Differential Expression

Antonio Osamu Katagiri Tanaka - A01212611@itesm.mx

April 03, 2020

```r
# Clear all objects (from the workspace)
rm(list = ls())

# Suppress Warning messages
options(warn = -1)

# Turn off scientific notation like 1e+06
# options(scipen=999)

options(stringsAsFactors = F)

# Load Libs

# # INSTALL with:
# if (!requireNamespace("BiocManager", quietly = TRUE))
#     install.packages("BiocManager")
# BiocManager::install("DESeq2")

library(tidyr)
library(DESeq2)
```

```
## Loading required package: S4Vectors

## Loading required package: stats4

## Loading required package: BiocGenerics

## Loading required package: parallel

##
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB

## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, basename, cbind, colnames,
##     dirname, do.call, duplicated, eval, evalq, Filter, Find, get, grep,
##     grepl, intersect, is.unsorted, lapply, Map, mapply, match, mget,
##     order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
##     rbind, Reduce, rownames, sapply, setdiff, sort, table, tapply,
##     union, unique, unsplit, which, which.max, which.min

##
## Attaching package: 'S4Vectors'

## The following object is masked from 'package:tidyr':
```

```
##
##     expand

## The following object is masked from 'package:base':
##
##     expand.grid

## Loading required package: IRanges

##
## Attaching package: 'IRanges'

## The following object is masked from 'package:grDevices':
##
##     windows

## Loading required package: GenomicRanges

## Loading required package: GenomeInfoDb

## Loading required package: SummarizedExperiment

## Loading required package: Biobase

## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.

## Loading required package: DelayedArray

## Loading required package: matrixStats

##
## Attaching package: 'matrixStats'

## The following objects are masked from 'package:Biobase':
##
##     anyMissing, rowMedians

## Loading required package: BiocParallel

##
## Attaching package: 'DelayedArray'

## The following objects are masked from 'package:matrixStats':
##
##     colMaxs, colMins, colRanges, rowMaxs, rowMins, rowRanges

## The following objects are masked from 'package:base':
##
##     aperm, apply, rowsum
```

```r
library(limma)
```

```
##
## Attaching package: 'limma'

## The following object is masked from 'package:DESeq2':
##
##     plotMA

## The following object is masked from 'package:BiocGenerics':
##
##     plotMA
```

```r
# LOAD provided functions
source("./script_ejercicios.R")
```

LOAD data/matrixes

```r
# LOAD data/matrixes
count_mat = read.table("./count_mat.txt", header = TRUE, sep = "\t")
FPKM_mat = read.table("./FPKM_mat.txt", header = TRUE, sep = "\t")

# Let's take a LOOK
count_mat[10000:10010, ]
```

```
##                     geneId geneName           type   specific_type d0_r1 d0_r2
## 10000 ENSG00000076351.12  SLC46A1 protein_coding protein_coding    13    10
## 10001 ENSG00000198242.13   RPL23A protein_coding protein_coding   485   452
## 10002 ENSG00000160602.13     NEK8 protein_coding protein_coding    24    18
## 10003 ENSG00000160606.10    TLCD1 protein_coding protein_coding    27    31
## 10004 ENSG00000109046.14     WSB1 protein_coding protein_coding   640   732
## 10005 ENSG00000076604.14    TRAF4 