Final Project: RNA-Seq Analysis

Code

Sandy Kim (404830610)

In this final project, I aim to explore various methods of RNA-seq data analysis in order to (1) try out different methods of analysis and (2) gain familiarity with R.

I will be following two different tutorials, where code is provided, but thorough analysis of resulting graphs is not: https://www.bioconductor.org/help/course-materials/2015/LearnBioconductorFeb2015/B02.1.1_RNASeqLab.html (https://www.bioconductor.org/help/course-materials/2015/LearnBioconductorFeb2015/B02.1.1_RNASeqLab.html) and https://bioc.ism.ac.jp/packages/3.8/workflows/vignettes/rnaseqGene/inst/doc/rnaseqGene.html (https://bioc.ism.ac.jp/packages/3.8/workflows/vignettes/rnaseqGene/inst/doc/rnaseqGene.html).

First, we need to load some packages. One of which include “airway”, which is a dataset from an RNA-Seq experiment on four human airway smooth muscle cell lines treated with dexamethasone, a synthetic glucocorticoid steroid with anti-inflammatory effects. Glucocorticoids are used, for example, in asthma patients to prevent or reduce inflammation of the airways. In the experiment, four primary human airway smooth muscle cell lines were treated with 1 micromolar dexamethasone for 18 hours. For each of the four cell lines, we have a treated and an untreated sample. It also contains a small subset of the raw data, namely eight BAM file each with a subset of the reads.

The reference for the experiment is: Himes BE, Jiang X, Wagner P, Hu R, Wang Q, Klanderman B, Whitaker RM, Duan Q, Lasky-Su J, Nikolos C, Jester W, Johnson M, Panettieri R Jr, Tantisira KG, Weiss ST, Lu Q. ‘RNA-Seq Transcriptome Profiling Identifies CRISPLD2 as a Glucocorticoid Responsive Gene that Modulates Cytokine Function in Airway Smooth Muscle Cells.’ PLoS One. 2014 Jun 13;9(6):e99625. PMID: 24926665.

Hide

```
library(htmltools)
library("DESeq2")
library(ggplot2)
library("airway")
library("gplots")
library("RColorBrewer")
library("genefilter")
library("pheatmap")
library("PoiClaClu")
```

Then we want to load in our data.

Hide

```
data("airway")
se <- airway
```

We need to make sure we have all the necessary information about the samples prior to performing analysis.

Hide

```
colData(se)
```

```
DataFrame with 8 rows and 9 columns
           SampleName     cell      dex    albut        Run avgLength Experiment    S
ample
             <factor> <factor> <factor> <factor>   <factor> <integer>   <factor>  <fa
ctor>
SRR1039508 GSM1275862  N61311     untrt    untrt SRR1039508       126  SRX384345 SRS5
08568
SRR1039509 GSM1275863  N61311     trt      untrt SRR1039509       126  SRX384346 SRS5
08567
SRR1039512 GSM1275866  N052611    untrt    untrt SRR1039512       126  SRX384349 SRS5
08571
SRR1039513 GSM1275867  N052611    trt      untrt SRR1039513        87  SRX384350 SRS5
08572
SRR1039516 GSM1275870  N080611    untrt    untrt SRR1039516       120  SRX384353 SRS5
08575
SRR1039517 GSM1275871  N080611    trt      untrt SRR1039517       126  SRX384354 SRS5
08576
SRR1039520 GSM1275874  N061011    untrt    untrt SRR1039520       101  SRX384357 SRS5
08579
SRR1039521 GSM1275875  N061011    trt      untrt SRR1039521        98  SRX384358 SRS5
08580
              BioSample
               <factor>
SRR1039508 SAMN02422669
SRR1039509 SAMN02422675
SRR1039512 SAMN02422678
SRR1039513 SAMN02422670
SRR1039516 SAMN02422682
SRR1039517 SAMN02422673
SRR1039520 SAMN02422683
SRR1039521 SAMN02422677
```

Here, we can see that this object contains an informative colData, because, well, it was already prepared for us in the package. Information includes sample / phenotypic information for the experiment at this stage.

Since we have our annotated dataset, we can move forward by constructing a DESeqDataSet object from it!

Hide

```
dds <- DESeqDataSet(se, design = ~ cell + dex)
```

Let's take a look at the count matrices, and the reads themselves.

