PEC2 - Análisis de Datos Ómicos

Juan Manuel Sancho Romero

2024-12-13

Contents

1. ABSTRACT

En esta segunda prueba de evaluación continua (PEC2) se desarrolla el **proceso de análisis de datos provenientes de *Microarrays*.** Para la realización de este análisis se siguieron las principales etapas que guían este tipo de análisis:

1. **Preparación de los datos**.
2. **Análisis exploratorio, control de calidad y normalización**.
3. **Filtrado de datos**.
4. **Contraste de las tres hipótesis a partir de la matriz de diseño y de contraste**.
5. **Anotación de las tres listas de sondas**.
6. **Análisis de significación biológica**.

El estudio, realizado por Sharma-Kuinkel et al., analiza los efectos diferenciales en la expresión génica del hospedador (*Mus musculus*) al ser tratado con los antibióticos linezolid y vancomicina antes y después de la infección por *Staphylococcus aureus* resistente a meticilina (MRSA), así como el propio efecto de la infección por MRSA en la expresión génica.

A partir de los datos crudos se realizó el análisis para contrastar tres hipótesis, con el objetivo de identificar genes cuya expresión difiera significativamente entre infectados y no infectados bajo las siguientes condiciones:

- Sin Tratamiento.
- Tratamiento con Linezolid.
- Tratamiento con Vancomicina.

Para los tres contrastes, se identificaron genes con expresión diferencial significativa antes y después de la infección. Sin embargo, los procesos metabólicos involucrados variaron dependiendo del tratamiento (Linezolid o Vancomicina) o su ausencia.

2. OBJETIVOS DEL ESTUDIO

En cuanto a los **objetivos de carácter didáctico**, relacionados con el aprendizaje práctico del flujo de trabajo con herramientas bioinformáticas en el ámbito del *Análisis de Datos de Microarrays*, se pueden establecer los siguientes:

1. **Aprender a manejar datos de microarrays**, al realizar tareas de obtención y procesado de datos, manejo de archivos `.CEL`, así como la creación de un `ExpressionSet`, junto a sus metadatos asociados.
2. **Aplicar técnicas de normalización y control de calidad** para asegurar la fiabilidad de los datos y evaluar su adecuación para el análisis estadístico.
3. **Implementar estrategias de filtrado para seleccionar sondas con mayor variabilidad y relevancia biológica**, utilizando medidas como el rango intercuartílico (IQR).
4. **Diseñar y realizar un análisis de contraste estadístico utilizando herramientas como el paquete `limma`**, para identificar genes diferencialmente expresados entre condiciones experimentales.

5. **Anotar las listas de genes seleccionados** con identificadores genómicos (como `GENENAME`, `SYMBOL`, `ENSEMBL` y `ENTREZID`) para facilitar su interpretación biológica.

6. **Realizar un análisis de significación biológica**, empleando bases de datos como `GO`, para asociar genes diferencialmente expresados con rutas metabólicas y funciones celulares.

7. **Desarrollo de la capacidad de interpretación y obtención de conclusiones relevantes a partir de datos transcriptómicos**.

Por otro lado, los **objetivos relacionados con los contrastes de hipótesis y la interpretación biológica del análisis** son los siguientes:

1. **Determinar los cambios en la expresión génica del hospedador inducidos por la infección con MRSA**, en ausencia de tratamiento, en tratamiento antibiótico con linezolid y en tratamiento antibiótico con Vancomicina.

2. **Comparar las distintas modulaciones de los procesos metabólicos y funciones celulares importantes del hospedador en respuesta a los tres tratamientos mediante el análisis de enriquecimiento biológico.**

3. MATERIALES Y MÉTODOS

3.1 ORIGEN Y NATURALEZA DE LOS DATOS

Los datos crudos utilizados provienen del repositorio público GEO (Gene Expression Omnibus), del experimento con código GSE38531 (https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE38531). Se obtuvo el archivo `GSE38531_RAW.tar` que incluye, tras su descompresión, la lista de archivos `.CEL` obtenidos en los microarrays de expresión génica de *Mus musculus* (Modelo murino).

Después de la obtención de los archivos `.CEL`, se realizó una selección representativa de los mismos, junto a la obtención de un dataframe `targets` de metadatos, utilizando la función `filter_microarray` facilitada en las instrucciones de este trabajo en el fichero `selectSamples.R`.

A partir de los datos crudos y de los metadatos mencionados se creó el primer `ExpressionSet` a partir del cuál se trabajó en el resto de etapas.

3.2 HERRAMIENTAS INFORMÁTICAS Y BIOINFORMÁTICAS

Para la realización de este trabajo se han utilizado las siguientes herramientas informáticas y bioinformáticas:

- **Lenguaje de programación R (versión 2024.09.1) y entorno Rstudio**.

Paquetes de R específicos para el manejo y análisis de datos ómicos:

- `geoquery`: Descarga y acceso a datos de la base de datos GEO (Gene Expression Omnibus).
- `oligo`: Lectura y procesamiento de archivos `.CEL`.
- `Biobase`: Creación y manejo de objetos `ExpressionSet`
- `limma`: Análisis estadístico de expresión diferencial.
- `clusterProfiler`: Análisis de significación biológica y anotación de rutas metabólicas.

- `genefilter`: Filtrado y análisis de variabilidad y significación de genes.
- `AnnotationDbi`: Anotación de genes desde paquetes de anotaciones como `org.Mm.eg.db` y `mouse4302.db`.

Paquetes de R de bases de datos de anotaciones de genes:

- `mouse4302.db`: Obtener anotaciones relacionadas con la plataforma de microarrays Mouse430 2.0.
- `org.Mm.eg.db`: Anotación de genes en el organismo modelo Mus musculus (ratón) con identificadores como `ENTREZID`, `SYMBOL`, etc.

Paquetes de R para el manejo, modificación y graficado de datos y conjuntos de datos:

- `r.utils`: Descompresión de archivos y manejo de archivos comprimidos.
- `fs`: Manipulación de rutas de archivos y directorios de manera eficiente.
- `dplyr`: Manipulación de datos, como filtrado, agrupamiento y creación de nuevas variables.
- `ggplot2`: Creación de gráficos y visualizaciones de datos.
- `kableExtra`: Manejo y edición de tablas de tipo `Kable`.

3.3 PROCEDIMIENTO GENERAL DEL ANÁLISIS

Las distintas fases del estudio han sido las siguientes:

- **1. Preparación de los datos**, con el fin de crear nuestro `expressionSet` a partir de los datos crudos del estudio y de sus correspondientes metadatos. Para ello, se descomprimieron los archivos `.CEL` y fueron seleccionados de manera representativa mediante la función `filter_microarray` que además genero un fichero `targets` con sus metadatos. A partir de estos archivos se creó el primero `ExpressionSet`.
- **2. Análisis exploratorio, control de calidad y normalización**, para inspeccionar si el objeto `expressionSet` ha sido creado correctamente, si necesita alguna modificación y la calidad de sus datos. Además, se comprobó si los datos reflejan o apoyan nuestras hipótesis preliminares. Por último, normalizamos los datos mediante el método `RMA` (Robust Multi-array Average).
- **3. Filtrado de datos**, donde se mantuvo, en un nuevo `expressionSet`, el 10% de las sondas con mayor variabilidad según el rango intercuartílico (IQR).
- **4. Contraste de las tres hipótesis a partir de la matriz de diseño y de contraste**, para ello se utilizó el paquete `limma` y se obtuvo una lista para cada contraste de las sondas seleccionadas (variabilidad con mayores significancias estadísticas).
- **5. Anotación de las tres listas de sondas** seleccionadas con sus correspondientes genes bajo los identificadores `GENENAME`, `SYMBOL`, `ENSEMBL` y `ENTREZID`.
- **6. Análisis de significación biológica**, donde se identificó con qué procesos metabólicos o funciones celulares están asociados los genes de cada uno de los tres contrastes.

3.4 MÉTODOS UTILIZADOS

En cuanto a los principales métodos utilizados en nuestro análisis tenemos:

- **Análisis Estadístico con `limma`**: este paquete permitió construir la matriz de diseño y posteriormente la matriz de contrastes con la que evaluó la expresión diferencial de genes para las tres hipótesis mencionadas. `limma` permite ajustar un modelo lineal, obteniendo distintos estadísticos (por ejemplo de análisis Bayesianos, p-valor ajustado, log-fold-change, etc) y así generar una lista de genes con significancia estadística (ordenados de mayor a menor) para cada comparación.
- **Anotación génica con `AnnotationDbi`**: Las sondas seleccionadas se anotaron con identificadores genómicos (`GENENAME`, `SYMBOL`, `ENSEMBL` y `ENTREZID`) utilizando la base de datos `mouse4302.db`, que contiene anotaciones para el microarray de tipo *Affymetrix Mouse Genome 430 2.0 Array*. Tras anotar el `ExpressionSet`, se anoto a partir de este los objetos `TopTable` con los listados de genes con significancia estadística.
- **Análisis de Significación Biológica con `clusterProfiler`**: Se realizó un análisis de sobrerepresentación para identificar las rutas metabólicas y procesos celulares relevantes relacionados con los genes con significancia estadística obtenidos en cada contraste. Para ello se utilizó la base de datos `org.Mm.eg.db` correspondiente al genoma completo de ratón (*Mus musculus*) y el análisis de GO (Gene Ontology) mediante la función `enrichGO`.

4. RESULTADOS

4.1 PPREPARACIÓN DE LOS DATOS

4.1.1 OBTENCIÓN DEL CONJUNTO DE DATOS ALEATORIO

Para ello se utilizó la función `filter_microarray` incluida en el fichero `selectSamples.R`, que se proporcionó junto a las instrucciones de esta PEC2.

```
> filter_microarray <- function(allTargets, seed = 123) {
+   # Configurar la semilla aleatoria
+   set.seed(seed)
+
+   # Filtrar las filas donde 'time' no sea 'hour 2'
+   filtered <- subset(allTargets, time != "hour 2")
+
+   # Dividir el dataset por grupos únicos de 'infection' + 'agent'
+   filtered$group <- interaction(filtered$infection, filtered$agent)
+
+   # Seleccionar 4 muestras al azar de cada grupo
+   selected <- do.call(rbind, lapply(split(filtered, filtered$group), function(group_data) {
+     if (nrow(group_data) > 4) {
+       group_data[sample(1:nrow(group_data), 4), ]
+     } else {
+       group_data
+     }
+   }))
+
+   # Obtener los índices originales como nombres de las filas seleccionadas
+   original_indices <- match(selected$sample, allTargets$sample)
```

```
+
+   # Modificar los rownames usando 'sample' y los índices originales
+   rownames(selected) <- paste0(selected$sample, ".", original_indices)
+
+   # Eliminar la columna 'group' y devolver el resultado
+   selected$group <- NULL
+   return(selected)
+ }
```

```
> # Simular el dataset basado en la descripción proporcionada
> allTargets <- data.frame(
+   sample = c("GSM944831", "GSM944838", "GSM944845", "GSM944852", "GSM944859",
+              "GSM944833", "GSM944840", "GSM944847", "GSM944854", "GSM944861",
+              "GSM944834", "GSM944841", "GSM944848", "GSM944855", "GSM944862",
+              "GSM944832", "GSM944839", "GSM944846", "GSM944853", "GSM944860",
+              "GSM944835", "GSM944842", "GSM944849", "GSM944856", "GSM944863",
+              "GSM944836", "GSM944843", "GSM944850", "GSM944857", "GSM944864",
+              "GSM944837", "GSM944844", "GSM944851", "GSM944858", "GSM944865"),
+   infection = c(rep("uninfected", 15), rep("S. aureus USA300", 20)),
+   time = c(rep("hour 0", 15), rep("hour 2", 5), rep("hour 24", 15)),
+   agent = c(rep("untreated", 5), rep("linezolid", 5), rep("vancomycin", 5),
+             rep("untreated", 5), rep("untreated", 5), rep("linezolid", 5), rep("vancomycin", 5))
+ )
>
> # Aplicar la función (cambiar 123 por vuestro ID de la UOC u otro número que podáis escribir en el do
> targets <- filter_microarray(allTargets, seed=78726399)
```

Tras aplicar esta función se obtuvo un dataframe `targets` con los códigos de las muestras seleccionadas y sus correspondientes metadatos acerca de las condiciones de esa muestra (presencia de infección, tiempo desde la infección, y tratamiento utilizado).

4.1.2 PROCESADO E IMPORTACIÓN DE LOS DATOS

Primero se crean los dos directorios de trabajo que contendrán los datos descargados (`data`) y algunos de los resultados (`results`) que se vayan obteniendo en el proceso de análisis.

```
> setwd("C:/Users/juanm/Desktop/MASTER_BIOINFORMATICA/Analisis_Datos_Omicos/ADO_PEC2")
> dir.create("data")            # Contendrá los datos que vamos a utilizar
> dir.create("results")         # Contendrá los resultados que vamos obteniendo
>
> dir_resultados <- "C:/Users/juanm/Desktop/MASTER_BIOINFORMATICA/Analisis_Datos_Omicos/ADO_PEC2/results
> dir_data <- "C:/Users/juanm/Desktop/MASTER_BIOINFORMATICA/Analisis_Datos_Omicos/ADO_PEC2/data"
```

De cara a realizar comparaciones entre tratamientos más adelante, se crea una columna que resuma la información contenida en las tres últimas columnas de `targets`. De este modo se guardarán las características resumidas de cada muestra. Para ello se utiliza el paquete `dplyr`.

```
> library(dplyr)
>
> # Creamos etiquetas resumidas para cada columna
> targets <- targets %>%
+   mutate(
```

```
+     # Información que contendrá cada nueva columna
+     infection_short = ifelse(infection == "S. aureus USA300", "Inf", "Uninf"),
+     # Si la columna contiene "x" llámalo "a" y si no, llámalo "b" y guárdalo en la nueva columna
+     time_short = ifelse(time == "hour 24", "h24", "h0"),
+     agent_short = case_when(
+       # Si la columna contiene la palabra "x" ahora llámala "y" y guárdalo en la nueva columna
+       agent == "linezolid" ~ "linez",
+       agent == "vancomycin" ~ "vanco",
+       agent == "untreated" ~ "untreat"
+     )
+   )
>
> # Creamos una columna combinada
> targets <- targets %>%
+   # Creamos una columna "Group" que combina las 3 nuevas columnas separadas por "."
+   mutate(Group = paste(infection_short, time_short, agent_short, sep = "."))
>
> # Añadimos índice basado en combinaciones únicas
> targets <- targets %>%
+   # Agrupamos según la columna "Group"
+   group_by(Group) %>%
+   # Creamos una nueva columna que incluya, el grupo al que pertenece esa fila + índice de su fila
+   mutate(ShortName = paste0(Group, ".", row_number())) %>%
+   # Desagrupamos para restablecer el orden de antes
+   ungroup()
>
> # Resultado
> head(targets)
```

```
# A tibble: 6 x 9
  sample    infection   time  agent infection_short time_short agent_short Group
  <chr>     <chr>       <chr> <chr> <chr>           <chr>      <chr>       <chr>
1 GSM944836 S. aureus ~ hour~ line~ Inf             h24        linez       Inf.~
2 GSM944850 S. aureus ~ hour~ line~ Inf             h24        linez       Inf.~
3 GSM944857 S. aureus ~ hour~ line~ Inf             h24        linez       Inf.~
4 GSM944864 S. aureus ~ hour~ line~ Inf             h24        linez       Inf.~
5 GSM944833 uninfected  hour~ line~ Uninf           h0         linez       Unin~
6 GSM944854 uninfected  hour~ line~ Uninf           h0         linez       Unin~
# i 1 more variable: ShortName <chr>
```

Para asegurar que el `ExpressionSet` se crea adecuadamente, se obtiene `targetsDf` que contendrá los códigos de cada muestra como nombres de sus filas.

Por otro lado, **es necesario y muy importante ordenar este dataframe de menor a mayor según la columna de códigos (`targetsDf$sample`)**. Esto permite que la función `read.celfiles()` establezca correctamente las correspondencias entre los archivos `.CEL` y las filas de `targetsDf`.

```
> # Convertir la primera columna de 'targets' en los rownames
> targetsDf <- as.data.frame(targets)
>
> # Extraer la primera columna como un vector de caracteres
> rownames(targetsDf) <- as.character(targetsDf[[1]])
>
```

```
> # IMPORTANTE: Reordenar targetsDf según la columna sample
> targetsDf <- targetsDf[order(targetsDf$sample), ]
>
> # Resultado
> head(targetsDf)
```

```
            sample        infection    time      agent infection_short
GSM944833 GSM944833       uninfected  hour 0  linezolid           Uninf
GSM944834 GSM944834       uninfected  hour 0 vancomycin           Uninf
GSM944835 GSM944835 S. aureus USA300 hour 24  untreated             Inf
GSM944836 GSM944836 S. aureus USA300 hour 24  linezolid             Inf
GSM944837 GSM944837 S. aureus USA300 hour 24 vancomycin             Inf
GSM944838 GSM944838       uninfected  hour 0  untreated           Uninf
          time_short agent_short            Group          ShortName
GSM944833         h0       linez   Uninf.h0.linez   Uninf.h0.linez.1
GSM944834         h0       vanco   Uninf.h0.vanco   Uninf.h0.vanco.2
GSM944835        h24     untreat  Inf.h24.untreat  Inf.h24.untreat.2
GSM944836        h24       linez    Inf.h24.linez    Inf.h24.linez.1
GSM944837        h24       vanco    Inf.h24.vanco    Inf.h24.vanco.2
GSM944838         h0     untreat Uninf.h0.untreat Uninf.h0.untreat.4
```

A continuación, la descarga y manejo de los archivos `.CEL` se realizará en la carpeta `data`, para ello se descarga, a partir de la base de datos GEO (Gene Expression Omnibus), el archivo comprimido `GSE38531_RAW.tar` que contiene los archivos `.CEL` asociados a este estudio.

```
> library(GEOquery)
> # Descargar los archivos .CEL asociados al estudio
> getGEOSuppFiles("GSE38531", baseDir = dir_data)
```

Tras esto, es necesario descomprimir el archivo `GSE38531_RAW.tar` y los archivos `gz` resultantes.

```
> library(R.utils)
>
> # Descompresión del archivo .tar
> untar("./data/GSE38531/GSE38531_RAW.tar", exdir = "./data/GSE38531")
>
> gz_files <- list.files("./data/GSE38531", pattern = "\\.gz$", full.names = TRUE)
>
> # Descompresión cada archivo gz
> lapply(gz_files, gunzip, overwrite = TRUE)
```

Después, se seleccionan los archivos `.CEL` cuyo nombre de muestra se coincida con los del dataframe `targetsDf` (columna `sample`), para ser copiado a un nuevo directorio llamado `Archivos_cel`.

```
> library(oligo)
> library(fs)
>
> # Definimos directorios
> source_dir <- "./data/GSE38531"
> cel_dir <- "./data/Archivos_cel"
>
> # Creamos un directorio de destino si no existe
```

```
> dir_create(cel_dir)
>
> # Listamos los archivos .CEL del directorio origen
> cel_files <- list.celfiles(source_dir, full.names = TRUE)
>
> # Filtramos los archivos cuyos códigos estén en targetsDF$sample
> selected_files <- cel_files[sub("_.*", # Extraemos el código GSM (contenido antes del primer "_")
+                                 "",    # Elimina el resto
+                                 # Nombres sin ruta completa que coinciden con la columna sample
+                                 basename(cel_files)) %in% targetsDf$sample]
>
> # Copiamos los archivos al directorio destino
> file_copy(selected_files, cel_dir, overwrite = TRUE)
>
> # Mensaje de confirmación
> cat(length(selected_files), "archivos copiados al directorio", cel_dir, "\n")
```

```
24 archivos copiados al directorio ./data/Archivos_cel
```

También será necesario modificar los nombres de los archivos `.CEL` seleccionados para que solo aparezca su código.

```
> library(oligo)
>
> # Listamos archivos actuales en el directorio destino
> copied_files <- list.celfiles(cel_dir,
+                               full.names = TRUE)
>
> # Creamos los nuevos nombres de archivos
> new_names <- paste0(sub("_.*", # Extraemos el código GSM (contenido antes del primer "_")
+                         "",    # Elimina el resto
+                         basename(copied_files)), # Nombre del archivo sin la ruta completa
+                     ".CEL")                      # Terminación
>
> # Renombramos los archivos
> file.rename(copied_files,         # Listados en copied_files
+             file.path(cel_dir,    # En el directorio cel_dir
+                       new_names)) # Con estos nombres
```

4.1.3 CREACIÓN DEL OBJETO `ExpressionSet`

Se realiza la creación del objeto `ExpressionSet` a partir del dataframe `targetsDf` (como phenoData o metadatos) y de los archivos `.CEL` seleccionados y guardados en la carpeta `Archivos_cel`. Se utiliza el paquete `Biobase`.

```
> library(oligo)
> celFiles <- list.celfiles("./data/Archivos_cel", full.names = TRUE)
>
> library(Biobase)
> targets.annotated <- AnnotatedDataFrame(targetsDf)
>
> rawData <- read.celfiles(celFiles, phenoData = targets.annotated)
```

4.2 ANÁLISIS EXPLORATORIO Y CONTROL DE CALIDAD

4.2.1 ANÁLISIS EXPLORATORIO

Este análisis exploratorio permite discernir si existe algún problema en nuestros datos o en el proceso de creación del `ExpressionSet`. Estas comprobaciones se incluyen en el Apéndice 1.

