Statistical Tests

Antonio Osamu Katagiri Tanaka - A01212611@itesm.mx

March 27, 2020

```
# 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)

# Load Libs
library(DT)
library(samr)
library(affy)
```

```
## 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, colMeans,
##     colnames, colSums, dirname, do.call, duplicated, eval, evalq,
##     Filter, Find, get, grep, grepl, intersect, is.unsorted, lapply,
##     lengths, Map, mapply, match, mget, order, paste, pmax, pmax.int,
##     pmin, pmin.int, Position, rank, rbind, Reduce, rowMeans, rownames,
##     rowSums, sapply, setdiff, sort, table, tapply, union, unique,
##     unsplit, which, which.max, which.min
```

```
## 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")'.
```

```
library(GEOquery)
```

```
## Setting options('download.file.method.GEOquery'='auto')
```

```
## Setting options('GEOquery.inmemory.gpl'=FALSE)
```

```r
library(biomaRt)
library(Biobase)
library(limma)
```

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

```r
library(affy)
library(siggenes)
```

```
## Loading required package: multtest
## Loading required package: splines
```

```r
library(PerformanceAnalytics)
```

```
## Loading required package: xts
## Loading required package: zoo
##
## Attaching package: 'zoo'
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
##
## Attaching package: 'PerformanceAnalytics'
## The following object is masked from 'package:graphics':
##
##     legend
```

```r
source("./statistical_tests_lib.R")
```

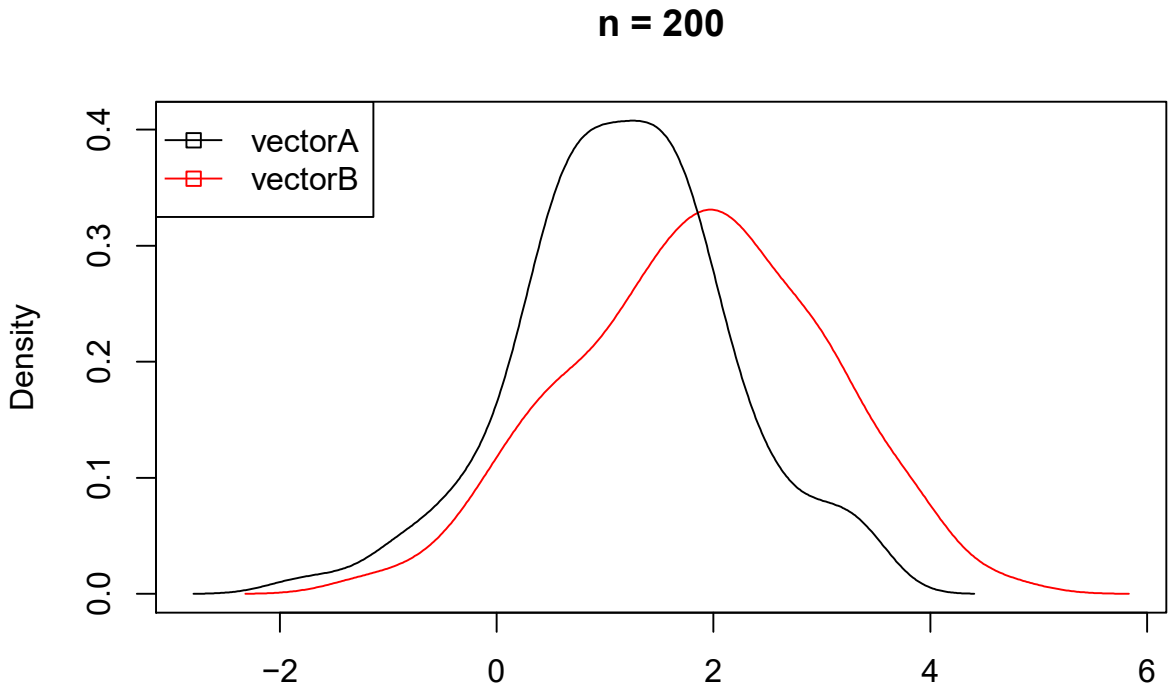
---

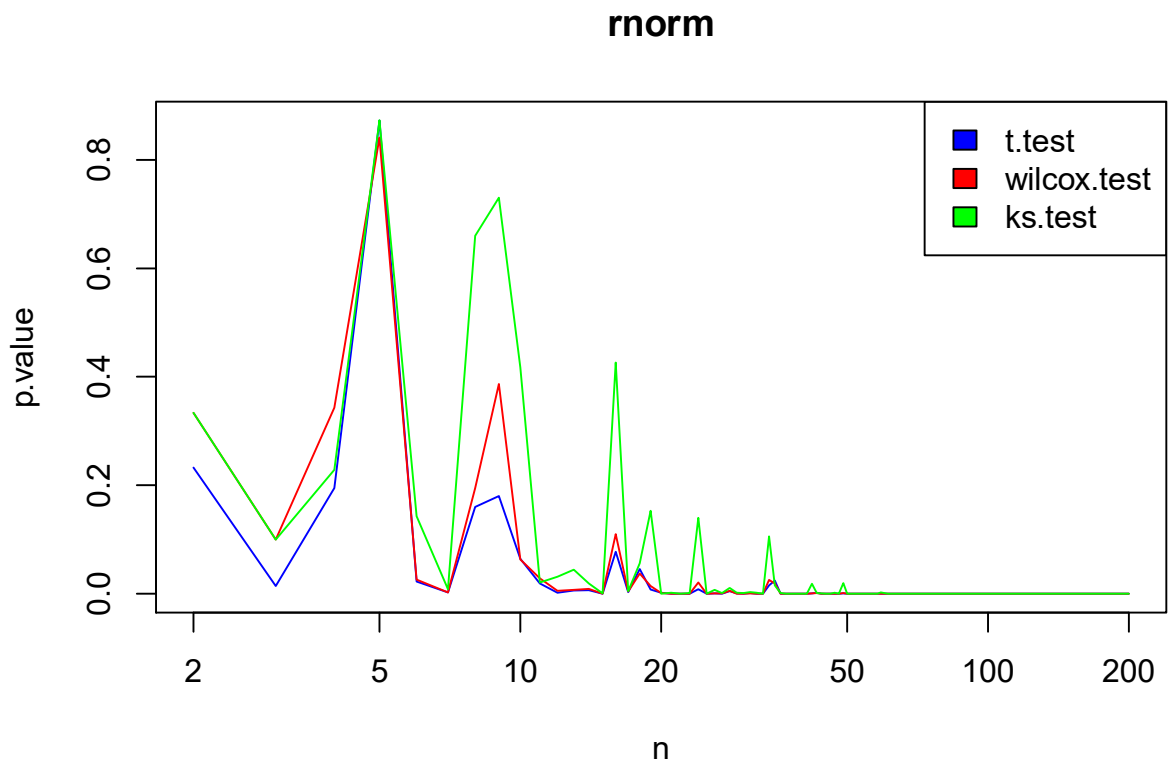
1) Obtener p-values con un t-test, Wilcoxon y Kolmogorov para:

A) 2 vectores de datos con distribución normal (rnorm) y diferente media, para número de muestras n= 2. . . . . 20. Graficar los resultados y comparar.

```r
getPvalues <- function(dist) {
  ns = 2:200
  ttest_pval = c()
  wtest_pval = c()
  ktest_pval = c()

  for (n in ns) {
    if (dist == "rnorm") {
      vectorA = rnorm(n = n, mean = 1, sd = 1)
      vectorB = rnorm(n = n, mean = 2, sd = 1)
    } else {
      vectorA = runif(n, -1, 1)
      vectorB = runif(n, -1, 1)
    }

    if (n == max(ns)) {
      plot.densities(vectorA, vectorB, main = paste0("n = ", n))
    }
```