Hide

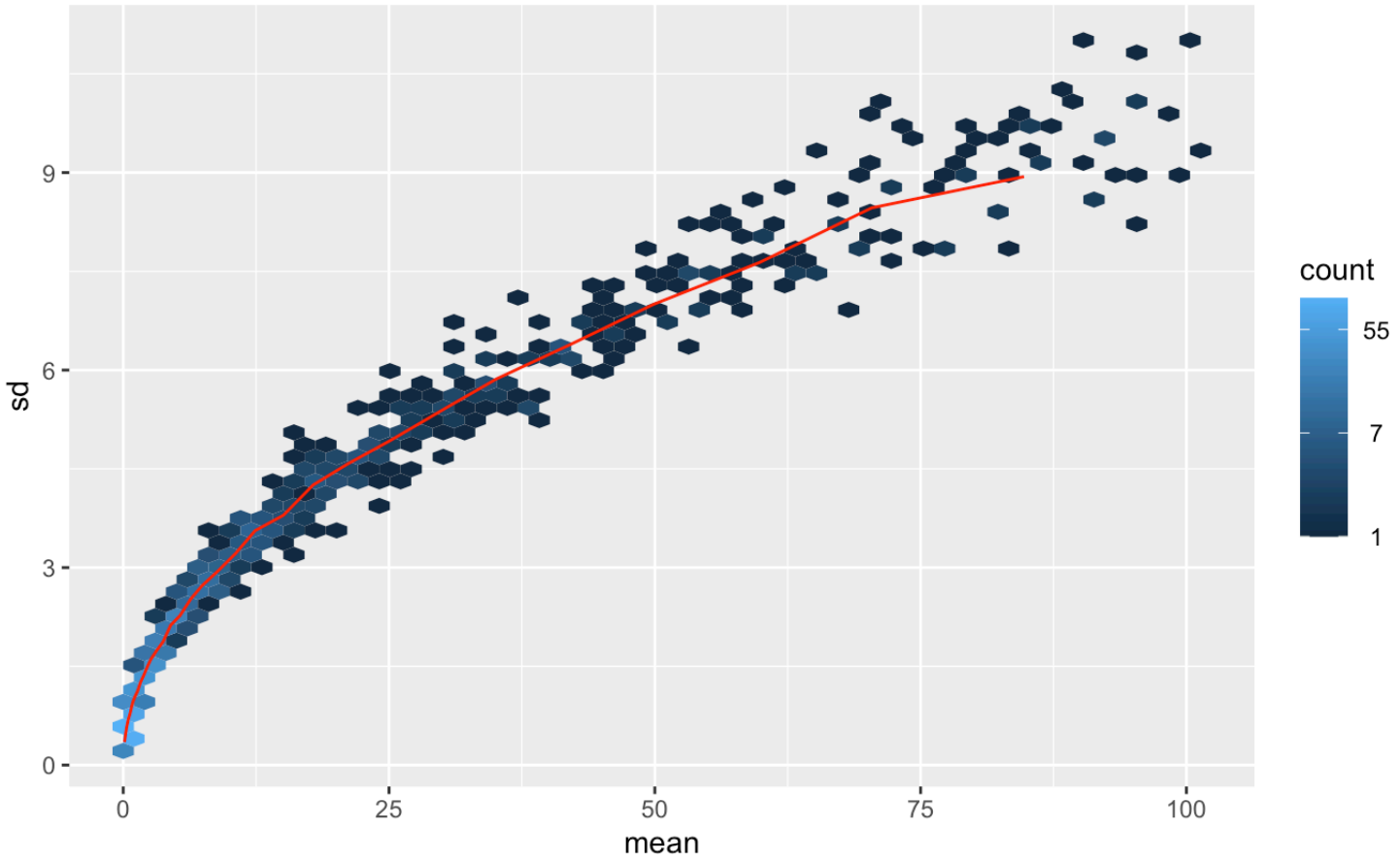
```
countdata <- assay(se)
coldata <- colData(se)
(ddsMat <- DESeqDataSetFromMatrix(countData = countdata,
                                  colData = coldata,
                                  design = ~ cell + dex))
```

```
class: DESeqDataSet
dim: 64102 8
metadata(1): version
assays(1): counts
rownames(64102): ENSG00000000003 ENSG00000000005 ... LRG_98 LRG_99
rowData names(0):
colnames(8): SRR1039508 SRR1039509 ... SRR1039520 SRR1039521
colData names(9): SampleName cell ... Sample BioSample
```

To survey the dataset, we will look at the standard deviation as mean increase.

Hide

```
lambda <- 10^seq(from = -1, to = 2, length = 1000)
cts <- matrix(rpois(1000*100, lambda), ncol = 100)
meanSdPlot(cts, ranks = FALSE)
```



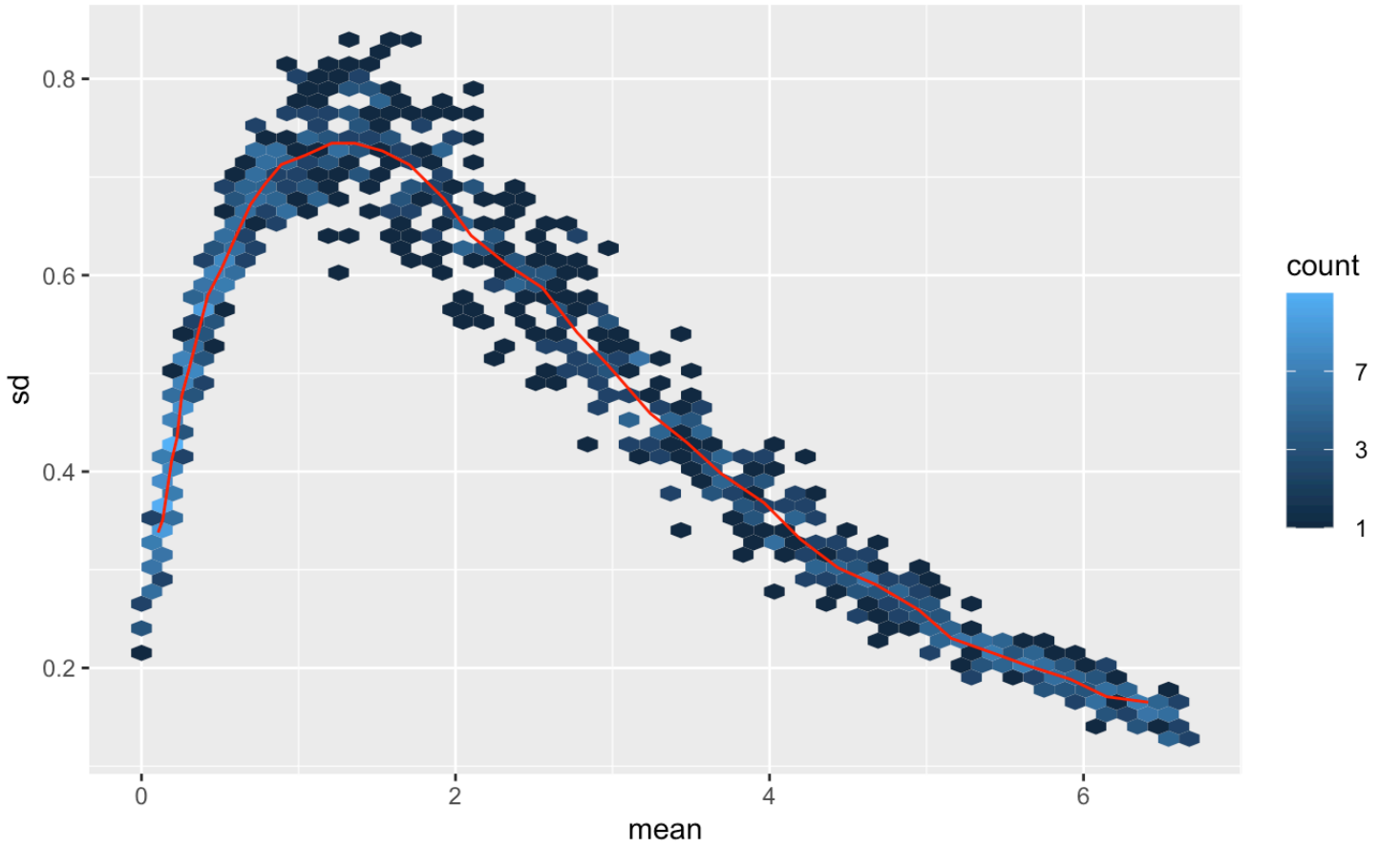
From the graph above, we can see that in our RNA-seq dataset, the variance will grow with the mean.