Una modificación necesaria es asignar como nombres la columna `ShortName` de `targets.annotated` para las filas en `pData(rawData)` y las columnas de `rawData`. Esto aportará más información en los análisis posteriores.

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

```
ExpressionFeatureSet (storageMode: lockedEnvironment)
assayData: 6 features, 24 samples
  element names: exprs
protocolData
  rowNames: Uninf.h0.linez.1 Uninf.h0.vanco.2 ... Inf.h24.vanco.1 (24
    total)
  varLabels: exprs dates
  varMetadata: labelDescription channel
phenoData
  rowNames: Uninf.h0.linez.1 Uninf.h0.vanco.2 ... Inf.h24.vanco.1 (24
    total)
  varLabels: sample infection ... ShortName (9 total)
  varMetadata: labelDescription channel
featureData: none
experimentData: use 'experimentData(object)'
Annotation: pd.mouse430.2
```

En cuanto a la estructura (Apéndice 1) del objeto `rawData` (`ExpressionSet`):

```
> rawData@annotation
```

```
[1] "pd.mouse430.2"
```

En el apartado `@annotation` se indica el paquete `pd.mouse430.2`; esto quiere decir que el objeto `ExpressionSet` está asociado al chip *Mouse430 2.0 de Affymetrix*, y es por tanto el que utilizaremos para realizar la anotación de cada gen más adelante.

Se ha comprobado que los datos fenotípicos, las dimensiones y los datos de expresión de rawData tienen el aspecto esperado. Estos resultados se encuentran en el Apéndice 1.

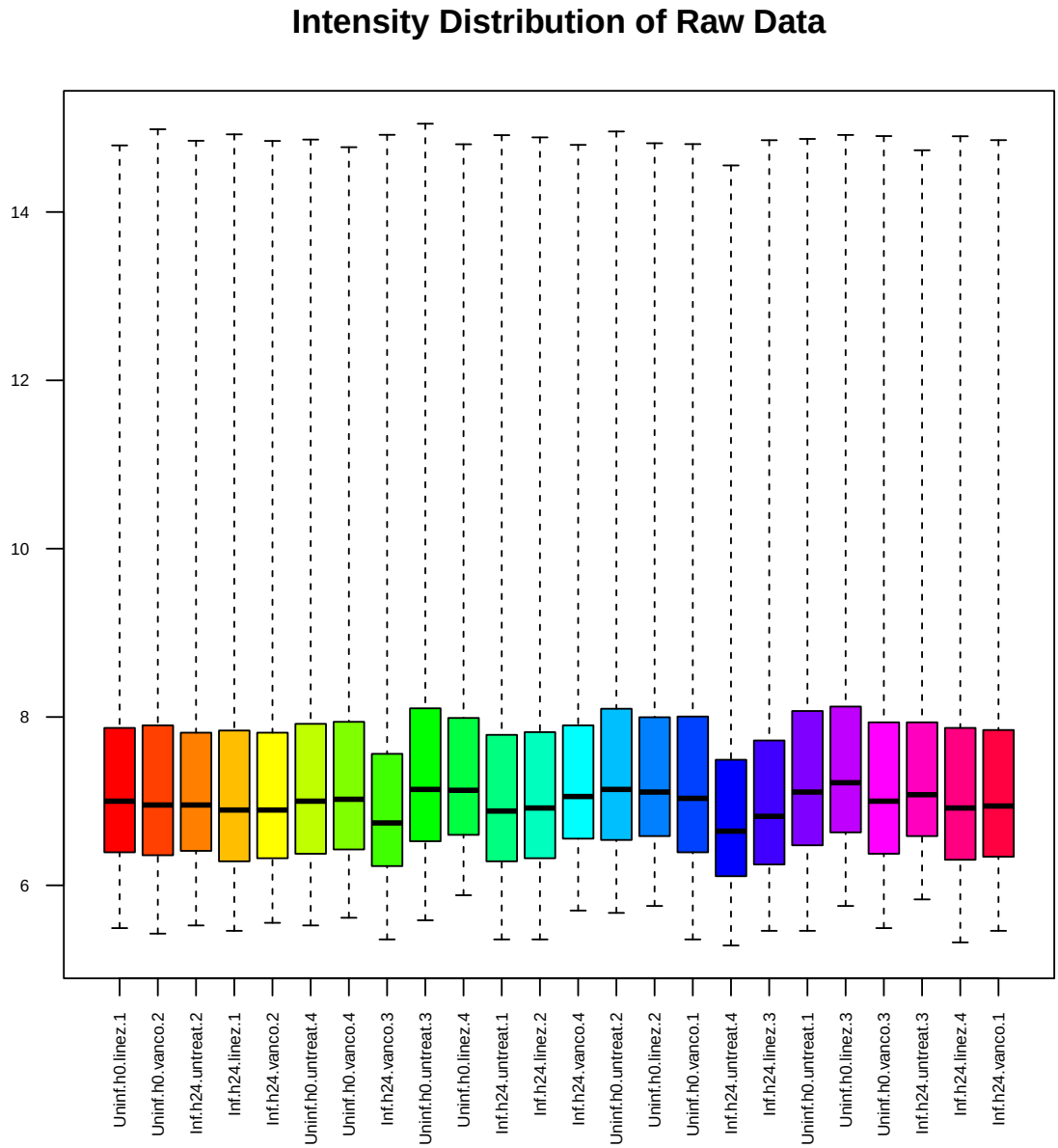
Por tanto, se puede concluir que el `ExpressionSet` ha sido creado con la estructura adecuada.

4.2.2 CONTROL DE CALIDAD

En este control de calidad analizaremos los datos de la matriz de expresión con el fin de comprobar la coherencia de los mismos o concordancia con nuestras hipótesis, su variabilidad, la presencia de outliers, etc.

```
> sampleColor <- rainbow(ncol(exprs(rawData)))  # Un color por muestra
>
> boxplot(rawData,       # Usamos los datos crudos
+         which = "all", # Graficamos todas las muestras
+         las = 2,       # Rotación de las etiquetas del eje Y (2 es vertical)
+         main = "Intensity Distribution of Raw Data", # Título del gráfico
+         cex.axis = 0.6,                              # Tamaño de las etiquetas de los ejes
+         col = sampleColor,                           # Colores de las muestras
+         names = sampleNames(rawData))                # Nombres de las muestras
```

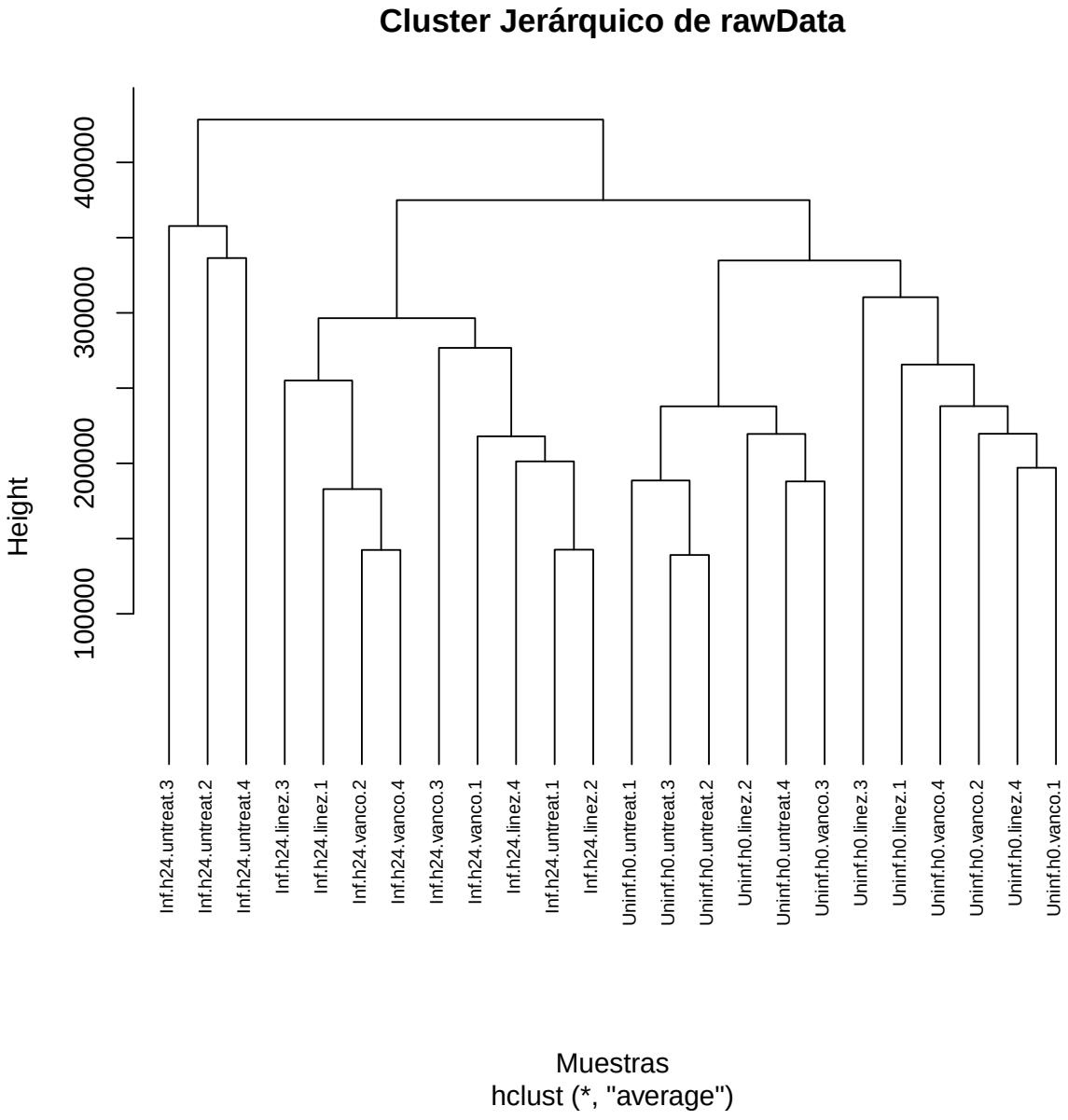
BOXPLOT DE LOS VALORES DE EXPRESIÓN DE CADA ARRAY



En este boxplot múltiple para todas las muestras se observa que la distribución de los valores de expresión de los `arrays` es muy parecida, lo que es una primera señal de que los datos son adecuados para ser analizados.

```
> clust.euclid.average <- hclust(dist(t(exprs(rawData))),
+                                method = "average") # calcula la distancia promedio en cada paso
>
> plot(clust.euclid.average,
+      labels = sampleNames(rawData),          # Etiquetas de las muestras
+      main = "Cluster Jerárquico de rawData", # Título principal
+      cex=0.7,                                # tamaño de las etiquetas
+      hang = -1,                              # Longitud de las ramas
+      xlab = "Muestras")                      # Nombre del eje X
```

CLÚSTER JERÁRQUICO

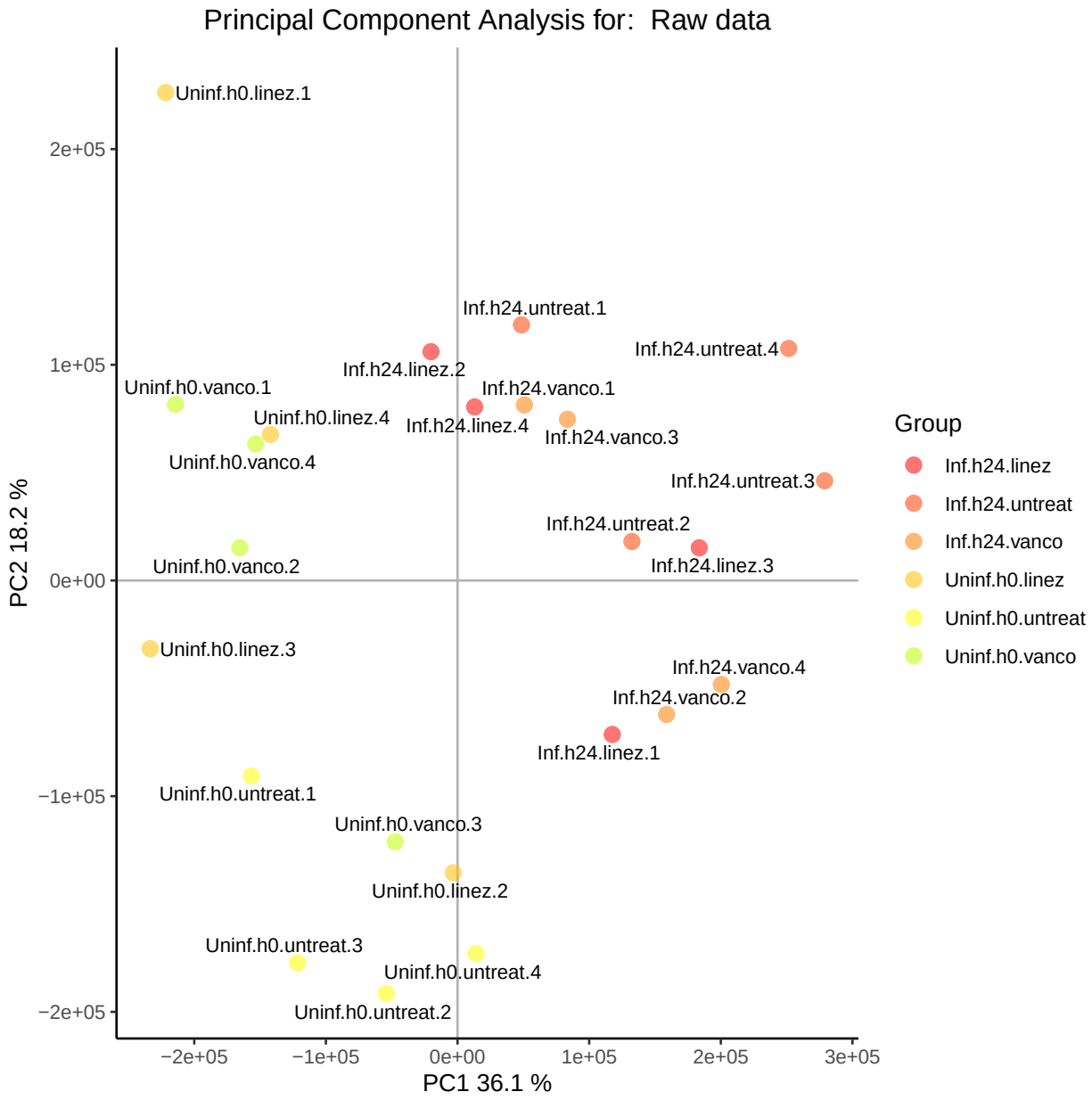


Se observan dos clústers principales que separan los ratones infectados (`Inf.24h`), frente a los no infectados (`Uninf.0h`), y a su vez, también se observan, dentro de estos, clústers definidos que separan los dos tipos de antibióticos usados (`linez` o `vanco`) y la ausencia de tratamiento (`untreat`). Estos agrupamientos son coherentes con nuestras hipótesis y apoyan el valor de un posterior análisis.

ANÁLISIS DE COMPONENTES PRINCIPALES (ACPs) Para realizar el ACPs, se ha adaptado la función utilizada en los apuntes de Gonzalo & Sanchez-Pla (2019).

```
> library(ggplot2)
> library(ggrepel)
>
> plotPCA3 <- function (datos, labels, factor, title, scale,colores, size = 1.5, glineas = 0.25) {
+     data <- prcomp(t(datos),scale=scale)
```

```
+    # plot adjustments
+    dataDf <- data.frame(data$x)
+    Group <- factor
+    loads <- round(data$sdev^2/sum(data$sdev^2)*100,1)
+    # main plot
+    p1 <- ggplot(dataDf,aes(x=PC1, y=PC2)) +
+      theme_classic() +
+      geom_hline(yintercept = 0, color = "gray70") +
+      geom_vline(xintercept = 0, color = "gray70") +
+      geom_point(aes(color = Group), alpha = 0.55, size = 3) +
+      coord_cartesian(xlim = c(min(data$x[,1])-5,max(data$x[,1])+5)) +
+      scale_fill_discrete(name = "Group")
+    # avoiding labels superposition
+    p1 + geom_text_repel(aes(y = PC2 + 0.25, label = labels),segment.size = 0.25, size = size) +
+      labs(x = c(paste("PC1",loads[1],"%")),y=c(paste("PC2",loads[2],"%"))) +
+      ggtitle(paste("Principal Component Analysis for: ",title,sep=" "))+
+      theme(plot.title = element_text(hjust = 0.5)) +
+      scale_color_manual(values=colores)
+    }
>
>
> plotPCA3(exprs(rawData), labels = targetsDf$ShortName, factor = targetsDf$Group,
+           title="Raw data", scale = FALSE, size = 3,
+           colores = sampleColor)
```



Se observa que la primera componente principal explica el 36.1% de la variabilidad de los datos, mientras que la segunda componente principal explica el 18.2%. En conjunto, ambas componentes principales aportan el 54.3% de la información total.

La diagonal que cruza el gráfico de izquierda a derecha, separando dos grupos principales (Infectados y No infectados), sugiere que las dos primeras componentes principales combinadas proporcionan información relevante para distinguir estos dos grupos en los datos.

```
> library(arrayQualityMetrics)
>
> dir_CC <- "C:/Users/juanm/Desktop/MASTER_BIOINFORMATICA/Analisis_Datos_Omicos/ADO_PEC2/results/CC_Raw
```

```
>
> arrayQualityMetrics(rawData,
+                     outdir = dir_CC, # Guardamos el resultado en su directorio
+                     force = TRUE)
```

CONTROL DE CALIDAD USANDO EL PAQUETE `arrayQualityMetrics` Esta función nos permite obtener una serie de gráficos y tablas que contienen información sobre la calidad de los datos. Tras observar los distintos gráficos, podemos concluir que la calidad de los datos es adecuada. En la siguiente imagen se muestra la tabla resumen a la que podemos acceder tras abrir el archivo `index.html` obtenido.