protein_coding protein_coding   300   314
## 10006 ENSG00000167536.13   DHRS13 protein_coding protein_coding     6    12
## 10007  ENSG00000132581.9     SDF2 protein_coding protein_coding    67    67
## 10008 ENSG00000109111.14   SUPT6H protein_coding protein_coding   279   285
## 10009  ENSG00000108255.7   CRYBA1 protein_coding protein_coding     0     0
## 10010 ENSG00000132589.15    FLOT2 protein_coding protein_coding   444   506
##       d0_r4 d14_r7 d14_r8 d14_r9 d14_r10 d28_r8 d28_r14
## 10000    10      9     11     14      17      6      11
## 10001   457    369    381    382     426    182     325
## 10002    15     19     16     18      22     14      19
## 10003    14     30     24     27      21      7      12
## 10004   675   4121   4480   4392    4898   1927    3104
## 10005   288    276    319    311     333    132     215
## 10006     3      7     15      9      11      4       7
## 10007    71     81     81     72     119     70      86
## 10008   252    364    322    336     353    270     405
## 10009     0      7      2      5      13      6       4
## 10010   377    264    313    269     286    180     322
```

```r
FPKM_mat[10000:10010, ]
```

```
##                     geneId geneName           type   specific_type     d0_r1
## 10000 ENSG00000076351.12  SLC46A1 protein_coding protein_coding   7.02413
## 10001 ENSG00000198242.13   RPL23A protein_coding protein_coding 593.58900
## 10002 ENSG00000160602.13     NEK8 protein_coding protein_coding   4.68598
## 10003 ENSG00000160606.10    TLCD1 protein_coding protein_coding  15.70910
## 10004 ENSG00000109046.14     WSB1 protein_coding protein_coding  72.31700
## 10005 ENSG00000076604.14    TRAF4 protein_coding protein_coding  78.97180
## 10006 ENSG00000167536.13   DHRS13 protein_coding protein_coding   8.33748
## 10007  ENSG00000132581.9     SDF2 protein_coding protein_coding  24.28920
## 10008 ENSG00000109111.14   SUPT6H protein_coding protein_coding  13.03820
## 10009  ENSG00000108255.7   CRYBA1 protein_coding protein_coding   0.10000
## 10010 ENSG00000132589.15    FLOT2 protein_coding protein_coding  58.67590
##           d0_r2     d0_r4    d14_r7     d14_r8    d14_r9   d14_r10    d28_r8
## 10000   5.20986   5.60889   4.62893   4.861480   5.84481   4.75972  10.58360
## 10001 586.60800 602.58700 623.14500 545.513000 603.53700 582.93400 451.62900
## 10002   3.03797   1.45348   4.47468   1.581870   2.35365   3.08434   3.32108
## 10003  16.07040   8.43733  21.02480  13.320500  14.78120  13.47100   8.49460
## 10004  79.03390  85.68770 488.05800 507.706000 518.06400 501.30000 387.55700
## 10005  83.54360  84.65800  77.87100  86.547300 108.91000  85.75220  70.10040
## 10006   7.21717   5.54178   8.25337  12.380700  11.62200  13.22590   7.08125
## 10007  15.38630  23.07850  30.58080  27.636300  25.10500  35.05070  36.90310
## 10008  11.52510  11.81470  20.13860  19.587700  20.84800  15.91880  27.40670
## 10009   0.10000   0.10000   3.82556   0.927344   2.45107   5.39699   4.79582
## 10010  63.57290  49.77280  39.34710  41.635700  37.40700  35.29930  43.90280
##         d28_r14
## 10000  12.07500
## 10001 527.98400
## 10002   2.18682
## 10003   6.76776
## 10004 384.04600
## 10005  68.74610
## 10006   7.53924
## 10007  33.17820
## 10008  24.02830
## 10009   2.01370
## 10010  47.59850
```