This type of data is not good to perform analysis on! For instance, if you perform PCA on the data as it is (raw or normalized), the resulting plot typically depends mostly on the genes with highest counts because they show the largest absolute differences between samples.

So, we log-transform our counts.

Hide

```
log.cts.one <- log2(cts + 1)
meanSdPlot(log.cts.one, ranks = FALSE)
```



We can see from the graph above, standard deviation no longer increases with the mean! However, differences are amplified when values are close to 0. This is seen where the standard deviation grows very quickly when the mean close to zero, and overcomes a “hill” at around mean ofone. In this case, low count genes with low signal-to-noise ratio will overly contribute to PCA plots.

Luckily DESeq offers the regularized-logarithm transformation or rlog (Love, Huber, and Anders 2014), which can transform our data such that the data has same range of variance at different ranges of the mean values and the variance doesn’t depend on the mean. This allows for computing distances between samples, like in a PCA plot!

Hide

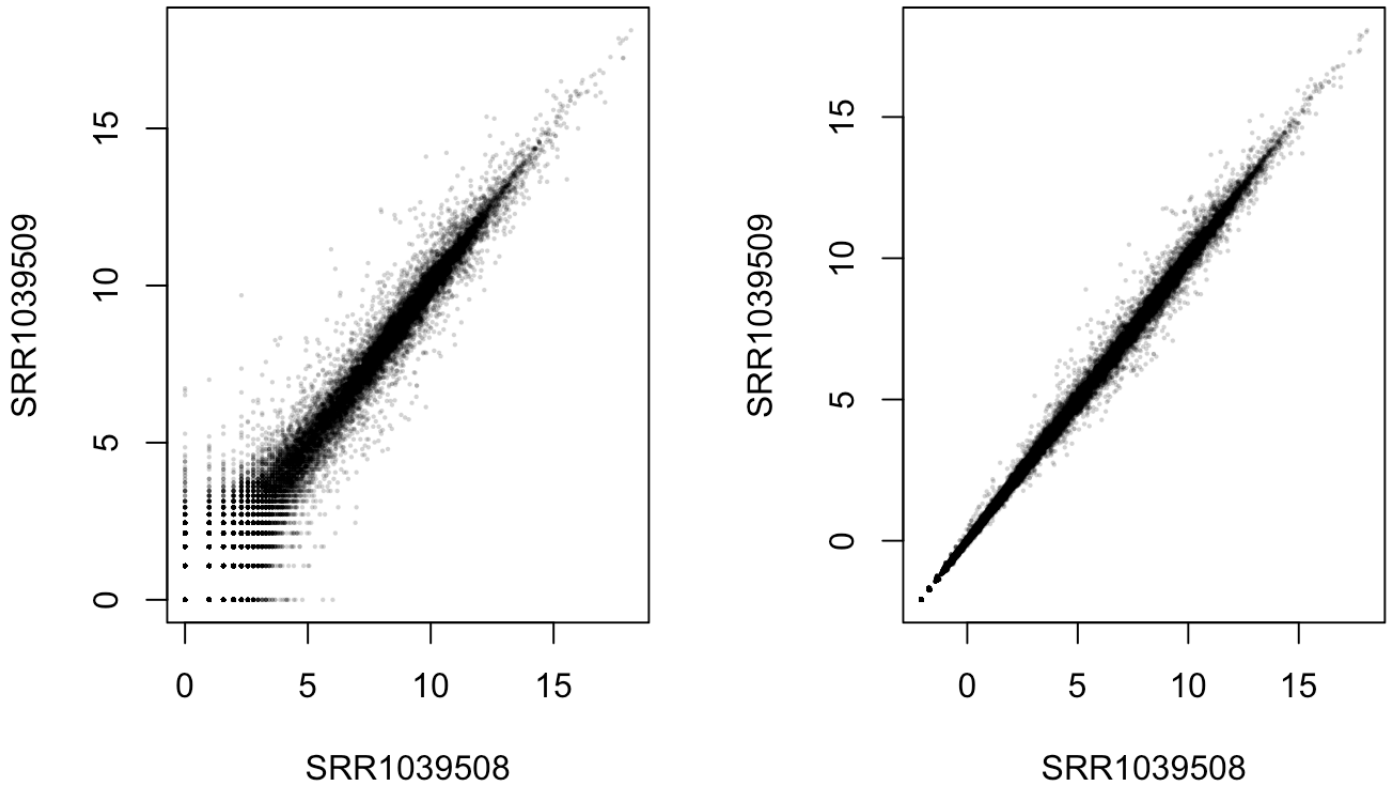
```
rld <- rlog(dds)
head(assay(rld))
```

```
                SRR1039508 SRR1039509 SRR1039512 SRR1039513 SRR1039516 SRR1039517
ENSG00000000003   9.399155   9.142506   9.501691   9.320807   9.757189   9.512179
ENSG00000000005   0.000000   0.000000   0.000000   0.000000   0.000000   0.000000
ENSG00000000419   8.901297   9.113966   9.032566   9.063920   8.981935   9.108522
ENSG00000000457   7.949882   7.882372   7.834287   7.916451   7.773848   7.886645
ENSG00000000460   5.849509   5.882324   5.487269   5.770392   5.940321   5.664001
ENSG00000000938  -1.369632  -1.367411  -1.305192  -1.362554  -1.338570  -1.373842
                SRR1039520 SRR1039521
ENSG00000000003   9.617365   9.315321
ENSG00000000005   0.000000   0.000000
ENSG00000000419   8.894845   9.052299
ENSG00000000457   7.946396   7.908333
ENSG00000000460   6.107519   5.907764
ENSG00000000938  -1.367863  -1.368288
```

