Informe PEC1. Análisis de Datos Ómicos.

Juan Manuel Sancho Romero

2024-10-29

Contents

1. ABSTRACT

En esta primera prueba de evaluación continua (PEC1) se integran los conocimientos y herramientas utilizadas hasta ahora en la asignatura. Para ello se ha organizado y desarrollado una simplificación del proceso de análisis de datos ómicos enfocado en el ámbito de la metabolómica.

En concreto, se ha seleccionado un dataset del repositorio de GitHub "nutrimetabolomics/metaboData" (https://github.com/nutrimetabolomics/metaboData/). Este dataset se corresponde a los datos utilizados

en el estudio "Metabotypes of response to bariatric surgery independent of the magnitude of weight loss" de Palau-Rodriguez et al. (2018).

Tras la clonación del repositorio mencionado, se procederá a cargar los datos en `R studio` y a continuación, crar un contenedor del tipo `SummarizedExperiment`. En este contenedor se almacenarán los datos y metadatos asociados al estudio proporcionados en el repositorio.

Posteriormente, se llevará a cabo una exploración del conjunto de datos para proporcionar una visión general del mismo y se exportará el objeto contenedor con los datos y metadatos en formato binario `.Rda`.

Finalmente, se entregarán los resultados de esta PEC1, mediante la creación de un repositorio de GitHub que guardará el presente informe, el objeto contenedor (`.Rda`), el código de R para la exploración de los datos, los datos en formato texto y los metadatos acerca del dataset en un archivo Rmarkdown.

2. OBJETIVOS DEL ESTUDIO

Los objetivos de este trabajo son:

(A) Familiarizarse con el proceso de análisis de datos metabolómicos, desde la selección del dataset y la comprensión de los datos hasta la comprensión y correcta interpretación de los resultados, así como de las limitaciones que puedan observarse durante el desarrollo de este trabajo.
(B) Integrar lo aprendido y desenvolverse adecuadamente en el uso de `R`, `Rmarkdown`, `BioConductor` y `Github`.
(C) Poner en práctica la creación y exploración de objetos de la clase `SummarizedExperiment`. Así como comprender el concepto, sus componentes e importancia en el campo del análisis de datos ómicos.

3. MATERIALES Y MÉTODOS

Los datos utilizados en este estudio fueron obtenidos del repositorio de GitHub "nutrimetabolomics/metaboData" (https://github.com/nutrimetabolomics/metaboData/), este conjunto de datos fue utilizado para el desarrollo del estudio de Palau-Rodriguez et al. (2018). En el se incluye información metabólica de 39 pacientes sometidos a cirugía bariátrica, así como los metadatos clínicos correspondientes a las variables y metabolitos analizados en el estudio. La naturaleza de los datos es de tipo metabolómico y clínico, permitiendo evaluar los cambios metabólicos en respuesta a una cirugía bariátrica.

En cuanto a las principales Herramientas Informáticas y Bioinformáticas, utilizadas para el análisis de los datos, se utilizaron:

- `R`: Lenguaje de programación y entorno para el análisis estadístico y gráfico.
- `Bioconductor`: Una colección de paquetes de software para análisis de datos biológicos y en concreto, el paquete `SummarizedExperiment`.
- `GitHub`: Plataforma utilizada para el almacenamiento y gestión de código y documentación del proyecto.

En cuanto a algunos de los métodos utilizados, en este trabajo se aborda la creación y manejo de objetos de la clase `SummarizedExperiment`con el objetivo de su exploración y el uso del Análisis de Componentes Principales (PCA), con el fin de entrever patrones dominantes o tendencias entre los metabolomas de los sujetos sometidos a una cirugía `by pass`, frente a los sometidos a una cirugía `tubular`.

4. RESULTADOS

4.1 SELECCIÓN DE UN DATASET

Primero accedemos al siguiente repositorio de `github`: https://github.com/nutrimetabolomics/metaboData/ Una vez dentro, seguimos el siguiente "path": Datasets/2018-MetabotypingPaper.

Aquí disponemos de 4 archivos, donde el archivo `description.md` aporta información sobre el contenido de los otros 3 archivos restantes. Este repositorio nos ofrece los datos utilizados en el paper "Metabotypes of response to bariatric surgery independent of the magnitude of weight loss" de Palau-Rodriguez et al. (2018).

Este estudio analiza las distintas respuestas metabólicas a dos tipos distintos de cirugía bariátrica (procedimientos quirúrgicos que se utilizan para tratar la obesidad y sus problemas de salud asociados) en personas con obesidad mórbida, centrándose en cambios en los metabolitos a corto plazo. Analizando metabolitos de muestras de suero tomadas antes de la cirugía y a los 1, 3 y 6 meses después, los investigadores identificaron dos "metabotipos" (fenotipos metabólicos) que mostraban respuestas distintas en términos de resistencia a la insulina, colesterol, lipoproteínas de baja densidad y ácido úrico. En conclusión, los resultados de este estudio enfatizaron el potencial de la cirugía bariátrica como un modificador metabólico, y ampliaron la comprensión de los beneficios metabólicos de esta intervención más allá de su papel en la reducción de peso.

4.2 CLONACIÓN DEL REPOSITORIO DE GITHUB

Primero clonamos el repositorio seleccionado desde github a R, siguiendo los mismos pasos que en la "Actividad 1.1. Introducción a las ómicas". Tras copiar el enlace del repositorio de Github, creamos un nuevo proyecto y clicamos "Version Control" < "Git". Añadimos el enlace del repositorio (https://github.com/nutrimetabolomics/metaboData.git) y un nombre al proyecto. Tras esto, ya tenemos disponibles los datos del estudio en el apartado "Files" de `R studio` y guardados en la carpeta "Analisis_Datos_Omicos".

Ahora vamos a importar los datos del estudio, donde "row.names = 1" especifica que la primera columna contiene los nombres (identificadores) de cada fila:

```
> # Valores clínicos y metabolómicos
> data_values <- read.csv("C:\\Users\\juanm\\Desktop\\MASTER_BIOINFORMATICA\\Analisis_Datos_Omicos\\UOC
>
> # Metadatos de columnas (de las variables)
> data_info <- read.csv("C:\\Users\\juanm\\Desktop\\MASTER_BIOINFORMATICA\\Analisis_Datos_Omicos\\UOC_m
>
> # Metadatos adicionales de los metabolitos
> aa_information <- read.csv("C:\\Users\\juanm\\Desktop\\MASTER_BIOINFORMATICA\\Analisis_Datos_Omicos\\U
```

4.3 CREACIÓN DE UN CONTENEDOR TIPO `SummarizedExperiment`

Observando los 3 dataframes, tenemos que el dataframe `aa_information` nos da información adicional para los metabolitos, que son variables relacionadas con la componente "colData" del objeto `SummarizedExperiment`. Los metadatos generales de los metabolitos junto a los de otras variables ya se encuentran incluidos en el dataframe `Data_Info_S013`, siendo la principal diferencia que en `aa_information` solo se muestra información adicional para los metabolitos ya incluidos en `Data_Info_S013`.

Por este motivo, incluiremos más adelante `aa_information` en una componente adicional del objeto `SummarizedExperiment` llamada `metadata`.

Como aclaración, también se ha observado que los metabolitos incluidos en `aa_information` presentan unas abreviaturas o nombres con distintas variaciones con respecto a como se mencionan a esos mismos metabolitos

en el dataframe `Data_Info_S013` por lo que no parece posible, o al menos sin que el resultado sea demasiado tedioso, unificar ambos metadatos para la componente "colData".

