Expression data in Staphylococcus aureus-infected mice with linezolid and vancomycin treatment.

Silvia Arroitajauregui Avilés

2024-12-16

I. Tabla de contenidos.

II. Introducción y Objetivos.

Debido a que los efectos de los antibióticos en la expresión génica del huésped son desconocidos, el presente estudio trata de investigar la utilidad de los antibióticos LINEZOLID y VANCOMICINA para inmunomodulación durante infecciones por *Staphylococcus aureus* resistente a meticilina (MRSA) en un modelo murino, en este caso de ratón (*Mus musculus)*.

Para ello, se trabajará sobre las muestras obtenidas en el estudio *Expression data in Staphylococcus aureus-infected mice with linezolid and vancomycin treatment* (Prakash et al., 2012).

Así pues, el objetivo principal es intentar caracterizar, a través de cambios en la expresión génica, el efecto de la infección y del tratamiento con antibióticos, así como comparar los efectos de estos.

Con este fin, se realizarán las siguientes comparaciones:

- Infectados vs no infectados sin tratamiento,
- infectados vs no infectados tratados con LINEZOLID,
- infectados vs no infectados tratados con VANCOMICINA.

III. Métodos.

0. Materiales.

Se utilizan las muestras de la serie GSE38531 de Gene Expression Omnibus, conformado por 35 muestras, divididas en distintos grupos:

- Estado de la infección:
    - 15 muestras previas a la infección.
    - 20 muestras infectadas.
- Tratamiento:
    - 15 muestras sin tratar.
    - 10 muestras tratadas con LINEZOLID.
    - 10 muestras tratadas con VANCOMYCIN.
- Hora cuando se realiza la muestra:
    - Hora 0: 15 muestras.
    - Hora 2: 5 muestras.
    - Hora 24: 15 muestras.

El tratamiento, así como el análisis de la expresión génica y las pruebas estadísticas se realizaron utilizando RStudio (versión 2024.12.0) con R (versión 4.4.2).

Así mismo, las bibliotecas utilizadas son:

- Bioconductor,
- affy,
- dplyr,
- Biobase,
- ggplot2,
- ggrepel,
- arrayQualityMetrics,
- limma,

- annotate,
- mouse4302.db,
- ReactomePA

1. Preparación de las muestras.

Se realiza una lectura de los metadatos de las muestras descargadas y un filtrado previo, descartando las que tienen un valor en time de “hour 2” y seleccionando muestras aleatorias, descartando un total de 11.

Tras el filtrado de las muestras, se cargan los archivos .CEL para las 24 muestras seleccionadas, los cuales contienen los datos de la expresión génica. Se almacenan los datos en crudo en un objeto ExpressionSet, el cual incluye las intensidades de expresión (assayData), los metadatos de las muestras (phenoData) y datos adicionales relacionados con las sondas (featureData).

2. Análisis exploratorio y control de calidad.

Se revisa el objeto ExpressionSet, incluyendo las dimensiones y el contenido, así como las filas y columnas de las intensidades de expresión, los metadatos y los datos de las sondas para confirmar que la generación de este se ha llevado a cabo correctamente.

Así mismo, se lleva un análisis de calidad a través de arrayQualityMetrics, el cual ayuda a detectar datos inusuales o muestras con calidad baja. A partir de la figura generada *infex.html*, obtenemos:

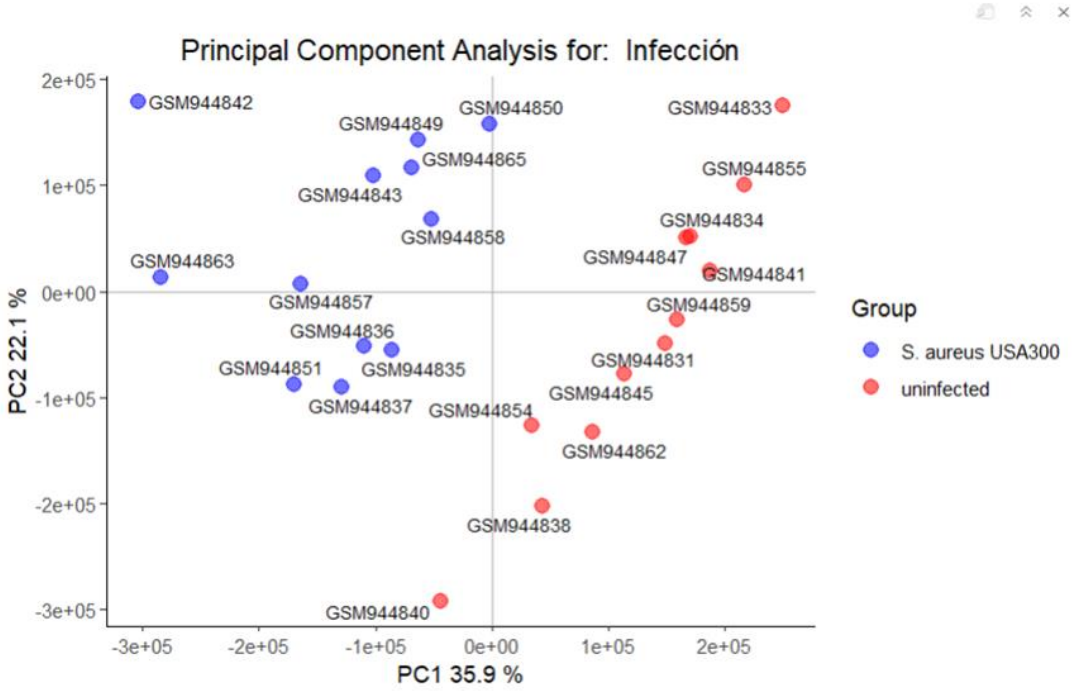
	array	sampleNames	*1	*2	*3	sample	infection	time	agent
☐	1	GSM944836				GSM944836	S. aureus USA300	hour 24	linezolid
☐	2	GSM944850				GSM944850	S. aureus USA300	hour 24	linezolid
☐	3	GSM944857				GSM944857	S. aureus USA300	hour 24	linezolid
☐	4	GSM944843				GSM944843	S. aureus USA300	hour 24	linezolid
☐	5	GSM944854				GSM944854	uninfected	hour 0	linezolid
☐	6	GSM944847		x		GSM944847	uninfected	hour 0	linezolid
☐	7	GSM944833				GSM944833	uninfected	hour 0	linezolid
☐	8	GSM944840				GSM944840	uninfected	hour 0	linezolid
☐	9	GSM944835				GSM944835	S. aureus USA300	hour 24	untreated
☐	10	GSM944849				GSM944849	S. aureus USA300	hour 24	untreated
☐	11	GSM944863		x		GSM944863	S. aureus USA300	hour 24	untreated
☐	12	GSM944842	x			GSM944842	S. aureus USA300	hour 24	untreated
☐	13	GSM944845			x	GSM944845	uninfected	hour 0	untreated
☐	14	GSM944838				GSM944838	uninfected	hour 0	untreated
☐	15	GSM944859				GSM944859	uninfected	hour 0	untreated
☐	16	GSM944831				GSM944831	uninfected	hour 0	untreated
☐	17	GSM944865				GSM944865	S. aureus USA300	hour 24	vancomycin
☐	18	GSM944837				GSM944837	S. aureus USA300	hour 24	vancomycin
☐	19	GSM944858				GSM944858	S. aureus USA300	hour 24	vancomycin
☐	20	GSM944851				GSM944851	S. aureus USA300	hour 24	vancomycin
☐	21	GSM944862				GSM944862	uninfected	hour 0	vancomycin
☐	22	GSM944855				GSM944855	uninfected	hour 0	vancomycin
☐	23	GSM944841				GSM944841	uninfected	hour 0	vancomycin
☐	24	GSM944834				GSM944834	uninfected	hour 0	vancomycin

En esta tabla, se observa que solo hay una marca en 4 muestras, lo que indica que los problemas potenciales son pequeños, por lo que se pueden mantener todos los arrays del análisis.

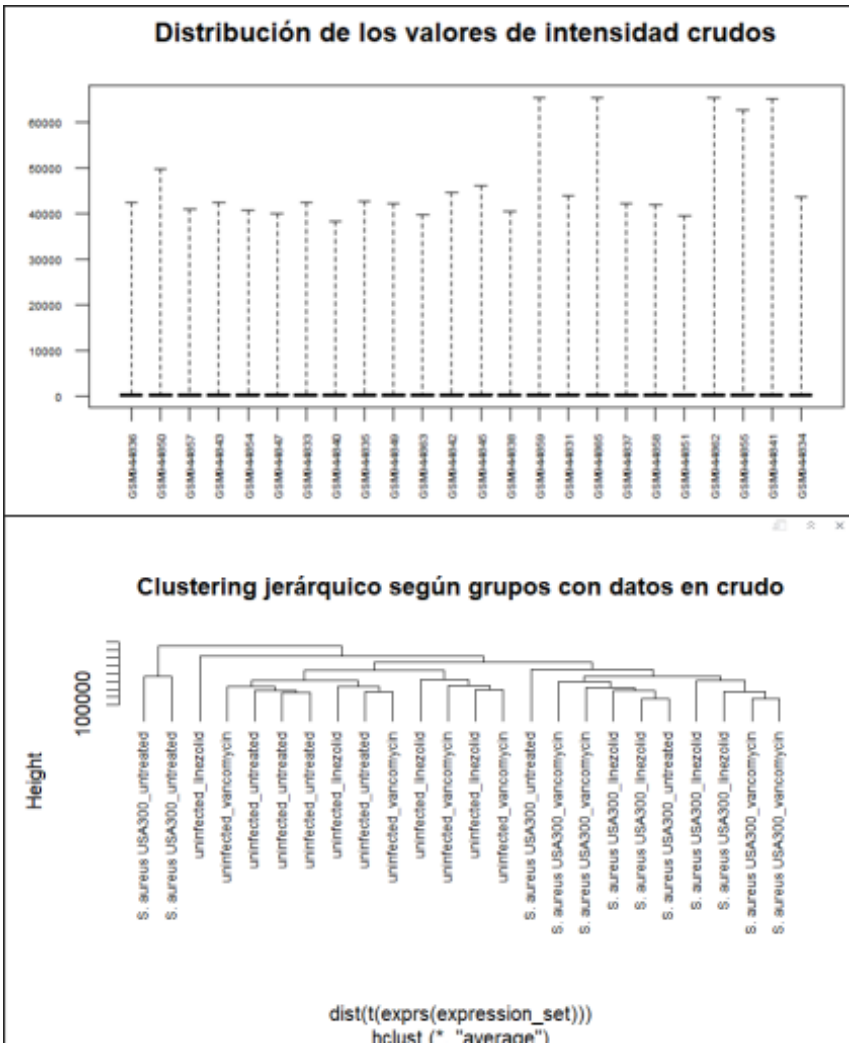
Para visualizar la variabilidad total de las muestras, se realiza un análisis de componentes principales (PCA) sobre el tratamiento utilizado, el estado de la infección y una combinación de ambos.

En este, se observa que el primer componente del PCA representa el 35.9% de la variabilidad total de las muestras, seguido por el segundo componente, el cual explica el 22.1% de esta.

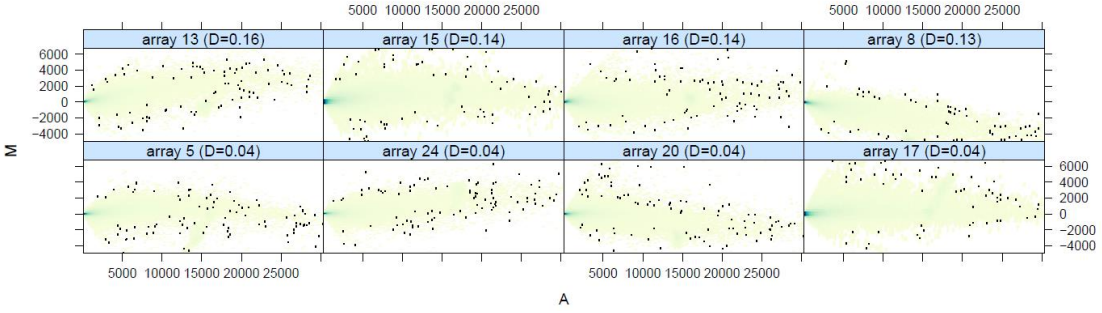
Así mismo, esta variabilidad puede estar aportada principalmente por la condiciñon del grupo, ya que las muestras sin infección tienden a la parte derecha, mientras que las muestras infectadas se encuentran en la parte derecha.



Para visualizar la distribución de las intensidades de expresión, se utiliza un boxplot, y para evaluar como se agrupan las muestras, se genera un análisis de clustering jerárquico. En estas, se observa que es necesario normalizar las muestras para corregir las variaciones entre muestras.



Así mismo, se obtiene el siguiente gráfico MA en el archivo arrangeQualityMetrics:

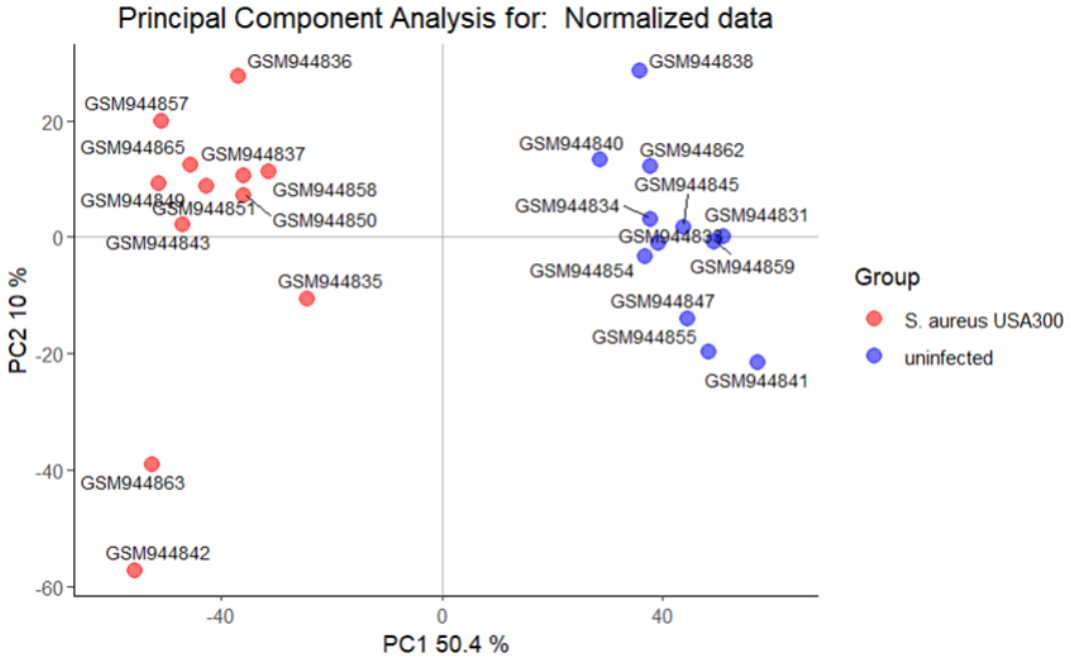


En este se observa una gran variabilidad, por lo que se realiza una normalización con RMA.

