

# 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(GEOquery)

## Loading required package: Biobase
## 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
## Welcome to Bioconductor
##
##   Vignettes contain introductory material; view with
##   'browseVignettes()'. To cite Bioconductor, see
##   'citation("Biobase)", and for packages 'citation("pkgname)".
## Setting options('download.file.method.GEOquery'='auto')
## Setting options('GEOquery.inmemory.gpl'=FALSE)
library(Biobase)
library(limma)

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

```
## The following object is masked from 'package:BiocGenerics':
##
##      plotMA
library(affy)
library(siggenes)

## Loading required package: multtest
## Loading required package: splines
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  $n = 2 \dots 20$ . Graficar los resultados y comparar.

```
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"
    )
  }

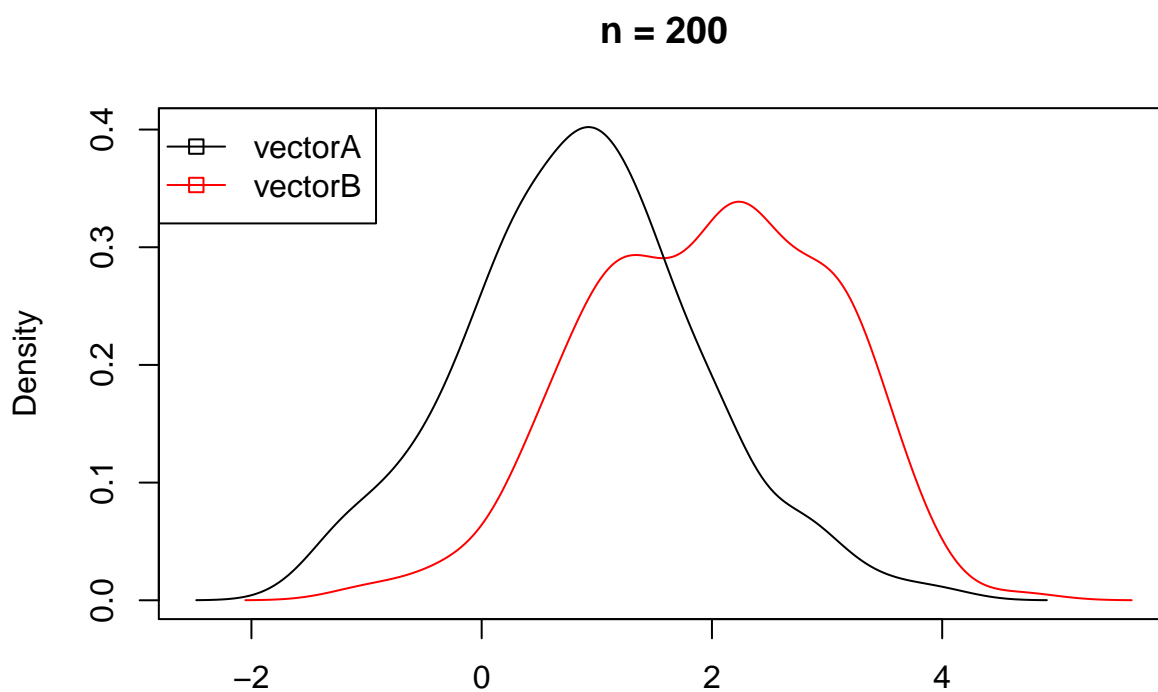
  lines(
    ns, wtest_pval,
    col="red"
  )

