Alumna: Consuelo Marín-Vicente

Asignatura: Análisis de datos ómicos (ADO)

PEC 1: datos completos se encuentran en el repositorio:

https://github.com/cmarinvic/MARIN_VICENTE_CONSUELO_PEC1_1

1. **Descarga de datos para el análisis:**
   He navegado en la página Metabolomics Workbench y he extraído los datos del dataset ST000567. Este dataset incluye un archivo disponible para ser descargado:

- ST000567_AN000871_Results.txt **(3.7M)**

Resumen del proyecto seleccionado:

La edad es uno de los mayores factor de riesgo en enfermedades cardiovasculares. Dos parámetros que contribuyen a este riesgo son la rigidez de las arterias y el desarrollo de disfunciones endoteliales que se indican mediante EDD (impaired nitric oxide induced endothelium dependent dilation. Disfunción en la dilatación del endotelio causada por óxido nítrico). La reducción de la rigidez en arterias y el incremento de la función endotelial vascular mediante el incremento en la disponibilidad del NO reducen el riesgo de enfermedad. Estudios preliminares han mostrado que el uso de curcumina en la dieta ayuda a reducir dicho riesgo. El objetivo del proyecto seleccionado es el estudio del metaboloma plasmídico tras la administración oral de curcumina.

Al inspeccionar el archivo veo que los metadatos están incluídos en el mismo archivo descargado. En este archivo se incluyen en las columnas los metabolitos identificados con la masa del precursor analizado así como la anotación al metabolito concreto si es el caso.

En las primera y segunda filas se incluyen los nombres de las muestras y la descripción de cada una de ellas, respectivamente.

Me quedo con la información de la primera fila en el archivo a estudiar, y la información de la primera columna. Igualmente, extraigo los metadatos en archivo adicional (dos primeras filas y la primera columna: metadata.xlsx).

2. **Creación de objeto de clase SummarizedExperiment:**
   Ahora que tenemos los datos de expresión, podemos proceder a crear el objeto SummarizedExperiment (ver R script: MARINVICENTE_CONSUELO_ADO_PEC1_3.R y en archivo MARIN_VICENTE_CONSUELO_ADO_PEC1_RCODE_AND_RESULTS.pdf)

```
# Crear el objeto SummarizedExperiment
se_object <- SummarizedExperiment(
  assays = list(counts = data_matrix),  # Los datos de expresión
  # No hay metadatos de las columnas (o puedes añadirlos si los tienes
)
)
```

```
# Verifica el objeto SummarizedExperiment
se_object
```

```
## class: SummarizedExperiment
## dim: 9814 144
## metadata(0):
## assays(1): counts
## rownames(9814): 413.858@0.70853347 440.0761@0.6910459 ...
##   402.2583@1.114 975.1623@5.697
## rowData names(0):
## colnames(144): X17may15_004.r001 X17may15_004.r002 ...
##   X18may15_040.r001 X18may15_040.r002
## colData names(0):
```

Diferencias entre ExpressionSEt y SummarizedExperiment: A resumir

- **ExpressionSet**: La clase ExpressionSet es parte del paquete **Biobase** de Bioconductor. Es una clase de datos más antigua que se utiliza principalmente para almacenar datos de expresión génica, como las matrices de expresión, junto con metadatos de las filas (genes) y las columnas (muestras). Se asocia principalmente con datos de arrays y algunos tipos de datos de expresión. En cuanto a su estructura, los datos de expresión se almacenan principalmente en una matriz de expresión (gen x muestra). Los metadatos relacionados con las filas (genes) y las columnas (muestras) se almacenan como objetos AnnotatedDataFrame. Es relativamente rígido en su estructura y no soporta de manera eficiente múltiples tipos de datos dentro del mismo objeto. En cuanto a compatibilidad con nuevos paquetes, aunque muchos paquetes antiguos y enfoques de análisis se basan en ExpressionSet, está un poco más limitado en comparación con el estándar actual de clases en Bioconductor.

