PEC 1

Andrés Molina Ribagorda

2024-11-05

Resumen

El objetivo de este estudio es aprender a utilizar herramientas bioinformáticas como Bioconductor para analizar datos ómicos, específicamente datos metabolómicos. Para ello, se seleccionó un conjunto de datos sobre metabotipos en respuesta a cirugías bariátricas, disponible en un repositorio de GitHub. Los datos se organizaron en un objeto de tipo SummarizedExperiment y se exploraron para caracterizar la variabilidad interna, identificar patrones y evaluar diferencias entre grupos experimentales. Además, se realizaron análisis de componentes principales (PCA) y de agrupamiento (k-means) para investigar la estructura de los datos. Los resultados muestran agrupaciones diferenciadas en los momentos de muestreo y sugieren que algunas características de los pacientes influyen en la distribución de las muestras en el espacio PCA. Sin embargo, limitaciones como la falta de datos en ciertos bloques y la necesidad de imputación influyen en la interpretación. A pesar de estas limitaciones, el trabajo cumple el objetivo de familiarizarse con el análisis de datos ómicos y el uso de herramientas bioinformáticas.

Índice

Objetivos del estudio

El objetivo del estudio principal consiste en familiarizarse con herramientas bioinformáticas como Bioconductor para realizar un análisis de datos ómicos. Para ello, será necesario aprender a buscar y descargar datos ómicos, en este caso metabolómicos, crear un contenedor del tipo SummarizedExperiment para trabajar con él y analizar los datos para obtener una visión general de su estructura y distribución.

Estos análisis tendrán como objetivo:

- Caracterizar la variabilidad de las mediciones dentro de cada grupo.
- Transformar los datos para su correcta visualización y tratamiento en R.
- Visualizar y explorar patrones y estructuras en los datos.
- Reducir la dimensionalidad para identificar patrones.
- Buscar relaciones entre las variables y su relación con las muestras.
- Identificar diferencias significativas entre grupos experimentales.

Además, también se trabajará el uso de repositorios como Github para la descarga de archivos y la entrega del trabajo.

Por último, decir que el estudio tiene un enfoque académico, por lo que se tratará de adquirir los conocimientos y destrezas necesarias para aprender a realizar un análisis de datos ómicos desde cero, haciendo hincapié en cada paso del proceso seguido.

Materiales y recursos

Los materiales y recursos utilizados durante este trabajo son aquellos relativos a la asignatura de ***Análisis de datos Ómicos***. Han sido de especial utilizad los enlaces relativos a la *Actividad 1.3*, los cuales han servido como ideas de los análisis que se podían hacer, muchos de los cuáles se han llevado a cabo aquí. Entre ellos destacan las páginas web Introduction to microarray data exploration and analysis with basic R functions y Casos y Ejemplos de Análisis Multivariante con R.

Además de estos enlaces, también se ha hecho uso de la página web oficial de Bioconductor para obtener toda la información relativa a *SummarizedExperiment*. Todo el trabajo se ha llevado a cabo en R, concretamente en formato RMarkdown, permitiendo intercalar secciones de código con explicaciones de los resultados obtenidos. Gran parte de las funciones y los filtrados utilizados se han obtenido gracias a la búsqueda en internet y la comunidad existente, necesaria en muchos casos en los que ha existido algún error de programación.

Por último, se ha hecho uso del repositorio *GitHub*, tanto para la descarga de los datos como para la entrega del trabajo. Al final de este documento se deja un enlace al repositorio del alumno con todos los documentos solicitados en el trabajo.

Resultados

Seleccionar un dataset

A la hora de seleccionar un dataset, se ha elegido uno del repositorio de github planteado en la actividad https://github.com/nutrimetabolomics/metaboData/. Esta decisión se ha tomado debido a la sencillez para acceder a los datos. Estos datos representan los metabotipos de respuesta a la cirugía bariátrica independientes de la magnitud de la pérdida de peso. Para poder acceder a estos datos se ha clonado el repositorio en el disco duro local mediante la creación de un nuevo projecto con control de versiones. Una vez que se han clonado los datos, se ha seleccionado la carpeta *2018-MetabotypingPaper*, dentro de la cual encontramos cuatro archivos. Los dos archivos que serán de interés son *DataInfo_S013.csv* y *DataValues_S013.csv*. El primero de ellos tiene los metadatos con la información de cada columna del segundo. El segundo archivo tiene los valores clínicos y metabolómicos para 39 pacientes en 5 puntos temporales distintos.

Creación del contenedor SummarizedExperiment

Una vez que se han seleccionado los datos, se creará un contenedor de tipo *SummarizedExperiment*. Esta clase se utiliza en diferentes herramientas bioinformáticas, por lo que será de gran utilidad en muchas ocasiones. Para poder crear este paquete es necesario importar las librerías indicadas a continuación:

```
# Cargar las librerías necesarias
library(readr)
library(dplyr)
library(xml2)
library(SummarizedExperiment)
library(rvest)
library(pheatmap)
library(ggplot2)
library(missMDA)
library(ggrepel)
```

Para llevar a cabo la creación del contenedor, se cargarán cada uno de los archivos. Por un lado, es necesario importar los valores de metabolómica en formato de matriz. En este caso, se han excluido las seis primmeras columnas, las cuales hacen referencia a metadatos relativos al paciente, el tipo de cirugía, la edad, el género o el grupo al que pertenecen. Dado que ahora tenemos la matriz traspuesta, el nombre de las columnas será el de los pacientes, por lo que estas se nombrarán con el identificador de *SUBJECT*, mientras que las filas se llamarán como las variables que venían expresadas en las columnas del dataframe original.

```
# Cargar los datos
# Leer el archivo de valores de datos y el archivo de información de datos
data_values <- read_csv("Datasets/2018-MetabotypingPaper/DataValues_S013.csv")
data_info <- read_csv("Datasets/2018-MetabotypingPaper/DataInfo_S013.csv")

# Excluir las columnas de metadatos y transponer el resto
expression_data <- as.matrix(data_values[, -(1:6)])
expression_data <- t(expression_data)  # Transponer para que los sujetos sean columnas

# Preparar los nombres de las dimensiones de expression_data
colnames(expression_data) <- data_values$SUBJECTS  # Asignar nombres de columna
rownames(expression_data) <- data_info$VarName[6:(nrow(expression_data)+5)]  # Asignar
↪  nombres de fila
```

