# Apéndice

## A1. Identificación de las muestras y los grupos

Para preparar el ambiente, se han creado las carpetas requeridas para el análisis, los datos y los resultados dentro de un directorio denominado ‘*PEC1’* dentro de la carpeta de la asignatura.

dir.create("MicroArrayAnalisis")

dir.create("Datos")

dir.create("Resultados")

Se han descargado los datos del estudio GSE55662 de la base de datos GEO y posteriormente se han guardado en la nueva carpeta ‘*Datos’*. Seguidamente, se ha procedido a crear y leer el archivo ‘*targets.csv’* que contiene toda la información relevante de los archivos .CEL individuales en un solo archivo.

targets.pec <- read.csv2("./Datos/targets.csv", header = TRUE, sep = ";")  
knitr::kable(targets.pec, booktabs = TRUE, caption = "Contenido del archivo targets usado para este análisis")

Tabla A1. Contenido del archivo targets usado para este análisis

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| FileName | Group | Genotype | Treatment | ShortName |
| GSM1341344\_01\_E\_coli\_2\_ | WT\_ntCPR | WT | Not-treated | wt\_nt.1 |
| GSM1341345\_02\_E\_coli\_2\_ | WT\_ntCPR | WT | Not-treated | wt\_nt.2 |
| GSM1341346\_03\_E\_coli\_2\_ | WT\_CPR | WT | CPR | wt\_cpr.1 |
| GSM1341347\_04\_E\_coli\_2\_ | WT\_CPR | WT | CPR | wt\_cpr.2 |
| GSM1341348\_07\_E\_coli\_2\_ | FUR\_ntCPR | fur | Not-treated | fur\_nt.1 |
| GSM1341349\_08\_E\_coli\_2\_ | FUR\_ntCPR | fur | Not-treated | fur\_nt.2 |
| GSM1341350\_09\_E\_coli\_2\_ | FUR\_CPR | fur | CPR | fur\_cpr.1 |
| GSM1341351\_10\_E\_coli\_2\_ | FUR\_CPR | fur | CPR | fur\_cpr.2 |

Finalmente, se procede a leer los archivos .CEL con la función **list.celfiles** del paquete **oligo**:

library(oligo)

## Loading required package: BiocGenerics

## Loading required package: parallel

##   
## Attaching package: 'BiocGenerics'

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

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

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

## Loading required package: oligoClasses

## Welcome to oligoClasses version 1.48.0

## Loading required package: Biobase

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

## Loading required package: Biostrings

## Loading required package: S4Vectors

## Warning: package 'S4Vectors' was built under R version 3.6.2

## Loading required package: stats4

##   
## Attaching package: 'S4Vectors'

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

## Loading required package: IRanges

## Warning: package 'IRanges' was built under R version 3.6.2

##   
## Attaching package: 'IRanges'

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

## Loading required package: XVector

##   
## Attaching package: 'Biostrings'

## The following object is masked from 'package:base':  
##   
## strsplit

## No methods found in package 'RSQLite' for request: 'dbListFields' when loading 'oligo'

## ================================================================================

## Welcome to oligo version 1.50.0

## ================================================================================

archivosCEL <- list.celfiles("./Datos", full.names = TRUE)

A continuación, el archivo ‘*targets.csv’* generado anteriormente es guardado en una nueva variable denominada mis.targets para asociar la información de los archivos .CEL al mismo. Se usa la función **read.AnnotatedDataFrame** del paquete **Biobase**.

library(Biobase)  
mis.targets <- read.AnnotatedDataFrame(file.path("./Datos", "targets.csv"), header = TRUE, row.names = 1, sep=";")  
rawData <- read.celfiles(archivosCEL, phenoData = mis.targets)

## Loading required package: pd.e.coli.2

## Loading required package: RSQLite

## Loading required package: DBI

## Platform design info loaded.

## Reading in : ./Datos/GSM1341344\_01\_E\_coli\_2\_.CEL  
## Reading in : ./Datos/GSM1341345\_02\_E\_coli\_2\_.CEL  
## Reading in : ./Datos/GSM1341346\_03\_E\_coli\_2\_.CEL  
## Reading in : ./Datos/GSM1341347\_04\_E\_coli\_2\_.CEL  
## Reading in : ./Datos/GSM1341348\_07\_E\_coli\_2\_.CEL  
## Reading in : ./Datos/GSM1341349\_08\_E\_coli\_2\_.CEL  
## Reading in : ./Datos/GSM1341350\_09\_E\_coli\_2\_.CEL  
## Reading in : ./Datos/GSM1341351\_10\_E\_coli\_2\_.CEL

## Warning in read.celfiles(archivosCEL, phenoData = mis.targets): 'channel'  
## automatically added to varMetadata in phenoData.

