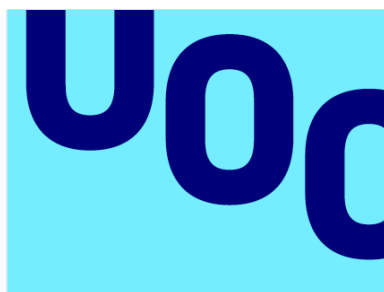


PEC1 Análisis de Datos Ómicos

Hansel Lopez Chevalier

2025-10-24



Universitat Oberta
de Catalunya

Abstract

Este estudio exploratorio analizo datos metabolomicos del estudio ST004239, descargados desde MetabolomicsWorkbench y procedido en R mediante Bioconductor. Los datos se estructuraron en un objeto tipo SummarizedExperiment, permitiendo una manipulacion eficiente. Se aplicaron tecnicas multivariantes como bloxplots, analisis de componentes principale, heatmaps y clustering jerarquico para evaluar la distribucion y agropamiento de las muestras. Se identificaron valores faltantes NA, que fueron tratados asignandos valores ceros para facilidad el analisis. La transfromacion logaritmica mejoro la simetria de los datos. El PCA revelo que el primer componente explica mas del 75% de la variabilidad y los agrupamientos reflejan diferencias sistematicas entre lineas de parasitos y los timepoints. Estos hallazgos validan la calidad del diseño experimental y ofrecen una base solida para estudios posteriores.

1- Objetivos

El objetivo principal del presente estudio es realizar un análisis exploratorio de los datos metabolomicos del estudio ID idendicado (ST004239) obtenidos de la base de datos metabolomicsWorkbench (<https://www.metabolomicsworkbench.org/>) utilizando el programa estadístico R y las librerías para análisis de datos ómicos integradas en Bioconductor. Por lo tanto, se requiere la obtencion de los datos y sea almacenados en un objeto de clase SummarizedExperiment para poder accecer a ellos a través de este objeto.

2- Materiales y Metodos

Los datos del estudio titulado *“Loss of P. falciparum Amino Acid Transporter (ApiAT2) Function Increases Intracellular Proline and Confers Resistance to Prolyl-tRNA Synthetase Inhibitors”* se descargaron de metabolomicsWorkbench utilizando el paquetemetabolomicsWorkbenchR(<https://www.bioconductor.org/packages/release/bioc/html/metabolomicsWorkbenchR.html>) de Bioconductor. Este paquete permite descargarlos los datos y crear un output en estrucutra objeto tipo

SummarizedExperiment, la cual contendrá la matriz de datos preprocesados. La exploración de los datos se llevó a cabo usando las funciones de las librerías SummarizedExperiment, ggplot2, pheatmap, factoextra, PCATools y naniar. Por lo que, la exploración consistió en análisis multivariante de los datos, mediante boxplots o histogramas para estudiar la forma general de los datos, además también se analizaron los datos mediante análisis de los componentes principales y agrupamientos jerárquicos para determinar si los grupos que aparecen están relacionados con las fuentes de variabilidad del estudio o si por el contrario podrían haber otras fuentes de variabilidad como son el efecto batch.

```
#instalamos la libreria requerida
if (!require("BiocManager", quietly = TRUE))
  install.packages("BiocManager")

BiocManager::install("metabolomicsWorkbenchR")

#ejecutamos la libreria y creamos la clase SummarizedExperiment
library(metabolomicsWorkbenchR)
mySE = do_query(
  context = 'study',
  input_item = 'study_id',
  input_value = 'ST004239',
  output_item = 'SummarizedExperiment'
)
mySE
```

```
## class: SummarizedExperiment
## dim: 127 72
## metadata(8): data_source study_id ... description subject_type
## assays(1): ''
## rownames(127): ME1147833 ME1147834 ... ME1147832 ME1147957
## rowData names(3): metabolite_name metabolite_id refmet_name
## colnames(72): Study5_Dd2_iRBC_labeled_0.5h_R1
##   Study5_Dd2_iRBC_labeled_0.5h_R2 ... Study5_uRBCs_labeled_6h_R2
##   Study5_uRBCs_labeled_6h_R3
## colData names(9): local_sample_id study_id ... Parasite_line Timepoint
```

2.1- Analisis exploratorio de los datos

Con los datos ya estructurados correctamente procedimos al análisis exploratorio inicial.

```
library(SummarizedExperiment)
data <- colData(mySE)[, c("local_sample_id", "study_id", "Parasite_line", "Timepoint")]

# Convertir a data.frame para facilitar de manipulación
data <- as.data.frame(data)
```

```
#calculamos la tabla de frecuencia de la columna 4
table(data$Timepoint)
```

```
##
##   0 h 0.5 h   1 h   10 h   2 h   6 h
##    12    12    12    12    12    12
```

```
#Calculamos el resumen sumerico de la columna 2
summary(data$Parasite_line)
```

```
## Control RBCs          Dd2
##              18         54
```

Como podemos apreciar los datos estan distribuidos de manera correctas o balanceados.