- Array metadata and outlier detection overview

	array	sampleNames	*1	*2	*3	sample	infection	time	agent	infection_short	time_short	agent_short	Group
☐	1	Uninf.h0.linez.1				GSM944833	uninfected	hour 0	linezolid	Uninf	h0	linez	Uninf.h0.linez
☐	2	Uninf.h0.vanco.2				GSM944834	uninfected	hour 0	vancomycin	Uninf	h0	vanco	Uninf.h0.vanco
☐	3	Inf.h24.untreat.2				GSM944835	S. aureus USA300	hour 24	untreated	Inf	h24	untreat	Inf.h24.untreat
☐	4	Inf.h24.linez.1				GSM944836	S. aureus USA300	hour 24	linezolid	Inf	h24	linez	Inf.h24.linez
☐	5	Inf.h24.vanco.2				GSM944837	S. aureus USA300	hour 24	vancomycin	Inf	h24	vanco	Inf.h24.vanco
☐	6	Uninf.h0.untreat.4				GSM944838	uninfected	hour 0	untreated	Uninf	h0	untreat	Uninf.h0.untreat
☐	7	Uninf.h0.vanco.4				GSM944841	uninfected	hour 0	vancomycin	Uninf	h0	vanco	Uninf.h0.vanco
☐	8	Inf.h24.vanco.3				GSM944844	S. aureus USA300	hour 24	vancomycin	Inf	h24	vanco	Inf.h24.vanco
☐	9	Uninf.h0.untreat.3			x	GSM944845	uninfected	hour 0	untreated	Uninf	h0	untreat	Uninf.h0.untreat
☐	10	Uninf.h0.linez.4				GSM944847	uninfected	hour 0	linezolid	Uninf	h0	linez	Uninf.h0.linez
☐	11	Inf.h24.untreat.1				GSM944849	S. aureus USA300	hour 24	untreated	Inf	h24	untreat	Inf.h24.untreat
☐	12	Inf.h24.linez.2				GSM944850	S. aureus USA300	hour 24	linezolid	Inf	h24	linez	Inf.h24.linez
☐	13	Inf.h24.vanco.4				GSM944851	S. aureus USA300	hour 24	vancomycin	Inf	h24	vanco	Inf.h24.vanco
☐	14	Uninf.h0.untreat.2			x	GSM944852	uninfected	hour 0	untreated	Uninf	h0	untreat	Uninf.h0.untreat
☐	15	Uninf.h0.linez.2				GSM944854	uninfected	hour 0	linezolid	Uninf	h0	linez	Uninf.h0.linez
☐	16	Uninf.h0.vanco.1				GSM944855	uninfected	hour 0	vancomycin	Uninf	h0	vanco	Uninf.h0.vanco
☐	17	Inf.h24.untreat.4	x		x	GSM944856	S. aureus USA300	hour 24	untreated	Inf	h24	untreat	Inf.h24.untreat
☐	18	Inf.h24.linez.3				GSM944857	S. aureus USA300	hour 24	linezolid	Inf	h24	linez	Inf.h24.linez
☐	19	Uninf.h0.untreat.1			x	GSM944859	uninfected	hour 0	untreated	Uninf	h0	untreat	Uninf.h0.untreat
☐	20	Uninf.h0.linez.3				GSM944861	uninfected	hour 0	linezolid	Uninf	h0	linez	Uninf.h0.linez
☐	21	Uninf.h0.vanco.3				GSM944862	uninfected	hour 0	vancomycin	Uninf	h0	vanco	Uninf.h0.vanco
☐	22	Inf.h24.untreat.3				GSM944863	S. aureus USA300	hour 24	untreated	Inf	h24	untreat	Inf.h24.untreat
☐	23	Inf.h24.linez.4				GSM944864	S. aureus USA300	hour 24	linezolid	Inf	h24	linez	Inf.h24.linez
☐	24	Inf.h24.vanco.1				GSM944865	S. aureus USA300	hour 24	vancomycin	Inf	h24	vanco	Inf.h24.vanco

The columns named *1, *2, ... indicate the calls from the different outlier detection methods:

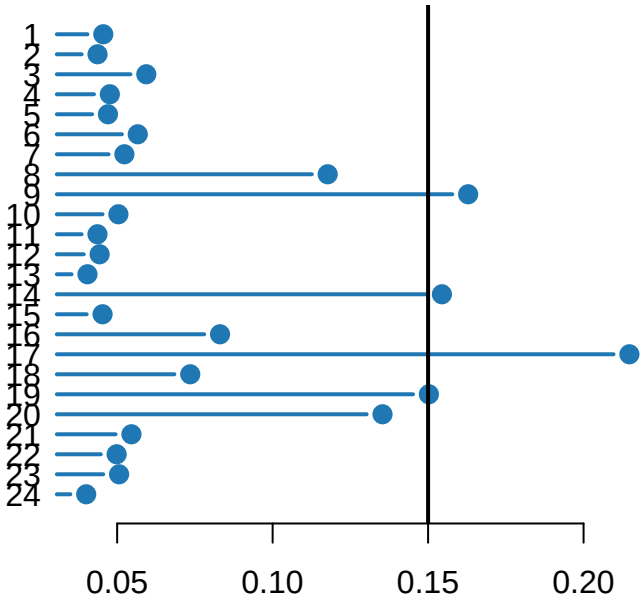
1. outlier detection by Distances between arrays
2. outlier detection by Boxplots
3. outlier detection by MA plots

En esta tabla tenemos información sobre los metadatos del array y los *outliers* detectados. En este caso, se han encontrado 4 tratamientos considerados como *outliers* detectados bajo el criterio 3 (criterio poco eficiente) y otro tratamiento detectados bajo el criterio 1 (más estricto) y 3.

En este caso la muestra que incluye el/los outliers detectados bajo el criterio 1 y 3, es la correspondiente a ratones infectados, tras 24 horas sin tratamiento.

Es llamativo que los otros *outliers* detectados bajo el criterio 3, se corresponden a los arrays de ratones no infectados y sin tratamiento.

Esto se puede corroborar en la **Figura 9: Outlier detection for MA plots.**



En conclusión, la calidad de los datos es adecuada y se puede proceder a la normalización de los mismos.

4.2.3 NORMALIZACIÓN DE LOS DATOS

El proceso de normalización de los datos de microarrays es crucial para asegurar que las muestras sean comparables entre sí. Gracias a la normalización es posible eliminar o suavizar sesgos en el análisis debido a causas técnicas o variabilidad no biológica.

Como consecuencia, tras la normalización es posible mejorar la sensibilidad y precisión del análisis estadístico.

Para la normalización de los datos utilizaremos el algoritmo RMA (Robust Multi-array Average).

```
> eset <- rma(rawData)
```

```
Background correcting
Normalizing
Calculating Expression
```

```
> # Guardamos el objeto expression set normalizado en la carpeta resultados
> write.exprs(eset, file.path(dir_resultados, "normData.txt"))
```

El algoritmo RMA normaliza los datos (Apéndice 2) de microarrays aplicando una serie de pasos que corrigen los efectos no biológicos (variabilidad entre arrays, el fondo de intensidad, y la variabilidad técnica entre sondas).

Primero, se realiza la corrección del fondo para reducir los efectos no deseados. Luego, se realiza la normalización entre arrays. Y finalmente, se realiza la sumarización o resumen de datos a nivel de características para estimar la señal de cada gen en lugar de tomar directamente la intensidad medida (media robusta).

Tras este proceso, los valores de expresión génica quedan en un nuevo objeto `ExpressionSet` más adecuado para el posterior análisis estadístico. Se puede observar que, al pasar de un `FeatureSet` como el anterior a un `ExpressionSet` en sentido estricto, las sondas son agrupadas, lo que reduce el número

de características, de 1,004,004 a 45,101. Esto es debido a que varias sondas pueden estar relacionadas con el mismo gen, y la normalización RMA permite combinar las señales correspondientes en una sola medida representativa para cada gen realizando su media robusta.

Fuente:

https://gtk-teaching.github.io/Microarrays-R/05-DataNormalisation/index.html

Al repetir el Boxplot múltiple para el `ExpressionSet` normalizado (Apéndice 2) denominado `eset`, se corrobora que la normalización RMA se ha realizado con éxito. Como se puede observar en el gráfico el aspecto de los distintos boxplot para cada muestra es similar.

También se ha llevado a cabo un nuevo control de calidad con `arrayQualityMetrics` para el `ExpressionSet` normalizado. En el apéndice (Apéndice 2) se incluye su tabla resumen. Se observa que tras la normalización, las muestras con outliers se han visto ligeramente reducidas en número (de 3 a 4) y que las muestras que presentan outliers ahora son otras. Esto guarda sentido con la explicación anterior del método RMA.

En conclusión, la calidad de los datos parece haber mejorado, en cuanto a que los gráficos de control de calidad corroboran la reducción de ruido y otras características no deseables que se han mencionado anteriormente (Apéndice 2).

4.3 FILTRADO DE LOS DATOS

El filtrado de los datos es importante a la hora de seguir eliminando ruido de las muestras, reducir el número de características no informativas (sondas no relevantes) y centrarse en la sondas más informativas (con mayor variabilidad). Esto, aunque también muy debatido, puede resultar un paso interesante para mejorar la calidad del análisis de datos de microarrays.

Tras instalar y cargar el paquete `mouse4302.db` para obtener los nombres de los genes y otras anotaciones asociadas a las sondas del chip, se filtra, es decir, se mantienen el 10% de los datos más variables del `ExpressionSet` normalizado. Se ha establecido como método, para medir y fijar un umbral de variabilidad, el rango intercuartílico (IQR).

Haciendo uso del paquete `genefilter`, se obtiene un nuevo `ExpressionSet` con el 10% de las sondas que presentaba el anterior.

```
> library(genefilter)
>
> # Vinculamos los identificadores de las sondas a genes con identificadores como entrez
> annotation(eset) <- "mouse4302.db"
>
> eset_fil <- nsFilter(eset, # Filtramos nuestro ExpressionSet normalizado
+                      # Rango intercuartílico (IQR) como medida de variabilidad
+                      var.func=IQR,
+                      # 10% de sondas con variabilidad (IQR) mayor al percentil 90
+                      var.cutoff=0.9,
+                      # Activa el filtrado basado en variabilidad
+                      var.filter = TRUE,
+                      # Quitamos sondas que sin identificador Entrez (apoyo en paquete mouse4302.db)
+                      require.entrez = TRUE,
+                      # Filtra sondas en un cuantil específico para reducir impacto de valores extremo
+                      filterByQuantile = TRUE)
>
> # Genes Eliminados
> cat("Nº de Genes Eliminados: \n")
```

```
Nº de Genes Eliminados:
```

```
> print(eset_fil$filter.log)
```

```
$numDupsRemoved
[1] 16958

$numLowVar
[1] 18432

$numRemoved.ENTREZID
[1] 7650

$feature.exclude
[1] 13
```

```
> # Genes que tenemos
> cat("\n Genes Incluidos: \n")
```

```
 Genes Incluidos:
```

```
> print(eset_fil$eset)
```

```
ExpressionSet (storageMode: lockedEnvironment)
assayData: 2048 features, 24 samples
  element names: exprs
protocolData
  rowNames: Uninf.h0.linez.1 Uninf.h0.vanco.2 ... Inf.h24.vanco.1 (24
    total)
  varLabels: exprs dates
  varMetadata: labelDescription channel
phenoData
  rowNames: Uninf.h0.linez.1 Uninf.h0.vanco.2 ... Inf.h24.vanco.1 (24
    total)
  varLabels: sample infection ... ShortName (9 total)
  varMetadata: labelDescription channel
featureData: none
experimentData: use 'experimentData(object)'
Annotation: mouse4302.db
```

A partir del objeto obtenido de tipo `EsetFiltered`, se procede a guardar en una nueva variable el `ExpressionSet` filtrado y su matriz de expresión.

```
> # Guardamos el eset filtrado
> eset_filtered <- eset_fil$eset
>
> # Matriz de Expresión del eset filtrado
> data_filtered <- exprs(eset_filtered)
>
> # Asignamos como nombres de las columnas los nombres cortos de cada tratamiento
> colnames(data_filtered) <- pData(eset_fil$eset)$ShortName
```

4.4 CONSTRUCCIÓN DE MATRIZ DE DISEÑO Y MATRICES DE CONTRASTES

4.4.1 CREACIÓN DE LA MATRIZ DE DISEÑO

La matriz de diseño refleja cómo se estructura el experimento y cómo se relacionan las muestras con las condiciones experimentales. Esta permitirá más adelante crear el modelo lineal para nuestros datos (Apéndice 3).

```
> library(limma)
>
> # Grupos de cada tratamiento
> new_group <- pData(eset_filtered)$Group
>
> # Lo transformamos a factor
> group_col <- factor(new_group, levels = unique(new_group))
>
> # Matriz de diseño
> design_mat <- model.matrix(~0+group_col)
>
> # Asignamos los grupos como niveles
> colnames(design_mat) <- levels(group_col)
>
> groupNames <- as.character(targetsDf$ShortName)
> rownames(design_mat) <- groupNames
```

4.4.2 CREACIÓN DE LA MATRIZ DE CONTRASTES CON LAS TRES COMPARACIONES

La matriz de contraste permite definir qué comparaciones entre tratamientos se van a realizar en el análisis de expresión diferencial (Apéndice 3).

```
> cont.matrix <- makeContrasts (Inf.h24.untreat.vs.Uninf.h0.untreat = Inf.h24.untreat-Uninf.h0.untreat,
+                               Inf.h24.linez.vs.Uninf.h0.linez = Inf.h24.linez-Uninf.h0.linez,
+                               Inf.h24.vanco.vs.Uninf.h0.vanco = Inf.h24.vanco-Uninf.h0.vanco,
+                               levels=design_mat)
```

4.4.3 ESTIMACIÓN DEL MODELO

A partir de la matriz de diseño y la matriz de contraste realizamos la estimación del modelo lineal utilizando las funciones del paquete `limma` (Apéndice 3).

```
> library(limma) # Funciones para el análisis de expresión diferencial
>
> # Modelo lineal (ml) para los datos de expresión 'eset_filtered' utilizando la matriz de diseño
> # 'lmFit' ajusta un ml para cada gen, según condiciones experimentales de 'design_mat'.
> fit <- lmFit(eset_filtered, design_mat)
>
> # Contrastes entre las condiciones utilizando la matriz de contrastes
> # 'contrasts.fit' ajusta los modelos lineales a los contrastes definidos
> fit.main <- contrasts.fit(fit, cont.matrix)
```

```
>
> # Aplicación del modelo de Bayes empírico a 'fit.main'
> # 'eBayes' permite estimaciones más robustas de los parámetros del ml mediante un enfoque bayesiano
> fit.main <- eBayes(fit.main)
```

Tras el análisis, se comprueban los resultados mediante la obtención de las tres listas de genes expresados de manera diferencial para cada contraste. Esto se realiza a partir de la creación de un objeto `topTab` para cada contraste.

A continuación, se obtienen los objetos `topTab`, donde se observa un listado de sondas para cada contraste y los parámetros de significación estadística asociados. Nótese que las sondas se encuentran ordenadas de menor a mayor significación estadística y que para cada listado, las dimensiones son distintas, es decir, han sido seleccionados en cada contraste distintos números de genes que cumplen la significación estadística exigida.

4.4.4 CONTRASTE 1. INFECTADOS VS NO INFECTADOS - SIN TRATAMIENTO

```
> topTab1 <- topTable(fit.main,
+                     coef = "Inf.h24.untreat.vs.Uninf.h0.untreat",
+                     number = nrow(fit.main),
+                     adjust = "fdr",  # Método de corrección FDR (False Discovery Rate)
+                     lfc = 3,         # Log-fold-change de 3
+                     p.value = 0.05)  # p-valor ajustado de 0.05
>
> # Dimensiones de la tabla ()
> cat("Nº de Genes Diferencialmente expresados:\n", dim(topTab1)[1])
```

```
Nº de Genes Diferencialmente expresados:
 62
```

```
> # Primeras 6 filas de la tabla
> head(topTab1)
```

```
               logFC   AveExpr        t      P.Value    adj.P.Val        B
1421262_at   7.310948  7.177899 19.44042 5.809286e-15 1.189742e-11 24.32253
1427747_a_at 6.255819 10.747379 17.88836 3.059588e-14 3.133018e-11 22.72585
1422953_at   3.801225 11.224319 16.98437 8.547669e-14 5.835209e-11 21.72933
1418722_at   6.099221 10.938369 15.90651 3.102012e-13 1.270584e-10 20.47028
1440865_at   4.294308 11.145293 15.18710 7.648889e-13 2.610821e-10 19.58346
1419532_at   4.811677  8.725073 15.01197 9.580322e-13 2.802929e-10 19.36160
```

4.4.5 CONTRASTE 2. INFECTADOS VS NO INFECTADOS - TRATAMIENTO CON LINEZOLID

```
> topTab2 <- topTable(fit.main,
+                     coef = "Inf.h24.linez.vs.Uninf.h0.linez",
+                     number = nrow(fit.main),
+                     adjust = "fdr",  # Método de corrección FDR (False Discovery Rate)
+                     lfc = 3,         # Log-fold-change de 3
```

```
+                       p.value = 0.05)  # p-valor ajustado de 0.05
>
> # Dimensiones de la tabla
> cat("Nº de Genes Diferencialmente expresados:\n", dim(topTab2)[1])
```

```
Nº de Genes Diferencialmente expresados:
 57
```

```
> # Primeras 6 filas de la tabla
> head(topTab2)
```

```
               logFC   AveExpr        t      P.Value    adj.P.Val        B
1421262_at   6.175381  7.177899 16.42085 1.661756e-13 3.403276e-10 20.97849
1427747_a_at 5.001954 10.747379 14.30296 2.439061e-12 2.432921e-09 18.40048
1448562_at   4.433876  7.664384 13.23492 1.073541e-11 5.496531e-09 16.95883
1419681_a_at 4.766888  7.217956 12.48172 3.236372e-11 1.104682e-08 15.87780
1440865_at   3.462548 11.145293 12.24553 4.623494e-11 1.352702e-08 15.52711
1418722_at   4.500394 10.938369 11.73684 1.015068e-10 2.097830e-08 14.75202
```

4.4.6 CONTRASTE 3. INFECTADOS VS NO INFECTADOS - TRATAMIENTO CON VANCOMICINA

```
> topTab3 <- topTable(fit.main,
+                       coef = "Inf.h24.vanco.vs.Uninf.h0.vanco",
+                       number = nrow(fit.main),
+                       adjust = "fdr",  # Método de corrección FDR (False Discovery Rate)
+                       lfc = 3,         # Log-fold-change de 3
+                       p.value = 0.05)  # p-valor ajustado de 0.05
>
> # Dimensiones de la tabla (Nº de Genes Diferencialmente expresados)
> cat("Nº de Genes Diferencialmente expresados:\n", dim(topTab3)[1])
```

```
Nº de Genes Diferencialmente expresados:
 30
```

```
> # Primeras 6 filas de la tabla
> head(topTab3)
```

```
               logFC   AveExpr        t      P.Value    adj.P.Val        B
1421262_at   6.680581  7.177899 17.76422 3.513535e-14 7.195719e-11 22.51139
1427747_a_at 5.141018 10.747379 14.70061 1.437484e-12 9.813227e-10 18.93899
1419681_a_at 5.334596  7.217956 13.96822 3.842571e-12 1.967396e-09 17.97837
1418722_at   5.159674 10.938369 13.45621 7.836424e-12 3.209799e-09 17.27902
1440865_at   3.600893 11.145293 12.73479 2.221100e-11 5.426192e-09 16.25236
1419709_at   5.207558  6.974553 12.41518 3.576585e-11 5.426192e-09 15.78133
```

4.5 ANOTACIÓN DE GENES DIFERENCIALMENTE EXPRESADOS PARA CADA COMPARACIÓN

El siguiente paso es anotar los listados de sondas obtenidos en el paso anterior (genes anotados según el criterio de la compañía que realizó los microarrays), es decir, vamos a asociar los identificadores `GENENAME`, `SYMBOL`, `ENSEMBL` y `ENTREZID` a cada gen.

En el paquete `mouse4302.db` se encuentran las anotaciones disponibles para nuestro conjunto de datos (Apéndice 4).

```
> library(mouse4302.db) # Cargamos el paquete `mouse4302.db`
```

Para realizar la anotación es necesario obtener los nombres de las sondas de nuestro `ExpressionSet` filtrado y así, después poder anotarlas, en nuestro caso, con los identificadores `GENENAME`, `SYMBOL`, `ENSEMBL` y `ENTREZID` (Apéndice 4).

```
> library(AnnotationDbi)
>
> # Obtenemos las sondas del objeto ExpressionSet
> probes <- rownames(eset_filtered)
>
> # Extraemos nombres de genes y descripciones asociadas a las sondas
> annotated_data <- AnnotationDbi::select(mouse4302.db, # Paquete de anotaciones del microarray
+                           # Nombres de las sondas del eset
+                           keys = probes,
+                           # Identificadores a añadir
+                           columns = c("GENENAME", "SYMBOL", "ENSEMBL", "ENTREZID"))
```

Una vez realizado esto, es posible asociar las anotaciones obtenidas para el conjunto de sondas de `eset_filtered` con los genes guardados en las `topTables` (genes que se han expresado significativamente más que el resto para cada contraste).