Así, se obtiene el *ExpressionSet* y puede cambiarse el nombre largo de los archivos .CEL por el nombre corto establecido en el archivo ‘*targets.csv’*:

mis.targets@data$ShortName -> rownames(pData(rawData))  
colnames(rawData) <- rownames(pData(rawData))  
head(rawData)

## ExpressionFeatureSet (storageMode: lockedEnvironment)  
## assayData: 1 features, 8 samples   
## element names: exprs   
## protocolData  
## rowNames: wt\_nt.1 wt\_nt.2 ... fur\_cpr.2 (8 total)  
## varLabels: exprs dates  
## varMetadata: labelDescription channel  
## phenoData  
## rowNames: wt\_nt.1 wt\_nt.2 ... fur\_cpr.2 (8 total)  
## varLabels: Group Genotype Treatment ShortName  
## varMetadata: labelDescription channel  
## featureData: none  
## experimentData: use 'experimentData(object)'  
## Annotation: pd.e.coli.2

De las 8 muestras existen 4 grupos distintos, con dos réplicas cada uno, en el siguiente orden: *wt\_nt* (sin tratamiento), *wt\_cpr* (tratado con ciprofloxacina), *fur\_nt* (sin tratamiento), *fur\_cpr* (tratado con ciprofloxacina).

## A2. Control de calidad de los datos crudos

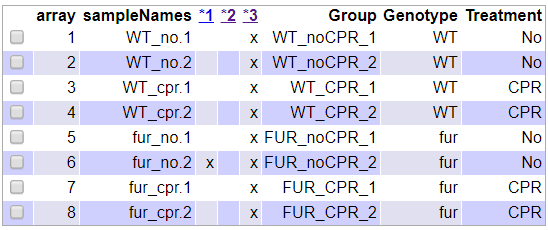
Una vez han sido cargados los datos crudos con el nombre ‘rawData’ puede determinarse la calidad de los mismos antes de su normalización.

library(arrayQualityMetrics)

arrayQualityMetrics(rawData, outdir = file.path("./MicroarrayAnalysis")

A continuación, se observa la tabla resumen del archivo ‘*index.html’* creado del análisis anterior:

TablaA1. Tabla resumen obtenida de la función arrayQualityMetrics de los datos crudos.



Se observa que todas las muestras han sido marcadas al menos una vez y la muestra 6 dos veces.

**Análisis de componentes principales** de los datos crudos:

library(ggplot2)

## Warning: package 'ggplot2' was built under R version 3.6.3

library(ggrepel)

## Warning: package 'ggrepel' was built under R version 3.6.3

plotPCA3 <- function (datos, labels, factor, title, scale, colores, size = 1.5, glineas = 0.25) {  
 data <- prcomp(t(datos),scale=scale)  
 dataDf <- data.frame(data$x)  
 Group <- factor  
 loads <- round(data$sdev^2/sum(data$sdev^2)\*100,1)  
 p1 <- ggplot(dataDf,aes(x=PC1, y=PC2)) +  
 theme\_classic() +  
 geom\_hline(yintercept = 0, color = "gray70") +  
 geom\_vline(xintercept = 0, color = "gray70") +  
 geom\_point(aes(color = Group), alpha = 0.55, size = 3) +  
 coord\_cartesian(xlim = c(min(data$x[,1])-5,max(data$x[,1])+5)) +  
 scale\_fill\_discrete(name = "Group")  
 p1 + geom\_text\_repel(aes(y = PC2 + 0.25, label = labels),segment.size = 0.25, size = size) +   
 labs(x = c(paste("PC1",loads[1],"%")),y=c(paste("PC2",loads[2],"%"))) +   
 ggtitle(paste("Análisis de componentes principales para:",title,sep=" "))+   
 theme(plot.title = element\_text(hjust = 0.5)) +  
 scale\_color\_manual(values=colores)  
}

plotPCA3(exprs(rawData), labels = targets.pec$ShortName, factor = targets.pec$Group,   
 title="Datos crudos", scale = FALSE, size = 3.5,   
 colores = c("darkred", "salmon", "darkgreen", "springgreen2"))

Análisis de componentes principales para: Datos crudos

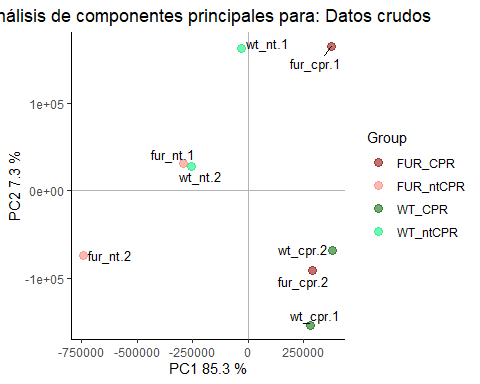


Figura A1. Gráfico de componentes principales para los datos crudos.

