# PEC 1. Análisis de datos ómicos.

Guillermo Prol Castelo

09/11/20

## Contents



## I. Abstract.

In this microarray analysis, a study of the effect of chronic ethanol consumption on rat pancreas has been chosen (GEO reference *GSE3311*). In the following, the experiment, data and analysis workflow are described. An in-depth analysis of the microarray is carried out to find out whether there exist any differences between the control and treatment with ethanol groups. It is decided that such a significant difference exist: ethanol had an effect on gene expression.

## II. Objetivos.

Queremos saber si existen diferencias en la expresión genética del páncreas de aquellos ratones que siguieron una dieta con etanol.

# III. Materiales y métodos.

## III.1. Los datos.

Están disponibles a través del identificador *GSE3311* en *Gene Expression Omnibus (GEO)*. Se trata de un microarray de Affymetrix en el que se analizó la expresión de RNA de los grupos control y de tratamiento. El tratamiento corresponde a una dieta que incluye etanol. Ambas se suministraron a ratones ( *Rattus norvegicus*) durante 8 semanas.

El experimento se diseñó con dos factores, correspondientes al tipo de dieta: control y etanol. Cada tipo de dieta se repitió 3 veces, cada una en 6 ratones (macho) escogidos al azar en un determinado rango de peso (90-120 g). Podemos decir que el diseño del experimento es aleatorio (los factores o dietas se asignaron al azar a las unidades experimentales o ratones).

## III.2. Métodos.

Para analizar los datos del microarray hemos seguido un *workflow* que comienza por crear todas las carpetas necesarias en el fichero en que nos encontremos. A continuación, obtenemos los datos de GEO y extraemos los ficheros .CEL. Una vez importados a R, podemos explorar estos datos, realizar un control de calidad, normalizarlos y filtrarlos. Una vez hecho esto, podemos anotar los resultados, realizar comparaciones múltiples y un análisis de significación biológica.

## III.3. Librerías, directorios y funciones.

Comenzamos cargando todas las librerías que nos harán falta a lo largo del análisis.

```
suppressMessages( library(Biobase) )
suppressMessages( library(GEOquery) )
suppressMessages( library(affy) )
suppressMessages( library(limma) )
suppressMessages( library(annotate) )
suppressMessages( library(annaffy) )
```

```
## Warning: Package 'KEGG.db' is deprecated and will be removed from Bioconductor
##   version 3.12
```

```
suppressMessages( library(gplots) )
suppressMessages( library(genefilter) )
suppressMessages( library(oligo) )
suppressMessages( library(arrayQualityMetrics) )
suppressMessages( library(ggplot2) )
suppressMessages( library(ggrepel) )
suppressMessages( library(gmodels) )
```

```
## Warning: package 'gmodels' was built under R version 4.0.3
```

```
suppressMessages( library(affyQCReport) )
```

Para facilitar el estudio, trabajaremos en un directorio escogido por nosotros y cuya localización se asigna a la variable `workingDir`. Los datos se copiarán en un subdirectorio del anterior denominado `data`, que se

almacenará en la variable `dataDir` y los resultados se almacenarán en un directorio `results`, cuyo nombre se almacenará en la variable `resultsDir`. También creamos la carpeta `celfiles` para guardar en ella los ficheros .CEL.

```
workingDir <- getwd()
# Creamos los directorios de datos y resultados:
system("mkdir data")
```

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

```
system("mkdir results")
```

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

```
system("mkdir celfiles")
```

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

```
# Los asignamos a variables:
dataDir <-file.path(workingDir, "data")
resultsDir <- file.path(workingDir, "results")
celfilesDir <- file.path(workingDir,"celfiles")
# Seleccionamos el directorio de trabajo:
setwd(workingDir)
```

Cargamos las funciones necesarias para el análisis.

```
# Declaramos el directorio de funciones:
functionsDir <- file.path(workingDir, "functions/")
# Cargamos la función plotPCA.R
source(file.path(functionsDir,"plotPCA.R"))
```

# IV. Resultados

## IV.1. Obtención y lectura de los datos.

Comenzamos por obtener los datos desde *GEO Accession*:

```
# Guardamos los datos en una variable en R:
gse <- getGEO("GSE3311", GSEMatrix = T, destdir = dataDir)
```

```
## Found 1 file(s)
```

```
## GSE3311_series_matrix.txt.gz
```

```
## Using locally cached version: C:/Users/G-mo10/Desktop/repos_ado/PEC1_ADO/data/GSE3311_series_matrix.t
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   ID_REF = col_character(),
##   GSM74493 = col_double(),
##   GSM74494 = col_double(),
##   GSM74495 = col_double(),
##   GSM74496 = col_double(),
##   GSM74497 = col_double(),
##   GSM74498 = col_double()
## )

## Using locally cached version of GPL341 found here:
## C:/Users/G-mo10/Desktop/repos_ado/PEC1_ADO/data/GPL341.soft

## Warning: 57 parsing failures.
##   row     col           expected    actual         file
## 15867 SPOT_ID 1/0/T/F/TRUE/FALSE --Control literal data
## 15868 SPOT_ID 1/0/T/F/TRUE/FALSE --Control literal data
## 15869 SPOT_ID 1/0/T/F/TRUE/FALSE --Control literal data
## 15870 SPOT_ID 1/0/T/F/TRUE/FALSE --Control literal data
## 15871 SPOT_ID 1/0/T/F/TRUE/FALSE --Control literal data
## ..... ....... .................. ......... .............
## See problems(...) for more details.
```

```
# Descargamos los datos ('raw data') en la carpeta de datos:
a <- getGEOSuppFiles("GSE3311",makeDirectory = F, baseDir = dataDir)
```

Podemos explorar su contenido:

```
rownames(pData(phenoData(gse[[1]])))
```

```
## [1] "GSM74493" "GSM74494" "GSM74495" "GSM74496" "GSM74497" "GSM74498"
```

```
colnames(pData(phenoData(gse[[1]])))
```

```
##  [1] "title"                  "geo_accession"
##  [3] "status"                 "submission_date"
##  [5] "last_update_date"       "type"
##  [7] "channel_count"          "source_name_ch1"
##  [9] "organism_ch1"           "characteristics_ch1"
## [11] "molecule_ch1"           "extract_protocol_ch1"
## [13] "label_ch1"              "label_protocol_ch1"
## [15] "taxid_ch1"              "hyb_protocol"
## [17] "scan_protocol"          "description"
## [19] "data_processing"        "platform_id"
## [21] "contact_name"           "contact_email"
## [23] "contact_phone"          "contact_institute"
## [25] "contact_address"        "contact_city"
## [27] "contact_state"          "contact_zip/postal_code"
## [29] "contact_country"        "supplementary_file"
## [31] "data_row_count"
```

```
head(pData(phenoData(gse[[1]])))
```

```
##                                       title geo_accession
## GSM74493 pancreas, control diet, replicate 1      GSM74493
## GSM74494 pancreas, control diet, replicate 2      GSM74494
## GSM74495 pancreas, control diet, replicate 3      GSM74495
## GSM74496 pancreas, ethanol diet, replicate 1      GSM74496
## GSM74497 pancreas, ethanol diet, replicate 2      GSM74497
## GSM74498 pancreas, ethanol diet, replicate 3      GSM74498
##                          status submission_date last_update_date type
## GSM74493 Public on Jul 01 2006     Sep 15 2005      Sep 15 2005  RNA
## GSM74494 Public on Jul 01 2006     Sep 15 2005      Sep 15 2005  RNA
## GSM74495 Public on Jul 01 2006     Sep 15 2005      Sep 15 2005  RNA
## GSM74496 Public on Jul 01 2006     Sep 15 2005      Sep 15 2005  RNA
## GSM74497 Public on Jul 01 2006     Sep 15 2005      Sep 15 2005  RNA
## GSM74498 Public on Jul 01 2006     Sep 15 2005      Sep 15 2005  RNA
##          channel_count source_name_ch1      organism_ch1
## GSM74493             1        pancreas Rattus norvegicus
## GSM74494             1        pancreas Rattus norvegicus
## GSM74495             1        pancreas Rattus norvegicus
## GSM74496             1        pancreas Rattus norvegicus
## GSM74497             1        pancreas Rattus norvegicus
## GSM74498             1        pancreas Rattus norvegicus
##                             characteristics_ch1 molecule_ch1
## GSM74493 pancreas, control diet, male Wistar rat    total RNA
## GSM74494 pancreas, control diet, male Wistar rat    total RNA
## GSM74495 pancreas, control diet, male Wistar rat    total RNA
## GSM74496 pancreas, ethanol diet, male Wistar rat    total RNA
## GSM74497 pancreas, ethanol diet, male Wistar rat    total RNA
## GSM74498 pancreas, ethanol diet, male Wistar rat    total RNA
##                                                                                                  extract_protocol_ch1
## GSM74493 Total RNA was extracted from the tissue lysate using an RNeasy kit (Qiagen).
## GSM74494 Total RNA was extracted from the tissue lysate using an RNeasy kit (Qiagen).
## GSM74495 Total RNA was extracted from the tissue lysate using an RNeasy kit (Qiagen).
## GSM74496 Total RNA was extracted from the tissue lysate using an RNeasy kit (Qiagen).
## GSM74497 Total RNA was extracted from the tissue lysate using an RNeasy kit (Qiagen).
## GSM74498 Total RNA was extracted from the tissue lysate using an RNeasy kit (Qiagen).
##          label_ch1
## GSM74493    biotin
## GSM74494    biotin
## GSM74495    biotin
## GSM74496    biotin
## GSM74497    biotin
## GSM74498    biotin
##                                                                                                  label_protocol_ch1
## GSM74493 5 Âµg of total RNA was processed to produce biotinylated cRNA targets.
## GSM74494 5 Âµg of total RNA was processed to produce biotinylated cRNA targets.
## GSM74495 5 Âµg of total RNA was processed to produce biotinylated cRNA targets.
## GSM74496 5 Âµg of total RNA was processed to produce biotinylated cRNA targets.
## GSM74497 5 Âµg of total RNA was processed to produce biotinylated cRNA targets.
## GSM74498 5 Âµg of total RNA was processed to produce biotinylated cRNA targets.
##          taxid_ch1                    hyb_protocol
## GSM74493     10116 standard Affymetrix procedures
```

```
## GSM74494     10116 standard Affymetrix procedures
## GSM74495     10116 standard Affymetrix procedures
## GSM74496     10116 standard Affymetrix procedures
## GSM74497     10116 standard Affymetrix procedures
## GSM74498     10116 standard Affymetrix procedures
##                          scan_protocol                description
## GSM74493 standard Affymetrix procedures Rat pancreas, control diet
## GSM74494 standard Affymetrix procedures Rat pancreas, control diet
## GSM74495 standard Affymetrix procedures Rat pancreas, control diet
## GSM74496 standard Affymetrix procedures Rat pancreas, ethanol diet
## GSM74497 standard Affymetrix procedures Rat pancreas, ethanol diet
## GSM74498 standard Affymetrix procedures Rat pancreas, ethanol diet
##                                         data_processing platform_id
## GSM74493 Ann Arbor quantile-normalized trimmed-mean method      GPL341
## GSM74494 Ann Arbor quantile-normalized trimmed-mean method      GPL341
## GSM74495 Ann Arbor quantile-normalized trimmed-mean method      GPL341
## GSM74496 Ann Arbor quantile-normalized trimmed-mean method      GPL341
## GSM74497 Ann Arbor quantile-normalized trimmed-mean method      GPL341
## GSM74498 Ann Arbor quantile-normalized trimmed-mean method      GPL341
##          contact_name  contact_email contact_phone      contact_institute
## GSM74493  Rork,,Kuick rork@umich.edu  734-936-9241 University of Michigan
## GSM74494  Rork,,Kuick rork@umich.edu  734-936-9241 University of Michigan
## GSM74495  Rork,,Kuick rork@umich.edu  734-936-9241 University of Michigan
## GSM74496  Rork,,Kuick rork@umich.edu  734-936-9241 University of Michigan
## GSM74497  Rork,,Kuick rork@umich.edu  734-936-9241 University of Michigan
## GSM74498  Rork,,Kuick rork@umich.edu  734-936-9241 University of Michigan
##                                                          contact_address
## GSM74493 University of Michigan, SPH II, 1415 Washington Heights, Room M2533
## GSM74494 University of Michigan, SPH II, 1415 Washington Heights, Room M2533
## GSM74495 University of Michigan, SPH II, 1415 Washington Heights, Room M2533
## GSM74496 University of Michigan, SPH II, 1415 Washington Heights, Room M2533
## GSM74497 University of Michigan, SPH II, 1415 Washington Heights, Room M2533
## GSM74498 University of Michigan, SPH II, 1415 Washington Heights, Room M2533
##          contact_city contact_state contact_zip/postal_code contact_country
## GSM74493    Ann Arbor            MI              48109-2029             USA
## GSM74494    Ann Arbor            MI              48109-2029             USA
## GSM74495    Ann Arbor            MI              48109-2029             USA
## GSM74496    Ann Arbor            MI              48109-2029             USA
## GSM74497    Ann Arbor            MI              48109-2029             USA
## GSM74498    Ann Arbor            MI              48109-2029             USA
##                                                                supplementary_file
## GSM74493 ftp://ftp.ncbi.nlm.nih.gov/geo/samples/GSM74nnn/GSM74493/suppl/GSM74493.CEL.gz
## GSM74494 ftp://ftp.ncbi.nlm.nih.gov/geo/samples/GSM74nnn/GSM74494/suppl/GSM74494.CEL.gz
## GSM74495 ftp://ftp.ncbi.nlm.nih.gov/geo/samples/GSM74nnn/GSM74495/suppl/GSM74495.CEL.gz
## GSM74496 ftp://ftp.ncbi.nlm.nih.gov/geo/samples/GSM74nnn/GSM74496/suppl/GSM74496.CEL.gz
## GSM74497 ftp://ftp.ncbi.nlm.nih.gov/geo/samples/GSM74nnn/GSM74497/suppl/GSM74497.CEL.gz
## GSM74498 ftp://ftp.ncbi.nlm.nih.gov/geo/samples/GSM74nnn/GSM74498/suppl/GSM74498.CEL.gz
##          data_row_count
## GSM74493          15923
## GSM74494          15923
## GSM74495          15923
## GSM74496          15923
## GSM74497          15923
## GSM74498          15923
```