Además, debemos tener en cuenta que para este objeto `SummarizedExperiment`, por ahora no disponemos de la componente `rowData`, que incluiría metadatos para las filas (en este caso de cada sujeto o muestra). ***A priori*, estas serán las distintas componentes de nuestro objeto `SummarizedExperiment`.**

Antes de crear el contenedor, primero, cargamos la librería de `SummarizedExperiment`:

```
> library(SummarizedExperiment)
```

A continuación, corroboramos que las dimensiones de las distintas componentes de datos son compatibles entre sí:

```
> # Dimensiones de los datos del assay
> cat("Dimensiones de data_values (assays):", dim(data_values), "\n")
```

```
Dimensiones de data_values (assays): 39 695
```

```
> # Dimensiones de colData
> cat("Dimensiones de data_info (colData):", dim(data_info), "\n")
```

```
Dimensiones de data_info (colData): 695 3
```

```
> # Dimensiones de los metadatos de los metabolitos
> cat("Dimensiones de aa_information (metadatos$metabolitos):", dim(aa_information), "\n")
```

```
Dimensiones de aa_information (metadatos$metabolitos): 188 7
```

Si observamos el conjunto de datos de `data_values`, se puede comprobar que las primeras 9 columnas no se corresponden con datos de expresión de metabolitos u otras variables continuas, sino que, podríamos considerar que se tratan de *metadatos asociados a cada paciente (colData)*, como el tipo de cirugía aplicada, la edad, el género, tratamiento, etc. Además, en los experimentos de datos ómicos, las filas representan las variables o características y las columnas las muestras o sujetos, por lo que, previamente, habría que transponer el dataframe y el dataset `data_info` pasaría a ser el que aporte la componente `rowData`.

```
> head(data_values[, 1:9]) # Seleccionamos solo las 9 primeras columnas
```

SUBJECTS	SURGERY	AGE	GENDER	Group	MEDDM_T0	MEDCOL_T0	MEDINF_T0	MEDHTA_T0
1	by pass	27	F	1	0	0	0	1
2	by pass	19	F	2	0	0	0	0
3	by pass	42	F	1	0	0	0	0
4	by pass	37	F	2	0	0	0	0
5	tubular	42	F	1	0	0	0	0
6	by pass	24	F	2	0	0	0	0

Estos metadatos para cada sujeto, agrupados en las 9 primeras columnas, podemos considerarlos como un subconjunto de datos a añadir en la componente "colData" de nuestro objeto `SummarizedExperiment`. Para ello dividimos el dataset `data_values` en dos subconjuntos, uno que contenga los metadatos para las filas (`rowData`) y otro que contenga los datos de expresión de los distintos metabolitos medidos y el resto de variables que se corresponderá con la componente `assays`:

```
> library(dplyr)
> # Transponemos la matriz
> data_values <- as.data.frame(t(data_values))
>
> # Dataset de metadatos de la componente "colData" (primeras 9 columnas)
> metadatos_colData <- data_values[1:9, ]
>
> # Dataset de la componente assays (resto de las columnas)
> assay_Data <- as.data.frame(data_values[-c(1:9),])
> # Transformamos sus variables de nuevo a tipo "double"
> assay_Data <- assay_Data %>% mutate_if(is.character, as.numeric)
>
> # Comprobamos que los datasets se han dividido correctamente
> head(metadatos_colData[, 1:9]) # Seleccionamos los 9 primeros sujetos
```

	1	2	3	4	5	6	7	8	9
SUBJECTS	1	2	3	4	5	6	7	8	9
SURGERY	by pass	by pass	by pass	by pass	tubular	by pass	tubular	tubular	tubular
AGE	27	19	42	37	42	24	33	55	40
GENDER	F	F	F	F	F	F	F	F	F
Group	1	2	1	2	1	2	1	1	1
MEDDM_T0	0	0	0	0	0	0	0	0	0

```
> head(assay_Data[, 1:9])
```

	1	2	3	4	5	6	7	8	9
GLU_T0	85.0	78.00	75.00	71.00	82.00	71.00	80.00	90.00	92.00
INS_T0	11.4	12.10	8.41	12.80	6.01	9.88	9.20	3.40	5.43
HOMA_T0	2.4	2.32	1.56	2.25	1.22	1.73	1.82	0.76	1.23
HBA1C_T0	NA	NA	5.40	5.10	5.60	5.10	5.60	5.50	5.70
HBA1C.mmol.mol_T0	NA	NA	35.51	32.23	37.69	32.23	37.69	36.60	38.78
PESO_T0	151.0	139.00	84.00	136.00	121.00	148.00	109.00	109.00	114.00

Para que se mantengan las mismas dimensiones, debemos dividir también el subset `data_info` (colData). Los metadatos de las 9 primeras filas (variables) los añadiremos más adelante como metadatos adicionales al objeto `SummarizedExperiment`:

```
> # Dataset de metadatos adicionales para las primeras 9 filas
> metadatos_fila9_info <- data_info[1:9, ]
>
> # Dataset de colData (resto de las columnas)
> rowData_info <- data_info[-c(1:9), ]
>
> # Comprobamos los datasets
> head(metadatos_fila9_info)
```

	VarName	varTpe	Description
SUBJECTS	SUBJECTS	integer	dataDesc
SURGERY	SURGERY	character	dataDesc
AGE	AGE	integer	dataDesc
GENDER	GENDER	character	dataDesc
Group	Group	integer	dataDesc
MEDDM_T0	MEDDM_T0	integer	dataDesc

```
> head(rowData_info)
```

	VarName	varTpe	Description
GLU_T0	GLU_T0	integer	dataDesc
INS_T0	INS_T0	numeric	dataDesc
HOMA_T0	HOMA_T0	numeric	dataDesc
HBA1C_T0	HBA1C_T0	numeric	dataDesc
HBA1C.mmol.mol_T0	HBA1C.mmol.mol_T0	numeric	dataDesc
PESO_T0	PESO_T0	integer	dataDesc

Ahora ya está todo listo para crear el objeto de la clase `SummarizedExperiment`:

```
>                                                    # Matriz de datos
> se <- SummarizedExperiment(assays = list(counts = as.matrix(assay_Data)),
+                                                    # Metadatos de columnas (sujetos)
+                            colData = t(metadatos_colData),
+                                                    # Metadatos de filas (variables)
+                            rowData = as.data.frame(rowData_info))
>
> # Añadimos los metadatos adicionales
> metadata(se)$metabolitos <- aa_information
> metadata(se)$metadata_adicional <- metadatos_fila9_info
>
> # Visualizamos el objeto `SummarizedExperiment`
> print(se)
```

```
class: SummarizedExperiment
dim: 686 39
metadata(2): metabolitos metadata_adicional
assays(1): counts
rownames(686): GLU_T0 INS_T0 ... SM.C24.0_T5 SM.C24.1_T5
rowData names(3): VarName varTpe Description
colnames(39): 1 2 ... 38 39
colData names(9): SUBJECTS SURGERY ... MEDINF_T0 MEDHTA_T0
```

4.4 EXPLORACIÓN DEL DATASET

4.4.1 RESUMEN GENERAL DEL OBJETO ´SummarizedExperiment´

```
> summary(se)
```

```
[1] "SummarizedExperiment object of length 686 with 3 metadata columns"
```

Lo primero que podemos hacer es corroborar que se trata de un objeto de tipo `SummarizedExperiment` con 39 muestras de largo y 3 columnas de metadatos para la componente `rowData`.