El siguiente paso consiste en seleccionar los metadatos de las filas, aquellos que hacen referencia a las variables. En este caso, esta información se encuentra en nuestra variable *data_info*. De esta seleccionamos las dos últimas columnas, las cuales tienen información acerca del tipo de variable y una descripción de la misma. Aunque no es muy extensa y clarificadora, nos vale para los propósitos de este trabajo. De entre todas los valores, seleccionamos solo aquellos que tenemos en las filas de *data_values*.

```
# Preparar los metadatos de las filas (variables)
# Seleccionamos solo las columnas necesarias y establecemos el nombre de la variable como
↪  rownames
rowData <- data_info[, c("varTpe", "Description")]

# Asignar los nombres de fila en rowData utilizando VarName
rownames(rowData) <- data_info$VarName

# Alinear rowData con expression_data
rowData <- rowData[rownames(expression_data), , drop = FALSE]  # Mantener solo las
↪  variables presentes
rownames(rowData) <- rownames(expression_data)
```

Para el caso de las columnas, tenemos que seleccionar los valores correspondientes a los metadatos que hacen referencia a cada paciente. Estos son aquellos que se han excluido al principio del código. Finalmente, también se excluye la columna relativa a *SUBJECTS*, ya que esta aparece implicitamente en *data_values*, siendo el nombre de las columnas.

```
# Preparar los metadatos de las columnas (sujetos)
# Extraemos solo las primeras columnas de identificación
colData <- data_values[, c("SUBJECTS", "SURGERY", "AGE", "GENDER", "Group")]
rownames(colData) <- rowData$SUBJECTS  # Asignar nombres de filas
colData <- colData[, -1]  # Quitar la columna SUBJECTS
```

Finalmente, creamos el contenedor con los elementos creados anteriormente.

```
se <- SummarizedExperiment(assays = list(counts = expression_data),
                           rowData = rowData,
                           colData = colData)
```

Exploración de los datos

Resumen de la estructura del objeto

Lo primero que podemos hacer con nuestro objeto es estudiar su estructura. Aquí podemos ver que nuestro objeto está construido correctamente y concuerda con los datos que teníamos de entrada.

```
dim(se)
```

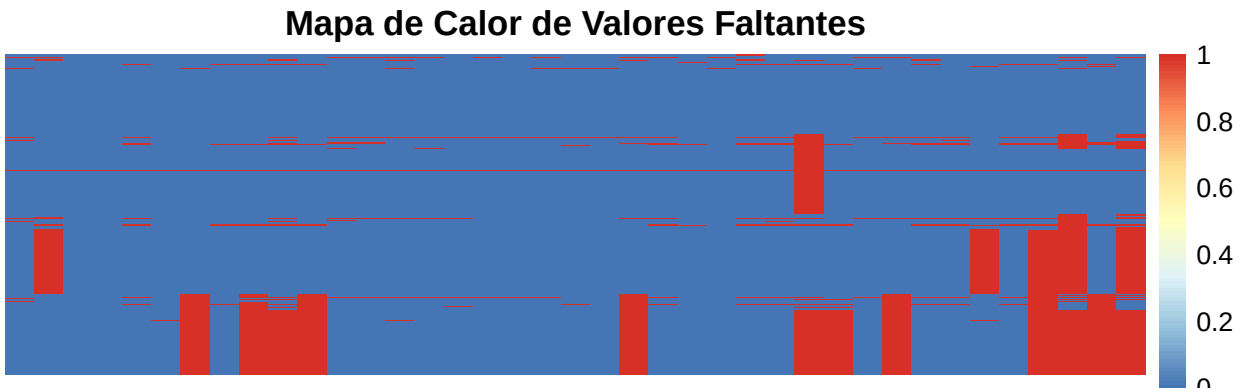
```
## [1] 690  39
```

Análisis de valores faltantes

De una manera bastante visual, podemos ver donde existen valores faltantes para cada uno de los pacientes y las variables. Como se ve en el mapa de calor, los mayores valores faltantes se acumulan en las últimas

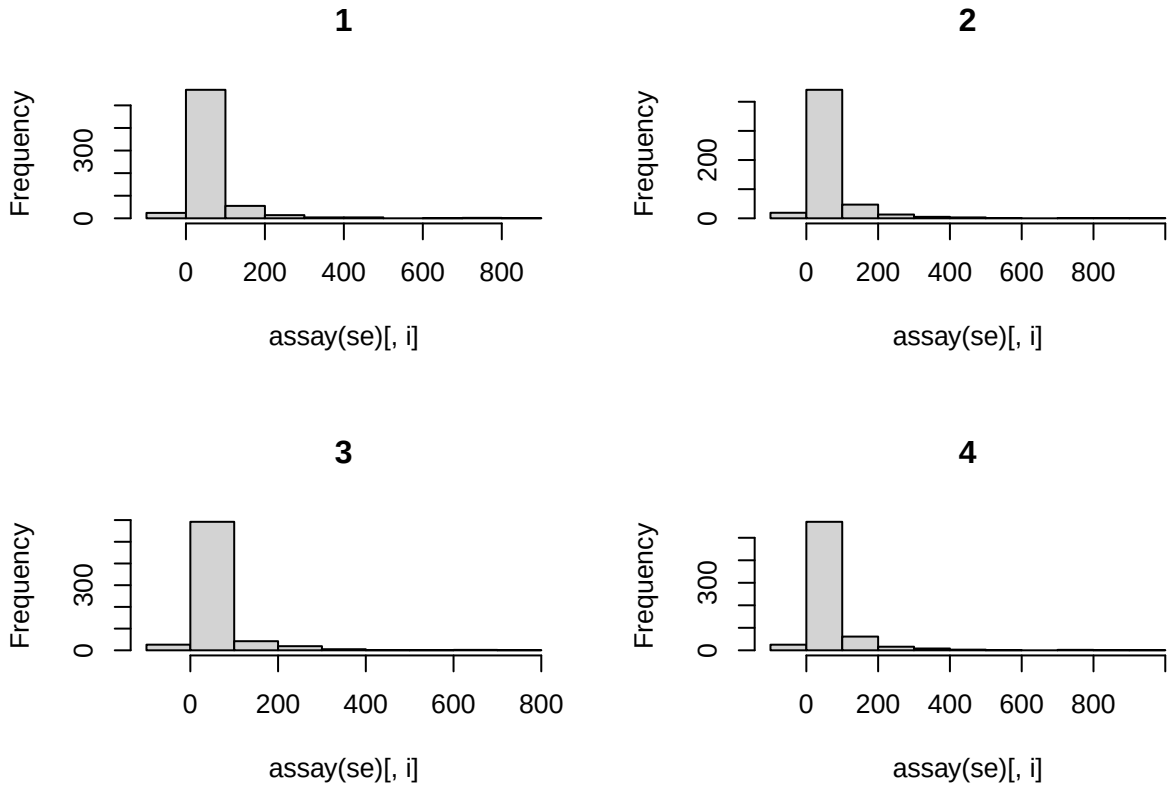
variables, dividido en dos bloques claramente distinguidos, ya que se corresponden con los momentos temporales en los que se han tomado las muestras. También existen valores sueltos, que no se pueden relacionar con ningún grupo a priori.

```
missing_data <- as.numeric(is.na(assay(se)))

# Crear el mapa de calor de los valores faltantes
pheatmap(matrix(missing_data, nrow = nrow(assay(se)), ncol = ncol(assay(se))),
         cluster_rows = FALSE, cluster_cols = FALSE,
         main = "Mapa de Calor de Valores Faltantes")
```