```
# FORMAT row names
count_mat = count_mat %>% unite("rowNames", geneId:geneName, remove = TRUE)
FPKM_mat = FPKM_mat %>% unite("rowNames", geneId:geneName, remove = TRUE)
row.names(count_mat) = count_mat[, 1]
row.names(FPKM_mat) = FPKM_mat[, 1]

# DROP extra data
drops <- c("rowNames", "type", "specific_type")
count_mat = count_mat[, !(names(count_mat) %in% drops)]
FPKM_mat = FPKM_mat[, !(names(FPKM_mat) %in% drops)]

# Let's take a LOOK
count_mat[10000:10010, ]
```

```
##                            d0_r1 d0_r2 d0_r4 d14_r7 d14_r8 d14_r9 d14_r10
## ENSG00000076351.12_SLC46A1    13    10    10      9     11     14      17
## ENSG00000198242.13_RPL23A    485   452   457    369    381    382     426
## ENSG00000160602.13_NEK8       24    18    15     19     16     18      22
## ENSG00000160606.10_TLCD1      27    31    14     30     24     27      21
## ENSG00000109046.14_WSB1      640   732   675   4121   4480   4392    4898
## ENSG00000076604.14_TRAF4     300   314   288    276    319    311     333
## ENSG00000167536.13_DHRS13      6    12     3      7     15      9      11
## ENSG00000132581.9_SDF2        67    67    71     81     81     72     119
## ENSG00000109111.14_SUPT6H    279   285   252    364    322    336     353
## ENSG00000108255.7_CRYBA1       0     0     0      7      2      5      13
## ENSG00000132589.15_FLOT2     444   506   377    264    313    269     286
##                            d28_r8 d28_r14
## ENSG00000076351.12_SLC46A1      6      11
## ENSG00000198242.13_RPL23A     182     325
## ENSG00000160602.13_NEK8        14      19
## ENSG00000160606.10_TLCD1        7      12
## ENSG00000109046.14_WSB1      1927    3104
## ENSG00000076604.14_TRAF4      132     215
## ENSG00000167536.13_DHRS13       4       7
## ENSG00000132581.9_SDF2         70      86
## ENSG00000109111.14_SUPT6H     270     405
## ENSG00000108255.7_CRYBA1        6       4
## ENSG00000132589.15_FLOT2      180     322
```

```
FPKM_mat[10000:10010, ]
```

```
##                                d0_r1     d0_r2     d0_r4    d14_r7     d14_r8
## ENSG00000076351.12_SLC46A1   7.02413   5.20986   5.60889   4.62893   4.861480
## ENSG00000198242.13_RPL23A  593.58900 586.60800 602.58700 623.14500 545.513000
## ENSG00000160602.13_NEK8      4.68598   3.03797   1.45348   4.47468   1.581870
## ENSG00000160606.10_TLCD1    15.70910  16.07040   8.43733  21.02480  13.320500
## ENSG00000109046.14_WSB1     72.31700  79.03390  85.68770 488.05800 507.706000
## ENSG00000076604.14_TRAF4    78.97180  83.54360  84.65800  77.87100  86.547300
## ENSG00000167536.13_DHRS13    8.33748   7.21717   5.54178   8.25337  12.380700
## ENSG00000132581.9_SDF2      24.28920  15.38630  23.07850  30.58080  27.636300
## ENSG00000109111.14_SUPT6H   13.03820  11.52510  11.81470  20.13860  19.587700
## ENSG00000108255.7_CRYBA1     0.10000   0.10000   0.10000   3.82556   0.927344
## ENSG00000132589.15_FLOT2    58.67590  63.57290  49.77280  39.34710  41.635700
##                               d14_r9   d14_r10    d28_r8   d28_r14
## ENSG00000076351.12_SLC46A1   5.84481   4.75972  10.58360  12.07500
## ENSG00000198242.13_RPL23A  603.53700 582.93400 451.62900 527.98400
## ENSG00000160602.13_NEK8      2.35365   3.08434   3.32108   2.18682
## ENSG00000160606.10_TLCD1    14.78120  13.47100   8.49460   