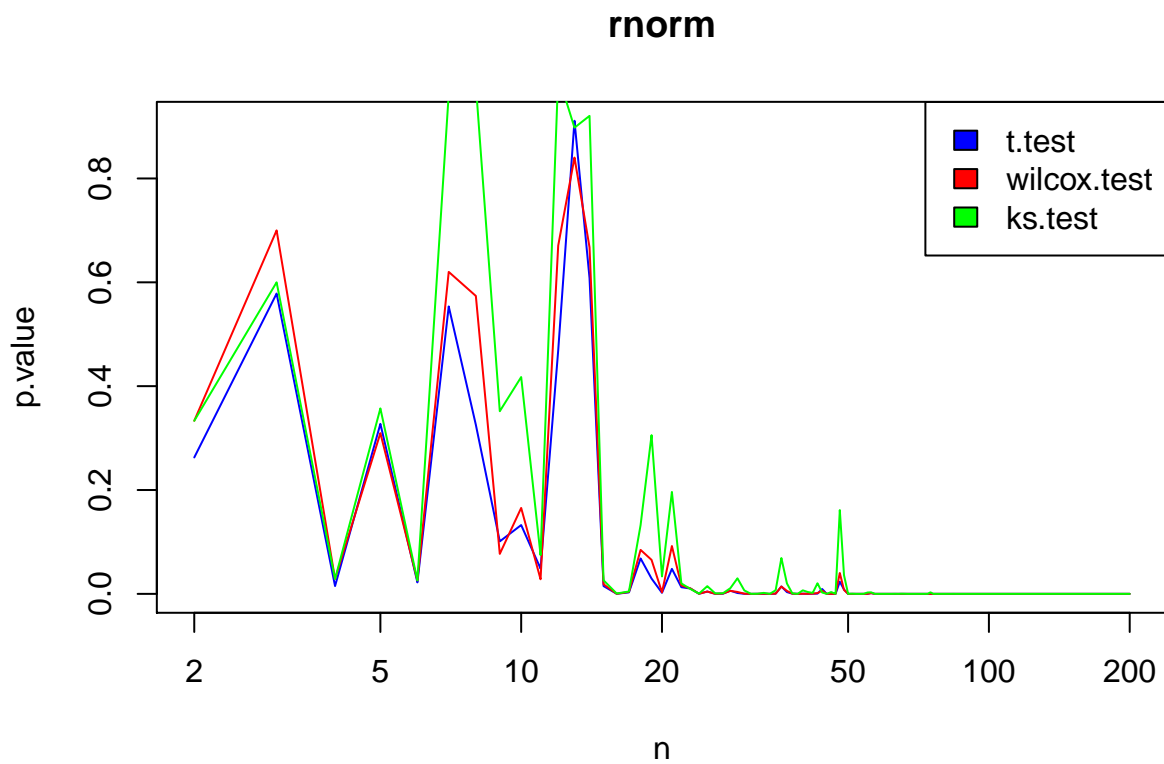
  lines(
    ns, ktest_pval,
    col="green"
  )

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

getPValues("rnorm")