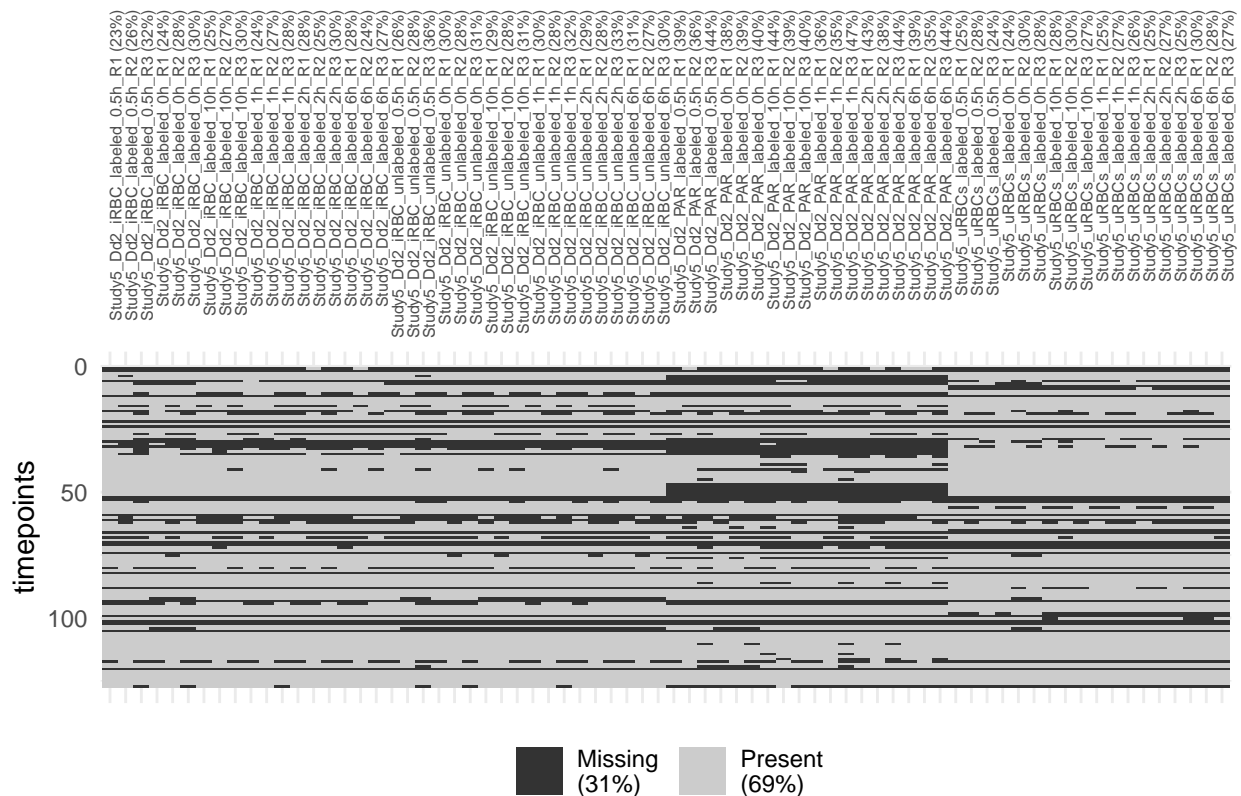
2.2.1-Valores faltantes Muchos datos omicos durante su generacion contienen valores faltantes, lo que podria dificultar la correcta interpretacion de los analisis. Por lo que debemos identificar los valores faltantes, los cuales a menudo aparecen como NA

Para ello inicialmente, dado que no conocemos el tipo de datos asignaremos el assay como “rawValues” , especialmente cuando se esta trabajando con múltiples capas de datos o se necesita claridad en el flujo de trabajo

```
# Visualización de valores faltantes
assayNames(mySE) <- "rawValues"
data_faltantes <- as.data.frame(assay(mySE, "rawValues"))
```

Como podemos apreciar en nuestro DF, se encuentran valores faltantes NA y determinar como abordar estos valores -ya sea eliminados o ignorandolos- resesenta un desafio. Por lo tanto, en este sentido, bajo el supuesto de que su ausencia se debe a que no fueron detectados o eestan realmentes ausentes, explorare rapidameente la distribucion de estos valores usando el paquete **nanier**

```
if (!require(nanier)) install.packages("nanier")
if (!require(ggplot2)) install.packages("ggplot2")
library(nanier)
library(ggplot2)
# Visualización con ajuste del tamaño de etiquetas
vis_miss(data_faltantes) + theme(axis.text.x = element_text(size = 6, angle = 90, hjust = 1)) +
  ylab("timepoints")
```