6.76776
## ENSG00000109046.14_WSB1    518.06400 501.30000 387.55700 384.04600
## ENSG00000076604.14_TRAF4   108.91000  85.75220  70.10040  68.74610
## ENSG00000167536.13_DHRS13   11.62200  13.22590   7.08125   7.53924
## ENSG00000132581.9_SDF2      25.10500  35.05070  36.90310  33.17820
## ENSG00000109111.14_SUPT6H   20.84800  15.91880  27.40670  24.02830
## ENSG00000108255.7_CRYBA1     2.45107   5.39699   4.79582   2.01370
## ENSG00000132589.15_FLOT2    37.40700  35.29930  43.90280  47.59850
```

Filtrar genes con baja expresión y con bajos conteos

```r
# FILTRAR genes - conteos

# Para cada gen, contar el número de muestras con mayor a 5 conteos
count_mat_filter <-
    apply(count_mat, 1, function(x)
        length(which(x >= 5)))
table(count_mat_filter)
```

```
## count_mat_filter
##     0     1     2     3     4     5     6     7     8     9
## 19546   882   681   727   474   447   562   640   686 11226
```

```r
# Filtrar genes con menor a 2 muestras con más de 5 conteos
count_mat <- count_mat[which(count_mat_filter >= 2),]

# Let's take a LOOK
dim(count_mat)
```

```
## [1] 15443     9
```

```r
count_mat[10000:10010, ]
```

```
##                            d0_r1 d0_r2 d0_r4 d14_r7 d14_r8 d14_r9 d14_r10
## ENSG00000171121.16_KCNMB3      8     4     2      9      8     10       8
## ENSG00000136521.12_NDUFB5    172   203   168    197    208    213     235
## ENSG00000058056.8_USP13      109   110    96     28     27     32      49
## ENSG00000078081.7_LAMP3       10     3     1      8     10      5       9
## ENSG00000205981.7_DNAJC19     59    63    41     57     42     57      58
## ENSG00000181449.3_SOX2       621   646   569    193    230    183     219
## ENSG00000043093.13_DCUN1D1   112   119   102     61     66     77      72
## ENSG00000172578.11_KLHL6       2     1     0      3      6      1       1
## ENSG00000078070.12_MCCC1      49    60    42     40     35     44      51
## ENSG00000058063.15_ATP11B     89   106   110    106    112    113     144
## ENSG00000053524.12_MCF2L2     10    13    11     16      8     12      17
##                            d28_r8 d28_r14
## ENSG00000171121.16_KCNMB3       1       3
## ENSG00000136521.12_NDUFB5      97     165
## ENSG00000058056.8_USP13        60      90
## ENSG00000078081.7_LAMP3         0       1
## ENSG00000205981.7_DNAJC19      41      58
## ENSG00000181449.3_SOX2        346     551
## ENSG00000043093.13_DCUN1D1     46      74
## ENSG00000172578.11_KLHL6        4      11
## ENSG00000078070.12_MCCC1       45      68
## ENSG00000058063.15_ATP11B      65      90
## ENSG00000053524.12_MCF2L2       9      13
```

```r
# FILTRAR genes - FPKM
# Transformar FPKM a log2
FPKM_mat <- glog(FPKM_mat)

#Para cada gen, contar el número de muestras con expresión mayor a 2
FPKM_mat_filter <-
    apply(FPKM_mat, 1, function(x)
        length(which(x >= 2)))
#Filtrar genes con menor a 2 muestras con expresión mayor a 2
FPKM_mat <- FPKM_mat[which(FPKM_mat_filter >= 2), ]

# Let's take a LOOK
dim(FPKM_mat)
```

```
## [1] 15787     9
```

```
FPKM_mat[10000:10010, ]
```

```