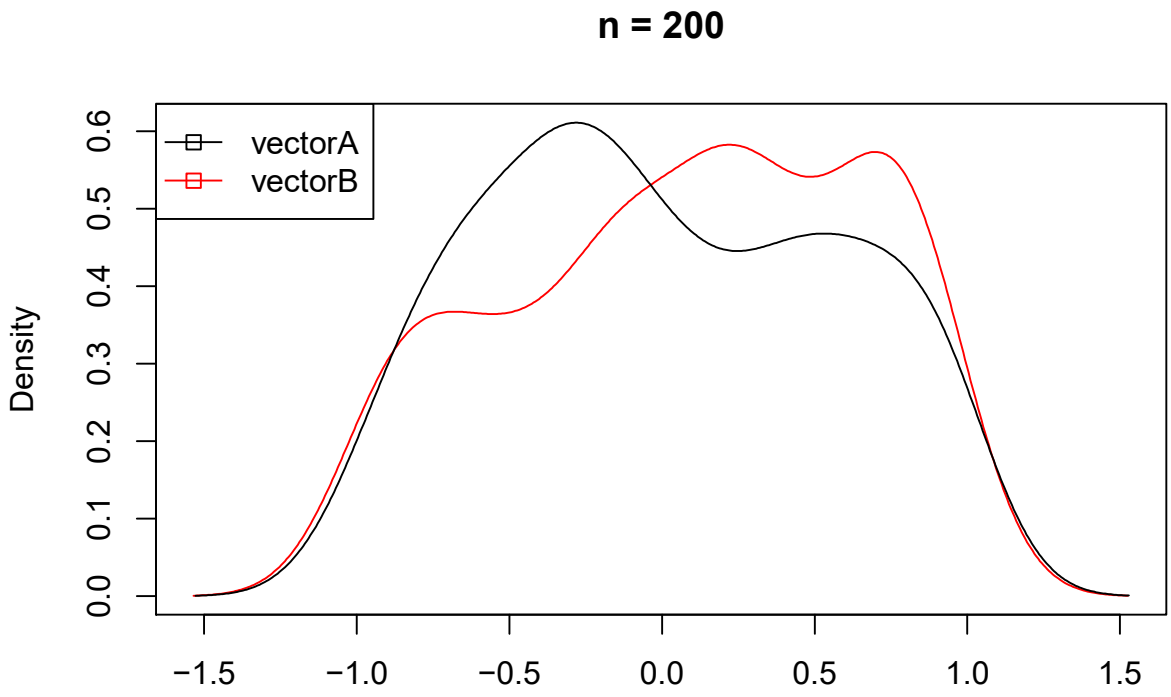
```
    ttest = t.test(vectorA, vectorB)
    ttest_pval = c(ttest_pval, ttest$p.value)
    #print(ttest$p.value)

    wtest = wilcox.test(vectorA, vectorB)
    wtest_pval = c(wtest_pval, wtest$p.value)
    #print(wtest$p.value)

    ktest = ks.test(vectorA, vectorB)
    ktest_pval = c(ktest_pval, ktest$p.value)
    #print(ktest$p.value)
  }

  if (dist == "rnorm") {
    plot(
      ns,
      ttest_pval,
      main = "rnorm",
      xlab = "n",
      ylab = "p.value",
      type = "l",
      col = "blue",
      log = "x"
    )
  } else {
    plot(
      ns,
      ttest_pval,
      main = "runif",
      xlab = "n",
      ylab = "p.value",
      type = "l",
      col = "blue"
    )
  }

  lines(ns, wtest_pval,
        col = "red")

  lines(ns, ktest_pval,
        col = "green")

  legend("topright",
         c("t.test", "wilcox.test", "ks.test"),
         fill = c("blue", "red", "green"))
}

getPvalues("rnorm")
```

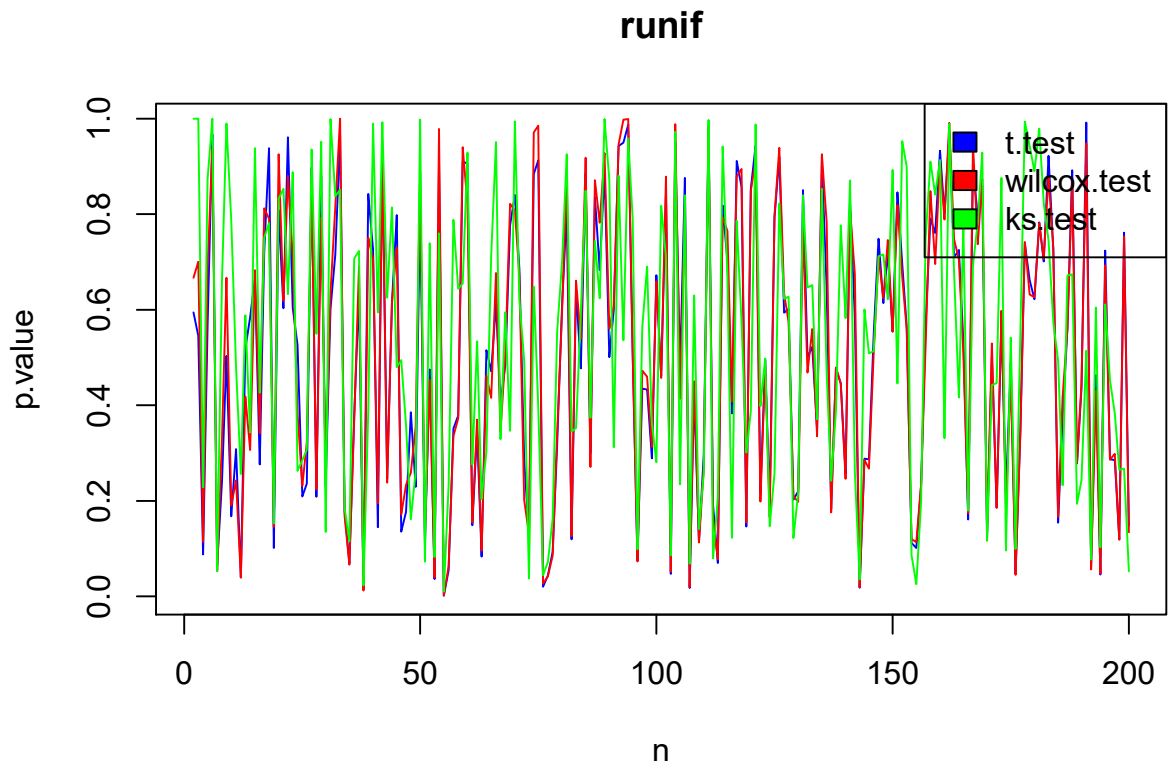




B) Repetir (A) con 2 vectores de datos generados con la función runif.

```
getPvalues("runif")
```


n = 200
vectorA
vectorB
Density
0.0
0.1
0.2
0.3
0.4
0.5
0.6
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5


runif
t.test
wilcox.test
ks.test
p.value
0.0
0.2
0.4
0.6
0.8
1.0
0
50
100
150
200
n

2) Obtener p-values con un t-test, Wilcoxon, Kolmogorov y SAM para una base de datos GEO de su elección.

```
calPvalues <- function(data.matrix, lvls) {
  pval = de.test(
    x = data.matrix,
    classes = lvls,
    test = c("ttest", "kolmogorov", "wilcoxon")
  )
  data.matrix_sam <- sam(data.matrix,
                         c(rep(1, dim(data.matrix)[2])),
                         method = "d.stat",
                         gene.names = colnames(data.matrix))

  return(
    list(
      "ttest_pval" = pval$ttest,
      "wtest_pval" = pval$kolmogorov,
      "ktest_pval" = pval$wilcoxon,
      "sam_pval"   = unname(data.matrix_sam@p.value)
    )
  )
}

plt_pval <- function(pval) {
  for (i in 1:length(pval)) {
    ns = 1:length(pval[[i]])
    plot(
      ns,
      pval[[i]],
      main = "runif",
      xlab = "",
      ylab = "p.value",
      #log = "y",
      col = "blue",
      type = "p"
    )
  }
}
```

Dataset details

Title: Lung response to highly virulent or low virulent influenza A H3N2 virus infection: time course

Summary: Analysis of total lung from BALB/c females infected with highly virulent influenza A (HVI) or low virulent influenza A (LVI), up to 96 hrs following infection. Results provide insight into the molecular mechanisms underlying the host pulmonary responses to HVI and LVI infections.