También se puede observar la distribución de cada los valores para cada uno de los pacientes. Podemos ver que todos los individuos siguen una distribución más o menos parecida, por lo que cada uno de ellos son comparables entre sí. Aunque aquí solo se presentan 4 gráficas, todas presentan una distribución parecida para cada una individuo.

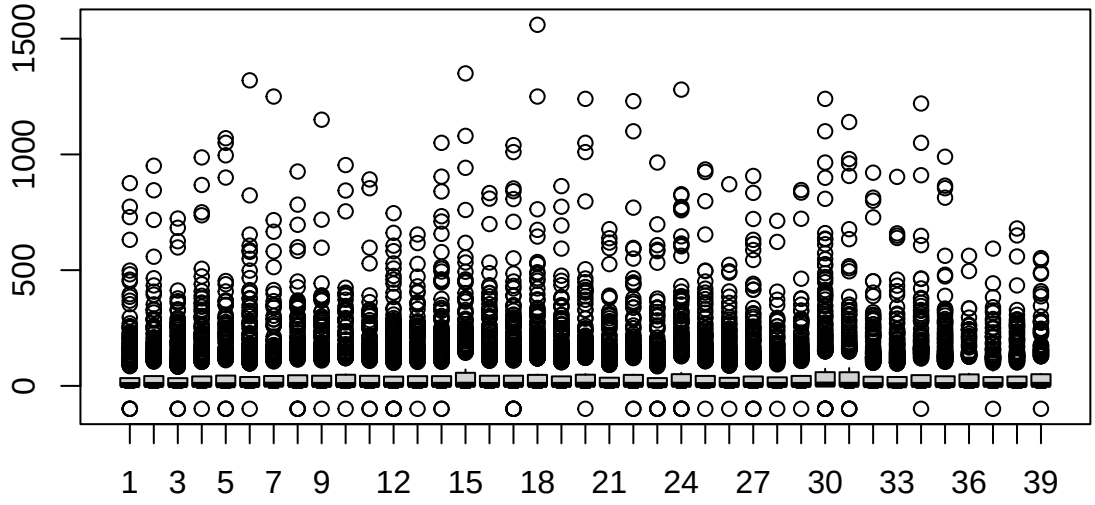
```
opt <- par(mfrow=c(2,2))
for (i in 1:4)
  hist(assay(se)[,i], main = names(assay(se)[1,][i]))
```




```
par(opt)
```

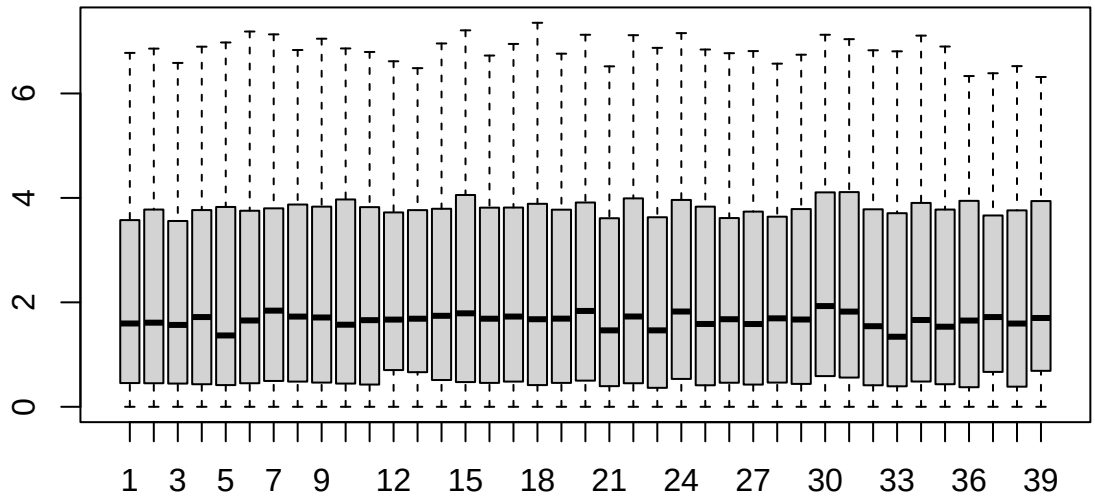
Si observamos el boxplot de los datos, vemos como existe una gran asimetría entre los datos, cosa que no habíamos sido capaces de observar mirando los histogramas.

```
boxplot(assay(se))
```