##                                  d0_r1     d0_r2     d0_r4    d14_r7    d14_r8
## ENSG00000163071.10_SPATA18 3.4336046 3.0139406 2.8855050 4.2941129 3.4622242
## ENSG00000128039.10_SRD5A3  4.9215789 4.8700784 5.4974315 4.7287757 4.7169045
## ENSG00000157404.15_KIT     3.2995702 2.9028885 3.1428312 0.1549432 0.9161432
## ENSG00000109255.11_NMU     6.6844867 6.5359997 5.9281671 3.7680237 4.3906178
## ENSG00000145216.15_FIP1L1  5.6672075 5.4511141 5.4976613 5.0626365 5.4839880
## ENSG00000134853.11_PDGFRA  2.7716235 2.4771938 2.2645815 1.5280089 0.7772394
## ENSG00000072201.13_LNX1    3.9186520 3.8251074 3.9531723 3.7407188 3.4208319
## ENSG00000109220.10_CHIC2   3.2826503 3.7588798 3.7557433 4.0469501 3.8226136
## ENSG00000180613.10_GSX2    0.1440301 0.5383846 0.1440301 0.1440301 0.3865912
## ENSG00000109189.12_USP46   4.3440926 4.5863101 4.6525301 3.3123610 3.3352603
## ENSG00000090989.17_EXOC1   4.5357614 4.4216986 4.6760370 5.5289232 4.8022884
##                                 d14_r9   d14_r10    d28_r8   d28_r14
## ENSG00000163071.10_SPATA18 3.8981823 4.0917750 2.418871 1.6158123
## ENSG00000128039.10_SRD5A3  4.4667592 4.6974370 4.155217 4.6443055
## ENSG00000157404.15_KIT     0.1440301 1.3753552 2.633109 2.7498866
## ENSG00000109255.11_NMU     3.7115514 4.2772701 7.023939 6.9509895
## ENSG00000145216.15_FIP1L1  5.9309642 5.4207433 5.720611 5.5529150
## ENSG00000134853.11_PDGFRA  1.3176551 1.7325312 1.099868 0.7497431
## ENSG00000072201.13_LNX1    3.4161771 3.0568830 2.473057 1.7441640
## ENSG00000109220.10_CHIC2   4.0550706 3.9307141 3.297071 4.1573350
## ENSG00000180613.10_GSX2    0.1440301 0.1440301 4.278237 4.5063607
## ENSG00000109189.12_USP46   3.7745207 3.5581715 3.875731 3.7372331
## ENSG00000090989.17_EXOC1   5.2167067 5.6053266 5.726469 5.7753734
```

```
# Let's see what genes share count_mat and FPKM_mat
length(intersect(rownames(count_mat),rownames(FPKM_mat)))
```

```
## [1] 14068
```

Conteos [28 vs 0 días]

```
count_mat_28v0 = count_mat[,-grep('d14_', colnames(count_mat))]

# Conteos
aux_classes = rep(0, times = ncol(count_mat_28v0)) # CLASSIFY "d0_" as 0s
aux_classes[grep(pattern = "d28_", x = colnames(count_mat_28v0))] = 1
aux_classes
```

```
## [1] 0 0 0 1 1
```

```
colnames(count_mat_28v0)
```

```
## [1] "d0_r1"   "d0_r2"   "d0_r4"   "d28_r8"  "d28_r14"
```

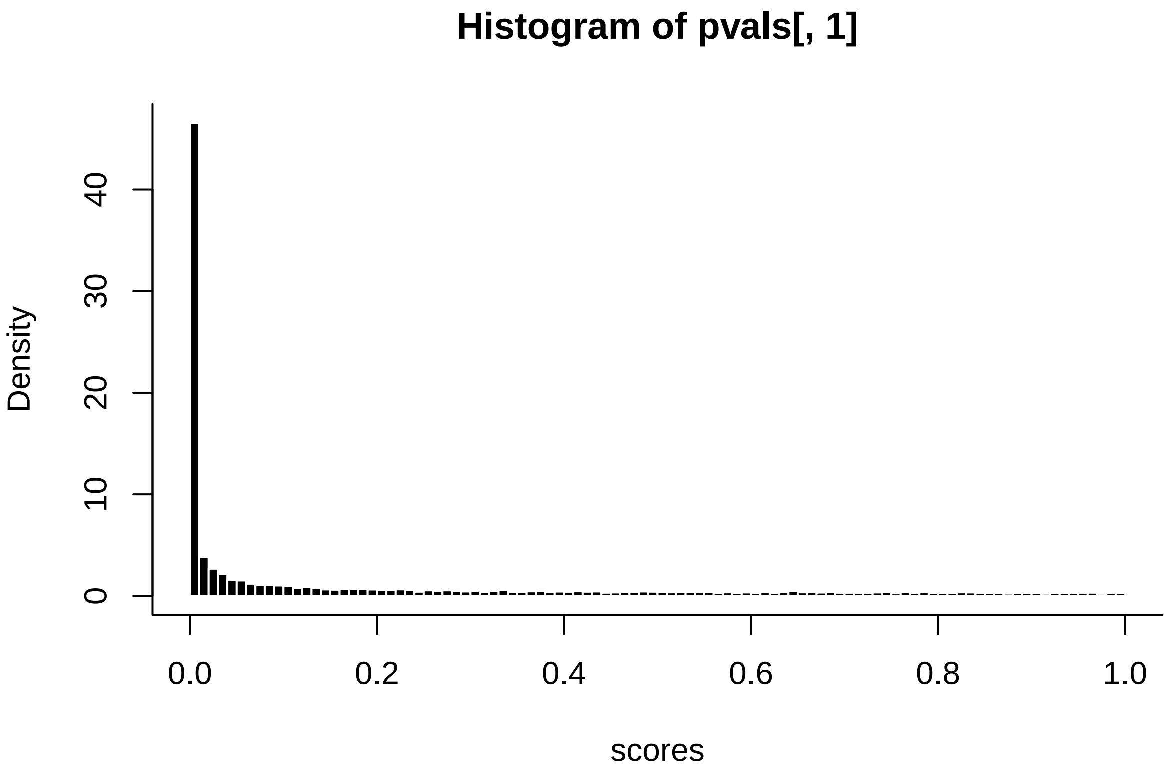
```
count_results_28v0 = DESeq_func(matrix_c = count_mat_28v0, classes_c = aux_classes)
```

```
## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing
```

```
count_results_28v0 = count_results_28v0[order(count_results_28v0$pvalue),]
summary(count_results_28v0)
```

```
##     baseMean        log2FoldChange           lfcSE               stat
##  Min.   :    0.00   Min.   :-11.283296   Min.   :0.02736   Min.   :-50.74271
##  1st Qu.:   14.41   1st Qu.: -0.766020   1st Qu.:0.20092   1st Qu.: -2.47482
##  Median :   49.43   Median :  0.012012   Median :0.31737   Median :  0.04049
##  Mean   :  127.01   Mean   :  0.003875   Mean   :0.49759   Mean   : -0.15351
##  3rd Qu.:  120.35   3rd Qu.:  0.854224   3rd Qu.:0.63628   3rd Qu.:  2.24502
##  Max.   :34898.94   Max.   : 10.403699   Max.   :4.56209   Max.   : 77.22379
##                     NA's   :6            NA's   :6         NA's   :6
##      pvalue              padj
##  Min.   :0.000000   Min.   :0.000000
##  1st Qu.:0.000008   1st Qu.:0.000032
##  Median :0.018872   Median :0.037741
##  Mean   :0.184113   Mean   :0.218235
##  3rd Qu.:0.292569   3rd Qu.:0.390084
##  Max.   :0.999977   Max.   :0.999977
##  NA's   :6          NA's   :6
```