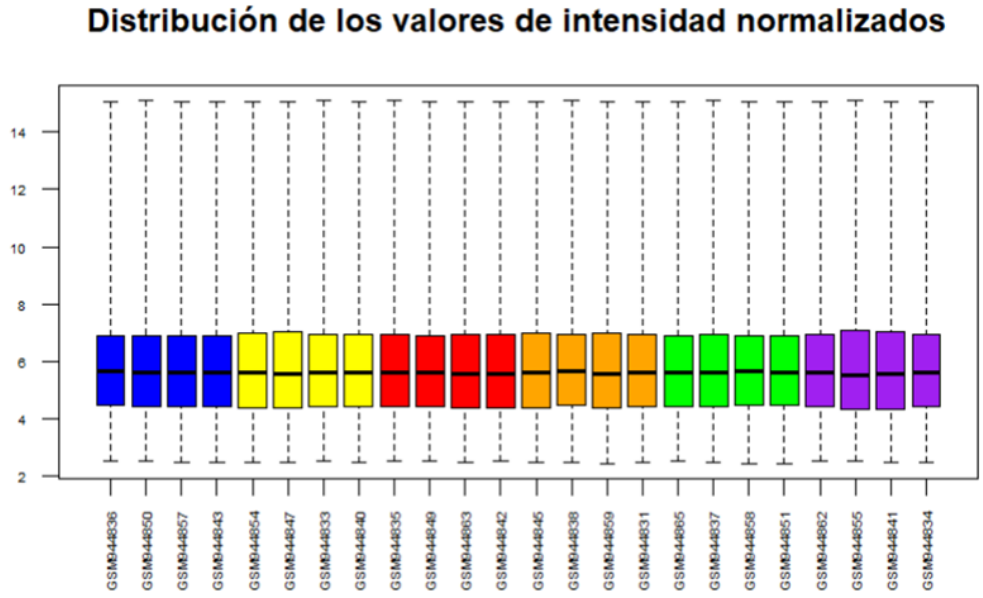
Tras la normalización, estos datos mejoran significativamente, como se puede contrastar con la siguiente tabla y gráficos:

	array	sampleNames	*1	*2	*3	sample	ScanDate
☐	1	GSM944836.CEL				1	2010-11-10T17:00:43Z
☐	2	GSM944850.CEL				2	2010-11-10T19:22:48Z
☐	3	GSM944857.CEL				3	2010-11-10T20:35:13Z
☐	4	GSM944843.CEL				4	2010-11-10T18:17:56Z
☐	5	GSM944854.CEL				5	2010-11-10T20:00:00Z
☐	6	GSM944847.CEL				6	2010-11-10T18:55:14Z
☐	7	GSM944833.CEL				7	2010-11-10T16:32:50Z
☐	8	GSM944840.CEL				8	2010-11-10T17:50:17Z
☐	9	GSM944835.CEL				9	2010-11-10T16:51:22Z
☐	10	GSM944849.CEL				10	2010-11-10T19:13:34Z
☐	11	GSM944863.CEL				11	2010-11-10T21:30:54Z
☐	12	GSM944842.CEL	x			12	2010-11-10T18:08:35Z
☐	13	GSM944845.CEL				13	2010-11-10T18:36:29Z
☐	14	GSM944838.CEL				14	2010-11-10T17:19:25Z
☐	15	GSM944859.CEL				15	2010-11-10T20:53:55Z
☐	16	GSM944831.CEL				16	2010-11-10T16:14:28Z
☐	17	GSM944865.CEL				17	2010-11-10T21:49:42Z
☐	18	GSM944837.CEL				18	2010-11-10T17:10:05Z
☐	19	GSM944858.CEL				19	2010-11-10T20:44:31Z
☐	20	GSM944851.CEL				20	2010-11-10T19:32:16Z
☐	21	GSM944862.CEL				21	2010-11-10T21:21:37Z
☐	22	GSM944855.CEL		x		22	2010-11-10T20:15:24Z
☐	23	GSM944841.CEL				23	2010-11-10T17:59:38Z
☐	24	GSM944834.CEL				24	2010-11-10T16:42:09Z

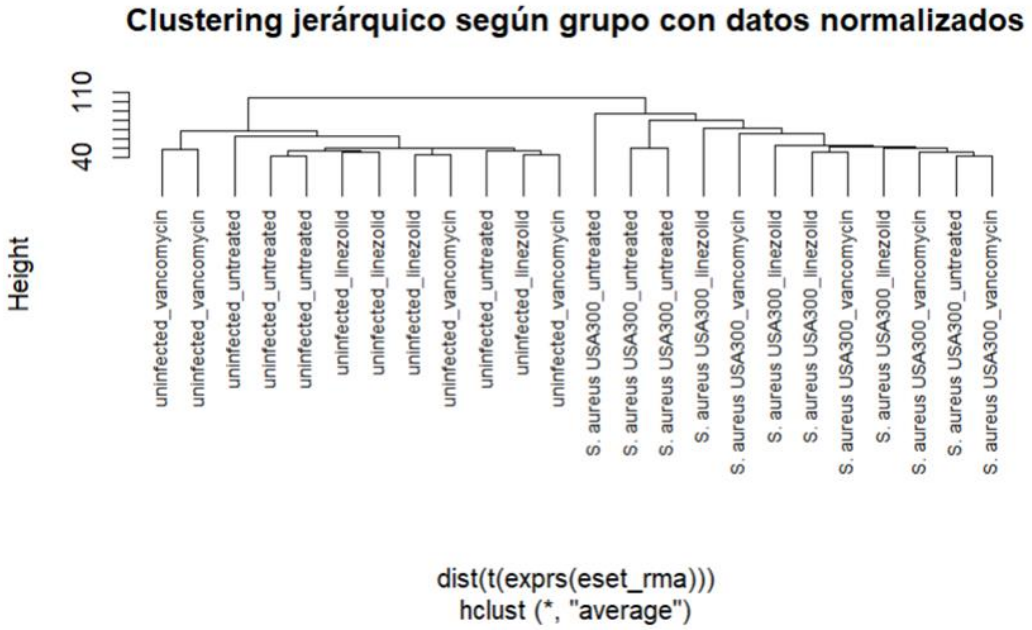
Se disminuyen las marcas en solo 2 muestras.



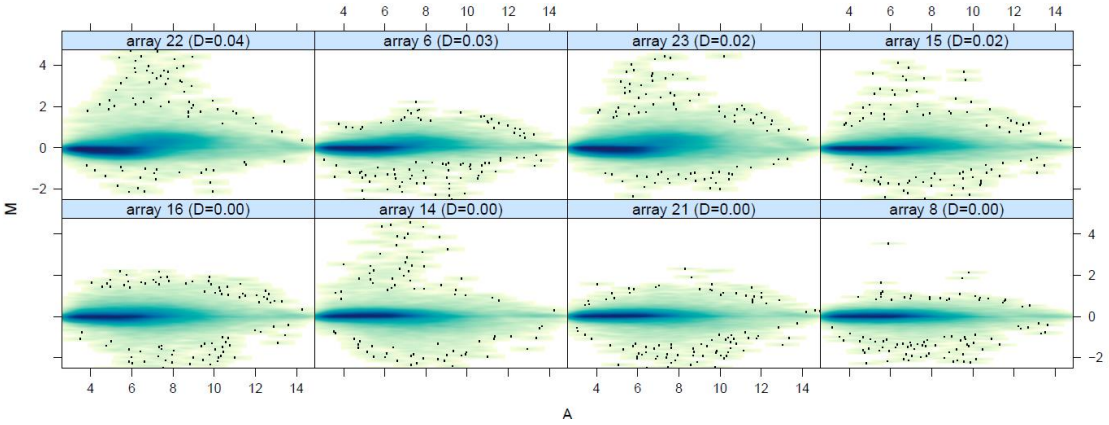
El primer componente del PCA representa, ahora, un 50.4% de la variabilidad total de las muestras, y el segundo un 10%, representando de manera más eficiente la mayor parte de la variabilidad, reduciendo la dimensionalidad.



La mejora en la simetría entre las muestras es notable.

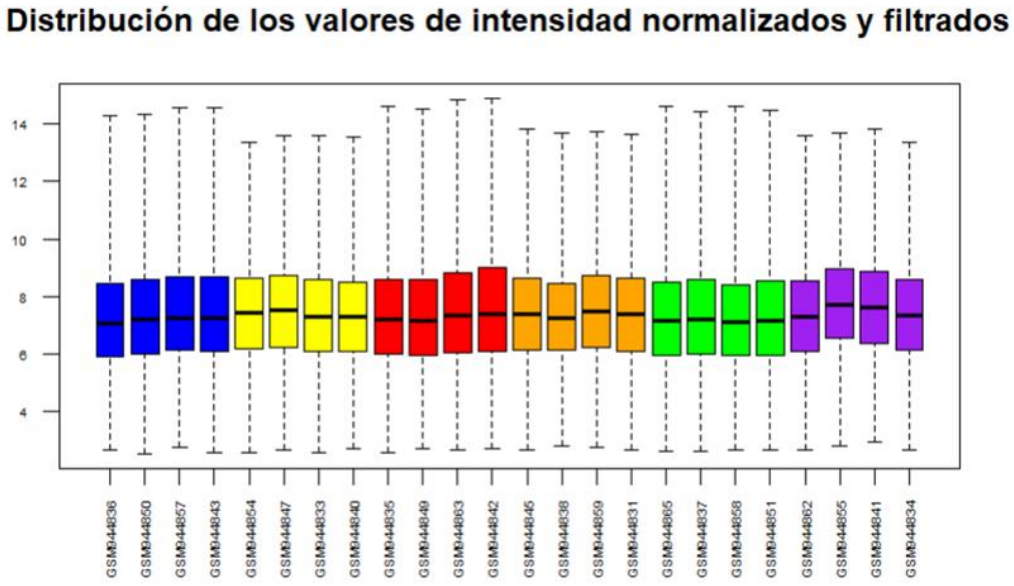


Así mismo, se observa una mejora en la similitud de las muestras, mostrando posibles patrones de expresión similares, así como en el MA del archivo arrangeQualityMetrics:



3. Filtrado de datos.

Con tal de reducir el ruido y destacar los genes más relevantes, se realiza un filtrado. Para ello, se calcula la desviación estándar de cada sonda y se seleccionan las de mayor variabilidad (el 10% superior). Tras ello, generamos el ExpressionSet, utilizándolo para realizar un análisis de los componentes principales (PCA), un boxplot y un dendograma para comparar con los resultados sin filtrar.



Tras realizar la comparación, se observa que la calidad en las muestras empeora, por lo que se concluye que se continuará el análisis con los datos normalizados pero sin filtrar.

4. Construcción de las matrices de diseño y de contraste.

Con tal de moderar las condiciones experimentales, se genera una matriz de diseño, en la cual se refleja el tratamiento y el estado de la infección por cada muestra.

Tras ello, se define la matriz de contraste, para comparar entre:

- Infectados y no infectados sin tratamiento,
- infectados y no infectados tratados con LINEZOLID,
- infectados y no infectados tratados con VANCOMYCIN.

5. Obtención de genes diferencialmente expresados.

Para revisar los genes cuya expresión varía entre las condiciones experimentales, realizamos un ajuste de modelo lineal a partir de la función lmFit del paquete limma, la cuál estima la expresión diferencial de cada gen entre los diferentes grupos. Tras ello, se aplica el ajuste bayesiano con eBayes para mejorar estas estimaciones y la fiabilidad de los resultados.

Así mismo, se genera una tabla de resultados por cada comparación entre los grupos a partir de topTable, ordenador por el p-value ajustado.

6. Anotación de los genes.

Para poder tener información biológica adicional sobre los genes identificados, se debe transformar el número de sonda (Probe ID) en los nombres de genes, su símbolo y su EntrezID.

A partir de la base de datos mouse4302.db se añade esta información a la tablas de los resultados.

7. Expresión diferencial.

A través de gráficos volcano plot se visualizan los genes que tienen una expresión significativamente diferente entre las condiciones experimentales. De este modo, se realizan estos gráficos para el estado de la infección entre las muestras sin tratamiento, entre las muestras tratadas con LINEZOLID y las muestras tratadas con VANCOMYCIN.

8. Comparaciones múltiples.

Con tal de revisar los genes comunes entre las diferentes condiciones experimentales, se realiza la prueba de hipótesis para cada contraste y se genera, a partir de ella, un diagrama de Venn, con el cual se visualizan los genes clave que responden a los diferentes tratamientos.

9. Análisis de la significación biológica.

Por último, se utilizan los resultados de la expresión diferencial para identificar los genes relevantes para cada condición experimental. Para ello, se extraen los genes diferencialmente expresados (p-value < 0.05).

Por otro lado, se realiza un análisis de enriquecimiento de Gene Ontology (GO), el cual clasifica los genes en procesos biológicos, funciones moleculares y componentes celurares. También se realiza un análisis del enriquecimiento de rutas metabólicas y de señalización a través de bases de datos como KEGG.

Finalmente, estos resultados se visualizan en un gráfico de red de las vías metabólicas.

IV. Resultados.

Se muestran las primeras líneas de las tablas annotatedTOP:

1) Para muestras sin tratamiento:

	PROBEID <chr>	SYMBOL <chr>	ENTRE... <chr>	GENENAME <chr>
1	1415670_at	Copg1	54161	coatomer protein complex, subunit gamma 1
2	1415671_at	Atp6v0...	11972	ATPase, H+ transporting, lysosomal V0 subunit D1
3	1415672_at	Golga7	57437	golgin A7
4	1415673_at	Psph	100678	phosphoserine phosphatase
5	1415674_a_at	Trappc4	60409	trafficking protein particle complex 4
6	1415675_at	Dpm2	13481	dolichyl-phosphate mannosyltransferase subunit 2, regulatory

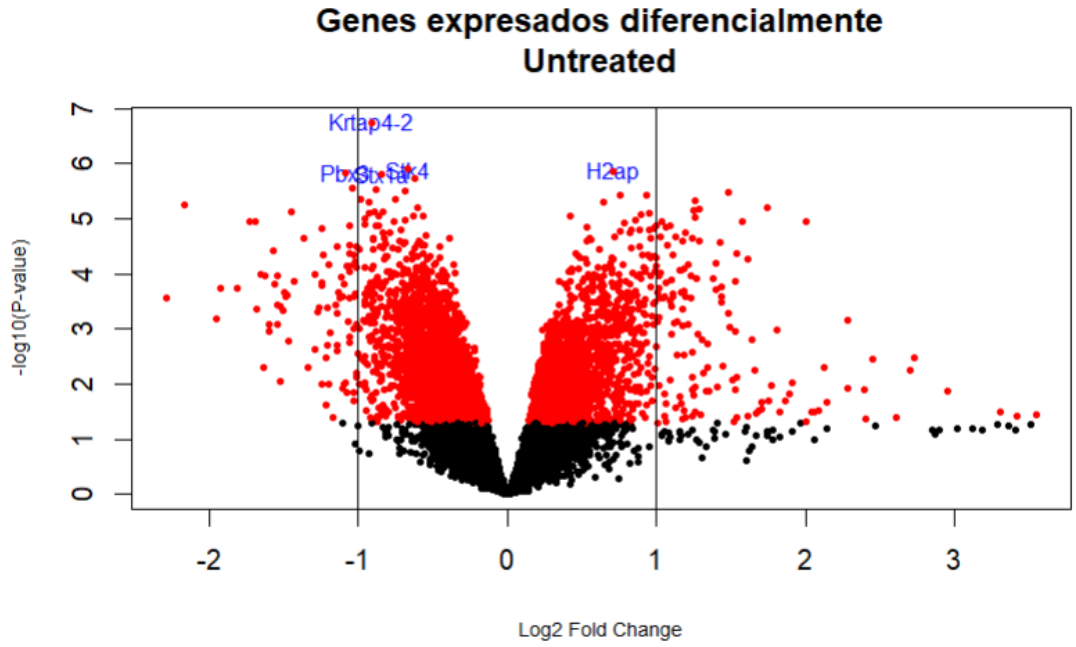
2) Para muestras tratadas con LINEZOLID:

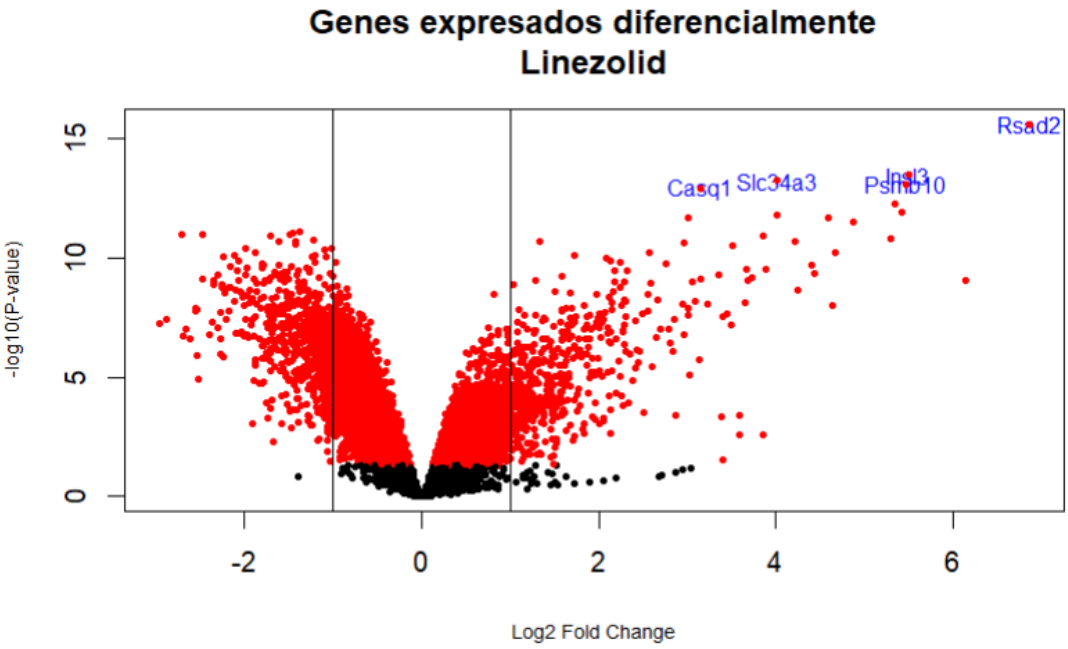
	PROBEID <chr>	SYMBOL <chr>	ENTRE... <chr>	GENENAME <chr>
1	1415670_at	Copg1	54161	coatomer protein complex, subunit gamma 1
2	1415671_at	Atp6v0...	11972	ATPase, H+ transporting, lysosomal V0 subunit D1
3	1415672_at	Golga7	57437	golgin A7
4	1415673_at	Psph	100678	phosphoserine phosphatase
5	1415674_a_at	Trappc4	60409	trafficking protein particle complex 4
6	1415675_at	Dpm2	13481	dolichyl-phosphate mannosyltransferase subunit 2, regulatory

