Pratica RNAseq

```
library(DESeq2)
library(ggplot2)
library(tximeta)
library(pheatmap)
library(viridis)
setwd("~/PRATICA_RNASEQ/")
```

Carregue as informações sobre o experimento, nomes de amostras e condição

```
targets<-read.table("metadata.csv",
                    header=T,
                    sep = ",",
                    stringsAsFactors=FALSE,
                    col.names = c("names", "condition", "cell"))
targets$names<-as.factor(targets$names)
targets$cell<-as.factor(targets$cell)
targets$group <- as.factor(paste0(targets$condition, "_", targets$cell))
```

fazer uma lista dos arquivos de quantificação criados pelo salmon. Por favor note que estas são apenas "strings"

```
files<-file.path(paste0("Quantification/",targets$names,"/quant.sf",se =""))
```

verificar se a sequência (nomes de arquivo) criada nas etapas anteriores são arquivos reais em disco

```
file.exists(files)
```

```
## [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

Adicione os nomes dos arquivos à descrição do experimento.

```
targets$files<-files
row.names(targets)<-targets$names
```

verificar o objeto targets

```
targets
```

```
##                 names     condition    cell                 group
## SRR1039508 SRR1039508     Untreated  N61311      Untreated_N61311
## SRR1039509 SRR1039509 Dexamethasone  N61311  Dexamethasone_N61311
## SRR1039512 SRR1039512     Untreated N052611     Untreated_N052611
## SRR1039513 SRR1039513 Dexamethasone N052611 Dexamethasone_N052611
## SRR1039516 SRR1039516     Untreated N080611     Untreated_N080611
## SRR1039517 SRR1039517 Dexamethasone N080611 Dexamethasone_N080611
## SRR1039520 SRR1039520     Untreated N061011     Untreated_N061011
## SRR1039521 SRR1039521 Dexamethasone N061011 Dexamethasone_N061011
##                                         files
## SRR1039508 Quantification/SRR1039508/quant.sf
## SRR1039509 Quantification/SRR1039509/quant.sf
## SRR1039512 Quantification/SRR1039512/quant.sf
```

```
## SRR1039513 Quantification/SRR1039513/quant.sf
## SRR1039516 Quantification/SRR1039516/quant.sf
## SRR1039517 Quantification/SRR1039517/quant.sf
## SRR1039520 Quantification/SRR1039520/quant.sf
## SRR1039521 Quantification/SRR1039521/quant.sf
```

Importaremos a quantificação da transcrição gerada pelo salmon, e a descrição do experimento usando tximeta Verifique o manual de tximeta para obter mais detalhes: https://bioconductor.org/packages/3.14/bioc/vignettes/tximeta/inst/doc/tximeta.html Agora, antes de carregar nossos dados, crie o objeto transcriptome, isso vai ajudar a realizar várias operações, como resumir os níveis de expressão da transcrição no nível genético, obter informações funcionais sobre os genes, e assim por diante.

```
makeLinkedTxome(indexDir="salmon_index",
                source="LocalEnsembl",
                organism="Homo sapiens",
                release="38",
                genome="GRCh38.p13",
                fasta="./references/gencode.v38.transcripts.fa.gz",
                gtf="./references/gencode.v38.annotation.gtf.gz",
                write=FALSE)
```

Agora crie o objeto tximeta que tem os dados de salmão, o experimento descrição, e acesso às informações do transcriptome. Este objeto é chamado Experiência resumida

```
se <- tximeta(targets)
```

Verifique o tamanho do objeto

```
dim(se)
```

```
## [1] 236186      8
```

verificar os nomes das transcrições, confirmar que você está vendo os nomes da transcrição e não nomes de genes. Como você faria agora.

```
head(rownames(se))
```

```
## [1] "ENST00000456328.2" "ENST00000450305.2" "ENST00000488147.1"
## [4] "ENST00000619216.1" "ENST00000473358.1" "ENST00000469289.1"
```

exploramos as informações do transcriptome para resumir os dados de expressão no nível genético. Você entende o que isso significa?