Organism: Mus musculus

Platform: GPL6103: Illumina mouseRef-8 v1.1 expression beadchip

Citation: Ivan FX, Rajapakse JC, Welsch RE, Rozen SG et al. Differential pulmonary transcriptomic profiles in murine lungs infected with low and highly virulent influenza H3N2 viruses reveal dysregulation of TREM1 signaling, cytokines, and chemokines. Funct Integr Genomics 2012 Mar;12(1):105-17. PMID: 21874528

Reference Series: GSE55994

Sample count: 18

Value type: transformed count

Series published: 2014/03/19

Dataset taken from: ncbi.nlm.nih.gov

Let's compare influenza A H3N2 infection levels. (from GSE55994)

```
# Download GDS file, put it in the current directory, and load it
gds5159 <- getGEO('GDS5159', destdir='./NCBI_GEO')
```

```
## Using locally cached version of GDS5159 found here:
## ./NCBI_GEO/GDS5159.soft.gz
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   ID_REF = col_character(),
##   IDENTIFIER = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

```
eset <- GDS2eSet(gds5159, do.log2 = TRUE)  # Convert the data to ESET object
```

```
## File stored at:
```

```
## C:\Users\oskat\AppData\Local\Temp\RtmpcDxcGJ/GPL6103.annot.gz
```

```
pData(eset)$infection         # Let's check at the infections
```

```
##  [1] mock                          mock
##  [3] mock                          mock
##  [5] mock                          mock
##  [7] low virulent influenza A      low virulent influenza A
##  [9] low virulent influenza A      low virulent influenza A
## [11] low virulent influenza A      low virulent influenza A
## [13] highly virulent influenza A highly virulent influenza A
## [15] highly virulent influenza A highly virulent influenza A
## [17] highly virulent influenza A highly virulent influenza A
## Levels: highly virulent influenza A low virulent influenza A mock
```

```
table(pData(eset)$infection)  # No. of samples in each infection
```

```
##
## highly virulent influenza A    low virulent influenza A
##                           6                           6
##                        mock
##                           6
```

Filter columns associated with 'highly virulent influenza A' and 'mock' values.

```
selected <- grep("mock|highly virulent influenza A", pData(eset)$infection) # Select indexes
pData(eset)$infection[selected] # Check selection
```

```
##  [1] mock                          mock
##  [3] mock                          mock
##  [5] mock                          mock
##  [7] highly virulent influenza A highly virulent influenza A
##  [9] highly virulent influenza A highly virulent influenza A
## [11] highly virulent influenza A highly virulent influenza A
## Levels: highly virulent influenza A low virulent influenza A mock
```

'highly virulent influenza A' as '1's and 'mock' '2's

```
y <- c(rep(1, 6), rep(2, 6))  # Vector of levels ID
y
```

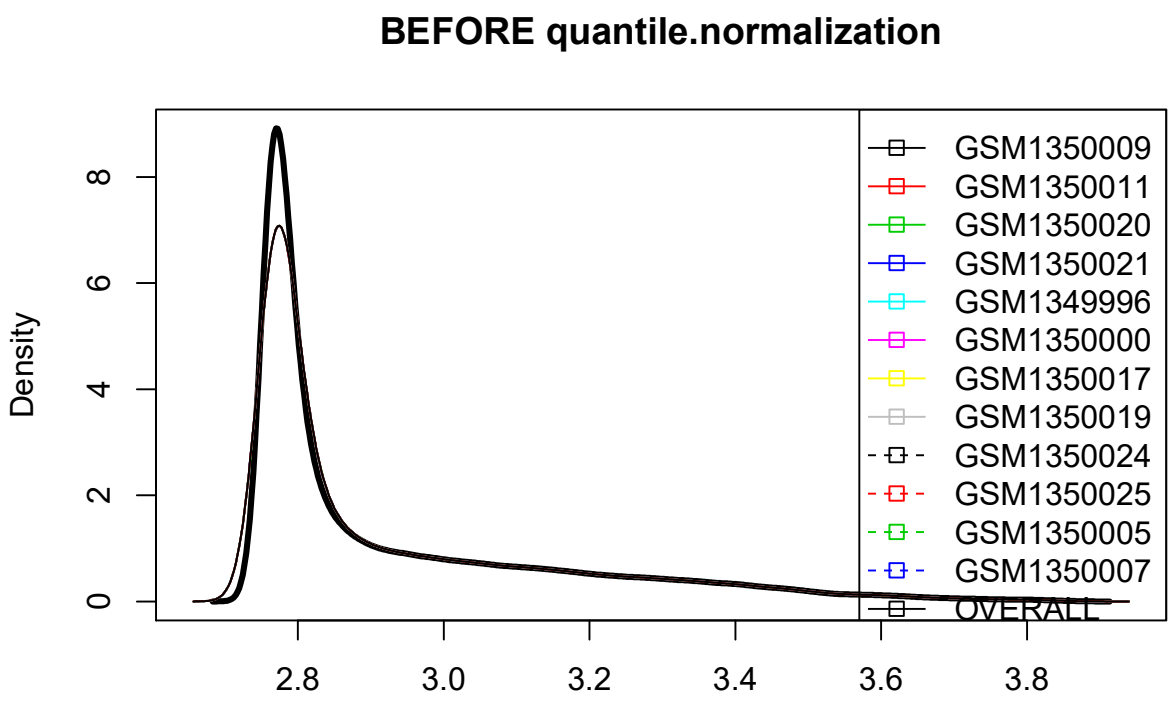
```
##  [1] 1 1 1 1 1 1 2 2 2 2 2 2
```

A) Verifique si necesita normalizar los datos con quantile normalization. Obtenga p-values antes y después de normalizar.

Filter the expression matrix according to selected, then quantile normalize

[There is small-to-large variability within groups and small variability across groups, so, let's use quantile normalization]

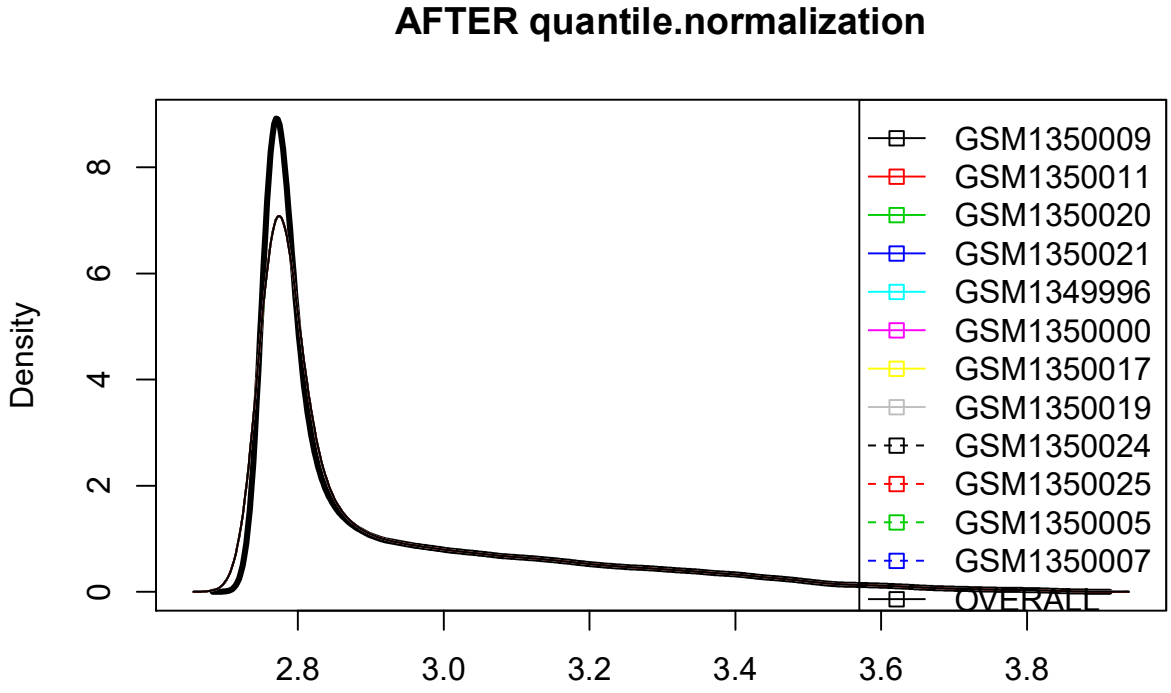
```
# Get a subset of expression values of the selected samples
exprs.selected <- exprs(eset)[, selected]
plot.densities(exprs.selected, main = "BEFORE quantile.normalization", legend.pos="topright")
```