4.5.1 ANOTACIÓN PARA EL CONTRASTE 1. INFECTADOS VS NO INFECTADOS - SIN TRATAMIENTO

```
> library(dplyr)
>
> topTab1_anot <- topTab1 %>%
+   mutate(PROBEID=rownames(topTab1)) %>% # Nueva columna llamada PROBEID como nombre de filas de topTa
+   left_join(annotated_data) %>% # Df con las columnas de topTab más sus anotaciones
+   arrange(P.Value) %>%    # Ordena de menor a mayor p-valor el df
+   select(7,8,9,10, 1:6, 11)   # Columnas 7, 8, 9 y 10 primero (Identificadores) y luego las restantes
>
> # Primeros 6 genes con p-valores más significativos
> head(topTab1_anot)
```

```
      PROBEID                                GENENAME SYMBOL
1   1421262_at                 lipase, endothelial   Lipg
2 1427747_a_at                          lipocalin 2   Lcn2
3   1422953_at            formyl peptide receptor 2   Fpr2
4   1418722_at          neutrophilic granule protein    Ngp
```

```
5   1440865_at interferon induced transmembrane protein 6 Ifitm6
6   1419532_at            interleukin 1 receptor, type II  Il1r2
             ENSEMBL    logFC   AveExpr        t      P.Value    adj.P.Val
1 ENSMUSG00000053846 7.310948  7.177899 19.44042 5.809286e-15 1.189742e-11
2 ENSMUSG00000026822 6.255819 10.747379 17.88836 3.059588e-14 3.133018e-11
3 ENSMUSG00000052270 3.801225 11.224319 16.98437 8.547669e-14 5.835209e-11
4 ENSMUSG00000032484 6.099221 10.938369 15.90651 3.102012e-13 1.270584e-10
5 ENSMUSG00000059108 4.294308 11.145293 15.18710 7.648889e-13 2.610821e-10
6 ENSMUSG00000026073 4.811677  8.725073 15.01197 9.580322e-13 2.802929e-10
         B ENTREZID
1 24.32253    16891
2 22.72585    16819
3 21.72933    14289
4 20.47028    18054
5 19.58346   213002
6 19.36160    16178
```

4.5.2 ANOTACIÓN PARA EL CONTRASTE 2. INFECTADOS VS NO INFECTADOS - TRATAMIENTO CON LINEZOLID

```
> library(dplyr)
>
> topTab2_anot <- topTab2 %>%
+   mutate(PROBEID=rownames(topTab2)) %>% # Nueva columna llamada PROBEID como nombre de filas de topTa
+   left_join(annotated_data) %>% # Df con las columnas de topTab más sus anotaciones
+   arrange(P.Value) %>%    # Ordena de menor a mayor p-valor el df
+   select(7,8,9,10, 1:6, 11)   # Columnas 7, 8, 9 y 10 primero (Identificadores) y luego las restantes
>
> # Primeros 6 genes con p-valores más significativos
> head(topTab2_anot)
```

```
       PROBEID                                  GENENAME SYMBOL
1   1421262_at                      lipase, endothelial   Lipg
2 1427747_a_at                             lipocalin 2   Lcn2
3   1448562_at                  uridine phosphorylase 1   Upp1
4 1419681_a_at                        prokineticin 2  Prok2
5   1440865_at interferon induced transmembrane protein 6 Ifitm6
6   1418722_at             neutrophilic granule protein    Ngp
             ENSEMBL    logFC   AveExpr        t      P.Value    adj.P.Val
1 ENSMUSG00000053846 6.175381  7.177899 16.42085 1.661756e-13 3.403276e-10
2 ENSMUSG00000026822 5.001954 10.747379 14.30296 2.439061e-12 2.432921e-09
3 ENSMUSG00000020407 4.433876  7.664384 13.23492 1.073541e-11 5.496531e-09
4 ENSMUSG00000030069 4.766888  7.217956 12.48172 3.236372e-11 1.104682e-08
5 ENSMUSG00000059108 3.462548 11.145293 12.24553 4.623494e-11 1.352702e-08
6 ENSMUSG00000032484 4.500394 10.938369 11.73684 1.015068e-10 2.097830e-08
         B ENTREZID
1 20.97849    16891
2 18.40048    16819
3 16.95883    22271
4 15.87780    50501
5 15.52711   213002
6 14.75202    18054
```

4.5.3 ANOTACIÓN PARA EL CONTRASTE 3. INFECTADOS VS NO INFECTADOS - TRATAMIENTO CON VANCOMICINA

```
> library(dplyr)
>
> topTab3_anot <- topTab3 %>%
+   mutate(PROBEID=rownames(topTab3)) %>% # Nueva columna llamada PROBEID como nombre de filas de topTa
+   left_join(annotated_data) %>% # Df con las columnas de topTab más sus anotaciones
+   arrange(P.Value) %>%    # Ordena de menor a mayor p-valor el df
+   select(7,8,9,10, 1:6, 11)   # Columnas 7, 8, 9 y 10 primero (Identificadores) y luego las restantes
>
> # Primeros 6 genes con p-valores más significativos
> head(topTab3_anot)
```

```
      PROBEID                                  GENENAME SYMBOL
1   1421262_at                     lipase, endothelial   Lipg
2 1427747_a_at                             lipocalin 2   Lcn2
3 1419681_a_at                          prokineticin 2  Prok2
4   1418722_at            neutrophilic granule protein    Ngp
5   1440865_at interferon induced transmembrane protein 6 Ifitm6
6   1419709_at                                stefin A3  Stfa3
            ENSEMBL    logFC   AveExpr        t      P.Value    adj.P.Val
1 ENSMUSG00000053846 6.680581  7.177899 17.76422 3.513535e-14 7.195719e-11
2 ENSMUSG00000026822 5.141018 10.747379 14.70061 1.437484e-12 9.813227e-10
3 ENSMUSG00000030069 5.334596  7.217956 13.96822 3.842571e-12 1.967396e-09
4 ENSMUSG00000032484 5.159674 10.938369 13.45621 7.836424e-12 3.209799e-09
5 ENSMUSG00000059108 3.600893 11.145293 12.73479 2.221100e-11 5.426192e-09
6 ENSMUSG00000054905 5.207558  6.974553 12.41518 3.576585e-11 5.426192e-09
         B ENTREZID
1 22.51139    16891
2 18.93899    16819
3 17.97837    50501
4 17.27902    18054
5 16.25236   213002
6 15.78133    20863
```

Fuente:

Gonzalo, R., & Sanchez-Pla, A. (2019). Statistical analysis of microarray data. Universidad Oberta de Catalunya. Asignatura: Análisis de datos ómicos. https://aspteaching.github.io/Omics_Data_Analysis-Case_Study_1-Microarrays/Case_Study_1-Microarrays_Analysis.html#read-the-cel-files

4.6 ANÁLISIS DE SIGNIFICACIÓN BIOLÓGICA

En este paso se realiza el análisis de sobre-representación (*Over-Representation Analysis*) y los gráficos para cada contraste utilizando el paquete `clusterProfiler`.

Existe la posibilidad de realizar el análisis con *Gene Ontology (GO)* o con *Genes and Genomes (KEGG)*.

En este caso, he decidido utilizar *Gene Ontology (GO)*, que ofrece más información sobre el funcionamiento de los genes en cuanto a los procesos biológicos y componentes celulares a los que se encuentran asociados.

4.6.1 ANÁLISIS DE SOBRE-REPRESENTACIÓN. CONTRASTE 1. INFECTADOS VS NO INFECTADOS - SIN TRATAMIENTO

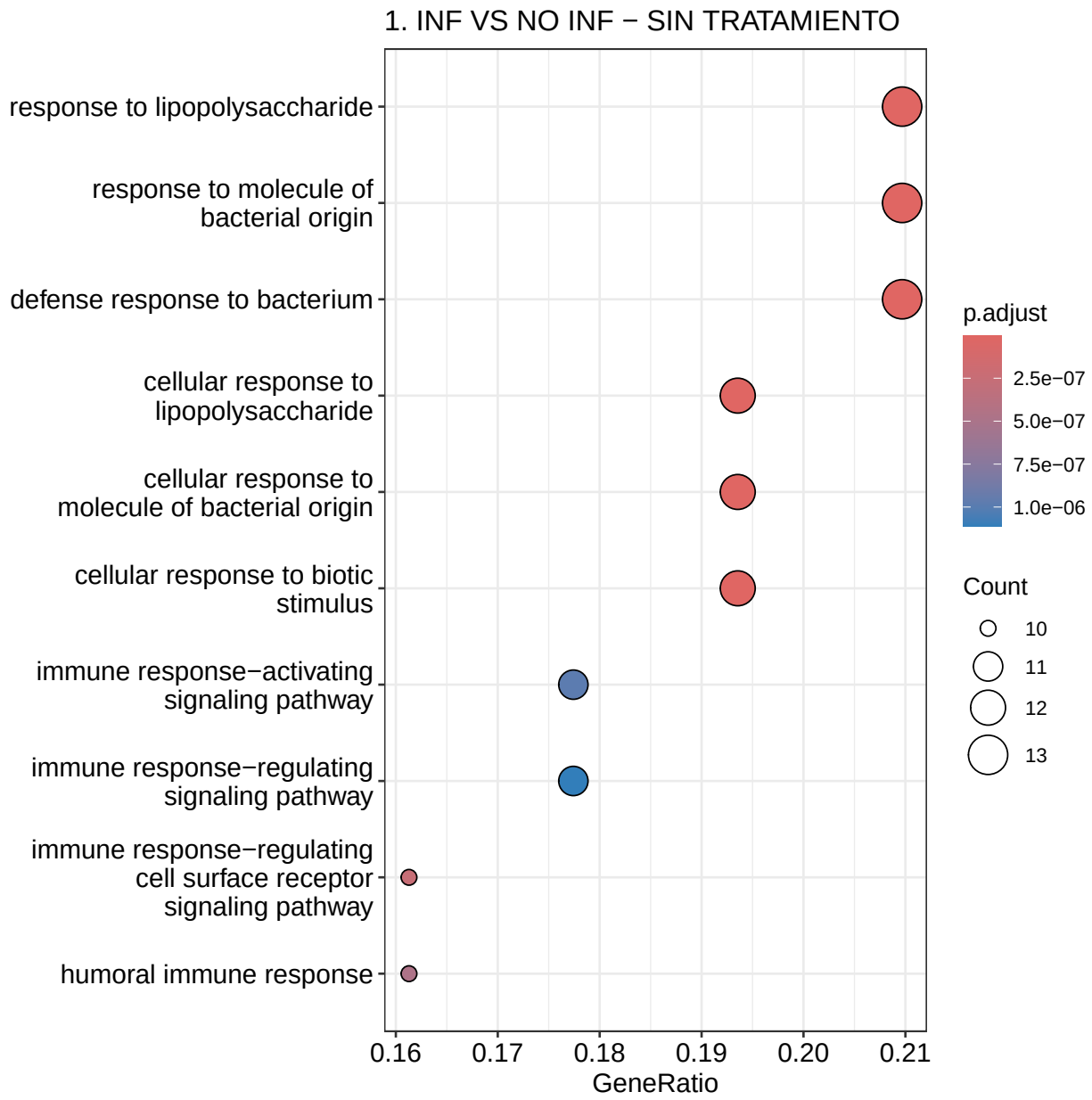
```
> library(clusterProfiler)
> library(org.Mm.eg.db)  # Base de datos de genes de ratón
>
> # Análisis de sobre-representación usando Gene Ontology (GO)
> go_enrichment.1 <- enrichGO(
+     gene = topTab1_anot$ENTREZID,    # IDs de los genes de interés
+     OrgDb = org.Mm.eg.db,            #  Base de datos utilizada
+     keyType = "ENTREZID",              # Tipo de ID utilizado
+     ont = "BP",                        # Ontología: BP (Biological Process), MF (Molecular Function),
+     pAdjustMethod = "BH",              # Método de corrección para múltiples comparaciones (por ejempl
+     qvalueCutoff = 0.05,               # Valor p corregido (FDR) umbral
+     readable = TRUE                    # Convertimos IDs a nombres legibles
+ )
>
> # Resultados
> head(go_enrichment.1)
```

```
                   ID                                     Description
GO:0032496 GO:0032496                   response to lipopolysaccharide
GO:0071222 GO:0071222           cellular response to lipopolysaccharide
GO:0071219 GO:0071219 cellular response to molecule of bacterial origin
GO:0002237 GO:0002237          response to molecule of bacterial origin
GO:0042742 GO:0042742                    defense response to bacterium
GO:0071216 GO:0071216              cellular response to biotic stimulus
           GeneRatio   BgRatio RichFactor FoldEnrichment   zScore       pvalue
GO:0032496     13/62 402/28905 0.03233831       15.07643 13.17695 2.693470e-12
GO:0071222     12/62 312/28905 0.03846154       17.93114 13.94082 2.701757e-12
GO:0071219     12/62 322/28905 0.03726708       17.37427 13.69906 3.908367e-12
GO:0002237     13/62 422/28905 0.03080569       14.36191 12.81995 4.950978e-12
GO:0042742     13/62 430/28905 0.03023256       14.09471 12.68390 6.261988e-12
GO:0071216     12/62 350/28905 0.03428571       15.98433 13.07630 1.033353e-11
               p.adjust       qvalue
GO:0032496 1.710563e-09 1.158268e-09
GO:0071222 1.710563e-09 1.158268e-09
GO:0071219 1.710563e-09 1.158268e-09
GO:0002237 1.710563e-09 1.158268e-09
GO:0042742 1.730813e-09 1.171980e-09
GO:0071216 2.380157e-09 1.611669e-09
                                                                                geneID
GO:0032496 Mmp8/Plscr1/Wfdc21/Tlr4/Cd14/Acod1/Fcgr4/Ltf/Cxcl10/Gbp2b/Gbp2/Cxcl2/Cxcl3
GO:0071222        Mmp8/Plscr1/Tlr4/Cd14/Acod1/Fcgr4/Ltf/Cxcl10/Gbp2b/Gbp2/Cxcl2/Cxcl3
GO:0071219        Mmp8/Plscr1/Tlr4/Cd14/Acod1/Fcgr4/Ltf/Cxcl10/Gbp2b/Gbp2/Cxcl2/Cxcl3
GO:0002237 Mmp8/Plscr1/Wfdc21/Tlr4/Cd14/Acod1/Fcgr4/Ltf/Cxcl10/Gbp2b/Gbp2/Cxcl2/Cxcl3
GO:0042742   Lcn2/Fpr2/Wfdc21/Tlr4/Wfdc17/Anxa3/Clec4e/Ltf/Hp/Camp/Gbp2b/Gbp2/Rnase2a
GO:0071216        Mmp8/Plscr1/Tlr4/Cd14/Acod1/Fcgr4/Ltf/Cxcl10/Gbp2b/Gbp2/Cxcl2/Cxcl3
           Count
GO:0032496    13
GO:0071222    12
GO:0071219    12
GO:0002237    13
```

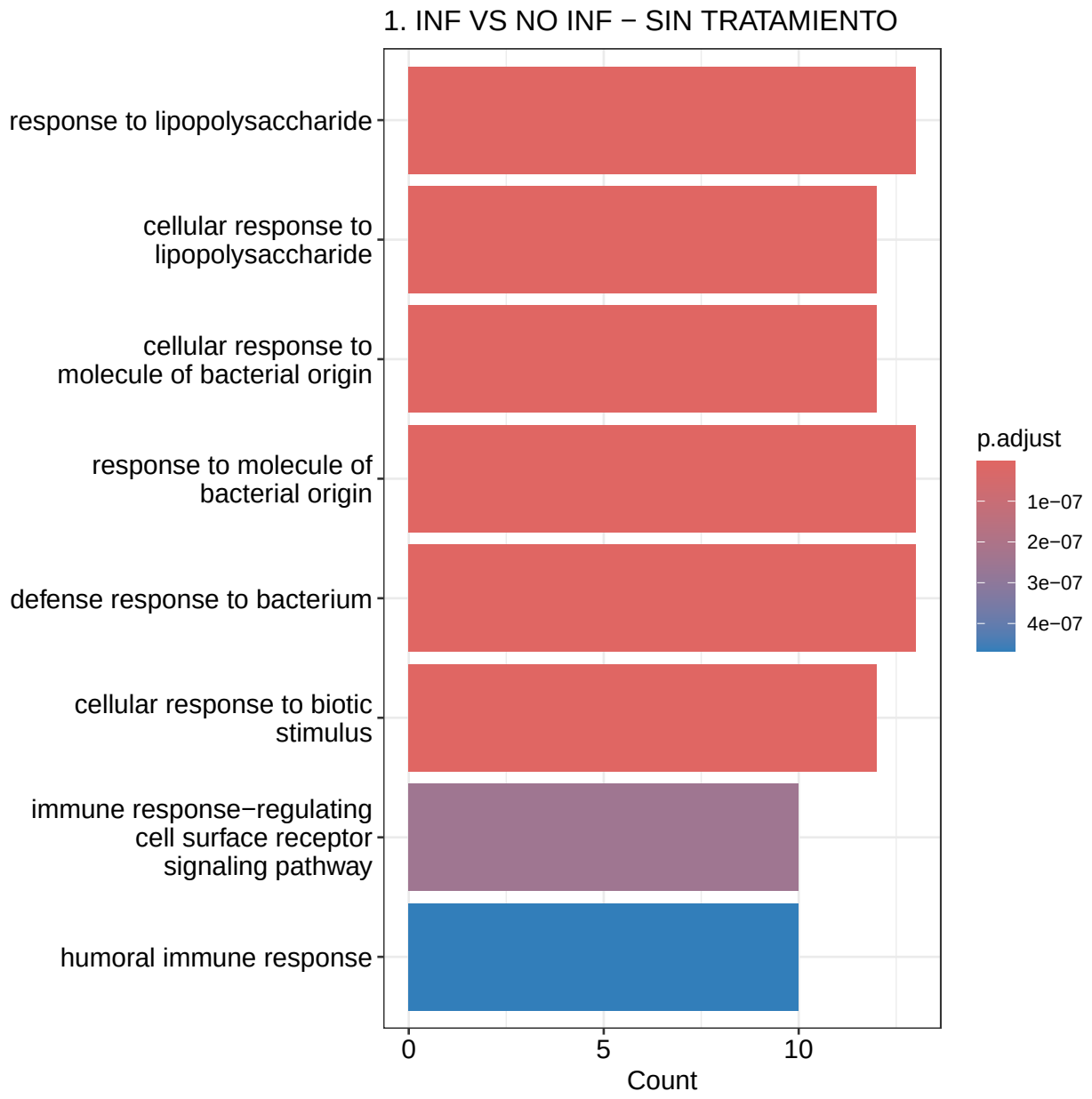
```
GO:0042742    13
GO:0071216    12
```

```
> # Visualizar los resultados como un gráfico de puntos
> dotplot(go_enrichment.1, title = "1. INF VS NO INF - SIN TRATAMIENTO")
```

GRÁFICOS



```
> # Visualizar como un gráfico de barras
> barplot(go_enrichment.1, title = "1. INF VS NO INF - SIN TRATAMIENTO")
```



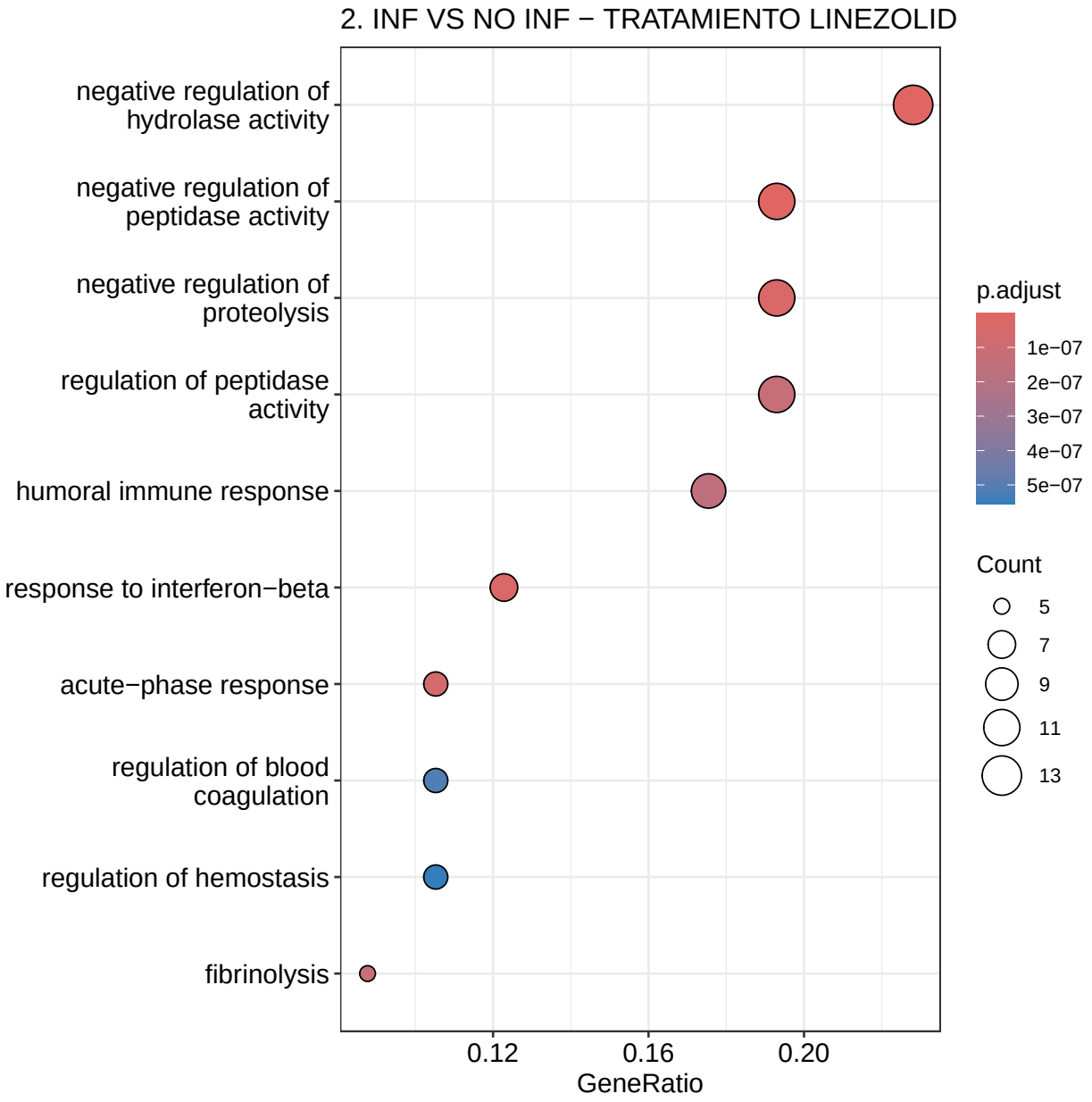
4.6.2 ANÁLISIS DE SOBRE-REPRESENTACIÓN. CONTRASTE 2. INFECTADOS VS NO INFECTADOS - TRATAMIENTO CON LINEZOLID

```
> library(clusterProfiler)
> library(org.Mm.eg.db)  # Base de datos de genes de ratón
>
> # Análisis de sobre-representación usando Gene Ontology (GO)
```