Cada color pertenece a cada uno de los cuatro grupos del estudio.

**Diagramas de caja** para visualizar la distribución de las intensidades de los *arrays*:

boxplot(rawData, cex.axis=0.7, las=2, which="all", col = c(rep("springgreen2", 2), rep("darkgreen", 2), rep("salmon", 2), rep("darkred", 2)), main = "Distribución de los valores de intensidad de los datos crudos", cex.main = 0.8)

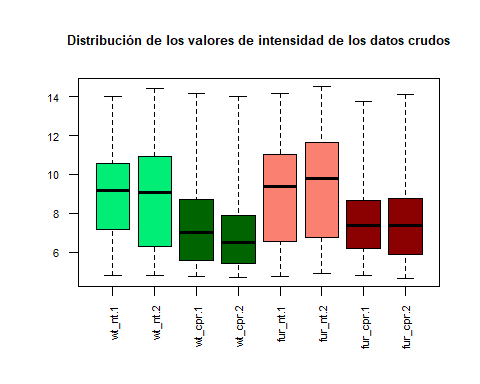


Figura A2. Diagrama de cajas de la distribución de la intensidad de los datos crudos.

**Gráfico de densidad** para observar la distribución de los distintos arrays:

hist(rawData, cex.axis=0.7, las=2, which="all", col = c(rep("springgreen2", 2), rep("darkgreen", 2), rep("salmon", 2), rep("darkred", 2)), main = "Distribución de los valores de intensidad de los datos crudos", cex.main = 0.8)

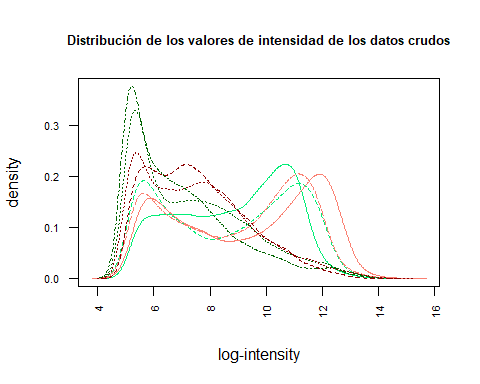


Figura A3. Gráfico de la distribución de la intensidad de los datos crudos.

## A3. Normalización

El proceso de normalización consta generalmente de tres pasos: corrección del ruido de fondo o *background*, normalización y sumarización. El método usado en este análisis es el método *Robust Multichip Average*(RMA):

eset\_rma.pec <- rma(rawData)

## Background correcting  
## Normalizing  
## Calculating Expression

## A4. Control de calidad de los datos normalizados

Una vez realizado el proceso de normalización, se analiza la calidad de los mismos con la función **arrayQualityMetrics**:

arrayQualityMetrics(eset\_rma.pec, outdir = file.path("./Resultados", "QCDir.Norm"), force = TRUE)

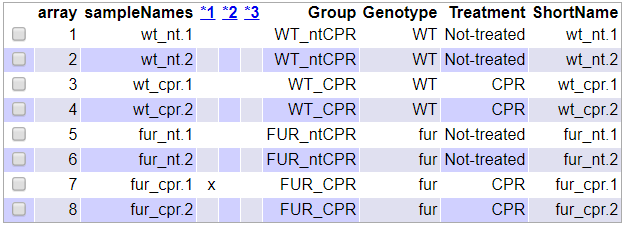
## The report will be written into directory './Resultados/QCDir.Norm'.

## Warning in svgStyleAttributes(style, svgdev): Removing non-SVG style attribute  
## name(s): subscripts, group.number, group.value  
## Warning in svgStyleAttributes(style, svgdev): Removing non-SVG style attribute  
## name(s): subscripts, group.number, group.value  
## Warning in svgStyleAttributes(style, svgdev): Removing non-SVG style attribute  
## name(s): subscripts, group.number, group.value  
## Warning in svgStyleAttributes(style, svgdev): Removing non-SVG style attribute  
## name(s): subscripts, group.number, group.value  
## Warning in svgStyleAttributes(style, svgdev): Removing non-SVG style attribute  
## name(s): subscripts, group.number, group.value  
## Warning in svgStyleAttributes(style, svgdev): Removing non-SVG style attribute  
## name(s): subscripts, group.number, group.value  
## Warning in svgStyleAttributes(style, svgdev): Removing non-SVG style attribute  
## name(s): subscripts, group.number, group.value  
## Warning in svgStyleAttributes(style, svgdev): Removing non-SVG style attribute  
## name(s): subscripts, group.number, group.value

## (loaded the KernSmooth namespace)

A continuación, se observa la tabla resumen del archivo ‘*index.html’* creado del análisis anterior:

TablaA2. Tabla resumen obtenida de la función arrayQualityMetrics de los datos normalizados.