```
# p values before quantile normalize
pval = calPvalues(exprs.selected,y)
```

```
## We're doing 4096 complete permutations
## and randomly select 100 of them.
```

```
exprs.selected.q <- normalizeQuantiles(exprs.selected)  # Quantile normalize the data
plot.densities(exprs.selected.q, main = "AFTER quantile.normalization", legend.pos="topright")
```

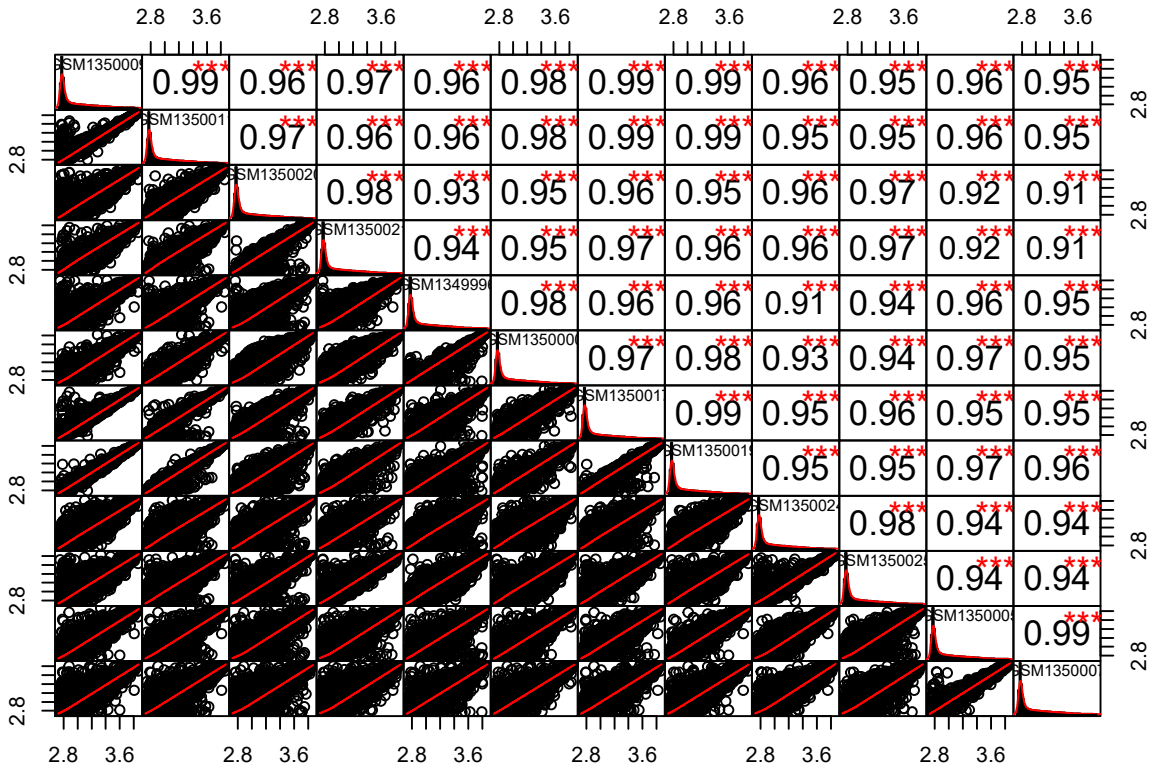



```
# p values after quantile normalize
pval.q = calPvalues(exprs.selected.q,y)
```

```
## We're doing 4096 complete permutations
## and randomly select 100 of them.
```

B) Compare los p-values y grafique. Comente sobre la correlación entre las distintas pruebas estadísticas y la cantidad de genes significativos.

```
chart.Correlation(
  exprs.selected.q,
  histogram = TRUE
)
```



In the above plot:

- The distribution of each variable is shown on the diagonal.
- On the bottom of the diagonal : the bivariate scatter plots with a fitted line are displayed
- On the top of the diagonal : the value of the correlation plus the significance level as stars
- Each significance level is associated to a symbol : p-values(0, 0.001, 0.01, 0.05, 0.1, 1) <=> symbols(“ ”, “”, “”, “.”, " “)

Let's prepare the data that required to run **samr()**

```
# Get row names = gene names or IDs
genenames <- rownames(exprs(eset))
data <- list(
  x = exprs.selected.q,
  y = y,
  geneid = genenames,
  genenames = genenames,
  logged2 = TRUE
)
```

Refer to help(samr) for more info.

```
# Get object names from samr.obj
samr.obj <-
  samr(data, resp.type = "Two class unpaired", nperms = 100)
```

```
## perm= 1
## perm= 2
## perm= 3
## perm= 4
## perm= 5
## perm= 6
## perm= 7
## perm= 8
## perm= 9
## perm= 10
## perm= 11
```

```
## perm= 12
## perm= 13
## perm= 14
## perm= 15
## perm= 16
## perm= 17
## perm= 18
## perm= 19
## perm= 20
## perm= 21
## perm= 22
## perm= 23
## perm= 24
## perm= 25
## perm= 26
## perm= 27
## perm= 28
## perm= 29
## perm= 30
## perm= 31
## perm= 32
## perm= 33
## perm= 34
## perm= 35
## perm= 36
## perm= 37
## perm= 38
## perm= 39
## perm= 40
## perm= 41
## perm= 42
## perm= 43
## perm= 44
## perm= 45
## perm= 46
## perm= 47
## perm= 48
## perm= 49
## perm= 50
## perm= 51
## perm= 52
## perm= 53
## perm= 54
## perm= 55
## perm= 56
## perm= 57
## perm= 58
## perm= 59
## perm= 60
## perm= 61
## perm= 62
## perm= 63
## perm= 64
## perm= 65
## perm= 66
## perm= 67
## perm= 68
## perm= 69
## perm= 70
## perm= 71
## perm= 72
## perm= 73
```

```
## perm= 74
## perm= 75
## perm= 76
## perm= 77
## perm= 78
## perm= 79
## perm= 80
## perm= 81
## perm= 82
## perm= 83
## perm= 84
## perm= 85
## perm= 86
## perm= 87
## perm= 88
## perm= 89
## perm= 90
## perm= 91
## perm= 92
## perm= 93
## perm= 94
## perm= 95
## perm= 96
## perm= 97
## perm= 98
## perm= 99
## perm= 100
```