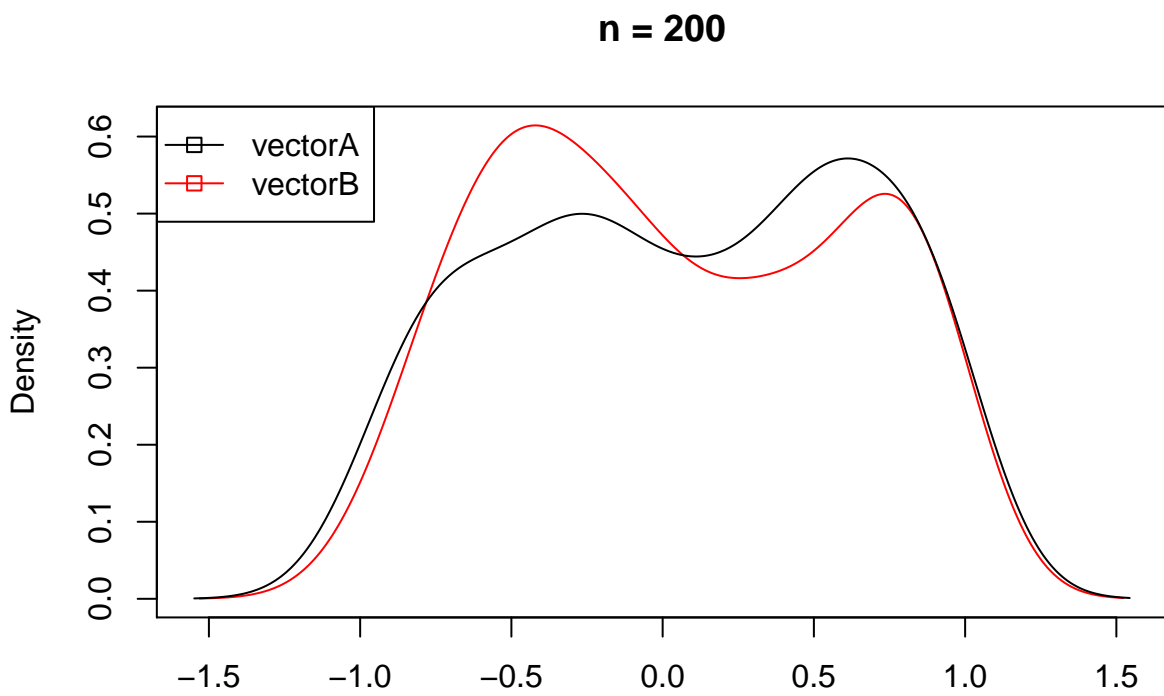
```

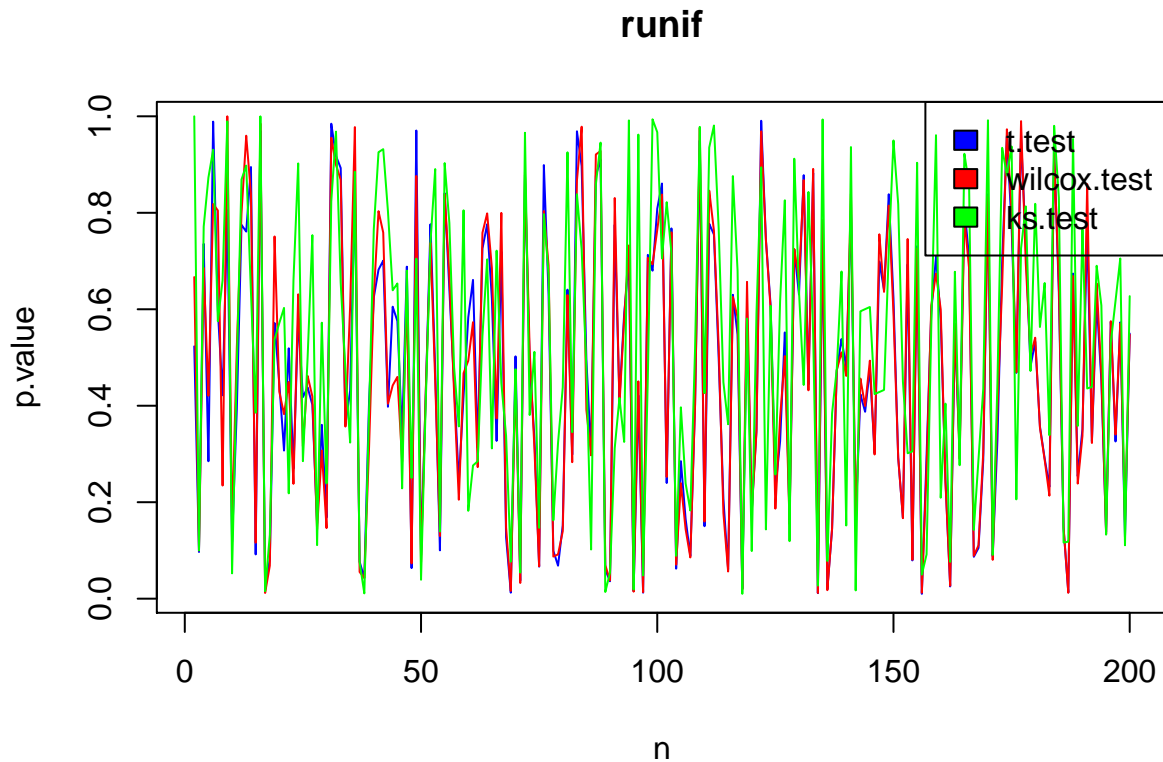




B) Repetir (A) con 2 vectores de datos generados con la función runif.

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






---

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

---

### Dataset details

**Title:** Severe acute respiratory syndrome expression profile

**Summary:** Expression profiling of peripheral blood mononuclear cells (PBMC) from 10 adult patients with severe acute respiratory syndrome (SARS). Results provide insight into the host immune response to the SARS coronavirus.

**Organism:** Homo sapiens

**Platform:** GPL201: [HG-Focus] Affymetrix Human HG-Focus Target Array

**Citation:**

Reghunathan R, Jayapal M, Hsu LY, Chng HH et al. *Expression profile of immune response genes in patients with Severe Acute Respiratory Syndrome*. BMC Immunol 2005 Jan 18;6:2. PMID: 15655079

**Reference Series:** GSE1739

**Sample count:** 14

**Value type:** count

**Series published:** 2005/01/18

**Dataset taken from:** GDSbrowser, GSE1739 query, and GSE1739 geo2r

---

The GSE1739 was selected for the Bioinformatics class assignments and activities. The reason behind this decision is the similarities that SARS shares with the novel coronavirus COVID-19. The purpose is to work with up-to-date data that is relevant to the current crisis.

---

Let's download .soft.gz from GEO and take a look to the data

```
# If you have network access, the more typical way to do this  
# would be to use this:
```

```
gse <- getGEO("GSE1739",GSEMatrix=FALSE)
```

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

```
## C:\Users\oskat\AppData\Local\Temp\RtmpmKxJRf/GSE1739.soft.gz
```

```
## Reading file....
```

```
## Parsing....
```

```
## Found 15 entities...
```

```
## GPL201 (1 of 16 entities)
```

```
## GSM30361 (2 of 16 entities)
```

```
## GSM30362 (3 of 16 entities)
```

```
## GSM30363 (4 of 16 entities)
```

```
## GSM30364 (5 of 16 entities)
```

```
## GSM30365 (6 of 16 entities)
```

```
## GSM30366 (7 of 16 entities)
```

```
## GSM30367 (8 of 16 entities)
```

```
## GSM30368 (9 of 16 entities)
```

```
## GSM30369 (10 of 16 entities)
```

```
## GSM30370 (11 of 16 entities)
```

```
## GSM30371 (12 of 16 entities)
```

```
## GSM30372 (13 of 16 entities)
```

```
## GSM30373 (14 of 16 entities)
```

```
## GSM30374 (15 of 16 entities)
```

```
# gse <- getGEO(filename=system.file("extdata/GSE781_family.soft.gz",package="GEOquery"))
```

