# ICGEB Workshop

# Introduction to Bulk RNA-sequencing with R and Bioconductor *

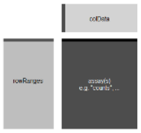
Saroj Kant Mohapatra

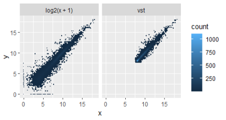
National Institute of Biomedical Genomics

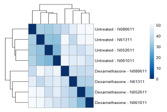
07-Aug-2024

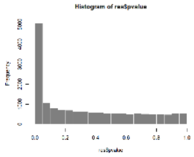
** This tutorial is based on the rnaseqGene workflow. Please follow this link to learn more about the analysis. https://www.bioconductor.org/packages/release/workflows/vignettes/rnaseqGene/inst/doc/rnaseqGene.html*

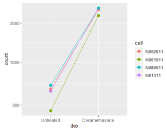
## Contents

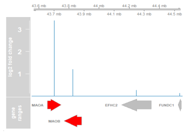






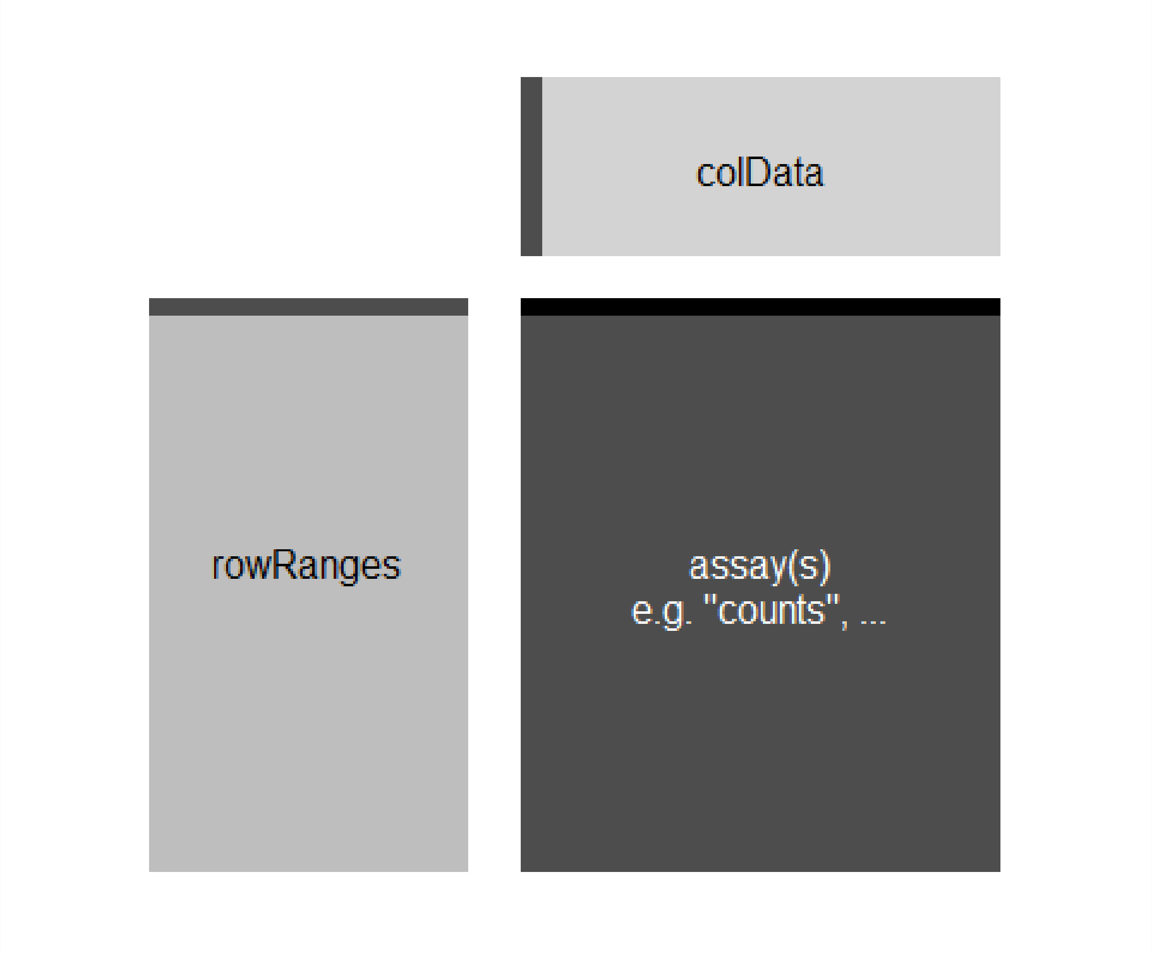






```r
# Set the working directory using setwd()
# Remove any objects from the workspace at the beginning.
rm(list=ls())

######################################################################
## 01. Preliminaries
######################################################################
library("dplyr")
library("airway")
library("ggplot2")
library("DESeq2")
library("vsn")
library("pheatmap")
library("RColorBrewer")
library("ggbeeswarm")
library("apeglm")
library("genefilter")
library("AnnotationDbi")
library("org.Hs.eg.db")
library("Gviz")
```

```
######################################################################
## Understand the structure of SummarizedExperiment object
######################################################################
```



```
# The component parts of a SummarizedExperiment object.
# The assay contains the matrix of counts, rowRanges contains information
# about the genomic ranges, colData contains information about the samples.
# Note that the first row of colData lines up with the first column of
# the assay. Further discussion in Huber et al 2015 Nature Methods 12: 115
```

```
######################################################################
## 02. Get the SummarizedExperiment object gse
######################################################################
# The data used in this tutorial are derived from the airway package
# that summarizes an RNA-seq experiment wherein airway smooth muscle cells
# were treated with dexamethasone, a synthetic glucocorticoid steroid
# with anti-inflammatory effects (Himes et al. 2014). Glucocorticoids are used,
# for example, by people with asthma to 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. For more description of the experiment see the
# PubMed entry 24926665.

# Load the data
data(gse)
gse

## class: RangedSummarizedExperiment
## dim: 58294 8
## metadata(6): tximetaInfo quantInfo ... txomeInfo txdbInfo
## assays(3): counts abundance length
## rownames(58294): ENSG00000000003.14 ENSG00000000005.5 ...
##   ENSG00000285993.1 ENSG00000285994.1
## rowData names(1): gene_id
## colnames(8): SRR1039508 SRR1039509 ... SRR1039520 SRR1039521
## colData names(3): names donor condition

# Look at the fist three lines of the count data
head(assay(gse), 3)

##                      SRR1039508 SRR1039509 SRR1039512 SRR1039513 SRR1039516
## ENSG00000000003.14    708.164    467.962    900.992    424.368   1188.295
## ENSG00000000005.5       0.000      0.000      0.000      0.000      0.000
## ENSG00000000419.12    455.000    510.000    604.000    352.000    583.000
##                      SRR1039517 SRR1039520 SRR1039521
## ENSG00000000003.14   1090.668    805.929    599.337
## ENSG00000000005.5       0.000      0.000      0.000
## ENSG00000000419.12    773.999    409.999    499.000

# Add all count data for each sample
colSums(assay(gse))

## SRR1039508 SRR1039509 SRR1039512 SRR1039513 SRR1039516 SRR1039517 SRR1039520
##   21100805   19298584   26145537   15688246   25268618   31891456   19683767
## SRR1039521
##   21813903

# Look at some meta data in the object
rowRanges(gse)
```

```
## GRanges object with 58294 ranges and 1 metadata column:
##                       seqnames              ranges strand |            gene_id
##                          <Rle>           <IRanges>  <Rle> |        <character>
##   ENSG00000000003.14     chrX 100627109-100639991      - | ENSG00000000003.14
##    ENSG00000000005.5     chrX 100584802-100599885      + |  ENSG00000000005.5
##   ENSG00000000419.12    chr20   50934867-50958555      - | ENSG00000000419.12
##   ENSG00000000457.13     chr1 169849631-169894267      - | ENSG00000000457.13
##   ENSG00000000460.16     chr1 169662007-169854080      + | ENSG00000000460.16
##                  ...      ...                 ...    ... .                ...
##    ENSG00000285990.1    chr14   19244904-19269380      - |  ENSG00000285990.1
##    ENSG00000285991.1     chr6 149817937-149896011      - |  ENSG00000285991.1
##    ENSG00000285992.1     chr8   47129262-47132628      + |  ENSG00000285992.1
##    ENSG00000285993.1    chr18   46409197-46410645      - |  ENSG00000285993.1
##    ENSG00000285994.1    chr10   12563151-12567351      + |  ENSG00000285994.1
##   -------
##   seqinfo: 25 sequences (1 circular) from hg38 genome
```

```
seqinfo(rowRanges(gse))
```

```
## Seqinfo object with 25 sequences (1 circular) from hg38 genome:
##   seqnames seqlengths isCircular genome
##   chr1      248956422      FALSE   hg38
##   chr2      242193529      FALSE   hg38
##   chr3      198295559      FALSE   hg38
##   chr4      190214555      FALSE   hg38
##   chr5      181538259      FALSE   hg38
##   ...             ...        ...    ...
##   chr21      46709983      FALSE   hg38
##   chr22      50818468      FALSE   hg38
##   chrX      156040895      FALSE   hg38
##   chrY       57227415      FALSE   hg38
##   chrM          16569       TRUE   hg38
```

```
colData(gse)
```