Ahora se observa la muestra 7 marcada una vez y es posible proceder con el análisis.

**Análisis de componentes principales** de los datos normalizados:

plotPCA3(exprs(eset\_rma.pec), labels = targets.pec$ShortName, factor = targets.pec$Group,   
 title="Datos normalizados", scale = FALSE, size = 3.5,   
 colores = c("darkred", "salmon", "darkgreen", "springgreen"))

Análisis de componentes principales para: Datos normalizados

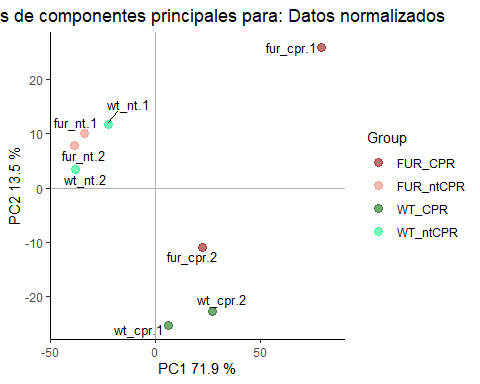


Figura A4. Gráfico de componentes principales para los datos normalizados.

**Diagramas de caja** para visualizar la distribución de las intensidades de los *arrays*:

boxplot(eset\_rma.pec, cex.axis=0.7, las=2, which="all",   
 col = c(rep("springgreen2", 2), rep("darkgreen", 2), rep("salmon", 2), rep("darkred", 2)),  
 main="Diagrama de cajas para la intensidad de los arrays: Datos normalizados", cex.main = 0.8)

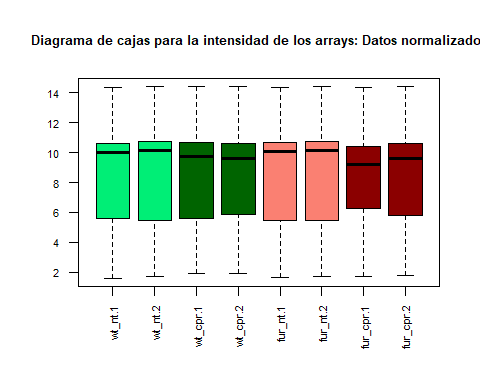


Figura A5. Diagrama de cajas de la distribución de la intensidad de los datos normalizados. Las intensidades de los distintos *arrays* son mucho más parecidas después del proceso de normalización.

**Principal Variation Component Analysis** (PVCA) para identificar y eliminar los efectos de los distintos lotes:

library(pvca)  
pData(eset\_rma.pec) <- targets.pec  
pct\_threshold <- 0.6  
batch.factors <- c("Genotype", "Treatment")  
pvcaObj <- pvcaBatchAssess(eset\_rma.pec, batch.factors, pct\_threshold)

## boundary (singular) fit: see ?isSingular  
## boundary (singular) fit: see ?isSingular  
## boundary (singular) fit: see ?isSingular

bp <- barplot(pvcaObj$dat, xlab = "Efectos", ylab = "Variación de la proporción promedia ponderada", ylim = c(0,1.1), col=c("deepskyblue3"), las=2, main="Estimación PVCA", xlim = c(0,8), cex.main = 1.1, cex.lab = 0.9)  
axis(1, at = bp, labels = pvcaObj$label, cex.axis = 0.8, las=2)  
values = pvcaObj$dat  
new\_values = round(values, 3)  
text(bp, pvcaObj$dat, labels = new\_values, pos =3, cex = 0.9)

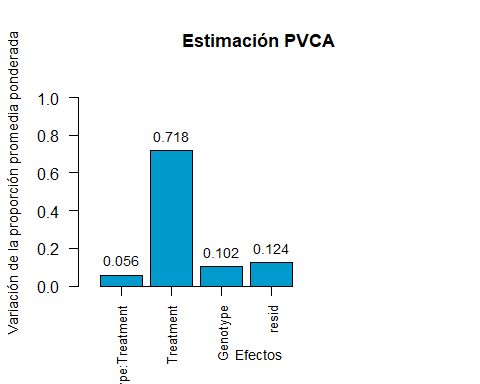


Figura A6. Importancia relativa de los distintos factores que influencian la expresión génica.