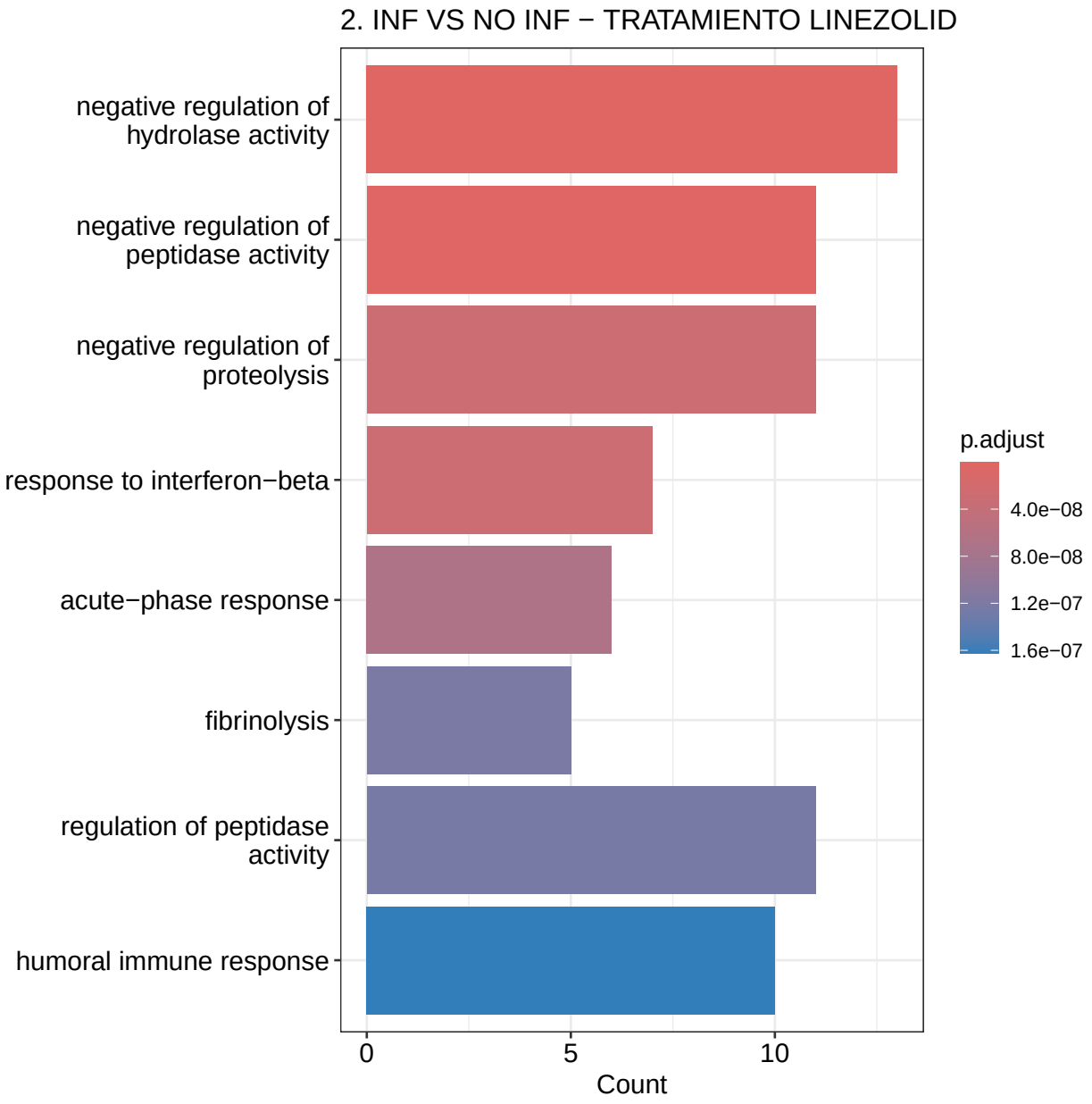
```
> go_enrichment.2 <- enrichGO(
+     gene = topTab2_anot$ENTREZID,     # IDs de los genes de interés
+     OrgDb = org.Mm.eg.db,             # Base de datos utilizada
+     keyType = "ENTREZID",             # Tipo de ID utilizado
+     ont = "BP",                       # Ontología: BP (Biological Process), MF (Molecular Function),
+     pAdjustMethod = "BH",             # Método de corrección para múltiples comparaciones (por ejempl
+     qvalueCutoff = 0.05,              # Valor p corregido (FDR) umbral
+     readable = TRUE                   # Convertimos IDs a nombres legibles
+ )
>
> # Resultados
> head(go_enrichment.2)
```

```
                   ID                                 Description GeneRatio
GO:0051346 GO:0051346 negative regulation of hydrolase activity     13/57
GO:0010466 GO:0010466 negative regulation of peptidase activity     11/57
GO:0045861 GO:0045861        negative regulation of proteolysis     11/57
GO:0035456 GO:0035456               response to interferon-beta      7/57
GO:0006953 GO:0006953                     acute-phase response      6/57
GO:0042730 GO:0042730                             fibrinolysis      5/57
             BgRatio RichFactor FoldEnrichment   zScore       pvalue
GO:0051346 329/28905 0.03951368       20.03759 15.43718 6.632351e-14
GO:0010466 251/28905 0.04382470       22.22374 15.01152 2.205056e-12
GO:0045861 357/28905 0.03081232       15.62509 12.35957 9.719672e-11
GO:0035456  72/28905 0.09722222       49.30190 18.24081 1.055917e-10
GO:0006953  44/28905 0.13636364       69.15072 20.10942 2.987106e-10
GO:0042730  22/28905 0.22727273      115.25120 23.82920 6.394698e-10
               p.adjust       qvalue
GO:0051346 7.521086e-11 4.475092e-11
GO:0010466 1.250267e-09 7.439162e-10
GO:0045861 2.993525e-08 1.781166e-08
GO:0035456 2.993525e-08 1.781166e-08
GO:0006953 6.774755e-08 4.031020e-08
GO:0042730 1.208598e-07 7.191231e-08
                                                                                                  geneID
GO:0051346 Ngp/Stfa3/Serping1/Apcs/Serpinc1/Itih4/Vtn/Itih3/Serpina1c/Serpina3k/Pzp/Serpina1b/Apoa1
GO:0010466            Ngp/Stfa3/Serping1/Serpinc1/Itih4/Vtn/Itih3/Serpina1c/Serpina3k/Pzp/Serpina1b
GO:0045861            Ngp/Stfa3/Serping1/Serpinc1/Itih4/Vtn/Itih3/Serpina1c/Serpina3k/Pzp/Serpina1b
GO:0035456                                                 Ifitm6/Ifitm1/Acod1/Iigp1/Ifi202b/Gbp2b/Gbp2
GO:0006953                                                          Saa3/Orm1/Saa1/Itih4/Saa2/Serpina1b
GO:0042730                                                                  Serping1/Fga/Vtn/Fgg/Fgb
           Count
GO:0051346    13
GO:0010466    11
GO:0045861    11
GO:0035456     7
GO:0006953     6
GO:0042730     5
```

```
> # Gráfico de puntos
> dotplot(go_enrichment.2, title = "2. INF VS NO INF - TRATAMIENTO LINEZOLID")
```




```
> # Gráfico de barras
> barplot(go_enrichment.2, title = "2. INF VS NO INF - TRATAMIENTO LINEZOLID")
```



4.6.2 ANÁLISIS DE SOBRE-REPRESENTACIÓN. CONTRASTE 3. INFECTADOS VS NO INFECTADOS - TRATAMIENTO CON VANCOMICINA

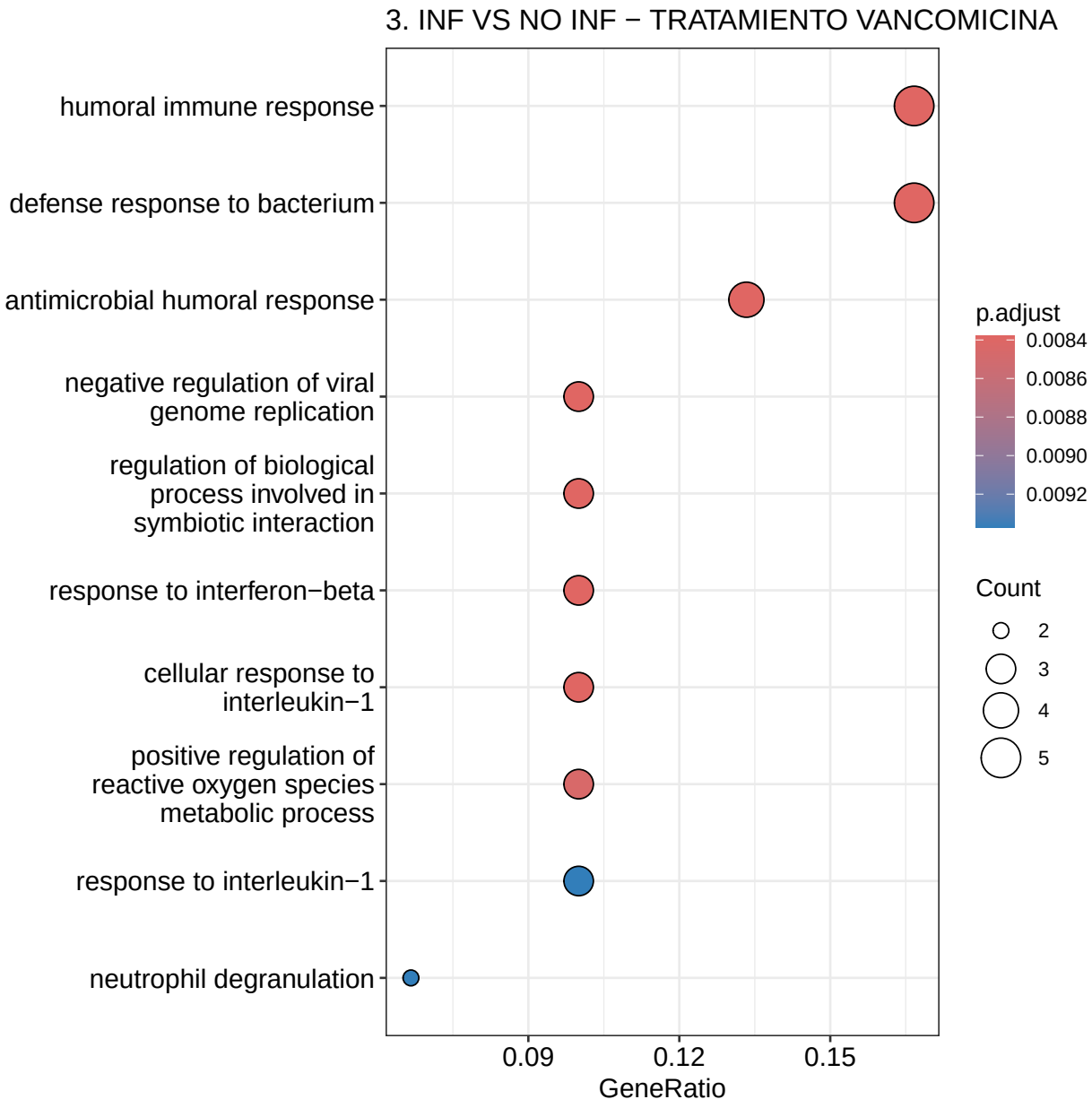
```
> library(clusterProfiler)
> library(org.Mm.eg.db)  # Base de datos de genes de ratón
>
> # Análisis de sobre-representación usando Gene Ontology (GO)
> go_enrichment.3 <- enrichGO(
+     gene = topTab3_anot$ENTREZID,      # IDs de los genes de interés
+     OrgDb = org.Mm.eg.db,              # Base de datos utilizada
+     keyType = "ENTREZID",              # Tipo de ID utilizado
+     ont = "BP",                        # Ontología: BP (Biological Process), MF (Molecular Function),
```

```
+     pAdjustMethod = "BH",             # Método de corrección para múltiples comparaciones (por ejempl
+     qvalueCutoff = 0.05,              # Valor p corregido (FDR) umbral
+     readable = TRUE                   # Convertimos IDs a nombres legibles
+ )
>
> # Resultados
> head(go_enrichment.3)
```

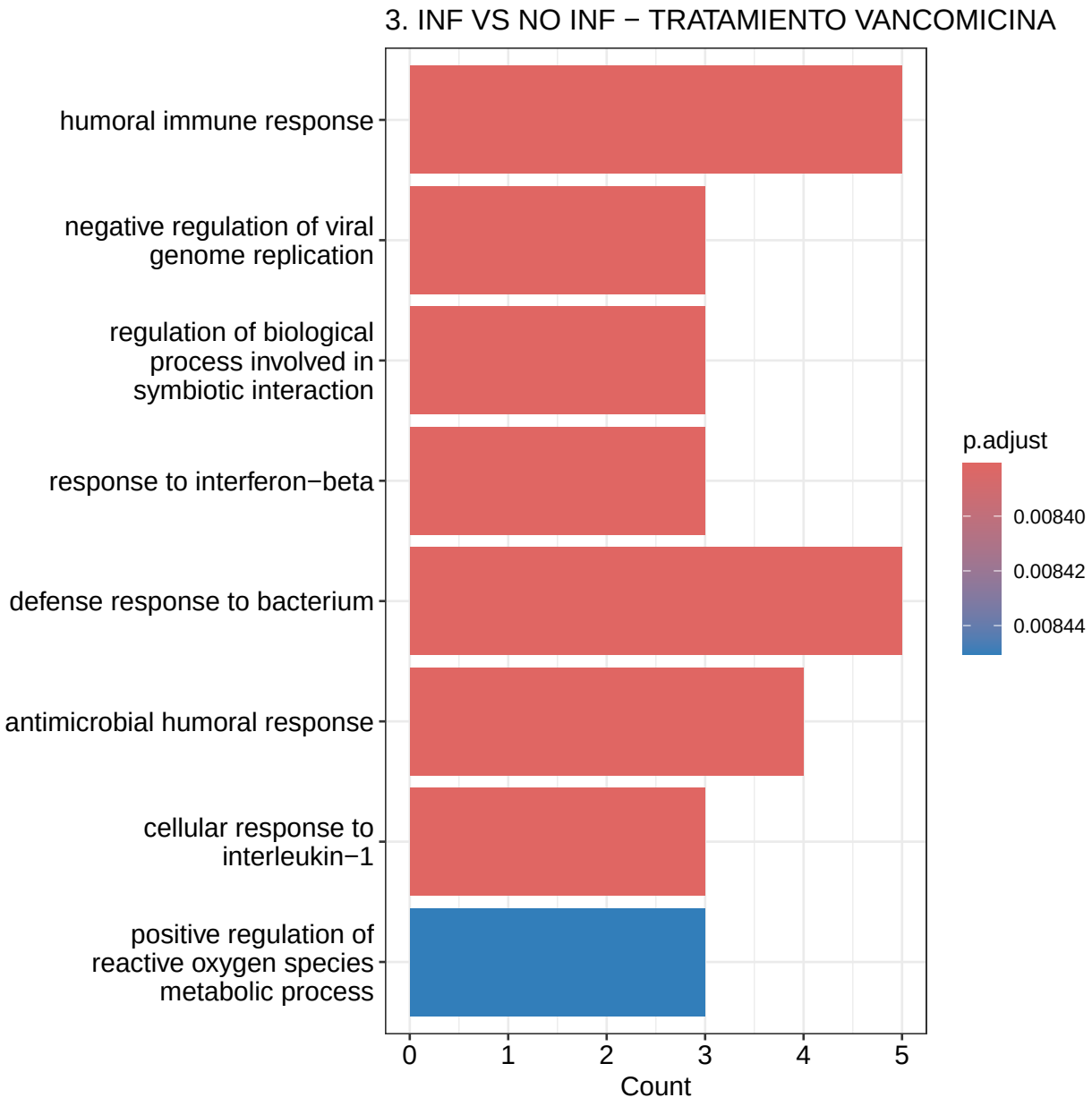
```
                  ID
GO:0006959 GO:0006959
GO:0045071 GO:0045071
GO:0043903 GO:0043903
GO:0035456 GO:0035456
GO:0042742 GO:0042742
GO:0019730 GO:0019730
                                                                Description
GO:0006959                                             humoral immune response
GO:0045071                     negative regulation of viral genome replication
GO:0043903 regulation of biological process involved in symbiotic interaction
GO:0035456                                       response to interferon-beta
GO:0042742                                    defense response to bacterium
GO:0019730                                  antimicrobial humoral response
           GeneRatio   BgRatio RichFactor FoldEnrichment    zScore       pvalue
GO:0006959      5/30 339/28905 0.01474926       14.21091  7.886542 2.412660e-05
GO:0045071      3/30  67/28905 0.04477612       43.14179 11.131314 4.620777e-05
GO:0043903      3/30  68/28905 0.04411765       42.50735 11.045441 4.830661e-05
GO:0035456      3/30  72/28905 0.04166667       40.14583 10.719770 5.732602e-05
GO:0042742      5/30 430/28905 0.01162791       11.20349  6.871153 7.464220e-05
GO:0019730      4/30 222/28905 0.01801802       17.36036  7.887433 7.933960e-05
              p.adjust      qvalue                          geneID Count
GO:0006959 0.008380572 0.004398116 Wfdc21/Fcer2a/Wfdc17/Ltf/Acod1     5
GO:0045071 0.008380572 0.004398116              Ifitm6/Ifitm1/Ltf     3
GO:0043903 0.008380572 0.004398116              Ifitm6/Ifitm1/Ltf     3
GO:0035456 0.008380572 0.004398116            Ifitm6/Ifitm1/Acod1     3
GO:0042742 0.008380572 0.004398116   Lcn2/Anxa3/Wfdc21/Wfdc17/Ltf     5
GO:0019730 0.008380572 0.004398116        Wfdc21/Wfdc17/Ltf/Acod1     4
```

```
> # Gráfico de puntos
> dotplot(go_enrichment.3, title = "3. INF VS NO INF - TRATAMIENTO VANCOMICINA")
```

GRÁFICOS




```
> # Gráfico de barras
> barplot(go_enrichment.3, title = "3. INF VS NO INF - TRATAMIENTO VANCOMICINA")
```



4.6.3 TABLAS COMPARATIVAS DE LOS GENES y PROCESOS METABÓLICOS MÁS SIGNIFICATIVOS

```
> library(dplyr)
> library(knitr)
> library(kableExtra)
>
> # Extraemos los 3 procesos más significativos de cada análisis de enriquecimiento GO
> topTab1_DF <- as.data.frame(topTab1_anot) %>%
+   arrange(adj.P.Val) %>%  # Ordenamos por p-value ajustado
+   slice_head(n = 8) %>%  # Seleccionamos los 3 más significativos
+   select(GENENAME, SYMBOL) %>%   # Seleccionamos las columnas "GENENAME" y SYMBOL
```

```
+   mutate(GEN = paste(GENENAME, "(", SYMBOL, ")", sep = " "))
>
> topTab2_DF <- as.data.frame(topTab2_anot) %>%
+   arrange(adj.P.Val) %>%
+   slice_head(n = 8) %>%
+   select(GENENAME, SYMBOL) %>%   # Seleccionamos las columnas "GENENAME" y SYMBOL
+   mutate(GEN = paste(GENENAME, "(", SYMBOL, ")", sep = " "))
>
> topTab3_DF <- as.data.frame(topTab3_anot) %>%
+   arrange(adj.P.Val) %>%
+   slice_head(n = 8) %>%
+   select(GENENAME, SYMBOL) %>%   # Seleccionamos las columnas "GENENAME" y SYMBOL
+   mutate(GEN = paste(GENENAME, "(", SYMBOL, ")", sep = " "))
>
> # Tabla combinada con los resultados
> comparison_table <- data.frame(
+   "SIN TRATAMIENTO" = topTab1_DF$GEN,
+   "TRATAMIENTO LINEZOLID" = topTab2_DF$GEN,
+   "TRATAMIENTO VANCOMICINA" = topTab3_DF$GEN
+ )
>
> # Tabla kable con los resultados
> kable(comparison_table) %>%
+   # Añadimos título a la tabla que abarque las 3 columnas
+   add_header_above(c("GENES MÁS SIGNIFICATIVOS POR CONTRASTE"=3)) %>%
+   # Añadimos un estilo
+   kable_styling(bootstrap_options = c("striped", "hover"),
+                 position = "center") %>% # Posición del texto centrada
+   column_spec(1, width = "15em") %>%  # Ajustamos el ancho relativo de la primera columna
+   column_spec(2, width = "15em") %>%
+   column_spec(3, width = "15em")
```