Dimensiones del dataset (número de muestras (39) y el número de variables de cada muestra (695)):

```
> dim(se)
```

```
[1] 686  39
```

Tenemos un total de 39 filas, correspondientes a las muestras o sujetos de estudio y 695 variables entre las cuales tenemos diversidad de concentraciones de metabolitos obtenidos de una muestra de suero para cada paciente.

Nombres de las dimensiones del objeto se, vamos a acceder a las 10 primeras:

```
> dimnames(se[1:10, 1:10])
```

```
[[1]]
 [1] "GLU_T0"            "INS_T0"            "HOMA_T0"
 [4] "HBA1C_T0"          "HBA1C.mmol.mol_T0" "PESO_T0"
 [7] "bmi_T0"            "CC_T0"             "CINT_T0"
[10] "CAD_T0"

[[2]]
 [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10"
```

En la lista se observa primero el nombre de las filas (variables) y después el de las columnas (sujetos). Acceder a esta información puede ser relevante para identificar y trabajar con diferentes muestras o variables de interés.

Metadatos de los sujetos (`colData`):

```
> colData(se)
```

```
DataFrame with 39 rows and 9 columns
       SUBJECTS     SURGERY         AGE      GENDER       Group    MEDDM_T0
    <character> <character> <character> <character> <character> <character>
1              1     by pass          27           F           1           0
2              2     by pass          19           F           2           0
3              3     by pass          42           F           1           0
4              4     by pass          37           F           2           0
5              5     tubular          42           F           1           0
...          ...         ...         ...         ...         ...         ...
35            35     tubular          39           M           2           0
36            36     tubular          35           M           1           0
37            37     by pass          46           M           2           0
38            38     tubular          41           M           1           0
39            39     by pass          26           M           1           0
```

```
      MEDCOL_T0   MEDINF_T0   MEDHTA_T0
    <character> <character> <character>
1             0           0           1
2             0           0           0
3             0           0           0
4             0           0           0
5             0           0           0
...         ...         ...         ...
35            0           0           1
36            0           0           0
37            0           1           0
38            0           0           0
39            0           0           0
```

Aquí se observa información adicional de cada paciente relacionada con el tipo de cirugía aplicada, edad, género, grupo o tratamientos.

Metadatos de cada variable (`rowData`):

```
> rowData(se)
```

```
DataFrame with 686 rows and 3 columns
                           VarName      varTpe Description
                       <character> <character> <character>
GLU_T0                       GLU_T0     integer    dataDesc
INS_T0                       INS_T0     numeric    dataDesc
HOMA_T0                     HOMA_T0     numeric    dataDesc
HBA1C_T0                   HBA1C_T0     numeric    dataDesc
HBA1C.mmol.mol_T0 HBA1C.mmol.mol_T0     numeric    dataDesc
...                             ...         ...         ...
SM.C18.0_T5             SM.C18.0_T5     numeric    dataDesc
SM.C18.1_T5             SM.C18.1_T5     numeric    dataDesc
SM.C20.2_T5             SM.C20.2_T5     numeric    dataDesc
SM.C24.0_T5             SM.C24.0_T5     numeric    dataDesc
SM.C24.1_T5             SM.C24.1_T5     numeric    dataDesc
```

Aquí se pueden observar los 686 variables (filas) y sus correspondientes metadatos agrupados en 3 columnas (`VarName`, `varTpe`, `Description`).

Metadatos adicionales para cada metabolito (Accedemos a los 6 primeros):

```
> head(metadata(se)$metabolitos)
```

X.1	Class	Metabolite.abbreviation	Metabolite	Platform	Data.type	X
1	aminoacids	Ala	Alanine	LC-MS/MS	Quantified	NA
2	aminoacids	Arg	Arginine	LC-MS/MS	Quantified	NA
3	aminoacids	Asn	Asparagine	LC-MS/MS	Quantified	NA
4	aminoacids	Asp	Aspartate	LC-MS/MS	Quantified	NA

X.1	Class	Metabolite.abbreviation	Metabolite	Platform	Data.type	X
5	aminoacids	Cit	Citrulline	LC-MS/MS	Quantified	NA
6	aminoacids	Gln	Glutamine	LC-MS/MS	Quantified	NA

Metadatos adicionales para las variables de `colData`

```
> metadata(se)$metadata_adicional
```

	VarName	varTpe	Description
SUBJECTS	SUBJECTS	integer	dataDesc
SURGERY	SURGERY	character	dataDesc
AGE	AGE	integer	dataDesc
GENDER	GENDER	character	dataDesc
Group	Group	integer	dataDesc
MEDDM_T0	MEDDM_T0	integer	dataDesc
MEDCOL_T0	MEDCOL_T0	integer	dataDesc
MEDINF_T0	MEDINF_T0	integer	dataDesc
MEDHTA_T0	MEDHTA_T0	integer	dataDesc

Acceso a la matriz de datos (39 sujetos y 695 variables) y **mostrar solo los datos de los sujetos o pacientes deseados**:

```
> # Dimensiones de la Matriz
> dim(se)
```

```
[1] 686  39
```

```
> # Datos de las primeras 10 variables de los tres primeros pacientes
> assay(se[1:10, 1:3])
```