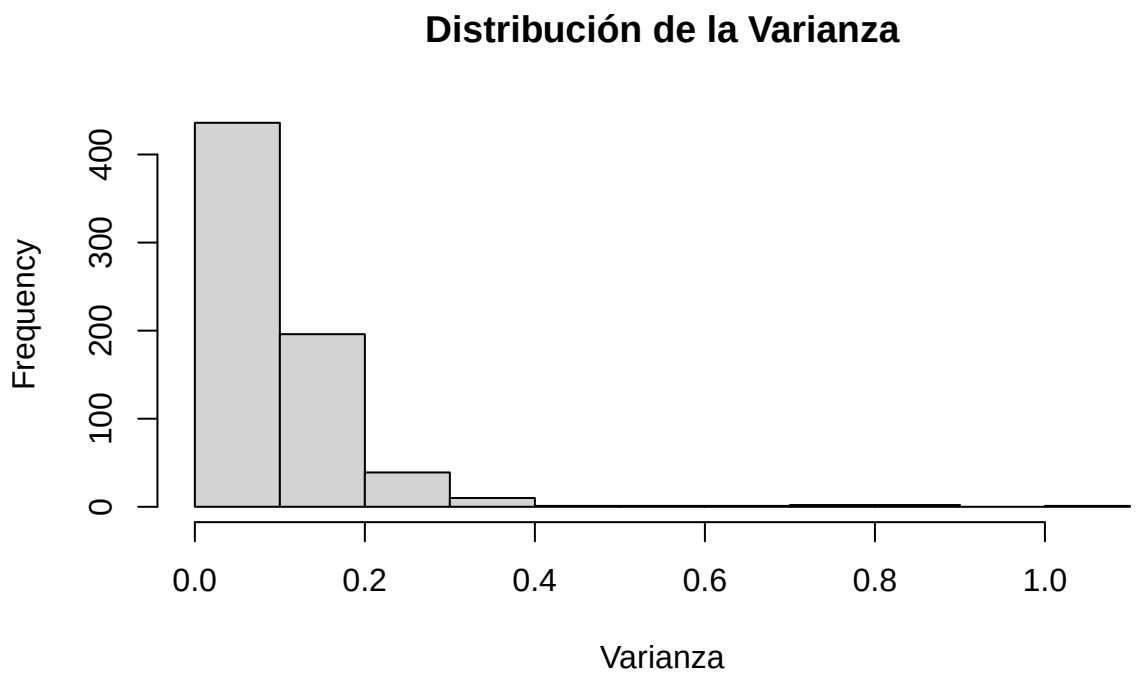
Para solucionar este problema, vamos a tomar logaritmos en los datos. Como vemos, tenemos un outlier negativo, lo que supondría un error. A pesar de todo ello, R nos permite calcular el logaritmo y excluir este dato, mostrando una distribución mucho más uniforme.

```
logX <- log(assay(se)+ 1)
boxplot(logX)
```



Tomando las varianzas de los logaritmos, vemos que la mayoría se concentran en torno al cero, lo que indica que la muestra presenta una distribución bastante homogénea.

```
variances <- apply(logX, 1, var, na.rm=TRUE)
hist(variances, main="Distribución de la Varianza", xlab="Varianza")
```

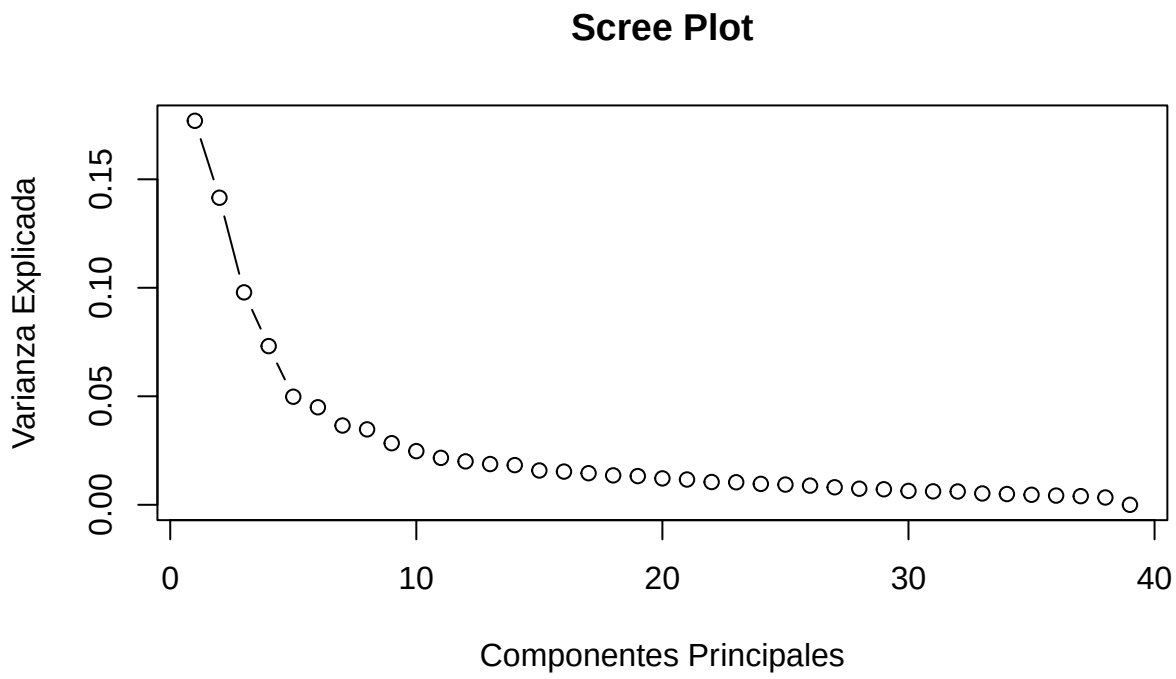


Para poder paliar el efecto de los valores se van a imputar utilizando una función que utiliza el cálculo de componentes principales para calcular de manera iterativa los valores faltantes.