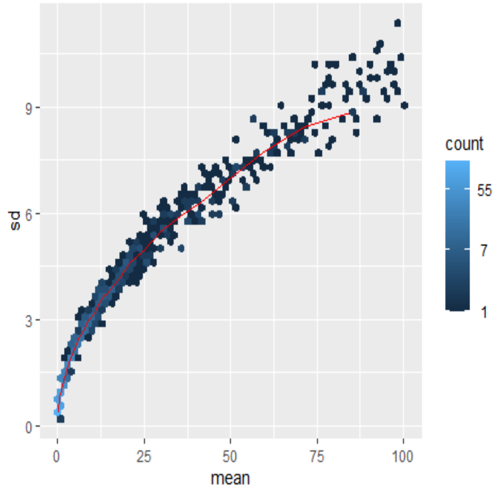
```
## DataFrame with 8 rows and 3 columns
##                 names    donor     condition
##              <factor> <factor>      <factor>
## SRR1039508 SRR1039508  N61311  Untreated
## SRR1039509 SRR1039509  N61311  Dexamethasone
## SRR1039512 SRR1039512  N052611 Untreated
## SRR1039513 SRR1039513  N052611 Dexamethasone
## SRR1039516 SRR1039516  N080611 Untreated
## SRR1039517 SRR1039517  N080611 Dexamethasone
## SRR1039520 SRR1039520  N061011 Untreated
## SRR1039521 SRR1039521  N061011 Dexamethasone
```

```
# List directly the donor subjects and the treatment conditions
gse$donor
```

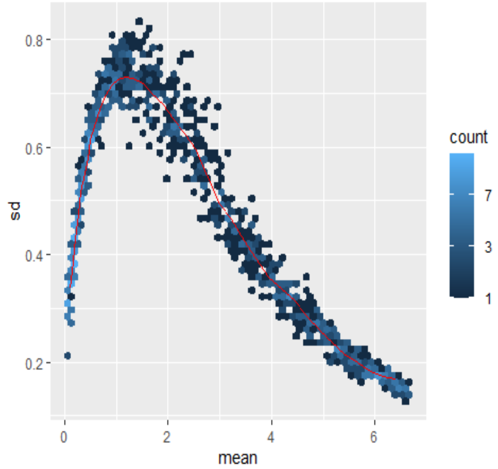
```
## [1] N61311  N61311  N052611 N052611 N080611 N080611 N061011 N061011
## Levels: N052611 N061011 N080611 N61311

gse$condition

## [1] Untreated     Dexamethasone Untreated     Dexamethasone Untreated
## [6] Dexamethasone Untreated     Dexamethasone
## Levels: Untreated Dexamethasone

# Let us use cell to denote the donor cell line,
# and dex to denote the treatment condition.
gse$cell <- gse$donor
gse$dex <- gse$condition


# quickly check the millions of fragments that could be mapped to the genes
# (the second argument of round tells how many decimal points to keep).
round( colSums(assay(gse)) / 1e6, 1 )

## SRR1039508 SRR1039509 SRR1039512 SRR1039513 SRR1039516 SRR1039517 SRR1039520
##       21.1       19.3       26.1       15.7       25.3       31.9       19.7
## SRR1039521
##       21.8

# Question 1: How many rows in the object gse? How many columns?
# Question 2: What is the total number of donors for this data set?
```

```
######################################################################
## 03. Convert SummarizedExperiment object gse to DESeqDataSet object dds
## and pre-filtering
######################################################################
# Pre-filtering the dataset: The DESeqDataSet dds contains many rows
# with only zeros, and additionally many rows with only a few fragments total
# In order to reduce the size of the object, and to increase the speed of our
# functions, we can remove the rows that have no or nearly no information
# about the amount of gene expression. Keep only rows that have a count of
# at least 10 for at least 4 of samples. The count of 10 is a reasonable
# choice for bulk RNA-seq.
dds <- DESeqDataSet(gse, design = ~ cell + dex)
smallestGroupSize <- 4
keep <- rowSums(counts(dds) >= 10) >= smallestGroupSize
dds <- dds[keep,]

# Clean-up
rm(keep, smallestGroupSize)

# Question 3: Compare the number of rows between gse and dds. Use nrow()
```

```
##################################################################
## 04. Variance stabilizing transformation
##################################################################
# For RNA-seq counts, the expected variance grows with the mean.
# This can be a problem, for example, while doing PCA directly on the
# matrix of counts. The resulting plot typically depends mostly on the
# genes with highest counts because they show the largest absolute
# differences between samples. See the plots below.
lambda <- 10^seq(from = -1, to = 2, length = 1000)
cts <- matrix(rpois(1000*100, lambda), ncol = 100)
meanSdPlot(cts, ranks = FALSE)
```



```
# The logarithm with a small pseudocount amplifies differences when
# the values are close to 0. The low count genes with low signal-to-noise
# ratio will overly contribute to sample-sample distances and PCA plots.
log.cts.one <- log2(cts + 1)
meanSdPlot(log.cts.one, ranks = FALSE)
```



```
rm(cts, log.cts.one, lambda)
```

```
# Use VST for applications other than differential testing.
# For differential testing, use the DESeq function applied to raw counts,
# as described later.
# Note that blind = FALSE, which means that differences between cell lines
# and treatment (the variables in the design) will not contribute to the
# expected variance-mean trend of the experiment.
# For a fully unsupervised transformation, one can set blind = TRUE (default)
vsd <- vst(dds, blind = FALSE, )
```