GENES MÁS SIGNIFICATIVOS POR CONTRASTE		
SIN.TRATAMIENTO	TRATAMIENTO.LINEZOLID	TRATAMIENTO.VANCOMICINA
lipase, endothelial (Lipg)	lipase, endothelial (Lipg)	lipase, endothelial (Lipg)
lipocalin 2 (Lcn2)	lipocalin 2 (Lcn2)	lipocalin 2 (Lcn2)
formyl peptide receptor 2 (Fpr2)	uridine phosphorylase 1 (Upp1)	prokineticin 2 (Prok2)
neutrophilic granule protein (Ngp)	prokineticin 2 (Prok2)	neutrophilic granule protein (Ngp)
interferon induced transmembrane protein 6 (Ifitm6)	interferon induced transmembrane protein 6 (Ifitm6)	interferon induced transmembrane protein 6 (Ifitm6)
interleukin 1 receptor, type II (Il1r2)	neutrophilic granule protein (Ngp)	stefin A3 (Stfa3)
expressed sequence AA467197 (AA467197)	WAP four-disulfide core domain 17 (Wfdc17)	uridine phosphorylase 1 (Upp1)
annexin A1 (Anxa1)	leucine-rich alpha-2-glycoprotein 1 (Lrg1)	olfactomedin 4 (Olfm4)

```
> library(dplyr)
> library(knitr)
> library(kableExtra)
```

```
>
> # Extraemos los 3 procesos más significativos de cada análisis de enriquecimiento GO
> top_go_1 <- as.data.frame(go_enrichment.1) %>%
+   arrange(p.adjust) %>%  # Ordenamos por p-value ajustado
+   slice_head(n = 8) %>%  # Seleccionamos los 3 más significativos
+   select(Description)    # Nos quedamos solo con la columna "Description"
>
> top_go_2 <- as.data.frame(go_enrichment.2) %>%
+   arrange(p.adjust) %>%
+   slice_head(n = 8) %>%
+   select(Description)
>
> top_go_3 <- as.data.frame(go_enrichment.3) %>%
+   arrange(p.adjust) %>%
+   slice_head(n = 8) %>%
+   select(Description)
>
> # Tabla combinada con los resultados
> comparison_table <- data.frame(
+   "SIN TRATAMIENTO" = top_go_1$Description,
+   "TRATAMIENTO LINEZOLID" = top_go_2$Description,
+   "TRATAMIENTO VANCOMICINA" = top_go_3$Description
+ )
>
> # Tabla kable con los resultados
> kable(comparison_table) %>%
+   # Añadimos título a la tabla que abarque las 3 columnas
+   add_header_above(c("PROCESOS BIOLÓGICOS MÁS SIGNIFICATIVOS POR CONTRASTE"=3)) %>%
+   # Añadimos un estilo
+   kable_styling(bootstrap_options = c("striped", "hover"),
+                 position = "center") %>% # Posición del texto centrada
+   column_spec(1, width = "15em") %>%  # Ajustamos el ancho relativo de la primera columna
+   column_spec(2, width = "15em") %>%
+   column_spec(3, width = "15em")
```

PROCESOS BIOLÓGICOS MÁS SIGNIFICATIVOS POR CONTRASTE		
SIN.TRATAMIENTO	TRATAMIENTO.LINEZOLID	TRATAMIENTO.VANCOMICINA
response to lipopolysaccharide	negative regulation of hydrolase activity	humoral immune response
cellular response to lipopolysaccharide	negative regulation of peptidase activity	negative regulation of viral genome replication
cellular response to molecule of bacterial origin	negative regulation of proteolysis	regulation of biological process involved in symbiotic interaction
response to molecule of bacterial origin	response to interferon-beta	response to interferon-beta
defense response to bacterium	acute-phase response	defense response to bacterium
cellular response to biotic stimulus	fibrinolysis	antimicrobial humoral response
immune response-regulating cell surface receptor signaling pathway	regulation of peptidase activity	cellular response to interleukin-1

humoral immune response	humoral immune response	positive regulation of reactive oxygen species metabolic process

Fuentes:

https://bioconductor.org/packages/release/bioc/html/clusterProfiler.html

https://rpubs.com/Alexis22/ClusterProfiler

https://cran.r-project.org/web/packages/kableExtra/vignettes/awesome_table_in_html.html#Grouped_Columns___Rows

Gonzalo, R., & Sanchez-Pla, A. (2019). Statistical analysis of microarray data. Universidad Oberta de Catalunya. Asignatura: Análisis de datos ómicos

5. DISCUSIÓN Y LIMITACIONES

El análisis de los datos de microarrays realizado sobre los datos crudos obtenidos en el estudio de Sharma-Kuinkel et al. (2013) ha permitido identificar la presencia de diferencias en la expresión génica en sangre de ratón (*Mus musculus*) en respuesta a la infección por *Staphylococcus aureus* resistente a la meticilina (MRSA) y a los tratamientos con antibióticos linezolid y vancomicina.

A través de la normalización de los datos de microarrays y de la eliminación de posibles sesgos mediante el filtrado del 10% de los datos, se ha logrado obtener un conjunto más claro y robusto de genes diferencialmente expresados, lo que ha facilitado la identificación de procesos biológicos relevantes en los que estos genes se ven implicados. Esto ha aportado información relevante sobre cómo la infección y los tratamientos con Linezolid o Vancomicina modulan la expresión génica y los procesos biológicos, en distinta medida y manera, en este organismo.

En cuanto a las limitaciones o dificultades identificadas durante el proceso de realización de esta PEC2, se pueden listar las siguientes:

- **Tamaño de la muestra reducido**: Aunque el tamaño de la muestra del estudio es adecuado, un tamaño mayor hubiera aportado seguramente mayor robustez estadística a los datos, sobre todo considerando que en nuestro caso, se realizó una selección de muestras menor, a partir de las muestras originales del estudio.
- **Cabe destacar la importancia, de cara a futuros análisis, de ordenar el dataframe de menor a mayor según la columna de códigos (`targetsDf$sample`)**. Esto permite que la función `read.celfiles()` establezca correctamente las correspondencias entre los archivos `.CEL` y las filas de `targetsDf`, puesto que esta función ordena previamente los archivos `.CEL` de menor a mayor según su nombre. El desconocimiento de este paso, fue motivo de resultados incoherentes durante la exploración de los datos, hasta que por fin se dio con la causa del problema.
- **La interpretación biológica de nuestro análisis es sesgada, debido a la falta de estudio y profundización sobre el tema y al haber realizado un análisis de la expresión génica general**, en lugar de centrarnos en ciertos genes que podrían considerarse de mayor interés como los mediadores de la inflamación y fiebre, como las citoquinas $IL-1\beta$, $IL-6$ y $TNF-\alpha$ (Sharma-Kuinkel et al., 2013)). Esta modulación de la fiebre y la inflamación (sepsis) se vería reducida después de que los antibióticos logren disminuir los niveles de toxinas bacterianas en sangre (En el estudio miden los niveles de leucocidina de Panton-Valentine (PVL) y alpha hemolisina).

  En resumen, Sharma-Kuinkel et al. (2013) realiza un estudio con un diseño más detallado y complejo, en comparación al nuestro, que se trata de un análisis de la expresión génica general y más simplificado.
- **El proceso de filtrado realizado en nuestro análisis requiere de mayor estudio**, con el fin de asegurar que no se ha perdido información importante en este paso.

- **El análisis de significación biológica requiere de mayor profundización y contraste mediante la utilización de otros tipos de análisis como *Gene Set Enrichment Analysis* o de otros paquetes con los que realizar este análisis, como `GOstats` o `ReactomePA`**. También sería interesante utilizar otros métodos para graficar los resultados, que nos permitan comparar con mayor precisión las rutas metabólicas involucradas y la interacción o regulación molecular que hay entre las mismas.

6. CONCLUSIONES

El análisis de datos de microarray, a partir de los datos crudos del estudio de Sharma-Kuinkel et al. (2013), para los tres contrastes propuestos (evolución de no infectado a infectado: sin tratamiento, tratamiento con linezolid y tratamiento con vancomicina), **ha proporcionado una visión valiosa sobre los efectos de ambos antibióticos y la ausencia de los mismos, en la expresión génica en un modelo murino (*Mus musculus*) de sepsis por *Staphylococcus aureus* resistente a meticilina (MRSA)**. Durante el proceso de desarrollo de este análisis con el fin de identificar los genes más significativos y los procesos biológicos asociados a cada contraste, se han obtenido conclusiones tanto a nivel de interpretación del análisis como desde el punto de vista del aprendizaje del proceso de *Análisis de Datos de Microarrays*.

La conclusión principal es la de haber cumplido de manera general con los objetivos propuestos.

De manera más específica, en referencia a los objetivos didácticos se puede concluir que se ha desarrollado un aprendizaje satisfactorio en cuanto a:

1. **El manejo de datos de microarrays**, al haber realizado tareas de obtención y procesado de datos, manejo de archivos `.CEL`, así como la creación de distintos `ExpressionSet`, junto al manejo de sus metadatos asociados.

2. **La aplicación de técnicas de normalización y control de calidad** que aseguren la fiabilidad de los datos y poder evaluar su adecuación para el análisis estadístico.

3. **La implementación de estrategias de filtrado para seleccionar el 10% de las sondas con mayor variabilidad y relevancia biológica**, habiéndose usado el método del rango intercuartílico (IQR) como medida de variabilidad.

4. **El diseño y realización de un análisis de contraste estadístico utilizando funciones del paquete `limma`** para la identificación de genes diferencialmente expresados entre los tres contrastes propuestos.

5. **La anotación de listas de los genes seleccionados** con identificadores genómicos (como `GENENAME`, `SYMBOL`, `ENSEMBL` y `ENTREZID`) para facilitar su interpretación biológica.

6. **La realización de un análisis de significación biológica**, empleando la base de datos `GO`, y el paquete `clusterProfiler` para asociar genes diferencialmente expresados con sus rutas metabólicas y funciones celulares. Para ello se realizó, para cada contraste, un análisis de sobre-representación.

7. **El desarrollo de la capacidad de interpretación y obtención de conclusiones relevantes a partir de datos transcriptómicos**. Este objetivo se ha cumplimentado con éxito al ser el que ha permitido completar el informe y el análisis con éxito, debido a que sin la comprensión del por qué y para qué de cada paso, difícilmente se hubiera alcanzado el desarrollo de este trabajo con una exposición clara de las ideas y con resultados coherentes con el estudio propuesto.

Por otro lado, en cuanto a las conclusiones relacionadas con los objetivos sobre **la interpretación biológica de los resultados del análisis**:

En general. se han observado cambios en la expresión génica de *Mus musculus* inducidos por la infección con MRSA, y se destacan los siguientes:

Para los tres tratamientos, los dos genes expresados de manera diferencial más significativamente, coinciden en ser **la lipasa endotelial (Lipg) y la lipocalina 2 (Lcn2)** relacionadas con el aumento de citoquinas que modulan la inflamación o sepsis como defensa (Pierart et al., 2012; Saeeda Al Jaberi et al., 2021).

Para el resto de genes se observan diferencias en los tipos de genes y la significancia estadística con la que aparecen los genes repetidos, sugiriendo niveles de expresión diferenciales distintos.

En resumen, **para cada condición se puede concluir lo siguiente:**

- **En ausencia de tratamiento:**

  Destaca la presencia de **genes relacionados en procesos de inflamación en respuesta a toxinas o deterioro de tejidos**. Por ejemplo, se puede destacar el aumento en la expresión del receptor de **interleucina 1, tipo II (Il1r2)** que suele estar relacionado con el aumento de citoquinas mediadoras de la inflamación (Zhang et al., 2024).

  En cuanto a los procesos biológicos que modulan los genes diferencialmente expresados en este tratamiento sin antibiótico, destacan **la respuesta a lipopolisacárido**, que es fundamental en la activación del sistema inmune frente a infecciones bacterianas, y **la respuesta celular a moléculas de origen bacteriano**, entre otros, que está estrechamente vinculada a la detección y respuesta frente a patógenos como S. aureus (Zecconi & Scali, 2013).

- **Tratamiento antibiótico con linezolid:**

  Destaca una expresión diferencial de genes distinta a la de la vancomicina y a la de ausencia de antibióticos. Algunos de estos genes son la **uridina fosforilasa 1 (Upp1), WAP four-disulfide core domain 17 (Wfdc17) y leucine-rich alpha-2-glycoprotein 1 (Lrg1)**.

  En cuanto a los procesos biológicos implicados por los genes diferencialmente expresados podríamos subrayar **la regulación negativa de la actividad de hidrolasas, peptidasas y la proteólisis**, entre otros.

- **Tratamiento antibiótico con Vancomicina:**

  Se vuelven a observar, genes diferencialmente expresados distintos como la **olfactomedin 4 (Olfm4), la prokineticina 2 (Prok2), o stefin A3 (Stfa3)** entre otros.

  Las respuestas metabólicas destacadas son **la respuesta inmune humoral y la regulación de procesos biológicos envueltos en interacciones simbióticas**.

En conclusión, en el presente trabajo se considera haber realizado un aprendizaje adecuado de los objetivos propuestos, **habiendo desarrollado (siempre en proceso de mejora) las habilidades y competencias relacionadas con el uso de herramientas bioinformáticas y el proceso de Análisis de Datos de Microarrays.**

En relación a las conclusiones de carácter biológico del análisis, existen diferencias significativas en la expresión diferencial de genes para las tres hipótesis planteadas, modulando estos genes distintas respuestas a la infección por MRSA según el tratamiento. Además, en base a los resultados **es posible sugerir que el mecanismo de acción del antibiótico Linezolid, es distinto al mecanismo de acción de la Vancomicina, al modular distintas respuestas del sistema inmune en el modelo murino** (*Mus musculus*).

7. BIBLIOGRAFÍA

Pierart Z, Camila, & Serrano L, Valentina. (2012). Lipasa endotelial y su relación con la enfermedad cardiovascular y diabetes mellitus tipo 2. Revista médica de Chile, 140(3), 373-378. https://dx.doi.org/10.4067/S0034-98872012000300015

Gonzalo, R., & Sanchez-Pla, A. (2019). Statistical analysis of microarray data. Universidad Oberta de Catalunya. Asignatura: Análisis de datos ómicos. https://aspteaching.github.io/Omics_Data_Analysis-Case_Study_1-Microarrays/Case_Study_1-Microarrays_Analysis.html#read-the-cel-files

Saeeda Al Jaberi, Cohen, A., D'Souza, C., Abdulrazzaq, Y. M., Ojha, S., Bastaki, S., & Adeghate, E. A. (2021). Lipocalin-2: Structure, function, distribution and role in metabolic disorders. Biomedicine & Pharmacotherapy, 142, 112002. https://doi.org/10.1016/j.biopha.2021.112002

Sharma-Kuinkel BK, Zhang Y, Yan Q, Ahn SH et al. (2013). Host gene expression profiling and in vivo cytokine studies to characterize the role of linezolid and vancomycin in methicillin-resistant Staphylococcus aureus (MRSA) murine sepsis model. PLoS One ;8(4):e60463. <PMID: 23565251>

Zhang, Y., Liu, K., Guo, M., Yang, Y., & Zhang, H. (2024). Negative regulator IL-1 receptor 2 (IL-1R2) and its roles in immune regulation of autoimmune diseases. International Immunopharmacology, 136, 112400. https://doi.org/10.1016/j.intimp.2024.112400

Zecconi, A., & Scali, F. (2013). Staphylococcus aureus virulence factors in evasion from innate immune defenses in human and animal diseases. Immunology Letters, 150(1–2), 12–22. https://doi.org/10.1016/j.imlet.2013.01.004

https://bioconductor.org/packages/release/bioc/html/clusterProfiler.html

https://rpubs.com/Alexis22/ClusterProfiler

https://gtk-teaching.github.io/Microarrays-R/05-DataNormalisation/index.html

https://cran.r-project.org/web/packages/kableExtra/vignettes/awesome_table_in_html.html#Grouped_Columns___Rows

8. APÉNDICES

Apéndice 1

En este apéndice se encuentran algunas comprobaciones relacionadas con la exploración.

Estructura de los datos

```
> str(rawData)
```

```
Formal class 'ExpressionFeatureSet' [package "oligoClasses"] with 9 slots
  ..@ manufacturer     : chr "Affymetrix"
  ..@ intensityFile    : chr NA
  ..@ assayData        :<environment: 0x0000020a27f789d8>
  ..@ phenoData        :Formal class 'AnnotatedDataFrame' [package "Biobase"] with 4 slots
  .. .. ..@ varMetadata      :'data.frame': 9 obs. of  2 variables:
  .. .. .. ..$ labelDescription: chr [1:9] NA NA NA NA ...
  .. .. .. ..$ channel         : Factor w/ 2 levels "exprs","_ALL_": 2 2 2 2 2 2 2 2 2
  .. .. ..@ data             :'data.frame': 24 obs. of  9 variables:
  .. .. .. ..$ sample         : chr [1:24] "GSM944833" "GSM944834" "GSM944835" "GSM944836" ...
  .. .. .. ..$ infection      : chr [1:24] "uninfected" "uninfected" "S. aureus USA300" "S. aureus USA3
  .. .. .. ..$ time           : chr [1:24] "hour 0" "hour 0" "hour 24" "hour 24" ...
  .. .. .. ..$ agent          : chr [1:24] "linezolid" "vancomycin" "untreated" "linezolid" ...
  .. .. .. ..$ infection_short: chr [1:24] "Uninf" "Uninf" "Inf" "Inf" ...
  .. .. .. ..$ time_short     : chr [1:24] "h0" "h0" "h24" "h24" ...
  .. .. .. ..$ agent_short    : chr [1:24] "linez" "vanco" "untreat" "linez" ...
  .. .. .. ..$ Group          : chr [1:24] "Uninf.h0.linez" "Uninf.h0.vanco" "Inf.h24.untreat" "Inf.h24
```

```
.. .. .. ..$ ShortName      : chr [1:24] "Uninf.h0.linez.1" "Uninf.h0.vanco.2" "Inf.h24.untreat.2" "In
.. .. ..@ dimLabels        : chr [1:2] "rowNames" "columnNames"
.. .. ..@ .__classVersion__:Formal class 'Versions' [package "Biobase"] with 1 slot
.. .. .. .. ..@ .Data:List of 1
.. .. .. .. .. ..$ : int [1:3] 1 1 0
.. .. .. .. ..$ names: chr "AnnotatedDataFrame"
..@ featureData      :Formal class 'AnnotatedDataFrame' [package "Biobase"] with 4 slots
.. .. ..@ varMetadata      :'data.frame': 0 obs. of  1 variable:
.. .. .. ..$ labelDescription: chr(0)
.. .. ..@ data             :'data.frame': 1004004 obs. of  0 variables
.. .. ..@ dimLabels        : chr [1:2] "featureNames" "featureColumns"
.. .. ..@ .__classVersion__:Formal class 'Versions' [package "Biobase"] with 1 slot
.. .. .. .. ..@ .Data:List of 1
.. .. .. .. .. ..$ : int [1:3] 1 1 0
.. .. .. .. ..$ names: chr "AnnotatedDataFrame"
..@ experimentData   :Formal class 'MIAME' [package "Biobase"] with 13 slots
.. .. ..@ name             : chr ""
.. .. ..@ lab              : chr ""
.. .. ..@ contact          : chr ""
.. .. ..@ title            : chr ""
.. .. ..@ abstract         : chr ""
.. .. ..@ url              : chr ""
.. .. ..@ pubMedIds        : chr ""
.. .. ..@ samples          : list()
.. .. ..@ hybridizations   : list()
.. .. ..@ normControls     : list()
.. .. ..@ preprocessing    : list()
.. .. ..@ other            : list()
.. .. ..@ .__classVersion__:Formal class 'Versions' [package "Biobase"] with 1 slot
.. .. .. .. ..@ .Data:List of 2
.. .. .. .. .. ..$ : int [1:3] 1 0 0
.. .. .. .. .. ..$ : int [1:3] 1 1 0
.. .. .. .. ..$ names: chr [1:2] "MIAxE" "MIAME"
..@ annotation       : chr "pd.mouse430.2"
..@ protocolData     :Formal class 'AnnotatedDataFrame' [package "Biobase"] with 4 slots
.. .. ..@ varMetadata      :'data.frame': 2 obs. of  2 variables:
.. .. .. ..$ labelDescription: chr [1:2] "Names of files used in 'exprs'" "Run dates for files used in
.. .. .. ..$ channel         : Factor w/ 2 levels "exprs","_ALL_": 2 2
.. .. ..@ data             :'data.frame': 24 obs. of  2 variables:
.. .. .. ..$ exprs: chr [1:24] "./data/Archivos_cel/GSM944833.CEL" "./data/Archivos_cel/GSM944834.CEL"
.. .. .. ..$ dates: chr [1:24] "2010-11-10T16:32:50Z" "2010-11-10T16:42:09Z" "2010-11-10T16:51:22Z" "2
.. .. ..@ dimLabels        : chr [1:2] "rowNames" "columnNames"
.. .. ..@ .__classVersion__:Formal class 'Versions' [package "Biobase"] with 1 slot
.. .. .. .. ..@ .Data:List of 1
.. .. .. .. .. ..$ : int [1:3] 1 1 0
.. .. .. .. ..$ names: chr "AnnotatedDataFrame"
..@ .__classVersion__:Formal class 'Versions' [package "Biobase"] with 1 slot
.. .. ..@ .Data:List of 4
.. .. .. ..$ : int [1:3] 4 4 0
.. .. .. ..$ : int [1:3] 2 64 0
.. .. .. ..$ : int [1:3] 1 3 0
.. .. .. ..$ : int [1:3] 1 0 0
.. .. ..$ names: chr [1:4] "R" "Biobase" "eSet" "NChannelSet"
```