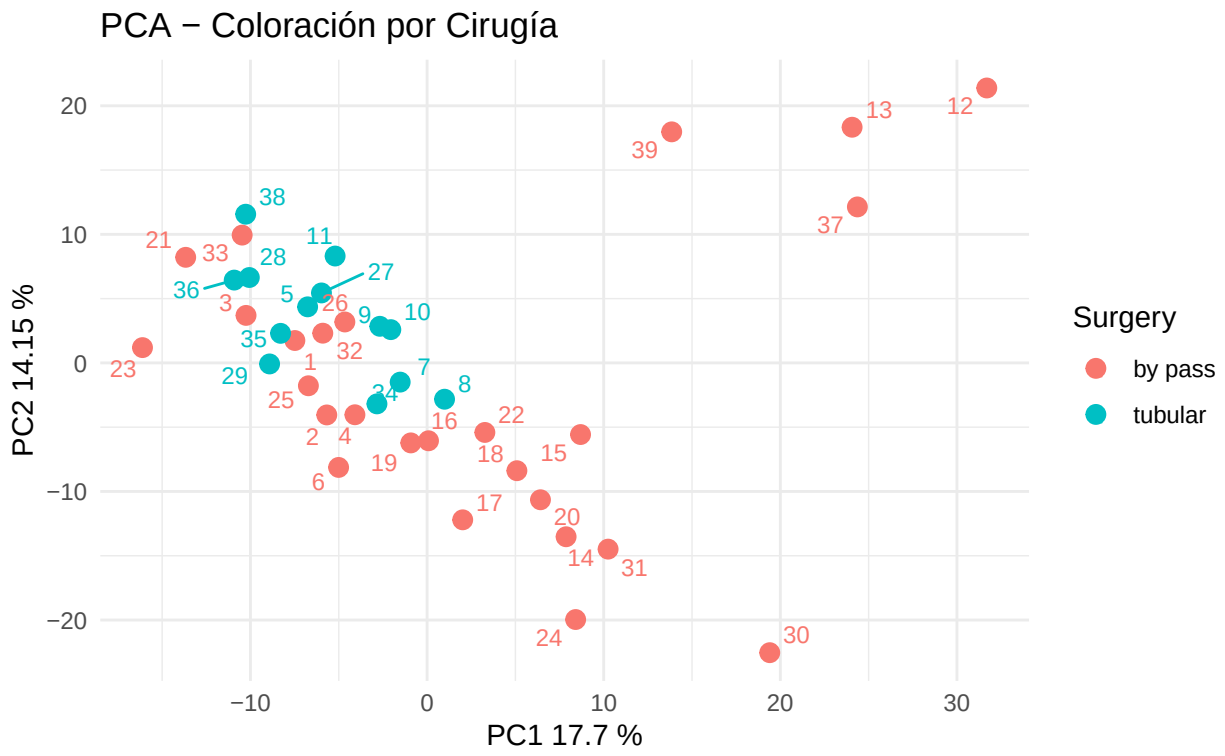
```
# Imputar los valores faltantes en el conjunto de datos
filled_data <- imputePCA(logX)$completeObs
any(is.na(filled_data))
```

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

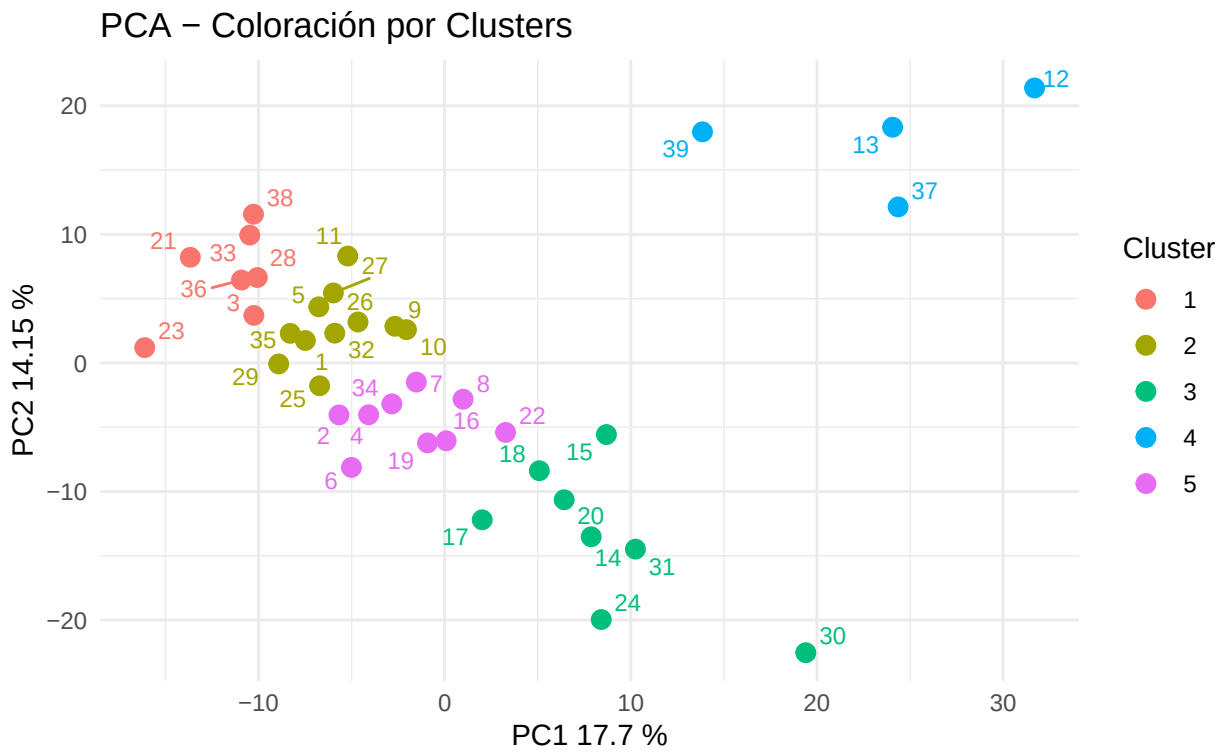
```
# Realizar el PCA con los datos completos
pca <- prcomp(t(filled_data), scale = TRUE)
varianza_explicada <- (pca$sdev)^2
loads <- round(varianza_explicada / sum(varianza_explicada) * 100, 2)
# Crear un Scree plot de la varianza explicada
explained_variance <- pca$sdev^2 / sum(pca$sdev^2)
plot(explained_variance, type = "b", xlab = "Componentes Principales",
     ylab = "Varianza Explicada", main = "Scree Plot")
```




```
xlab<-c(paste("PC1",loads[1],"%"))
ylab<-c(paste("PC2",loads[2],"%"))
pca_data <- as.data.frame(pca$x)
pca_data$Sample <- rownames(pca_data)

pca_data$Surgery <- colData(se)$SURGERY

ggplot(pca_data, aes(x = PC1, y = PC2, color = Surgery, label = Sample)) +
  geom_point(size = 3) +
  geom_text_repel(size = 3) +
  labs(x = xlab, y = ylab, title = "PCA - Coloración por Cirugía") +
  theme_minimal()
```