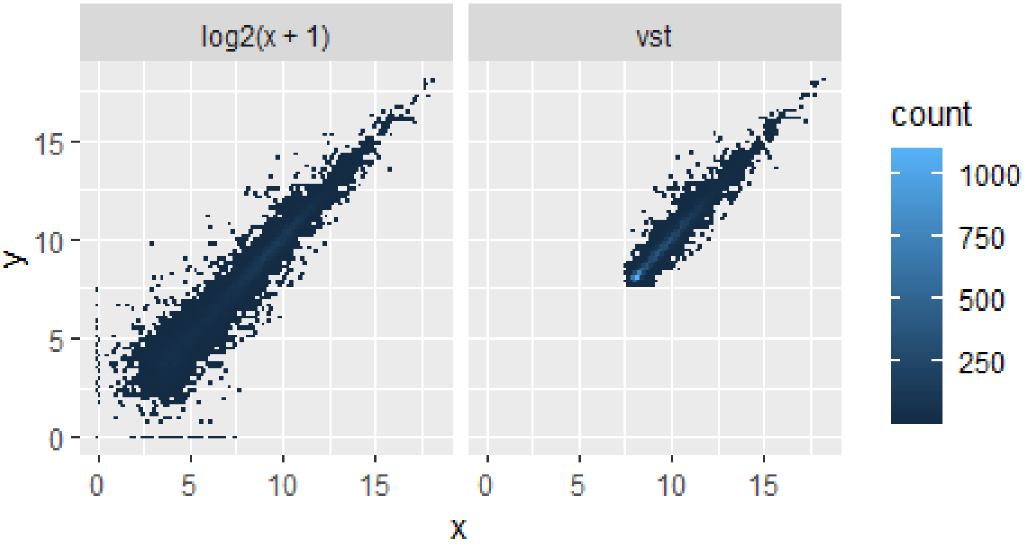
```
head(assay(vsd), 3)
```

```
##                      SRR1039508 SRR1039509 SRR1039512 SRR1039513 SRR1039516
## ENSG00000000003.14  10.082167   9.828058  10.152774   9.970690  10.410407
## ENSG00000000419.12   9.663489   9.900634   9.779942   9.775148   9.740478
## ENSG00000000457.13   9.417376   9.280001   9.333639   9.430438   9.250501
##                      SRR1039517 SRR1039520 SRR1039521
## ENSG00000000003.14  10.171994  10.300682   9.976235
## ENSG00000000419.12   9.848458   9.659469   9.817695
## ENSG00000000457.13   9.363013   9.450889   9.444291
```

```
colData(vsd)
```

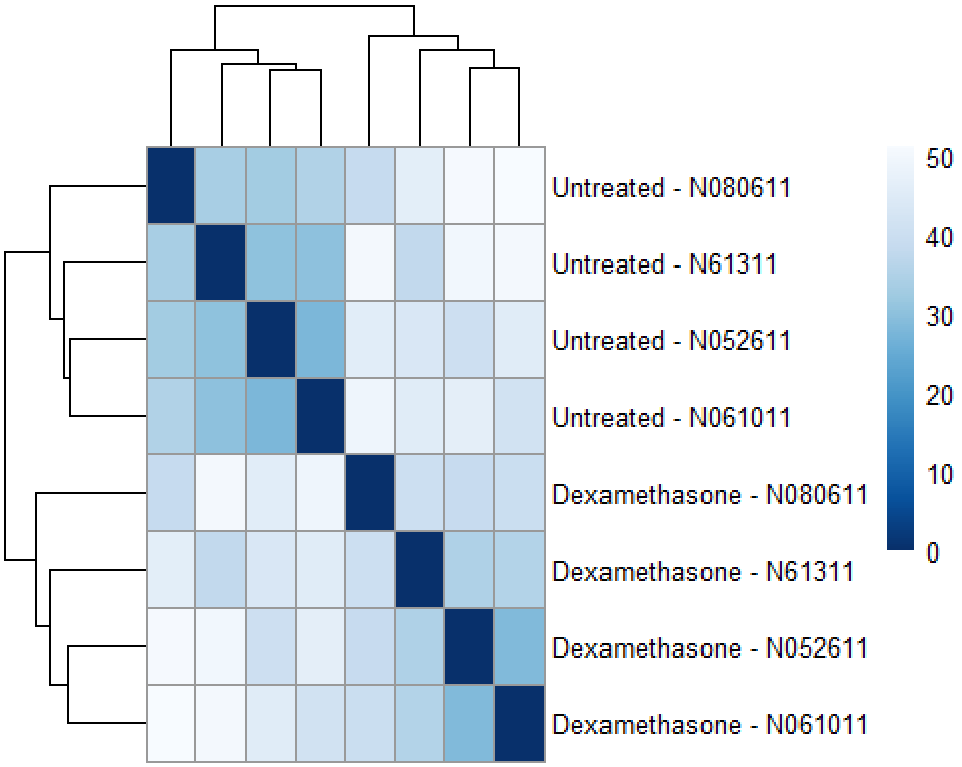
```
## DataFrame with 8 rows and 5 columns
##                  names    donor     condition     cell           dex
##               <factor> <factor>      <factor> <factor>      <factor>
## SRR1039508 SRR1039508  N61311  Untreated      N61311  Untreated
## SRR1039509 SRR1039509  N61311  Dexamethasone  N61311  Dexamethasone
## SRR1039512 SRR1039512  N052611 Untreated      N052611 Untreated
## SRR1039513 SRR1039513  N052611 Dexamethasone  N052611 Dexamethasone
## SRR1039516 SRR1039516  N080611 Untreated      N080611 Untreated
## SRR1039517 SRR1039517  N080611 Dexamethasone  N080611 Dexamethasone
## SRR1039520 SRR1039520  N061011 Untreated      N061011 Untreated
## SRR1039521 SRR1039521  N061011 Dexamethasone  N061011 Dexamethasone
```

```
###################################################################
## 05. Show the effect of transformation
###################################################################
# The differences between samples (deviation from y=x in these scatterplots)
# will contribute to the distance calculations and the PCA plot.
# We can see how genes with low counts (bottom left-hand corner) seem to be
# excessively variable on the ordinary logarithmic scale,
# while the VST transformation compresses differences for the low count genes
# for which the data provide little information about differential expression.
dds <- estimateSizeFactors(dds)

df <- bind_rows(
  as_data_frame(log2(counts(dds, normalized=TRUE)[, 1:2]+1)) %>%
    mutate(transformation = "log2(x + 1)"),
  as_data_frame(assay(vsd)[, 1:2]) %>% mutate(transformation = "vst"))

colnames(df)[1:2] <- c("x", "y")

lvls <- c("log2(x + 1)", "vst")
df$transformation <- factor(df$transformation, levels=lvls)

g=ggplot(df, aes(x = x, y = y)) + geom_hex(bins = 80) +
  coord_fixed() + facet_grid( . ~ transformation)
print(g)
```