## 

## A5. Filtraje no específico

Distribución de la variabilidad de los genes de todas las muestras:

sds <- apply(exprs(eset\_rma.pec), 1, sd)  
sds0 <- sort(sds)  
plot(1:length(sds0), sds0, main="Distribución de la variabilidad de todos los genes", sub="Linias verticales representan los percentiles 90 y 95%", xlab= "Índice de genes (del menos al más variable)", ylab="Desviación estándar", col ="deepskyblue4", cex.main = 1, cex.lab = 0.9)  
abline(v=length(sds)\*c(0.9,0.95))

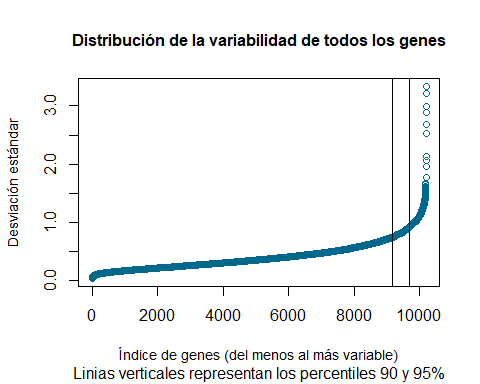


Figura A7. Valores de las desviaciones estándar de todos los genes ordenados de menor a mayor.

Posteriormente, se han eliminado los genes menos variables con la función **nsFilter** del paquete **genefitler** en el lindar 0.5:

library(genefilter)  
BiocManager::install("ecoli2.db")

## Bioconductor version 3.10 (BiocManager 1.30.10), R 3.6.1 (2019-07-05)

## Installing package(s) 'ecoli2.db'

## installing the source package 'ecoli2.db'

## Installation path not writeable, unable to update packages: class

## Old packages: 'backports', 'DelayedArray', 'dplyr', 'gdtools', 'glue',  
## 'lattice', 'lme4', 'nlme', 'Rcpp', 'reshape2', 'rgl', 'S4Vectors', 'sf',  
## 'survival', 'tibble'

BiocManager::install("AnnotationDbi")

## Bioconductor version 3.10 (BiocManager 1.30.10), R 3.6.1 (2019-07-05)

## Installing package(s) 'AnnotationDbi'

## package 'AnnotationDbi' successfully unpacked and MD5 sums checked  
##   
## The downloaded binary packages are in  
## C:\Users\Judith\AppData\Local\Temp\RtmpQlRONF\downloaded\_packages

## Installation path not writeable, unable to update packages: class

## Old packages: 'backports', 'DelayedArray', 'dplyr', 'gdtools', 'glue',  
## 'lattice', 'lme4', 'nlme', 'Rcpp', 'reshape2', 'rgl', 'S4Vectors', 'sf',  
## 'survival', 'tibble'

BiocManager::install("org.EcK12.eg.db")

## Bioconductor version 3.10 (BiocManager 1.30.10), R 3.6.1 (2019-07-05)

## Installing package(s) 'org.EcK12.eg.db'

## installing the source package 'org.EcK12.eg.db'

## Installation path not writeable, unable to update packages: class

## Old packages: 'backports', 'DelayedArray', 'dplyr', 'gdtools', 'glue',  
## 'lattice', 'lme4', 'nlme', 'Rcpp', 'reshape2', 'rgl', 'S4Vectors', 'sf',  
## 'survival', 'tibble'

library(ecoli2.db)

## Loading required package: AnnotationDbi

## Loading required package: org.EcK12.eg.db

##

##   
## ecoli2.db is providing annotations for only one of the species that are  
## supported by this platform. You may want to get other annotations  
## from other sources/packages in order to cover all the species that  
## are represented by probes on this platform.

annotation(eset\_rma.pec) <- "ecoli2.db"  
filtered.pec <- nsFilter(eset\_rma.pec, require.entrez = TRUE, remove.dupEntrez = TRUE, var.filter = TRUE, var.func = IQR, var.cutoff = 0.5, filterByQuantile = TRUE, feature.exclude = "^AFFX")  
print(filtered.pec$filter.log)

## $numDupsRemoved  
## [1] 179  
##   
## $numLowVar  
## [1] 2066  
##   
## $numRemoved.ENTREZID  
## [1] 5897

eset\_filtered.pec <- filtered.pec$eset  
eset\_filtered.pec

## ExpressionSet (storageMode: lockedEnvironment)  
## assayData: 2066 features, 8 samples   
## element names: exprs   
## protocolData  
## rowNames: wt\_nt.1 wt\_nt.2 ... fur\_cpr.2 (8 total)  
## varLabels: exprs dates  
## varMetadata: labelDescription channel  
## phenoData  
## rowNames: 1 2 ... 8 (8 total)  
## varLabels: FileName Group ... ShortName (5 total)  
## varMetadata: labelDescription channel  
## featureData: none  
## experimentData: use 'experimentData(object)'  
## Annotation: ecoli2.db