```
gse <- summarizeToGene(se)
```

Verifique o tamanho do objeto e verifique os nomes dos genes. Deve ser agora nomes de genes e não nomes de transcrições

```
dim(gse)
```

```
## [1] 60230     8
```

```
head(rownames(gse))
```

```
## [1] "ENSG00000000003.15" "ENSG00000000005.6"  "ENSG00000000419.14"
## [4] "ENSG00000000457.14" "ENSG00000000460.17" "ENSG00000000938.13"
```

esses dois objetos são compossados de colData, intervalos e dados de ensaio, ver figura vamos ver cada um desses elementos em nossos objetos “se” e “gse” Veja o número na seção 2.5 de https://bioconductor.org/packages/release/workflows/vignettes/rnaseqGene/inst/doc/rnaseqGene.html

```
colData(gse)
```

```
## DataFrame with 8 rows and 4 columns
##                 names     condition     cell                group
##              <factor>   <character> <factor>             <factor>
## SRR1039508 SRR1039508     Untreated  N61311  Untreated_N61311
## SRR1039509 SRR1039509 Dexamethasone  N61311  Dexamethasone_N61311
## SRR1039512 SRR1039512     Untreated  N052611 Untreated_N052611
## SRR1039513 SRR1039513 Dexamethasone  N052611 Dexamethasone_N052611
## SRR1039516 SRR1039516     Untreated  N080611 Untreated_N080611
## SRR1039517 SRR1039517 Dexamethasone  N080611 Dexamethasone_N080611
## SRR1039520 SRR1039520     Untreated  N061011 Untreated_N061011
## SRR1039521 SRR1039521 Dexamethasone  N061011 Dexamethasone_N061011
```

```
head(assay(gse))
```

```
##                    SRR1039508 SRR1039509 SRR1039512 SRR1039513 SRR1039516
## ENSG00000000003.15     56.109     35.919     61.439     46.000     77.001
## ENSG00000000005.6       0.000      0.000      0.000      0.000      0.000
## ENSG00000000419.14     39.003     54.013     39.970     41.000     46.001
## ENSG00000000457.14     30.645     29.552     22.995     25.501     23.330
## ENSG00000000460.17      5.701      9.453      2.000      3.499     11.000
## ENSG00000000938.13      0.000      0.000      0.000      0.000      0.000
##                    SRR1039517 SRR1039520 SRR1039521
## ENSG00000000003.15     77.984     63.480     55.973
## ENSG00000000005.6       0.000      0.000      0.000
## ENSG00000000419.14     41.017     40.389     36.001
## ENSG00000000457.14     22.251     19.409     27.784
## ENSG00000000460.17     11.739     10.591      5.000
## ENSG00000000938.13      0.000      0.000      0.000
```

```
rowRanges(gse)
```

```
## GRanges object with 60230 ranges and 2 metadata columns:
##                       seqnames              ranges strand |            gene_id
##                          <Rle>           <IRanges>  <Rle> |        <character>
##   ENSG00000000003.15     chrX 100627108-100639991      - | ENSG00000000003.15
##    ENSG00000000005.6     chrX 100584936-100599885      + |  ENSG00000000005.6
##   ENSG00000000419.14    chr20   50934867-50959140      - | ENSG00000000419.14
##   ENSG00000000457.14     chr1 169849631-169894267      - | ENSG00000000457.14
##   ENSG00000000460.17     chr1 169662007-169854080      + | ENSG00000000460.17
##                  ...      ...                 ...    ... .                ...
##    ENSG00000288721.1     chr6   41793314-41921139      - |  ENSG00000288721.1
##    ENSG00000288722.1     chrX 154886355-154888061      + |  ENSG00000288722.1
##    ENSG00000288723.1     chr1 241722926-241848128      - |  ENSG00000288723.1
##    ENSG00000288724.1     chr3   46284775-46293795      - |  ENSG00000288724.1
##    ENSG00000288725.1    chr16   56430556-56501497      - |  ENSG00000288725.1
##                                                                          tx_ids
##                                                                   <CharacterList>
##   ENSG00000000003.15  ENST00000373020.9,ENST00000612152.4,ENST00000614008.4,...
##    ENSG00000000005.6                       ENST00000373031.5,ENST00000485971.1
##   ENSG00000000419.14 ENST00000466152.5,ENST00000371582.8,ENST00000371588.10,...
##   ENSG00000000457.14 ENST00000367771.11,ENST00000367770.5,ENST00000367772.8,...
##   ENSG00000000460.17  ENST00000498289.5,ENST00000472795.5,ENST00000359326.9,...
##                  ...                                                        ...
```