```
names(samr.obj)
```

```
##  [1] "n"                    "x"                    "xresamp"
##  [4] "y"                    "argy"                 "censoring.status"
##  [7] "testStatistic"        "nperms"               "nperms.act"
## [10] "tt"                   "numer"                "sd"
## [13] "sd.internal"          "s0"                   "s0.perc"
## [16] "evo"                  "perms"                "permsy"
## [19] "nresamp"              "nresamp.perm"         "all.perms.flag"
## [22] "ttstar"               "ttstar0"              "eigengene"
## [25] "eigengene.number"     "pi0"                  "foldchange"
## [28] "foldchange.star"      "sdstar.keep"          "resp.type"
## [31] "resp.type.arg"        "assay.type"           "stand.contrasts"
## [34] "stand.contrasts.star" "stand.contrasts.95"   "depth"
## [37] "call"
```

Compute thresholds for different deltas

```
delta <- 1.2 # define the delta
delta.table <- samr.compute.delta.table(samr.obj, min.foldchange = delta)
```

```
##
## Computing delta table
## 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
## 31
## 32
## 33
## 34
## 35
## 36
## 37
## 38
## 39
## 40
## 41
## 42
## 43
## 44
## 45
## 46
## 47
## 48
## 49
## 50
```

```
# Look at the whole delta table
datatable(delta.table)
```

Show 10 entries Search:

delta	# med false pos	90th perc false pos	# called	median FDR	90th perc FDR	cutlo	cuthi
0	1.87949457603705	12.4046642018446	54	0.0348054551117973	0.229716003737862	-1.74810434357576	1.6775552443206
0.00102499764432707	1.87949457603705	12.4046642018446	54	0.0348054551117973	0.229716003737862	-1.74810434357576	1.6775552443206
0.00409999057730828	1.87949457603705	12.4046642018446	54	0.0348054551117973	0.229716003737862	-1.74810434357576	1.6775552443206
0.00922497879894364	1.87949457603705	12.4046642018446	54	0.0348054551117973	0.229716003737862	-1.74810434357576	1.6775552443206
0.0163999623092331	1.87949457603705	12.4046642018446	54	0.0348054551117973	0.229716003737862	-1.74810434357576	1.6775552443206
0.0256249411081768	1.87949457603705	12.4046642018446	54	0.0348054551117973	0.229716003737862	-1.74810434357576	1.6775552443206
0.0368999151957746	1.87949457603705	12.4046642018446	54	0.0348054551117973	0.229716003737862	-1.74810434357576	1.6775552443206
0.0502248845720265	1.87949457603705	12.4046642018446	54	0.0348054551117973	0.229716003737862	-1.74810434357576	1.6775552443206
0.0655998492369326	1.87949457603705	12.4046642018446	54	0.0348054551117973	0.229716003737862	-1.74810434357576	1.6775552443206
0.0830248091904928	1.87949457603705	12.4046642018446	54	0.0348054551117973	0.229716003737862	-1.74810434357576	1.6775552443206

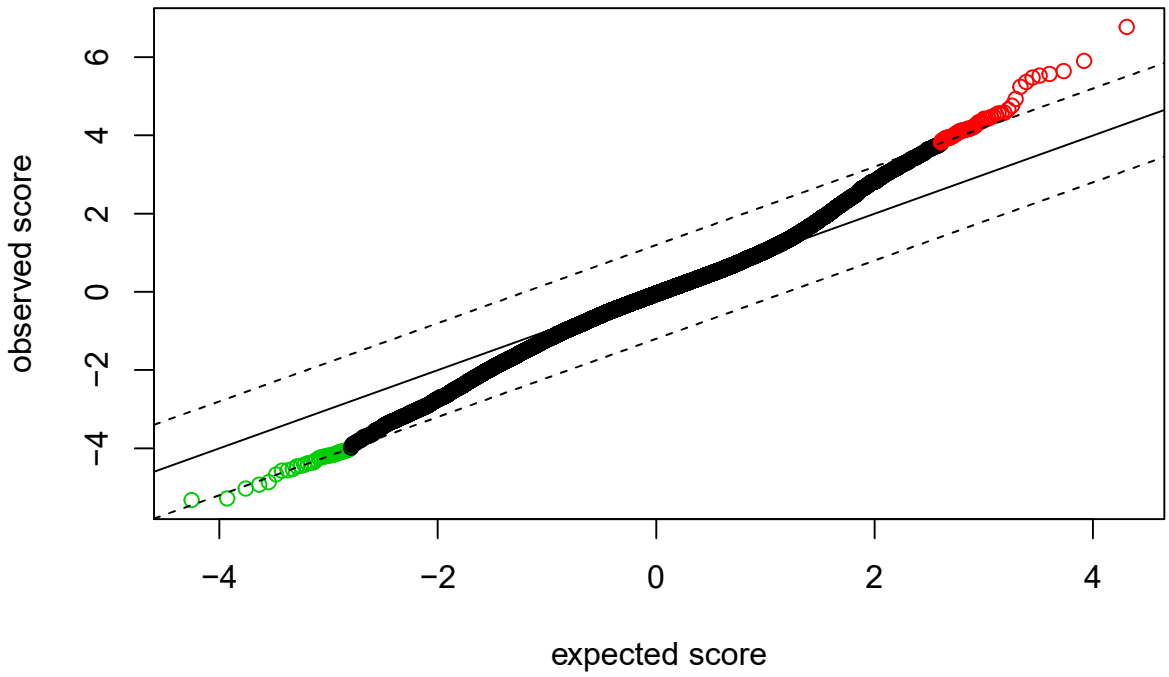
Showing 1 to 10 of 50 entries Previous 1 2 3 4 5 Next

Let's select delta with median FDR <10%.

```
# Check delta related to median FDR of 0.1
delta.table[delta.table[, "median FDR"] < 0.1,][1,]
```

```
##               delta     # med false pos 90th perc false pos            # called
##          0.00000000          1.87949458         12.40466420         54.00000000
##          median FDR       90th perc FDR                cutlo               cuthi
##          0.03480546          0.22971600         -1.74810434          1.67755524
```

```
samr.plot(samr.obj, delta)  # Check SAM plot
```



B) Comente sobre la cantidad de genes significativos.

Get significant genes

```
siggenes.table <- samr.compute.siggenes.table(samr.obj,
                                              delta,
                                              data,
                                              delta.table,
                                              min.foldchange = delta)

datatable(siggenes.table$genes.lo)  # Check how table with the results look like
```

Show 10 entries Search:

Row	Gene ID	Gene Name	Score(d)	Numerator(r)	Denominator(s+s0)	Fold Change	q-value(%)
No data available in table							

Showing 0 to 0 of 0 entries Previous Next