Vemos que los tres primeros ficheros CEL corresponden al tratamiento control y los otros tres al tratamiento con etanol.

```
# get info from gse
filedata <- pData(phenoData(gse[[1]]))
sampleNames <- rownames(filedata)
# get dataframe with .CEL files names:
file_to_txt <- data.frame(file_name=paste(sampleNames,'.CEL',sep=''),dataDir,filedata)
# Creamos una tabla con las muestras seleccionadas:
write.table(file_to_txt,
            file=file.path(dataDir, "targets.txt"),
            sep="\t",
            row.names=FALSE,
            quote=FALSE)
```

```
# Directory the .tar was saved to:
unpackDir <-  celfilesDir
# Decompress .tar file:
#untar(file.path(dataDir, 'GSE3311_RAW.tar'), exdir = celfilesDir)
# Delete .tar file:
#file.remove(file.path(unpackDir, 'GSE18198_RAW.tar'))

# Decompress all .gz files in our directory:
#for (i in list.files(unpackDir)) {
#  gunzip(file.path(unpackDir, i))
#}
```

Lectura de datos (los cargamos en R):

```
#require(Biobase)
sampleInfo <-read.AnnotatedDataFrame(
  file.path(dataDir,"targets.txt"),
  header = TRUE, row.names = 1)
show(pData(sampleInfo))
```

```
##                                                    dataDir
## GSM74493.CEL C:/Users/G-mo10/Desktop/repos_ado/PEC1_ADO/data
## GSM74494.CEL C:/Users/G-mo10/Desktop/repos_ado/PEC1_ADO/data
## GSM74495.CEL C:/Users/G-mo10/Desktop/repos_ado/PEC1_ADO/data
## GSM74496.CEL C:/Users/G-mo10/Desktop/repos_ado/PEC1_ADO/data
## GSM74497.CEL C:/Users/G-mo10/Desktop/repos_ado/PEC1_ADO/data
## GSM74498.CEL C:/Users/G-mo10/Desktop/repos_ado/PEC1_ADO/data
##                                              title geo_accession
## GSM74493.CEL pancreas, control diet, replicate 1      GSM74493
## GSM74494.CEL pancreas, control diet, replicate 2      GSM74494
## GSM74495.CEL pancreas, control diet, replicate 3      GSM74495
## GSM74496.CEL pancreas, ethanol diet, replicate 1      GSM74496
## GSM74497.CEL pancreas, ethanol diet, replicate 2      GSM74497
## GSM74498.CEL pancreas, ethanol diet, replicate 3      GSM74498
##                              status submission_date last_update_date type
## GSM74493.CEL Public on Jul 01 2006     Sep 15 2005      Sep 15 2005  RNA
## GSM74494.CEL Public on Jul 01 2006     Sep 15 2005      Sep 15 2005  RNA
## GSM74495.CEL Public on Jul 01 2006     Sep 15 2005      Sep 15 2005  RNA
```

```
## GSM74496.CEL Public on Jul 01 2006     Sep 15 2005      Sep 15 2005  RNA
## GSM74497.CEL Public on Jul 01 2006     Sep 15 2005      Sep 15 2005  RNA
## GSM74498.CEL Public on Jul 01 2006     Sep 15 2005      Sep 15 2005  RNA
##              channel_count source_name_ch1      organism_ch1
## GSM74493.CEL             1        pancreas Rattus norvegicus
## GSM74494.CEL             1        pancreas Rattus norvegicus
## GSM74495.CEL             1        pancreas Rattus norvegicus
## GSM74496.CEL             1        pancreas Rattus norvegicus
## GSM74497.CEL             1        pancreas Rattus norvegicus
## GSM74498.CEL             1        pancreas Rattus norvegicus
##                                 characteristics_ch1 molecule_ch1
## GSM74493.CEL pancreas, control diet, male Wistar rat    total RNA
## GSM74494.CEL pancreas, control diet, male Wistar rat    total RNA
## GSM74495.CEL pancreas, control diet, male Wistar rat    total RNA
## GSM74496.CEL pancreas, ethanol diet, male Wistar rat    total RNA
## GSM74497.CEL pancreas, ethanol diet, male Wistar rat    total RNA
## GSM74498.CEL pancreas, ethanol diet, male Wistar rat    total RNA
##                                                                                                  extract_protocol_ch1
## GSM74493.CEL Total RNA was extracted from the tissue lysate using an RNeasy kit (Qiagen).
## GSM74494.CEL Total RNA was extracted from the tissue lysate using an RNeasy kit (Qiagen).
## GSM74495.CEL Total RNA was extracted from the tissue lysate using an RNeasy kit (Qiagen).
## GSM74496.CEL Total RNA was extracted from the tissue lysate using an RNeasy kit (Qiagen).
## GSM74497.CEL Total RNA was extracted from the tissue lysate using an RNeasy kit (Qiagen).
## GSM74498.CEL Total RNA was extracted from the tissue lysate using an RNeasy kit (Qiagen).
##              label_ch1
## GSM74493.CEL    biotin
## GSM74494.CEL    biotin
## GSM74495.CEL    biotin
## GSM74496.CEL    biotin
## GSM74497.CEL    biotin
## GSM74498.CEL    biotin
##                                                                                                  label_protocol_ch1
## GSM74493.CEL 5 Âµg of total RNA was processed to produce biotinylated cRNA targets.
## GSM74494.CEL 5 Âµg of total RNA was processed to produce biotinylated cRNA targets.
## GSM74495.CEL 5 Âµg of total RNA was processed to produce biotinylated cRNA targets.
## GSM74496.CEL 5 Âµg of total RNA was processed to produce biotinylated cRNA targets.
## GSM74497.CEL 5 Âµg of total RNA was processed to produce biotinylated cRNA targets.
## GSM74498.CEL 5 Âµg of total RNA was processed to produce biotinylated cRNA targets.
##              taxid_ch1                   hyb_protocol
## GSM74493.CEL     10116 standard Affymetrix procedures
## GSM74494.CEL     10116 standard Affymetrix procedures
## GSM74495.CEL     10116 standard Affymetrix procedures
## GSM74496.CEL     10116 standard Affymetrix procedures
## GSM74497.CEL     10116 standard Affymetrix procedures
## GSM74498.CEL     10116 standard Affymetrix procedures
##                               scan_protocol                description
## GSM74493.CEL standard Affymetrix procedures Rat pancreas, control diet
## GSM74494.CEL standard Affymetrix procedures Rat pancreas, control diet
## GSM74495.CEL standard Affymetrix procedures Rat pancreas, control diet
## GSM74496.CEL standard Affymetrix procedures Rat pancreas, ethanol diet
## GSM74497.CEL standard Affymetrix procedures Rat pancreas, ethanol diet
## GSM74498.CEL standard Affymetrix procedures Rat pancreas, ethanol diet
##                                              data_processing platform_id
## GSM74493.CEL Ann Arbor quantile-normalized trimmed-mean method      GPL341
```

```
## GSM74494.CEL Ann Arbor quantile-normalized trimmed-mean method      GPL341
## GSM74495.CEL Ann Arbor quantile-normalized trimmed-mean method      GPL341
## GSM74496.CEL Ann Arbor quantile-normalized trimmed-mean method      GPL341
## GSM74497.CEL Ann Arbor quantile-normalized trimmed-mean method      GPL341
## GSM74498.CEL Ann Arbor quantile-normalized trimmed-mean method      GPL341
##              contact_name  contact_email contact_phone      contact_institute
## GSM74493.CEL  Rork,,Kuick rork@umich.edu  734-936-9241 University of Michigan
## GSM74494.CEL  Rork,,Kuick rork@umich.edu  734-936-9241 University of Michigan
## GSM74495.CEL  Rork,,Kuick rork@umich.edu  734-936-9241 University of Michigan
## GSM74496.CEL  Rork,,Kuick rork@umich.edu  734-936-9241 University of Michigan
## GSM74497.CEL  Rork,,Kuick rork@umich.edu  734-936-9241 University of Michigan
## GSM74498.CEL  Rork,,Kuick rork@umich.edu  734-936-9241 University of Michigan
##                                                              contact_address
## GSM74493.CEL University of Michigan, SPH II, 1415 Washington Heights, Room M2533
## GSM74494.CEL University of Michigan, SPH II, 1415 Washington Heights, Room M2533
## GSM74495.CEL University of Michigan, SPH II, 1415 Washington Heights, Room M2533
## GSM74496.CEL University of Michigan, SPH II, 1415 Washington Heights, Room M2533
## GSM74497.CEL University of Michigan, SPH II, 1415 Washington Heights, Room M2533
## GSM74498.CEL University of Michigan, SPH II, 1415 Washington Heights, Room M2533
##              contact_city contact_state contact_zip.postal_code contact_country
## GSM74493.CEL    Ann Arbor            MI              48109-2029             USA
## GSM74494.CEL    Ann Arbor            MI              48109-2029             USA
## GSM74495.CEL    Ann Arbor            MI              48109-2029             USA
## GSM74496.CEL    Ann Arbor            MI              48109-2029             USA
## GSM74497.CEL    Ann Arbor            MI              48109-2029             USA
## GSM74498.CEL    Ann Arbor            MI              48109-2029             USA
##                                                                    supplementary_file
## GSM74493.CEL ftp://ftp.ncbi.nlm.nih.gov/geo/samples/GSM74nnn/GSM74493/suppl/GSM74493.CEL.gz
## GSM74494.CEL ftp://ftp.ncbi.nlm.nih.gov/geo/samples/GSM74nnn/GSM74494/suppl/GSM74494.CEL.gz
## GSM74495.CEL ftp://ftp.ncbi.nlm.nih.gov/geo/samples/GSM74nnn/GSM74495/suppl/GSM74495.CEL.gz
## GSM74496.CEL ftp://ftp.ncbi.nlm.nih.gov/geo/samples/GSM74nnn/GSM74496/suppl/GSM74496.CEL.gz
## GSM74497.CEL ftp://ftp.ncbi.nlm.nih.gov/geo/samples/GSM74nnn/GSM74497/suppl/GSM74497.CEL.gz
## GSM74498.CEL ftp://ftp.ncbi.nlm.nih.gov/geo/samples/GSM74nnn/GSM74498/suppl/GSM74498.CEL.gz
##              data_row_count
## GSM74493.CEL          15923
## GSM74494.CEL          15923
## GSM74495.CEL          15923
## GSM74496.CEL          15923
## GSM74497.CEL          15923
## GSM74498.CEL          15923
```