Como podemos apreciar en la grafica, los datos contiene un alto numero de valores faltantes, esto se confirmo mediante un resumen numerico

```
n_miss(data_faltantes)
```

```
## [1] 2831
```

```
n_complete(data_faltantes)
```

```
## [1] 6313
```

```
prop_miss(data_faltantes)
```

```
## [1] 0.3096019
```

```
prop_complete(data_faltantes)
```

```
## [1] 0.6903981
```

dado la naturaleza de los datos, detectar los algunos valores faltantes puede resultar complicado, por lo que para trabajar los datos, asignamos valor de 0

```

rawValues <- assay(mySE, "rawValues")
rawNoMissings <- rawValues
rawNoMissings[is.na(rawNoMissings)] <- 0
sum(is.na(rawNoMissings))

```

```
## [1] 0
```

```

assays(mySE)$rawNoMissings <- rawNoMissings
show(mySE)

```

```

## class: SummarizedExperiment
## dim: 127 72
## metadata(8): data_source study_id ... description subject_type
## assays(2): rawValues rawNoMissings
## rownames(127): ME1147833 ME1147834 ... ME1147832 ME1147957
## rowData names(3): metabolite_name metabolite_id refmet_name
## colnames(72): Study5_Dd2_iRBC_labeled_0.5h_R1
##      Study5_Dd2_iRBC_labeled_0.5h_R2 ... Study5_uRBCs_labeled_6h_R2
##      Study5_uRBCs_labeled_6h_R3
## colData names(9): local_sample_id study_id ... Parasite_line Timepoint

```

3- Resultados

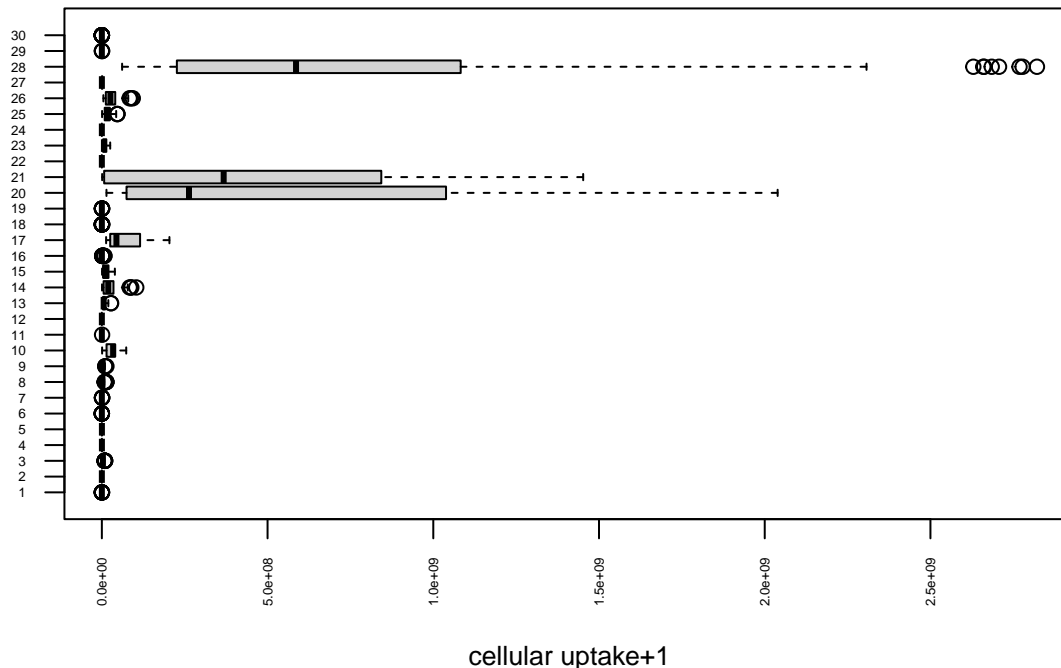
Inicialmente exploramos como se distribuyen nuestros datos en cada tratamiento con un boxplot multiple, el cual se trabajo directamente de la clase SummarizedExperiment y algunas de sus funciones e.g: nrow, ncol y assay. Para la visualizacion de los datos, solo veremos los primeros 30

```

boxplot(t(assay(mySE, "rawNoMissings")[1:30, ]), xlab = "cellular uptake+1", cex.lab = 0.8,
        horizontal = TRUE, cex.axis = 0.4, las = 2, main = "Distribución de los valores por timepoints",
        cex.main = 0.8)

```

Distribución de los valores por timepoints



Como podemos apreciar, la distribución de los valores es muy variante entre los timepoints al momento de que el parásito toma los isótopos de los aminoácidos.