For genes with high counts, the rlog transformation differs not much from an ordinary log2 transformation. For genes with lower counts, however, the values are shrunken towards the genes' averages across all samples.

Hide

```
par( mfrow = c( 1, 2 ) )
dds <- estimateSizeFactors(dds)
plot( log2( 1 + counts(dds, normalized=TRUE)[ , 1:2] ),
     col=rgb(0,0,0,.2), pch=16, cex=0.3 )
plot( assay(rld)[ , 1:2],
     col=rgb(0,0,0,.2), pch=16, cex=0.3 )
```



We can see above on the left (ordinary log scale), that genes with low read counts have a high variability. You can see on the left (rlog scale), this variance is compressed after rlog transformation as this data would not provide good information anyway.

We will begin our exploratory analysis by assessing similarity between two samples. Here, we compute the Poisson distance between samples and plot it in a heatmap to visualize their overall similarity. The Poisson distance takes the original count matrix, not normalized.

Hide

```
poisd <- PoissonDistance(t(counts(dds)))
```

From the heatmap above, we can see that the untreated samples are more close to one another and the same goes for the treated samples.

We can also visualize distances using a principal component analysis plot.

Hide

```
(data <- plotPCA(rld, intgroup = c( "dex", "cell"), returnData=TRUE))
```

	PC1	PC2	group	dex	cell	name
	<dbl>	<dbl>	<fctr>	<fctr>	<fctr>	<chr>

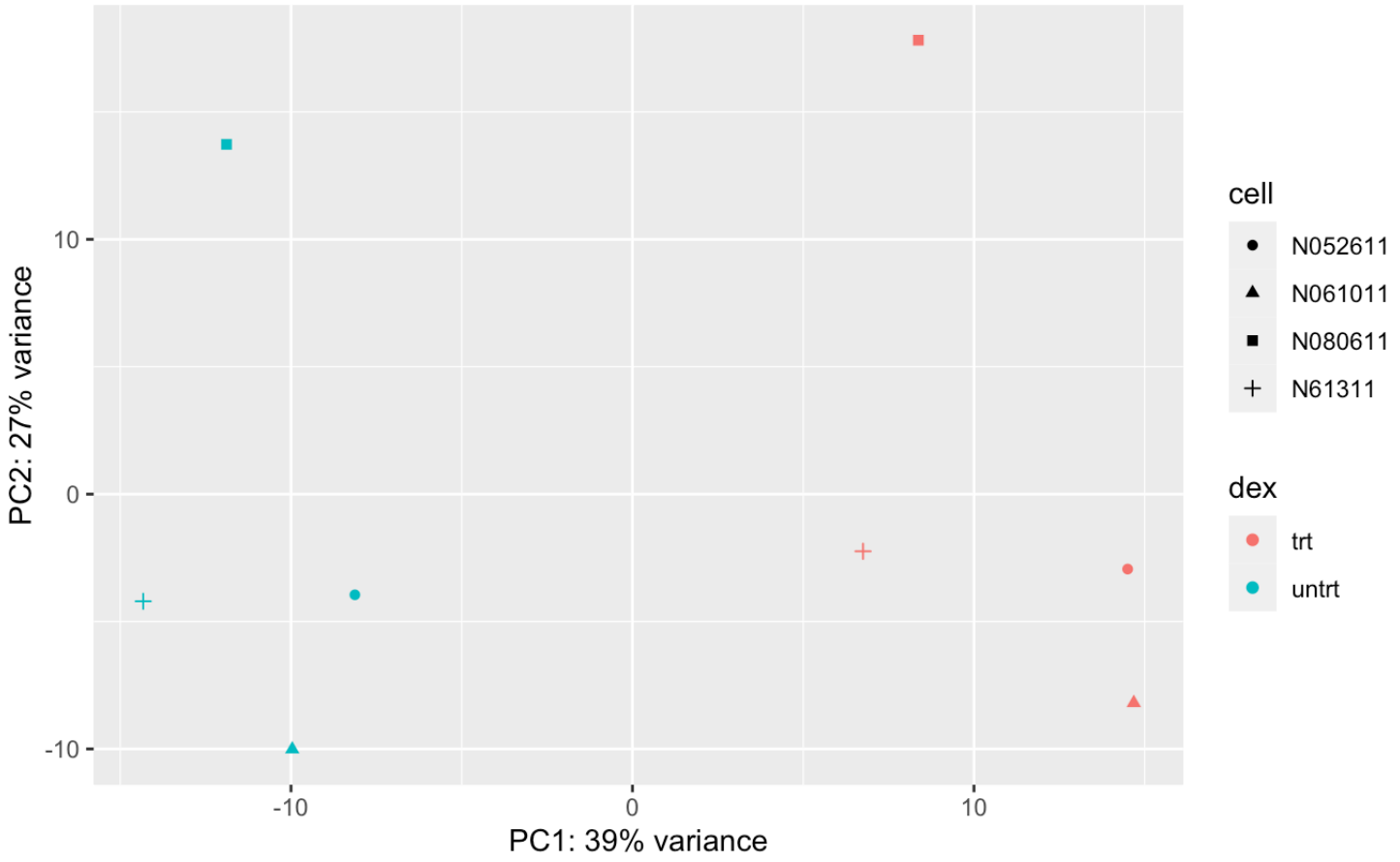
SRR1039508	-14.326791	-4.205271	untrt:N61311	untrt	N61311	SRR1039508
SRR1039509	6.752236	-2.242264	trt:N61311	trt	N61311	SRR1039509
SRR1039512	-8.128701	-3.950350	untrt:N052611	untrt	N052611	SRR1039512
SRR1039513	14.500816	-2.940329	trt:N052611	trt	N052611	SRR1039513
SRR1039516	-11.887297	13.728272	untrt:N080611	untrt	N080611	SRR1039516
SRR1039517	8.371955	17.815856	trt:N080611	trt	N080611	SRR1039517
SRR1039520	-9.962960	-10.012013	untrt:N061011	untrt	N061011	SRR1039520
SRR1039521	14.680742	-8.193902	trt:N061011	trt	N061011	SRR1039521