Comprobamos si los nombres de las muestras coinciden entre `targetsDf` y `pData(rawData)`

```
> # Debe devolver TRUE si coinciden
> identical(rownames(targetsDf$sample), rownames(pData(rawData)$sample))
```

```
[1] TRUE
```

Comprobamos que el orden de las muestras para las columnas en rawData y las filas de pData es el mismo.

```
> # Verificamos que los nombres de las muestras en rawData y el pData
> all(colnames(rawData) == rownames(pData(rawData)))
```

```
[1] TRUE
```

```
> identical(sampleNames(rawData), rownames(pData(rawData)))
```

```
[1] TRUE
```

Dimensiones del `ExpresionSet`.

```
> dim(rawData)
```

```
Features  Samples
 1004004       24
```

Se comprueba que los datos fenotípicos de rawData son correctos.

```
> pData(rawData)
```

```
                     sample        infection    time      agent
Uninf.h0.linez.1   GSM944833       uninfected  hour 0  linezolid
Uninf.h0.vanco.2   GSM944834       uninfected  hour 0 vancomycin
Inf.h24.untreat.2  GSM944835 S. aureus USA300 hour 24  untreated
Inf.h24.linez.1    GSM944836 S. aureus USA300 hour 24  linezolid
Inf.h24.vanco.2    GSM944837 S. aureus USA300 hour 24 vancomycin
Uninf.h0.untreat.4 GSM944838       uninfected  hour 0  untreated
Uninf.h0.vanco.4   GSM944841       uninfected  hour 0 vancomycin
Inf.h24.vanco.3    GSM944844 S. aureus USA300 hour 24 vancomycin
Uninf.h0.untreat.3 GSM944845       uninfected  hour 0  untreated
Uninf.h0.linez.4   GSM944847       uninfected  hour 0  linezolid
Inf.h24.untreat.1  GSM944849 S. aureus USA300 hour 24  untreated
Inf.h24.linez.2    GSM944850 S. aureus USA300 hour 24  linezolid
Inf.h24.vanco.4    GSM944851 S. aureus USA300 hour 24 vancomycin
Uninf.h0.untreat.2 GSM944852       uninfected  hour 0  untreated
Uninf.h0.linez.2   GSM944854       uninfected  hour 0  linezolid
Uninf.h0.vanco.1   GSM944855       uninfected  hour 0 vancomycin
Inf.h24.untreat.4  GSM944856 S. aureus USA300 hour 24  untreated
Inf.h24.linez.3    GSM944857 S. aureus USA300 hour 24  linezolid
Uninf.h0.untreat.1 GSM944859       uninfected  hour 0  untreated
```

```
Uninf.h0.linez.3   GSM944861       uninfected  hour 0  linezolid
Uninf.h0.vanco.3   GSM944862       uninfected  hour 0 vancomycin
Inf.h24.untreat.3  GSM944863 S. aureus USA300 hour 24  untreated
Inf.h24.linez.4    GSM944864 S. aureus USA300 hour 24  linezolid
Inf.h24.vanco.1    GSM944865 S. aureus USA300 hour 24 vancomycin
                   infection_short time_short agent_short           Group
Uninf.h0.linez.1             Uninf         h0       linez   Uninf.h0.linez
Uninf.h0.vanco.2             Uninf         h0       vanco   Uninf.h0.vanco
Inf.h24.untreat.2              Inf        h24     untreat  Inf.h24.untreat
Inf.h24.linez.1                Inf        h24       linez    Inf.h24.linez
Inf.h24.vanco.2                Inf        h24       vanco    Inf.h24.vanco
Uninf.h0.untreat.4           Uninf         h0     untreat Uninf.h0.untreat
Uninf.h0.vanco.4             Uninf         h0       vanco   Uninf.h0.vanco
Inf.h24.vanco.3                Inf        h24       vanco    Inf.h24.vanco
Uninf.h0.untreat.3           Uninf         h0     untreat Uninf.h0.untreat
Uninf.h0.linez.4             Uninf         h0       linez   Uninf.h0.linez
Inf.h24.untreat.1              Inf        h24     untreat  Inf.h24.untreat
Inf.h24.linez.2                Inf        h24       linez    Inf.h24.linez
Inf.h24.vanco.4                Inf        h24       vanco    Inf.h24.vanco
Uninf.h0.untreat.2           Uninf         h0     untreat Uninf.h0.untreat
Uninf.h0.linez.2             Uninf         h0       linez   Uninf.h0.linez
Uninf.h0.vanco.1             Uninf         h0       vanco   Uninf.h0.vanco
Inf.h24.untreat.4              Inf        h24     untreat  Inf.h24.untreat
Inf.h24.linez.3                Inf        h24       linez    Inf.h24.linez
Uninf.h0.untreat.1           Uninf         h0     untreat Uninf.h0.untreat
Uninf.h0.linez.3             Uninf         h0       linez   Uninf.h0.linez
Uninf.h0.vanco.3             Uninf         h0       vanco   Uninf.h0.vanco
Inf.h24.untreat.3              Inf        h24     untreat  Inf.h24.untreat
Inf.h24.linez.4                Inf        h24       linez    Inf.h24.linez
Inf.h24.vanco.1                Inf        h24       vanco    Inf.h24.vanco
                          ShortName
Uninf.h0.linez.1     Uninf.h0.linez.1
Uninf.h0.vanco.2     Uninf.h0.vanco.2
Inf.h24.untreat.2   Inf.h24.untreat.2
Inf.h24.linez.1       Inf.h24.linez.1
Inf.h24.vanco.2       Inf.h24.vanco.2
Uninf.h0.untreat.4 Uninf.h0.untreat.4
Uninf.h0.vanco.4     Uninf.h0.vanco.4
Inf.h24.vanco.3       Inf.h24.vanco.3
Uninf.h0.untreat.3 Uninf.h0.untreat.3
Uninf.h0.linez.4     Uninf.h0.linez.4
Inf.h24.untreat.1   Inf.h24.untreat.1
Inf.h24.linez.2       Inf.h24.linez.2
Inf.h24.vanco.4       Inf.h24.vanco.4
Uninf.h0.untreat.2 Uninf.h0.untreat.2
Uninf.h0.linez.2     Uninf.h0.linez.2
Uninf.h0.vanco.1     Uninf.h0.vanco.1
Inf.h24.untreat.4   Inf.h24.untreat.4
Inf.h24.linez.3       Inf.h24.linez.3
Uninf.h0.untreat.1 Uninf.h0.untreat.1
Uninf.h0.linez.3     Uninf.h0.linez.3
Uninf.h0.vanco.3     Uninf.h0.vanco.3
Inf.h24.untreat.3   Inf.h24.untreat.3
Inf.h24.linez.4       Inf.h24.linez.4
```

```
Inf.h24.vanco.1       Inf.h24.vanco.1
```

Comprobación de los datos de expresión de assayData en el objeto rawData

```
> # Extraer la matriz de expresión de rawData
> expr_matrix <- exprs(rawData)
>
> # Ver las primeras filas de la matriz de expresión
> head(expr_matrix, 5)
```

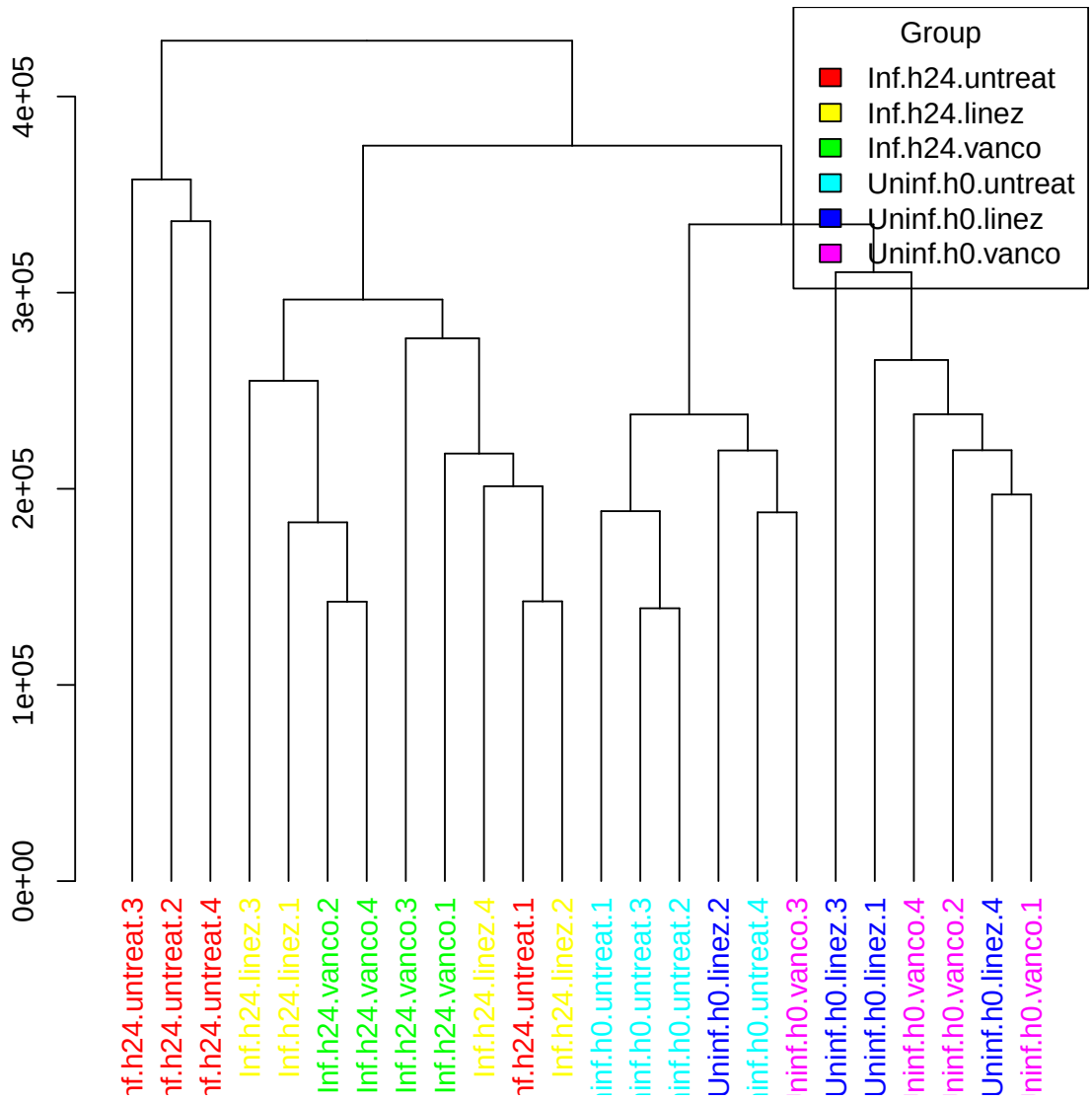
```
  Uninf.h0.linez.1 Uninf.h0.vanco.2 Inf.h24.untreat.2 Inf.h24.linez.1
1              237              165               293             196
2            22496            18858             13113           14487
3              628              556               425             426
4            22699            19351             13090           14813
5              321              251               368             326
  Inf.h24.vanco.2 Uninf.h0.untreat.4 Uninf.h0.vanco.4 Inf.h24.vanco.3
1             223                188               243             371
2           13532              14128             20067           17258
3             467                479               665             643
4           13603              13954             20460           17510
5             311                391               372             439
  Uninf.h0.untreat.3 Uninf.h0.linez.4 Inf.h24.untreat.1 Inf.h24.linez.2
1                233                324               205             163
2              15558              17424             17265           18149
3                534                594               551             486
4              15529              17062             16934           17515
5                252                617               511             263
  Inf.h24.vanco.4 Uninf.h0.untreat.2 Uninf.h0.linez.2 Uninf.h0.vanco.1
1             232                276               237             233
2           13376              14659             12904           20051
3             457                598               506             586
4           13061              15003             12999           20271
5             299                518               525             209
  Inf.h24.untreat.4 Inf.h24.linez.3 Uninf.h0.untreat.1 Uninf.h0.linez.3
1               333               365                206             290
2             12385             13342              17565           20128
3               499               567                614             709
4             12671             13277              17537           20197
5               607               586                216             317
  Uninf.h0.vanco.3 Inf.h24.untreat.3 Inf.h24.linez.4 Inf.h24.vanco.1
1              314                231             204             238
2            13779              13049           17994           19952
3              506                502             572             603
4            13746              12993           17551           20252
5              481                512             231             392
```

Apéndice 2

En este apéndice encontramos tablas y gráficos relacionados con el control de calidad de los datos en crudo y normalizados.

Cluster jerárquico de RawData con colores asociados al tipo de tratamiento de la muestra.

```
> library(dendextend)
>
> # Calculamos clustering jerárquico
> clust.euclid.average <- hclust(dist(t(exprs(rawData))),
+                                method = "average")
>
> # Convertimos clustering a dendrograma
> dend <- as.dendrogram(clust.euclid.average)
>
> # Extraemos etiquetas del dendrograma
> labels_dend <- labels(dend)
>
> # Nombres del grupo antes del 3er punto
> group_names <- sub("(\\w+\\.\\w+\\.\\w+)\\..*", "\\1", labels_dend)
>
> # Identificamos los grupos únicos
> group_levels <- unique(group_names)
>
> # Paleta de colores para los grupos
> group_colors <- setNames(rainbow(length(group_levels)), group_levels)
>
> # Asignamos colores a las etiquetas
> labels_colors(dend) <- group_colors[group_names]
>
> # Graficamos el dendrograma
> plot(dend,
+      main = "Cluster Jerárquico de rawData",
+      cex = 0.5)
>
> # Añadir la leyenda
> legend("topright", legend = group_levels,
+        fill = group_colors, title = "Group")
```



Información del eset normalizado

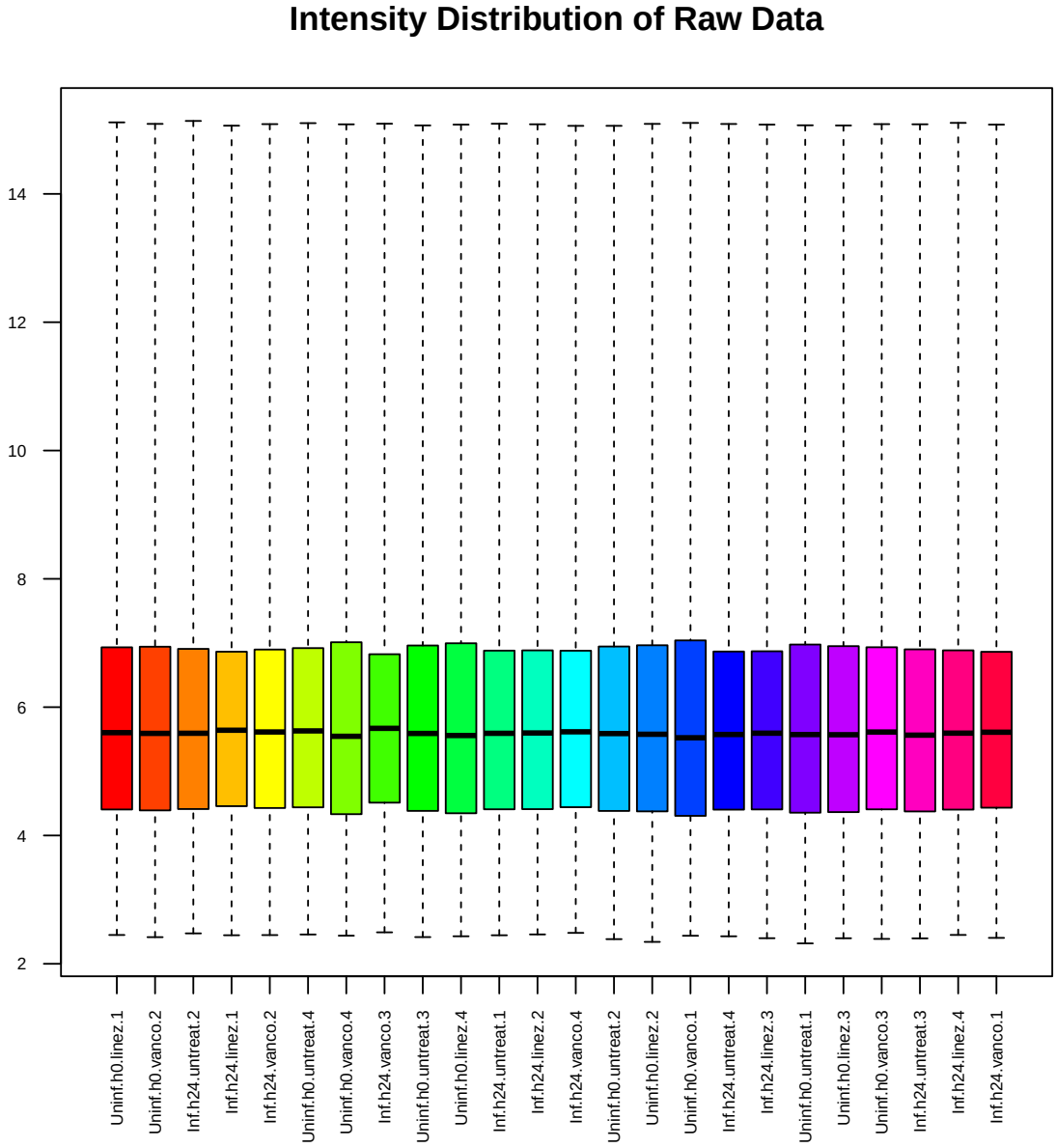
```
> eset
```

```
ExpressionSet (storageMode: lockedEnvironment)
assayData: 45101 features, 24 samples
  element names: exprs
protocolData
  rowNames: Uninf.h0.linez.1 Uninf.h0.vanco.2 ... Inf.h24.vanco.1 (24
    total)
  varLabels: exprs dates
  varMetadata: labelDescription channel
phenoData
  rowNames: Uninf.h0.linez.1 Uninf.h0.vanco.2 ... Inf.h24.vanco.1 (24
```

```
    total)
  varLabels: sample infection ... ShortName (9 total)
  varMetadata: labelDescription channel
featureData: none
experimentData: use 'experimentData(object)'
Annotation: mouse4302.db
```

Boxplot múltiple para el `ExpressionSet` normalizado.

```
> sampleColor <- rainbow(ncol(exprs(eset)))  # Un color por muestra
>
> boxplot(eset,
+         which = "all",
+         las = 2,
+         main = "Intensity Distribution of Raw Data",
+         cex.axis = 0.6,
+         col = sampleColor,
+         names = sampleNames(eset))
```



Algunas figuras de interés del control de calidad para el `ExpressionSet` de RawData usando `arrayQualityMetrics`.

Figure 4: Boxplots.