```
##    ENSG00000288721.1      ENST00000684631.1,ENST00000682596.1,ENST00000683313.1
##    ENSG00000288722.1                                          ENST00000610495.2
##    ENSG00000288723.1                                          ENST00000684005.1
##    ENSG00000288724.1                                          ENST00000683399.1
##    ENSG00000288725.1                                          ENST00000684388.1
##   -------
##   seqinfo: 25 sequences (1 circular) from an unspecified genome; no seqlengths
```

Com estes objetos (gse) podemos agora iniciar a análise diferencial de expressão genética Para isso existem muitos pacotes R diferentes que poderiam ser usados hoje vamos usar o pacote DESeq2. Então, primeiro precisamos transformar nosso objeto SummarizedExperiment (gse) para um objeto que é nativo do DESeq2, que é DESeqDataSet, além dos dados, o DESeqDataSet requer uma descrição do design experimental, em que dizemos quais são o fator de interesse no estudo, e/ou o fatores que devem ser fontes de variação e como lidar com eles. Isso é feito usando uma fórmula, com o mesmo sintax que em modelos lineares simples (lm) em R.

Neste exemplo em particular, temos pelo menos (que sabemos) fontes de variação. Primeiro, a condição, amostras tratadas ou não com Dexametasona. Segundo, a linha celular. A célula ASM primária foi obtida de quatro doadores, e depois divididos nos grupos tratados e não tratados. Então podemos ter um efeito do doador. Verifique se os alvos se opõem para ver isso. Usaremos esses dois fatores em nossas análises. Queremos testar o efeito de Dexametasona, enquanto controla o efeito de diferentes linhas celulares. Note que este é um design experimental emparelhado. Como cada amostra (cellLine) foi tratado e não tratado nesse caso, especificaremos a fórmula como ~ cellLine + condição

```
dds <- DESeqDataSet(gse, design = ~cell+condition)
```

É comum que os genes no conjunto de dados não expressos em tudo, e outros para ser muito humildemente expresso, é uma boa prática para removê-los de análises adicionais

```
nrow(dds) #Number of genes pre-filtering
```

```
## [1] 60230
```

```
keep <- rowSums(counts(dds)) > 1
dds <- dds[keep,]
nrow(dds) #Number of genes after filtering out non-expressed genes.
```

```
## [1] 22637
```

Verifique o efeito da normalização

```
head(counts(dds),2)
```

```
##                    SRR1039508 SRR1039509 SRR1039512 SRR1039513 SRR1039516
## ENSG00000000003.15         56         36         61         46         77
## ENSG00000000419.14         39         54         40         41         46
##                    SRR1039517 SRR1039520 SRR1039521
## ENSG00000000003.15         78         63         56
## ENSG00000000419.14         41         40         36
```

```
dds <- estimateSizeFactors(dds)
head(counts(dds,normalized=TRUE),2)
```

```
##                    SRR1039508 SRR1039509 SRR1039512 SRR1039513 SRR1039516
## ENSG00000000003.15   44.39943   29.97241   81.84358   40.49370   67.06722
## ENSG00000000419.14   38.28429   58.30182   38.52643   42.48666   42.47223
##                    SRR1039517 SRR1039520 SRR1039521
## ENSG00000000003.15   89.53866   93.24759   48.53583
## ENSG00000000419.14   40.73567   37.69782   39.36373
```