El contenido del archivo targets se utiliza en la lectura de los datos y la creación del objeto rawData de la clase `affybatch` que contendrá las intensidades "crudas" de cada archivo .CEL.

```
fileNames <- rownames(pData(sampleInfo))
rawData <- read.affybatch(filenames=file.path(celfilesDir,fileNames),
                          phenoData=sampleInfo)
show(rawData)
```

```
## Warning: replacing previous import 'AnnotationDbi::tail' by 'utils::tail' when
## loading 'rae230acdf'

## Warning: replacing previous import 'AnnotationDbi::head' by 'utils::head' when
## loading 'rae230acdf'
```

```
##
```

```
## AffyBatch object
## size of arrays=602x602 features (33 kb)
## cdf=RAE230A (15923 affyids)
## number of samples=6
## number of genes=15923
## annotation=rae230a
## notes=
```

## IV.2. Exploración, control de calidad y normalización.

Veamos los nombres de cada muestra.

```
sampleNames <- filedata$title
sampleNames
```

```
## [1] "pancreas, control diet, replicate 1" "pancreas, control diet, replicate 2"
## [3] "pancreas, control diet, replicate 3" "pancreas, ethanol diet, replicate 1"
## [5] "pancreas, ethanol diet, replicate 2" "pancreas, ethanol diet, replicate 3"
```

Como son muy largos para utilizarlos directamente en los siguientes gráficos, simplificaremos sus nombres:

```
sampleNames <- c('Control_1','Control_2','Control_3','Ethanol_1','Ethanol_3','Ethanol_3')
sampleNames
```

```
## [1] "Control_1" "Control_2" "Control_3" "Ethanol_1" "Ethanol_3" "Ethanol_3"
```

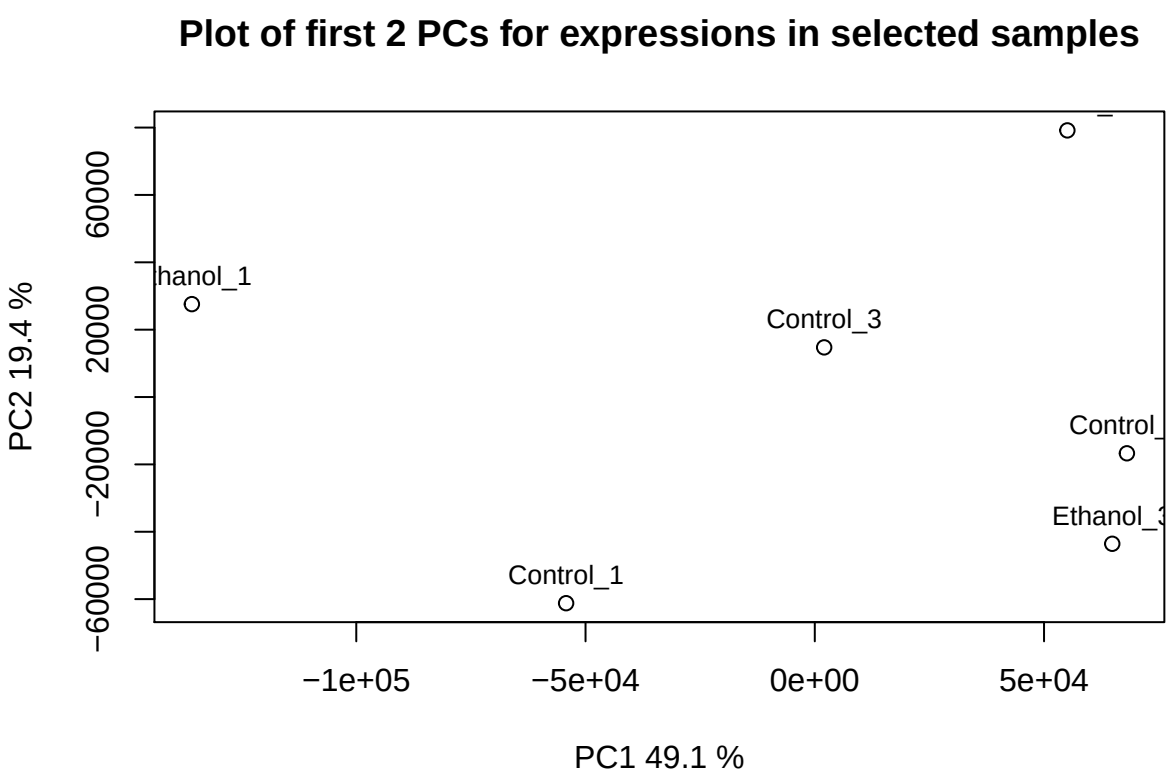
Una forma más computacional de crear este vector sería la siguiente:

```
dummy_control <- c()
for (i in 1:3) {
  dummy_control <- c(dummy_control,paste('Control_',as.character(i),sep = ''))
}

dummy_ethanol <- c()
for (i in 1:3) {
  dummy_ethanol <- c(dummy_ethanol,paste('Ethanol_',as.character(i),sep = ''))
}

sampleNames <- c(dummy_control,dummy_ethanol)
sampleNames
```

```
## [1] "Control_1" "Control_2" "Control_3" "Ethanol_1" "Ethanol_2" "Ethanol_3"
```