```
datatable(siggenes.table$genes.up)  # Check how table with the results look like
```

Show 10 entries Search:

Row	Gene ID	Gene Name	Score(d)	Numerator(r)	Denominator(s+s0)	Fold Change	q-value(%)
16314	ILMN_2763243	ILMN_2763243	6.77045829674065	0.500645010330988	0.0739455127538416	1.41484598036509	0
16859	ILMN_2772632	ILMN_2772632	5.90318005355945	0.453304298362458	0.0767898478870094	1.36917257338006	0
20749	ILMN_2769917	ILMN_2769917	5.36573836049137	0.460717358143325	0.0858628071647411	1.37622595543983	0
17788	ILMN_2591265	ILMN_2591265	5.24181740583111	0.28403474372971	0.05418631015528	1.21759534107198	0
24526	ILMN_1246800	ILMN_1246800	4.7570649235383	0.294288593643489	0.0618634806069869	1.22628013798711	0
13079	ILMN_2702303	ILMN_2702303	4.46658048188941	0.286387767044293	0.064117901424928	1.21958284905855	0
17953	ILMN_2771766	ILMN_2771766	4.42236050906219	0.292311422203738	0.066098505900806	1.2246007079222	0
1835	ILMN_2595732	ILMN_2595732	4.42122076791449	0.282670569382864	0.0639349591936801	1.21644455921938	0
11691	ILMN_2771176	ILMN_2771176	4.3429845676493	0.488089114442603	0.112385643292025	1.40258588488981	0
17748	ILMN_1225204	ILMN_1225204	4.24130709678923	0.314798789689812	0.0742221165565027	1.24383816513269	0

Showing 1 to 10 of 18 entries Previous 1 2 Next

---

3) Repetir el ejercicio (2) pero con una base de datos de GEO con menos o más muestras, según sea el caso.

Let's compare lung cells related gene expressions due to different bacterial strains. (from GDS858=

```
# Download GDS file, put it in the current directory, and load it
gds858 <- getGEO('GDS858', destdir='./NCBI_GEO')
```

```
## Using locally cached version of GDS858 found here:
## ./NCBI_GEO/GDS858.soft.gz
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   ID_REF = col_character(),
##   IDENTIFIER = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

```
eset <- GDS2eSet(gds858, do.log2 = TRUE)  # Convert the data to ESET object
```

```
## File stored at:
```

```
## C:\Users\oskat\AppData\Local\Temp\RtmpcDxcGJ/GPL96.annot.gz
```

```
pData(eset)$infection         # Let's check at the infections
```

```
##  [1] uninfected uninfected uninfected uninfected FRD875     FRD875
##  [7] FRD875     FRD875     FRD1234    FRD1234    FRD1234    FRD1
## [13] FRD1       FRD1       FRD1       FRD440     FRD440     FRD440
## [19] FRD440
## Levels: FRD1 FRD1234 FRD440 FRD875 uninfected
```

```
table(pData(eset)$infection)  # No. of samples in each infection
```

```
##
##       FRD1    FRD1234     FRD440     FRD875 uninfected
##          4          3          4          4          4
```

Filter columns associated with 'uninfected' and 'FRD440' values.

```
selected <- grep("uninfected|FRD440", pData(eset)$infection) # Select indexes
pData(eset)$infection[selected]                              # Check selection
```

```
## [1] uninfected uninfected uninfected uninfected FRD440     FRD440     FRD440
## [8] FRD440
## Levels: FRD1 FRD1234 FRD440 FRD875 uninfected
```

'Uninfected' as '1's and 'FRD440' '2's

```
y <- c(rep(1, 4), rep(2, 4))  # Vector of levels ID
y
```

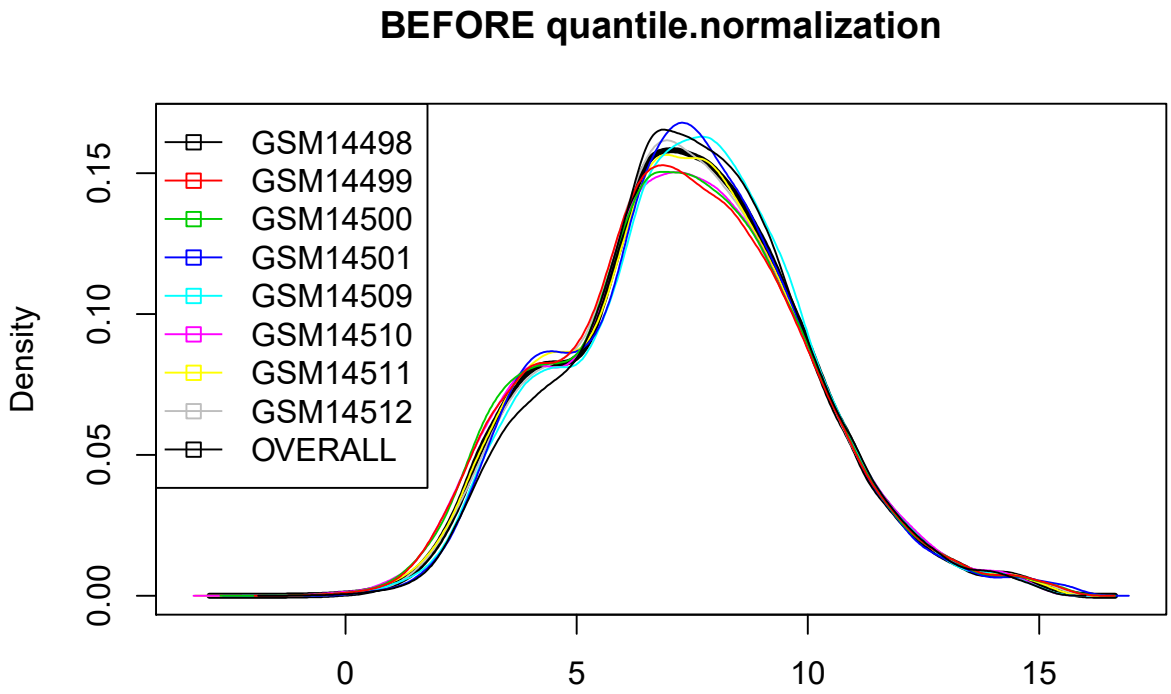
```
## [1] 1 1 1 1 2 2 2 2
```

A) Verifique si necesita normalizar los datos con quantile normalization. Obtenga p-values antes y después de normalizar.

Filter the expression matrix according to selected, then quantile normalize

[There is small-to-large variability within groups and small variability across groups, so, let's use quantile normalization]

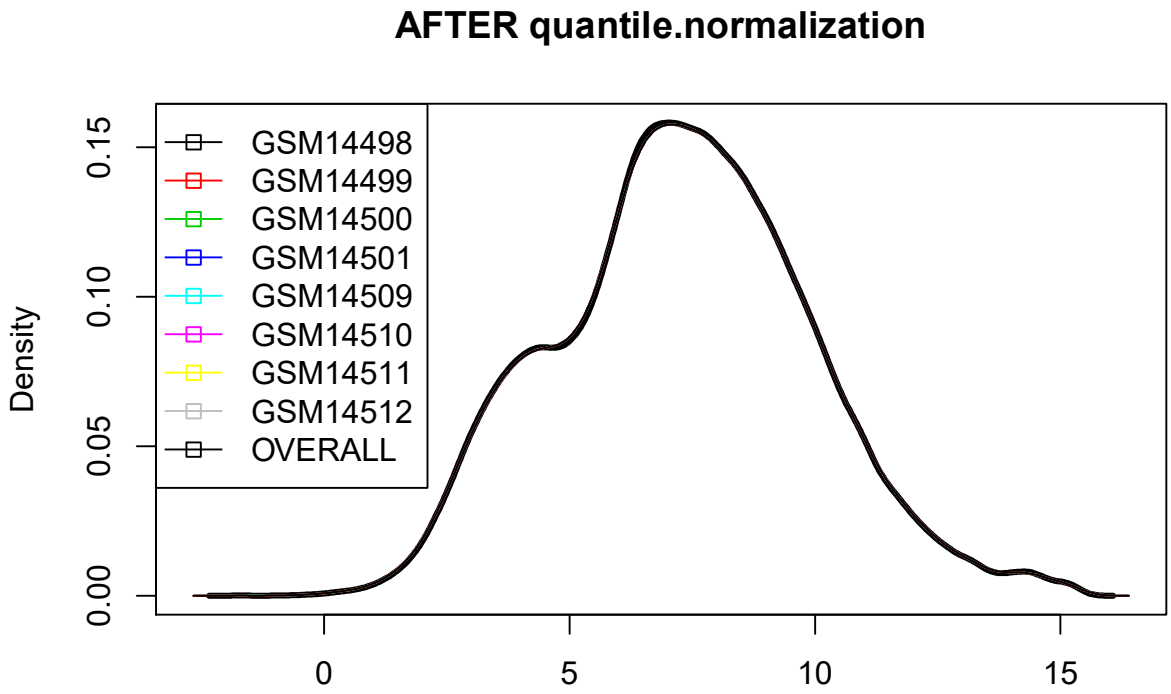
```
# Get a subset of expression values of the selected samples
exprs.selected <- exprs(eset)[, selected]
plot.densities(exprs.selected, main = "BEFORE quantile.normalization")
```




```
# p values before quantile normalize
pval = calPvalues(exprs.selected,y)
```

```
## We're doing 256 complete permutations
## and randomly select 100 of them.
```

```
exprs.selected.q <- normalizeQuantiles(exprs.selected)  # Quantile normalize the data
plot.densities(exprs.selected.q, main = "AFTER quantile.normalization")
```

