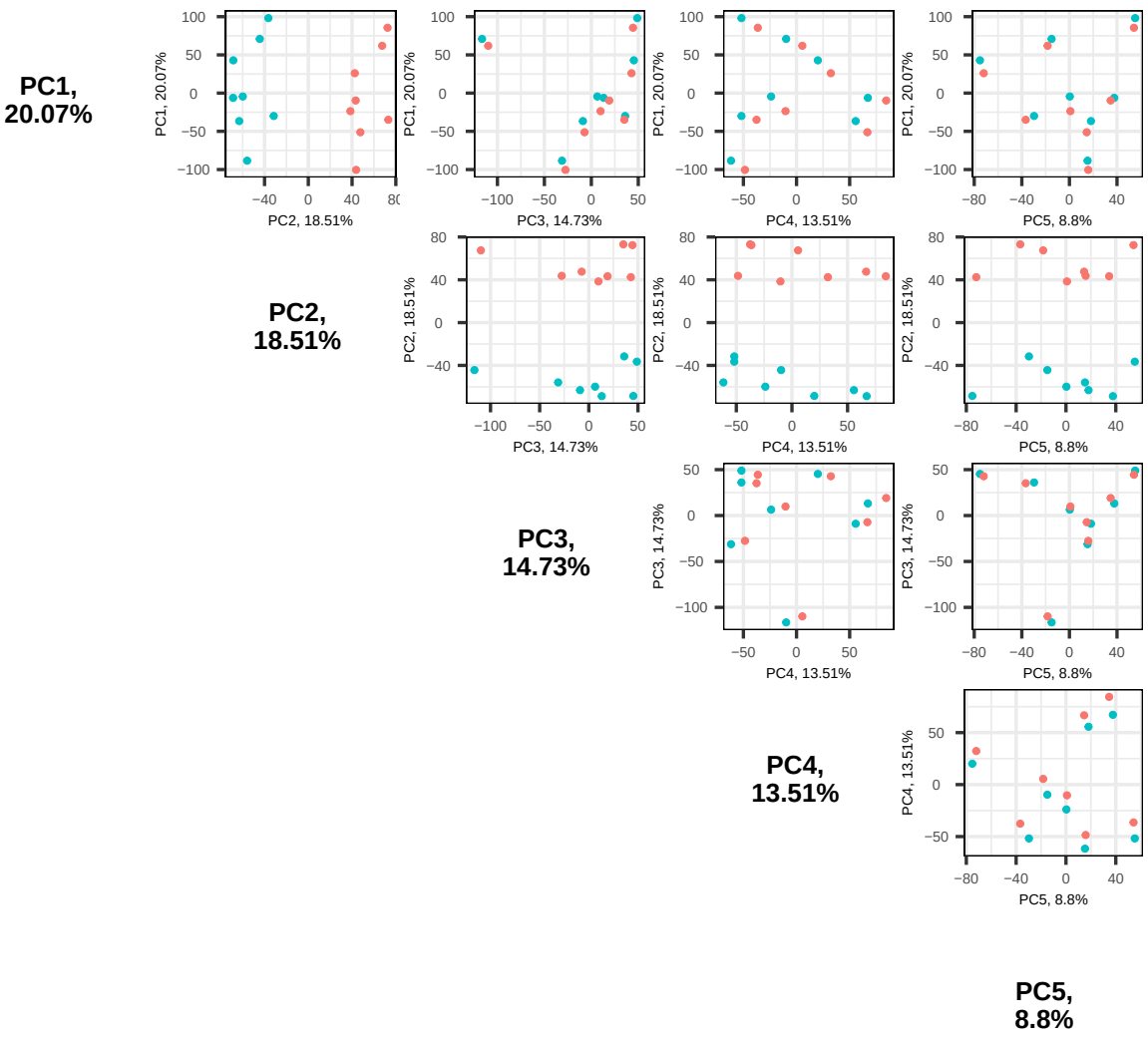
Whoa. PC2 is obviously ribo depletion, it's almost a little disturbing how perfect the shift is. Not sure what PC1 is besides that it's sample-specific. It doesn't seem to have any relationship to RIN scores, though.

For thoroughness, I'm going to check the lower PCs. 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)
```



PC1 is still a bit of a mystery, PC2 is mRNA enrichment method, and PC3-5 also seem to be some sort of sample-specific structure.

Conclusions

- There is definitely differential expression happening between the ribo-depleted and poly-A captured cells, with what seems like more differentially upregulated genes in the poly-A 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 mostly by patient.
- mRNA enrichment method is definitely informing samples' distance from one another, as the second 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, "ribo_vs_polyA_data.rds", sep = "/"))

# Save results files
saveRDS(res, file = paste(deseq_path, "ribo_vs_polyA_FDR_0.1.rds", sep = "/"))
saveRDS(res05, file = paste(deseq_path, "ribo_vs_polyA_FDR_0.05.rds", sep = "/"))
```