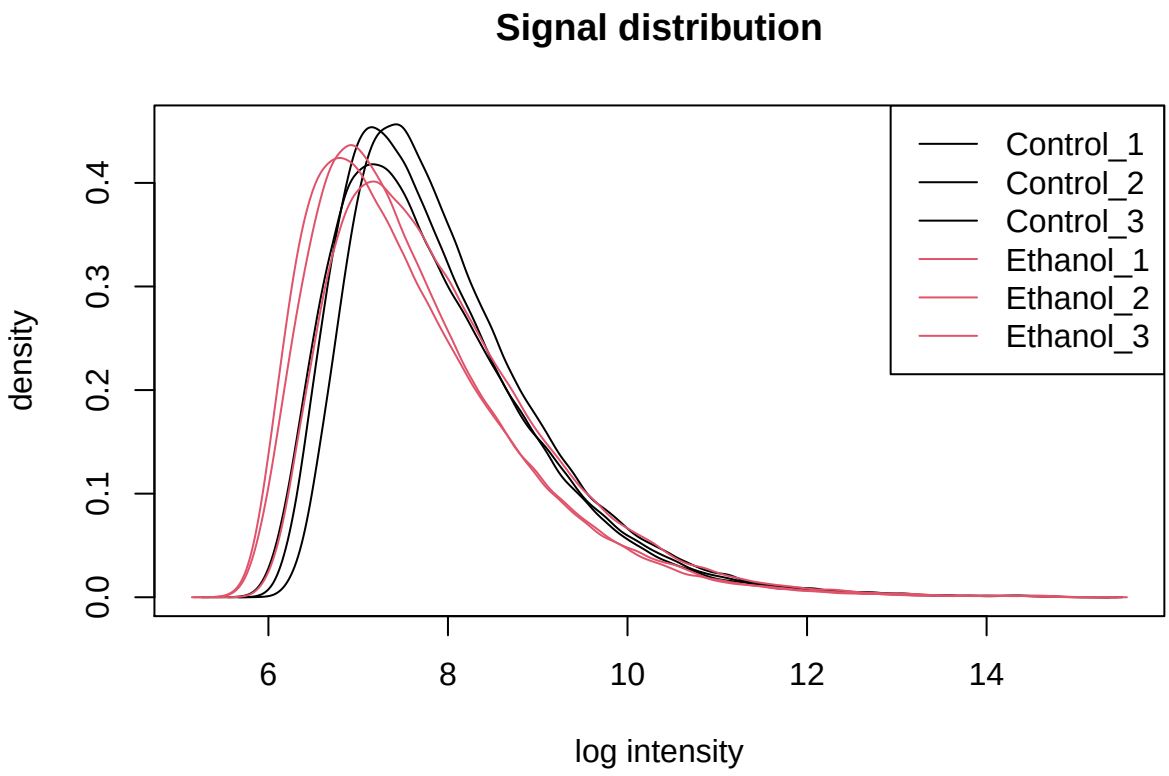
Podemos visualizar los datos con varios tipos de gráficos.

```
plotPCA(exprs(rawData), labels=sampleNames, dataDesc="selected samples")
```



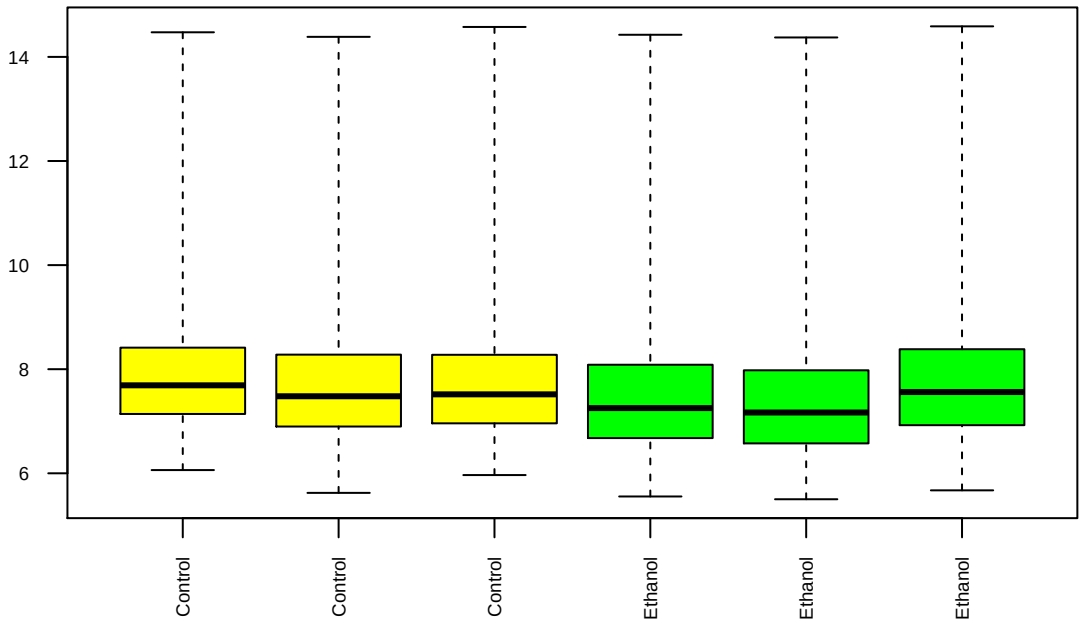

Distribución de señales:

```
info <- data.frame(grupo=c(1,1,1,2,2,2))

hist(rawData, main="Signal distribution", col=info$grupo, lty=1:ncol(info))
legend (x="topright",
        legend=sampleNames,#c(rep('Control',3),rep('Ethanol',3)) ,
        col=info$grupo, lty=1:ncol(info)
        )
```



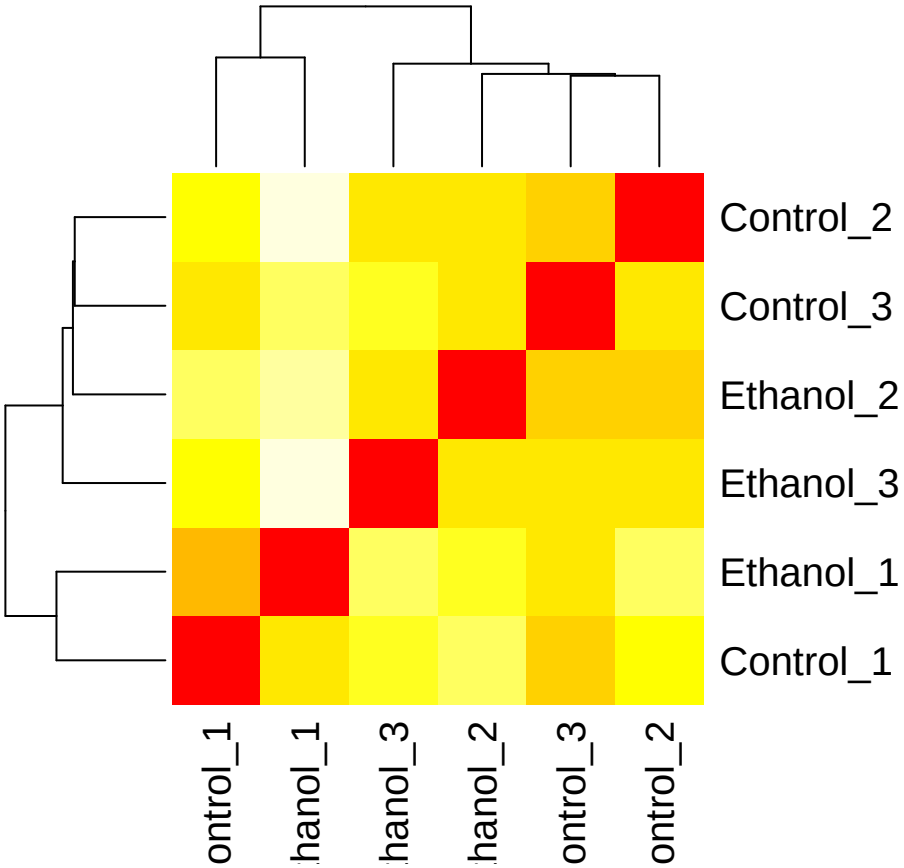

Boxplot:

```
colores <- c(rep("yellow", 3), rep("green", 3))
boxplot(rawData, cex.axis=0.6, col=colores, las=2,
        names=c(rep('Control',3),rep('Ethanol',3)),
        main="Signal distribution for diets")
```



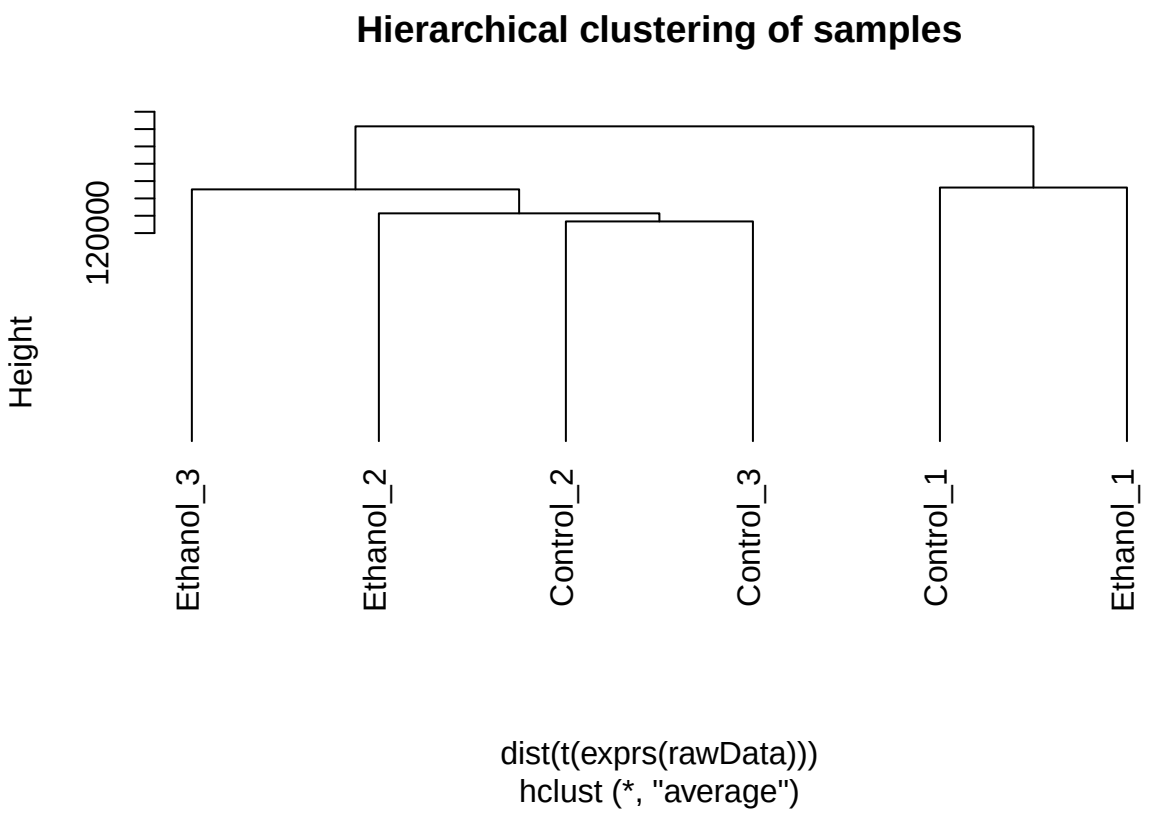

Heatmap:

```
#heatmap:
manDist <-  dist(t(exprs(rawData)))
heatmap (as.matrix(manDist),  col=heat.colors(16),
         labRow = sampleNames, labCol = sampleNames)
```




Dendrograma:

```
## ----plotDendro
clust.euclid.average <- hclust(dist(t(exprs(rawData))),method="average")
plot(clust.euclid.average, labels=sampleNames, main="Hierarchical clustering of samples",  hang=-1)
```