```
colSums(assay(gse))
```

```
## SRR1039508 SRR1039509 SRR1039512 SRR1039513 SRR1039516 SRR1039517 SRR1039520
##    1788789    1752398    1711969    1661466    1638343    1636627    1652720
## SRR1039521
##    1645795
```

Também é uma boa ideia remover genes em que um determinado número de amostras não têm uma contagem

```
keep <- rowSums(counts(dds) >= 5) >= 4
dds <- dds[keep,]
nrow(dds) # Number of genes, that have counts in at least 4 samples, and the sum of counts is greated t
```

```
## [1] 12493
```

Agora que os dados estão no formato adequado e filtrados podemos realizar algumas análises exploratórias para verificar se os dados parecem bons

Muitas das análises exploratórias requerem dados homocedastic, que nossas contagens não são. No DESeq2 a função vst e rlog, pode normalizar os dados para torná-lo mais hocedasticos . Como regra vst é melhor quando você tem mais de 30 amostras

```
rld <- rlog(dds, blind = TRUE)
head(assay(rld))
```

```
##                    SRR1039508 SRR1039509 SRR1039512 SRR1039513 SRR1039516
## ENSG00000000003.15   5.668507   5.412780   6.131921   5.606821   5.977217
## ENSG00000000419.14   5.328875   5.627996   5.332968   5.399332   5.399182
## ENSG00000000457.14   4.660658   4.683921   4.567988   4.858900   4.463051
## ENSG00000000460.17   2.710491   3.192373   2.643494   2.710112   3.287736
## ENSG00000000971.16   8.355375   8.681668   8.717486   9.024253   8.727782
## ENSG00000001036.14   7.378111   7.354549   7.537149   7.312464   7.249918
##                    SRR1039517 SRR1039520 SRR1039521
## ENSG00000000003.15   6.209182   6.237475   5.732909
## ENSG00000000419.14   5.370584   5.318391   5.347868
## ENSG00000000457.14   4.561684   4.576634   4.843399
## ENSG00000000460.17   3.009248   2.911021   2.707349
## ENSG00000000971.16   9.155519   8.788360   9.286782
## ENSG00000001036.14   7.075706   7.401943   7.059771
```

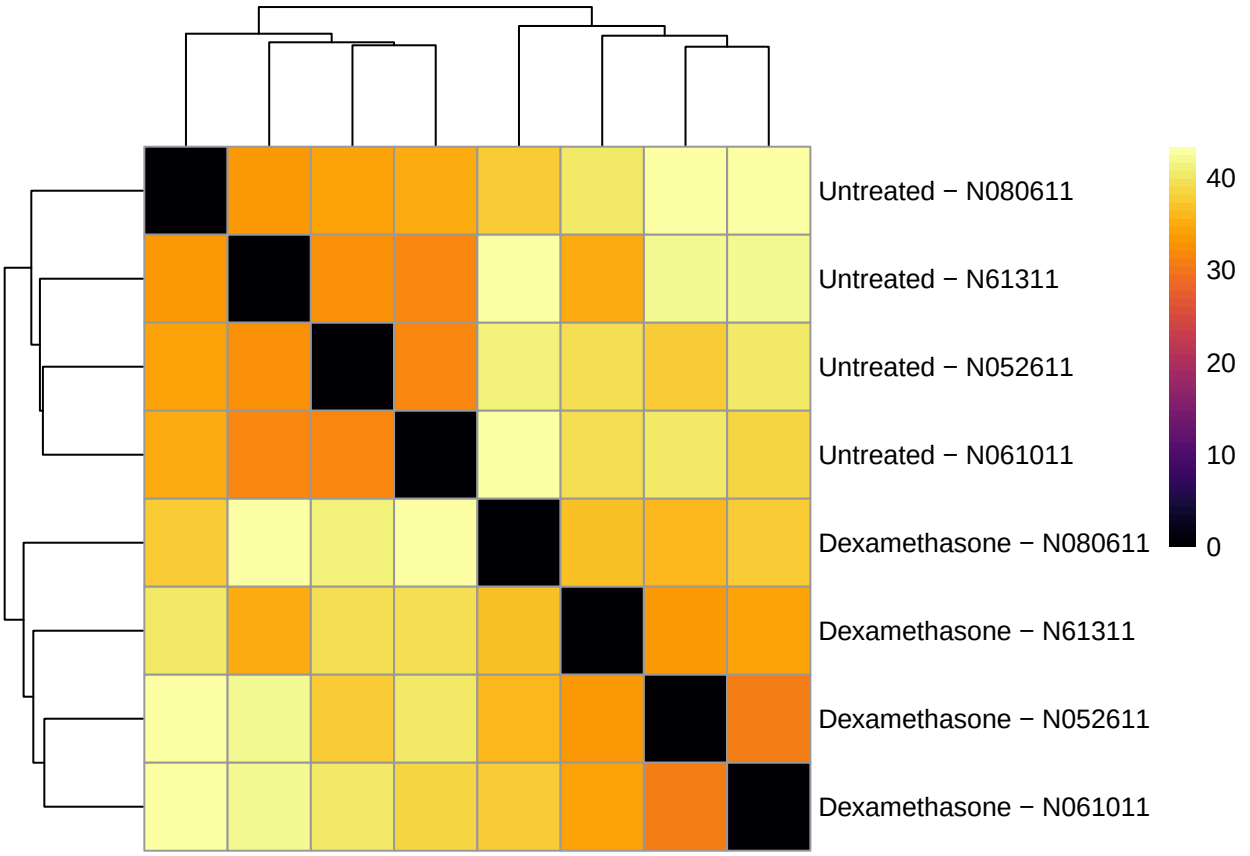
Agora vamos olhar para a semelhança entre as amostras vamos calcular a distância euclidiana entre as amostras com a função dist Esta função pressupõe que as amostras são as linhas, por isso devemos primeiro transpor a matriz Vamos visualizar as distâncias entre as amostras como um mapa de calor

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