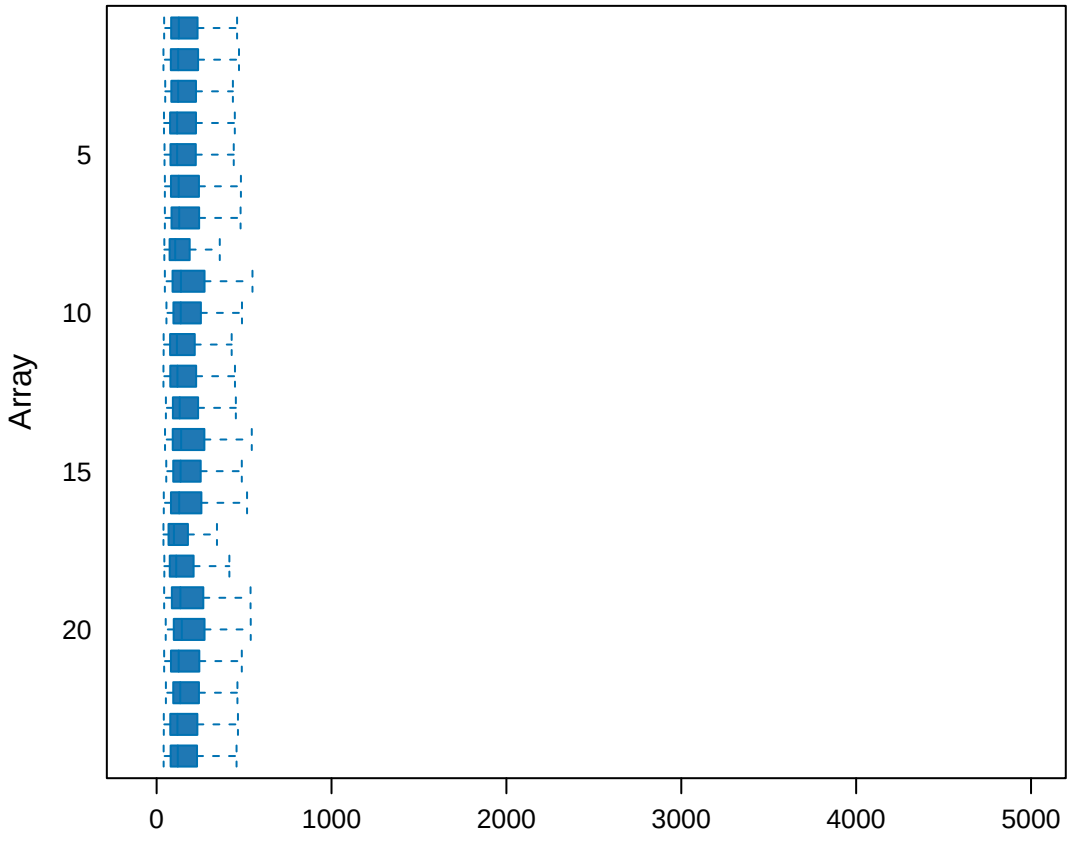


Figure 6: Density plots.

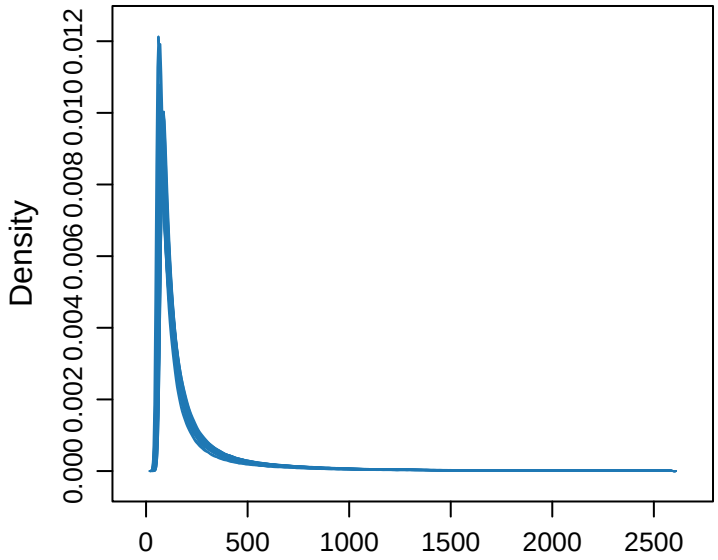
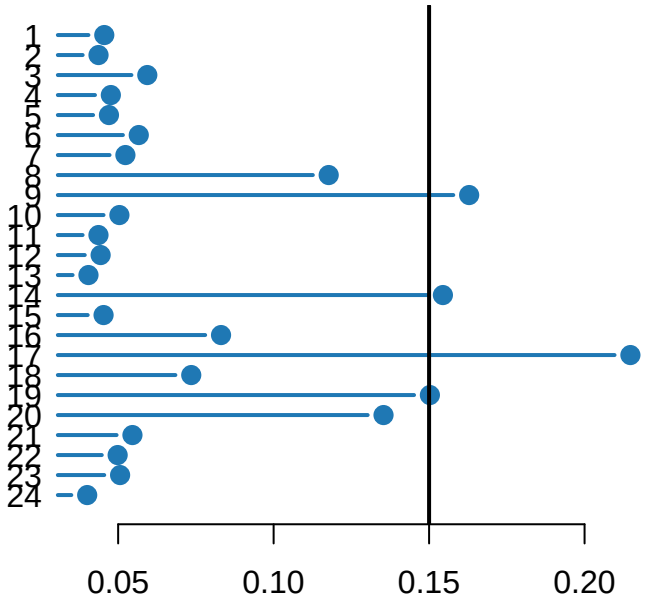


Figure 9: Outlier detection for MA plots



Control de calidad para el `ExpressionSet` normalizado usando `arrayQualityMetrics`.

```
> library(arrayQualityMetrics)
>
> dir_CC <- "C:/Users/juanm/Desktop/MASTER_BIOINFORMATICA/Analisis_Datos_Omicos/ADO_PEC2/results/CC_eset
>
> arrayQualityMetrics(eset,
+                     outdir = dir_CC, # Guardamos el resultado en su directorio
+                     force = TRUE)
```

Tabla resumen a la que podemos acceder tras abrir el archivo `index.html` obtenido.

- Array metadata and outlier detection overview

	array	sampleNames	*1	*2	*3	sample	infection	time	agent	infection_short	time_short	agent_short	Group
☐	1	Uninf.h0.linez.1				GSM944833	uninfected	hour 0	linezolid	Uninf	h0	linez	Uninf.h0.linez
☐	2	Uninf.h0.vanco.2				GSM944834	uninfected	hour 0	vancomycin	Uninf	h0	vanco	Uninf.h0.vanco
☐	3	Inf.h24.untreat.2				GSM944835	S. aureus USA300	hour 24	untreated	Inf	h24	untreat	Inf.h24.untreat
☐	4	Inf.h24.linez.1				GSM944836	S. aureus USA300	hour 24	linezolid	Inf	h24	linez	Inf.h24.linez
☐	5	Inf.h24.vanco.2				GSM944837	S. aureus USA300	hour 24	vancomycin	Inf	h24	vanco	Inf.h24.vanco
☐	6	Uninf.h0.untreat.4				GSM944838	uninfected	hour 0	untreated	Uninf	h0	untreat	Uninf.h0.untreat
☐	7	Uninf.h0.vanco.4		x		GSM944841	uninfected	hour 0	vancomycin	Uninf	h0	vanco	Uninf.h0.vanco
☐	8	Inf.h24.vanco.3	x	x		GSM944844	S. aureus USA300	hour 24	vancomycin	Inf	h24	vanco	Inf.h24.vanco
☐	9	Uninf.h0.untreat.3				GSM944845	uninfected	hour 0	untreated	Uninf	h0	untreat	Uninf.h0.untreat
☐	10	Uninf.h0.linez.4				GSM944847	uninfected	hour 0	linezolid	Uninf	h0	linez	Uninf.h0.linez
☐	11	Inf.h24.untreat.1				GSM944849	S. aureus USA300	hour 24	untreated	Inf	h24	untreat	Inf.h24.untreat
☐	12	Inf.h24.linez.2				GSM944850	S. aureus USA300	hour 24	linezolid	Inf	h24	linez	Inf.h24.linez
☐	13	Inf.h24.vanco.4				GSM944851	S. aureus USA300	hour 24	vancomycin	Inf	h24	vanco	Inf.h24.vanco
☐	14	Uninf.h0.untreat.2				GSM944852	uninfected	hour 0	untreated	Uninf	h0	untreat	Uninf.h0.untreat
☐	15	Uninf.h0.linez.2				GSM944854	uninfected	hour 0	linezolid	Uninf	h0	linez	Uninf.h0.linez
☐	16	Uninf.h0.vanco.1		x		GSM944855	uninfected	hour 0	vancomycin	Uninf	h0	vanco	Uninf.h0.vanco
☐	17	Inf.h24.untreat.4				GSM944856	S. aureus USA300	hour 24	untreated	Inf	h24	untreat	Inf.h24.untreat
☐	18	Inf.h24.linez.3				GSM944857	S. aureus USA300	hour 24	linezolid	Inf	h24	linez	Inf.h24.linez
☐	19	Uninf.h0.untreat.1				GSM944859	uninfected	hour 0	untreated	Uninf	h0	untreat	Uninf.h0.untreat
☐	20	Uninf.h0.linez.3				GSM944861	uninfected	hour 0	linezolid	Uninf	h0	linez	Uninf.h0.linez
☐	21	Uninf.h0.vanco.3				GSM944862	uninfected	hour 0	vancomycin	Uninf	h0	vanco	Uninf.h0.vanco
☐	22	Inf.h24.untreat.3				GSM944863	S. aureus USA300	hour 24	untreated	Inf	h24	untreat	Inf.h24.untreat
☐	23	Inf.h24.linez.4				GSM944864	S. aureus USA300	hour 24	linezolid	Inf	h24	linez	Inf.h24.linez
☐	24	Inf.h24.vanco.1				GSM944865	S. aureus USA300	hour 24	vancomycin	Inf	h24	vanco	Inf.h24.vanco

The columns named *1, *2, ... indicate the calls from the different outlier detection methods:

1. outlier detection by Distances between arrays
2. outlier detection by Boxplots
3. outlier detection by MA plots

Figure 4: Boxplots.

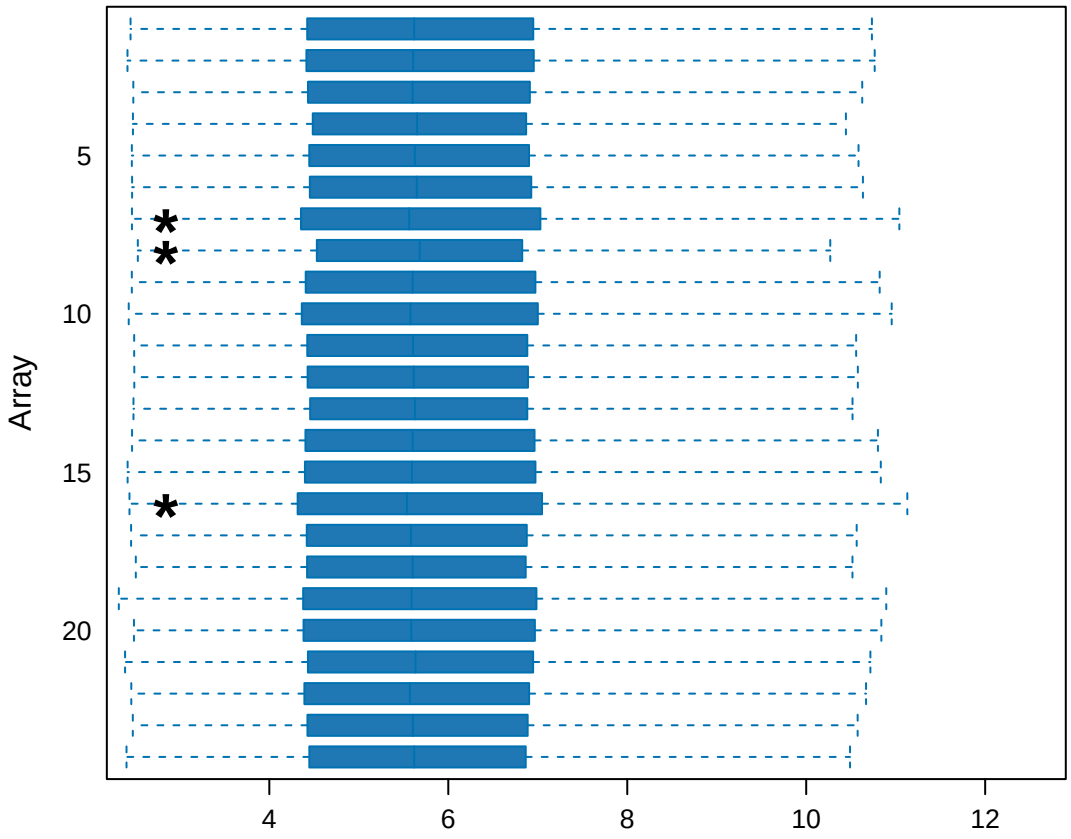


Figure 6: Density plots.

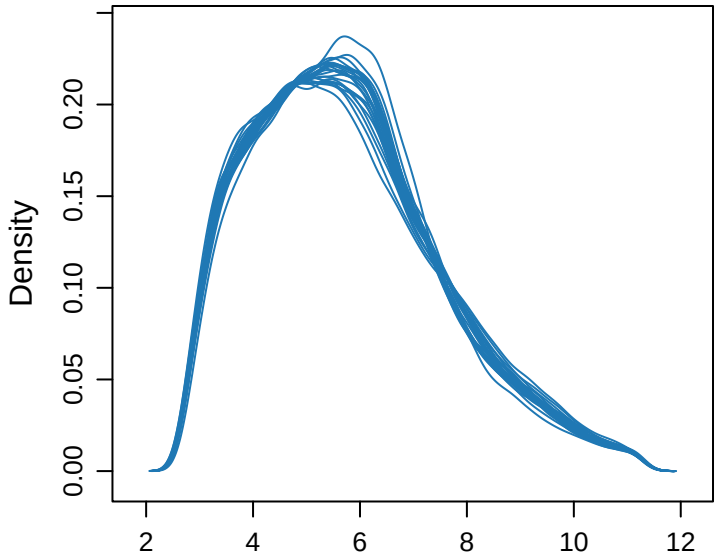
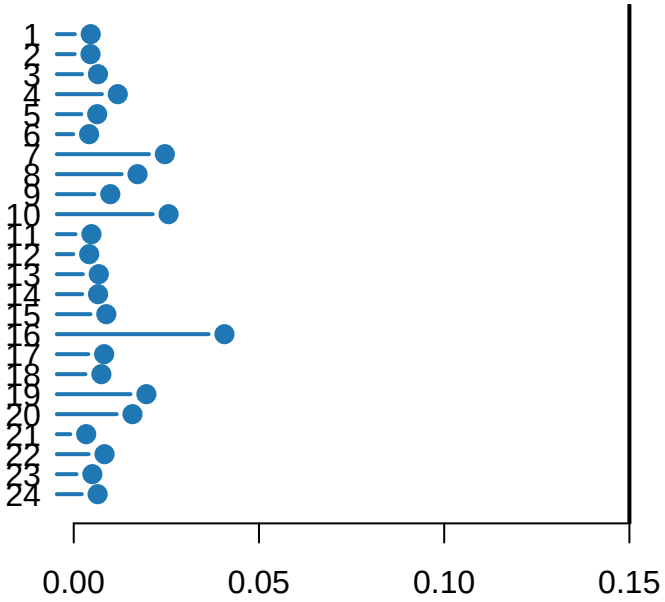


Figure 9: Outlier detection for MA plots



Apéndice 3

Estructura de la matriz de diseño

```
> print(design_mat)
```

```
Uninf.h0.linez Uninf.h0.vanco Inf.h24.untreat Inf.h24.linez
```

```
Uninf.h0.linez.1                1              0              0              0
Uninf.h0.vanco.2                0              1              0              0
Inf.h24.untreat.2               0              0              1              0
Inf.h24.linez.1                 0              0              0              1
Inf.h24.vanco.2                 0              0              0              0
Uninf.h0.untreat.4              0              0              0              0
Uninf.h0.vanco.4                0              1              0              0
Inf.h24.vanco.3                 0              0              0              0
Uninf.h0.untreat.3              0              0              0              0
Uninf.h0.linez.4                1              0              0              0
Inf.h24.untreat.1               0              0              1              0
Inf.h24.linez.2                 0              0              0              1
Inf.h24.vanco.4                 0              0              0              0
Uninf.h0.untreat.2              0              0              0              0
Uninf.h0.linez.2                1              0              0              0
Uninf.h0.vanco.1                0              1              0              0
Inf.h24.untreat.4               0              0              1              0
Inf.h24.linez.3                 0              0              0              1
Uninf.h0.untreat.1              0              0              0              0
Uninf.h0.linez.3                1              0              0              0
Uninf.h0.vanco.3                0              1              0              0
Inf.h24.untreat.3               0              0              1              0
Inf.h24.linez.4                 0              0              0              1
Inf.h24.vanco.1                 0              0              0              0
                  Inf.h24.vanco Uninf.h0.untreat
Uninf.h0.linez.1              0                0
Uninf.h0.vanco.2              0                0
Inf.h24.untreat.2             0                0
Inf.h24.linez.1               0                0
Inf.h24.vanco.2               1                0
Uninf.h0.untreat.4            0                1
Uninf.h0.vanco.4              0                0
Inf.h24.vanco.3               1                0
Uninf.h0.untreat.3            0                1
Uninf.h0.linez.4              0                0
Inf.h24.untreat.1             0                0
Inf.h24.linez.2               0                0
Inf.h24.vanco.4               1                0
Uninf.h0.untreat.2            0                1
Uninf.h0.linez.2              0                0
Uninf.h0.vanco.1              0                0
Inf.h24.untreat.4             0                0
Inf.h24.linez.3               0                0
Uninf.h0.untreat.1            0                1
Uninf.h0.linez.3              0                0
Uninf.h0.vanco.3              0                0
Inf.h24.untreat.3             0                0
Inf.h24.linez.4               0                0
Inf.h24.vanco.1               1                0
attr(,"assign")
[1] 1 1 1 1 1 1
attr(,"contrasts")
attr(,"contrasts")$group_col
[1] "contr.treatment"
```

Estructura de la matriz de contraste

```
> print(cont.matrix)
```

```
                 Contrasts
Levels            Inf.h24.untreat.vs.Uninf.h0.untreat
  Uninf.h0.linez                                    0
  Uninf.h0.vanco                                    0
  Inf.h24.untreat                                   1
  Inf.h24.linez                                     0
  Inf.h24.vanco                                     0
  Uninf.h0.untreat                                 -1
                 Contrasts
Levels            Inf.h24.linez.vs.Uninf.h0.linez
  Uninf.h0.linez                                 -1
  Uninf.h0.vanco                                  0
  Inf.h24.untreat                                 0
  Inf.h24.linez                                   1
  Inf.h24.vanco                                   0
  Uninf.h0.untreat                                0
                 Contrasts
Levels            Inf.h24.vanco.vs.Uninf.h0.vanco
  Uninf.h0.linez                                  0
  Uninf.h0.vanco                                 -1
  Inf.h24.untreat                                 0
  Inf.h24.linez                                   0
  Inf.h24.vanco                                   1
  Uninf.h0.untreat                                0
```

Clase del objeto `fit.main`

```
> class(fit.main)
```

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

Apéndice 3

Anotaciones dispobibles en el paquete `mouse4302.db`

```
> keytypes(mouse4302.db)
```

```
 [1] "ACCNUM"       "ALIAS"        "ENSEMBL"      "ENSEMBLPROT"  "ENSEMBLTRANS"
 [6] "ENTREZID"     "ENZYME"       "EVIDENCE"     "EVIDENCEALL"  "GENENAME"
[11] "GENETYPE"     "GO"           "GOALL"        "IPI"          "MGI"
[16] "ONTOLOGY"     "ONTOLOGYALL"  "PATH"         "PFAM"         "PMID"
[21] "PROBEID"      "PROSITE"      "REFSEQ"       "SYMBOL"       "UNIPROT"
```

Primeras 6 filas del listado de genes del `ExpressionSet`

```
> head(annotated_data)
```

```
       PROBEID                              GENENAME        SYMBOL
1   1436530_at   WAP four-disulfide core domain 17        Wfdc17
2   1457728_at         niban apoptosis regulator 3        Niban3
3   1446929_at          RIKEN cDNA D130062J21 gene D130062J21Rik
4 1426113_x_at T cell receptor alpha variable 9D-3      Trav9d-3
5   1436040_at    small nucleolar RNA host gene 12        Snhg12
6   1440451_at            ankyrin repeat domain 66       Ankrd66
             ENSEMBL  ENTREZID
1 ENSMUSG00000069792 100034251
2 ENSMUSG00000043243 100037278
3               <NA> 100038651
4 ENSMUSG00000095495 100038850
5 ENSMUSG00000086290 100039864
6 ENSMUSG00000096140 100043332
```

9. ENLACE A REPOSITORIO DE GitHub

Enlace a Repositorio de GitHub con el proyecto subido a distintos formatos (PDF, HTML y Rmarkdown).

https://github.com/jsanchorom/SanchoRomero_JuanMa_ADO_PEC2.git