- **SummarizedExperiment**: La clase SummarizedExperiment, que pertenece al paquete **SummarizedExperiment**, es más moderna y flexible que ExpressionSet. Está diseñada para ser más general y se utiliza para almacenar y manejar datos de cualquier tipo de experimento omico. Es una clase más versátil y extensible que permite manejar datos más complejos. Tiene una estructura más flexible: el objeto puede almacenar **varios "assays"** (por ejemplo, matrices de expresión para diferentes condiciones o tecnologías) dentro de una lista. Además, los metadatos de las filas y las columnas se almacenan en **rowData** y **colData** como objetos DataFrame. Esto permite una mayor flexibilidad en la gestión de diferentes tipos de datos y metadatos. Está diseñado para ser la clase de elección en nuevos desarrollos en Bioconductor.

3. **Análisis exploratorio:**

Un análisis exploratorio de datos normalmente incluye varios pasos, como la inspección de las dimensiones del conjunto de datos, la visualización de las distribuciones de las variables, la evaluación de la calidad de los datos y la detección de posibles outliers. Esto lo vamos a hacer

con funciones de exploración sobre el objeto creado tal cual se muestra en el R script y en el código destacado más abajo (ver R script: MARINVICENTE_CONSUELO_ADO_PEC1_3.R y en archivo MARIN_VICENTE_CONSUELO_ADO_PEC1_RCODE_AND_RESULTS.pdf)

Los box plots nos muestran una similitud en la distribución de la señal por muestra. La distribución de las muestras en base a esta señal deja ver en el PCA una acumulación de las muestras en un cluster común mientras que hay algunas muestras que son outliers de este agrupamiento. El análisis de variabilidad nos indica cuáles son los metabolitos más abundantes a lo largo del ensayo.

```
#ANÁLISIS EXPLORATORIO:

# Ver el objeto SummarizedExperiment
se_object
```

```
## class: SummarizedExperiment
## dim: 9814 144
## metadata(0):
## assays(1): counts
## rownames(9814): 413.858@0.70853347 440.0761@0.6910459 ...
##   402.2583@1.114 975.1623@5.697
## rowData names(0):
## colnames(144): X17may15_004.r001 X17may15_004.r002 ...
##   X18may15_040.r001 X18may15_040.r002
## colData names(0):
```

```
# Obtener dimensiones de la matriz de expresión
dim(se_object)
```

```
## [1] 9814  144
```

```
# Ver los nombres de las filas (metabolitos)
head(rownames(se_object))
```

```
## [1] "413.858@0.70853347"  "440.0761@0.6910459"  "542.0147@0.6911163
"
## [4] "565.0907@0.69125646" "418.0785@0.7215334"  "667.0658@0.7038285
"
```

```
# Ver los nombres de las columnas (muestras)
head(colnames(se_object))
```

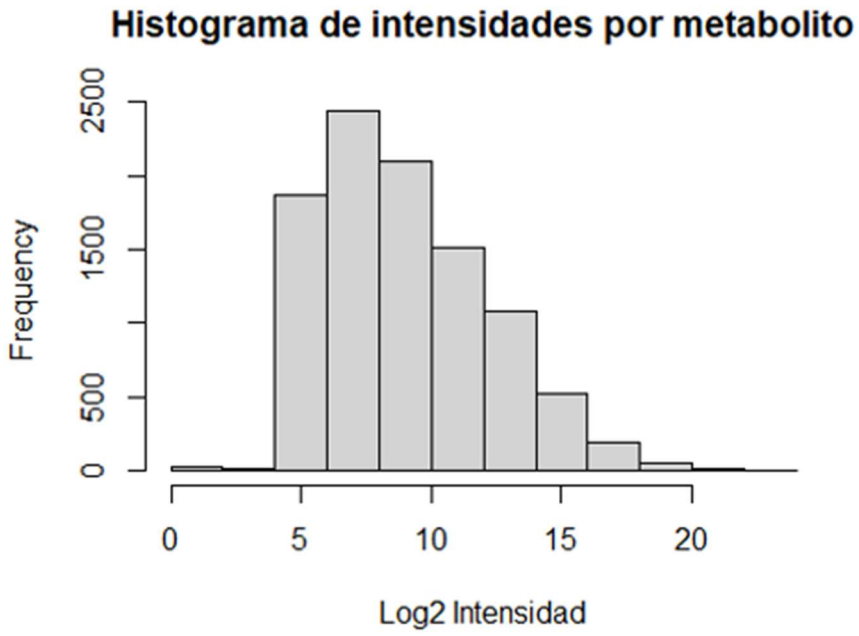
```
## [1] "X17may15_004.r001" "X17may15_004.r002" "X17may15_005.r001"
## [4] "X17may15_005.r002" "X17may15_006.r001" "X17may15_006.r002"
```

```
#Para calcular la media:
# Acceder a la matriz de datos
data_matrix <- assays(se_object)$counts

# Calcular la media y desviación estándard por fila excluyendo la prim
```