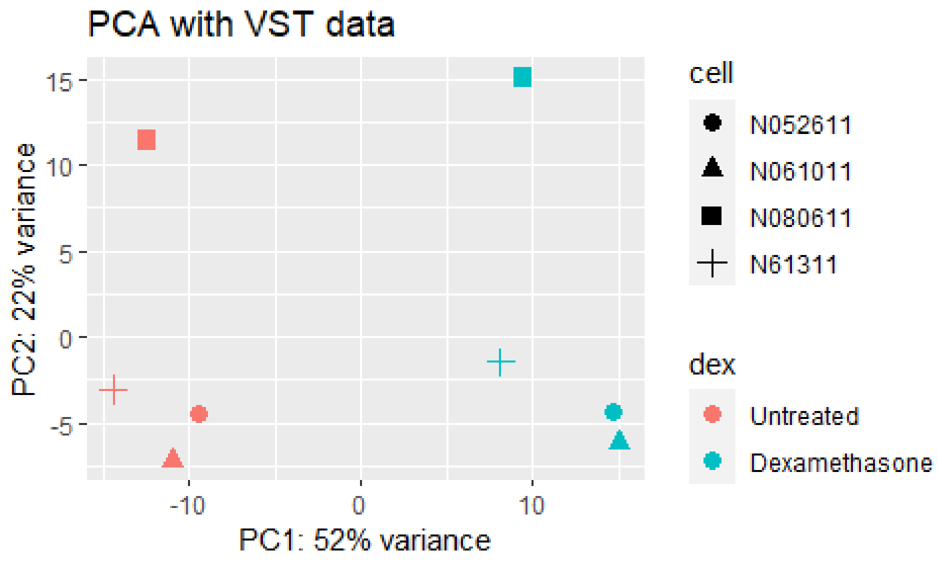
```
# Clean-up
rm(lvls, df, g)
```

```
###################################################################
## 06. Sample Distances
###################################################################
# A useful first step in an RNA-seq analysis is often to assess overall
# similarity between samples: Which samples are similar to each other,
# which are different? Does this fit to the expectation from the
# experiment’s design? We use the R function dist to calculate the
# Euclidean distance between samples. To ensure we have a roughly equal
# contribution from all genes, we use it on the VST data. We need to
# transpose the matrix of values using t, because the dist function expects
# the different samples to be rows of its argument, and different dimensions
# (here, genes) to be columns.
options(digits=3)
sampleDists <- dist(t(assay(vsd)))
sampleDists

##            SRR1039508 SRR1039509 SRR1039512 SRR1039513 SRR1039516 SRR1039517
## SRR1039509       37.8
## SRR1039512       30.3       43.4
## SRR1039513       49.7       34.8       40.2
## SRR1039516       33.6       46.1       32.9       50.6
## SRR1039517       50.1       40.0       45.6       38.7       38.5
## SRR1039520       29.9       45.3       27.7       46.3       35.0       49.0
## SRR1039521       50.1       35.4       45.5       28.5       51.2       39.6
##            SRR1039520
## SRR1039509
## SRR1039512
## SRR1039513
## SRR1039516
## SRR1039517
## SRR1039520
## SRR1039521       41.5

# In order to plot the sample distance matrix with the rows/columns
# arranged by the distances in our distance matrix,
# we manually provide sampleDists to the clustering_distance argument of the
# pheatmap function. Otherwise the pheatmap function would assume that
# the matrix contains the data values themselves, and would calculate
# distances between the rows/columns of the distance matrix,
# which is not desired. Here, we also manually specify a blue color palette
# using the colorRampPalette function from the RColorBrewer package.
sampleDistMatrix <- as.matrix(sampleDists )
rownames(sampleDistMatrix) <- paste(vsd$dex, vsd$cell, sep = " - " )
colnames(sampleDistMatrix) <- NULL
colors <- colorRampPalette( rev(brewer.pal(9, "Blues")) )(255)
pheatmap(sampleDistMatrix,
         clustering_distance_rows = sampleDists,
         clustering_distance_cols = sampleDists,
         col = colors)
```



```
# Question 4: Re-draw the heat map with green colours


# Clean-up
rm(sampleDistMatrix, colors, sampleDists)
```

```
####################################################################
## 07. Principal Components Analysis (PCA)
####################################################################
# PCA is a way to visualize sample-to-sample distances. In this method,
# the data points (i.e., the samples) are projected onto the 2D plane
# such that they spread out in the two directions that explain most of the
# differences (figure below). The x-axis is the direction that separates
# the data points the most. The values of the samples in this direction
# are written PC1. The y-axis is a direction (orthogonal to the first
# direction) that separates the data the second most.
# The values of the samples in this direction are written PC2.
# The percent of the total variance that is contained in the direction
# is printed in the axis label. Note how the samples with the same treatment
# are closer to each other.
pcaData <- plotPCA(vsd, intgroup = c( "dex", "cell"), returnData = TRUE)

pcaData

##               PC1   PC2                 group           dex    cell       name
## SRR1039508 -14.43 -3.06      Untreated:N61311     Untreated  N61311 SRR1039508
## SRR1039509   8.14 -1.42  Dexamethasone:N61311 Dexamethasone  N61311 SRR1039509
## SRR1039512  -9.48 -4.41     Untreated:N052611     Untreated N052611 SRR1039512
## SRR1039513  14.63 -4.29 Dexamethasone:N052611 Dexamethasone N052611 SRR1039513
## SRR1039516 -12.43 11.40     Untreated:N080611     Untreated N080611 SRR1039516
## SRR1039517   9.48 15.10 Dexamethasone:N080611 Dexamethasone N080611 SRR1039517
## SRR1039520 -10.97 -7.20     Untreated:N061011     Untreated N061011 SRR1039520
## SRR1039521  15.05 -6.11 Dexamethasone:N061011 Dexamethasone N061011 SRR1039521

percentVar <- round(100 * attr(pcaData, "percentVar"))

g = ggplot(pcaData, aes(x = PC1, y = PC2, color = dex, shape = cell)) +
  geom_point(size =3) +
  xlab(paste0("PC1: ", percentVar[1], "% variance")) +
  ylab(paste0("PC2: ", percentVar[2], "% variance")) +
  coord_fixed() +
  ggtitle("PCA with VST data")
print(g)
```