```
# HISTOGRAM
pvals = count_results_28v0["pvalue"]
hist(
    pvals[, 1],
    prob = TRUE,
    col = "black",
    border = "white",
    xlab = "scores",
    breaks = 100
)
box(bty = "l")
# Draw density function (assuming normal dist)
score_mean = mean(pvals[, 1])
score_sd   = sd(pvals[, 1])
curve(
    dnorm(x, mean = score_mean, sd = score_sd),
    add = TRUE,
    col = "red",
    lwd = 2
)
```



```
# Let's take a look to some genes
count_mat_28v0["ENSG00000128567.16_PODXL",]
```

```
##                            d0_r1 d0_r2 d0_r4 d28_r8 d28_r14
## ENSG00000128567.16_PODXL    2585  2998  2581     77     117
```

```
count_mat_28v0["ENSG00000185559.14_DLK1",]
```

```
##                           d0_r1 d0_r2 d0_r4 d28_r8 d28_r14
## ENSG00000185559.14_DLK1      18    20    11   9148   14299
```

Conteos [14 vs 0 días]

```
count_mat_14v0 = count_mat[,-grep('d28_', colnames(count_mat))]

# Conteos
aux_classes = rep(0, times = ncol(count_mat_14v0)) # CLASSIFY "d0_" as 0s
aux_classes[grep(pattern = "d14_", x = colnames(count_mat_14v0))] = 1
aux_classes
```

```
## [1] 0 0 0 1 1 1 1
```

```
colnames(count_mat_14v0)
```

```
## [1] "d0_r1"   "d0_r2"   "d0_r4"   "d14_r7"  "d14_r8"  "d14_r9"  "d14_r10"
```

```
count_results_14v0 = DESeq_func(matrix_c = count_mat_14v0, classes_c = aux_classes)
```

```
## estimating size factors
```

```
## estimating dispersions
```

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

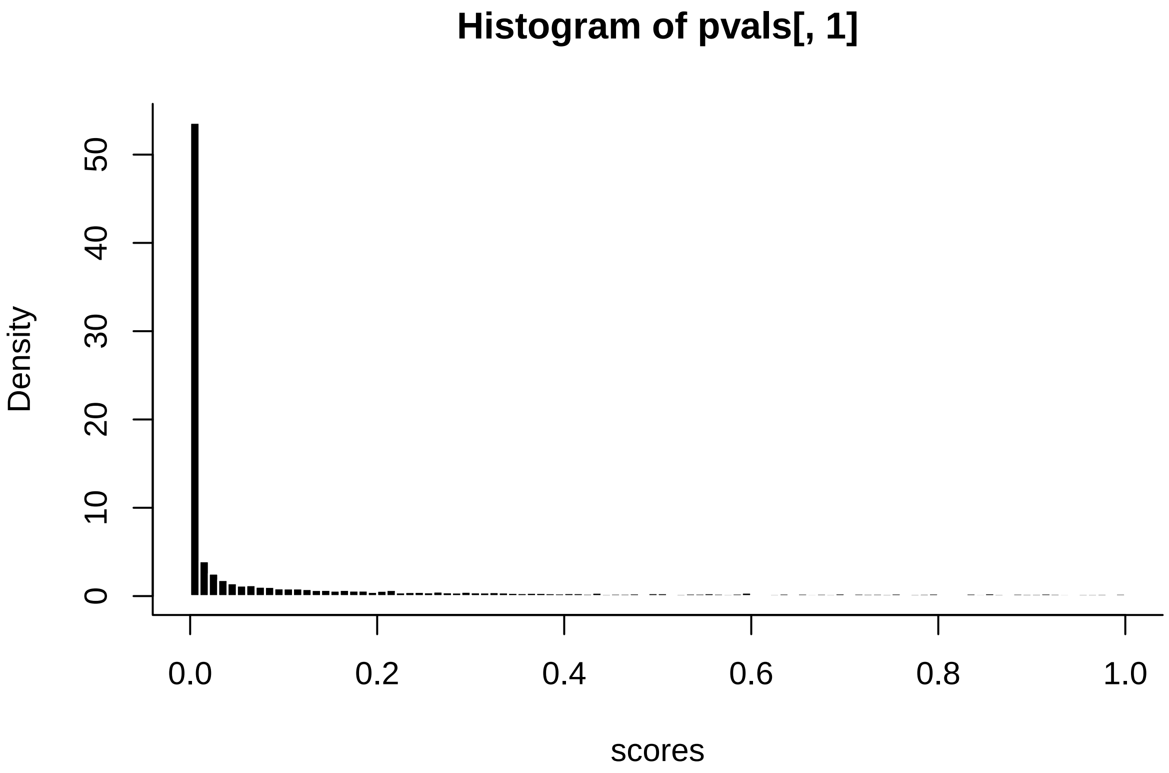
```
## mean-dispersion relationship
```

```
## final dispersion estimates
## fitting model and testing
```

```
count_results_14v0 = count_results_14v0[order(count_results_14v0$pvalue),]
summary(count_results_14v0)
```

```
##     baseMean        log2FoldChange         lfcSE              stat         
##  Min.   :    0.00   Min.   :-9.506879   Min.   :0.02349   Min.   :-68.5253  
##  1st Qu.:   14.51   1st Qu.:-0.651381   1st Qu.:0.13890   1st Qu.: -3.0632  
##  Median :   51.97   Median : 0.034909   Median :0.22005   Median :  0.1444  
##  Mean   :  140.32   Mean   : 0.009636   Mean   :0.34805   Mean   : -0.4737  
##  3rd Qu.:  127.81   3rd Qu.: 0.779743   3rd Qu.:0.43589   3rd Qu.:  2.6507  
##  Max.   :40956.81   Max.   : 9.102615   Max.   :3.78069   Max.   : 99.0111  
##                     NA's   :2           NA's   :2         NA's   :2         
##      pvalue               padj          
##  Min.   :0.000000   Min.   :0.0000000  
##  1st Qu.:0.000000   1st Qu.:0.0000001  
##  Median :0.004802   Median :0.0096040  
##  Mean   :0.149940   Mean   :0.1719656  
##  3rd Qu.:0.186303   3rd Qu.:0.2483857  
##  Max.   :0.999762   Max.   :0.9997616  
##  NA's   :2          NA's   :2          
```

```
# HISTOGRAM
pvals = count_results_14v0["pvalue"]
hist(
    pvals[, 1],
    prob = TRUE,
    col = "black",
    border = "white",
    xlab = "scores",
    breaks = 100
)
box(bty = "l")
# Draw density function (assuming normal dist)
score_mean = mean(pvals[, 1])
score_sd   = sd(pvals[, 1])
curve(
    dnorm(x, mean = score_mean, sd = score_sd),
    add = TRUE,
    col = "red",
    lwd = 2
)
```



```
# Let's take a look to some genes
count_mat_14v0["ENSG00000265992.1_ESRG",]
```

```