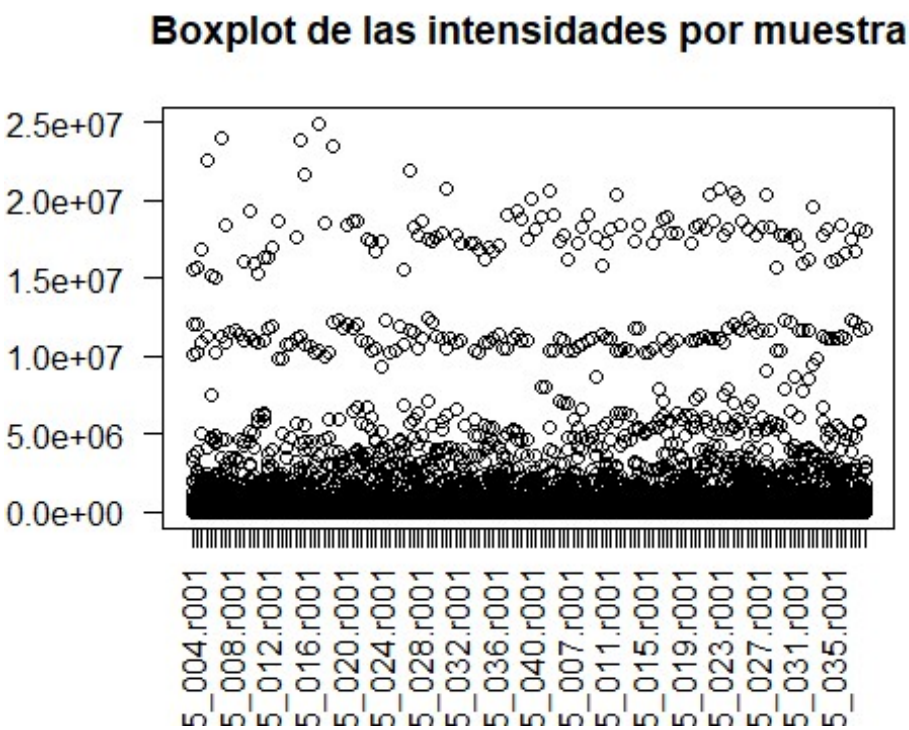
```r
era columna
row_means <- rowMeans(data_matrix[, 2:ncol(data_matrix)], na.rm = TRUE
)
row_sds <- apply(assays(se_object)$counts, 1, sd)

# Ver las primeras medias calculadas
head(row_means)
```

```
##  413.858@0.70853347  440.0761@0.6910459  542.0147@0.6911163 565.090
7@0.69125646
##             545.042           24545.615           10704.748
15148.420
##  418.0785@0.7215334  667.0658@0.7038285
##            3826.483           53671.280
```