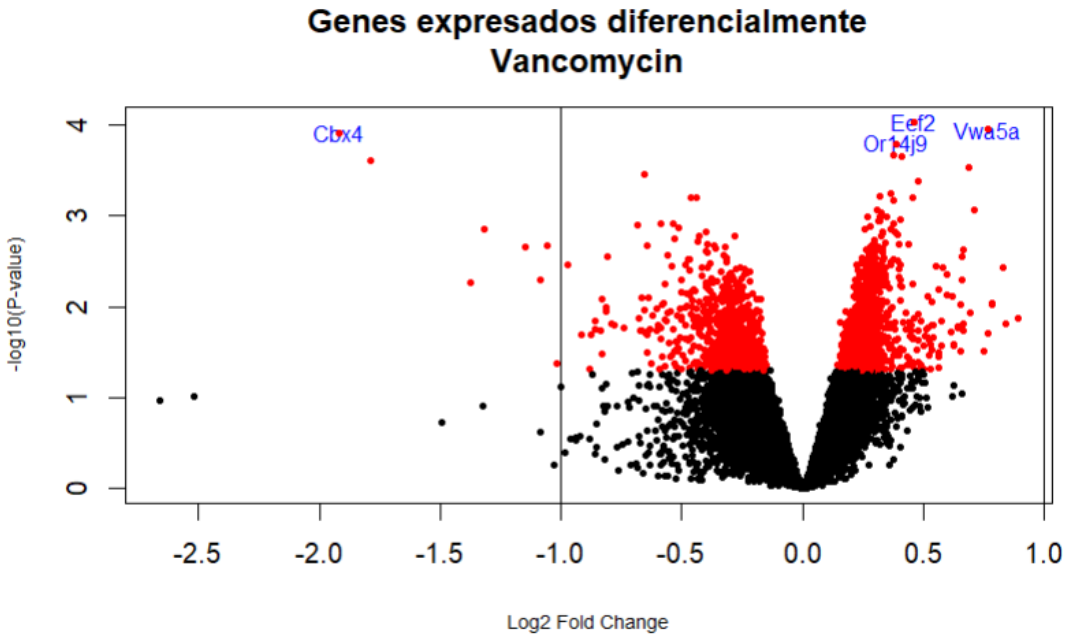
3) Para muestras tratadas con VANCOMYCIN:

	PROBEID <chr>	SYMBOL <chr>	ENTRE... <chr>	GENENAME <chr>
1	1415670_at	Copg1	54161	coatomer protein complex, subunit gamma 1
2	1415671_at	Atp6v0...	11972	ATPase, H+ transporting, lysosomal V0 subunit D1
3	1415672_at	Golga7	57437	golgin A7
4	1415673_at	Psph	100678	phosphoserine phosphatase
5	1415674_a_at	Trappc4	60409	trafficking protein particle complex 4
6	1415675_at	Dpm2	13481	dolichyl-phosphate mannosyltransferase subunit 2, regulatory

Los volcano plot ayudan a visualizar los genes con un gran cambio en la expresión entre las condiciones, siendo estadísticamente significativos:


Genes expresados diferencialmente
Untreated
Krtap4-2
H2ap
-log10(P-value)
Log2 Fold Change


Genes expresados diferencialmente
Linezolid
Rsad2
Casq1
Slc34a3
Psmb10
-log10(P-value)
Log2 Fold Change

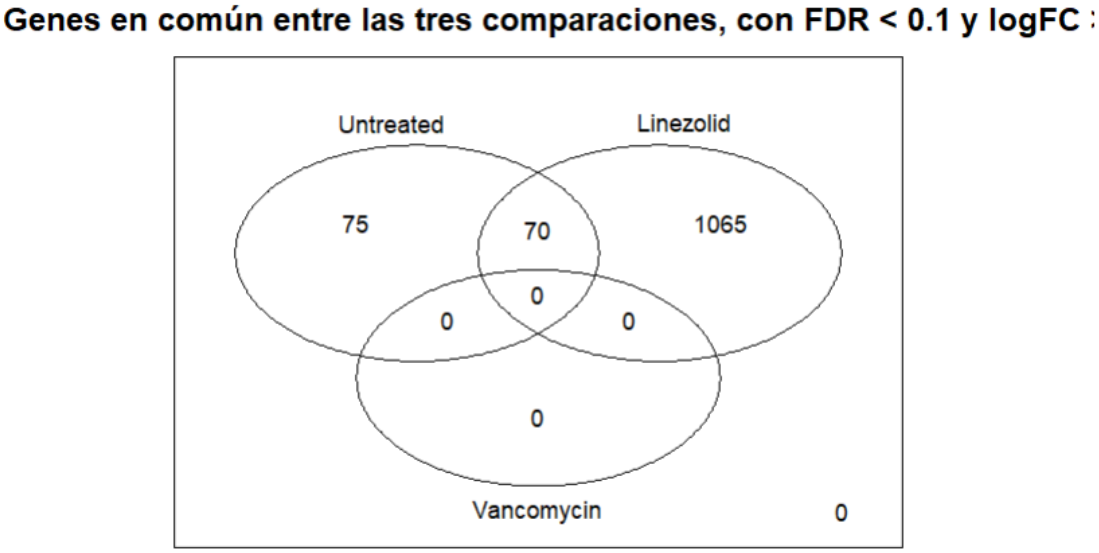


Se marcan en rojo los genes con un p-value < 0.05 (estadísticamente significativos) y se etiquetan los 4 genes más diferenciados de cada condición.

Referente a las comparaciones múltiples, se obtiene:

	Untreated	Linezolid	Vancomycin
Down	71	655	0
NotSig	44956	43966	45101
Up	74	480	0

Y al ilustrarlo con el diagrama de Venn:

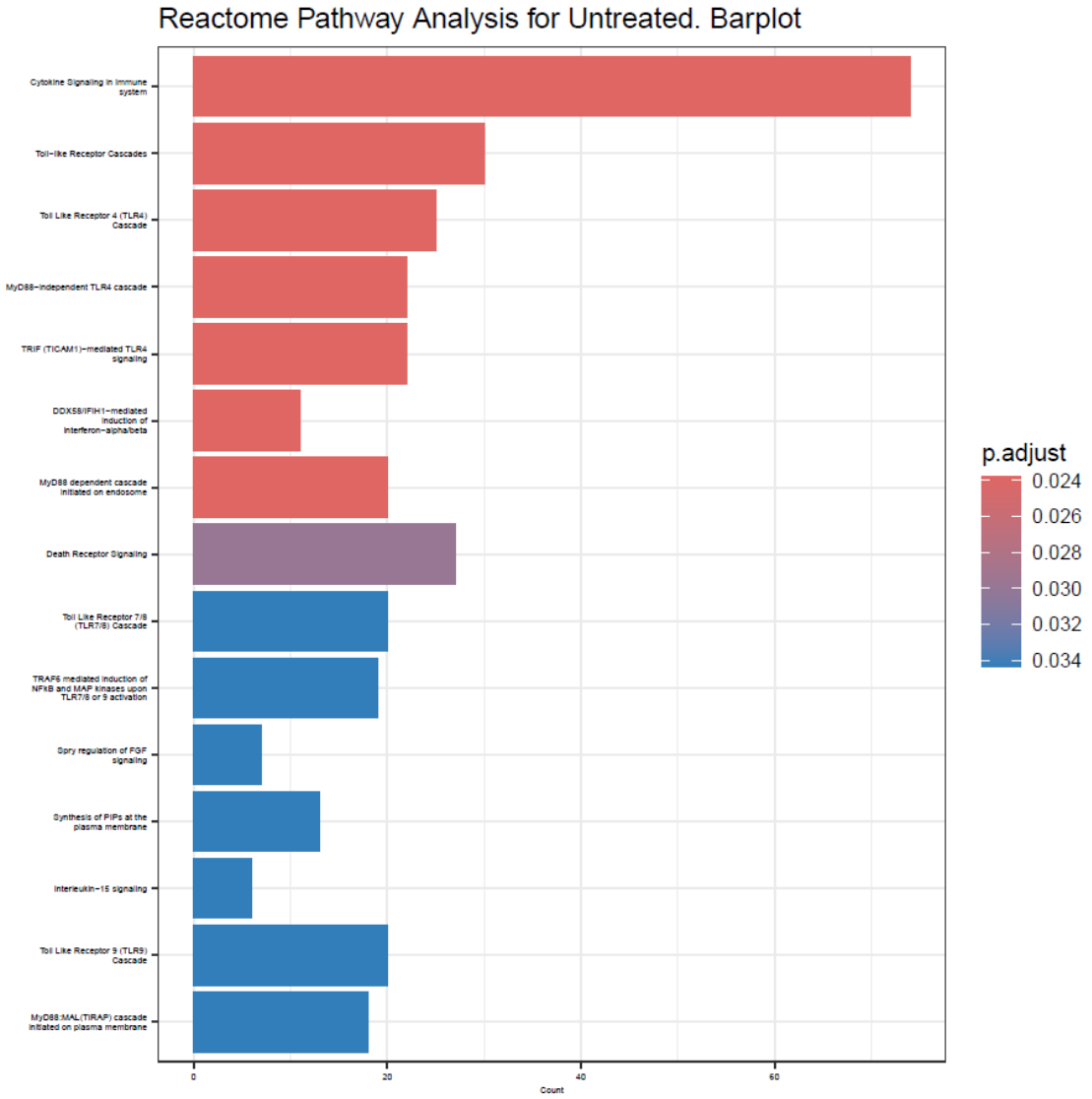


Finalmente, en cuanto el análisis de la significación biológica, se obtiene la siguiente cantidad de genes con diferencia significativa por cada grupo de tratamiento:

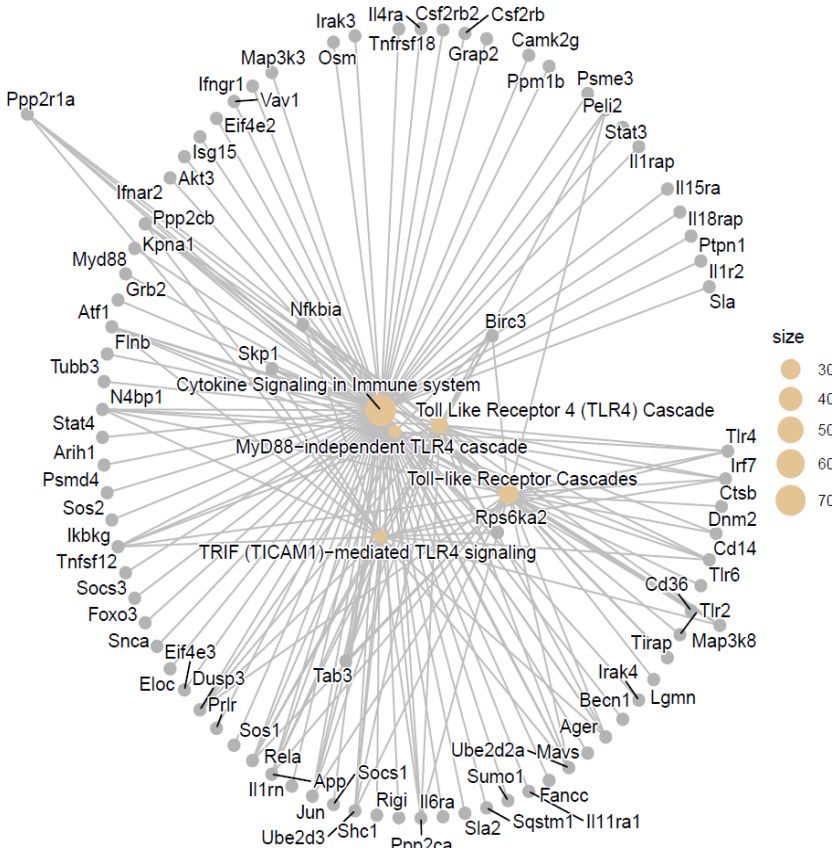
Untreated	Linezolid	Vancomycin
2186	13647	0

Debido a que VANCOMYCIN no tiene un efecto significativo sobre los genes, vamos a mostrar los resultados sobre Untreated y LINEZOLID.

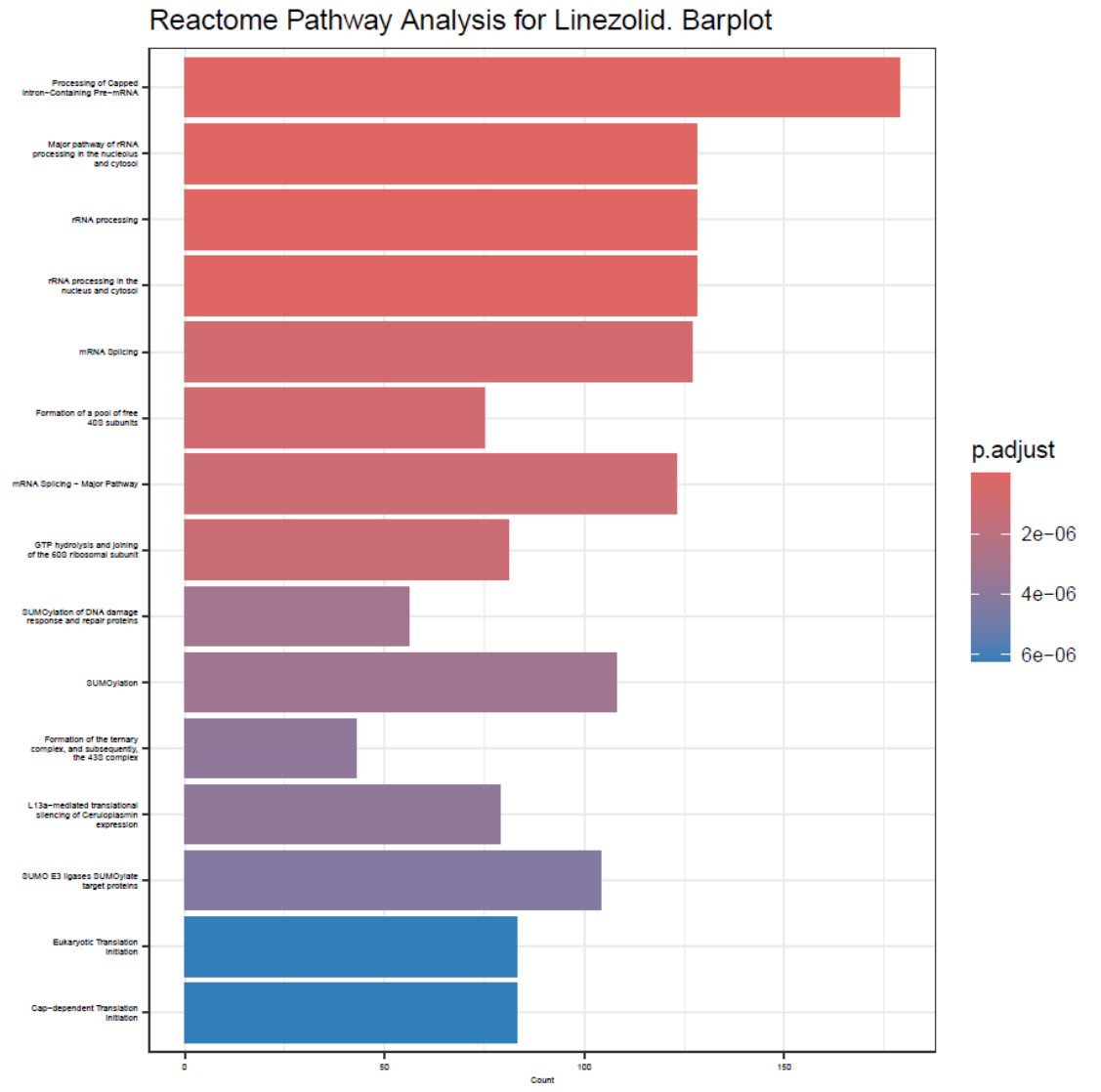
Para Untreated, estas son las rutas biológicas con más cambios significativos:



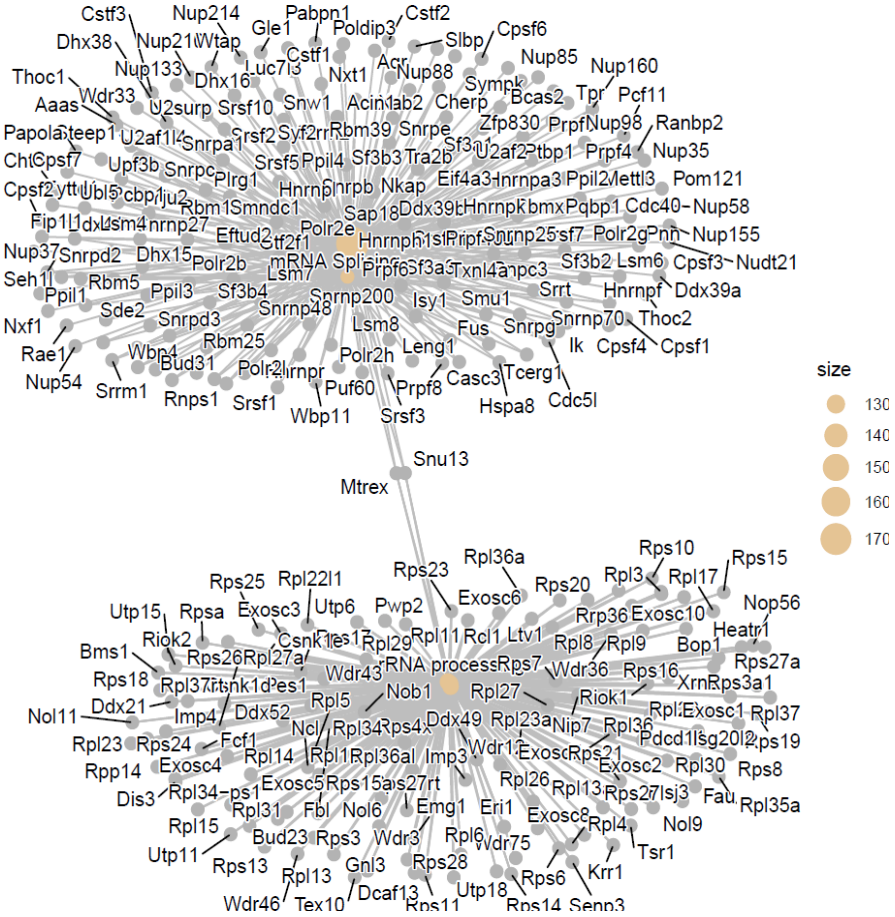
Esto puede ilustrarse también con su correspondiente cnetplot:



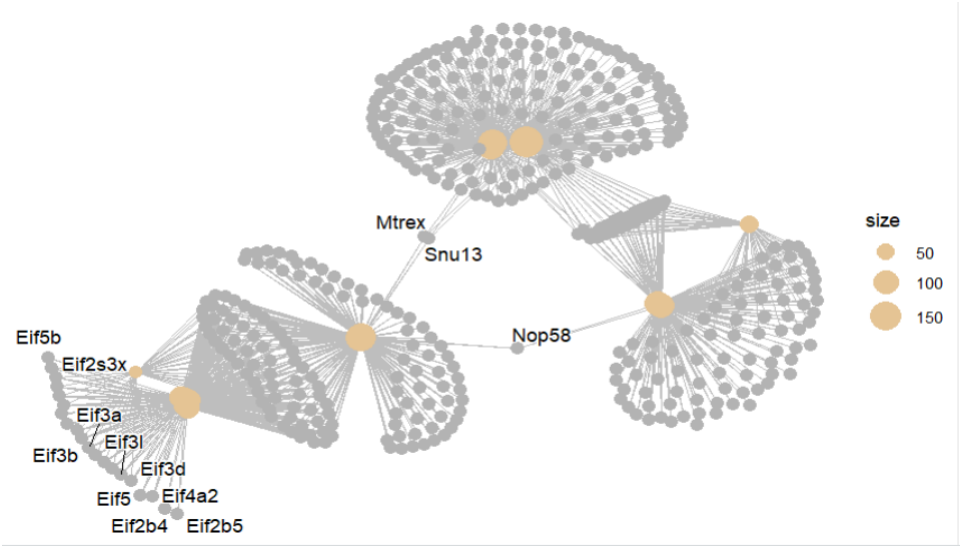
Mientras que para LINEZOLID, se obtienen las siguientes rutas:



En este caso, el cnetplot obtenido no es de gran utilidad, debido a la superposición de los genes, dificultando la visualización:



Finalmente, en cuanto la visualización de los genes y vías metabólicas en común entre Untreated y LINEZOLID:



V. Discusión.

Análisis de cada tratamiento.

Grupo sin tratamiento.

Se encuentran 71 genes con una expresión significativamente disminuida (Down) y 74 con la expresión significativamente aumentada (Up) en comparación con las muestras no infectadas.

Los genes diferencialmente expresados en el grupo infectado sin tratamiento (siendo un total de 2186, siendo que la mayoría no muestra cambios significativos) están relacionados con vías asociadas a la respuesta inmunitaria, destacando las siguientes vías:

- Cascadas de señalización de receptores tipo Toll (TLR), relacionadas con la detección de estructuras microbianas conservadas para identificar agentes infecciosos (Kumar et al., 2022).
- Vías relacionadas con citoquinas proinflamatorias, destacando la participación de IL-1 y TNF, relacionadas con la inflamación aguda (Thermo Fisher Scientific, n.d.).

Así pues, esto parece indicar que, al no haber tratamiento, hay una respuesta inflamatoria como consecuencia de la infección.

Tratamiento con LINEZOLID.

Se detectan 655 genes con una expresión significativa menor (Down) y 480 genes con expresión significativamente mayor (Up) en comparación con el grupo sin infección.

Se identifican un total de 13647 genes relacionados con las siguientes vías biológicas:

- Procesamiento de pre-mRNA y splicing, mostrando una regulación de genes como Srsf1 y Prpf4b, involucrados en la maquinaria de empalme (Zhang & Liu, 2020).
- Rutas implicadas en la regeneración y reparación celular, con genes como Atf4 y Gadd45b mostrando un incremento en su expresión. Esto puede mitigar el daño celular provocado por la infección (Wang et al., 2022).

Así pues, parece que LINEZOLID reduce la respuesta inflamatoria, y tiene una respuesta marcada en las células del paciente.

Tratamiento con VANCOMICINA.

En este caso, no se encuentran cambios significativos en la expresión génica, ni hacia arriba (Up) ni hacia abajo (Down). Esto puede ser debido a que el antibiótico solo actúa sobre la población bacteriana, pero sin generar una modulación inmunitaria en el paciente (Nelson et al., 2017).

Análisis de vías comunes.

Se han detectado 70 genes en común entre las muestras sin tratamiento y las muestras tratadas con LINELOZID. Esto puede ser debido a que se activan las mismas rutas biológicas en las respuestas inmunes iniciales debido a la infección.

Así mismo, se vuelve a destacar que VANCOMICINA no tiene genes relevantes en común con los otros grupos, lo que parece reforzar que no genera una respuesta inmunitaria.

VI. Referencias.

ASP Teaching. (n.d.). Case study 1: Microarrays analysis. Retrieved December 17, 2024, from https://aspteaching.github.io/Omics_Data_Analysis-Case_Study_1-Microarrays/Case_Study_1-Microarrays_Analysis.html#gene-annotation

ASP Teaching. (n.d.). Diseño experimental en el análisis de datos ómicos. Retrieved December 17, 2024, from https://aspteaching.github.io/Analisis_de_datos_omicos-Materiales_para_un_curso/MDAExpDesign.html

ASP Teaching. (n.d.). Ejemplo PCA: detección del efecto batch en datos de microarrays. Retrieved December 17, 2024, from https://aspteaching.github.io/AMVCasos/#ejemplo-pca-2-detecci%C3%B3n-del-efecto-batch-en-datos-de-microarrays

Kumar, S., Kumar, A., Bhattacharya, A., Roy, R., Patel, S., Singh, S., ... & Kumar, A. (2022). Transcriptomics of host responses in *Staphylococcus aureus*-infected mice treated with linezolid and vancomycin. *Frontiers in Microbiology, 13*, 935877. https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE38531

Nelson, J., Richards, D., & Lee, D. (2017). Vancomycin and its role in the management of MRSA. *Antibiotics, 6*(2), 16. https://www.ncbi.nlm.nih.gov/books/NBK482221/

Prakash, A., Wang, X., Li, Y., et al. (2012). Gene Expression Omnibus (GEO) accession GSE38531. National Center for Biotechnology Information (NCBI). https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE38531

Thermo Fisher Scientific. (n.d.). *Proinflammatory cytokines: An overview*. Thermo Fisher Scientific. Retrieved December 20, 2024, from https://www.thermofisher.com/es/es/home/life-science/cell-analysis/cell-analysis-learning-center/immunology-at-work/proinflammatory-cytokines-overview.html

Wang, Y., Zhang, X., & Liu, Z. (2022). Gene expression alterations in host tissues during MRSA infections and their implications for treatment strategies. *Scientific Reports, 12*(1), 2337. https://www.nature.com/articles/s41598-024-76212-4

Zhang, Z., & Liu, P. (2020). Splicing and post-transcriptional regulation during bacterial infections: The role of pre-mRNA processing. *Journal of Molecular Biology, 432*(14), 4103-4112. https://pmc.ncbi.nlm.nih.gov/articles/PMC7115970/

VII. Apéndices.

1. Preparación de los datos.

```
# Cargar las librerías necesarias
library(affy)  # Para trabajar con archivos .CEL y ExpressionSet

library(dplyr) # Para manipulación de datos

filter_microarray <- function(allTargets, seed = 123) {
  set.seed(seed)
  filtered <- subset(allTargets, time != "hour 2")
  filtered$group <- interaction(filtered$infection, filtered$agent)
  selected <- do.call(rbind, lapply(split(filtered, filtered$group),
function(group_data) {
    if (nrow(group_data) > 4) {
      group_data[sample(1:nrow(group_data), 4), ]
    } else {
      group_data
    }
  }))

  original_indices <- match(selected$sample, allTargets$sample)

  rownames(selected) <- paste0(selected$sample, ".", original_indices)

  selected$group <- NULL
  return(selected)
}

# Cargar el archivo allTargets
allTargets <- read.table("C:/Users/silvi/Desktop/MSc Bioinformática y
```

```
Bioestadística/202425-1/Anàlisi de dates òmiques/PACS/PAC2/Muestras
CEL/allTargets.txt", header = TRUE, sep = " ")

# Verificar que se ha cargado correctamente
head(allTargets)

##      sample  infection   time     agent
## 1 GSM944831 uninfected hour 0 untreated
## 2 GSM944838 uninfected hour 0 untreated
## 3 GSM944845 uninfected hour 0 untreated
## 4 GSM944852 uninfected hour 0 untreated
## 5 GSM944859 uninfected hour 0 untreated
## 6 GSM944833 uninfected hour 0 linezolid

# Aplicar la función de filtrado
filteredTargets <- filter_microarray(allTargets, seed=53635628)  # Cambia
"123" por tu identificador de la UOC

# Verificar que la función se ha aplicado correctamente
head(filteredTargets)

##                 sample        infection    time     agent
## GSM944836.26 GSM944836 S. aureus USA300 hour 24 linezolid
## GSM944850.28 GSM944850 S. aureus USA300 hour 24 linezolid
## GSM944857.29 GSM944857 S. aureus USA300 hour 24 linezolid
## GSM944843.27 GSM944843 S. aureus USA300 hour 24 linezolid
## GSM944854.9  GSM944854       uninfected  hour 0 linezolid
## GSM944847.8  GSM944847       uninfected  hour 0 linezolid

# Cargar los archivos CEL de las muestras seleccionadas
celFiles <- paste0("C:/Users/silvi/Desktop/MSc Bioinformática y
Bioestadística/202425-1/Anàlisi de dates òmiques/PACS/PAC2/Muestras
CEL/", filteredTargets$sample, ".CEL")

# Leer los archivos CEL para crear un ExpressionSet
rawData <- ReadAffy(filenames = celFiles)

# Ver el ExpressionSet creado
rawData

## Warning: replacing previous import 'AnnotationDbi::head' by
'utils::head' when
## loading 'mouse4302cdf'

## Warning: replacing previous import 'AnnotationDbi::tail' by
'utils::tail' when
## loading 'mouse4302cdf'

##
```

```
## AffyBatch object
## size of arrays=1002x1002 features (28 kb)
## cdf=Mouse430_2 (45101 affyids)
## number of samples=24
## number of genes=45101
## annotation=mouse4302
## notes=

# Verificar las intensidades de expresión génica
exprs(rawData)

#Guardar el objeto
saveRDS(rawData, file = "rawData.rds")

# Revisar la clase del objeto creado
class(rawData)

## [1] "AffyBatch"
## attr(,"package")
## [1] "affy"
```

Para crear el objeto ExpressionSet, se deben tener en cuenta los metadatos de filteredTargets, por lo que:

```
library(Biobase)

# Extraer las intensidades de expresión de rawData
expr_matrix <- exprs(rawData)

# Eliminar la extensión .CEL de los nombres de las columnas.
colnames(expr_matrix) <- sub("\\.CEL$", "", colnames(expr_matrix))

# Como las filas de filteredTargets deben coincidir con las muestras de
expr_matrix, se ordenan para que coincidan con las muestras seleccionadas
filteredTargets <- filteredTargets[match(filteredTargets$sample,
colnames(expr_matrix)), ]

# Eliminar la información adicional después del . en los nombres de las
filas
rownames(filteredTargets) <- sub("\\..*","", rownames(filteredTargets))

# Verificar si los nombres coinciden
if(!all(filteredTargets$sample == colnames(expr_matrix))) {
  stop("Los nombres de las muestras no coinciden entre 'filteredTargets'
y 'expr_matrix'")
}

# Crear un data frame con una columna que contiene los nombres de las
filas de expr_matrix (las sondas)
feature_data <- data.frame(Gene = rownames(expr_matrix))
```

Se genera el ExpressionSet:

```
expression_set <- ExpressionSet(
  # Añadir en assayData la matrix de expresión génica
  assayData = expr_matrix,
  # Añadir en phenoData los metadatos
  phenoData = new("AnnotatedDataFrame", data = filteredTargets),
  # featureData debe contener información relacionada con el nombre de
las sondas
  featureData = new("AnnotatedDataFrame", data = feature_data)
)

# Verificar la clase del objeto
print(class(expression_set))  # Debería devolver "ExpressionSet"

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

Se guarda el objeto ExpressionSet:

```
saveRDS(expression_set, file = "C:/Users/silvi/Desktop/MSc Bioinformática
y Bioestadística/202425-1/Anàlisi de dates òmiques/PACS/PAC2/Muestras
CEL/expressionSet.rds")
```

2. Análisis exploratorio y de control de calidad.

Ahora, se revisa el objeto ExpressionSet para comprobar que se ha generado correctamente:

```
dim(exprs(expression_set))

## [1] 1004004      24

head(expression_set)

## ExpressionSet (storageMode: lockedEnvironment)
## assayData: 6 features, 24 samples
##   element names: exprs
## protocolData: none
## phenoData
##   sampleNames: GSM944836 GSM944850 ... GSM944834 (24 total)
##   varLabels: sample infection time agent
##   varMetadata: labelDescription
## featureData
##   featureNames: 1 2 ... 6 (6 total)
##   fvarLabels: Gene
##   fvarMetadata: labelDescription
## experimentData: use 'experimentData(object)'
## Annotation:

dim(pData(expression_set))
```

```
## [1] 24  4

head(pData(expression_set))

##              sample        infection    time     agent
## GSM944836 GSM944836 S. aureus USA300 hour 24 linezolid
## GSM944850 GSM944850 S. aureus USA300 hour 24 linezolid
## GSM944857 GSM944857 S. aureus USA300 hour 24 linezolid
## GSM944843 GSM944843 S. aureus USA300 hour 24 linezolid
## GSM944854 GSM944854       uninfected  hour 0 linezolid
## GSM944847 GSM944847       uninfected  hour 0 linezolid

dim(fData(expression_set))

## [1] 1004004       1

head(fData(expression_set))

##   Gene
## 1    1
## 2    2
## 3    3
## 4    4
## 5    5
## 6    6
```

Se realizan aproximaciones de calidad con el paquete arrayQualityMetrics:

```
library(arrayQualityMetrics)

arrayQualityMetrics(expression_set)
```

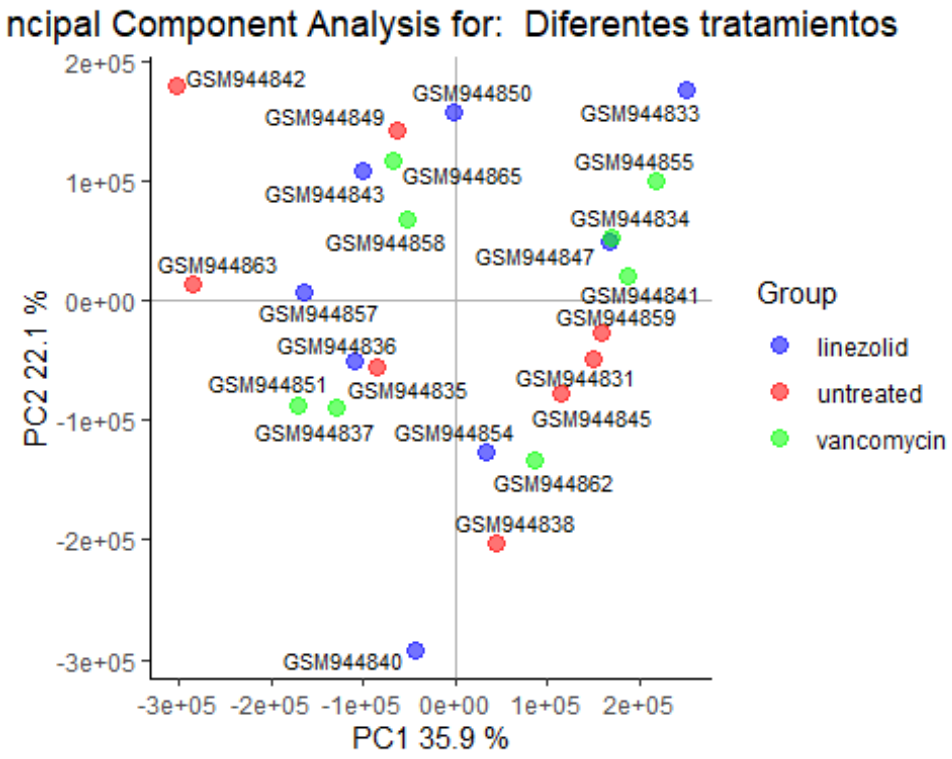
Realizar el análisis de componentes principales:

```
library(ggplot2)
library(ggrepel)

# Generar la función para calcular el PCA
plotPCA3 <- function (datos, labels, factor, title, scale,colores, size =
1.5, glineas = 0.25) {
   data <- prcomp(t(datos),scale=scale)
   dataDf <- data.frame(data$x)
   Group <- factor
   loads <- round(data$sdev^2/sum(data$sdev^2)*100,1)

   # Creación del gráfico
   p1 <- ggplot(dataDf,aes(x=PC1, y=PC2)) +
     theme_classic() +
     geom_hline(yintercept = 0, color = "gray70") +
     geom_vline(xintercept = 0, color = "gray70") +
     geom_point(aes(color = Group), alpha = 0.55, size = 3) +
     coord_cartesian(xlim = c(min(data$x[,1])-5,max(data$x[,1])+5)) +
```

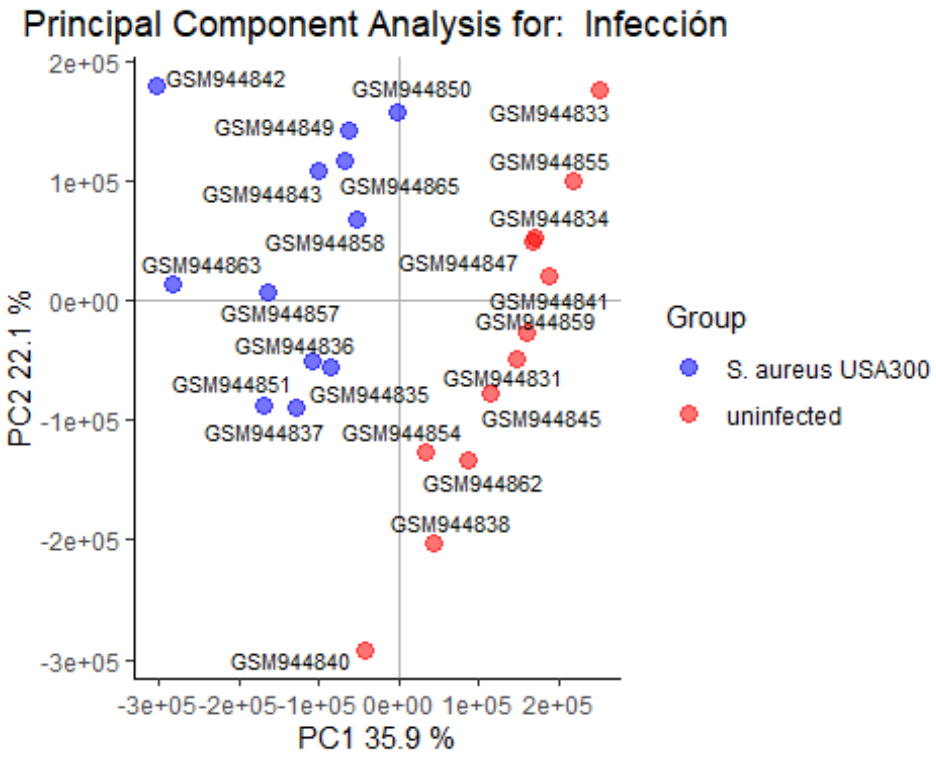
```
    scale_fill_discrete(name = "Group")
  p1 + geom_text_repel(aes(y = PC2 + 0.25, label = labels),segment.size
= 0.25, size = size) +
    labs(x =
c(paste("PC1",loads[1],"%")),y=c(paste("PC2",loads[2],"%"))) +
    ggtitle(paste("Principal Component Analysis for: ",title,sep=" "))+
    theme(plot.title = element_text(hjust = 0.5)) +
    scale_color_manual(values=colores)
  }
```

Se revisa el PCA para los diferentes tratamientos y también para el estado de la infección:

```
plotPCA3(exprs(expression_set), labels = expression_set$sample, factor =
expression_set$agent,
         title="Diferentes tratamientos", scale = FALSE, size = 3,
         colores = c("blue", "red", "green"))
```

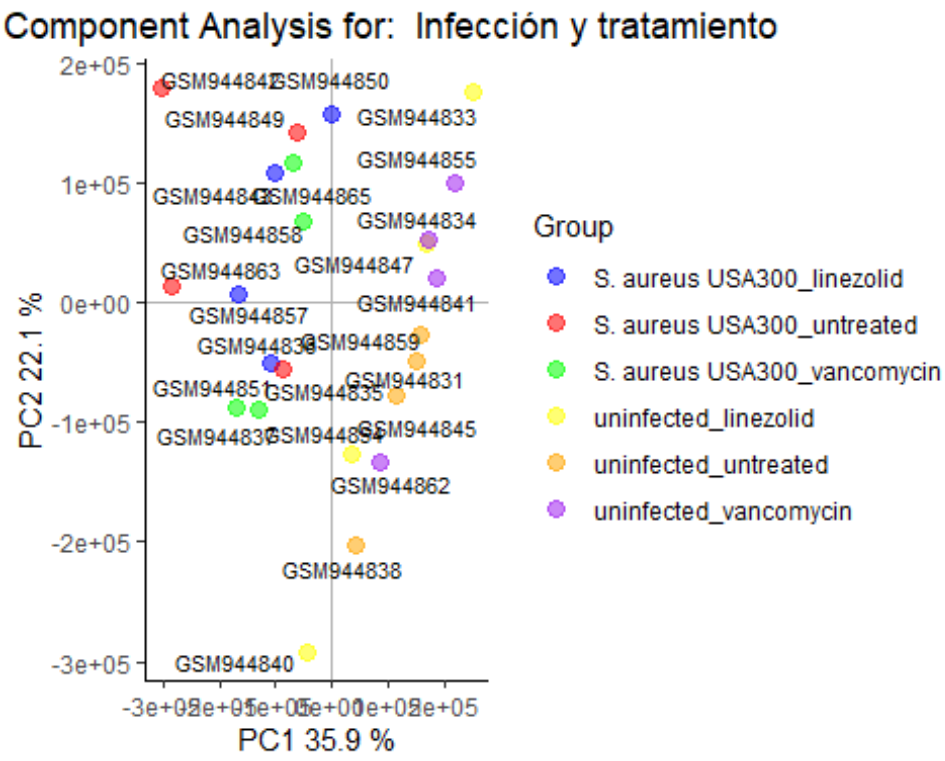



```
plotPCA3(exprs(expression_set), labels = expression_set$sample, factor =
expression_set$infection,
         title="Infección", scale = FALSE, size = 3,
         colores = c("blue", "red"))
```



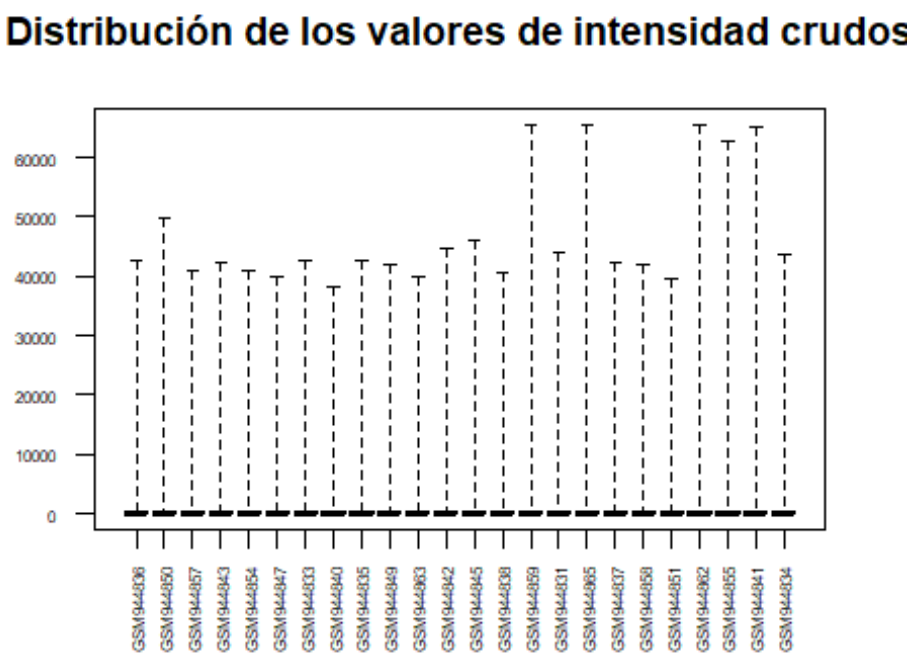
Finalmente, se realiza el análisis para una combinación de ambos:

```
expression_set$Group <- paste(expression_set$infection,
expression_set$agent, sep = "_")

plotPCA3(exprs(expression_set), labels = expression_set$sample, factor =
expression_set$Group,
          title="Infección y tratamiento", scale = FALSE, size = 3,
          colores = c("blue", "red", "green", "yellow", "orange",
"purple"))
```



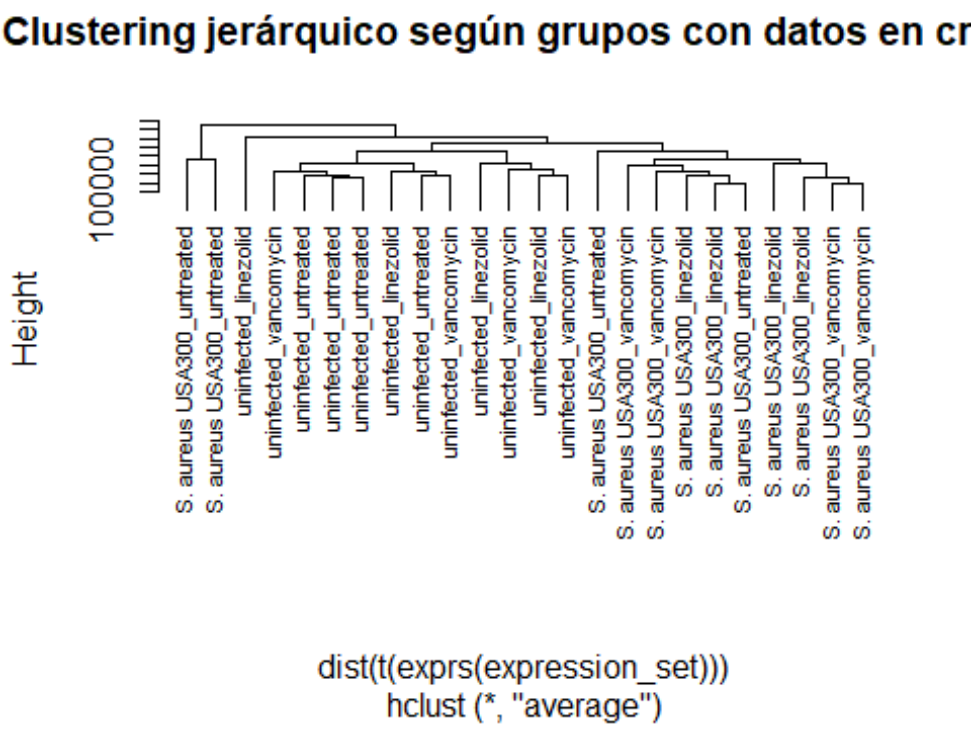
Se realiza un boxplot para visualizar la distribución de intensidades de las matrices:

```
boxplot(expression_set, cex.axis=0.5, las=2,  which="all",
          col = c("blue", "red", "green", "yellow", "orange",
"purple")[as.numeric(as.factor(expression_set$Group))],
          main="Distribución de los valores de intensidad crudos")
```



Generar un análisis de clustering jerárquico:

```
clust.euclid.average <- hclust(dist(t(exprs(expression_set))), method =
"average")

# Graficar el dendrograma
plot(clust.euclid.average, labels = expression_set$Group,
     main = "Clustering jerárquico según grupos con datos en crudo", hang
= -1, cex = 0.7)
```



Normalización de los datos:

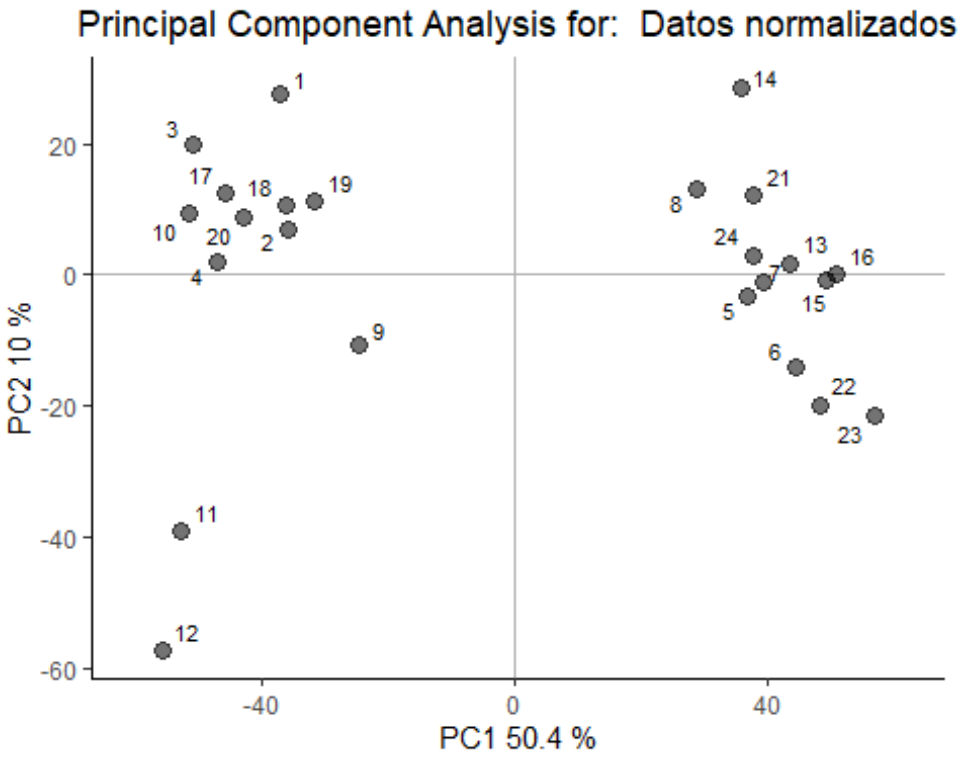
```
eset_rma <- rma(rawData)