##                            d0_r1 d0_r2 d0_r4 d14_r7 d14_r8 d14_r9 d14_r10
## ENSG00000265992.1_ESRG  5258  6429  5532     42     54     55      63
```

```
count_mat_14v0["ENSG00000185559.14_DLK1",]
```

```
##                             d0_r1 d0_r2 d0_r4 d14_r7 d14_r8 d14_r9 d14_r10
## ENSG00000185559.14_DLK1    18    20    11   4704   5550   5124    5854
```

FPKM [28 vs 0 días]

```
FPKM_mat_28v0 = FPKM_mat[,-grep('d14_', colnames(FPKM_mat))]

# FPKM
aux_classes = rep(0, times = ncol(FPKM_mat_28v0)) # CLASSIFY "d0_" as 0s
aux_classes[grep(pattern = "d28_", x = colnames(FPKM_mat_28v0))] = 1
aux_classes
```

```
## [1] 0 0 0 1 1
```

```
colnames(FPKM_mat_28v0)
```

```
## [1] "d0_r1"   "d0_r2"   "d0_r4"   "d28_r8"  "d28_r14"
```

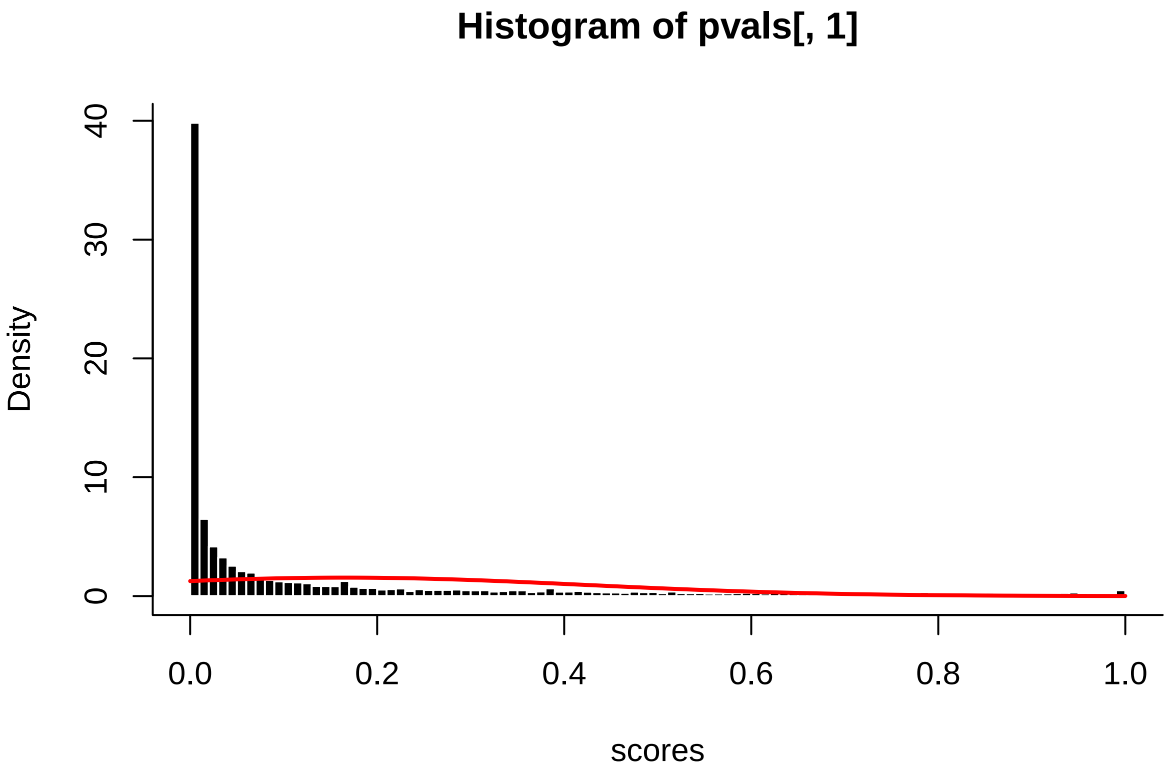
```
fpkm_results_28v0 =
    limma4DS_fdr(
        matrix_e = FPKM_mat_28v0,
        classes_e = aux_classes,
        classes_names = c("d0_", "d28_")
    )
```

```
## toptable() is deprecated and will be removed in the future version of limma. Please use topTable() ins
```

```r
fpkm_results_28v0 = fpkm_results_28v0[order(fpkm_results_28v0$p.value),]
summary(fpkm_results_28v0)
```

```
##       d0_               d28_                FC               p.value
##  Min.   : 0.144   Min.   : 0.144   Min.   :-8.8251   Min.   :0.00000
##  1st Qu.: 2.793   1st Qu.: 2.447   1st Qu.:-0.5499   1st Qu.:0.00194
##  Median : 4.235   Median : 4.009   Median : 0.1784   Median :0.02871
##  Mean   : 4.188   Mean   : 4.031   Mean   : 0.1574   Mean   :0.16522
##  3rd Qu.: 5.503   3rd Qu.: 5.438   3rd Qu.: 0.9620   3rd Qu.:0.21941
##  Max.   :13.203   Max.   :13.138   Max.   :10.0295   Max.   :1.00000
##     q.value
##  Min.   :0.0000665
##  1st Qu.:0.0077607
##  Median :0.0574105
##  Mean   :0.1974394
##  3rd Qu.:0.2925361
##  Max.   :1.0000000
```

```r
# HISTOGRAM
pvals = fpkm_results_28v0["p.value"]
hist(
    pvals[, 1],
    prob = TRUE,
    col = "black",
    border = "white",
    xlab = "scores",
    breaks = 100
)
box(bty = "l")
# Draw density function (assuming normal dist)
score_mean = mean(pvals[, 1])
score_sd   = sd(pvals[, 1])
curve(
    dnorm(x, mean = score_mean, sd = score_sd),
    add = TRUE,
    col = "red",
    lwd = 2
)
```



```
# Let's take a look to some genes
FPKM_mat_28v0["ENSG00000128567.16_PODXL",]
```

```
##                               d0_r1    d0_r2    d0_r4   d28_r8  d28_r14
## ENSG00000128567.16_PODXL 8.229662 8.286831 8.265865 3.880638 4.236951
```

```
FPKM_mat_28v0["ENSG00000185559.14_DLK1",]
```

```
##                              d0_r1   d0_r2    d0_r4   d28_r8  d28_r14
## ENSG00000185559.14_DLK1 3.451555 3.35383 2.714869 13.19995 13.20579
```

FPKM [14 vs 0 días]

```
FPKM_mat_14v0 = FPKM_mat[,-grep('d28_', colnames(FPKM_mat))]

# FPKM
aux_classes = rep(0, times = ncol(FPKM_mat_14v0)) # CLASSIFY "d0_" as 0s
aux_classes[grep(pattern = "d14_", x = colnames(FPKM_mat_14v0))] = 1
aux_classes
```

```
## [1] 0 0 0 1 1 1 1
```

```
colnames(FPKM_mat_14v0)
```

```
## [1] "d0_r1"   "d0_r2"   "d0_r4"   "d14_r7"  "d14_r8"  "d14_r9"  "d14_r10"
```

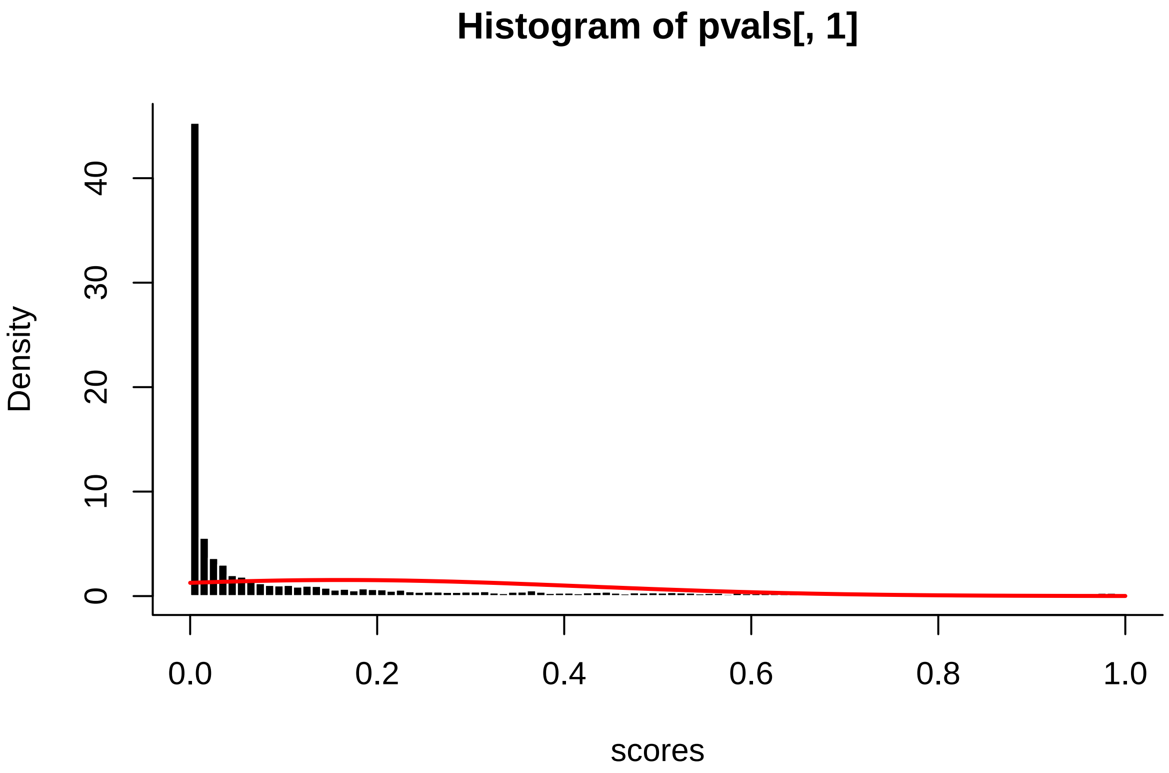
```
fpkm_results_14v0 =
    limma4DS_fdr(
        matrix_e = FPKM_mat_14v0,
        classes_e = aux_classes,
        classes_names = c("d0_", "d14_")
    )
```

```
## toptable() is deprecated and will be removed in the future version of limma. Please use topTable() ins
```