Dado los resultados anteriores, se consideró hacer un análisis logarítmico para simetrizar los datos. Para evitar errores se añadió 1 como valor antes de tomar logarítmicos, esto da la ventaja de hacerlo como el SummarizedExperiment, ya que permite tener dos datos, uno con los valores originales y otro con los metadatos.

```
logNoMissings <- log(assay(mySE, "rawNoMissings") + 1)
assays(mySE, withDimnames = FALSE)$logNoMissings <- logNoMissings
show(mySE)
```

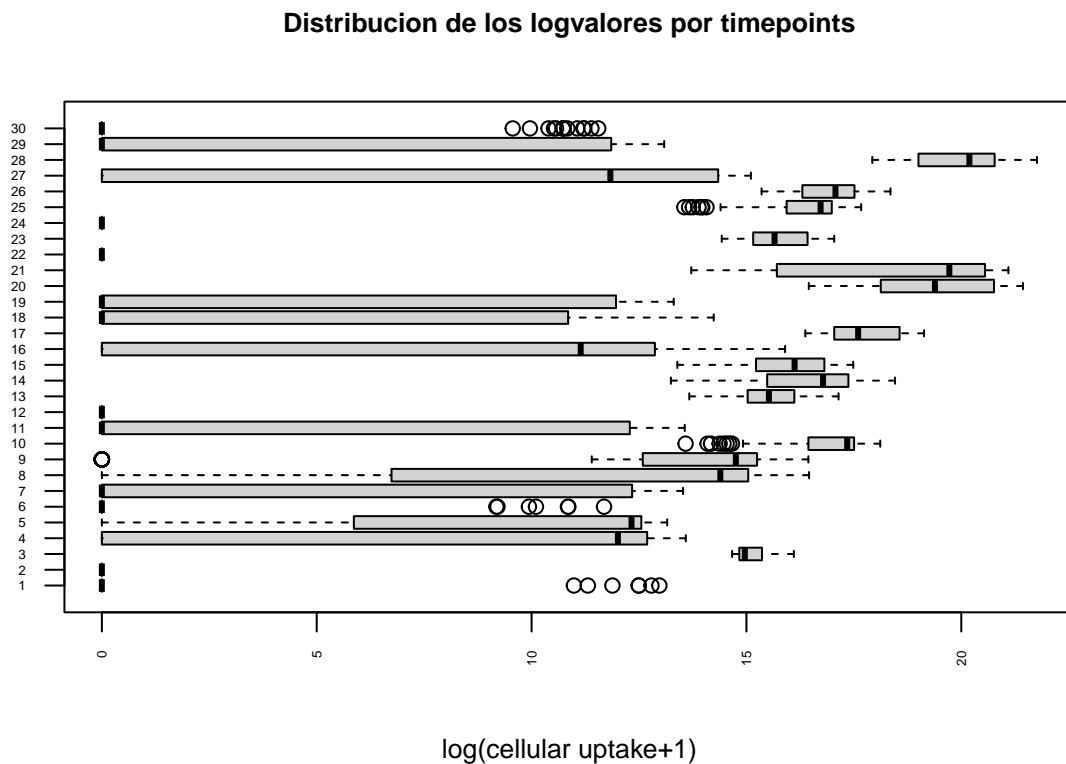
```
## class: SummarizedExperiment
## dim: 127 72
## metadata(8): data_source study_id ... description subject_type
## assays(3): rawValues rawNoMissings logNoMissings
## rownames(127): ME1147833 ME1147834 ... ME1147832 ME1147957
## rowData names(3): metabolite_name metabolite_id refmet_name
## colnames(72): Study5_Dd2_iRBC_labeled_0.5h_R1
##   Study5_Dd2_iRBC_labeled_0.5h_R2 ... Study5_uRBCs_labeled_6h_R2
##   Study5_uRBCs_labeled_6h_R3
## colData names(9): local_sample_id study_id ... Parasite_line Timepoint
```

Guardamos el objeto para otros estudios.

```
save(mySE, file = "metaboda.Rda")
```

ya con los datos transformados repetimos el boxplot para ver su efecto, esto nos da varias ventajas, como es mejorar la simetria de los datos, evita que la variabilidad aumente con la media y reduce el impacto de valores externos

