initialAnalysis

Protocol:

1. Use salmon to quantify transcripts

2. Use tximeta to get count matrix

3. use with DEseq for expression

import data table & create table with file paths to sample quant files

```
## importing quantifications
```

```
## reading in files with read_tsv
```

```
## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
## found matching transcriptome:
## [ Ensembl - Mus musculus - release 99 ]
## loading existing EnsDb created: 2022-04-14 17:58:52
## loading existing transcript ranges created: 2022-04-14 17:58:54
## loading existing EnsDb created: 2022-04-14 17:58:52
## obtaining transcript-to-gene mapping from database
## loading existing gene ranges created: 2022-04-14 18:01:41
## summarizing abundance
## summarizing counts
## summarizing length
```

```
## using counts and average transcript lengths from tximeta
```

```
## Warning in DESeqDataSet(gse, design = ~SampleGenotype + SampleTime): some
## variables in design formula are characters, converting to factors
```

```
dds = DESeq(dds)
```

```
## estimating size factors
```

```
## using 'avgTxLength' from assays(dds), correcting for library size
```

```
## estimating dispersions
```

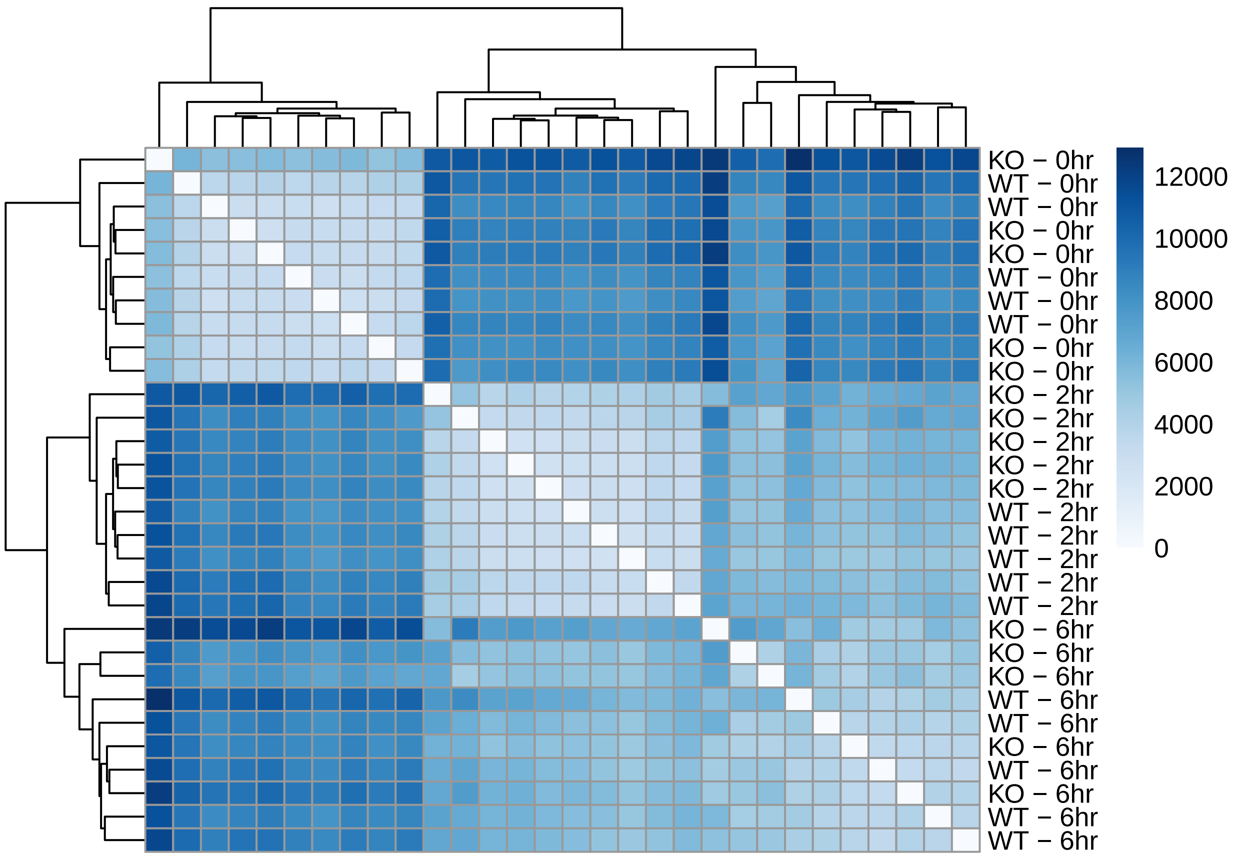
```
## gene-wise dispersion estimates
```

```
## mean-dispersion relationship
```