### IV.2.1. Control de calidad.

El paquete affyQCReport encapsula los análisis que pueden realizarse con el paquete simpleaffy, de forma que con una instrucción se pueden realizar todos los análisis y enviar la salida a un archivo.

```
stopifnot(require(affyQCReport))
QCReport(rawData,file=file.path(resultsDir,"QCReport.pdf"))
```

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

El paquete affyPLM realiza un control de calidad basado en lo que se conoce como modelos a nivel de sonda o probe-level models, conocidos por sus siglas (PLM).

```
stopifnot(require(affyPLM))
```

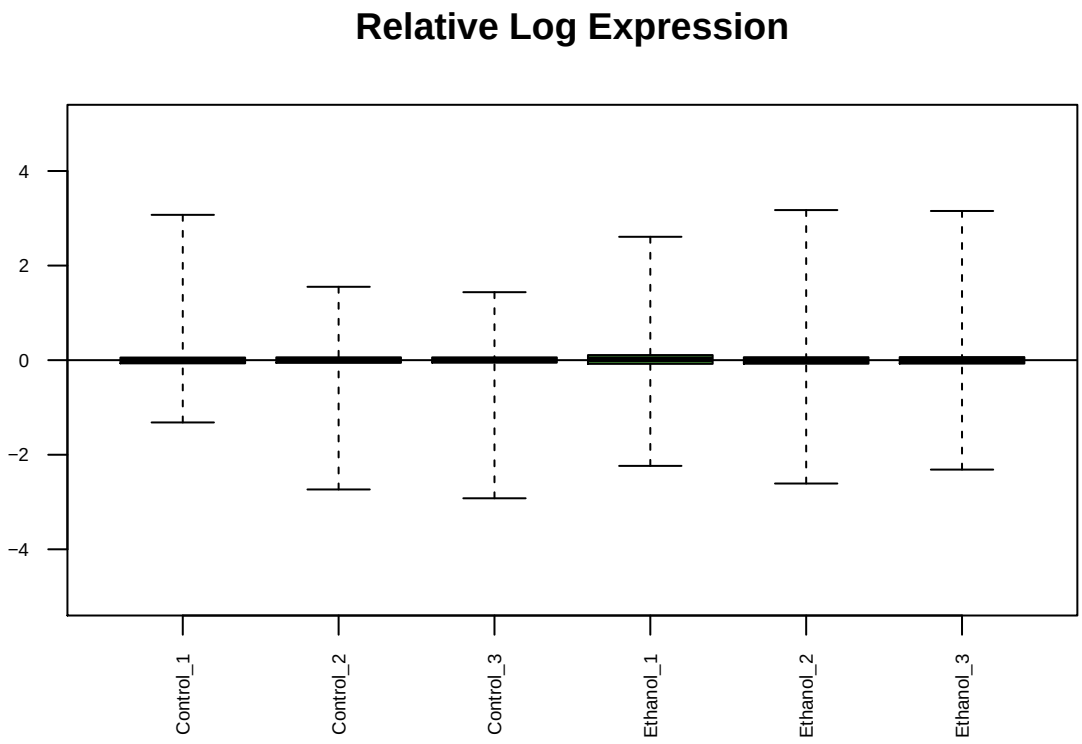
```
## Loading required package: affyPLM

## Loading required package: gcrma

## Loading required package: preprocessCore

##
## Attaching package: 'affyPLM'
```

```
## The following objects are masked from 'package:oligo':
##
##     coefs.probe, NUSE, RLE, se, se.probe
```

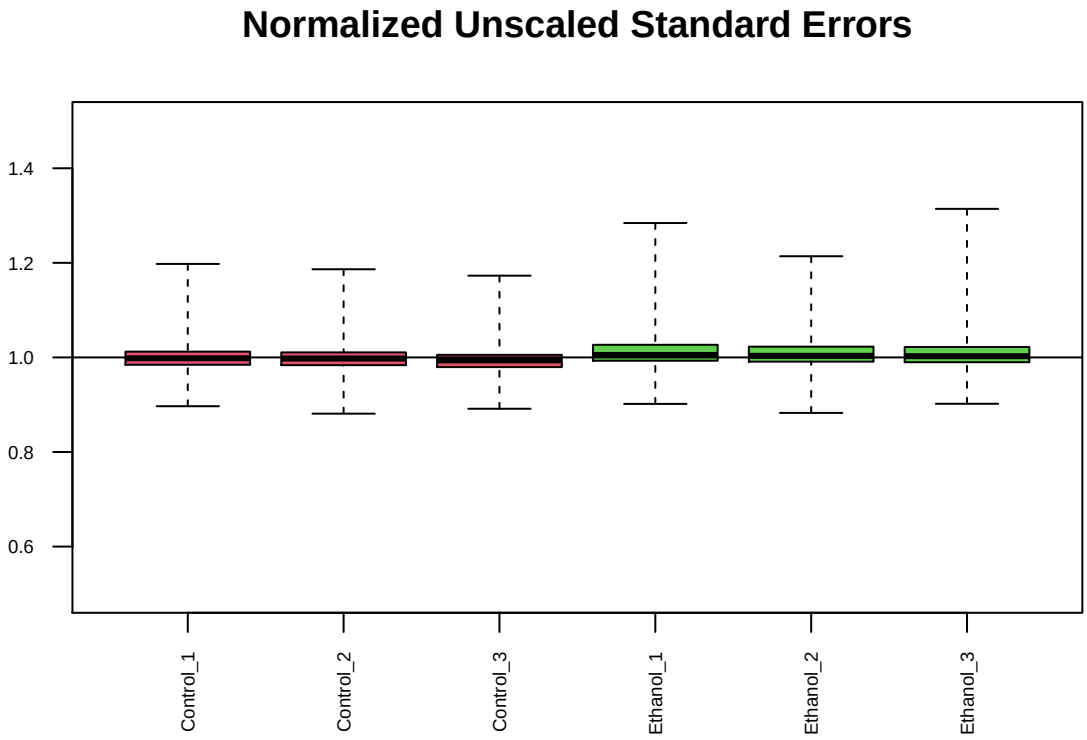
```
computePLM <- T
if(computePLM){
  Pset<- fitPLM(rawData)
  save(Pset, file=file.path(dataDir,"PLM.Rda"))
}else{
  load (file=file.path(dataDir,"PLM.Rda"))
}
```

Como resultado del ajuste PLM se pueden obtener dos gráficos, uno de expresiones relativas y otro con errores estandarizados (figura 27). Si los datos son de calidad, ambos gráficos deben ser centrados y relativamente simétricos. Cambios en esta situación sugieren problemas en los arrays que no los verifiquen.

```
RLE(Pset, main = "Relative Log Expression", names=sampleNames, las=2, col=info$grupo+1, cex.axis=0.6,yl
```



```
NUSE(Pset, main = "Normalized Unscaled Standard Errors", las=2, names=sampleNames, col=info$grupo+1, ce
```




### IV.2.2. Normalización.

El procesado mediante RMA implica un proceso en tres etapas:

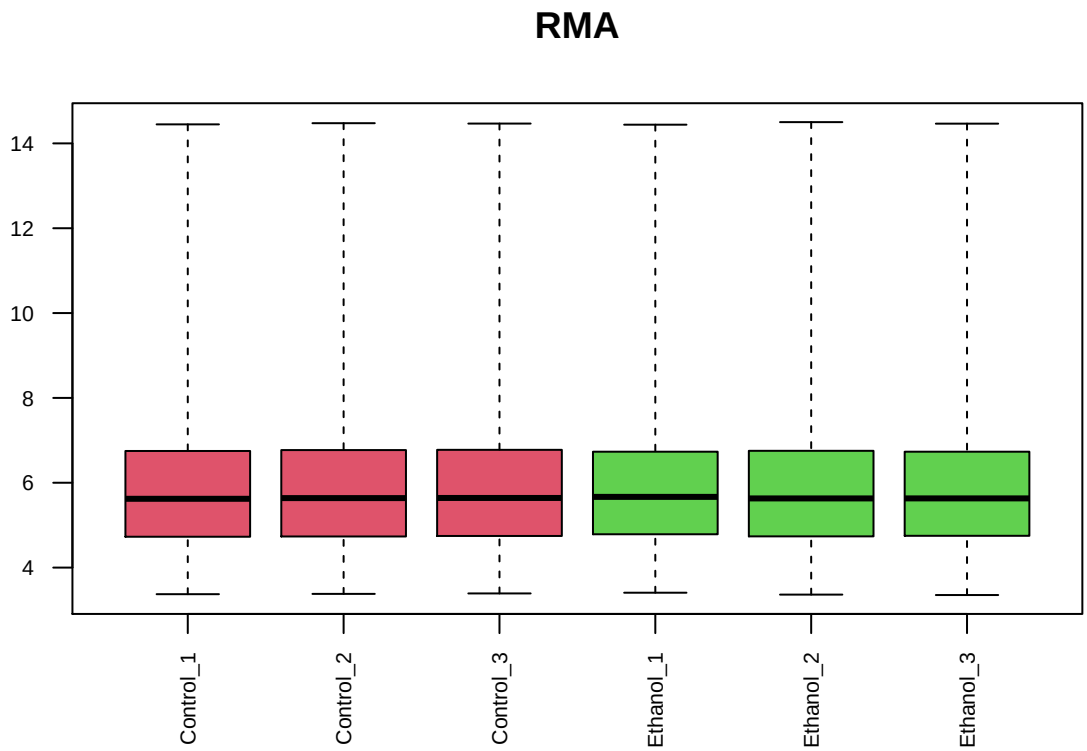
- Corrección de fondo (el RMA hace precisamente esto).
- Normalización para hacer los valores de los arrays comparables.
- Resumen (sumarización) de las diversas sondas asociadas a cada grupo de sondas para dar un único valor.

```
stopifnot(require(affy))
normalize <- T
if(normalize){
  eset_rma <- affy::rma(rawData)
  save(eset_rma, file=file.path(dataDir,"normalized.Rda"))
}else{
  load (file=file.path(dataDir,"normalized.Rda"))
}
```

```
## Background correcting
## Normalizing
## Calculating Expression
```

Un boxplot de los valores normalizados sugiere que los valores ya están en una escala en donde se pueden comparar.

```
boxplot(eset_rma,main="RMA", names=sampleNames, cex.axis=0.7, col=info$grupo+1,las=2)
```




### IV.2.3. Control de calidad de los datos normalizados.

El paquete affyQCReport encapsula los análisis que pueden realizarse con el paquete simpleaffy, de forma que con una instrucción se pueden realizar todos los análisis y enviar la salida a un archivo.