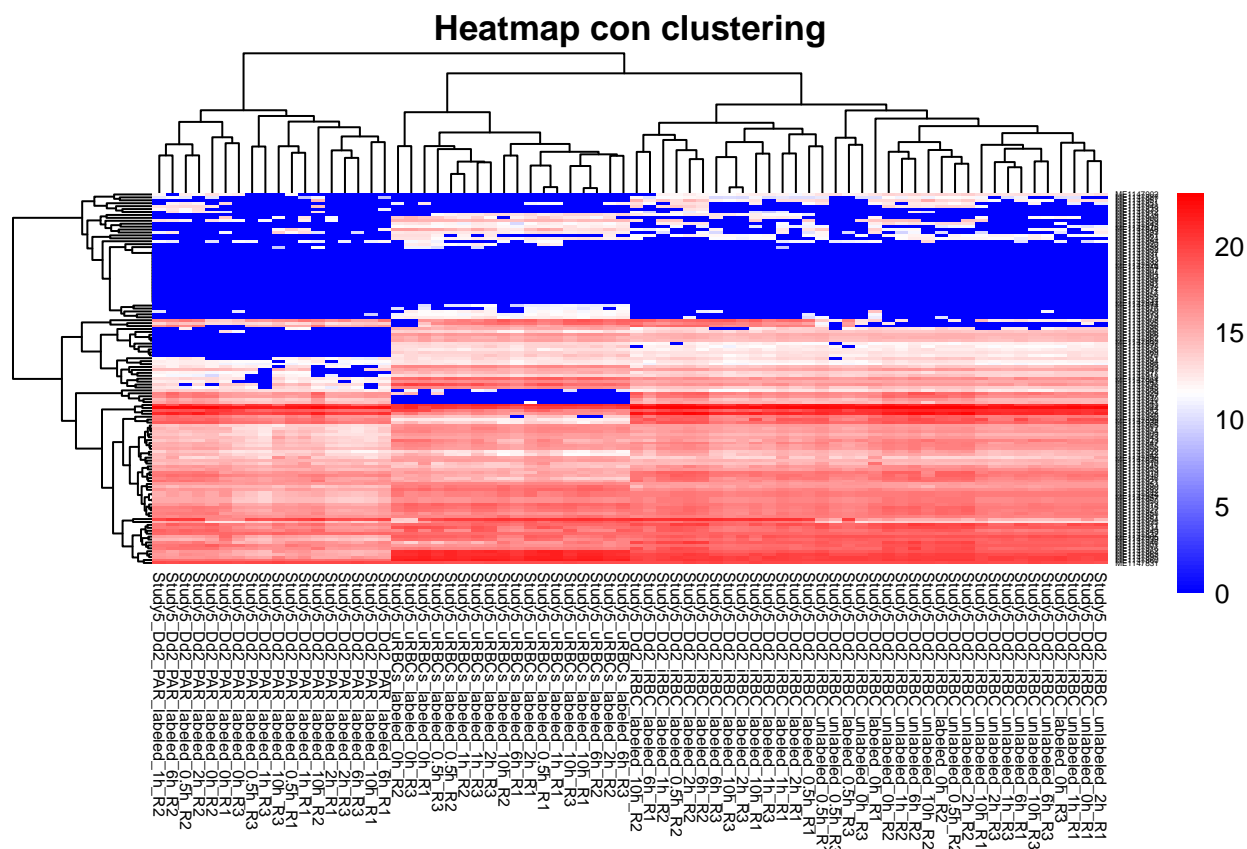
```
boxplot(t(assay(mySE, "logNoMissings")[1:30, ]), xlab = "log(cellular uptake+1)",
        cex.lab = 0.8, horizontal = TRUE, cex.axis = 0.4, las = 2, main = "Distribucion de los logvalores p",
        cex.main = 0.8)
```



3.1 Visualizacion paralela de los datos

Para la detección de posibles grupos de variables que varían de forma coordinada se creó un heatmap

```
if (!require(pheatmap)) install.packages("pheatmap")
library(pheatmap)
pheatmap(assay(mySE, "logNoMissings"), main = "Heatmap con clustering", fontsize_row = 3,
        fontsize_col = 6, color = colorRampPalette(c("blue", "white", "red"))(100))
```



Como podemos apreciar la maestris revela una separacion clara de los datos, lo cual queda evidenciada por el agrupamiento de las muestras en el dendrograma superior. Este patro sugiere la presencia de diferencias sistematicas entre los grupos, lo cual podria estar asociado a procesos biologicos relevantes del parasito

3.2 Analisis de los componentes principales

Creamos un objeto para la exploracion de los datos, inicialmente vemos el peso de cada coomponente ocn la funcion screeplot para ver el resultado por cada componente

```
if (!require(PCAtools)) BiocManager::install("PCAtools", dep = TRUE)
library(PCAtools)
p <- pca(assay(mySE, "rawNoMissings"), metadata = colData(mySE), removeVar = 0.1)
class(p)
```

```
## [1] "pca"
```