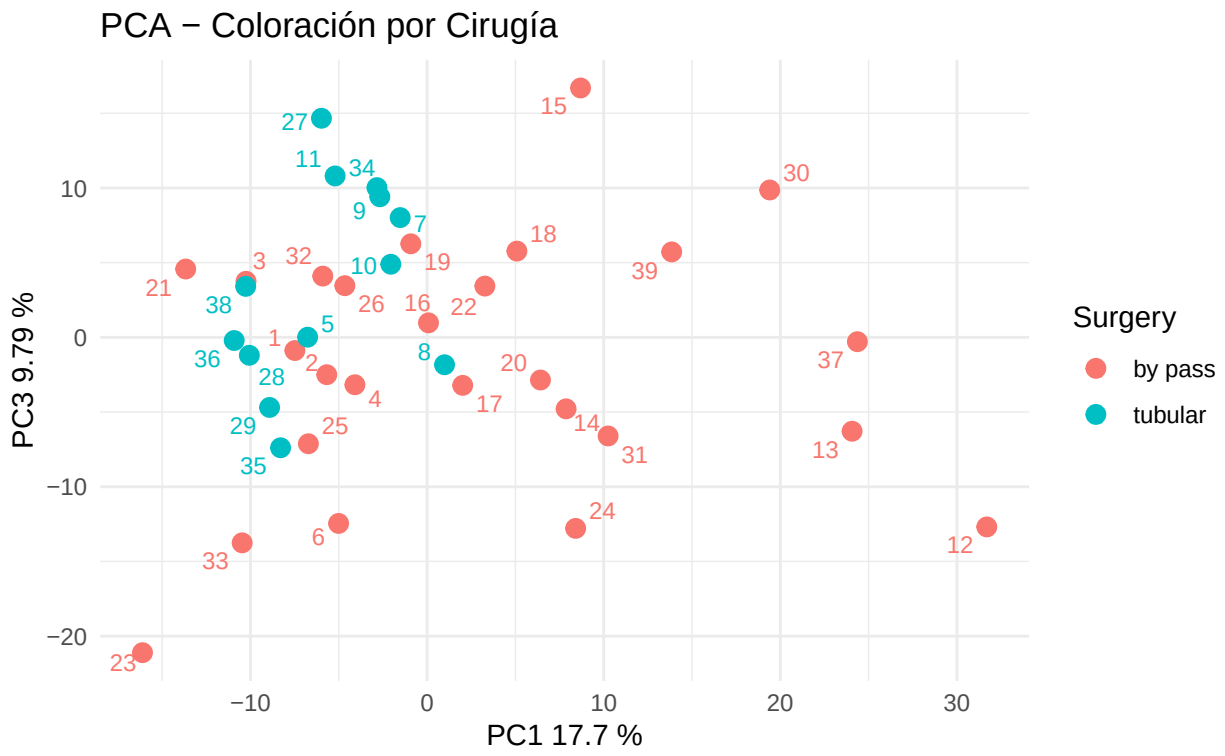



```
clusters <- kmeans(pca_data[, c("PC1", "PC2")], centers = 5)$cluster
pca_data$Cluster <- as.factor(clusters)


ggplot(pca_data, aes(x = PC1, y = PC2, color = Cluster, label = Sample)) +
  geom_point(size = 3) +
  geom_text_repel(size = 3) +
  labs(x = xlab, y = ylab, title = "PCA - Coloración por Clusters") +
  theme_minimal()
```