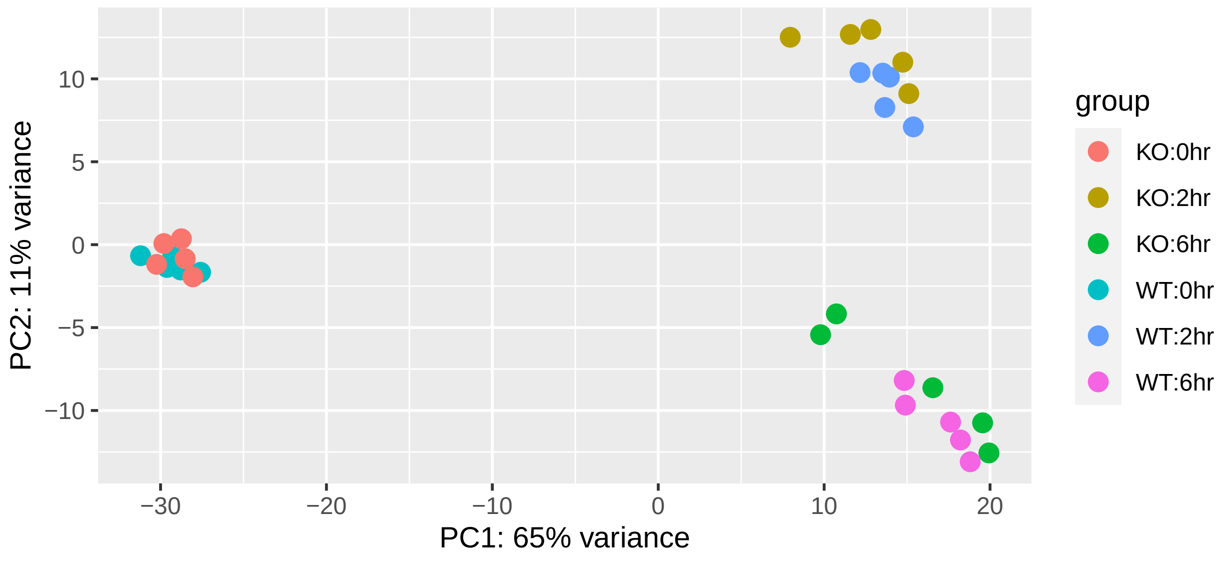
```
## final dispersion estimates
```

```
## fitting model and testing
```

```
#log2 fold changes and pvalues for WT vs KO
dds.res = results(dds, contrast = c("SampleGenotype", "WT", "KO"))
```



```r
library(vsn)

#variance stabilizing transformation
vsd = vst(dds)
plotPCA(vsd, intgroup = c("SampleGenotype", "SampleTime"))
```



```r
library(genefilter)
```