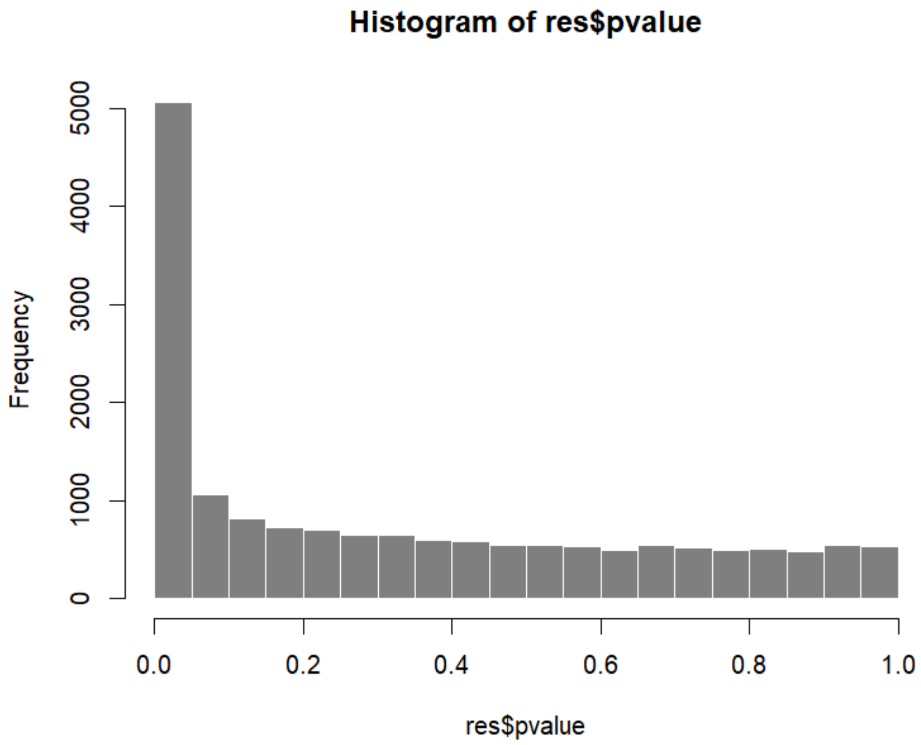
```
# Clean-up
rm(g, percentVar, pcaData)
```

```
####################################################################
## 08. Differential Expression Analysis
####################################################################
# We run the differential expression pipeline on the raw counts with a
# single call to the function DESeq.
# The column log2FoldChange is the effect size estimate.
# It tells us how much the gene’s expression seems to have changed due to
# treatment with dexamethasone in comparison to untreated samples.
# This value is reported on a logarithmic scale to base 2.
# For each gene a hypothesis testing is done to see whether evidence is
# sufficient to decide against the null hypothesis that there is zero effect
# of the treatment on the gene. The result of this test is reported as a
# p value, and it is found in the column pvalue. padj refers to adjusted
# (FDR-corrected) p value
dds <- DESeq(dds)

res <- results(dds)
res

## log2 fold change (MLE): dex Dexamethasone vs Untreated
## Wald test p-value: dex Dexamethasone vs Untreated
## DataFrame with 16637 rows and 6 columns
##                      baseMean log2FoldChange     lfcSE      stat      pvalue
##                     <numeric>      <numeric> <numeric> <numeric>   <numeric>
## ENSG00000000003.14  740.1093      -0.365327 0.1073385 -3.403501 6.65282e-04
## ENSG00000000419.12  511.6990       0.202232 0.1278579  1.581694 1.13720e-01
## ENSG00000000457.13  314.1680       0.033792 0.1552106  0.217717 8.27650e-01
## ENSG00000000460.16   79.7988      -0.120633 0.3055270 -0.394836 6.92964e-01
## ENSG00000000971.15 5715.3064       0.442982 0.0904089  4.899766 9.59508e-07
## ...                      ...            ...       ...       ...         ...
## ENSG00000285953.1    29.5747      -1.920562  0.649661 -2.956253  0.00311401
## ENSG00000285967.1   181.1650      -0.325885  0.179340 -1.817132  0.06919688
## ENSG00000285976.1   875.4424       0.262132  0.142980  1.833351  0.06675037
## ENSG00000285979.1    38.3502       0.338383  0.348445  0.971124  0.33148631
## ENSG00000285991.1    11.2772      -0.115472  0.723139 -0.159681  0.87313221
##                           padj
##                      <numeric>
## ENSG00000000003.14 4.63552e-03
## ENSG00000000419.12 2.88125e-01
## ENSG00000000457.13 9.21718e-01
## ENSG00000000460.16 8.48743e-01
## ENSG00000000971.15 1.36387e-05
## ...                        ...
## ENSG00000285953.1    0.0171889
## ENSG00000285967.1    0.2019448
## ENSG00000285976.1    0.1965808
## ENSG00000285979.1    0.5713913
## ENSG00000285991.1           NA

summary(res)

##
## out of 16637 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 2362, 14%
## LFC < 0 (down)     : 2019, 12%
## outliers [1]       : 0, 0%
## low counts [2]     : 646, 3.9%
## (mean count < 12)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

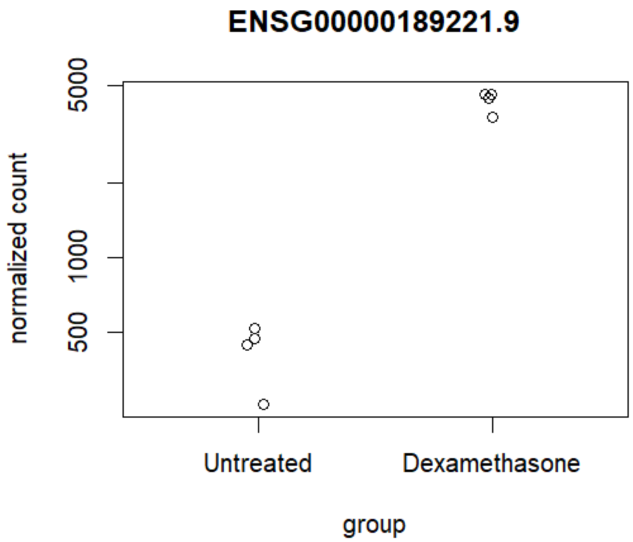
```
# If we consider 10% false-discovery as acceptable, then how many such
# genes are there?
sum(res$padj < 0.1, na.rm=TRUE)
```

```
## [1] 4381
```

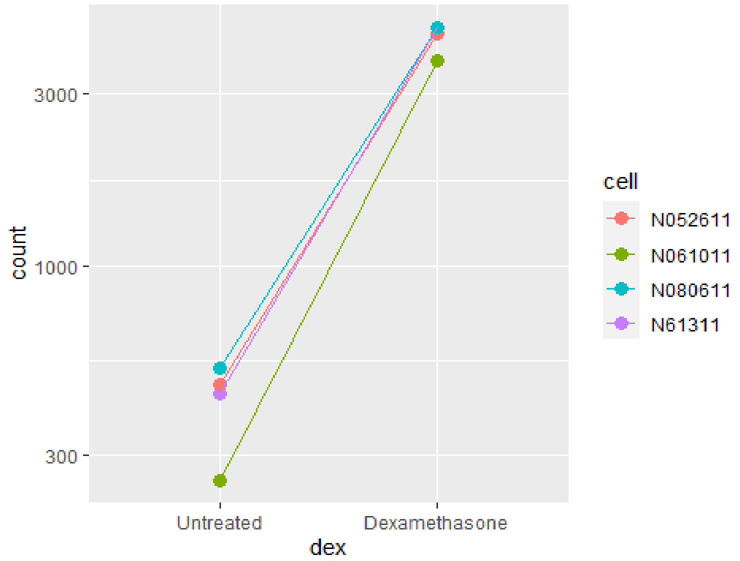
```r
###################################################################
## 09. Histogram of p-values
###################################################################
# The histogram represents a mix of two groups of genes
# - those that are differentially expressed (corresponding to
# the peak on the left side) and those are unchanged in expression
# (corresponding to a uniform distribution of p-values)
hist(res$pvalue, col = "grey50", border = "white")
```