Se procede a guardar los datos normalizados y filtrados:

write.csv(exprs(eset\_rma.pec), file="./Resultados/Datos.normalizados.csv")  
write.csv(exprs(eset\_filtered.pec), file="./Resultados/Datos.normalizados.filtrados.csv")  
save(eset\_rma.pec, eset\_filtered.pec, file="./Resultados/Datos.normalizados.Rda")

## A6. Identificación de genes diferencialmente expresados

Creación de la **matriz de diseño:**

if(!exists("eset\_filtered.pec")) load(file="./Resultados/Datos.normalizados.Rda")

library(limma)

##   
## Attaching package: 'limma'

## The following object is masked from 'package:oligo':  
##   
## backgroundCorrect

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

Matdesign <- model.matrix(~0+Group, pData(eset\_filtered.pec))  
colnames(Matdesign) <- c("WT\_ntCPR", "WT\_CPR", "FUR\_ntCPR", "FUR\_CPR")  
print(Matdesign)

## WT\_ntCPR WT\_CPR FUR\_ntCPR FUR\_CPR  
## 1 0 0 0 1  
## 2 0 0 0 1  
## 3 0 0 1 0  
## 4 0 0 1 0  
## 5 0 1 0 0  
## 6 0 1 0 0  
## 7 1 0 0 0  
## 8 1 0 0 0  
## attr(,"assign")  
## [1] 1 1 1 1  
## attr(,"contrasts")  
## attr(,"contrasts")$Group  
## [1] "contr.treatment"

Creación de la **matriz de contrastes**:

Matcontrast <- makeContrasts(nt.WTvsFUR=FUR\_ntCPR-WT\_ntCPR, WT.ntvsCPR=WT\_CPR-WT\_ntCPR, FUR.ntvsCPR=FUR\_CPR-FUR\_ntCPR, levels = Matdesign)  
print(Matcontrast)

## Contrasts  
## Levels nt.WTvsFUR WT.ntvsCPR FUR.ntvsCPR  
## WT\_ntCPR -1 -1 0  
## WT\_CPR 0 1 0  
## FUR\_ntCPR 1 0 -1  
## FUR\_CPR 0 0 1

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

library(limma)  
fit.pec <- lmFit(eset\_filtered.pec, Matdesign)  
fit.main.pec <- contrasts.fit(fit.pec, Matcontrast)  
fit.main.pec <- eBayes(fit.main.pec)  
class(fit.main.pec)

## [1] "MArrayLM"  
## attr(,"package")  
## [1] "limma"

A partir de aquí, es posible obtener las listas de los genes diferencialmente expresados con la función **topTable** ordenados por el p-valor de menor a mayor para cada una de las comparaciones. Los seis primeros resultados obtenidos con la función **head**(tt\_X) se muestran en el apartado *4. Resultados* de este análisis.

tt\_nt.WTvsFUR <- topTable(fit.main.pec, number=nrow(fit.main.pec), coef = "nt.WTvsFUR", adjust = "fdr")  
library(xlsx)  
write.xlsx(tt\_nt.WTvsFUR, file = "tt\_nt.WTvsFUR.xlsx")

tt\_WT.ntvsCPR <- topTable(fit.main.pec, number=nrow(fit.main.pec), coef = "WT.ntvsCPR", adjust = "fdr")  
write.xlsx(tt\_WT.ntvsCPR, file = "tt\_WT.ntvsCPR.xlsx")

tt\_FUR.ntvsCPR <- topTable(fit.main.pec, number=nrow(fit.main.pec), coef="FUR.ntvsCPR", adjust = "fdr")  
write.xlsx(tt\_FUR.ntvsCPR, file = "tt\_FUR.ntvsCPR.xlsx")

## A7. Anotación de los resultados

Con el objetivo de asociar los identificadores de Affymetrix a nombres más familiares, se ha creado una función para facilitar la generación de estas tablas de anotaciones:

annotatedTopTable <- function(topTab, anotPackage)  
{  
 topTab <- cbind(PROBEID=rownames(topTab), topTab)  
 myProbes <- rownames(topTab)  
 thePackage <- eval(parse(text = anotPackage))  
 geneAnots <- select(thePackage, myProbes, c("SYMBOL", "ENTREZID", "GENENAME"))  
 annotatedTopTab<- merge(x=geneAnots, y=topTab, by.x="PROBEID", by.y="PROBEID")  
return(annotatedTopTab)  
}

topAnotados\_nt.WTvsFUR <- annotatedTopTable(tt\_nt.WTvsFUR, anotPackage = "ecoli2.db")

## 'select()' returned 1:1 mapping between keys and columns

topAnotados\_WT.ntvsCPR <- annotatedTopTable(tt\_WT.ntvsCPR, anotPackage = "ecoli2.db")