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

Control de los datos normalizados:

```
arrayQualityMetrics(eset_rma)
```

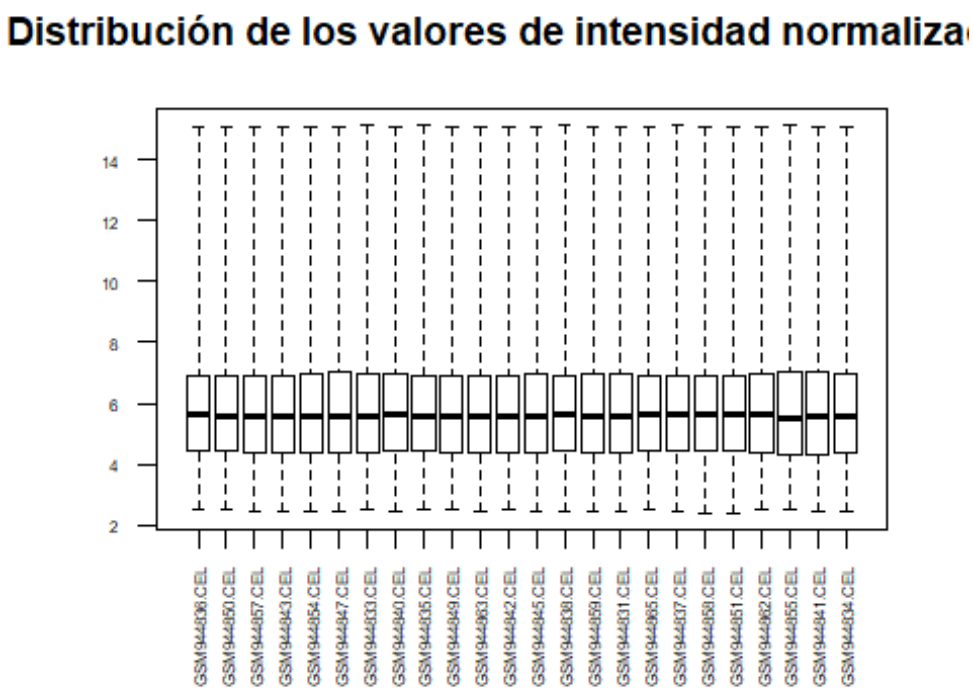
Revisar el análisis de los componentes principales para los datos normalizados:

```
plotPCA3(exprs(eset_rma), labels = eset_rma$sample, factor =
eset_rma$infection,
         title="Datos normalizados", scale = FALSE, size = 3,
         colores = c("red", "blue"))
```



Comparar la distribución de los datos normalizados:

```
boxplot(eset_rma, cex.axis=0.5, las=2,  which="all",
          col = c("blue", "red", "green", "yellow", "orange",
"purple")[as.numeric(as.factor(filteredTargets$Group))],
          main="Distribución de los valores de intensidad normalizados")
```



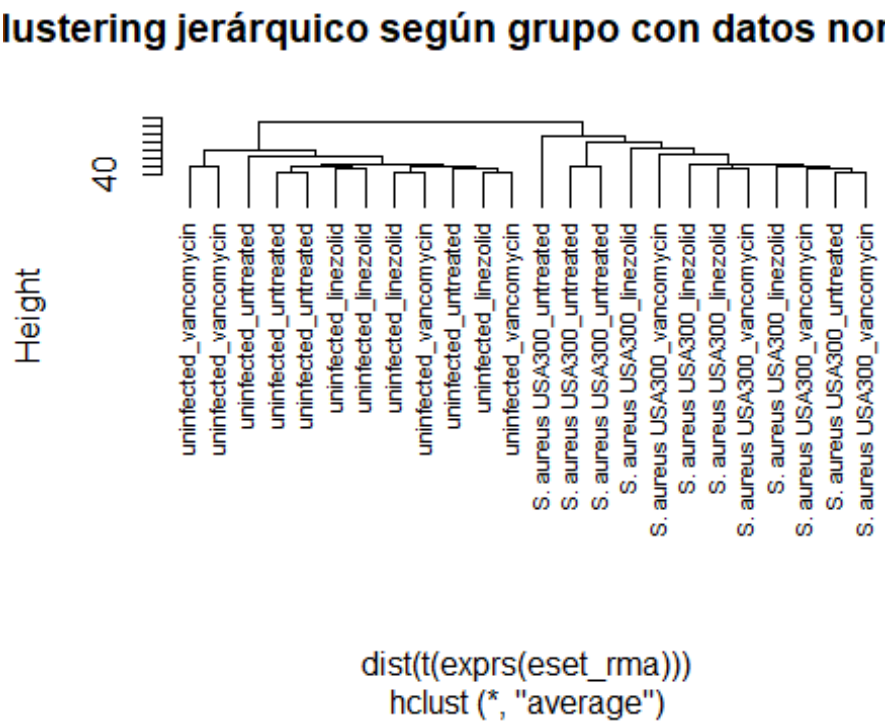
Como los datos han mejorado, se completa el ExpressionSet añadiéndole la información faltante:

```r
colnames(eset_rma) <- sub("\\.CEL$", "", colnames(expr_matrix))

filteredTargets$Group <- paste(filteredTargets$infection,
filteredTargets$agent, sep = "_")

# Asegúrate de que el orden de filteredTargets coincida con las columnas
de eset_rma
filteredTargets <- filteredTargets[match(colnames(exprs(eset_rma)),
filteredTargets$sample), ]

# Crear el objeto AnnotatedDataFrame
pheno_data <- new("AnnotatedDataFrame", data = filteredTargets)

# Asignar el phenoData al ExpressionSet
phenoData(eset_rma) <- pheno_data

# Verificar que el phenoData ha sido añadido correctamente
pData(eset_rma)

##              sample        infection    time      agent
## GSM944836 GSM944836 S. aureus USA300 hour 24  linezolid
## GSM944850 GSM944850 S. aureus USA300 hour 24  linezolid
## GSM944857 GSM944857 S. aureus USA300 hour 24  linezolid
```

```
## GSM944843 GSM944843 S. aureus USA300 hour 24  linezolid
## GSM944854 GSM944854       uninfected  hour 0  linezolid
## GSM944847 GSM944847       uninfected  hour 0  linezolid
## GSM944833 GSM944833       uninfected  hour 0  linezolid
## GSM944840 GSM944840       uninfected  hour 0  linezolid
## GSM944835 GSM944835 S. aureus USA300 hour 24  untreated
## GSM944849 GSM944849 S. aureus USA300 hour 24  untreated
## GSM944863 GSM944863 S. aureus USA300 hour 24  untreated
## GSM944842 GSM944842 S. aureus USA300 hour 24  untreated
## GSM944845 GSM944845       uninfected  hour 0  untreated
## GSM944838 GSM944838       uninfected  hour 0  untreated
## GSM944859 GSM944859       uninfected  hour 0  untreated
## GSM944831 GSM944831       uninfected  hour 0  untreated
## GSM944865 GSM944865 S. aureus USA300 hour 24 vancomycin
## GSM944837 GSM944837 S. aureus USA300 hour 24 vancomycin
## GSM944858 GSM944858 S. aureus USA300 hour 24 vancomycin
## GSM944851 GSM944851 S. aureus USA300 hour 24 vancomycin
## GSM944862 GSM944862       uninfected  hour 0 vancomycin
## GSM944855 GSM944855       uninfected  hour 0 vancomycin
## GSM944841 GSM944841       uninfected  hour 0 vancomycin
## GSM944834 GSM944834       uninfected  hour 0 vancomycin
##                                 Group
## GSM944836  S. aureus USA300_linezolid
## GSM944850  S. aureus USA300_linezolid
## GSM944857  S. aureus USA300_linezolid
## GSM944843  S. aureus USA300_linezolid
## GSM944854        uninfected_linezolid
## GSM944847        uninfected_linezolid
## GSM944833        uninfected_linezolid
## GSM944840        uninfected_linezolid
## GSM944835  S. aureus USA300_untreated
## GSM944849  S. aureus USA300_untreated
## GSM944863  S. aureus USA300_untreated
## GSM944842  S. aureus USA300_untreated
## GSM944845        uninfected_untreated
## GSM944838        uninfected_untreated
## GSM944859        uninfected_untreated
## GSM944831        uninfected_untreated
## GSM944865 S. aureus USA300_vancomycin
## GSM944837 S. aureus USA300_vancomycin
## GSM944858 S. aureus USA300_vancomycin
## GSM944851 S. aureus USA300_vancomycin
## GSM944862       uninfected_vancomycin
## GSM944855       uninfected_vancomycin
## GSM944841       uninfected_vancomycin
## GSM944834       uninfected_vancomycin
```

Generar un dendograma a partir de clustering jerárquico para revisar la similitud entre las muestras:

```r
clust.euclid.average <- hclust(dist(t(exprs(eset_rma))), method =
"average")

plot(clust.euclid.average, labels = eset_rma$Group,
     main = "Clustering jerárquico según grupo con datos normalizados",
hang = -1, cex = 0.7)
```



3. Filtrado de datos.

```r
# Extraer la matriz de expresión normalizada
expr_matrix_norm <- exprs(eset_rma)

# Calcular la desviación estándar de cada sonda
variabilidad <- apply(expr_matrix_norm, 1, sd)

# Ordenar por su SD en orden descendiente, para visualizar los genes con
mayor variabilidad primero
variabilidad_orden <- sort(variabilidad, decreasing = TRUE)

# Seleccionar el 10% de los genes con mayor variabilidad
top10 <- variabilidad_orden[1:floor(length(variabilidad_orden)*0.1)]

# Filtrar la matriz de expresión con los top10 genes
expr_matrix_filter <- expr_matrix_norm [rownames(expr_matrix_norm) %in%
names(top10), ]
```

```r
# Revisar la dimensión y la clase del objeto
dim(expr_matrix_filter)
```

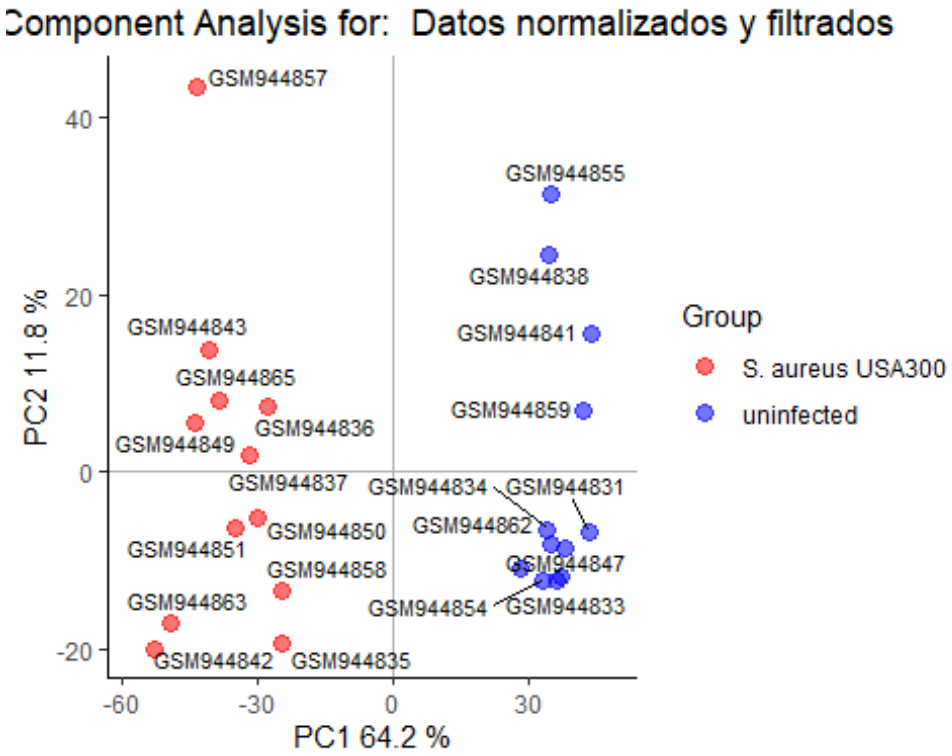
```
## [1] 4510   24
```

```r
class(expr_matrix_filter)
```

```
## [1] "matrix" "array"
```

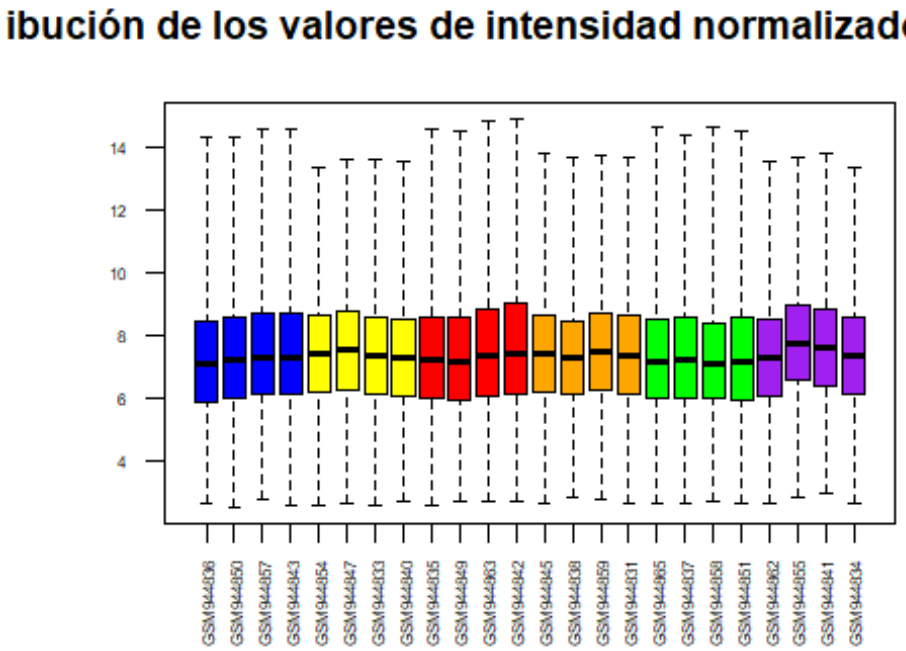
```r
# Crear el objeto ExpressionSet
expression_set_filtered <- ExpressionSet(
  assayData = expr_matrix_filter,
  phenoData = new("AnnotatedDataFrame", data = filteredTargets))

# Realizar un análisis de calidad de los datos normalizados y filtrados
arrayQualityMetrics(expression_set_filtered)

# Revisar el análisis de componentes principales de los datos filtrados
plotPCA3(exprs(expression_set_filtered), labels =
expression_set_filtered$sample, factor =
expression_set_filtered$infection,
         title="Datos normalizados y filtrados", scale = FALSE, size = 3,
         colores = c("red", "blue"))
```

```
## Warning: ggrepel: 2 unlabeled data points (too many overlaps).
Consider
## increasing max.overlaps
```



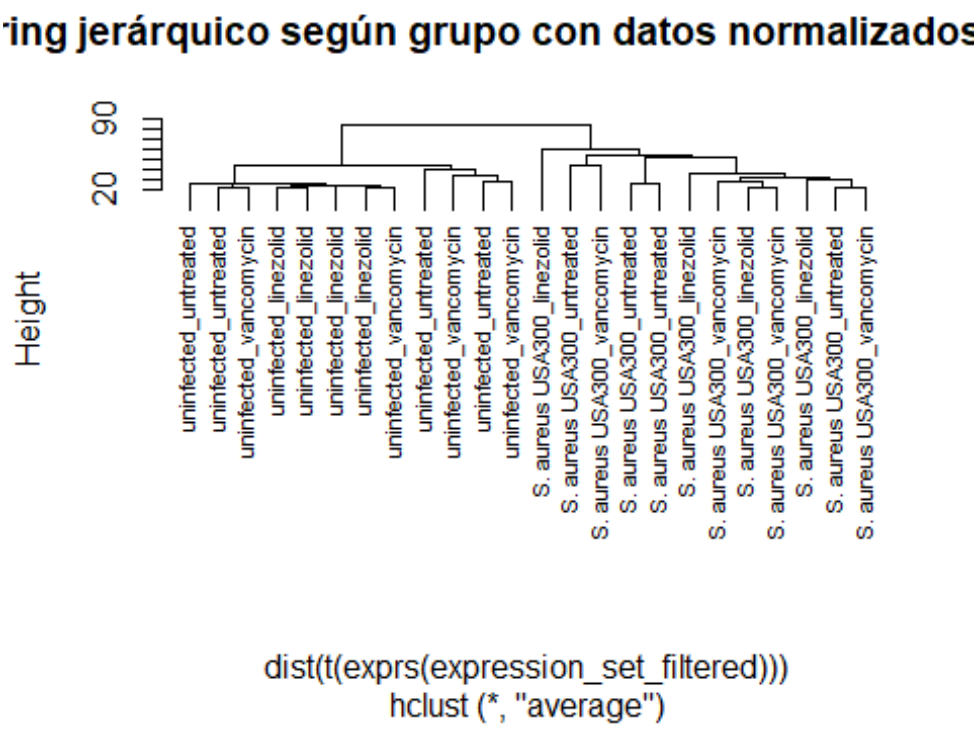
Comprobar si la simetria de los datos normalizados y filtrados ha mejorado:

```
boxplot(expression_set_filtered, cex.axis=0.5, las=2,  which="all",
          col = c("blue", "red", "green", "yellow", "orange",
"purple")[as.numeric(as.factor(expression_set_filtered$Group))],
          main="Distribución de los valores de intensidad normalizados y
filtrados")
```



Revisar también el dendograma de los dastos normalizados y filtrados:

```
clust.euclid.average <- hclust(dist(t(exprs(expression_set_filtered))),
method = "average")

plot(clust.euclid.average, labels = expression_set_filtered$Group,
     main = "Clustering jerárquico según grupo con datos normalizados y
filtrados", hang = -1, cex = 0.7)
```