```
# Look at gse metadata:
```

```
head(Meta(gse))
```

```
## $contact_address
```

```
## [1] " "
```

```
##
```

```
## $contact_city
```

```
## [1] "Singapore"
```

```
##
```

```
## $contact_country
```

```
## [1] "Singapore"
```

```
##
```

```
## $contact_institute
```

```
## [1] "NUS"
```

```
##
```

```
## $contact_name
```

```
## [1] "Jayapal,,Manikandan"
```

```
##
```

```
## $`contact_zip/postal_code`
```

```
## [1] "117597"
```

```
# names of all the GSM objects contained in the GSE
```

```
names(GSMList(gse))
```

```
## [1] "GSM30361" "GSM30362" "GSM30363" "GSM30364" "GSM30365" "GSM30366"
## [7] "GSM30367" "GSM30368" "GSM30369" "GSM30370" "GSM30371" "GSM30372"
## [13] "GSM30373" "GSM30374"
```

```
# and get the first GSM object on the list
GSMList(gse)[[1]]
```

```
## An object of class "GSM"
## channel_count
## [1] "1"
## contact_address
## [1] " "
## contact_city
## [1] "Singapore"
## contact_country
## [1] "Singapore"
## contact_institute
## [1] "NUS"
## contact_name
## [1] "Jayapal,,Manikandan"
## contact_zip/postal_code
## [1] "117597"
## data_row_count
## [1] "8793"
## description
## [1] "PBMC normal sample RNA for Control"
## geo_accession
## [1] "GSM30361"
## last_update_date
## [1] "May 27 2005"
## molecule_ch1
## [1] "total RNA"
## organism_ch1
## [1] "Homo sapiens"
## platform_id
## [1] "GPL201"
## series_id
## [1] "GSE1739"
## source_name_ch1
## [1] "PBMC normal sample RNA"
## status
## [1] "Public on Jan 18 2005"
## submission_date
## [1] "Sep 08 2004"
## supplementary_file
## [1] "NONE"
## taxid_ch1
## [1] "9606"
## title
## [1] "N1"
## type
## [1] "RNA"
## An object of class "GEODataTable"
## ***** Column Descriptions *****
##           Column           Description
## 1           ID_REF
## 2           VALUE           raw signal intensity
## 3           ABS_CALL present, absent, marginal
## 4 DETECTION P-VALUE           p-value
## ***** Data Table *****
##           ID_REF  VALUE  ABS_CALL  DETECTION  P-VALUE
## 1 1007_s_at    321.8         P      0.035163
## 2  1053_at    204.8         P      0.006532
```

```
## 3    117_at  538.6      P      0.001141
## 4    121_at 1277.8      P      0.011447
## 5 1255_g_at  51.3      A      0.418069
## 8788 more rows ...
```

```
# and the names of the GPLs represented
names(GPLList(gse))
```

```
## [1] "GPL201"
```

Let's prepare the data for further analysis

```
# First, we need to make sure that all of the GSMs are from the same platform:
gsmplatforms <- lapply(GSMList(gse),function(x) {Meta(x)$platform_id})
head(gsmplatforms)
```

```
## $GSM30361
## [1] "GPL201"
##
## $GSM30362
## [1] "GPL201"
##
## $GSM30363
## [1] "GPL201"
##
## $GSM30364
## [1] "GPL201"
##
## $GSM30365
## [1] "GPL201"
##
## $GSM30366
## [1] "GPL201"
```

```
# If there are more GPLs, we can filter the original GSMList to include only those GSMs within
# a specific platform and use this list for further processing
gsmlist = Filter(function(gsm) {Meta(gsm)$platform_id=='GPL201'},GSMList(gse))
length(gsmlist)
```

```
## [1] 14
```

```
# So, now we would like to know what column represents the data that we would like to extract.
# Looking at the first few rows of the Table of a single GSM will likely give us an idea (and
# by the way, GEO uses a convention that the column that contains the single measurement for
# each array is called the VALUE column, which we could use if we don't know what other column
# is most relevant).
```

```
Table(gsmlist[[1]])[1:5,]
```

```
##      ID_REF  VALUE ABS_CALL DETECTION P-VALUE
## 1 1007_s_at  321.8      P      0.035163
## 2 1053_at   204.8      P      0.006532
## 3 117_at    538.6      P      0.001141
## 4 121_at   1277.8      P      0.011447
## 5 1255_g_at  51.3      A      0.418069
```