8 rows

```
Warning messages:
1: In readChar(file, size, TRUE) : truncating string with embedded nuls
2: In readChar(file, size, TRUE) : truncating string with embedded nuls
3: In readChar(file, size, TRUE) : truncating string with embedded nuls
4: In readChar(file, size, TRUE) : truncating string with embedded nuls
5: In readChar(file, size, TRUE) : truncating string with embedded nuls
6: In readChar(file, size, TRUE) : truncating string with embedded nuls
```

Hide

```
percentVar <- round(100 * attr(data, "percentVar"))
qplot(PC1, PC2, color=dex, shape=cell, data=data) +
  xlab(paste0("PC1: ",percentVar[1],"% variance")) +
  ylab(paste0("PC2: ",percentVar[2],"% variance"))
```



From above, we can see that the untreated samples are closer to one another, and the treated samples are closer to one another, regardless of cell type. While the distances between different cell types are considerable, differences due to the treatment are stronger. This indicates that dexamethasone has a significant effect. It is important to note that cell types also do locate similarly relative to other cell types, although it is not as of a strong separation. But it's also interesting to note that these two PCs only capture a little over 60% of the variance of the original dataset.

Let's take a look at differential expression analysis to see what genes are causing these considerable distances.

Hide

```
dds$dex <- relevel(dds$dex, "untrt")
dds <- DESeq(dds)
```

```
using pre-existing size factors
estimating dispersions
gene-wise dispersion estimates
mean-dispersion relationship
final dispersion estimates
fitting model and testing
```

Hide

```
(res <- results(dds))
```

```
log2 fold change (MLE): dex trt vs untrt
Wald test p-value: dex trt vs untrt
DataFrame with 64102 rows and 6 columns
                 baseMean log2FoldChange     lfcSE      stat      pvalue       padj
                <numeric>      <numeric> <numeric> <numeric>   <numeric>  <numeric>
ENSG00000000003  708.6022     -0.3812539  0.100654 -3.787751 0.000152017 0.00128364
ENSG00000000005    0.0000             NA        NA        NA          NA         NA
ENSG00000000419  520.2979      0.2068127  0.112219  1.842944 0.065337210 0.19654584
ENSG00000000457  237.1630      0.0379206  0.143445  0.264357 0.791504963 0.91145800
ENSG00000000460   57.9326     -0.0881677  0.287142 -0.307053 0.758803336 0.89503445
...                    ...            ...       ...       ...         ...        ...
LRG_94                   0             NA        NA        NA          NA         NA
LRG_96                   0             NA        NA        NA          NA         NA
LRG_97                   0             NA        NA        NA          NA         NA
LRG_98                   0             NA        NA        NA          NA         NA
LRG_99                   0             NA        NA        NA          NA         NA
```