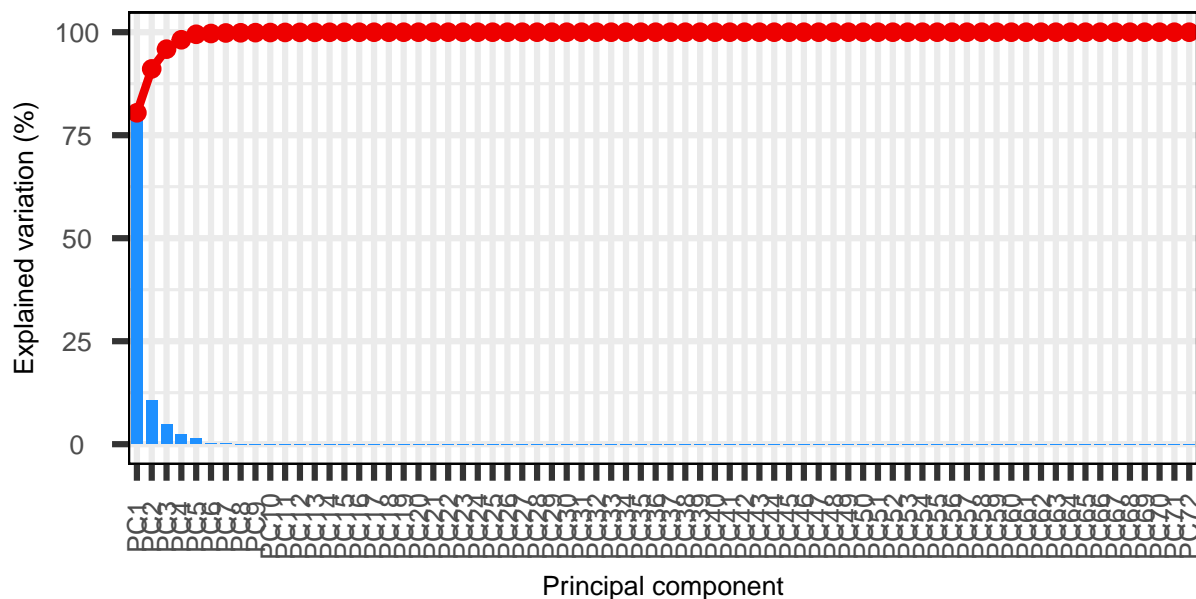
```
screeplot(p, axisLabSize = 10, titleLabSize = 22)
```

```
## Warning: Using 'size' aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use 'linewidth' instead.
## i The deprecated feature was likely used in the PCAtools package.
## Please report the issue to the authors.
## This warning is displayed once every 8 hours.
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.
```



```
## Warning: The 'size' argument of 'element_rect()' is deprecated as of ggplot2 3.4.0.
## i Please use the 'linewidth' argument instead.
## i The deprecated feature was likely used in the PCAtools package.
##   Please report the issue to the authors.
## This warning is displayed once every 8 hours.
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.
```