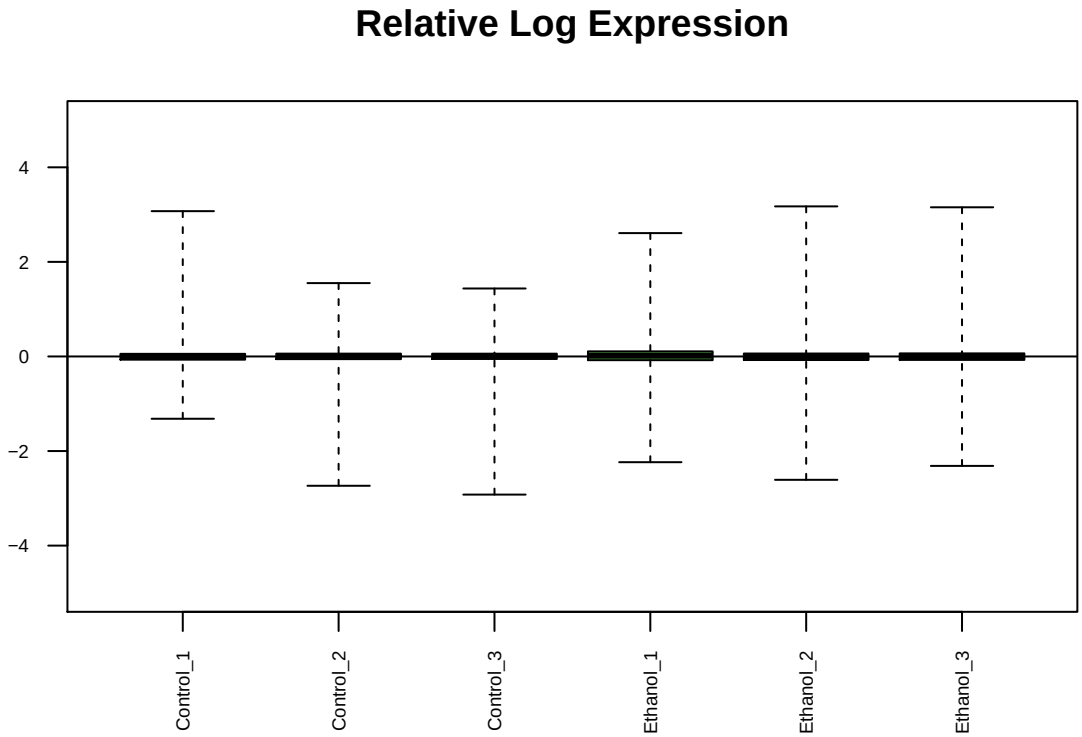
```
# Guardamos el informe en el fichero de resultados:
#QCReport(ReadAffy(eset_rma),file=file.path(resultsDir,"QCReport_normalized.pdf"))
```

El paquete affyPLM realiza un control de calidad basado en lo que se conoce como modelos a nivel de sonda o probe-level models, conocidos por sus siglas (PLM).

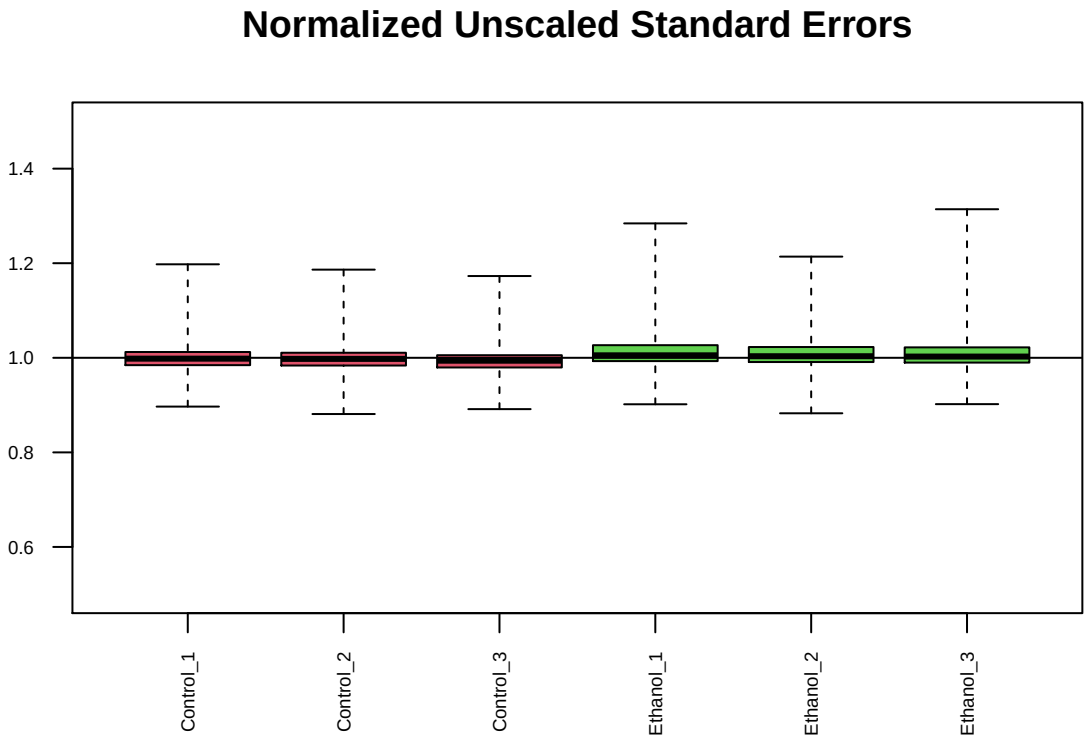
```
#stopifnot(require(affyPLM))
#computePLM <- T
#if(computePLM){
#  Pset<- fitPLM(eset_rma)
#  save(Pset, file=file.path(dataDir,"PLM_normalized.Rda"))
#}else{
#  load (file=file.path(dataDir,"PLM_normalized.Rda"))
#}
```

Como resultado del ajuste PLM se pueden obtener dos gráficos, uno de expresiones relativas y otro con errores estandarizados (figura 27). Si los datos son de calidad, ambos gráficos deben ser centrados y relativamente simétricos. Cambios en esta situación sugieren problemas en los arrays que no los verifiquen.

```
RLE(Pset, main = "Relative Log Expression", names=sampleNames, las=2, col=info$grupo+1, cex.axis=0.6,yl
```




```
NUSE(Pset, main = "Normalized Unscaled Standard Errors", las=2, names=sampleNames, col=info$grupo+1, ce
```




### IV.2.4. Filtraje.

El filtraje no específico permite eliminar los genes que varían poco entre condiciones o que deseamos quitar por otras razones, como por ejemplo que no disponemos de anotación para ellos. La función nsFilter permite eliminar los genes que, o bien varían poco, o bien no se dispone de anotación para ellos.

Si al filtrar deseamos usar las anotaciones, o la falta de ellas, como criterio de filtraje, debemos disponer del correspondiente paquete de anotaciones.

En nuestro caso, tenemos que instalar el paquete 'rae230a.db':

```
if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")

BiocManager::install("rae230a.db")
```

```
## Bioconductor version 3.11 (BiocManager 1.30.10), R 4.0.2 (2020-06-22)

## Installing package(s) 'rae230a.db'

## installing the source package 'rae230a.db'

## Installation path not writeable, unable to update packages: codetools,
##   KernSmooth, nlme
```

```
## Old packages: 'backports', 'cpp11', 'data.table', 'digest', 'ff', 'rlang',
##   'tibble'
```

```
filtered <- nsFilter(eset_rma, require.entrez=TRUE,
                     remove.dupEntrez=TRUE, var.func=IQR,
                     var.cutoff=0.5, var.filter=TRUE,
                     filterByQuantile=TRUE, feature.exclude="^AFFX")
```

```
##
```

```
##
```

```
class(filtered)
```

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

```
names(filtered)
```

```
## [1] "eset"       "filter.log"
```

```
dim(exprs(filtered$eset))
```

```
## [1] 5226    6
```

La función `nsFilter` devuelve los valores filtrados en un objeto `expressionSet` y un informe de los resultados del filtraje.

```
class(filtered$eset)
```

```
## [1] "ExpressionSet"
## attr(,"package")
## [1] "Biobase"
```

```
print(filtered$filter.log)
```

```
## $numDupsRemoved
## [1] 2347
##
## $numLowVar
## [1] 5227
##
## $numRemoved.ENTREZID
## [1] 3117
##
## $feature.exclude
## [1] 6
```

```
eset_filtered <-filtered$eset
```

Podemos grabar el objeto `eset_rma` y los datos filtrados para su posterior uso.

```
# Guardamos en el directorio de resultados:
save(eset_rma, eset_filtered, file=file.path(resultsDir, "estrogen-normalized.Rda"))
```

Después del filtraje han quedado 5227 genes disponibles para analizar.

## IV.3. Selección de genes diferencialmente expresados.

### IV.3.1. Análisis basado en modelos lineales.

**Matriz de diseño**:

Manualmente:

Tomamos como modelo lineal el siguiente:

$$Y_{ij} = \alpha_i + \epsilon_{ij}$$

Donde $\alpha_i$ tiene en cuenta el tipo de tratamiento, es decir:

$$\begin{cases} \alpha_1 = (Control) \\ \alpha_2 = (Ethanol) \end{cases}$$

Por tanto, la matriz de diseño es:

$$\begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \end{pmatrix}$$

La escribimos en forma de código:

```
design.1<-matrix(
c(1,1,1,0,0,0,
  0,0,0,1,1,1),
nrow=6,
byrow=F)
colnames(design.1)<-c("Control", "Ethanol")
rownames(design.1) <-  sampleNames
print(design.1)
```

```
##           Control Ethanol
## Control_1       1       0
## Control_2       1       0
## Control_3       1       0
## Ethanol_1       0       1
## Ethanol_2       0       1
## Ethanol_3       0       1
```

Computacionalmente, utilizando el paquete `limma`.

```
if (!exists("eset_rma"))  load(file.path(dataDir, "normalized.rda"))
targets <- pData(eset_rma)
```

```
lev<-factor(c(rep('Control',3),rep('Ethanol',3)),
            levels=c('Control','Ethanol')
            )
design <-model.matrix(~0+lev)
colnames(design)<-levels(lev)
rownames(design) <- sampleNames
print(design)
```

```
##           Control Ethanol
## Control_1       1       0
## Control_2       1       0
## Control_3       1       0
## Ethanol_1       0       1
## Ethanol_2       0       1
## Ethanol_3       0       1
## attr(,"assign")
## [1] 1 1
## attr(,"contrasts")
## attr(,"contrasts")$lev
## [1] "contr.treatment"
```

Vemos que coincide con nuestra propuesta.