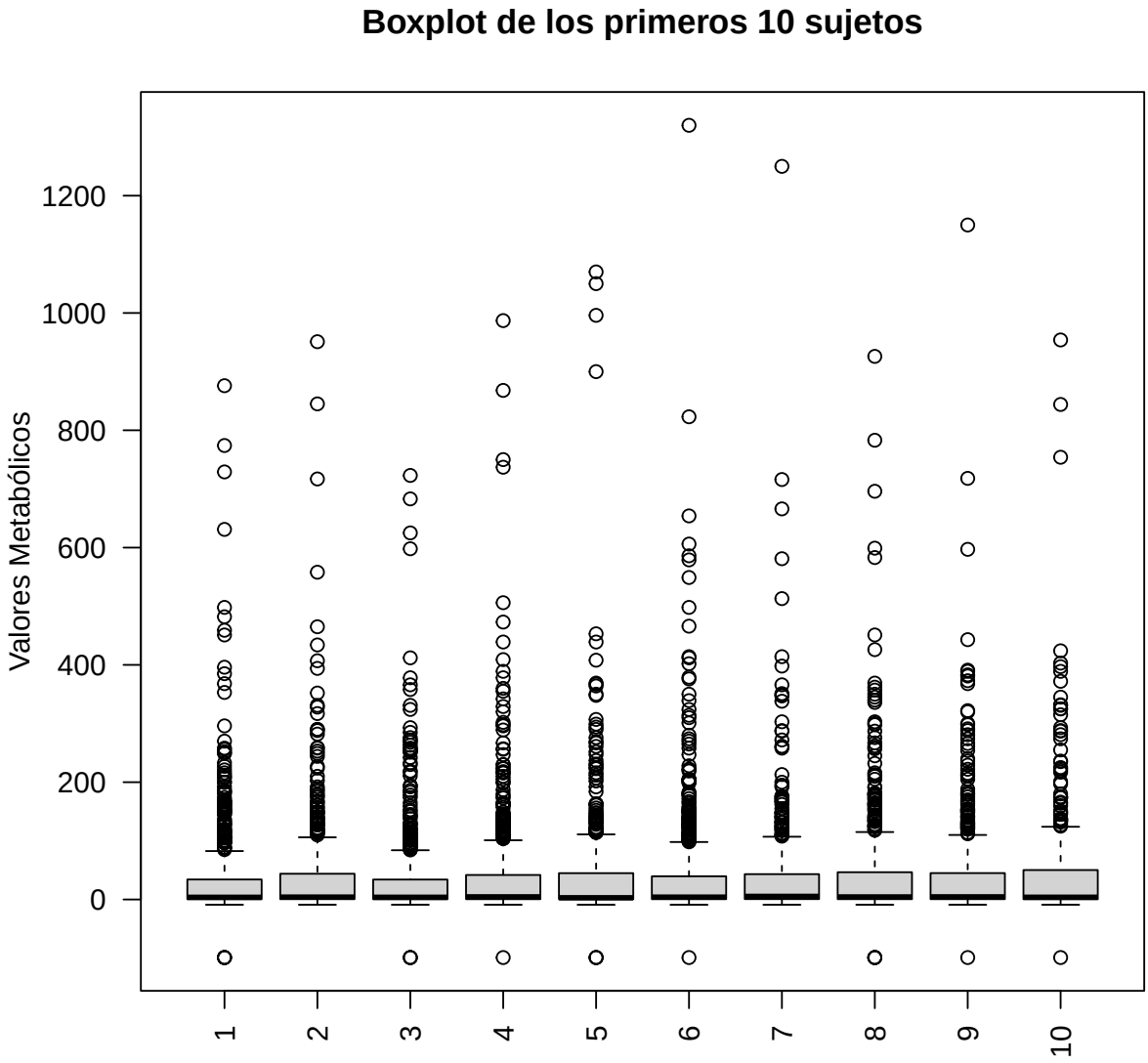
	1	2	3
GLU_T0	85.0	78.00	75.00
INS_T0	11.4	12.10	8.41
HOMA_T0	2.4	2.32	1.56
HBA1C_T0	NA	NA	5.40
HBA1C.mmol.mol_T0	NA	NA	35.51
PESO_T0	151.0	139.00	84.00
bmi_T0	62.9	47.00	29.80
CC_T0	0.7	NA	0.70
CINT_T0	116.0	NA	90.00
CAD_T0	167.0	NA	126.00

```
> # Datos de las primeras 5 variables del segundo paciente
> assay(se[1:5, 2])
```

	2
GLU_T0	78.00
INS_T0	12.10
HOMA_T0	2.32
HBA1C_T0	NA
HBA1C.mmol.mol_T0	NA

Gráfico tipo Boxplot de los 10 primeros sujetos

```
> boxplot(assay(se, "counts")[, 1:10],
+         main = "Boxplot de los primeros 10 sujetos",
+         ylab = "Valores Metabólicos", las = 2)
```



A priori, se observa que los valores del dataset no se encuentran normalizados.

4.4.2 PRE-PROCESADO DE DATOS CON PAQUETE `POMA`

```
> library("POMA")
> library("Rcpp")
> library("ggtext")
> library("magrittr")
> library("dplyr")
```

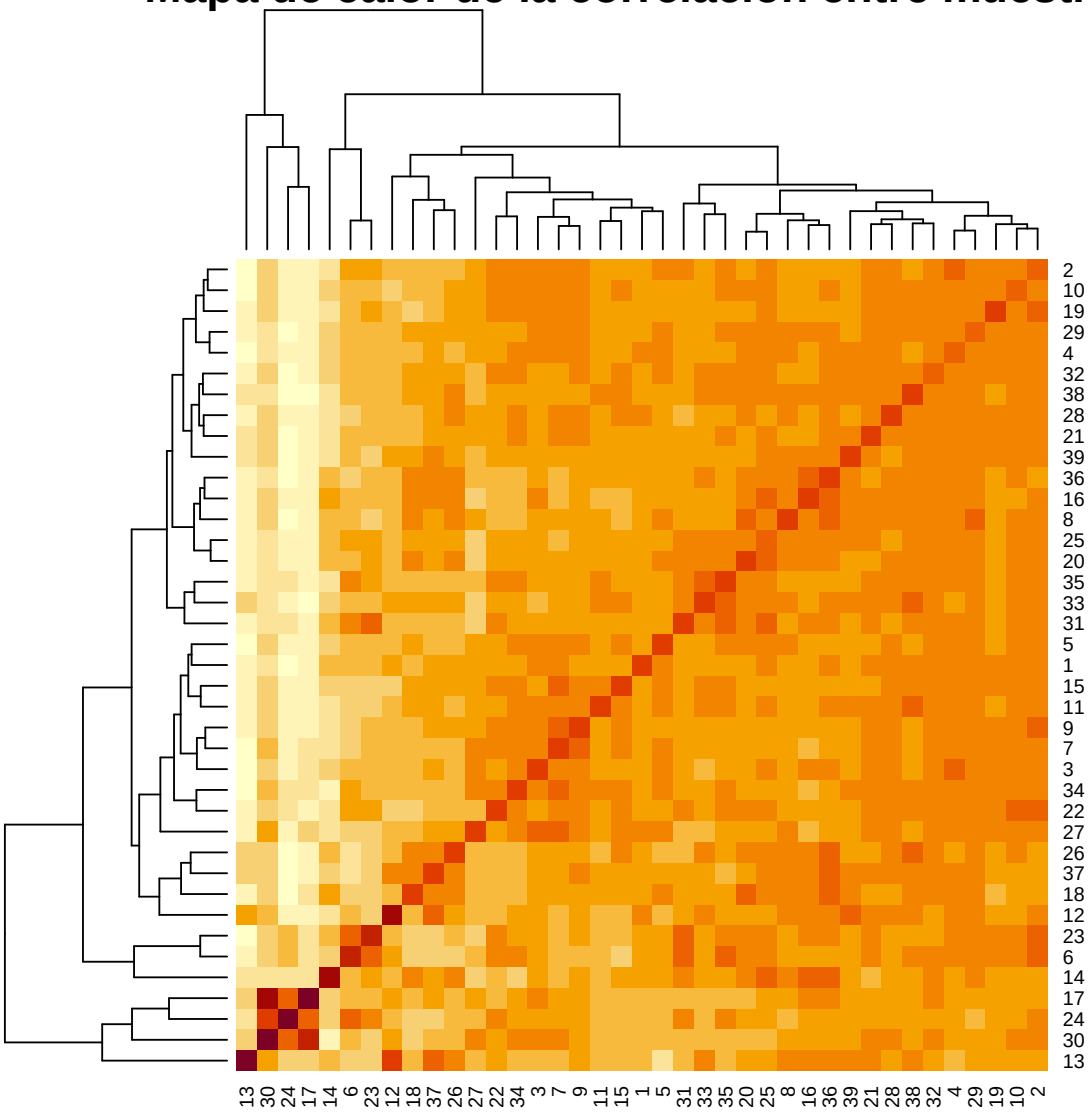
Aunque quizás el método de imputación de los datos más idóneo sea el de k-vecinos, seleccionando previamente el k óptimo mediante validación cruzada, para simplificar el problema utilizaremos el método "median" que sustituye los valores faltantes (NA) por sus respectivas medianas. Se han consultado los pasos del tutorial sobre el uso de `POMA` en https://www.bioconductor.org/packages/release/bioc/vignettes/POMA/inst/doc/POMA-workflow.html#the-poma-workflow.