```
#####################################################################
## 10. Plotting Results
#####################################################################
# Let us make some plots for the gene with the smallest padj value
topGene <- rownames(res)[which.min(res$padj)]
plotCounts(dds, gene = topGene, intgroup=c("dex"))
```

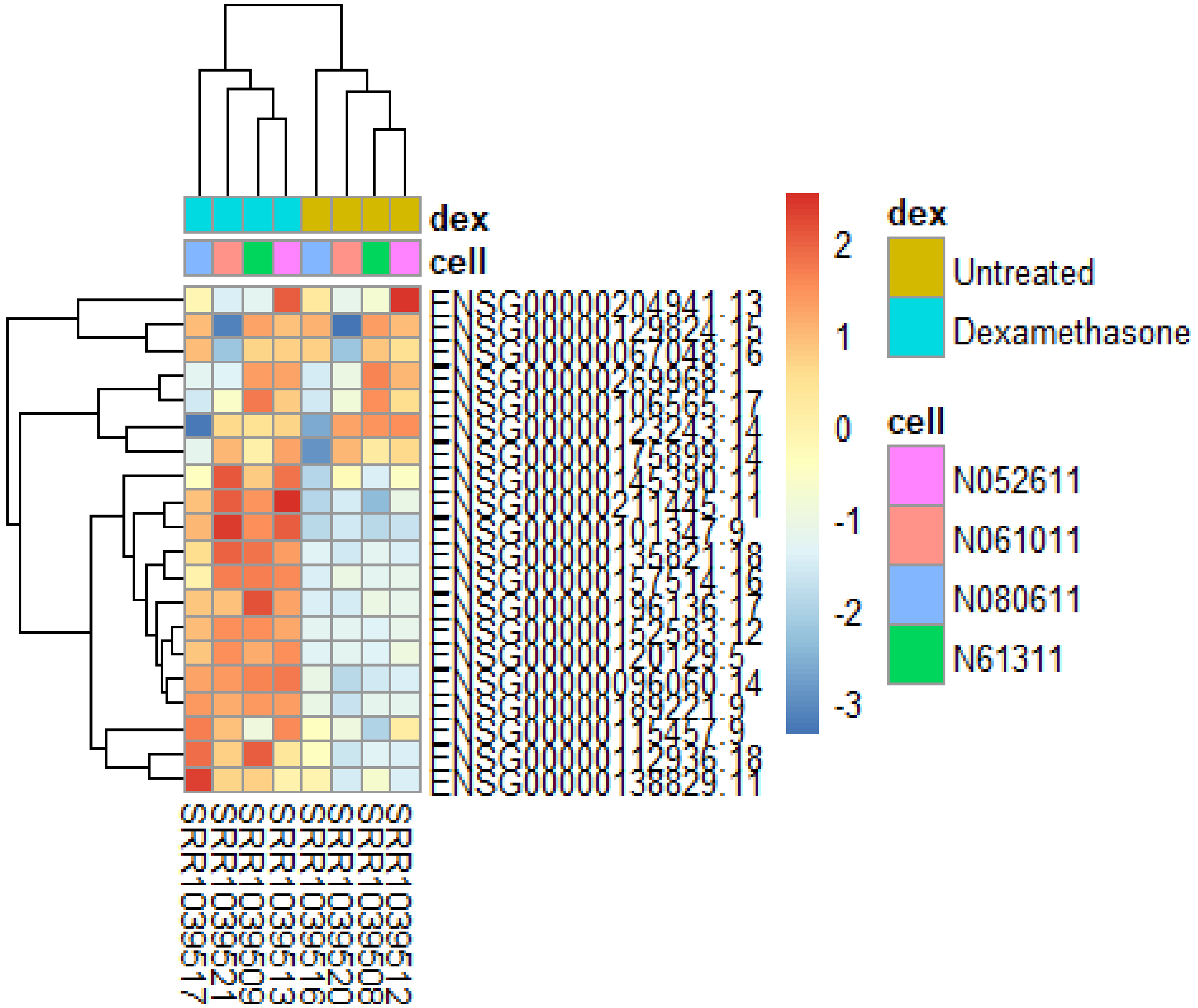


```
geneCounts <- plotCounts(dds, gene = topGene, intgroup = c("dex","cell"),
                         returnData = TRUE)
g = ggplot(geneCounts, aes(x = dex, y = count, color = cell, group = cell)) +
  scale_y_log10() + geom_point(size = 3) + geom_line()
print(g)
```



```
# Clean-up
rm(g, geneCounts)
```

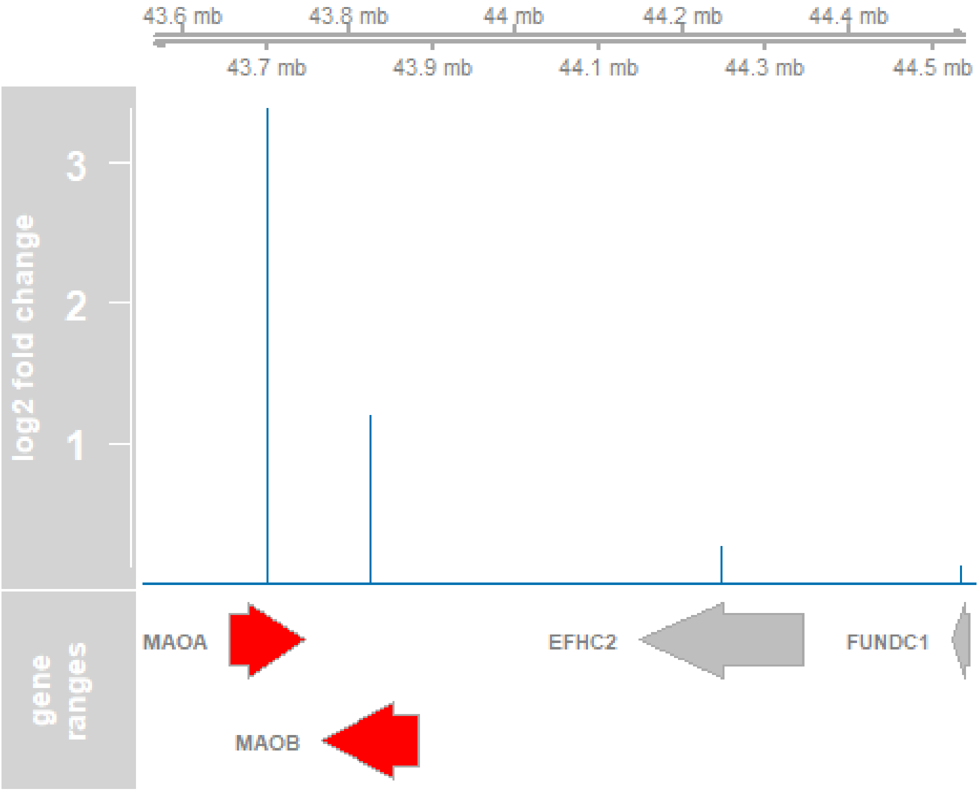
```
##################################################################
## 11. Gene Clustering
##################################################################
# In the sample distance heatmap made previously, the dendrogram
# at the side shows us a hierarchical clustering of the samples.
# Such a clustering can also be performed for the genes.
# Let us consider a subset of the most highly variable genes, i.e.,
# the 20 genes with the highest variance across samples.
# We work with the VST data vsd.
topVarGenes <- head(order(rowVars(assay(vsd)), decreasing = TRUE), 20)
mat  <- assay(vsd)[ topVarGenes, ]
mat  <- mat - rowMeans(mat)
anno <- as.data.frame(colData(vsd)[, c("cell","dex")])
pheatmap(mat, annotation_col = anno)
```