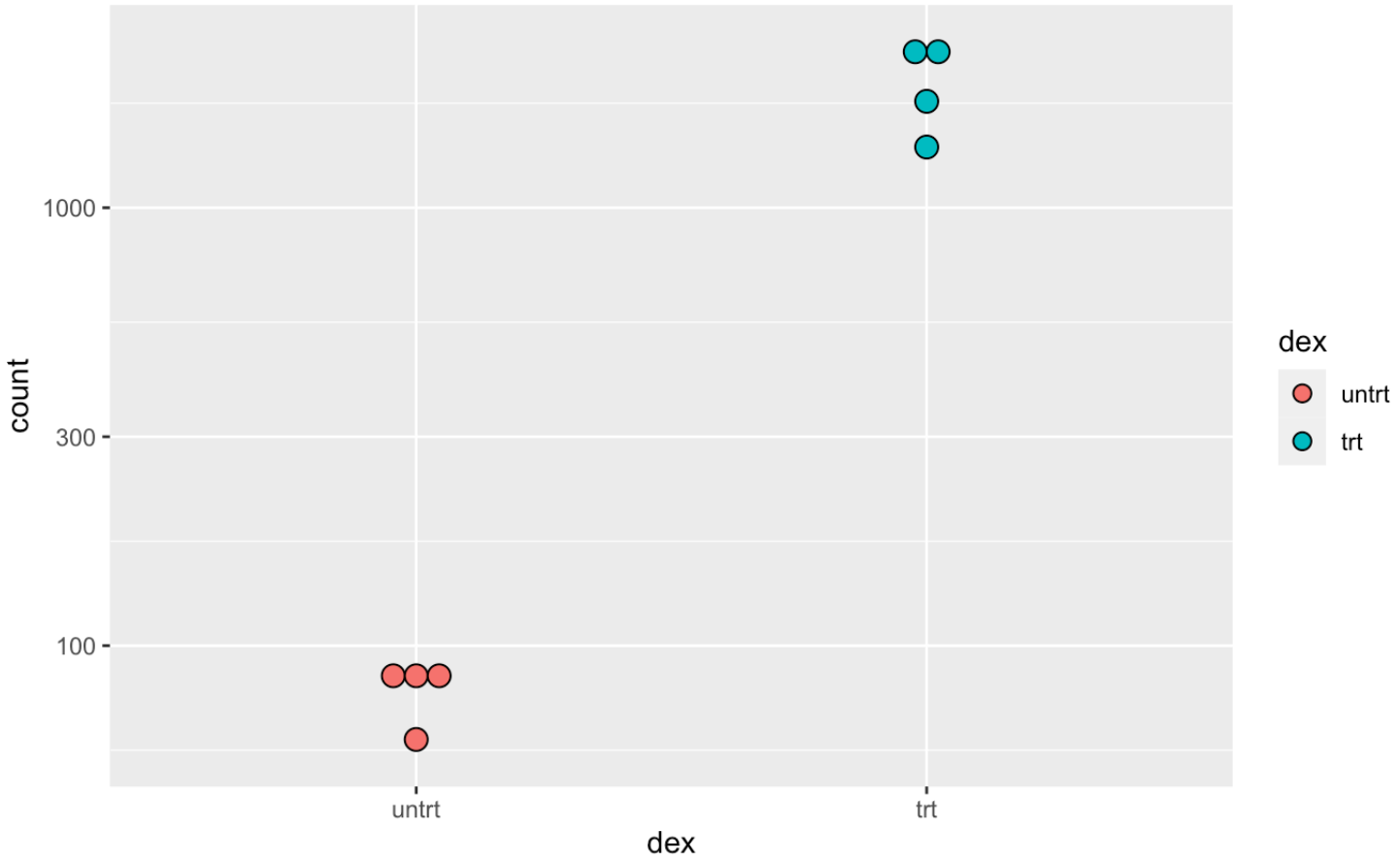
Hide

```
mcols(res, use.names=TRUE)
```

```
DataFrame with 6 rows and 2 columns
                       type                                 description
                <character>                                 <character>
baseMean       intermediate mean of normalized counts for all samples
log2FoldChange      results  log2 fold change (MLE): dex trt vs untrt
lfcSE               results          standard error: dex trt vs untrt
stat                results          Wald statistic: dex trt vs untrt
pvalue              results       Wald test p-value: dex trt vs untrt
padj                results                       BH adjusted p-values
```

Hide

```
topGene <- rownames(res)[which.min(res$padj)]
data <- plotCounts(dds, gene=topGene, intgroup=c("dex","cell"), returnData=TRUE)
ggplot(data, aes(x=dex, y=count, fill=dex)) +
  scale_y_log10() +
  geom_dotplot(binaxis="y", stackdir="center")
```