Imputación de Valores Faltantes (NA):

```
> se_imputado <- se %>% PomaImpute(method = "median",
+                                  zeros_as_na = FALSE,
+                                  remove_na = TRUE,
+                                  cutoff = 60)
```

Matriz de correlación entre muestras

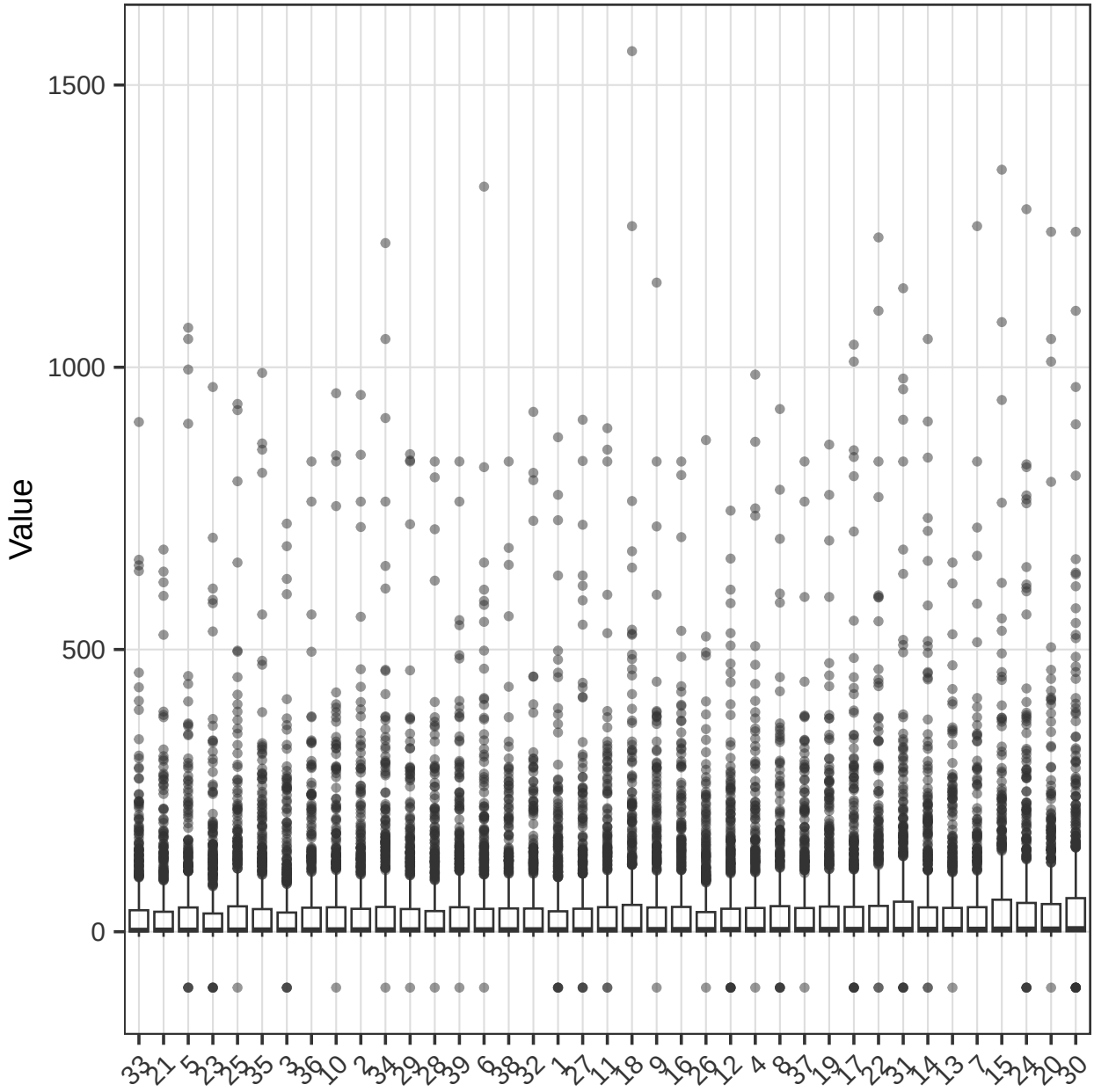
```
> correlation_matrix <- cor(assay(se_imputado))
> heatmap(correlation_matrix,
+         main = "Mapa de calor de la correlación entre muestras")
```



El resultado es un mapa de calor creado a partir de la matriz de correlación entre muestras de la componente "assay", que da una visión general de las relaciones de similitud entre las distintas muestras. De este modo se puede pre-visualizar los sujetos más correlacionados entre sí en función de la intensidad del color naranja.

Boxplot de todas las variables usando `POMA`

```
> PomaBoxplots(se_imputado,
+              x = "samples")
```



De nuevo, corroboramos que los datos no se encuentran normalizados y que hay algunas observaciones con valores menores a 0.

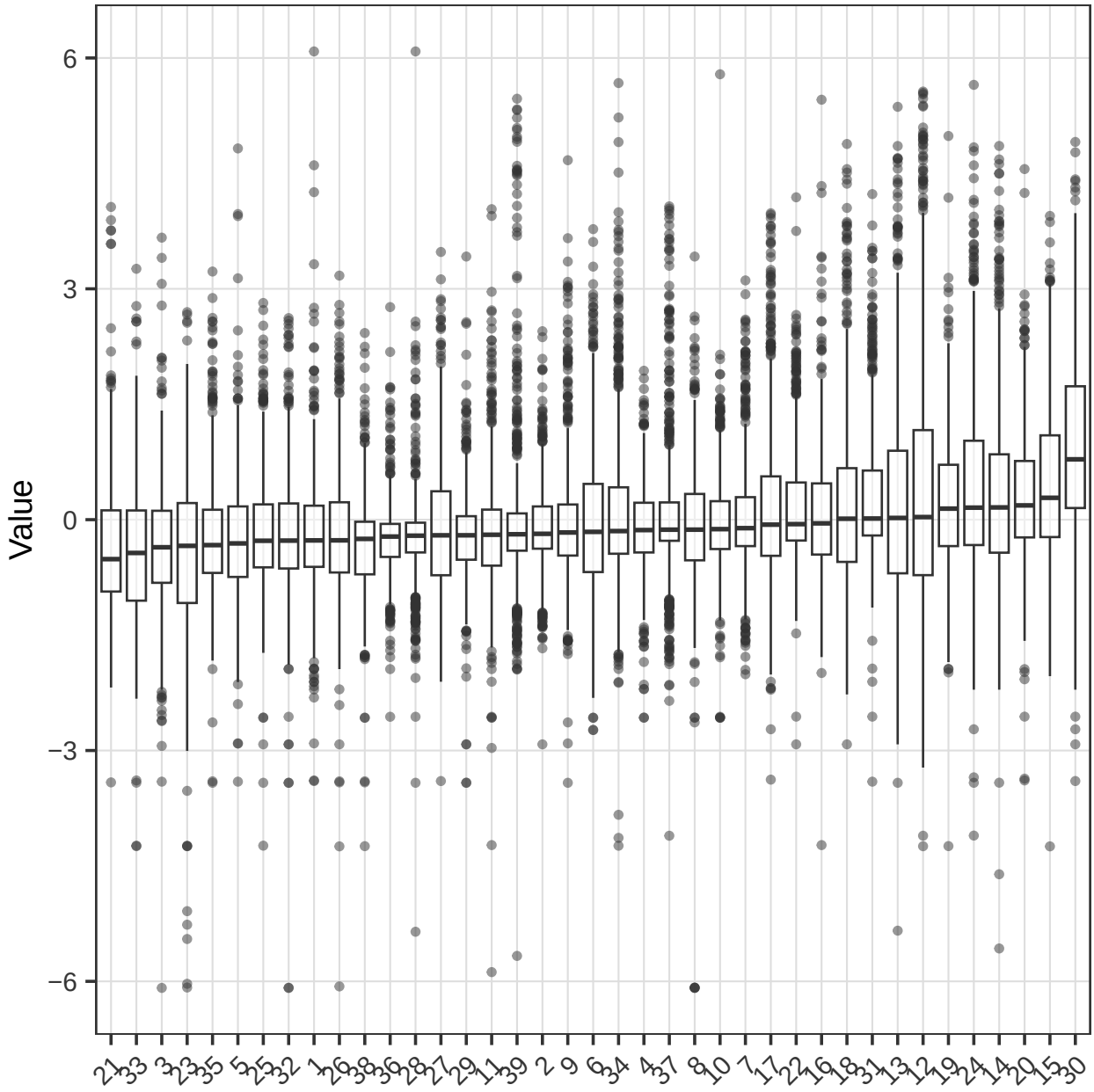
Normalización de los datos:

```
> se_normal <- se_imputado %>%
+   PomaNorm(method = "auto_scaling")
```