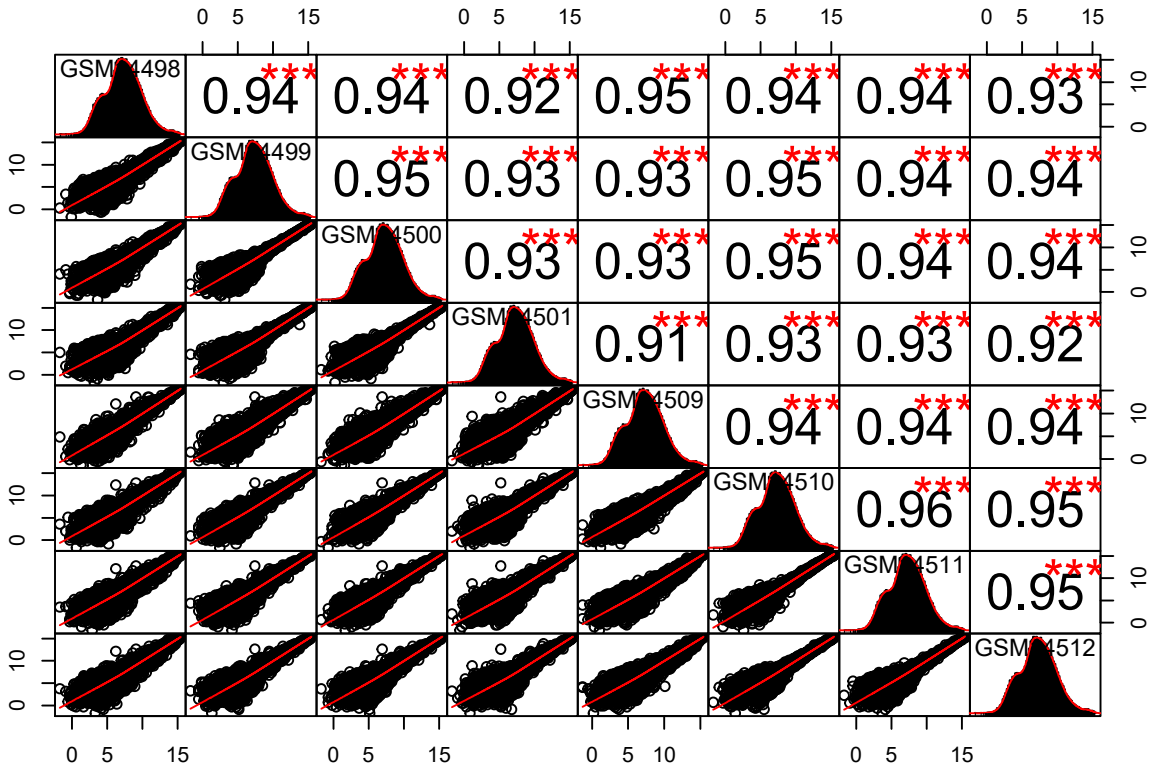



```
# p values after quantile normalize
pval.q = calPvalues(exprs.selected.q,y)
```

```
## We're doing 256 complete permutations
## and randomly select 100 of them.
```

B) Compare los p-values y grafique. Comente sobre la correlación entre las distintas pruebas estadísticas y la cantidad de genes significativos.

```
chart.Correlation(
  exprs.selected.q,
  histogram = TRUE
)
```



In the above plot:

- The distribution of each variable is shown on the diagonal.
- On the bottom of the diagonal : the bivariate scatter plots with a fitted line are displayed
- On the top of the diagonal : the value of the correlation plus the significance level as stars
- Each significance level is associated to a symbol : p-values(0, 0.001, 0.01, 0.05, 0.1, 1) <=> symbols(“ ”, “”, “”, “.”, " “)

Let's prepare the data that required to run **samr()**

```
# Get row names = gene names or IDs
genenames <- rownames(exprs(eset))
data <- list(
  x = exprs.selected.q,
  y = y,
  geneid = genenames,
  genenames = genenames,
  logged2 = TRUE
)
```

Refer to help(samr) for more info.

```
# Get object names from samr.obj
samr.obj <-
  samr(data, resp.type = "Two class unpaired", nperms = 100)
```

```
## perm= 1
## perm= 2
## perm= 3
## perm= 4
## perm= 5
## perm= 6
## perm= 7
## perm= 8
## perm= 9
## perm= 10
## perm= 11
```

```
## perm= 12
## perm= 13
## perm= 14
## perm= 15
## perm= 16
## perm= 17
## perm= 18
## perm= 19
## perm= 20
## perm= 21
## perm= 22
## perm= 23
## perm= 24
## perm= 25
## perm= 26
## perm= 27
## perm= 28
## perm= 29
## perm= 30
## perm= 31
## perm= 32
## perm= 33
## perm= 34
## perm= 35
## perm= 36
## perm= 37
## perm= 38
## perm= 39
## perm= 40
## perm= 41
## perm= 42
## perm= 43
## perm= 44
## perm= 45
## perm= 46
## perm= 47
## perm= 48
## perm= 49
## perm= 50
## perm= 51
## perm= 52
## perm= 53
## perm= 54
## perm= 55
## perm= 56
## perm= 57
## perm= 58
## perm= 59
## perm= 60
## perm= 61
## perm= 62
## perm= 63
## perm= 64
## perm= 65
## perm= 66
## perm= 67
## perm= 68
## perm= 69
## perm= 70
## perm= 71
## perm= 72
## perm= 73
```

```
## perm= 74
## perm= 75
## perm= 76
## perm= 77
## perm= 78
## perm= 79
## perm= 80
## perm= 81
## perm= 82
## perm= 83
## perm= 84
## perm= 85
## perm= 86
## perm= 87
## perm= 88
## perm= 89
## perm= 90
## perm= 91
## perm= 92
## perm= 93
## perm= 94
## perm= 95
## perm= 96
## perm= 97
## perm= 98
## perm= 99
## perm= 100
```

```
names(samr.obj)
```

```
##  [1] "n"                    "x"                    "xresamp"
##  [4] "y"                    "argy"                 "censoring.status"
##  [7] "testStatistic"        "nperms"               "nperms.act"
## [10] "tt"                   "numer"                "sd"
## [13] "sd.internal"          "s0"                   "s0.perc"
## [16] "evo"                  "perms"                "permsy"
## [19] "nresamp"              "nresamp.perm"         "all.perms.flag"
## [22] "ttstar"               "ttstar0"              "eigengene"
## [25] "eigengene.number"     "pi0"                  "foldchange"
## [28] "foldchange.star"      "sdstar.keep"          "resp.type"
## [31] "resp.type.arg"        "assay.type"           "stand.contrasts"
## [34] "stand.contrasts.star" "stand.contrasts.95"   "depth"
## [37] "call"
```