## 'select()' returned 1:1 mapping between keys and columns

topAnotados\_FUR.ntvsCPR <- annotatedTopTable(tt\_FUR.ntvsCPR, anotPackage = "ecoli2.db")

## 'select()' returned 1:1 mapping between keys and columns

write.csv(topAnotados\_nt.WTvsFUR, file="./Resultados/topAnotados\_nt.WTvsFUR.csv")  
write.csv(topAnotados\_WT.ntvsCPR, file="./Resultados/topAnotados\_WT.ntvsCPR.csv")  
write.csv(topAnotados\_FUR.ntvsCPR, file="./Resultados/topAnotados\_FUR.ntvsCPR.csv")

### A8. Visualización de los perfiles de expresión

Código para la representación de los genes diferencialmente expresados en cada una de las comparaciones mediante ***volcano plots***:

library(ecoli2.db)  
geneSymbols <- select(ecoli2.db, rownames(fit.main.pec), c("SYMBOL"))

## 'select()' returned 1:1 mapping between keys and columns

SYMBOLS <- geneSymbols$SYMBOL

volcanoplot(fit.main.pec, coef=1, highlight=4, names=SYMBOLS, main=paste("Genes diferencialmente expresados", colnames(Matcontrast)[1], sep="\n"), hl.col="deepskyblue4", col=c("darkolivegreen"), cex.main = 1, cex.lab = 0.9)  
abline(v=c(-1,1))

volcanoplot(fit.main.pec, coef=2, highlight=4, names=SYMBOLS, main=paste("Genes diferencialmente expresados", colnames(Matcontrast)[2], sep="\n"), hl.col="deepskyblue4", col=c("darkorange"), cex.main = 1, cex.lab = 0.9)  
abline(v=c(-1,1))

volcanoplot(fit.main.pec, coef=3, highlight=4, names=SYMBOLS, main=paste("Differentially expressed genes", colnames(Matcontrast)[3], sep="\n"), hl.col="deepskyblue4", col=c("darkorange4"), cex.main = 1, cex.lab = 0.9)  
abline(v=c(-1,1))

Los distintos plots se muestran en el apartado *4. Resultados* de este análisis.

Código para la generación de mapas de calor (***heatmaps***) de los distintos genes *up* o *down-regulated* agrupando los genes y muestras por similitud usando dendogramas:

probesInHeatmap.pec <- rownames(res.selected.pec)  
DatosHM <- exprs(eset\_filtered.pec)[rownames(exprs(eset\_filtered.pec)) %in% probesInHeatmap.pec,]  
  
geneSymbols <- select(ecoli2.db, rownames(DatosHM), c("SYMBOL"))

## 'select()' returned 1:1 mapping between keys and columns

SYMBOLS<- geneSymbols$SYMBOL  
rownames(DatosHM) <- SYMBOLS  
write.csv(DatosHM, file = file.path("./Resultados/DatosHM.csv"))

heatmap.2(DatosHM,  
 Rowv = TRUE,  
 Colv = TRUE,  
 dendrogram = "both",  
 main = "Differentially expressed genes \n FDR < 0,1, logFC >=1",  
 scale = "row",  
 col = my\_palette,  
 sepcolor = "white",  
 sepwidth = c(0.05,0.05),  
 cexRow = 0.5,  
 cexCol = 0.9,  
 key = TRUE,  
 keysize = 1.5,  
 density.info = "histogram",  
 ColSideColors = c(rep("springgreen",2),rep("darkgreen",2), rep("salmon",2), rep("darkred",2)),  
 tracecol = NULL,  
 srtCol = 30)

El mapa de calor se muestra en el apartado *4. Resultados* de este análisis.

## A9. Comparación entre distintas comparaciones

Con el objetivo de averiguar si existen genes diferencialmente expresados entre las distintas comparaciones se usa la función **decidetest** del paquete **limma** y posteriormente se realiza un diagrama de Venn:

library(limma)  
res.pec <- decideTests(fit.main.pec, method ="separate", adjust.method = "fdr", p.value = 0.1, lfc = 1)

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

## nt.WTvsFUR WT.ntvsCPR FUR.ntvsCPR  
## Down 42 70 141  
## NotSig 2012 1572 1778  
## Up 12 424 147

vennDiagram(res.selected.pec[,1:3], cex = 0.9, circle.col = "deepskyblue4")  
title("Genes in common between the three comparisons \n Genes selected with FDR < 0.1 and logFC > 1", cex.main = 1)

El diagrama de Venn se muestra en el apartado *4. Resultados* de este análisis.