**Contrastes**:

En este caso, nos interesa estudiar el efecto del etanol en la dieta de los ratones, es decir, la diferencia entre la dieta con etanol y la dieta control.

Lo podemos formular como:

$$\beta = \alpha_2 - \alpha_1 \rightarrow efecto\ del\ etanol$$

La matriz de contraste correspondiente es, simplemente:

$$\begin{pmatrix} 1 & -1 \end{pmatrix}$$

Esta se puede hallar también de forma computacional:

```
cont.matrix <- makeContrasts (
  Diet = Ethanol - Control,
  levels=design
  )
cont.matrix
```

```
##          Contrasts
## Levels    Diet
##   Control   -1
##   Ethanol    1
```

También coincide con nuestra matriz de contraste.

**Estimación del modelo y selección de genes.**

Una vez definida la matriz de diseño y los contrastes, podemos pasar a estimar el modelo, estimar los contrastes y realizar las pruebas de significación que nos indiquen, para cada gen y cada comparación, si puede considerarse diferencialmente expresado.

El análisis proporciona los estadísticos de test habituales como *Fold-change*, t-moderados o p-valores ajustados, que se utilizan para ordenar los genes de más a menos diferencialmente expresados.

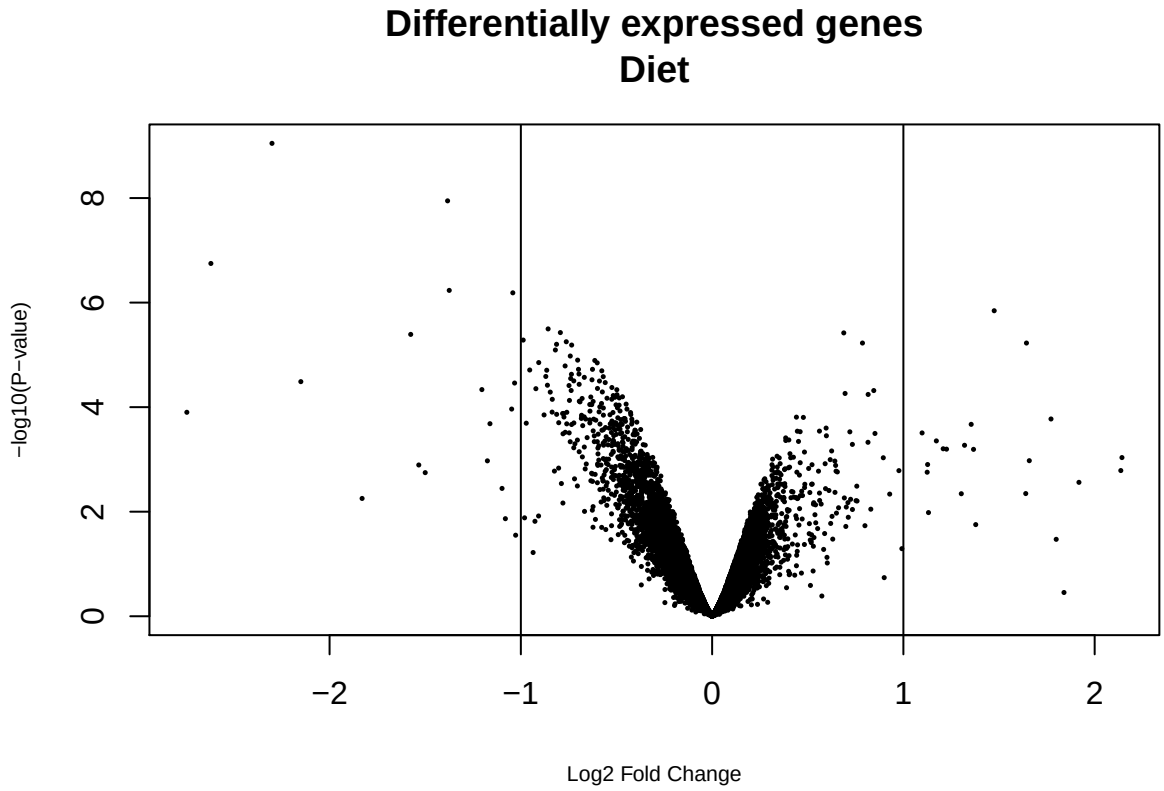
La función `topTable` genera para cada contraste una lista de genes ordenados de más a menos diferencialmente expresados.

```
# Linear model fit:

fit<-lmFit(eset_rma, design)
fit.main<-contrasts.fit(fit, cont.matrix)
fit.main<-eBayes(fit.main)
```

```
topTab  <- topTable (fit.main, number=nrow(fit.main) , coef="Diet", adjust="fdr")
```

Una forma de visualizar los resultados es mediante un *volcano plot*, que representa en abscisas los cambios de expresión en escala logarítmica, y en ordenadas el "menos logaritmo" del p-valor o alternativamente el estadístico B. Los genes cuyo *log odds* es superior a 0 y cuyo *log fold change* es, en valor absoluto, superior a 1, son candidatos a estar diferencialmente expresados.

```
coefnum = 1
opt <- par(cex.lab = 0.7)
volcanoplot(fit.main, coef=coefnum, highlight=0, names=rownames(fit.main),
            main=paste("Differentially expressed genes",colnames(cont.matrix)[coefnum], sep="\n"))
abline(v=c(-1,1))
```




Vemos que hay varios puntos a la izquierda de -1 y a la derecha de 1.

### IV.3.2. Comparaciones múltiples.

Cuando se realizan varias comparaciones a la vez puede resultar importante ver qué genes cambian simultáneamente en más de una comparación. Si el número de comparaciones es alto, también puede ser necesario realizar un ajuste de p-valores entre las comparaciones, distinto del realizado entre genes.

La función `decidetests` permite realizar ambas cosas. En este caso no se ajustarán los p-valores entre comparaciones. Tan solo se seleccionarán los genes que cambian en una o más condiciones.

El resultado del análisis es una tabla, que llamaremos `res` y que para cada gen y cada comparación contiene un 1 (si el gen esta sobreexpresado o *up* en esta condición), un 0 (si no hay cambio significativo) o un -1 (si está *down* regulado).

```
require('rae230a.db')
anotPackage <- annotation(eset_rma)
fit.Symbols <- getSYMBOL (rownames(fit.main), anotPackage)
res<-decideTests(fit.main, method="separate", adjust.method="fdr", p.value=0.01)
```

```
sum.res.rows<-apply(abs(res),1,sum)
res.selected<-res[sum.res.rows!=0,]
print(summary(res))
```

```
##         Diet
## Down      14
```

```
## NotSig 15905
## Up         4
```

### IV.3.3. Anotación de resultados.

Para saber qué anotaciones están disponibles, debe cargarse el paquete y llamar la función del mismo nombre.

```
rae230a()
```

```
## Quality control information for rae230a:
##
##
## This package has the following mappings:
##
## rae230aACCNUM has 15923 mapped keys (of 15923 keys)
## rae230aALIAS2PROBE has 19462 mapped keys (of 64537 keys)
## rae230aCHR has 12804 mapped keys (of 15923 keys)
## rae230aCHRLENGTHS has 953 mapped keys (of 953 keys)
## rae230aCHRLOC has 11680 mapped keys (of 15923 keys)
## rae230aCHRLOCEND has 11680 mapped keys (of 15923 keys)
## rae230aENSEMBL has 12146 mapped keys (of 15923 keys)
## rae230aENSEMBL2PROBE has 9971 mapped keys (of 21000 keys)
## rae230aENTREZID has 12810 mapped keys (of 15923 keys)
## rae230aENZYME has 1989 mapped keys (of 15923 keys)
## rae230aENZYME2PROBE has 814 mapped keys (of 960 keys)
## rae230aGENENAME has 12810 mapped keys (of 15923 keys)
## rae230aGO has 12106 mapped keys (of 15923 keys)
## rae230aGO2ALLPROBES has 18932 mapped keys (of 22754 keys)
## rae230aGO2PROBE has 14506 mapped keys (of 18279 keys)
## rae230aPATH has 4832 mapped keys (of 15923 keys)
## rae230aPATH2PROBE has 225 mapped keys (of 225 keys)
## rae230aPMID has 10955 mapped keys (of 15923 keys)
## rae230aPMID2PROBE has 84668 mapped keys (of 157386 keys)
## rae230aREFSEQ has 12800 mapped keys (of 15923 keys)
## rae230aSYMBOL has 12810 mapped keys (of 15923 keys)
## rae230aUNIGENE has 11999 mapped keys (of 15923 keys)
## rae230aUNIPROT has 11757 mapped keys (of 15923 keys)
##
##
## Additional Information about this package:
##
## DB schema: RATCHIP_DB
## DB schema version: 2.1
## Organism: Rattus norvegicus
## Date for NCBI data: 2015-Sep27
## Date for GO data: 20150919
## Date for KEGG data: 2011-Mar15
## Date for Golden Path data: 2014-Aug1
## Date for Ensembl data: 2015-Jul16
```

#### Tablas de anotación sencillas.

El paquete `annafy` permite de forma muy simple generar una tabla de anotaciones con hiperenlaces a las bases de datos para cada anotación seleccionada.

```
#require(annaffy)
genesSelected <- rownames(res.selected)
at <- aafTableAnn(genesSelected, "rae230a.db")
```

```
## Warning in chkPkgs(chip): The rae230a.db package does not appear to contain
## annotation data.
```

```
## Warning in result_fetch(res@ptr, n = n): SQL statements must be issued with
## dbExecute() or dbSendStatement() instead of dbGetQuery() or dbSendQuery().
```

```
## Warning in result_fetch(res@ptr, n = n): SQL statements must be issued with
## dbExecute() or dbSendStatement() instead of dbGetQuery() or dbSendQuery().
```

```
saveHTML (at, file.path(resultsDir, "anotations.html"),
          "Annotations for selected genes")
```

### IV.3.4. Visualización de los perfiles de expresión.

Tras seleccionar los genes diferencialmente expresados, podemos visualizar las expresiones de cada gen agrupándolas para destacar los genes que se encuentran up o down regulados simultáneamente constituyendo perfiles de expresión.