```
##            SRR1039508 SRR1039509 SRR1039512 SRR1039513 SRR1039516 SRR1039517
## SRR1039509   35.46829
## SRR1039512   32.38620   39.08480
## SRR1039513   41.89259   33.74070   37.39767
## SRR1039516   33.68100   40.43479   34.23108   42.86046
## SRR1039517   42.84794   36.91977   40.91517   36.25173   37.64534
## SRR1039520   31.77332   39.63508   31.46431   40.20266   35.05061   42.50047
## SRR1039521   42.12258   34.43171   40.38196   31.01407   43.32233   37.44675
##            SRR1039520
## SRR1039509
## SRR1039512
```

```
## SRR1039513
## SRR1039516
## SRR1039517
## SRR1039520
## SRR1039521   38.17167
```

```
sampleDistMatrix <- as.matrix( sampleDists )
rownames(sampleDistMatrix) <- paste( rld$condition, rld$cell, sep = " - " )
colnames(sampleDistMatrix) <- NULL
colors <- inferno(50)
pheatmap(sampleDistMatrix,
         clustering_distance_rows = sampleDists,
         clustering_distance_cols = sampleDists,
         col = colors)
```

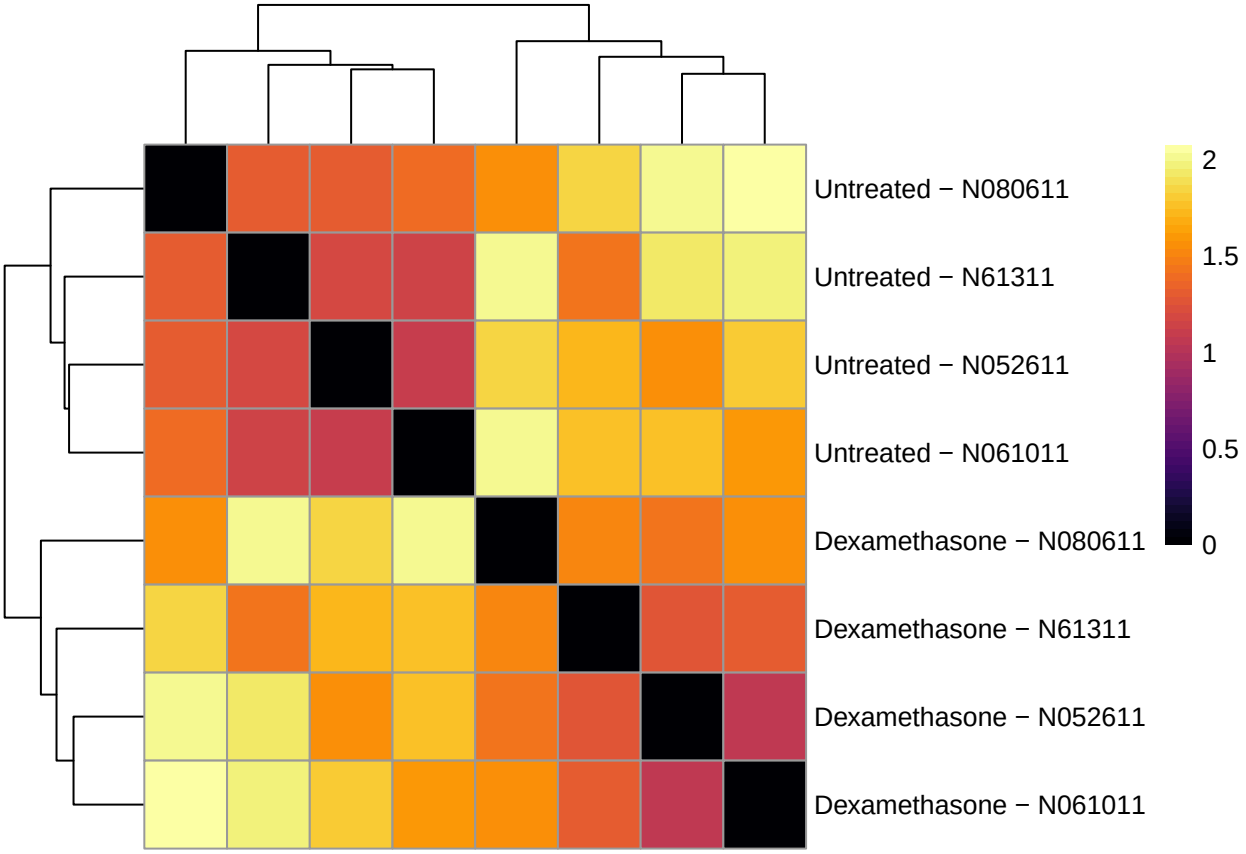