Se ha utilizado el **método "auto_scaling"**. Este método escala cada variable para que tenga una media de 0 y una desviación estándar de 1. Es de utilidad cuando se quiere dar el mismo peso a todas las variables, independientemente de sus escalas originales, y es común de cara a realizar análisis de componentes principales (PCA). (Consulta: https://rdrr.io/bioc/POMA/man/PomaNorm.html).

Efecto de la Normalización:

```
> PomaBoxplots(se_normal,
+              x = "samples")
```



4.4.3 ANÁLISIS DE COMPONENTES PRINCIPALES

Vamos a realizar un análisis de componentes principales muy simplificado, suponiendo que estamos interesados en analizar cómo las diferentes condiciones de cirugía ("SURGERY" en colData) afectan a los perfiles de expresión de los metabolitos de cada sujeto.

```
> library(ggplot2)
```

A continuación, es necesario eliminar observaciones anómalas de signo negativo. Vamos a proceder de manera muy simplificada a eliminar estas observaciones. Sabemos que son anómalas porque en el estudio Palau-

Rodriguez et al. (2018) hace mención a que en el pre-procesado de datos han excluido las variables con alta variabilidad analítica (con un coeficiente de variación (CV) > 25%). Si observamos las tablas S5* y S6* del estudio, se concluye que tras este procesado no hay ninguna concentración de ningún metabolito que presente un valor negativo. Por este motivo, simplificando enormemente el proceso y asumiendo mayor margen de error con el fin de ser más prácticos de cara a los objetivos reales de esta PEC, vamos a eliminar estas observaciones del objeto `se_imputado`.

S5 Table. Concentration of metabolites of metabolically healthy (MH) and unhealthy (MU) individuals after surgery S6 Table. Changes in metabolite concentrations of both clusters after surgery.

Reemplazo de valores negativos en el assay del objeto `se_imputado` por 0

```
> # Reemplazamos valores negativos en el assay del objeto `se_imputado` por 0
> assay(se_imputado)[assay(se_imputado) < 0] <- 0
>
> # Comprobamos que ya no hay observaciones menores que 0
> (negative_rows <- rownames(assay(se_imputado))[rowSums(assay(se_imputado) < 0) > 0])
```

```
character(0)
```

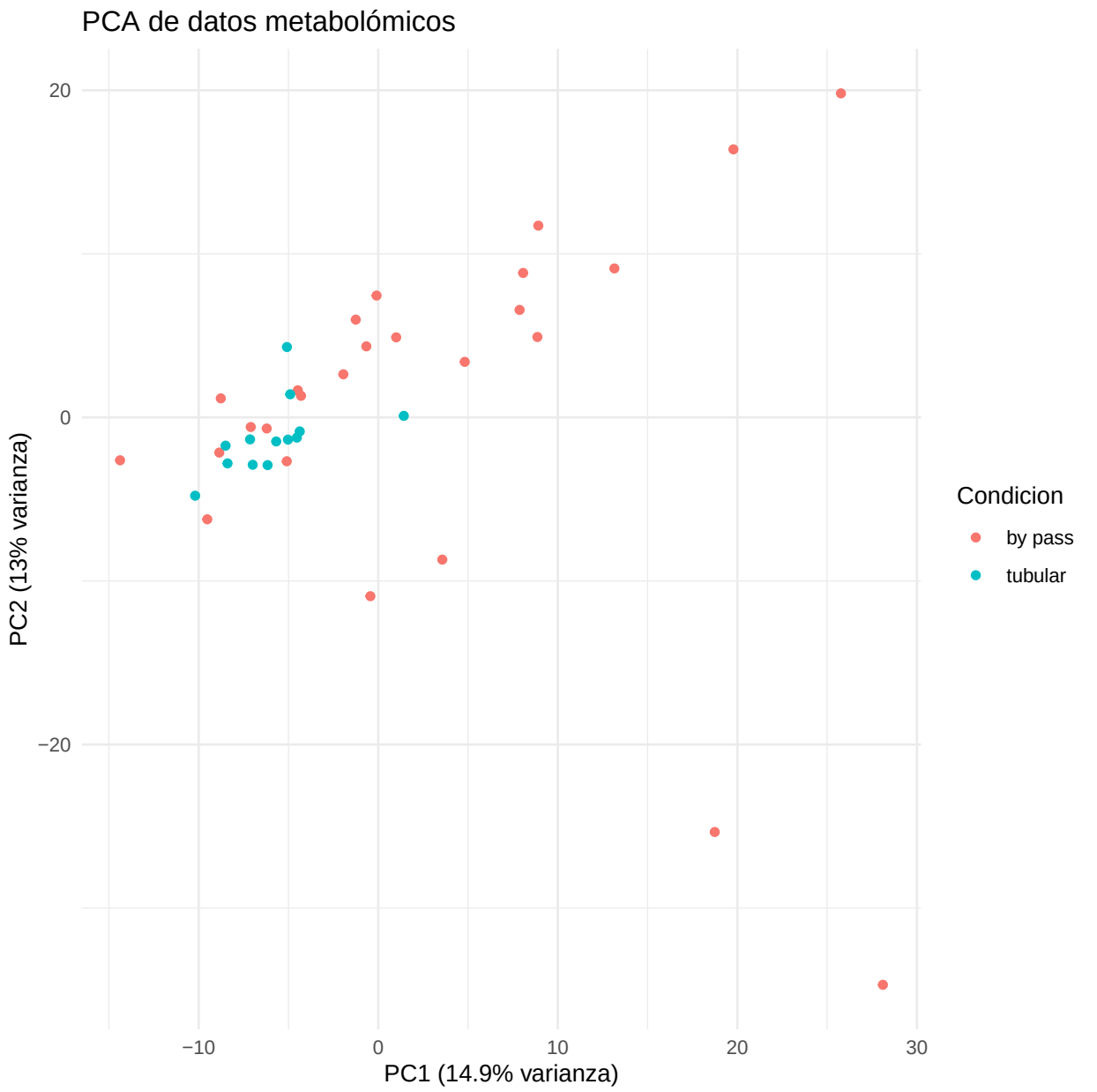
Normalizamos y realizamos PCA

```
> # Normalizamos los datos
> se_normal <- se_imputado %>%
+   PomaNorm(method = "auto_scaling")
>
> # Extraemos la matriz del assay (datos metabolómicos)
> se_matrix <- t(assay(se_normal)) # Trasponemos para poder graficar por "SURGERY"
>
> # Realizamos el PCA
> pca_result <- prcomp(se_matrix, center = TRUE, scale. = TRUE)
```

La función `prcomp()` calcula las componentes principales mediante la descomposición en valores singulares de la matriz de datos..