```
# and get the column descriptions
Columns(gsmlist[[1]])#[1:5,]
```

```
##      Column      Description
## 1      ID_REF
## 2      VALUE      raw signal intensity
## 3      ABS_CALL  present, absent, marginal
## 4 DETECTION P-VALUE      p-value
```



*#We will indeed use the VALUE column. We then want to make a matrix of these values like so:*

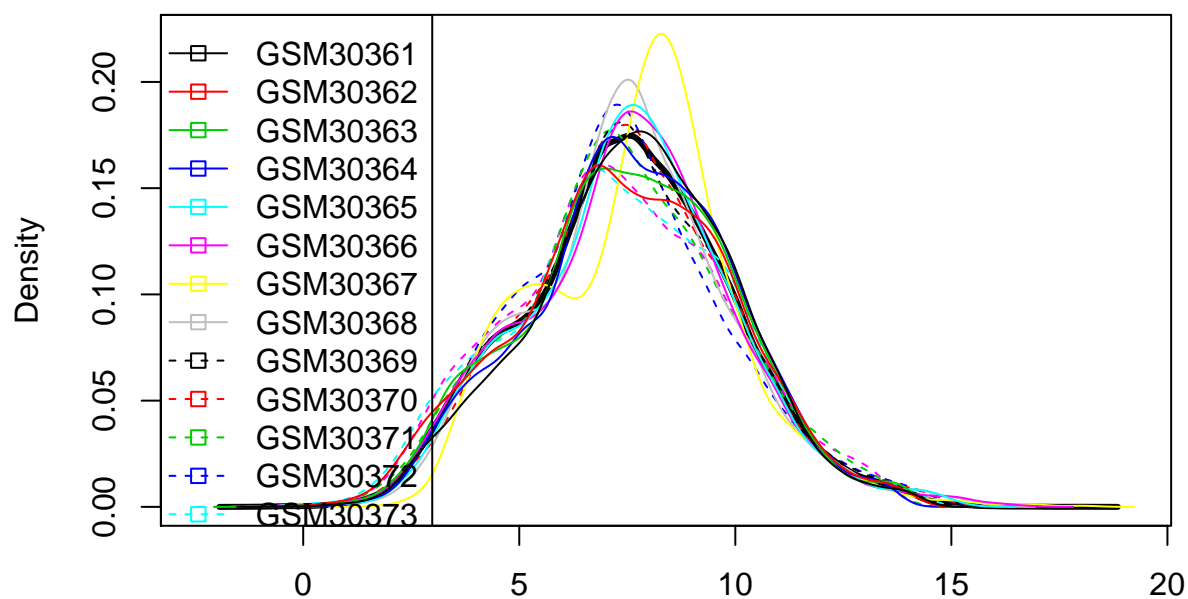
```
# get the probeset ordering
probesets <- Table(GPLList(gse)[[1]])$ID
# make the data matrix from the VALUE columns from each GSM
# being careful to match the order of the probesets in the platform
# with those in the GSMs
data.matrix <- do.call('cbind',lapply(gsmList,function(x)
  {tab <- Table(x)
    mymatch <- match(probesets,tab$ID_REF)
    return(tab$VALUE[mymatch])
  }))
data.matrix <- apply(data.matrix,2,function(x) {as.numeric(as.character(x))})
data.matrix <- log2(data.matrix)
data.matrix[1:5,]
```

```
##      GSM30361 GSM30362 GSM30363 GSM30364 GSM30365 GSM30366 GSM30367
## [1,] 8.330021 8.006186 8.370687 8.518850 7.680887 7.806711 8.033974
## [2,] 7.678072 8.197217 7.764208 8.032321 7.753551 8.053111 5.809929
## [3,] 9.073070 8.519636 9.047124 8.500244 9.132114 8.513333 8.106432
## [4,] 10.319446 9.781688 10.010948 10.016948 9.627351 9.988827 10.186981
## [5,] 5.680887 5.459432 5.375039 4.722466 5.741467 5.189825 4.478972
##      GSM30368 GSM30369 GSM30370 GSM30371 GSM30372 GSM30373 GSM30374
## [1,] 8.292322 7.997179 7.879583 7.622052 7.624978 8.186857 7.050937
## [2,] 7.668885 7.488644 7.089583 7.566054 7.309249 6.072535 7.260214
## [3,] 10.018617 9.144403 9.201389 10.172177 10.452653 8.000000 8.724855
## [4,] 10.215654 10.021813 9.954778 9.737585 10.087728 9.471472 9.545737
## [5,] 6.203593 5.672425 5.409391 6.002252 5.057450 5.133399 6.213347
```

```
# #Correr SAM
# data.matrix_sam <- sam(
#   data.matrix,
#   c(rep(1,dim(data.matrix)[2])),
#   method = "d.stat",
#   gene.names = colnames(data.matrix)
# )
# sam_pval = data.matrix_sam@p.value
# length(sam_pval)
#
# #Mostrar resultados de SAM con cierto valor de delta
# data.matrix_sam_sum <- summary(data.matrix_sam, 1.9)
#
# #Acceder a matrix de Genes significativos
# dim(data.matrix_sam_sum@mat.sig)
```