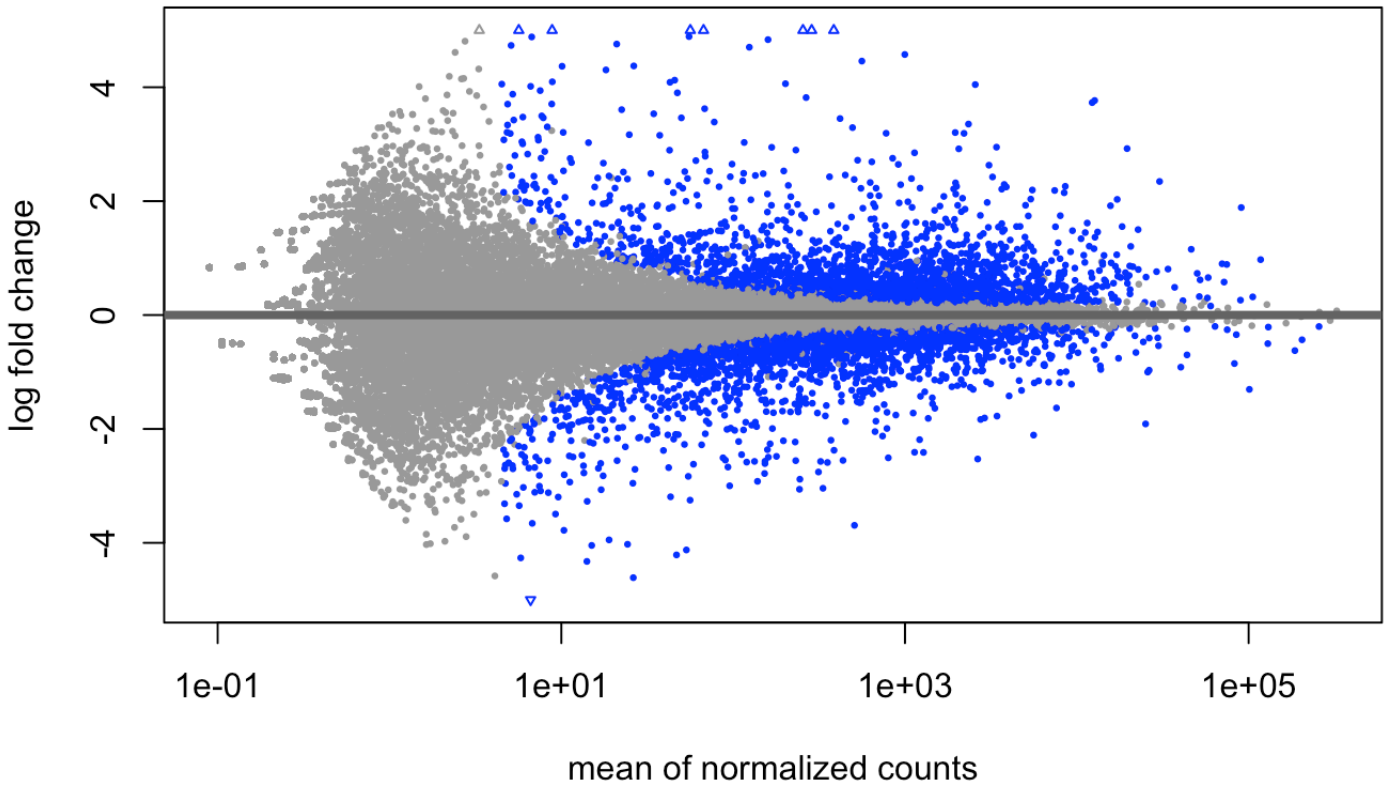


As you can see from the graph above, the untreated samples have much lower count values than the treated samples. This suggests that genes in treated cells have significant upregulation, while those in untreated cells have significant downregulation. It's extremely important to note that these are relative to one another.

We can also take a look at an MA plot, where each dot represents a gene. On the y-axis, the "M" stands for "minus" – subtraction of log values is equivalent to the log of the ratio – and on the x-axis, the "A" stands for "average". The vertical dashed line indicates a mean or normalized counts threshold.

Hide

```
plotMA(res, ylim=c(-5,5))
```

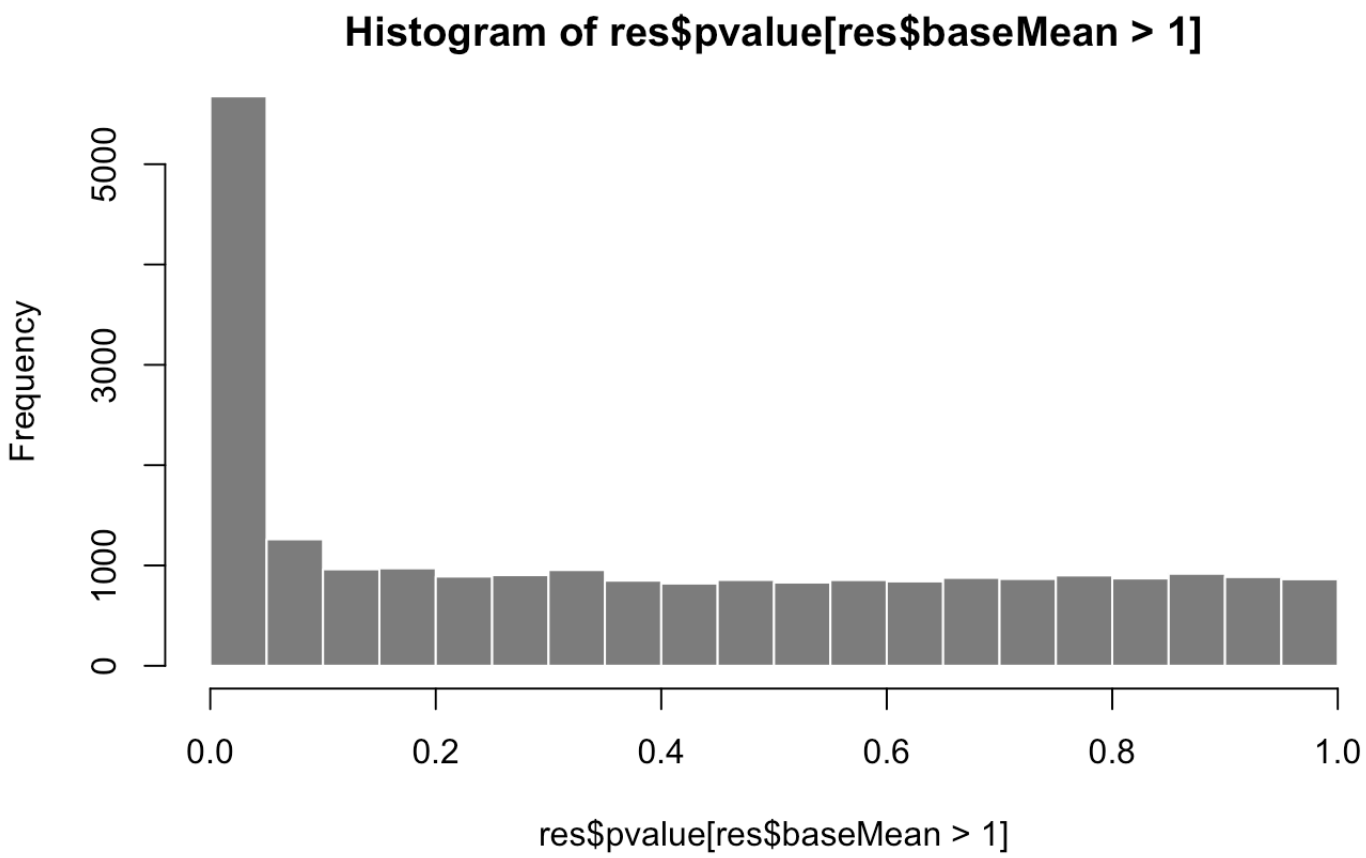


From the MA plot, we can see the blue colored genes represent genes that are significantly differentially expressed between the two samples. Here, the dots in blue represent genes that are either significantly highly or lowly expressed. Since log2 fold change is 'dex trt vs untrt', we can see that the directionality in which genes are expressed of samples treated with dex relative to those that are not treated. In a more explicit manner, log2 fold change is dex treated divided by dex untreated (see table two code blocks earlier). So compared to the untreated samples, the blue dots above the log fold change 0 are genes that are overexpressed in treated samples, and the blue dots under the log fold change 0 are genes that are underexpressed in treated samples. Also, most of these genes are objectively highly expressed as seen by their location on the M axis, seeing that they tend to take the right side of the graph.