Gráfico de las 2 Primeras Componentes Principales

```
> library(ggplot2)
>
> # Convertimos los resultados del PCA a un data frame para poder graficarlos
> pca_data <- as.data.frame(pca_result$x) # Contiene las componentes principales
> # Añadimos la condición para colorear según tipo de cirugía
> pca_data$Condicion <- colData(se)$SURGERY
>
> # Calculamos la proporción de varianza explicada por cada CP
> varianza_explicada <- pca_result$sdev^2 / sum(pca_result$sdev^2)
> # Convertimos a porcentaje
> porcent_var_expl <- round(varianza_explicada * 100, 1)
>
> # Graficamos las primeras dos componentes principales
> ggplot(pca_data, aes(x = PC1, y = PC2, color = Condicion)) +
+   geom_point() +
+   labs(title = "PCA de datos metabolómicos",
+        x = paste0("PC1 (", porcent_var_expl[1], "% varianza)"),
+        y = paste0("PC2 (", porcent_var_expl[2], "% varianza)")) +
+   theme_minimal()
```



Aunque algunos de los sujetos de cada grupo están superpuestos, parece que una proporción significativa de los sujetos operados por el método "by pass" se encuentran formando un cluster propio, lo que puede sugerir la presencia de diferencias significativas en los perfiles metabólicos de estos sujetos. Al menos, se abre la posibilidad de considerar la relevancia de un estudio más exhaustivo al respecto, como el realizado en Palau-Rodriguez et al. (2018), donde se pretende obtener conclusiones significativas a cerca de esta hipótesis, entre otras.

Varianza explicada por cada una de las 21 Componentes

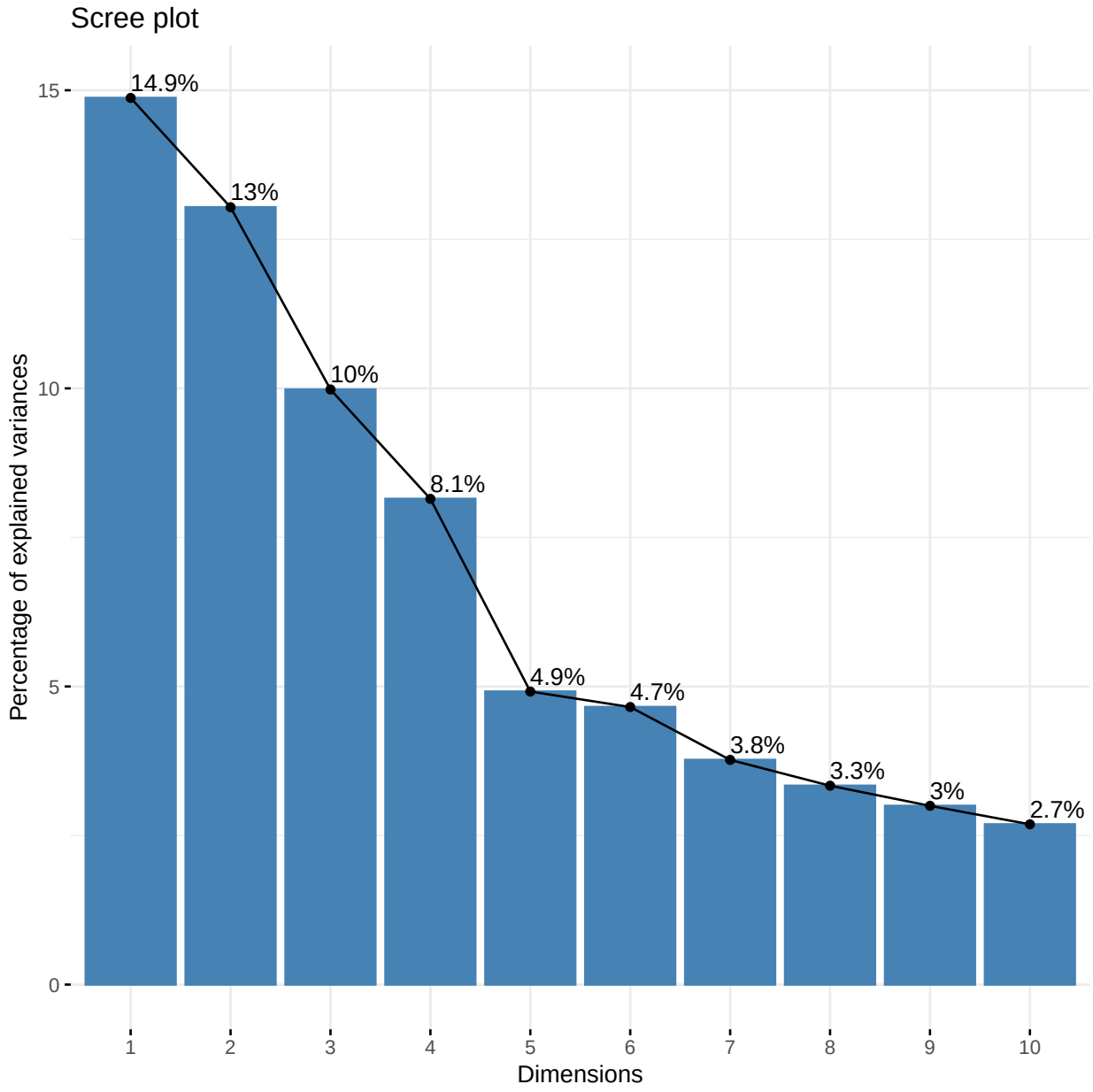
```
> porcent_var_expl
```

```
 [1] 14.9 13.0 10.0  8.1  4.9  4.7  3.8  3.3  3.0  2.7  2.3  2.2  2.1  1.9  1.9
[16]  1.7  1.6  1.6  1.4  1.4  1.2  1.2  1.1  1.0  1.0  0.9  0.8  0.8  0.7  0.7
[31]  0.6  0.6  0.6  0.5  0.5  0.4  0.4  0.3  0.0
```

Las dos primeras componentes principales (CP) explican un 27.9 de la varianza total. Las 9 primeras CP parecen explicar la mayor parte de la variabilidad de los datos. A continuación, observamos esto con mayor detalle usando un "Scree Plot".