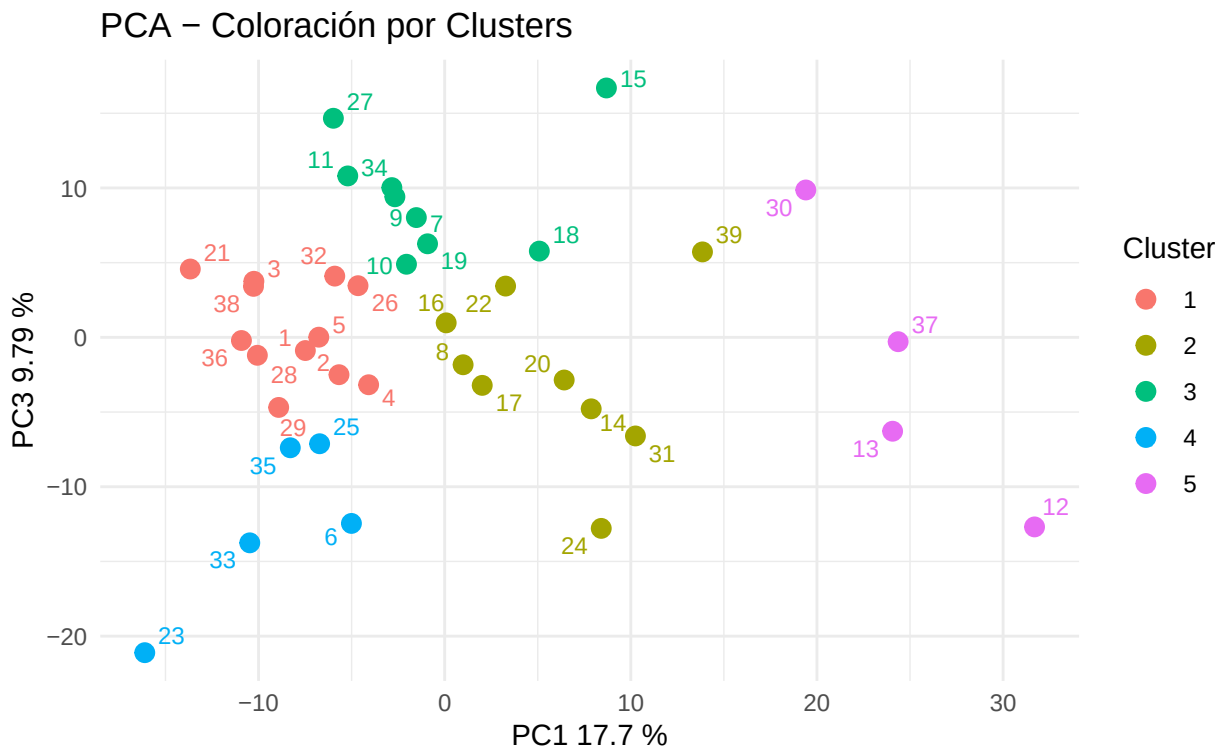


Como vemos en el análisis de PCA, ninguna de las componentes principales parece explicarse por ninguno de los metadatos correspondientes a las muestras. Si que existe un cierto valor límite a partir del cual solo encontramos un tipo de valores para la clasificación según *Surgery*. Por otro lado, se ha decidido hacer una representación según la clasificación k-means ($k = 5$) debido a los 5 momentos temporales en los que se han tomado las muestras. No existe ninguna manera de determinar si estos 5 grupos depende de estos momentos temporales, pero si parecen formarse 5 grupos bien diferenciados. Para comprobar si existen otras influencias, se ha probado el mismo análisis pero utilizando la PC3, la cual se ha visto que también tiene cierta relevancia.

```
xlab<-c(paste("PC1",loads[1],"%"))
ylab<-c(paste("PC3",loads[3],"%"))
pca_data <- as.data.frame(pca$x)
pca_data$Sample <- rownames(pca_data)

pca_data$Surgery <- colData(se)$SURGERY

ggplot(pca_data, aes(x = PC1, y = PC3, color = Surgery, label = Sample)) +
  geom_point(size = 3) +
  geom_text_repel(size = 3) +
  labs(x = xlab, y = ylab, title = "PCA - Coloración por Cirugía") +
  theme_minimal()
```



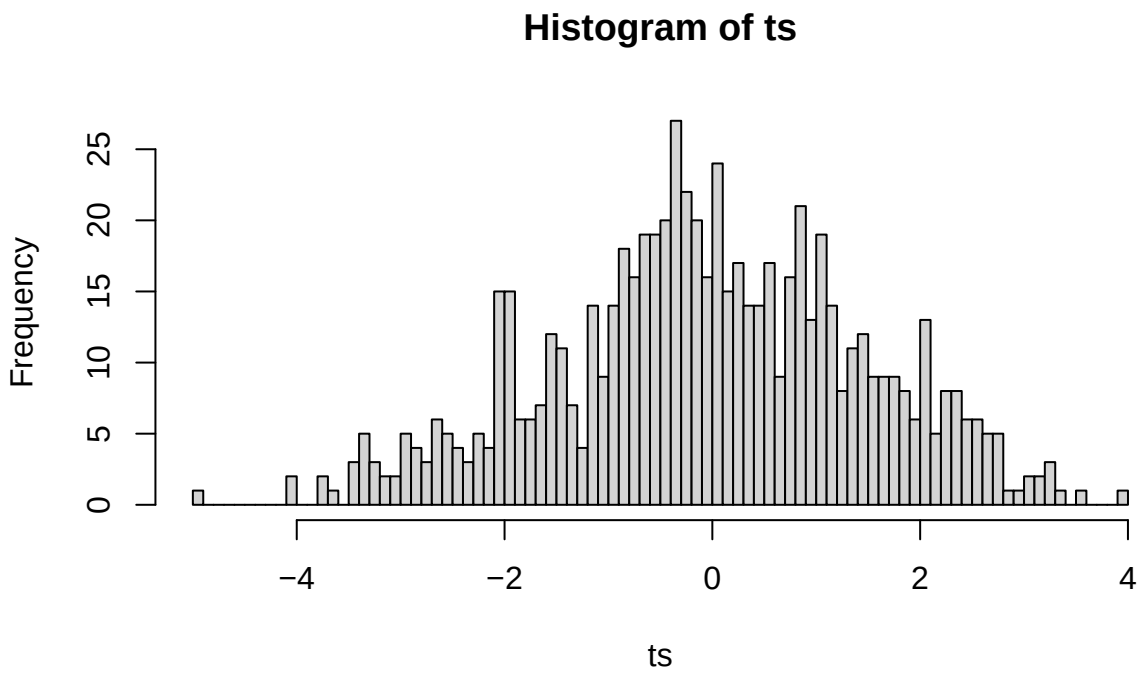

```
clusters <- kmeans(pca_data[, c("PC1", "PC3")], centers = 5)$cluster
pca_data$Cluster <- as.factor(clusters)


ggplot(pca_data, aes(x = PC1, y = PC3, color = Cluster, label = Sample)) +
  geom_point(size = 3) +
  geom_text_repel(size = 3) +
  labs(x = xlab, y = ylab, title = "PCA - Coloración por Clusters") +
  theme_minimal()
```



En este caso vemos como la PC3 tiene una cierta influencia según el metadato *Group*. Además, vemos como se mantiene la ligera diferenciación que habíamos observado para *Surgery* en cuanto a la PC1. En este caso también encontramos 5 grupos diferenciados, aunque no parecen tan claros como en el gráfico anterior. Además, se ve que los puntos son distintos, por lo que se puede descartar la hipótesis anterior.

Dado que se ha visto que existe una cierta influencia del grupo al que pertenecen, se ha decidido realizar la comparación tomando esto como medida de filtrado. Se entiende que no es la mejor debido a la naturaleza de los datos, pero a modo ilustrativo se muestra aquí como se realizaría el proceso. Existe un problema y es que no existe la misma cantidad de individuos en los dos grupos, por lo que el estudio puede ser erróneo en cuanto a resultados.