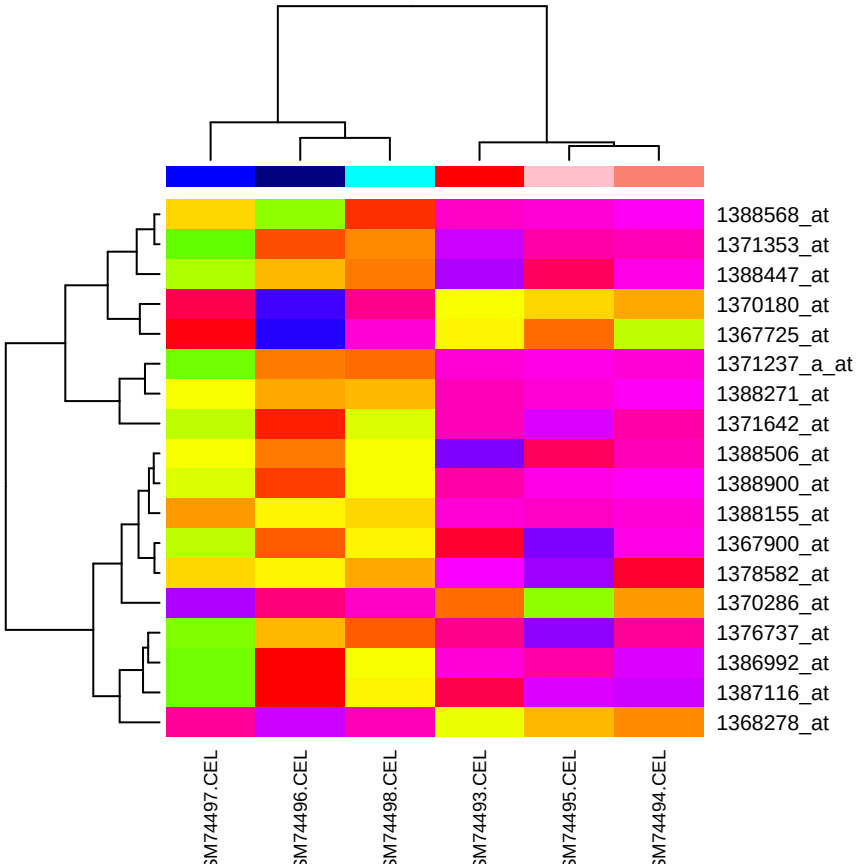
Hay distintas formas de visualización, pero aquí tan solo se presenta el uso de mapas de color o Heatmaps.

En primer lugar seleccionamos los genes a visualizar: se toman todos aquellos que han resultado diferencialmente expresados en alguna de las tres comparaciones.

```
probeNames<-rownames(res)
probeNames.selected<-probeNames[sum.res.rows!=0]
exprs2cluster <-exprs(eset_rma)[probeNames.selected,]
```

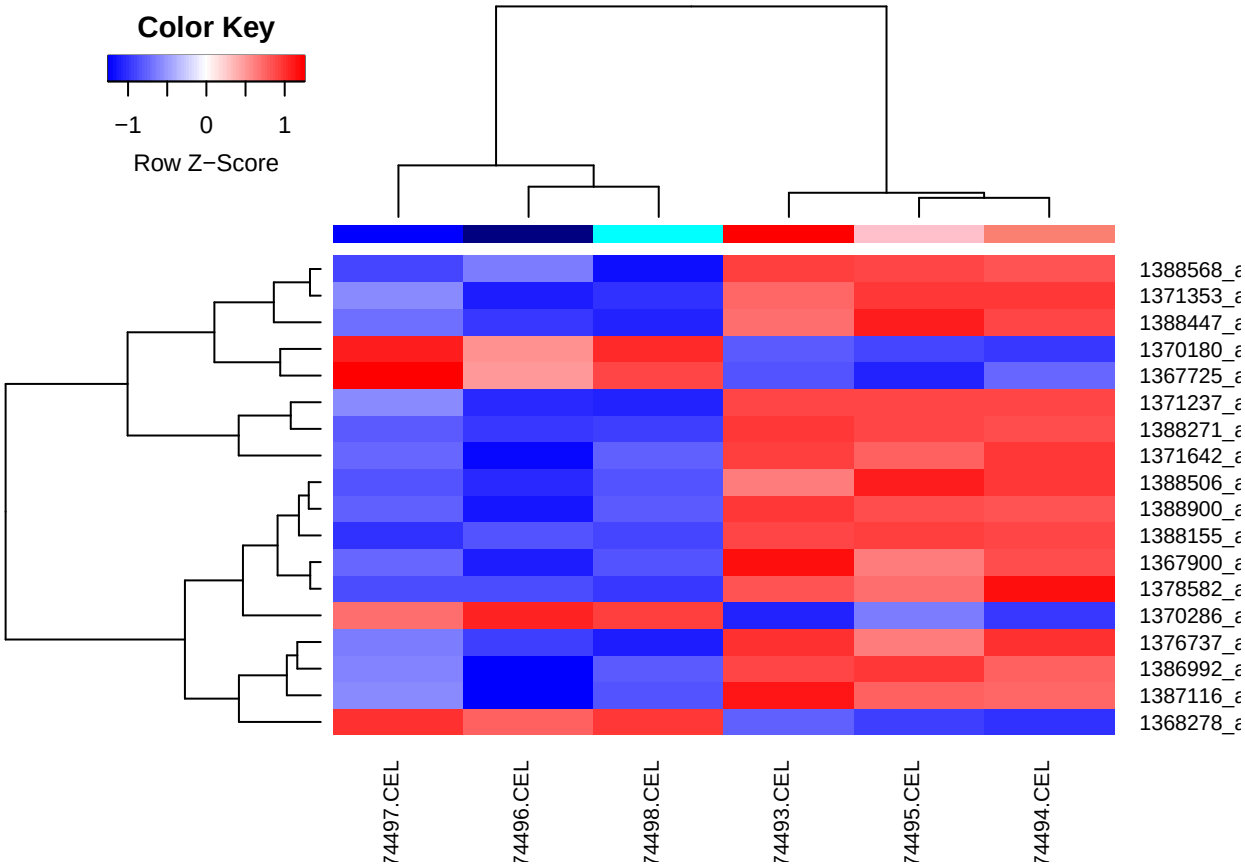
Para representar el Heatmap solo necesitamos la matriz de datos resultante.

```
grupColors <- c('red','salmon','pink','navy','blue','cyan')
heatmap(exprs2cluster, col=rainbow(100), ColSideColors=grupColors, cexCol=0.9)
```




Si se desea realizar mapas de color más sofisticados, puede utilizarse el paquete gplots que implementa una versión mejorada en la función `heatmap.2`.

```
#grupColors <- unlist(lapply(pData(eset_rma)$dataDir, color.map))

#require("gplots")

heatmap.2(exprs2cluster,
          col=bluered(75), scale="row",
          ColSideColors=grupColors, key=TRUE, symkey=FALSE,
          density.info="none", trace="none", cexCol=1)
```




### IV.3.5. Análisis de significación biológica.

Vamos a estudiar las funciones de los genes buscando sus anotaciones en bases de datos de anotación funcional como la Gene Ontology (GO).

Antes de empezar a hacer inferencias sobre el significado de una anotación debería poderse establecer si dicha anotación está relacionada con el proceso que se está estudiando o aparece por azar entre la muchas anotaciones de losgenes de la lista.

Para ello, realizamos un análisis de enriquecimiento con Bioconductor.

```
require(GOstats)
```

```
## Loading required package: GOstats

## Loading required package: Category

## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following object is masked from 'package:S4Vectors':
##
##     expand
```

```
## Loading required package: graph

##
## Attaching package: 'graph'

## The following object is masked from 'package:Biostrings':
##
##     complement

## The following object is masked from 'package:XML':
##
##     addNode

##
## Attaching package: 'GOstats'

## The following object is masked from 'package:AnnotationDbi':
##
##     makeGOGraph
```

```
require(rae230a.db)


  # Seleccionamos la "topTable"
  topTab <- topTab
  # Definimos el universo de genes: todos los que se han incluido en el analisis
  # EL programa trabaja con identificadores "entrez" y no admite duplicados

  entrezUniverse <- unique(getEG(as.character(rownames(topTab)), "rae230a.db"))

  # Filtramos posibles NA:
  entrezUniverse <- entrezUniverse[!is.na(entrezUniverse)]

  # Escogemos los grupos de sondas a incluir en el analisis
  # Este analisis trabaja bien con varios centenares de genes
  # por lo que es habitual basarse en p-valores sin ajustar para incluirlos

  whichGenes<-topTab["adj.P.Val"]<0.001
  geneIds <-   unique(getEG(as.character(rownames(topTab)[whichGenes[,1]]),"rae230a.db"))

  # Filtramos posibles NA:
  geneIds <- geneIds[!is.na(geneIds)]

  # Creamos los "hiperparametros" en que se basa el analisis
  GOparams = new("GOHyperGParams",
    geneIds=geneIds, universeGeneIds=entrezUniverse,
    annotation="rae230a.db", ontology="BP",
    pvalueCutoff=0.001, conditional=FALSE,
    testDirection="over")
  KEGGparams = new("KEGGHyperGParams",
    geneIds=geneIds, universeGeneIds=entrezUniverse,
    annotation="rae230a.db",
```

```
    pvalueCutoff=0.01, testDirection="over")

  # Ejecutamos los analisis

  GOhyper = hyperGTest(GOparams)
  KEGGhyper = hyperGTest(KEGGparams)

# Creamos un informe html con los resultados
   comparison = "Diet"
   GOfilename =file.path(resultsDir,
     paste("GOResults.",comparison,".html", sep=""))
   KEGGfilename =file.path(resultsDir,
     paste("KEGGResults.",comparison,".html", sep=""))
  htmlReport(GOhyper, file = GOfilename, summary.args=list("htmlLinks"=TRUE))
  htmlReport(KEGGhyper, file = KEGGfilename, summary.args=list("htmlLinks"=TRUE))
```

```
## Warning: No results met the specified criteria. Returning 0-row data.frame
```

```
## Warning in htmlReportFromDf(r = df, caption = paste(label, description(r)), : No
## rows to report. Skipping
```

# V.Discusión.

En este análisis hemos visto que existen diferencias significativas entre la expresión génica de los grupos según estos fuesen tratados con o sin etanol. Sin embargo, al principio del análisis vimos cómo en el dendograma no se distinguían los grupos según su tratamiento.

Una limitación importante de nuestro análisis es que en el análisis de significación biológica sólo se ha hallado una única entrada de Gene Ontology. Quizás cabería la posibilidad de ser más flexibles con nuestro p-valor, hasta un 1% o 5%.