Gráfico de Codo o Scree plot usando los paquetes `FactoMineR` y `factoextra`

```
> library("FactoMineR")
> library("factoextra")
> se_pca <- PCA(se_matrix, graph = FALSE)
> fviz_eig(se_pca, addlabels = TRUE, ylim = c(0, 15))
```



El "codo" en el gráfico, es donde la varianza explicada comienza a estabilizarse y los componentes adicionales no aportan una cantidad de variabilidad significativa. Según el gráfico podríamos decir que esto ocurre a partir de la sexta CP.

Fuente: http://www.sthda.com/english/articles/31-principal-component-methods-in-r-practical-guide/112-pca-principal-component-analysis-essentials/#visualization-and-interpretation https://aspteaching.github.io/AMVCasos/

4.5 EXPORTACIÓN DEL OBJETO CONTENEDOR

Exportamos el objeto contenedor original con los datos y metadatos (en formato `.Rda`), sin normalizar ni imputar los datos

```
> save(se, file = "C:/Users/juanm/Desktop/MASTER_BIOINFORMATICA/Analisis_Datos_Omicos/ADO_PEC1/datos_se
```

4.6 EXPORTACIÓN DEL CÓDIGO DE R (DOCUMENTO `.R`)

Extraemos el código R del archivo `R Markdown`, creando un archivo `.R` que solo contiene el código sin el texto explicativo.

```
> library(knitr)
> purl("C:/Users/juanm/Desktop/MASTER_BIOINFORMATICA/Analisis_Datos_Omicos/ADO_PEC1/ADO_PEC1.Rmd",
+      output = "codigo_exploracion_datos.R", documentation = 1)
```

4.6 CREACIÓN DE UN REPOSITORIO DE GITHUB CON EL INFORME Y DATOS DEL ESTUDIO

Los pasos a seguir son:

(1) Abrimos GitHub y creamos una cuenta.

(2) Seleccionamos la opción de "Create New Repository". Seleccionamos que el repositorio sea de tipo "Public"

(3) Como se indica en el enunciado, su nombre será "Sancho-Romero-JuanManuel-PEC1"

(4) Clicamos en la opción "uploading an existing file." y una vez dentro, podemos seleccionar o arrastrar los archivos indicados dentro de nuestro repositorio.

(5) La dirección (url) del repositorio ha sido incluida en la última sección de este informe.

5. DISCUSIÓN Y LIMITACIONES.

En esta primera Prueba de Evaluación Continua (PEC) se ha simulado de manera simplificada el proceso de análisis ómicos a partir de los datos metabolómicos de pacientes sometidos a cirugía bariátrica (Palau-Rodriguez et al. 2018) obtenidos de un repositorio de GitHub. Esto ha permitido aplicar las herramientas bioinformáticas ya mencionadas (R, Bioconductor y GitHub) para poder realizar una exploración de datos y a partir de la creación de un contenedor `SummarizedExperiment` y así manejar los datos de forma estructurada, lo que optimizó la organización y simplificó el acceso a la información. La experiencia práctica de construir el contenedor de la clase `SummarizedExperiment` y realizar alguna indagación básica inicial sobre el funcionamiento de este objeto ha sido esencial para comprender su potencial y aplicabilidad en el proceso de análisis de datos ómicos. A su vez, el uso de GitHub, la clonación y creación de repositorios, ha sido de gran valor para desarrollar en el futuro un flujo de trabajo mucho más eficiente y profesional en el ámbito de la Bioinformática.

En cuanto a las limitaciones de este trabajo, la más evidente o significativa es la simplicidad del análisis que se ha desarrollado. Debido a que el objetivo es una toma de contacto inicial para familiarizaros con las herramientas y el entorno de trabajo, el análisis no se ha desarrollado con la idea de responder de manera exhaustiva a ninguna pregunta de interés biológico. Sin embargo, mediante este Análisis de Componentes Principales muy simplificado, se propone la posibilidad de considerar si existen diferencias significativas entre los perfiles metabolómicos de los sujetos con intervenciones quirúrgicas de tipo "by pass" con respecto a los sujetos con intervenciones de tipo "tubular". Evidenciando así, la relevancia de un estudio más exhaustivo al respecto, como el realizado por Palau-Rodriguez et al. (2018), donde se pretende obtener conclusiones más significativas a cerca de esta hipótesis, entre otras.

Otra limitación ha sido que, al haber trabajado exclusivamente con datos metabolómicos, no ha sido posible aplicar el `paquete Rqc`, que fue una de las herramientas propuestas y exploradas en esta unidad, para el análisis de control de calidad en datos de secuenciación. Esto ha restringido la posibilidad de usar técnicas de análisis de calidad de datos, obligando a enfocar el análisis exploratorio a métodos más simples y compatibles con los datos metabolómicos.

6. CONCLUSIONES DEL ESTUDIO.

El proceso desarrollado en esta PEC ha permitido consolidar e integrar el conocimiento y los distintos objetivos propuestos para la gestión de datos ómicos, desde la descarga de datos hasta el análisis exploratorio de un objeto `SummarizedExperiment`. La creación de este tipo de contenedor ha sido clave para aprender a organizar y estructurar datos complejos en un formato que facilita su análisis y visualización de forma sencilla.

Por otro lado, este trabajo ha sido de gran interés para una primera toma de contacto en el pre-procesado de datos y creación de gráficos haciendo uso del paquete `POMA`, así como una iniciación al Análisis de Componentes Principales (PCA), con el fin de entrever patrones dominantes o tendencias en los datos relacionadas con las hipótesis del estudio de Palau-Rodriguez et al. (2018).

Además, el uso de GitHub como repositorio de trabajo ha aportado una visión práctica sobre la importancia de la gestión de versiones y la colaboración en proyectos de investigación, lo que a su vez ha subrayado la relevancia de desarrollar en el futuro un flujo de trabajo mucho más eficiente y profesional en cualquier ámbito de la Bioinformática.

En conclusión, el presente trabajo me ha brindado una iniciación idónea para abordar futuros estudios relacionados con el proceso de análisis de datos ómicos más complejos, y al mismo tiempo ha dado espacio a distintas reflexiones sobre las necesidades y desafíos cotidianos dados contextos reales en este ámbito de estudio.

7. BIBLIOGRAFÍA

- Palau-Rodriguez, M., Tulipani, S., Marco-Ramell, A., Miñarro, A., Jáuregui, O., Sanchez-Pla, A., Ramos-Molina, B., Tinahones, F. J., & Andres-Lacueva, C. (2018). Metabotypes of response to bariatric surgery independent of the magnitude of weight loss. PLOS ONE, 13(6), e0198214. https://doi.org/10.1371/journal.pone.0198214
- https://uw.pressbooks.pub/appliedmultivariatestatistics/chapter/eigenanalysis/
- https://uw.pressbooks.pub/appliedmultivariatestatistics/chapter/pca/
- https://www.datanovia.com/en/blog/cluster-analysis-in-r-practical-guide/
- https://aspteaching.github.io/AMVCasos/
- https://www.rdocumentation.org/packages/DESeq2/versions/1.12.3/topics/varianceStabilizingTransformation

- http://www.sthda.com/english/articles/31-principal-component-methods-in-r-practical-guide/112-pca-principal-component-analysis-essentials/#visualization-and-interpretation
- https://www.bioconductor.org/packages/release/bioc/vignettes/POMA/inst/doc/POMA-workflow.html#the-poma-workflow

8. DIRECCIÓN (URL) DEL REPOSITORIO

https://github.com/Jm14sr/Sancho-Romero-JuanManuel-PEC1.git