```
plot.densities(data.matrix, main="BEFORE quantile.normalization()")
```

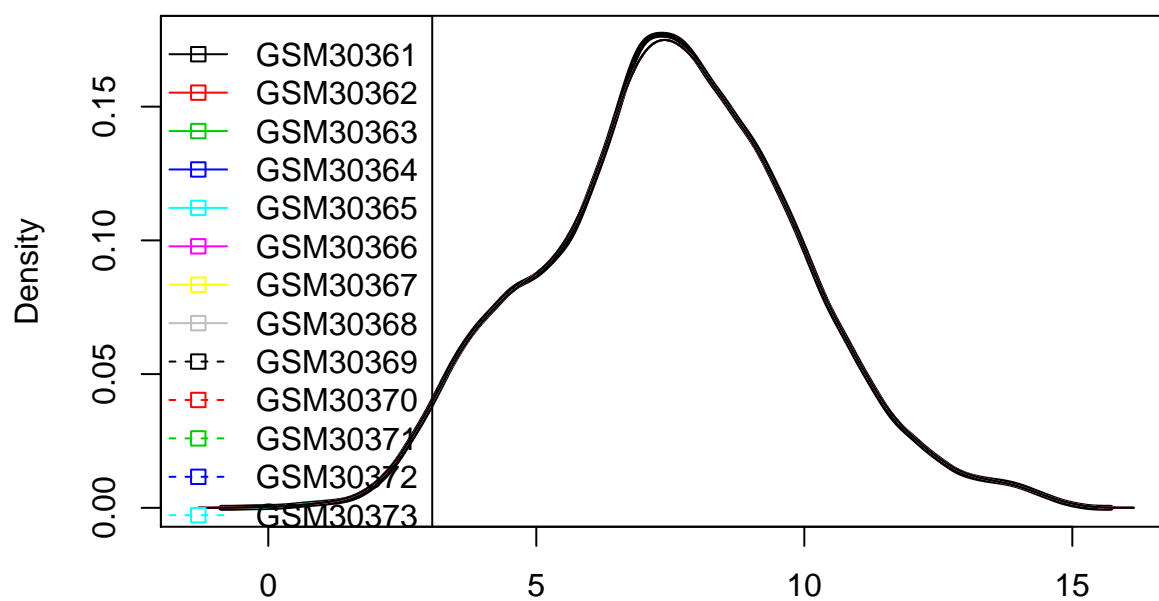
## BEFORE quantile.normalization()



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

```
data.matrix_qn <- quantile.normalization(data.matrix)
plot.densities(data.matrix_qn, main="AFTER quantile.normalization()")
```

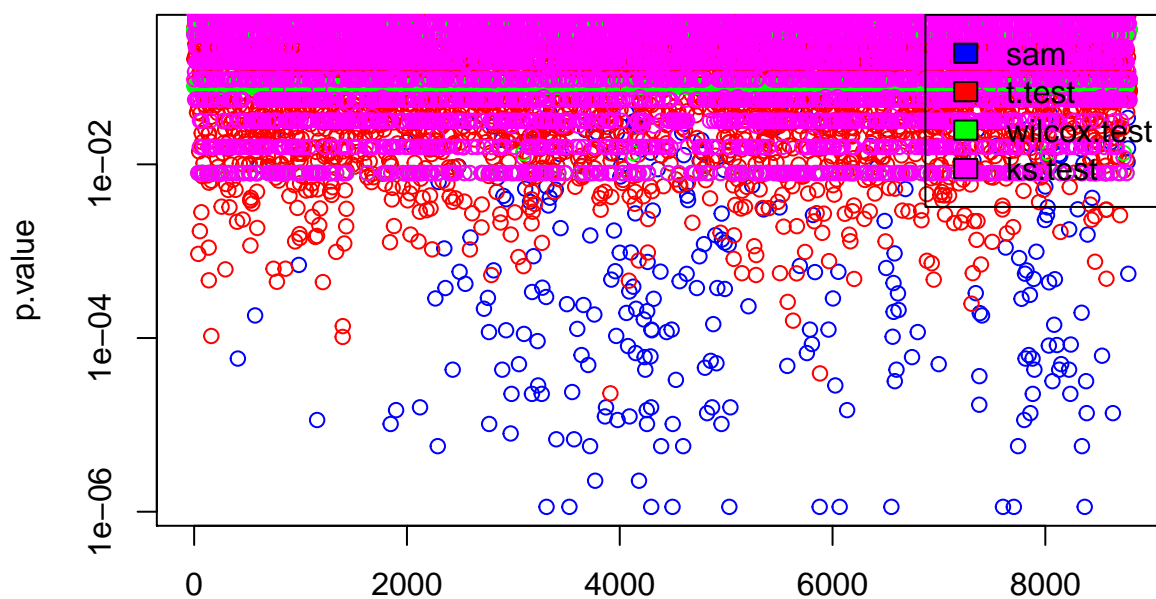
## AFTER quantile.normalization()



## Compute the p-values for t-test, Wilcoxon, Kolmogorov & SAM