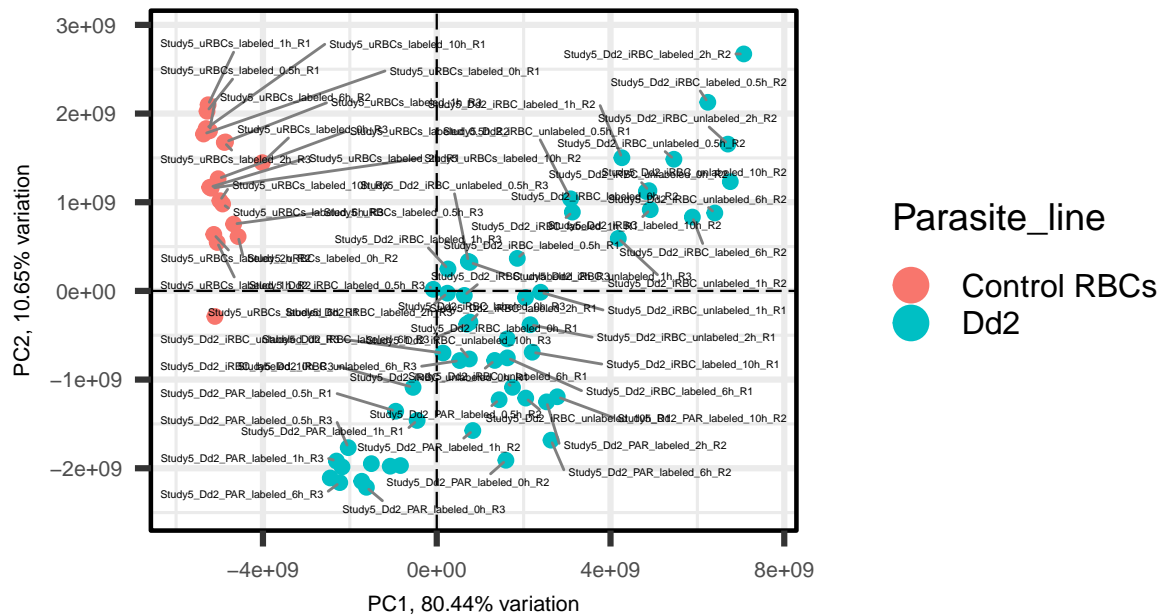
SCREE plot



Como podemos apreciar, el primer componente explica mas del 75% de la variabilidad, mientras que los componentes 2 y 3 solo representan poco de menos del 10%. Dado estos resultados se visualizo unicamente las muestras en el mismo grafico para las muestras y las variables y ver la correlacion con los componentes

```
biplot(p, showLoadings = FALSE, labSize = 1.5, pointSize = 1.5, axisLabSize = 8, title = "PCA de los da
      sizeLoadingsNames = 3, colby = "Parasite_line", hline = 0, vline = 0, legendPosition = "right")
```

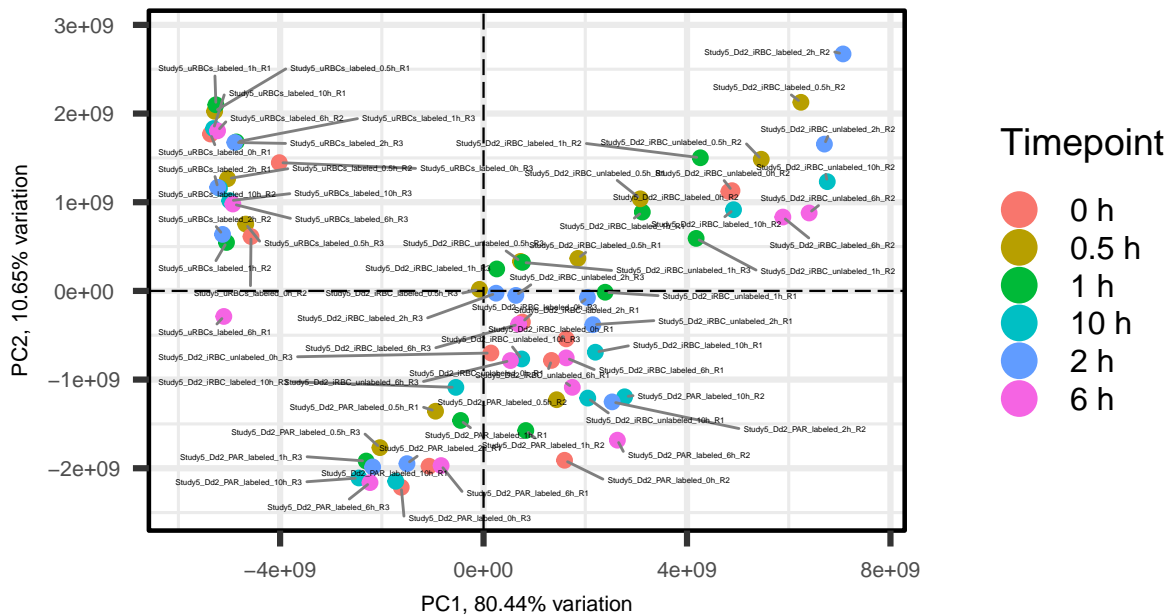
PCA de los datos



Como podemos apreciar al colorear “Parasite_line” todas las muestras quedan bien separadas por el primer PC. Por otro lado, al visualizar los datos de otra variable de los metadatos.

```
biplot(p, showLoadings = FALSE, labSize = 1, pointSize = 1.5, axisLabSize = 8, title = "PCA de los datos",
       sizeLoadingsNames = 3, colby = "Timepoint", hline = 0, vline = 0, legendPosition = "right")
```