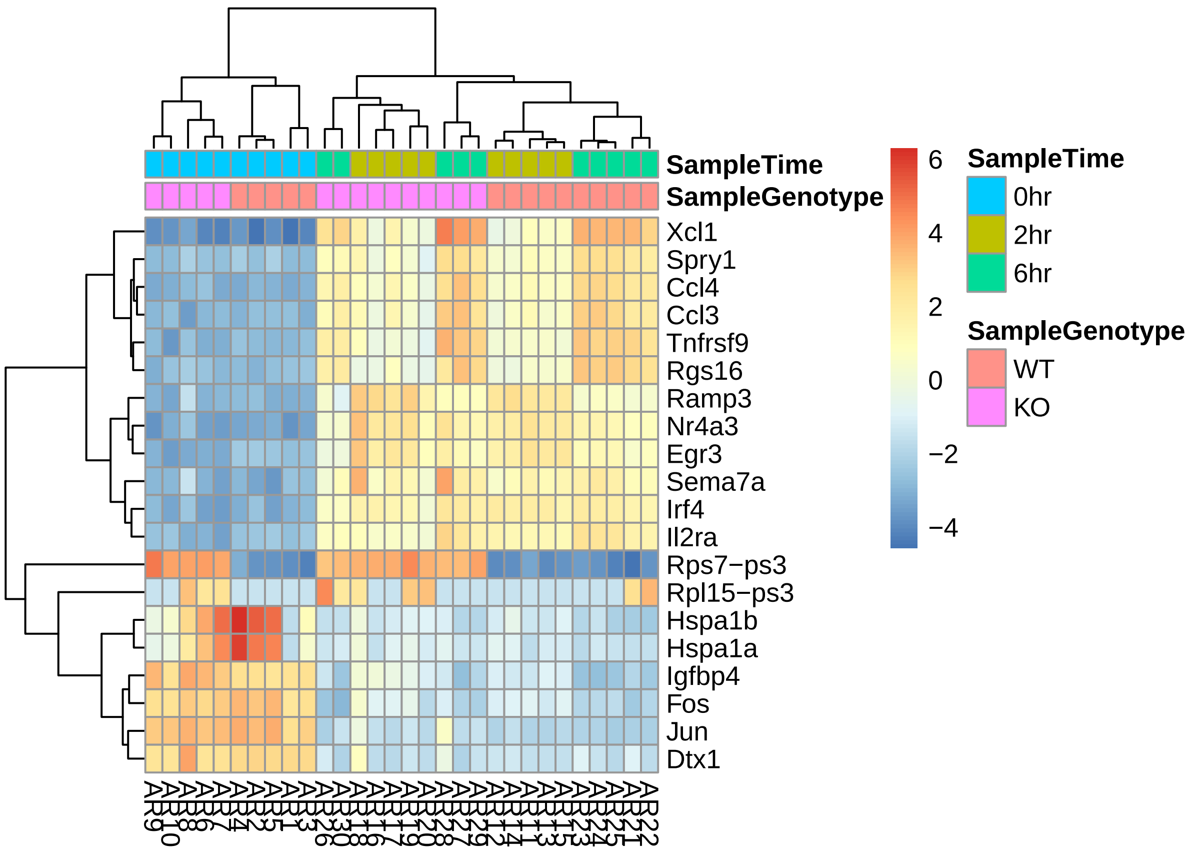
```
##
```

```
## Attaching package: 'genefilter'

## The following objects are masked from 'package:MatrixGenerics':
##
##     rowSds, rowVars

## The following objects are masked from 'package:matrixStats':
##
##     rowSds, rowVars
```

```
topVarGenes = head(order(rowVars(assay(vsd)), decreasing = T), 20)
mat = assay(vsd)[ topVarGenes, ]

rownames(mat) <- mapIds(org.Mm.eg.db,
                    keys = rownames(mat),
                    column = "SYMBOL",
                    keytype = "ENSEMBL",
                    multiVals = "first")
```

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

```
mat = mat - rowMeans(mat)
anno = as.data.frame(colData(vsd)[, c("SampleGenotype", "SampleTime")])

pheatmap::pheatmap(mat, annotation_col = anno)
```