```
pval = de.test(  
  x=data.matrix,  
  classes=c(rep(1,5),rep(0,5)),  
  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)  
)  
  
## We're doing 16384 complete permutations  
## and randomly select 100 of them.  
sam_pval = data.matrix_sam@p.value  
ttest_pval = pval$ttest  
wtest_pval = pval$kolmogorov  
ktest_pval = pval$wilcoxon  
  
ns = 1:length(sam_pval)  
plot(  
  ns, sam_pval,  
  main="runif",  
  xlab="",  
  ylab="p.value",  
  log="y",  
  col="blue",  
  type="p"  
)  
  
lines(  
  ns, ttest_pval,  
  col="red",  
  type="p"  
)  
  
lines(  
  ns, wtest_pval,  
  col="green",  
  type="p"  
)  
  
lines(  
  ns, ktest_pval,  
  col="magenta",  
  type="p"  
)  
  
legend(  
  "topright",  
  c("sam", "t.test", "wilcox.test", "ks.test"),  
  fill=c("blue", "red", "green", "magenta")  
)
```

## runif



```
# pval_qn = de.test(
#   x=data.matrix_qn,
#   classes=c(rep(1,5),rep(0,5)),
#   test=c("ttest", "kolmogorov", "wilcoxon")
# )
# data.matrix_qn_sam <- sam(
#   data.matrix_qn,
#   c(rep(1,dim(data.matrix_qn)[2])),
#   method = "d.stat",
#   gene.names = colnames(data.matrix_qn)
# )
#
# sam_pval_qn = data.matrix_qn_sam@p.value
# ttest_pval_qn = pval_qn$ttest
# wtest_pval_qn = pval_qn$kolmogorov
# ktest_pval_qn = pval_qn$wilcoxon

# go through the necessary steps to make a compliant ExpressionSet
# rownames(data.matrix) <- probesets
# colnames(data.matrix) <- names(gsmlist)
# pdata <- data.frame(samples=names(gsmlist))
# rownames(pdata) <- names(gsmlist)
# pheno <- as(pdata,"AnnotatedDataFrame")
# eset2 <- new('ExpressionSet',exprs=data.matrix,phenoData=pheno)
# eset2
gset = getGEO("GSE1739",GSEMatrix=TRUE,destdir="./NCBI_GEO")

## Found 1 file(s)
## GSE1739_series_matrix.txt.gz
## Using locally cached version: ./NCBI_GEO/GSE1739_series_matrix.txt.gz
## Parsed with column specification:
## cols(
##   ID_REF = col_character(),
```

```
## GSM30361 = col_double(),
## GSM30362 = col_double(),
## GSM30363 = col_double(),
## GSM30364 = col_double(),
## GSM30365 = col_double(),
## GSM30366 = col_double(),
## GSM30367 = col_double(),
## GSM30368 = col_double(),
## GSM30369 = col_double(),
## GSM30370 = col_double(),
## GSM30371 = col_double(),
## GSM30372 = col_double(),
## GSM30373 = col_double(),
## GSM30374 = col_double()
## )

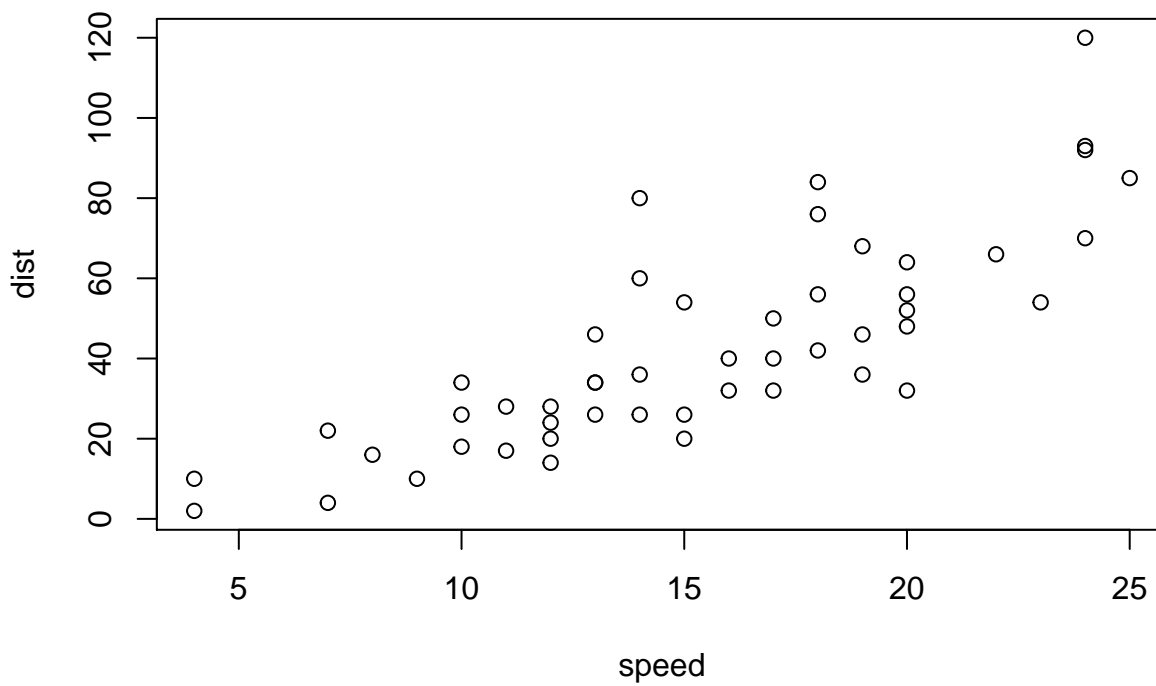
## Using locally cached version of GPL201 found here:
## ./NCBI_GEO/GPL201.soft