## A10. Análisis de significación biológica: ‘Gene Enrichment Analysis’

Para realizar el análisis de significación, primero es necesario preparar la lista de genes a analizar:

ListaTablas.pec <- list(nt.WTvsFUR = tt\_nt.WTvsFUR, WT.ntvsCPR = tt\_WT.ntvsCPR, FUR.ntvsCPR = tt\_FUR.ntvsCPR)  
ListaSeleccionados.pec <- list()  
for (i in 1:length(ListaTablas.pec)){  
 topTab <- ListaTablas.pec[[i]]  
 whichGenes<-topTab["adj.P.Val"]<0.15  
 selectedIDs <- rownames(topTab)[whichGenes]  
 EntrezIDs<- select(ecoli2.db, selectedIDs, c("ENTREZID"))  
 EntrezIDs <- EntrezIDs$ENTREZID  
 ListaSeleccionados.pec[[i]] <- EntrezIDs  
 names(ListaSeleccionados.pec)[i] <- names(ListaTablas.pec)[i]  
}

## 'select()' returned 1:1 mapping between keys and columns  
## 'select()' returned 1:1 mapping between keys and columns  
## 'select()' returned 1:1 mapping between keys and columns

sapply(ListaSeleccionados.pec, length)

## nt.WTvsFUR WT.ntvsCPR FUR.ntvsCPR   
## 115 1893 1753

Se cargan los paquetes necesarios para este análisis y se procede a aplicar el análisis a las tres comparaciones realizadas en este estudio:

BiocManager::install("clusterProfiler")

## Bioconductor version 3.10 (BiocManager 1.30.10), R 3.6.1 (2019-07-05)

## Installing package(s) 'clusterProfiler'

## package 'clusterProfiler' successfully unpacked and MD5 sums checked  
##   
## The downloaded binary packages are in  
## C:\Users\Judith\AppData\Local\Temp\Rtmpuk3p4j\downloaded\_packages

## Installation path not writeable, unable to update packages: class

## Old packages: 'backports', 'beadarray', 'DelayedArray', 'dplyr', 'ff',  
## 'gdtools', 'glue', 'lattice', 'lme4', 'nlme', 'purrr', 'Rcpp', 'RCurl',  
## 'reshape2', 'rgl', 'S4Vectors', 'sf', 'survival', 'systemfonts', 'tibble',  
## 'tinytex'

library(AnnotationDbi)  
library(clusterProfiler)

##

## Registered S3 method overwritten by 'enrichplot':  
## method from  
## fortify.enrichResult DOSE

## clusterProfiler v3.14.3 For help: https://guangchuangyu.github.io/software/clusterProfiler  
##   
## If you use clusterProfiler in published research, please cite:  
## Guangchuang Yu, Li-Gen Wang, Yanyan Han, Qing-Yu He. clusterProfiler: an R package for comparing biological themes among gene clusters. OMICS: A Journal of Integrative Biology. 2012, 16(5):284-287.

##   
## Attaching package: 'clusterProfiler'

## The following object is masked from 'package:igraph':  
##   
## simplify

ListaDatos.pec <- ListaSeleccionados.pec[1:3]  
comparisonsNames <- names(ListaDatos.pec)  
universe <- mapped\_genes  
  
for (i in 1:length(ListaDatos.pec)){  
 genesIn <- ListaDatos.pec[[i]]  
 comparison <- comparisonsNames[i]  
 enrich.result <- enrichGO(gene = genesIn, OrgDb = "org.EcK12.eg.db", ont = "ALL", pAdjustMethod ="BH", pvalueCutoff = 0.05, readable = TRUE)  
   
 cat("##################################")  
 cat("\nComparison: ", comparison,"\n")  
 print(head(enrich.result))  
  
 if (length(rownames(enrich.result@result)) != 0) {  
 write.csv(as.data.frame(enrich.result),   
 file =paste0("./Resultados/","Enrich.Results.",comparison,".csv"),   
 row.names = FALSE)

write.xlsx(as.data.frame(enrich.result),   
 file =paste0("./Resultados/","Enrich.Results.",comparison,".xlsx"),   
 row.names = FALSE)

pdf(file=paste0("./Resultados/","Enrich.Barplot.",comparison,".pdf"))  
 print(barplot(enrich.result, showCategory = 15, font.size = 6,   
 title = paste0("EnrichGO Pathway Analysis for ", comparison,". Barplot")))  
 dev.off()  
  
 pdf(file = paste0("./Resultados/","EnrichGOcnetplot.",comparison,".pdf"))  
 print(cnetplot(enrich.result, categorySize = "geneNum", schowCategory = 15,   
 vertex.label.cex = 0.75))  
 dev.off()  
 }  
}

cnetplot(enrich.result, categorySize = "geneNum", schowCategory = 15, vertex.label.cex = 0.75)

La salida de este código se encuentra resumida en el apartado *4. Resultados* de este análisis y los archivos pueden encontrarse en el repositorio Github del mismo.