In addition, we can graph histogram of the p values, for genes with mean normalized count larger than 1.

Hide

```
hist(res$pvalue[res$baseMean > 1], breaks = 0:20/20,
     col = "grey50", border = "white")
```

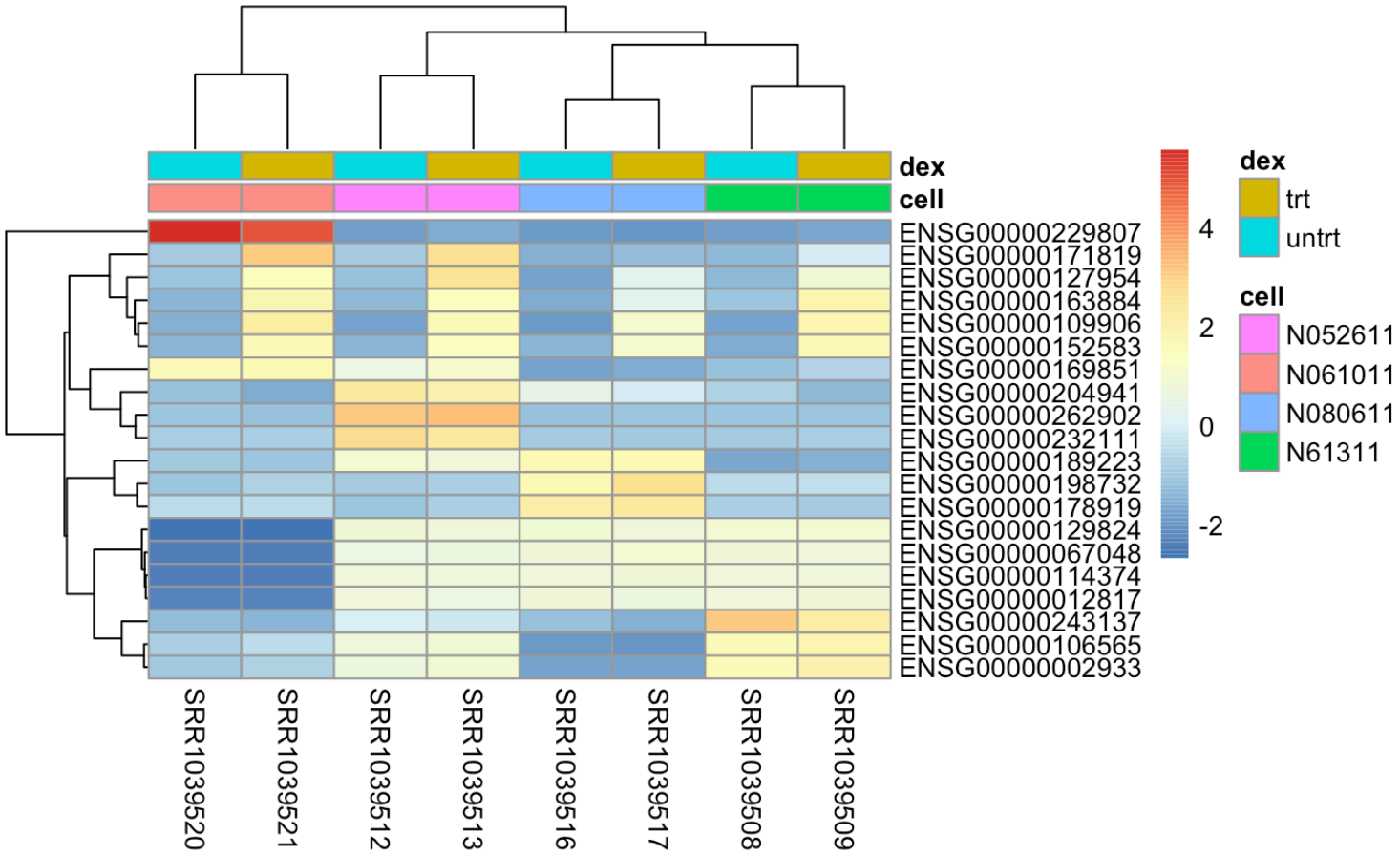


As you can see from the histogram above, there are quite a number of genes that fall in p <= 0.05.

We can also cluster genes. It's of great interest to only cluster genes that are highly variable as these genes are the ones that actually carry a signal. Here we look at the top 20. Instead of looking at absolute expression strength, we look at the amount by which each gene deviates in a specific sample from the gene's average across all samples.

Hide

```
topVarGenes <- head(order(rowVars(assay(rld)), decreasing = TRUE), 20)
mat  <- assay(rld)[ topVarGenes, ]
mat  <- mat - rowMeans(mat)
anno <- as.data.frame(colData(rld)[, c("cell","dex")])
pheatmap(mat, annotation_col = anno)
```



As seen in the heatmap above, there are bars that indicate treatment conditions and cell type conditions. It's interesting to note that the first horizontal bar, you can see that the N061011 cell type is seperated from all other cell types in the fact that gene expression is higher than average compared to the rest of the cell types. In the 8th horizontal bar onwards, gene expression is lower than average compared to the rest of the cell types. In addition, dex-untreated samples tend to have lower expression than average (blue squares), than treated samples.

Cool. Well, that's all I have. Thanks Derek and Professor Lee for a great quarter! :)