```
sampleDists <- dist(t(assay(rld)))
sampleDists
```

```
##            SRR1039508 SRR1039509 SRR1039512 SRR1039513 SRR1039516 SRR1039517
## SRR1039509   35.46829
## SRR1039512   32.38620   39.08480
## SRR1039513   41.89259   33.74070   37.39767
## SRR1039516   33.68100   40.43479   34.23108   42.86046
## SRR1039517   42.84794   36.91977   40.91517   36.25173   37.64534
## SRR1039520   31.77332   39.63508   31.46431   40.20266   35.05061   42.50047
## SRR1039521   42.12258   34.43171   40.38196   31.01407   43.32233   37.44675
##            SRR1039520
## SRR1039509
```

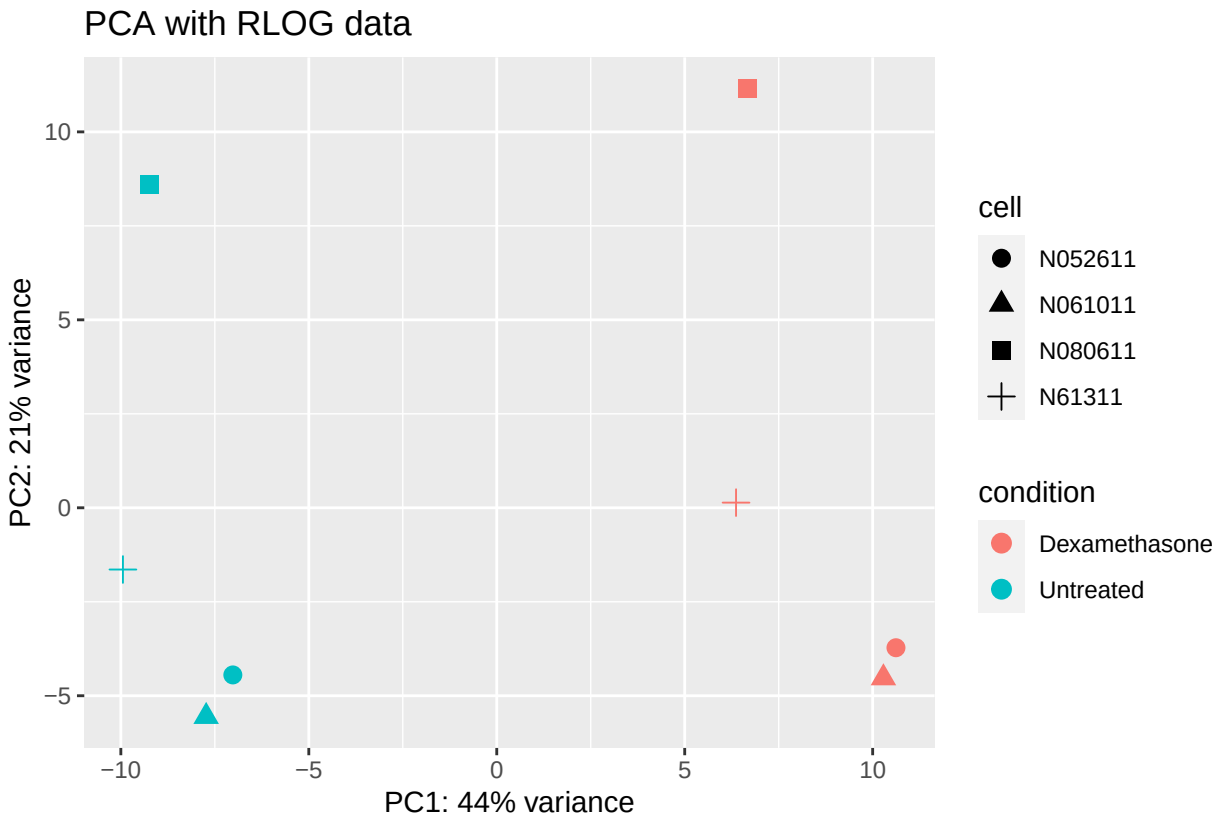
```
## SRR1039512
## SRR1039513
## SRR1039516
## SRR1039517
## SRR1039520
## SRR1039521   38.17167
```

```
samplecor <- as.dist((1-(cor(assay(rld), method='pearson')))*100)
sampleCorMatrix <- as.matrix( samplecor )
rownames(sampleCorMatrix) <- paste( rld$condition, rld$cell, sep = " - " )
colnames(sampleCorMatrix) <- NULL
pheatmap(sampleCorMatrix,
         clustering_distance_rows = samplecor,
         clustering_distance_cols = samplecor,
         col=colors)
```



Também é comum mostrar um enredo PCA das amostras, colorido com os fatores de interesse

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



Agora fazemos o análise de expressão diferencial e olhamos seu nível de expressão nas diferentes condições