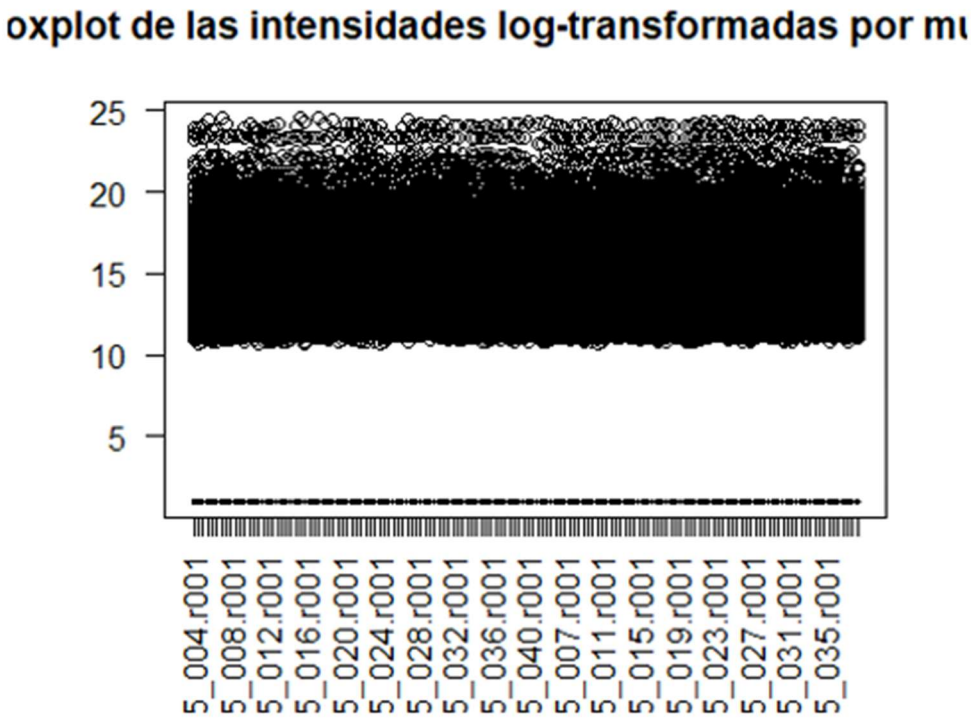
```r
# Calcular la media y desviación estándar por muestra (columna)
col_means <- colMeans(assays(se_object)$counts)
col_sds <- apply(assays(se_object)$counts, 2, sd)


#Visualización de datos:

# Histograma de las intensidades para todos los metabolitos (filas)
hist(log2(row_means + 1), main = "Histograma de intensidades por metab
olito", xlab = "Log2 Intensidad")
```



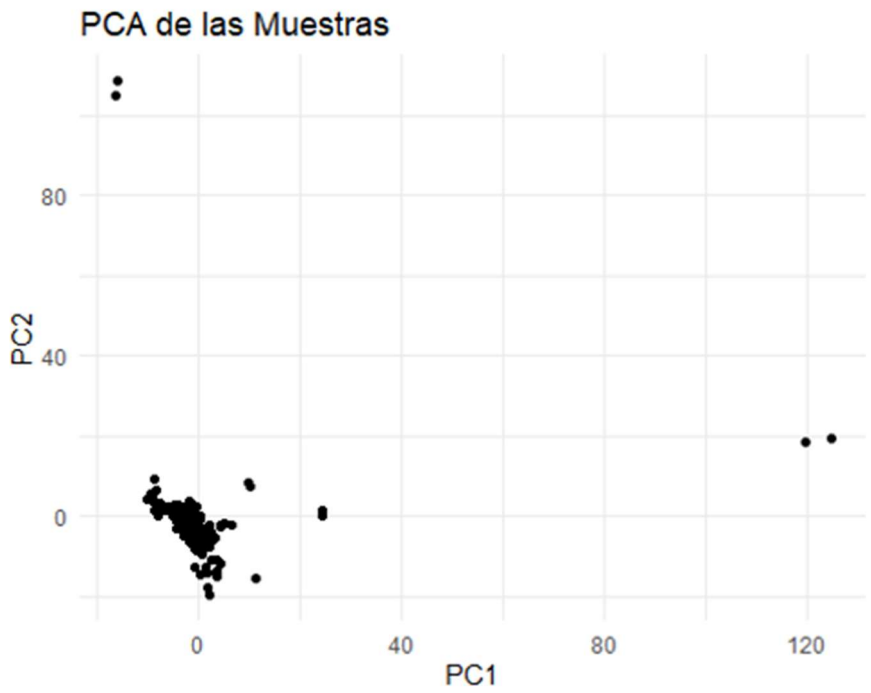

```r
# Boxplot de las intensidades para las muestras (columnas)
boxplot(assays(se_object)$counts, main = "Boxplot de las intensidades
por muestra", las = 2)
```