gset

## $GSE1739_series_matrix.txt.gz
## ExpressionSet (storageMode: lockedEnvironment)
## assayData: 8793 features, 14 samples
## element names: exprs
## protocolData: none
## phenoData
## sampleNames: GSM30361 GSM30362 ... GSM30374 (14 total)
## varLabels: title geo_accession ... data_row_count (21 total)
## varMetadata: labelDescription
## featureData
## featureNames: 1007_s_at 1053_at ... AFFX-r2-P1-cre-5_at (8793 total)
## fvarLabels: ID GB_ACC ... Gene Ontology Molecular Function (16 total)
## fvarMetadata: Column Description labelDescription
## experimentData: use 'experimentData(object)'
## pubMedIds: 15655079
## Annotation: GPL201
```

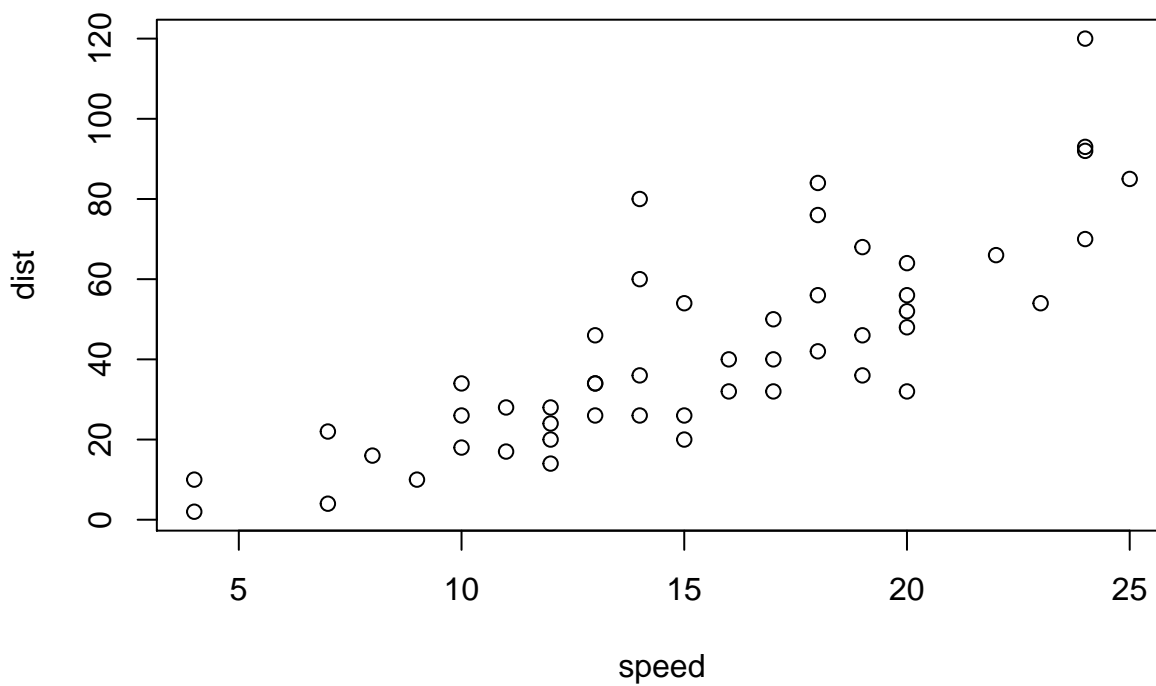
A) Verifique si necesita normalizar los datos con quantile normalization. Obtenga p-values antes y después de normalizar.

```
plot(cars)
```



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

```
plot(cars)
```



C) Usando el GPL de los datos de GEO, dar una significancia biológica de los genes significativos.

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

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