Guardamos los objetos:

```
write.csv(exprs(eset_rma), file="normalized.Data.csv")
write.csv(exprs(expression_set_filtered),
file="normalized.Filtered.Data.csv")
save(eset_rma, expression_set_filtered, file="normalized.Data.Rda")
```

4. Construcción de las matrices de diseño y de contraste.

Matriz de diseño:

```
library(limma)

##
## Adjuntando el paquete: 'limma'

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

# Para crear la matrix de diseño, indicamos '~0+Group' para que no se
incluya un intercepto y cada grupo se compare independientemente
designMat <- model.matrix(~0+Group, pData(eset_rma))

# Asignar nombres para cada condición experimental
colnames(designMat) <- c("S. aureus USA300_linezolid",
"uninfected_linezolid", "S. aureus USA300_untreated",
"uninfected_untreated", "S. aureus USA300_vancomycin",
```

```
"uninfected_vancomycin")

print(head(designMat))
```

```
##           S. aureus USA300_linezolid uninfected_linezolid
## GSM944836                          1                    0
## GSM944850                          1                    0
## GSM944857                          1                    0
## GSM944843                          1                    0
## GSM944854                          0                    0
## GSM944847                          0                    0
##           S. aureus USA300_untreated uninfected_untreated
## GSM944836                          0                    0
## GSM944850                          0                    0
## GSM944857                          0                    0
## GSM944843                          0                    0
## GSM944854                          0                    1
## GSM944847                          0                    1
##           S. aureus USA300_vancomycin uninfected_vancomycin
## GSM944836                           0                     0
## GSM944850                           0                     0
## GSM944857                           0                     0
## GSM944843                           0                     0
## GSM944854                           0                     0
## GSM944847                           0                     0
```

Para la matriz de contraste:

```
# Generar la matriz de contraste
cont.matrix <- makeContrasts(
  # Entre infectados y no infectados sin tratamiento
  Untreated = uninfected_untreated - S_aur_SAUSA300_untreated,
  # Entre infectados y no infectados tratados con Linezolid
  Linezolid = uninfected_linezolid - S_aur_SAUSA300_linezolid,
  # Entre infectados y no infectados tratados con Vancomycin
  Vancomycin = uninfected_vancomycin - S_aur_SAUSA300_vancomycin,
  # Indicar los niveles de cada condición experimental
  levels = c("uninfected_untreated", "S_aur_SAUSA300_untreated",
"uninfected_linezolid",
             "S_aur_SAUSA300_linezolid", "uninfected_vancomycin",
"S_aur_SAUSA300_vancomycin")
)

print(cont.matrix)
```

```
##                            Contrasts
## Levels                      Untreated Linezolid Vancomycin
##   uninfected_untreated              1         0          0
##   S_aur_SAUSA300_untreated         -1         0          0
##   uninfected_linezolid              0         1          0
##   S_aur_SAUSA300_linezolid          0        -1          0
```

```
##   uninfected_vancomycin             0         0          1
##   S_aur_SAUSA300_vancomycin         0         0         -1
```

5. Obtención del listado de genes diferencialmente expresados para cada comparación.

```
# Ajustamos el modelo considerando los grupos
fit <- lmFit(eset_rma, designMat)

# Renombramos las columnas por cada grupo
colnames(fit$coefficients) <- c("uninfected_untreated",
"S_aur_SAUSA300_untreated",
"uninfected_linezolid", "S_aur_SAUSA300_linezolid",
"uninfected_vancomycin", "S_aur_SAUSA300_vancomycin")

# Aplicamos los contrastes
fit.main <- contrasts.fit(fit, cont.matrix)

# Ajustar los resultados con un enfoque bayesiano para mejorar las
estimaciones
fit.main <- eBayes(fit.main)

# Revisar la clase del objeto creado
class(fit.main)

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

# Generar la tabla con los resultados del análisis diferencial de
expresión para el contraste entre los infectados y no infectados sin
tratamiento
topTab_Untreated <- topTable(fit.main, coef = "Untreated", number =
nrow(fit.main), adjust = "fdr")

head(topTab_Untreated)

##                   logFC   AveExpr         t      P.Value   adj.P.Val
B
## 1420818_at   -0.9113741  7.606501 -7.361741 1.778773e-07 0.008022444
7.066757
## 1435332_at   -0.6670584  5.556057 -6.507219 1.246225e-06 0.012070058
5.355283
## 1420617_at    0.7094318  9.739900  6.466812 1.369499e-06 0.012070058
5.271555
## 1421457_a_at -1.0846924  7.961252 -6.448204 1.430395e-06 0.012070058
5.232914
## 1438657_x_at -0.8493506 10.666215 -6.417054 1.538594e-06 0.012070058
5.168117
```

```
## 1420012_at   -0.6197429  5.980868 -6.339627 1.845258e-06 0.012070058
5.006439

# Generar la tabla para la comparación entre los infectados y no
infectados tratados con Linezolid
topTab_Linezolid <- topTable(fit.main, coef = "Linezolid", number =
nrow(fit.main), adjust = "fdr")
head(topTab_Linezolid)

##                 logFC   AveExpr        t      P.Value    adj.P.Val
B
## 1421262_at   6.863455  7.254610 20.74075 2.418891e-16 1.090944e-11
26.20931
## 1427747_a_at 5.504104 10.690681 16.50025 3.315885e-14 7.477487e-10
22.01542
## 1440865_at   4.016714 11.224037 16.08244 5.704791e-14 8.576393e-10
21.53455
## 1450188_s_at 5.466055  6.583792 15.84540 7.802505e-14 8.797519e-10
21.25555
## 1422953_at   3.148086 11.295737 15.55495 1.151383e-13 1.038570e-09
20.90741
## 1418722_at   5.347673 11.041386 14.49089 5.052984e-13 3.798244e-09
19.57009

# Generar la tabla para la comparación entre infectados y no infectados
tratados con Vancomycin
topTab_Vancomycin <- topTable(fit.main, coef = "Vancomycin", number =
nrow(fit.main), adjust = "fdr")
head(topTab_Vancomycin)

##                 logFC   AveExpr         t      P.Value adj.P.Val
B
## 1442160_at  0.4634657  5.702372  4.729643 9.225381e-05   0.92957 -
1.753238
## 1426824_at  0.7643554  8.048683  4.648914 1.127567e-04   0.92957 -
1.816047
## 1419764_at -1.9194417 10.001179 -4.615027 1.226728e-04   0.92957 -
1.842621
## 1455967_at  0.3874379  5.252219  4.495255 1.652720e-04   0.92957 -
1.937516
## 1459173_at  0.3777258  4.140264  4.384443 2.177757e-04   0.92957 -
2.026616
## 1456220_at  0.4086697  4.498128  4.381171 2.195567e-04   0.92957 -
2.029265
```

6. Anotación de los genes.

```
# Función para añadir anotaciones a la tabla de resultados de los genes
diferencialmente expresados
annotatedTopTable <- function(topTab, anotPackage)
{
```

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

library(mouse4302.db, lib.loc = "C:/Users/silvi/AppData/Local/R/win-

library (annotate)

# Utilizar la función para agregar las anotaciones a las tablas de
resultados para los tres tratamientos
topAnnotated_Untreated <- annotatedTopTable(topTab_Untreated,
anotPackage = "mouse4302.db")

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

topAnnotatedLinezolid <- annotatedTopTable(topTab_Linezolid, anotPackage
= "mouse4302.db")

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

topAnnotatedVancomycin <- annotatedTopTable(topTab_Vancomycin,
anotPackage = "mouse4302.db")

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

# Guardar los resultados
write.csv(topAnnotated_Untreated, file = "topAnnotatedUntreated.csv")
write.csv(topAnnotatedLinezolid, file = "topAnnotatedLinezolid.csv")
write.csv(topAnnotatedVancomycin, file = "topAnnotatedVancomycin.csv")

# Revisar las primeras líneas
head(topAnnotated_Untreated)

##        PROBEID   SYMBOL ENTREZID
## 1   1415670_at    Copg1    54161
## 2   1415671_at Atp6v0d1    11972
## 3   1415672_at   Golga7    57437
## 4   1415673_at     Psph   100678
## 5 1415674_a_at  Trappc4    60409
## 6   1415675_at     Dpm2    13481
##                                                      GENENAME
logFC
## 1                   coatomer protein complex, subunit gamma 1 -
0.28815682
## 2             ATPase, H+ transporting, lysosomal V0 subunit D1
0.07641546
```

```
## 3                                                    golgin A7
0.49821400
## 4                                   phosphoserine phosphatase -
0.20463780
## 5                       trafficking protein particle complex 4 -
0.10275363
## 6 dolichyl-phosphate mannosyltransferase subunit 2, regulatory -
0.20518586
##     AveExpr         t      P.Value  adj.P.Val          B
## 1  7.407278 -2.817351 0.0097965077 0.17466063 -2.7380422
## 2 10.393203  0.839159 0.4100542802 0.78325502 -5.7428887
## 3 11.099827  4.286554 0.0002778619 0.03253291  0.4774293
## 4  6.934834 -1.896210 0.0706182549 0.41428755 -4.4381105
## 5  8.400096 -1.215769 0.2364566804 0.65118353 -5.3777902
## 6  7.873548 -1.907877 0.0690226271 0.41052216 -4.4192743
```

```
head(topAnnotatedLinezolid)
```

```
##        PROBEID   SYMBOL ENTREZID
## 1   1415670_at    Copg1    54161
## 2   1415671_at Atp6v0d1    11972
## 3   1415672_at   Golga7    57437
## 4   1415673_at     Psph   100678
## 5 1415674_a_at  Trappc4    60409
## 6   1415675_at     Dpm2    13481
##                                                     GENENAME
logFC
## 1                   coatomer protein complex, subunit gamma 1 -
0.34041029
## 2             ATPase, H+ transporting, lysosomal V0 subunit D1
0.32641309
## 3                                                    golgin A7 -
0.10755783
## 4                                   phosphoserine phosphatase -
0.15444636
## 5                       trafficking protein particle complex 4
0.03653352
## 6 dolichyl-phosphate mannosyltransferase subunit 2, regulatory -
0.10040011
##     AveExpr          t     P.Value  adj.P.Val         B
## 1  7.407278 -3.3282412 0.002936039 0.02318247 -2.107392
## 2 10.393203  3.5845166 0.001576157 0.01450215 -1.512486
## 3 11.099827 -0.9254104 0.364401739 0.56605645 -6.311339
## 4  6.934834 -1.4311273 0.165898826 0.35203365 -5.734867
## 5  8.400096  0.4322605 0.669594932 0.80688819 -6.648217
## 6  7.873548 -0.9335490 0.360275437 0.56258052 -6.303838
```

```
head(topAnnotatedVancomycin)
```

```
##        PROBEID   SYMBOL ENTREZID
## 1   1415670_at    Copg1    54161
```

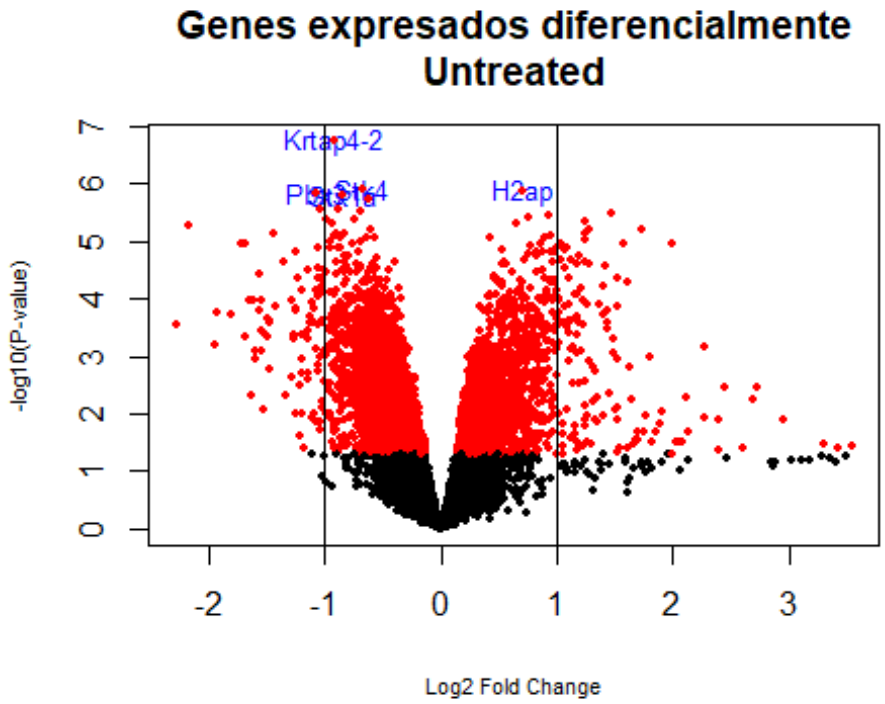
```
## 2   1415671_at Atp6v0d1    11972
## 3   1415672_at   Golga7    57437
## 4   1415673_at     Psph   100678
## 5 1415674_a_at  Trappc4    60409
## 6   1415675_at     Dpm2    13481
##                                                          GENENAME
logFC
## 1                    coatomer protein complex, subunit gamma 1
0.001489393
## 2             ATPase, H+ transporting, lysosomal V0 subunit D1
0.282434443
## 3                                                    golgin A7
0.275904827
## 4                                  phosphoserine phosphatase -
0.146915270
## 5                      trafficking protein particle complex 4 -
0.116618532
## 6 dolichyl-phosphate mannosyltransferase subunit 2, regulatory -
0.163324649
##     AveExpr           t     P.Value adj.P.Val         B
## 1  7.407278  0.01456201 0.988507721 0.9988662 -4.912646
## 2 10.393203  3.10156357 0.005045455 0.9295700 -3.123180
## 3 11.099827  2.37384120 0.026374979 0.9295700 -3.748257
## 4  6.934834 -1.36134287 0.186649906 0.9295700 -4.484482
## 5  8.400096 -1.37981738 0.180966362 0.9295700 -4.473477
## 6  7.873548 -1.51863946 0.142538428 0.9295700 -4.387266
```

7. Expresión diferencial.

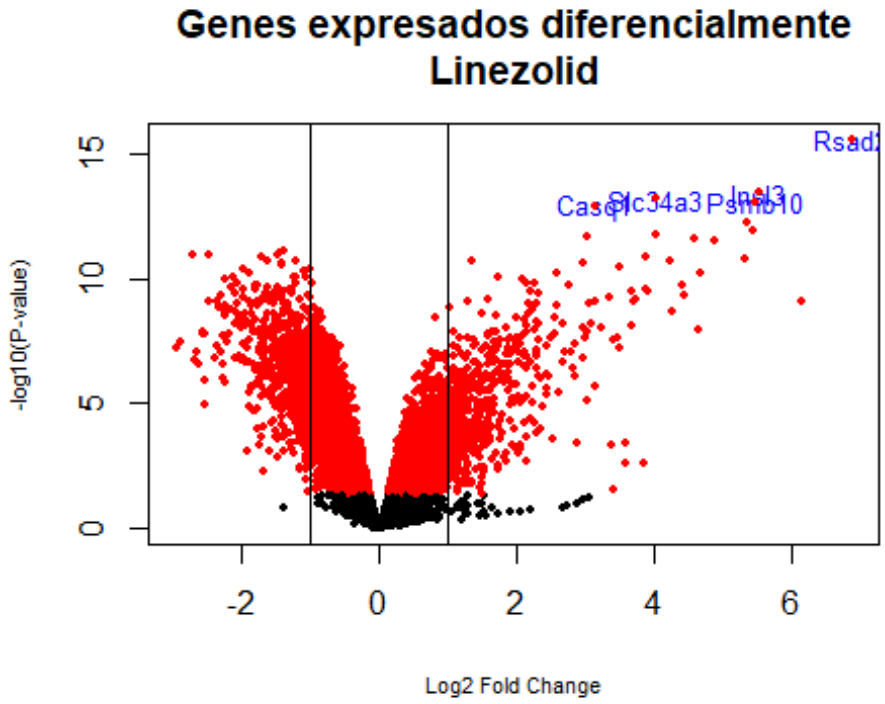
Generar volcano plots para visualizar la expresión diferencial

```
# Para el grupo sin tratamiento, con coefnum=1
coefnum = 1
# Calcular los p-values
p_values <- fit.main$p.value[, coefnum]
# Ajustar el tamaño de las etiquetas del gráfico
opt <- par(cex.lab = 0.7)

# Crear el gráfico volcano plot, escogiendo con highlight destacar los 5
puntos más significativos
volcanoplot(fit.main, coef = coefnum, highlight = 5, names =
topAnnotated_Untreated$SYMBOL,
            main = paste("Genes expresados diferencialmente",
colnames(cont.matrix)[coefnum], sep="\n"))