```
# Clean-up
rm(topVarGenes, mat, anno)
```

```
####################################################################
## 12. Annotation and Exporting Results
####################################################################
# Our result table so far only contains the Ensembl gene IDs,
# but alternative gene names may be more informative for interpretation.
# Bioconductor’s annotation packages help with mapping various ID schemes
# to each other. We need the AnnotationDbi package and
# the annotation package org.Hs.eg.db. These are the mappings available:
columns(org.Hs.eg.db)

##  [1] "ACCNUM"       "ALIAS"        "ENSEMBL"      "ENSEMBLPROT"  "ENSEMBLTRANS"
##  [6] "ENTREZID"     "ENZYME"       "EVIDENCE"     "EVIDENCEALL"  "GENENAME"
## [11] "GENETYPE"     "GO"           "GOALL"        "IPI"          "MAP"
## [16] "OMIM"         "ONTOLOGY"     "ONTOLOGYALL"  "PATH"         "PFAM"
## [21] "PMID"         "PROSITE"      "REFSEQ"       "SYMBOL"       "UCSCKG"
## [26] "UNIPROT"

# Add additional IDs (Entrez ID, Symbol, Name) to our result
# using he mapIds() function
ens.str <- substr(rownames(res), 1, 15)
res$entrez <- mapIds(org.Hs.eg.db,
                     keys=ens.str,
                     column="ENTREZID",
                     keytype="ENSEMBL",
                     multiVals="first")

res$symbol <- mapIds(org.Hs.eg.db,
                     keys=ens.str,
                     column="SYMBOL",
                     keytype="ENSEMBL",
                     multiVals="first")

res$name <- mapIds(org.Hs.eg.db,
                     keys=ens.str,
                     column="GENENAME",
                     keytype="ENSEMBL",
                     multiVals="first")

# Display the result for top DE genes with external Gene IDs.
resOrdered <- res[order(res$pvalue),]
head(resOrdered)

## log2 fold change (MLE): dex Dexamethasone vs Untreated
## Wald test p-value: dex Dexamethasone vs Untreated
## DataFrame with 6 rows and 9 columns
##                     baseMean log2FoldChange     lfcSE      stat       pvalue
##                    <numeric>      <numeric> <numeric> <numeric>    <numeric>
## ENSG00000189221.9   2371.265        3.39426  0.135575   25.0361 2.47518e-138
## ENSG00000120129.5   3417.255        2.96990  0.120928   24.5593 3.44226e-133
## ENSG00000101347.9  14106.720        3.74934  0.156084   24.0212 1.66892e-127
## ENSG00000152583.12   973.479        4.50022  0.197111   22.8309 2.26145e-115
## ENSG00000196136.17  2708.309        3.24329  0.143803   22.5538 1.23286e-112
## ENSG00000211445.11 12502.886        3.76804  0.168068   22.4197 2.52771e-111
##                               padj      entrez      symbol                   name
##                          <numeric> <character> <character>            <character>
## ENSG00000189221.9  3.95806e-134        4128        MAOA    monoamine oxidase A
## ENSG00000120129.5  2.75226e-129        1843       DUSP1 dual specificity pho..
## ENSG00000101347.9  8.89591e-124       25939      SAMHD1 SAM and HD domain co..
## ENSG00000152583.12 9.04071e-112        8404     SPARCL1           SPARC like 1
## ENSG00000196136.17 3.94293e-109          12    SERPINA3 serpin family A memb..
## ENSG00000211445.11 6.73678e-108        2878        GPX3 glutathione peroxida..

# To write data to file called results.csv, uncomment the following line.
#  write.csv(resOrdered, file = "results.csv")

# Clean-up
rm(resOrdered, ens.str)
```

```
###################################################################
## 13. Plotting Fold Changes in Genomic Space
###################################################################
# For this, we shall use the lfcShrink() function
# We specify a window of 1 million base pairs upstream and downstream
# from the gene with the smallest p value. We create a subset of
# our full results, for genes within the window. We add the gene symbol
# as a name if the symbol exists and is not duplicated in our subset.
# Log2 fold changes in genomic region surrounding the gene
# with smallest adjusted p value. Genes highlighted in red
# have adjusted p value less than 0.1.
resGR <- lfcShrink(dds, coef="dex_Dexamethasone_vs_Untreated",
                   type="apeglm",
                   format="GRanges")

## 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

resGR

## GRanges object with 16637 ranges and 5 metadata columns:
##                       seqnames              ranges strand |  baseMean
##                          <Rle>           <IRanges>  <Rle> | <numeric>
##   ENSG00000000003.14     chrX 100627109-100639991      - |  740.1093
##   ENSG00000000419.12    chr20   50934867-50958555      - |  511.6990
##   ENSG00000000457.13     chr1 169849631-169894267      - |  314.1680
##   ENSG00000000460.16     chr1 169662007-169854080      + |   79.7988
##   ENSG00000000971.15     chr1 196651878-196747504      + | 5715.3064
##                  ...      ...                 ...    ... .       ...
##    ENSG00000285953.1     chr7   92131774-92245924      - |   29.5747
##    ENSG00000285967.1     chr5   36864425-36876700      - |  181.1650
##    ENSG00000285976.1     chr6   63572012-63583587      + |  875.4424
##    ENSG00000285979.1    chr16   57177349-57181390      + |   38.3502
##    ENSG00000285991.1     chr6 149817937-149896011      - |   11.2772
##                       log2FoldChange     lfcSE      pvalue        padj
##                            <numeric> <numeric>   <numeric>   <numeric>
##   ENSG00000000003.14     -0.3401220 0.1063304 6.65282e-04 4.63552e-03
##   ENSG00000000419.12      0.1770178 0.1218213 1.13720e-01 2.88125e-01
##   ENSG00000000457.13      0.0269533 0.1394669 8.27650e-01 9.21718e-01
##   ENSG00000000460.16     -0.0616870 0.2231260 6.92964e-01 8.48743e-01
##   ENSG00000000971.15      0.4206549 0.0903614 9.59508e-07 1.36387e-05
##                  ...            ...       ...         ...         ...
##    ENSG00000285953.1     -1.2313653  0.916996  0.00311401   0.0171889
##    ENSG00000285967.1     -0.2623422  0.169901  0.06919688   0.2019448
##    ENSG00000285976.1      0.2247352  0.136898  0.06675037   0.1965808
##    ENSG00000285979.1      0.1614980  0.257628  0.33148631   0.5713913
##    ENSG00000285991.1     -0.0168702  0.290081  0.87313221          NA
##   -------
##   seqinfo: 25 sequences (1 circular) from hg38 genome

ens.str <- substr(names(resGR), 1, 15)
resGR$symbol <- mapIds(org.Hs.eg.db, ens.str, "SYMBOL", "ENSEMBL")