```
dds <- DESeq(dds)
res <- results(dds, lfcThreshold = 1, altHypothesis = "greaterAbs", parallel = T)
```

Olhamos o tamanho do objeto gerado

```
dim(res)
```

```
## [1] 12493     6
```

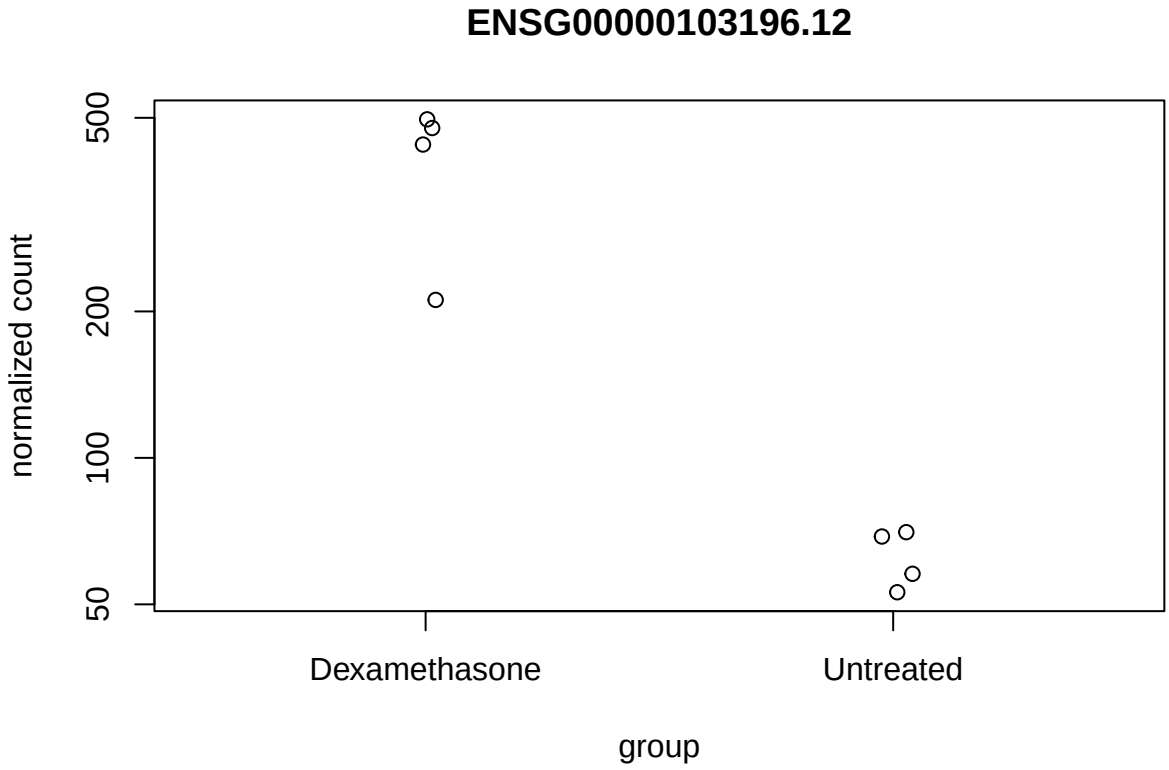
Vamos ver se o gene ENSG00000103196.12 está nos genes que estão expressos diferencialmente

```
df <- as.data.frame(res)
df["ENSG00000103196.12",]
```

```
##                     baseMean log2FoldChange     lfcSE      stat       pvalue
## ENSG00000103196.12 233.8174      -2.657034 0.2180493 -7.599355 2.976106e-14
##                          padj
## ENSG00000103196.12 2.855709e-11
```

fazemos uma grafica dos níveis de expressão do gene “ENSG00000103196.12” nas diferentes condições

```
plotCounts(dds, gene = "ENSG00000103196.12")
```


ENSG00000103196.12
normalized count
500
200
100
50
Dexamethasone
Untreated
group