```
fpkm_results_14v0 = fpkm_results_14v0[order(fpkm_results_14v0$p.value),]
summary(fpkm_results_14v0)
```

```
##       d0_              d14_              FC               p.value
##  Min.   : 0.144   Min.   : 0.144   Min.   :-8.37991   Min.   :0.0000000
##  1st Qu.: 2.593   1st Qu.: 2.447   1st Qu.:-0.51672   1st Qu.:0.0005287
##  Median : 4.070   Median : 4.009   Median : 0.12407   Median :0.0180993
##  Mean   : 4.095   Mean   : 4.031   Mean   : 0.06439   Mean   :0.1613872
##  3rd Qu.: 5.404   3rd Qu.: 5.438   3rd Qu.: 0.76548   3rd Qu.:0.2099093
##  Max.   :13.460   Max.   :13.138   Max.   : 8.45595   Max.   :1.0000000
##     q.value
##  Min.   :0.0000003
##  1st Qu.:0.0021142
##  Median :0.0361964
##  Mean   :0.1893386
##  3rd Qu.:0.2798635
##  Max.   :1.0000000
```

```
# HISTOGRAM
pvals = fpkm_results_14v0["p.value"]
hist(
    pvals[, 1],
    prob = TRUE,
    col = "black",
    border = "white",
    xlab = "scores",
    breaks = 100
)
box(bty = "l")
# Draw density function (assuming normal dist)
score_mean = mean(pvals[, 1])
score_sd   = sd(pvals[, 1])
curve(
    dnorm(x, mean = score_mean, sd = score_sd),
    add = TRUE,
    col = "red",
    lwd = 2
)
```



```
# Let's take a look to some genes
FPKM_mat_14v0["ENSG00000265992.1_ESRG",]
```

```
##                             d0_r1    d0_r2    d0_r4   d14_r7   d14_r8   d14_r9
## ENSG00000265992.1_ESRG 9.978663 10.05719 10.08886 3.084105 3.475052 3.134707
##                           d14_r10
## ENSG00000265992.1_ESRG 3.681465
```

```
FPKM_mat_14v0["ENSG00000185559.14_DLK1",]
```

```
##                              d0_r1   d0_r2    d0_r4   d14_r7   d14_r8   d14_r9
## ENSG00000185559.14_DLK1 3.451555 3.35383 2.714869 11.58105 11.69617 11.63993
##                            d14_r10
## ENSG00000185559.14_DLK1 11.60031
```

Comparar si obtiene los mismos genes diferencialmente expresados.

```
# Filtrar genes con pvalue menor a 0.05 (the most significat)
pv = 0.05

# For COUNTS
count_results_28v0_de = count_results_28v0[which(count_results_28v0$pvalue <= pv),]
fpkm_results_28v0_de = fpkm_results_28v0[which(fpkm_results_28v0$p.value <= pv),]

# For FPKM
count_results_14v0_de = count_results_14v0[which(count_results_14v0$pvalue <= pv),]
fpkm_results_14v0_de = fpkm_results_14v0[which(fpkm_results_14v0$p.value <= pv),]
```

```
# No. of differential expressed genes by DeSeq2 (28 vs 0 días)
length(rownames(count_results_28v0_de))
```

```
## [1] 8765
```

```
# No. of differential expressed genes by LIMMA (28 vs 0 días)
length(rownames(fpkm_results_28v0_de))
```

```
## [1] 8891
```

```
# No. of differential expressed genes by DeSeq2 and LIMMA (28 vs 0 días)
length(intersect(rownames(count_results_28v0_de),rownames(fpkm_results_28v0_de)))
```

```
## [1] 6865
```

```
# No. of differential expressed genes by DeSeq2 (28 vs 0 días)
length(rownames(count_results_14v0_de))
```

```
## [1] 9784
```

```
# No. of differential expressed genes by LIMMA (28 vs 0 días)
length(rownames(fpkm_results_14v0_de))
```

```
## [1] 9400
```

```
# No. of differential expressed genes by DeSeq2 and LIMMA (28 vs 0 días)
length(intersect(rownames(count_results_14v0_de),rownames(fpkm_results_14v0_de)))
```

```
## [1] 7824
```

```
# Enable Warning messages
options(warn = 0)
```