PCA de los datos

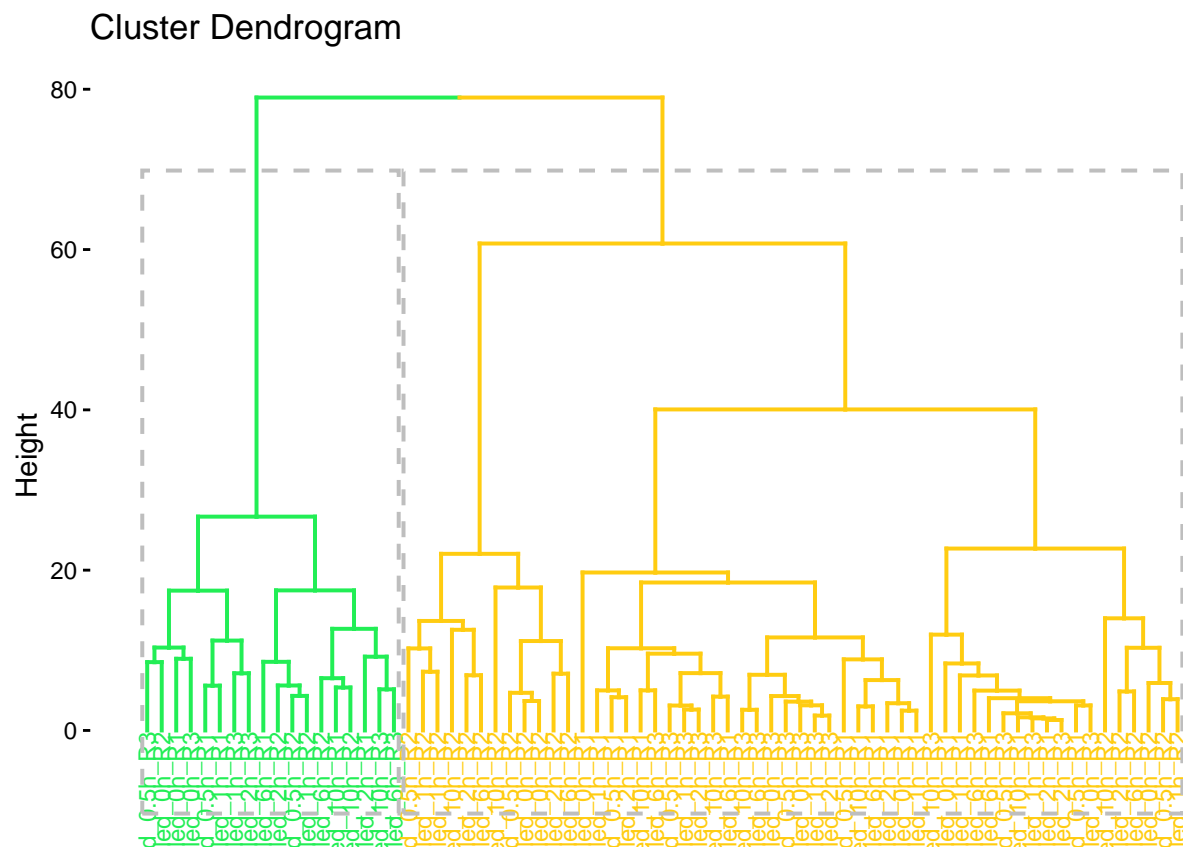


Este grafico, las muestras se agrupan de foorma coherente segun el timepoint, como se aprecian los colores y la proximidad espacial de los puntos, las muestras de 0h t 0.5h se agrupan en un extremo mientras que las d e6h y 10h se posicionan en el extremo opuesto, indicadndo cambios acumulativos en la estructura de los datos.

Por ultimo, para confirmar los datos se realizo un agrupamiento jerarquico y se visualizo con un dendograma. y Como se puede observar hay una tendencia bastante clara.

```
if (!require(factoextra)) install.packages("factoextra")
res.hc <- t(assay(mySE, "rawNoMissings")) %>%
  scale() %>%
  dist(method = "euclidean") %>%
  hclust(method = "ward.D2")

library(factoextra)
fviz_dend(res.hc, cex = 0.6, k = 2, k_colors = c("#2E5", "#FC1"), color_labels_by_k = TRUE,
  rect = TRUE)
```



4- Conclusion

EL analisis de los componentes principales y el clustering jerarquico aplicado a los datos de capacion de aminoacidos marcados en plasmodium revelaron una clara tendencias a la separacion de la muestras en dos grupos principales, lo cual es consistence con el estudio de Bopp et al. (2025). Esta seperacion reflejo principalmente la linea del parasito (control vs wt), lo cual validad la robustuez del diseño experimental y la sensibilidad del enfoque multivariado. Por lo tanto, estos resultados refuerzan la utilidad de herramientas como R para el analisis de datos omicos.

5- Apendice y referencia

MetabolomicsWorkbenchR. (s. f.). Bioconductor. <https://www.bioconductor.org/packages/release/bioc/html/metabolomicsWorkbenchR.html>

todos los datos generados durante el analisis, asi como los script se encuentran adjuto en el siguiente repositorio: <https://github.com/hanseln3s5/DocumentosGeneradosPEC1.git>