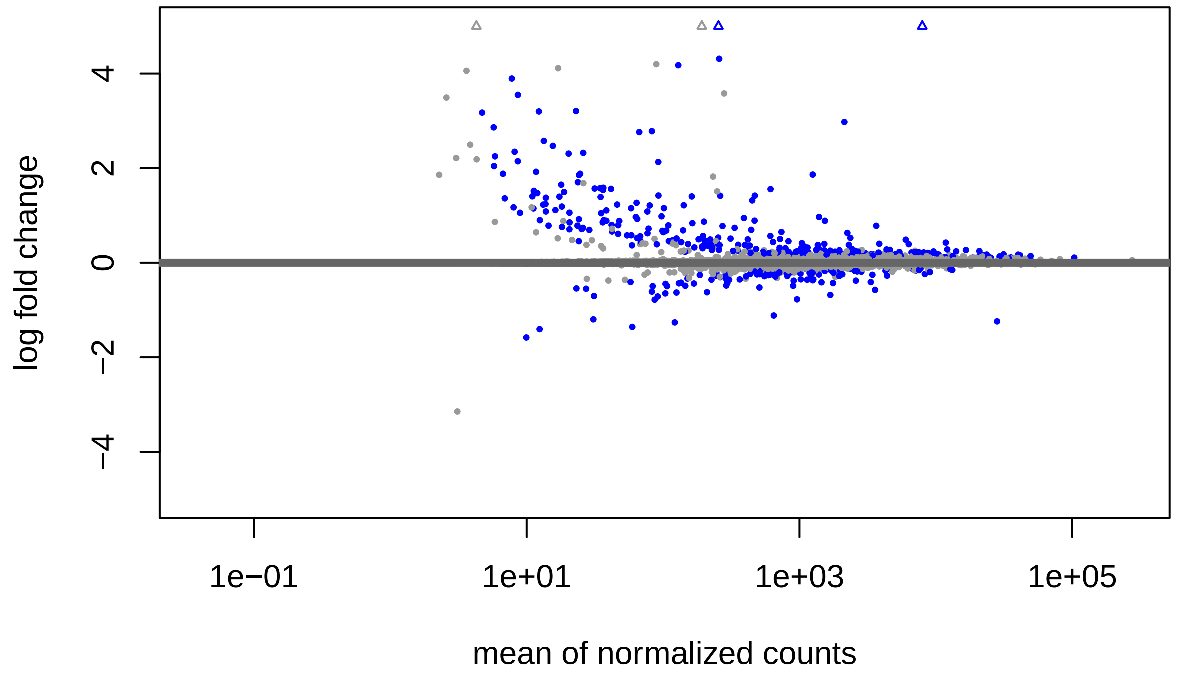
```
library(apeglm)

resultsNames(dds)
```

```
## [1] "Intercept"                 "SampleGenotype_KO_vs_WT"
## [3] "SampleTime_2hr_vs_0hr"    "SampleTime_6hr_vs_0hr"
```

```
#calculate lfc shrink for 0hr
res = lfcShrink(dds, coef = "SampleGenotype_KO_vs_WT", type = "apeglm")
```

```
## using 'apeglm' for LFC shrinkage. If used in published research, please cite:
##     Zhu, A., Ibrahim, J.G., Love, M.I. (2018) Heavy-tailed prior distributions for
##     sequence count data: removing the noise and preserving large differences.
##     Bioinformatics. https://doi.org/10.1093/bioinformatics/bty895
```

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



```
ddsTC = DESeqDataSet(gse, design = ~ SampleGenotype + SampleTime + SampleGenotype:SampleTime)
```

```
## using counts and average transcript lengths from tximeta

## Warning in DESeqDataSet(gse, design = ~SampleGenotype + SampleTime +
## SampleGenotype:SampleTime): some variables in design formula are characters,
## converting to factors
```

```
#likelyhood ratio test
ddsTC = DESeq(ddsTC, test="LRT", reduced = ~ SampleGenotype + SampleTime)
```

```
## estimating size factors

## using 'avgTxLength' from assays(dds), correcting for library size

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing
```

```r
resTC = results(ddsTC)
resTC$symbol = mcols(ddsTC)$symbol
```