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

window <- resGR[topGene] + 1e6
strand(window) <- "*"
resGRsub <- resGR[resGR %over% window]
naOrDup <- is.na(resGRsub$symbol) | duplicated(resGRsub$symbol)
resGRsub$group <- ifelse(naOrDup, names(resGRsub), resGRsub$symbol)
status <- factor(ifelse(resGRsub$padj < 0.05 & !is.na(resGRsub$padj),
                        "sig", "notsig"))
options(ucscChromosomeNames = FALSE)
g <- GenomeAxisTrack()
a <- AnnotationTrack(resGRsub, name = "gene ranges", feature = status)
d <- DataTrack(resGRsub, data = "log2FoldChange", baseline = 0,
               type = "h", name = "log2 fold change", strand = "+")
plotTracks(list(g, d, a), groupAnnotation = "group",
           notsig = "grey", sig = "red")
```



```
# Clean-up
rm(a,d,g,resGR, resGRsub, ens.str, naOrDup, status, window)
```

```
####################################################################
## 14. Session Information
####################################################################

sessionInfo()

## R version 4.3.1 (2023-06-16 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19045)
## 
## Matrix products: default
## 
## 
## locale:
## [1] LC_COLLATE=English_India.utf8  LC_CTYPE=English_India.utf8   
## [3] LC_MONETARY=English_India.utf8 LC_NUMERIC=C                  
## [5] LC_TIME=English_India.utf8    
## 
## time zone: Asia/Calcutta
## tzcode source: internal
## 
## attached base packages:
## [1] grid      stats4    stats     graphics  grDevices utils     datasets 
## [8] methods   base     
## 
## other attached packages:
##  [1] Gviz_1.46.1                 org.Hs.eg.db_3.18.0        
##  [3] AnnotationDbi_1.64.1        genefilter_1.84.0          
##  [5] apeglm_1.24.0               ggbeeswarm_0.7.2           
##  [7] glmpca_0.2.0                RColorBrewer_1.1-3         
##  [9] pheatmap_1.0.12             vsn_3.70.0                 
## [11] DESeq2_1.42.1               ggplot2_3.4.4              
## [13] airway_1.22.0               SummarizedExperiment_1.32.0
## [15] Biobase_2.62.0              GenomicRanges_1.54.1       
## [17] GenomeInfoDb_1.38.2         IRanges_2.36.0             
## [19] S4Vectors_0.40.2            BiocGenerics_0.48.1        
## [21] MatrixGenerics_1.14.0       matrixStats_1.2.0          
## [23] dplyr_1.1.3                
## 
## loaded via a namespace (and not attached):
##   [1] rstudioapi_0.15.0        magrittr_2.0.3           
GenomicFeatures_1.54.4  
##   [4] farver_2.1.1             rmarkdown_2.25           BiocIO_1.12.0           
##   [7] zlibbioc_1.48.0          vctrs_0.6.4              memoise_2.0.1           
##  [10] Rsamtools_2.18.0         RCurl_1.98-1.13          base64enc_0.1-3         
##  [13] htmltools_0.5.7          S4Arrays_1.2.0           progress_1.2.3          
##  [16] curl_5.2.0               SparseArray_1.2.2        Formula_1.2-5           
##  [19] htmlwidgets_1.6.4        plyr_1.8.9               cachem_1.0.8            
##  [22] GenomicAlignments_1.38.2 lifecycle_1.0.4          pkgconfig_2.0.3         
##  [25] Matrix_1.6-4             R6_2.5.1                 fastmap_1.1.1           
##  [28] GenomeInfoDbData_1.2.11  digest_0.6.33            numDeriv_2016.8-1.1
##  [31] colorspace_2.1-0         Hmisc_5.1-1              RSQLite_2.3.4
##  [34] labeling_0.4.3           filelock_1.0.3           fansi_1.0.5
##  [37] httr_1.4.7               abind_1.4-5              compiler_4.3.1
##  [40] bit64_4.0.5              withr_2.5.2              backports_1.4.1
##  [43] htmlTable_2.4.2          BiocParallel_1.36.0      DBI_1.2.0
##  [46] highr_0.10               hexbin_1.28.3            biomaRt_2.58.2
##  [49] MASS_7.3-60              rappdirs_0.3.3           DelayedArray_0.28.0
##  [52] rjson_0.2.21             tools_4.3.1              foreign_0.8-86
##  [55] vipor_0.4.7              beeswarm_0.4.0           nnet_7.3-19
##  [58] glue_1.6.2               restfulr_0.0.15          checkmate_2.3.1
##  [61] cluster_2.1.6            generics_0.1.3           gtable_0.3.4
##  [64] BSgenome_1.70.2          preprocessCore_1.64.0    ensembldb_2.26.1
##  [67] data.table_1.14.10       hms_1.1.3                xml2_1.3.6
##  [70] utf8_1.2.4               XVector_0.42.0           pillar_1.9.0
##  [73] stringr_1.5.1            emdbook_1.3.13           limma_3.58.1
##  [76] splines_4.3.1            BiocFileCache_2.10.1     lattice_0.22-5
##  [79] deldir_2.0-4             survival_3.5-7           rtracklayer_1.62.0
##  [82] bit_4.0.5                annotate_1.80.0          biovizBase_1.50.0
##  [85] tidyselect_1.2.1         locfit_1.5-9.8           Biostrings_2.70.1
##  [88] knitr_1.45               gridExtra_2.3            ProtGenerics_1.34.0
##  [91] xfun_0.41                statmod_1.5.0            stringi_1.7.12
##  [94] lazyeval_0.2.2           yaml_2.3.8               evaluate_0.23
##  [97] codetools_0.2-19         interp_1.1-6             bbmle_1.0.25.1
## [100] tibble_3.2.1             BiocManager_1.30.22      cli_3.6.1
## [103] affyio_1.72.0            rpart_4.1.23             xtable_1.8-4
## [106] munsell_0.5.0            dichromat_2.0-0.1        Rcpp_1.0.11
## [109] dbplyr_2.5.0             coda_0.19-4              png_0.1-8
## [112] bdsmatrix_1.3-7          XML_3.99-0.16            parallel_4.3.1
## [115] blob_1.2.4               prettyunits_1.2.0        jpeg_0.1-10
## [118] latticeExtra_0.6-30      AnnotationFilter_1.26.0  bitops_1.0-7
## [121] mvtnorm_1.2-4            VariantAnnotation_1.48.1 scales_1.3.0
## [124] affy_1.80.0              crayon_1.5.2             rlang_1.1.1
## [127] KEGGREST_1.42.0
```