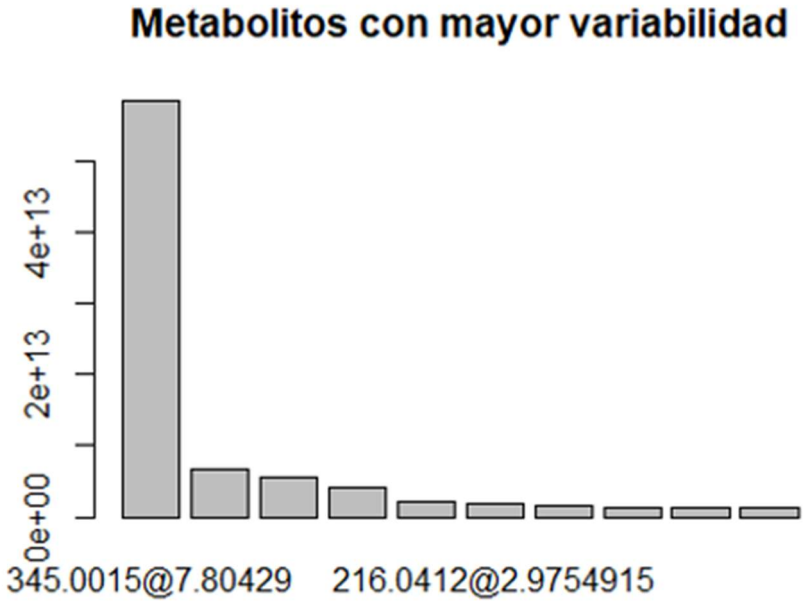

```r
#Escalado y normalización de los datos:

# Log-transformación de los datos de expresión
log_counts <- log2(assays(se_object)$counts + 1)

# Boxplot después de log-transformar los datos
boxplot(log_counts, main = "Boxplot de las intensidades log-transforma
das por muestra", las = 2)
```




```r
# Realizar PCA sobre los datos
pca_result <- prcomp(t(log_counts), center = TRUE, scale. = TRUE)

# Visualizar los resultados del PCA
library(ggplot2)
pca_data <- data.frame(pca_result$x)

# Graficar las primeras dos componentes principales
ggplot(pca_data, aes(x = PC1, y = PC2)) +
  geom_point(aes(color = colData(se_object)$condition)) +  # Cambia
'condition' por el nombre de la variable de interés
labs(title = "PCA de las Muestras", x = "PC1", y = "PC2") +
  theme_minimal()
```




```r
#Estudiar variabilidad:

# Calcular la varianza por metabolito
var_per_metabolite <- apply(assays(se_object)$counts, 1, var)

# Visualizar los metabolitos con mayor varianza
top_var_metabolites <- order(var_per_metabolite, decreasing = TRUE)[1:
10]
barplot(var_per_metabolite[top_var_metabolites], main = "Metabolitos c
on mayor variabilidad")
```


Metabolitos con mayor variabilidad
0e+00
2e+13
4e+13
345.0015@7.80429
216.0412@2.9754915