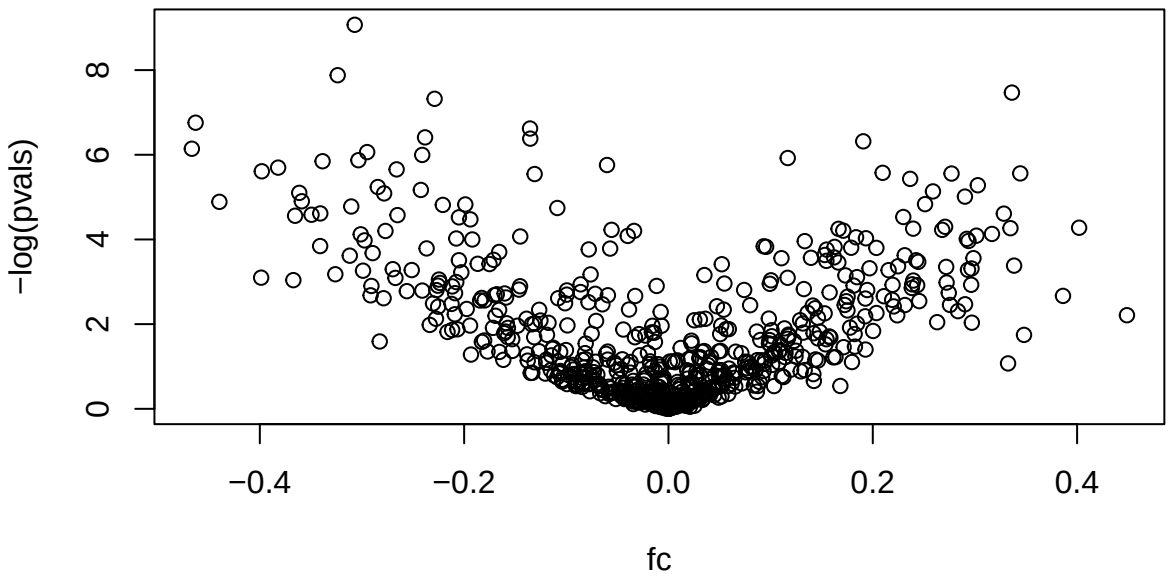
```
group_values <- colData(se)$Group

# Ordenar las columnas de filled_data según Group
filled_data <- filled_data[, order(group_values)]
ttest=function(x){tt=t.test(x[1:24], x[25:36])
return(c(tt$statistic,
         tt$p.value,
         tt$estimate[1]-tt$estimate[2]))
}

ans <- apply(filled_data,1,ttest)
ts <- ans[1,]
pvals<-ans[2,]
fc <-  ans[3,]
hist(ts, breaks=100)
```




```
for (i in c(0.01, 0.001, 0.0001, 0.00001, 0.000001, 0.0000001))
  print(paste("genes with p-values smaller than", i, length(which(pvals < i))))
```

```
## [1] "genes with p-values smaller than 0.01 40"
## [1] "genes with p-values smaller than 0.001 4"
## [1] "genes with p-values smaller than 1e-04 0"
## [1] "genes with p-values smaller than 1e-05 0"
## [1] "genes with p-values smaller than 1e-06 0"
## [1] "genes with p-values smaller than 1e-07 0"
```

```
plot(fc, -log(pvals))
```



Por un lado, vemos que los *t-values* siguen una distribución más o menos simétrica, cosa que tiene sentido debido a la normalización que hemos realizado sobre ellos. Esto concuerda con el gráfico visto anteriormente donde veíamos que la varianza era pequeña. Parece que existen algunos outliers, lo que puede estar provocado por el planteamiento del problema y el análisis, donde se están comparando grupos de tamaños distintos.

Por otro lado, vemos como el volcano plot muestra las variables up y down regulated así como su significancia. Si observamos la lista creada, vemos que realmente existen muy pocos p-values por debajo de 0.01, por lo que la mayoría de estos puntos se encontrarán entre 0.01 y 0.05 (-log(p-value) = 1.3).

Discusión y limitaciones y conclusiones del estudio

Las principales limitaciones que se encuentran en este estudio desde mi punto de vista, es que existen grandes bloques en los que no se han tomado datos para muchos pacientes. Estos se han tenido que imputar y puede dar lugar a modificaciones en los resultados. Aunque se trata de una solución bastante útil, podría no ser la más indicada. Sin embargo, se ha visto después de la imputación que los resultados siguen manteniendo su normalidad.

Por otro lado, ha sido bastante difícil diferenciar entre distintos grupos dentro del análisis de PCA. Para mi lo más lógico habría sido separar las variables y tratar de ver si existía algún efecto de batch. El problema está en que lo que se representa en este gráfico son las muestras, individuos en nuestro caso, y estas no llevaban asociadas marcas temporales, sino que el tiempo estaba indicado en las propias variables.

Por otro lado, considero que para poder realizar un análisis completo se requiere de mucho más tiempo para poder investigar a fondo los datos y todo lo que se puede hacer con ellos. Creo que el trabajo realizado es bastante bueno y aunque no se haya conseguido arrojar mucha luz sobre los datos y sus relaciones, creo que si que se ha conseguido el objetivo propuesto al principio de este documento, a pesar de que no se hayan cumplido en su totalidad.

En conclusión, considero que el estudio cumplió su objetivo principal: adquirir conocimientos en el uso de herramientas bioinformáticas para el análisis de datos ómicos. Aunque no se identificaron diferencias

significativas entre los grupos a un nivel concluyente, el análisis realizado aporta una base para continuar explorando estos datos con metodologías más complejas y dirigidas. El uso de técnicas adicionales, como la integración de datos temporales explícitos o el análisis de efectos batch, podría mejorar la interpretación y validez de los resultados en futuros estudios.

Repositorio git

Todos los documentos relatios a la PEC están subidos en el repositorio:

https://github.com/AndresMolinaRib/Molina-Ribagorda-Andres-PEC1