# Asignar color rojo a los puntos con p-value < 0.05, y negro a los demás
points(fit.main$coefficients[, coefnum], -log10(p_values),
       col = ifelse(p_values < 0.05, "red", "black"), pch = 16, cex =
0.5)
```

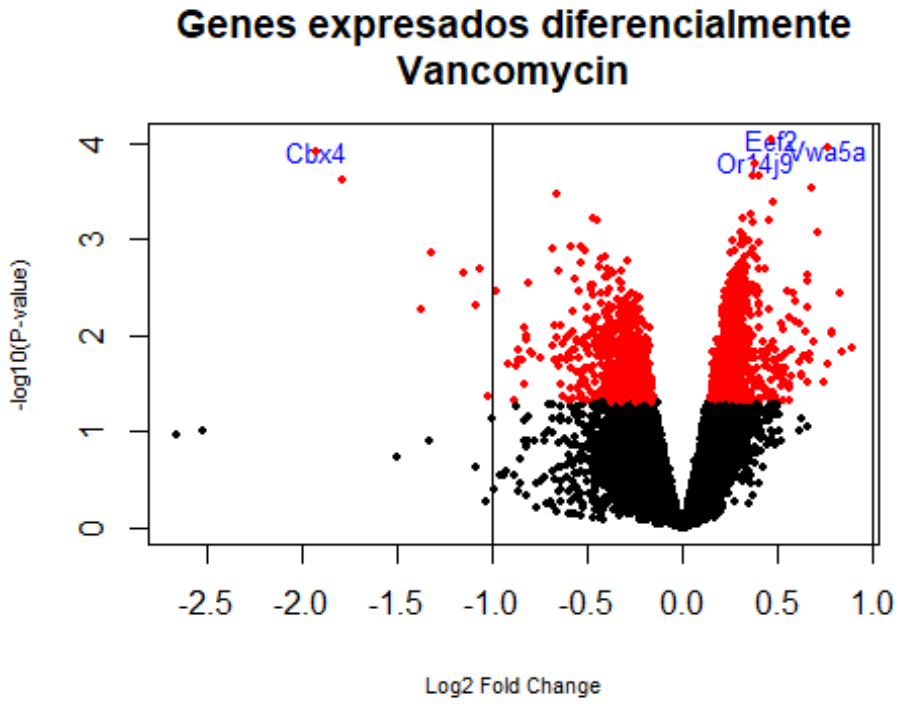
```r
# Añadir línea de corte en el eje X
abline(v=c(-1,1))
```




```r
# Restaurar la configuración de los gráficos
par(opt)

# Aplicar nuevamente el script para el grupo Linezolid
coefnum = 2  #

p_values <- fit.main$p.value[, coefnum]

opt <- par(cex.lab = 0.7)

volcanoplot(fit.main, coef = coefnum, highlight = 5, names =
topAnnotatedLinezolid$SYMBOL,
            main = paste("Genes expresados diferencialmente",
colnames(cont.matrix)[coefnum], sep="\n"))

points(fit.main$coefficients[, coefnum], -log10(p_values),
       col = ifelse(p_values < 0.05, "red", "black"), pch = 16, cex =
0.5)
abline(v=c(-1,1))
```




```r
par(opt)

# Generar el volcano plot para el grupo Vancomycin
coefnum = 3

p_values <- fit.main$p.value[, coefnum]

opt <- par(cex.lab = 0.7)

volcanoplot(fit.main, coef = coefnum, highlight = 5, names =
topAnnotatedVancomycin$SYMBOL,
            main = paste("Genes expresados diferencialmente",
colnames(cont.matrix)[coefnum], sep="\n"))

points(fit.main$coefficients[, coefnum], -log10(p_values),
       col = ifelse(p_values < 0.05, "red", "black"), pch = 16, cex =
0.5)
abline(v=c(-1,1))
```



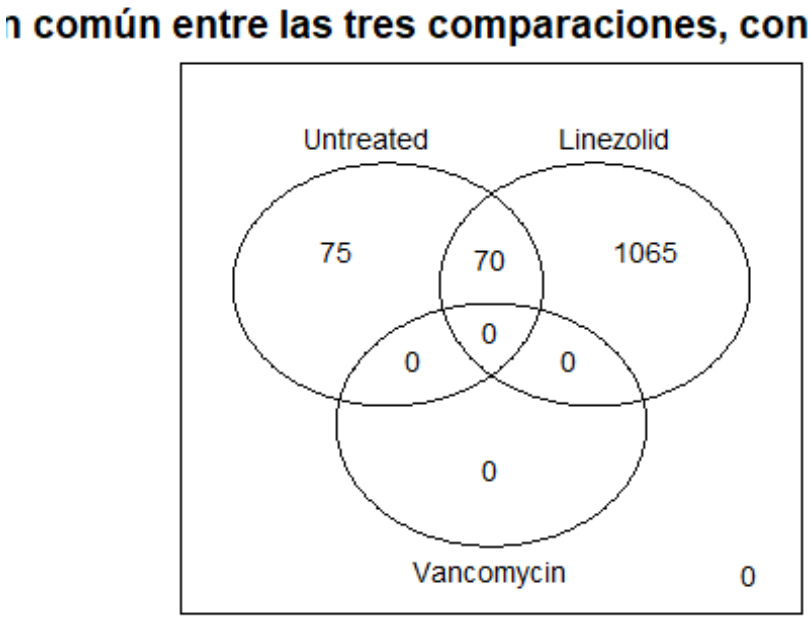

```
par(opt)
```

8. Comparaciones múltiples.

```
# Realizar la prueba de hipótesis en cada coeficiente por separado
res <- decideTests(fit.main, method = "separate", adjust.method = "fdr",
p.value = 0.1, lfc = 1)

# Sumar, por gen, cuántos contrastes fueron significativos para las
condiciones seleccionadas (p-value < 0.1y logFC >= 1)
sum.res.rows <- apply(abs(res), 1, sum)

# Filtrar los resultados solo para los genes significativos
res.selected <- res[sum.res.rows!=0,]

# Mostrar un resumen de los resultados
print(summary(res))

##        Untreated Linezolid Vancomycin
## Down          71       655          0
## NotSig     44956     43966      45101
## Up            74       480          0
```

Para mejor visualización, se genera un diagrama de Venn:

```
vennDiagram(res.selected[,1:3], cex = 0.9)
title("Genes en común entre las tres comparaciones, con FDR < 0.1 y logFC
>1 ")
```



9. Análisis de la significación biológica.

```
# Crear una lista con la comparación de genes por cada tratamiento
listOfTables <- list(Untreated = topTab_Untreated,
                     Linezolid = topTab_Linezolid,
                     Vancomycin = topTab_Vancomycin)

# Generar una función para calcular cuantos genes son significativos por
tratamiento y guardarlos con su EntrezID
listOfSelected <- list()
for (i in 1:length(listOfTables)){
  topTab <- listOfTables[[i]]
  whichGenes <- topTab["adj.P.Val"] < 0.15
  selectIDs <- rownames(topTab)[whichGenes]
  EntrezIDs <- select(mouse4302.db, selectIDs, c("ENTREZID"))
  EntrezIDs <- EntrezIDs$ENTREZID
  listOfSelected[[i]] <- EntrezIDs
  names(listOfSelected)[i] <- names(listOfTables)[i]
}

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

```r
# Aplicar la función sobre nuestras muestras
sapply(listOfSelected, length)
```

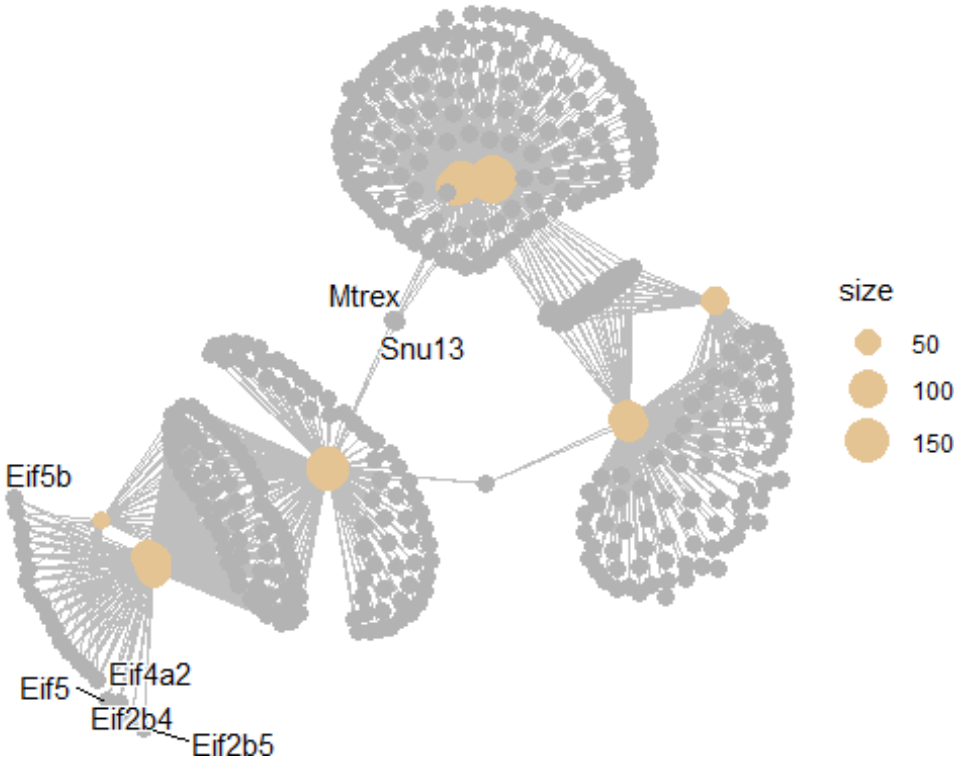
```
##  Untreated  Linezolid Vancomycin
##       2186      13647          0
```

```r
# Creamos lista de genes con información sobre las funciones biológicas, procesos y componentes celulares
mapped_genes2GO <- mappedkeys(org.Mm.egGO)

# Creamos lista de genes con información sobre las rutas metabólicas y de señalización celular
mapped_genes2KEGG <- mappedkeys(org.Mm.egPATH)

# Combinamos ambas listas, revisando que no hayan duplicados
mapped_genes <- union(mapped_genes2GO , mapped_genes2KEGG)

library(ReactomePA)

# Crear un script para identificar las vía biológicas relevantes asociadas a los genes diferencialmente expresados. Los resultados se guardane en archivos CSV
listOfData <- listOfSelected[1:2]
comparisonsNames <- names(listOfData)
universe <- mapped_genes

for (i in 1:length(listOfData)){
   genesIn <- listOfData[[i]]
   comparison <- comparisonsNames[i]
   enrich.result <- enrichPathway(gene = genesIn,
                                  pvalueCutoff = 0.05,
                                  readable = T,
                                  pAdjustMethod = "BH",
                                  organism = "mouse",
                                  universe = universe)

   cat("####################################")
   cat("\nComparison: ", comparison,"\n")
   print(head(enrich.result))

   if (length(rownames(enrich.result@result)) != 0) {
   write.csv(as.data.frame(enrich.result),
             file =paste0("ReactomePA.Results.",comparison,".csv"),
             row.names = FALSE)

   pdf(file=paste0("ReactomePABarplot.",comparison,".pdf"))
     print(barplot(enrich.result, showCategory = 15, font.size = 4,
             title = paste0("Reactome Pathway Analysis for ",
comparison,". Barplot")))
   dev.off()
```

```
   pdf(file = paste0("ReactomePAcnetplot.",comparison,".pdf"))
     print(cnetplot(enrich.result, categorySize = "geneNum",
schowCategory = 15,
          vertex.label.cex = 0.75))
   dev.off()
   }
 }

## ####################################
## Comparison:  Untreated
##                           ID
## R-MMU-1280215 R-MMU-1280215
## R-MMU-168898   R-MMU-168898
## R-MMU-166016   R-MMU-166016
## R-MMU-166166   R-MMU-166166
## R-MMU-937061   R-MMU-937061
## R-MMU-168928   R-MMU-168928
##                                                          Description
GeneRatio
## R-MMU-1280215                    Cytokine Signaling in Immune system
74/864
## R-MMU-168898                            Toll-like Receptor Cascades
30/864
## R-MMU-166016                     Toll Like Receptor 4 (TLR4) Cascade
25/864
## R-MMU-166166                          MyD88-independent TLR4 cascade
22/864
## R-MMU-937061                   TRIF (TICAM1)-mediated TLR4 signaling
22/864
## R-MMU-168928  DDX58/IFIH1-mediated induction of interferon-alpha/beta
11/864
##                 BgRatio RichFactor FoldEnrichment   zScore       pvalue
## R-MMU-1280215 467/8684  0.1584582       1.592652 4.376128 2.761487e-05
## R-MMU-168898  144/8684  0.2083333       2.093943 4.399825 6.473297e-05
## R-MMU-166016  112/8684  0.2232143       2.243510 4.402564 8.077951e-05
## R-MMU-166166   96/8684  0.2291667       2.303337 4.268102 1.417572e-04
## R-MMU-937061   96/8684  0.2291667       2.303337 4.268102 1.417572e-04
## R-MMU-168928   32/8684  0.3437500       3.455006 4.624424 1.605409e-04
##                  p.adjust     qvalue
## R-MMU-1280215 0.02380824 0.02213871
## R-MMU-168898  0.02380824 0.02213871
## R-MMU-166016  0.02380824 0.02213871
## R-MMU-166166  0.02380824 0.02213871
## R-MMU-937061  0.02380824 0.02213871
## R-MMU-168928  0.02380824 0.02213871
##
geneID
## R-MMU-1280215
##               Count
```

```
## R-MMU-1280215    74
## R-MMU-168898     30
## R-MMU-166016     25
## R-MMU-166166     22
## R-MMU-937061     22
## R-MMU-168928     11

## ###################################
## Comparison:  Linezolid
##                          ID
## R-MMU-72203     R-MMU-72203
## R-MMU-6791226 R-MMU-6791226
## R-MMU-72312     R-MMU-72312
## R-MMU-8868773 R-MMU-8868773
## R-MMU-72172     R-MMU-72172
## R-MMU-72689     R-MMU-72689
##
Description
## R-MMU-72203                 Processing of Capped Intron-Containing
Pre-mRNA
## R-MMU-6791226 Major pathway of rRNA processing in the nucleolus and
cytosol
## R-MMU-72312                                                    rRNA
processing
## R-MMU-8868773                    rRNA processing in the nucleus and
cytosol
## R-MMU-72172                                                     mRNA
Splicing
## R-MMU-72689                        Formation of a pool of free 40S
subunits
##                GeneRatio  BgRatio RichFactor FoldEnrichment   zScore
## R-MMU-72203    179/3969 259/8684  0.6911197       1.512140 7.676940
## R-MMU-6791226  128/3969 176/8684  0.7272727       1.591241 7.270125
## R-MMU-72312    128/3969 176/8684  0.7272727       1.591241 7.270125
## R-MMU-8868773  128/3969 176/8684  0.7272727       1.591241 7.270125
## R-MMU-72172    127/3969 191/8684  0.6649215       1.454819 5.831223
## R-MMU-72689     75/3969 101/8684  0.7425743       1.624720 5.793771
##                     pvalue     p.adjust       qvalue
## R-MMU-72203   9.966158e-15 1.135145e-11 9.892723e-12
## R-MMU-6791226 1.943183e-13 5.533214e-11 4.822162e-11
## R-MMU-72312   1.943183e-13 5.533214e-11 4.822162e-11
## R-MMU-8868773 1.943183e-13 5.533214e-11 4.822162e-11
## R-MMU-72172   3.923238e-09 8.187969e-07 7.135765e-07
## R-MMU-72689   4.313241e-09 8.187969e-07 7.135765e-07
##
geneID
## R-MMU-72203   ##               Count
## R-MMU-72203     179
## R-MMU-6791226   128
## R-MMU-72312     128
```

```
## R-MMU-8868773   128
## R-MMU-72172     127
## R-MMU-72689      75

## Warning: ggrepel: 34 unlabeled data points (too many overlaps).
Consider
## increasing max.overlaps

# Generar un gráfico de red de las vías biológicas enriquecidas
cnetplot(enrich.result, categorySize = "geneNum", showCategory = 15,
         vertex.label.cex = 0.75)

## Warning: ggrepel: 416 unlabeled data points (too many overlaps).
Consider
## increasing max.overlaps
```



10. Listado de archivos generados.

expressionSets.rds

filtered_rawData.rds

arrayQualityMetrics report for eset_rma

arrayQualityMetrics report for expression_set

arrayQualityMetrics report for expression_set_filtered

normalized.Data

normalized.Data.Rda

normalized.Filtered.Data.csv

rawData.rds

ReactomePA.Results.Linezolid

ReacomtePA.Results.Untreated.csv

ReactomePABarplot.Linezolid.csv

ReactomePABarplot.Untreated.pdf

ReactomePAcnetplot.Linezolid.pdf

ReactomePAcnetplot.Untreated.pdf

topAnnotatedLinezolid.csv

topAnnotatedUntreated.csv

topAnnotatedVancomycin.csv