Compute thresholds for different deltas

```
delta <- 1.9 # define the delta
delta.table <- samr.compute.delta.table(samr.obj, min.foldchange = delta)
```

```
##
## Computing delta table
## 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
## 31
## 32
## 33
## 34
## 35
## 36
## 37
## 38
## 39
## 40
## 41
## 42
## 43
## 44
## 45
## 46
## 47
## 48
## 49
## 50
```

```
# Look at the whole delta table
datatable(delta.table)
```

Show 10 entries Search:

delta	# med false pos	90th perc false pos	# called	median FDR	90th perc FDR	cutlo	cuthi
0	1164.68967374231	1387.99928196383	1671	0.697001600085167	0.830639905424194	-0.647430045547494	0.697219939439642
0.00260135755549737	1164.68967374231	1387.99928196383	1671	0.697001600085167	0.830639905424194	-0.647430045547494	0.697219939439642
0.0104054302219895	1164.68967374231	1387.99928196383	1671	0.697001600085167	0.830639905424194	-0.647430045547494	0.697219939439642
0.0234122179994764	1164.68967374231	1387.99928196383	1671	0.697001600085167	0.830639905424194	-0.647430045547494	0.697219939439642
0.041621720887958	1164.68967374231	1387.99928196383	1671	0.697001600085167	0.830639905424194	-0.647430045547494	0.697219939439642
0.0650339388874343	1164.68967374231	1387.99928196383	1671	0.697001600085167	0.830639905424194	-0.647430045547494	0.697219939439642
0.0936488719979054	1164.68967374231	1387.99928196383	1671	0.697001600085167	0.830639905424194	-0.647430045547494	0.697219939439642
0.127466520219371	1158.77754341875	1380.90472557555	1656	0.69974489336881	0.833879665202627	-0.710796217943926	0.800696018108877
0.166486883551832	1069.25099851905	1301.59772023516	1568	0.681920279667762	0.830100586884666	-0.927596212436981	0.989389470572704
0.210709961995287	888.508728627205	1110.04469775165	1374	0.646658463338577	0.807892793123471	-1.12694971929253	1.17586508300884

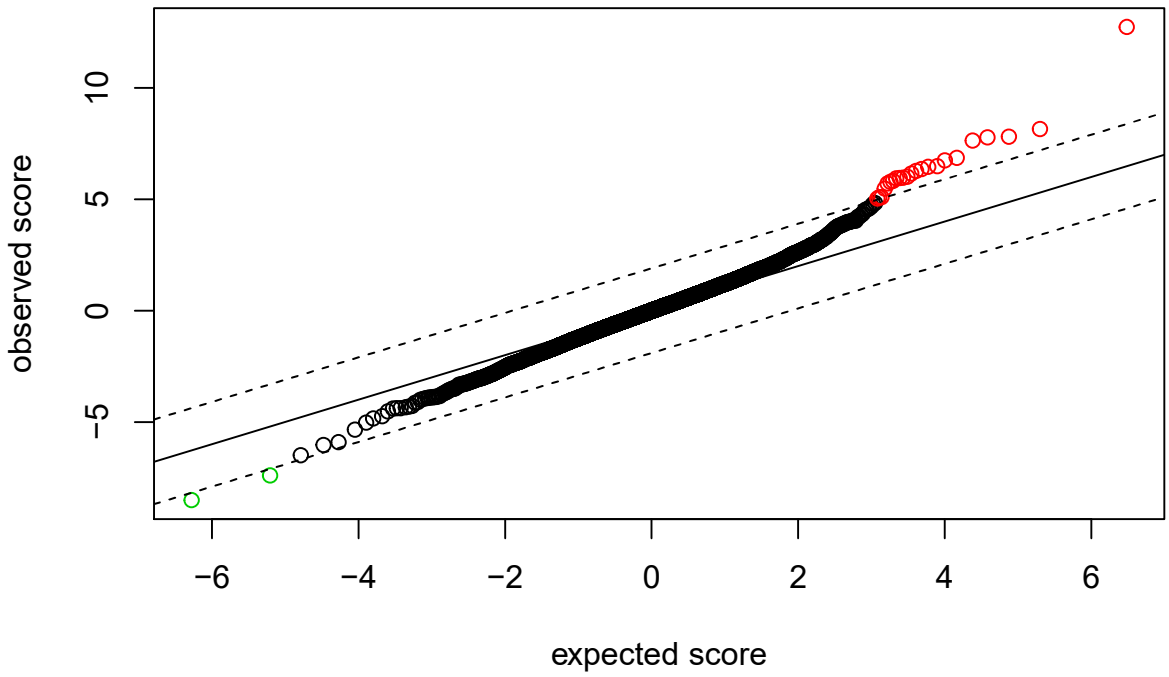
Showing 1 to 10 of 50 entries Previous 1 2 3 4 5 Next

Let's select delta with median FDR <10%.

```
# Check delta related to median FDR of 0.1
delta.table[delta.table[, "median FDR"] < 0.1,][1,]
```

```
##               delta     # med false pos 90th perc false pos           # called
##          1.49838195          3.37836018          6.75672037         35.00000000
##          median FDR       90th perc FDR               cutlo               cuthi
##          0.09652458          0.19304915         -5.91083339          4.45412422
```

```
samr.plot(samr.obj, delta)  # Check SAM plot
```



B) Comente sobre la cantidad de genes significativos.

Get significant genes

```
# Summarize
siggenes.table <- samr.compute.siggenes.table(samr.obj,
                                              delta,
                                              data,
                                              delta.table,
                                              min.foldchange = delta)

datatable(siggenes.table$genes.lo)  # Check how table with the results look like
```

Show 10 entries Search:

Row	Gene ID	Gene Name	Score(d)	Numerator(r)	Denominator(s+s0)	Fold Change	q-value(%)
14069	214692_s_at	214692_s_at	-8.50617104840418	-3.62955195559847	0.42669632845902	0.0807971404364847	3.83904566465263
15033	215659_at	215659_at	-7.39961498615012	-2.53948326562867	0.34319127013795	0.172004323535609	3.83904566465263

Showing 1 to 2 of 2 entries Previous 1 Next

```
datatable(siggenes.table$genes.up)  # Check how table with the results look like
```

Show 10 entries Search:

Row	Gene ID	Gene Name	Score(d)	Numerator(r)	Denominator(s+s0)	Fold Change	q-value(%)
3356	203828_s_at	203828_s_at	12.7306592719215	3.82917455857621	0.300783681095114	14.2133483704722	0
6722	207196_s_at	207196_s_at	8.15319831291686	1.70708362956261	0.209375948437086	3.26500144340122	0
16212	216841_s_at	216841_s_at	7.81296600184981	2.33033953287497	0.298265669186738	5.02923697090547	0
3948	204420_at	204420_at	7.7778136546489	1.46234052049765	0.188014342516884	2.75555041084664	0
2039	202510_s_at	202510_s_at	7.63293710482856	2.52750740656639	0.33113169568337	5.7657464946743	0
2388	202859_x_at	202859_x_at	6.85934644804566	2.05505726711766	0.299599573032673	4.15560136460615	0
5418	205890_s_at	205890_s_at	6.74959784875184	2.10594719020738	0.312010765292753	4.30480291911734	0
2167	202638_s_at	202638_s_at	6.48020898878246	2.62930742400152	0.405744232717335	6.18728900908746	0
2172	202643_s_at	202643_s_at	6.44957963641253	1.241728077032	0.192528528529449	2.36481623167662	0
10840	211429_s_at	211429_s_at	6.35794408086204	1.22566535837773	0.192776995643465	2.33863280551558	0

Showing 1 to 10 of 23 entries Previous 1 2 3 Next

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