PEC 1 Datos Ómicos Daniel Zamora Solé

Daniel Zamora Solé

2024-11-05

Esta documentación esta destinada a guiar al usuario por los distintos pasos que sean realizado para llegar a construir el objeto de SummarizedExperiment de la fuente de datos 2023-UGrX-4MetaboAnalystTutorial en el repositorio de https://raw.githubusercontent.com/nutrimetabolomics/metaboData.

Primero realizamos instalamos los componentes necesarios de Bioconductor y la libreria de SummarizedExperiment.

```
# Instalación Bioconductor y SummirizedExperiment
if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")

BiocManager::install("SummarizedExperiment")
```

```
## Bioconductor version 3.18 (BiocManager 1.30.25), R 4.3.1 (2023-06-16 ucrt)
```

```
## Warning: package(s) not installed when version(s) same as or greater than current; use
##   'force = TRUE' to re-install: 'SummarizedExperiment'
```

```
## Installation paths not writeable, unable to update packages
##   path: C:/Program Files/R/R-4.3.1/library
##   packages:
##     boot, cluster, codetools, foreign, KernSmooth, lattice, mgcv, nlme, rpart,
##     spatial, survival
```

```
## Old packages: 'cli', 'digest', 'rlang', 'yaml'
```

```
# Cargamos SummarizedExperimenta
library(SummarizedExperiment)
```

```
## Loading required package: MatrixGenerics
```

```
## Loading required package: matrixStats
```

```
## Warning: package 'matrixStats' was built under R version 4.3.3
```

```
##
## Attaching package: 'MatrixGenerics'
```

```
## The following objects are masked from 'package:matrixStats':
##
##     colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,
##     colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
##     colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,
##     colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
##     colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,
##     colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,
##     colWeightedMeans, colWeightedMedians, colWeightedSds,
##     colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
##     rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
##     rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,
##     rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
##     rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
##     rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,
##     rowWeightedMads, rowWeightedMeans, rowWeightedMedians,
##     rowWeightedSds, rowWeightedVars

## Loading required package: GenomicRanges

## Loading required package: stats4

## Loading required package: BiocGenerics

##
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':
##
##     anyDuplicated, aperm, append, as.data.frame, basename, cbind,
##     colnames, dirname, do.call, duplicated, eval, evalq, Filter, Find,
##     get, grep, grepl, intersect, is.unsorted, lapply, Map, mapply,
##     match, mget, order, paste, pmax, pmax.int, pmin, pmin.int,
##     Position, rank, rbind, Reduce, rownames, sapply, setdiff, sort,
##     table, tapply, union, unique, unsplit, which.max, which.min

## Loading required package: S4Vectors

## Warning: package 'S4Vectors' was built under R version 4.3.2

##
## Attaching package: 'S4Vectors'

## The following object is masked from 'package:utils':
##
##     findMatches
```

```
## The following objects are masked from 'package:base':
##
##     expand.grid, I, unname

## Loading required package: IRanges

##
## Attaching package: 'IRanges'

## The following object is masked from 'package:grDevices':
##
##     windows

## Loading required package: GenomeInfoDb

## Warning: package 'GenomeInfoDb' was built under R version 4.3.3

## Loading required package: Biobase

## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.

##
## Attaching package: 'Biobase'

## The following object is masked from 'package:MatrixGenerics':
##
##     rowMedians

## The following objects are masked from 'package:matrixStats':
##
##     anyMissing, rowMedians
```

A continuación extraeremos los datos planos donde tenemos los experimentos, los metabolitos y la concentración de metabolitos en cada uno, al ser un texto plano tenemos que coger los datos que estan en el bloque delimitado por 'MS_METABOLITE_DATA_START' y 'MS_METABOLITE_DATA_END'.

```r
# Definimos la url de nuestra fuente de datos
data_url <- "https://raw.githubusercontent.com/nutrimetabolomics/metaboData/main/Datasets/2023-UGrX-4Me


lines <- readLines(data_url)
# Encontramos la línea que contiene 'MS_METABOLITE_DATA_START' y 'MS_METABOLITE_DATA_END'
start_line <- grep("MS_METABOLITE_DATA_START", lines)
end_line <- grep("MS_METABOLITE_DATA_END", lines)

data_lines <- lines[(start_line + 1):(end_line - 1)]
```

```r
#Creamos el dataframe
data <- read.table(text = data_lines, sep = "\t", header = TRUE, check.names = FALSE, stringsAsFactors =



# Miramos los 6 primers para ver el resultado
head(data)
```

```
##                  Samples                            LabF_684508
## 1                Factors Transplantation:After transplantation
## 2            1-monoolein                              6047.0000
## 3          1-monostearin                              9771.0000
## 4 2-hydroxybutanoic acid                             13238.0000
## 5 2-hydroxyglutaric acid                              7160.0000
## 6  2-ketoisocaproic acid                               812.0000
##                              LabF_684512                              LabF_684516
## 1 Transplantation:After transplantation Transplantation:After transplantation
## 2                             2902.0000                              1452.0000
## 3                             6521.0000                              1302.0000
## 4                            29774.0000                              4134.0000
## 5                            11501.0000                              3202.0000
## 6                             2011.0000                               738.0000
##                              LabF_684520                              LabF_684524
## 1 Transplantation:After transplantation Transplantation:After transplantation
## 2                             3428.0000                              2985.0000
## 3                             2781.0000                              5789.0000
## 4                             4419.0000                             13334.0000
## 5                            17238.0000                             20376.0000
## 6                             2550.0000                               871.0000
##                              LabF_684528                              LabF_684483
## 1 Transplantation:After transplantation Transplantation:Before transplantation
## 2                            16334.0000                             244142.0000
## 3                             4338.0000                              16848.0000
## 4                             2115.0000                              11587.0000
## 5                             1109.0000                               8276.0000
## 6                              628.0000                               2096.0000
##                              LabF_684487                              LabF_684491
## 1 Transplantation:Before transplantation Transplantation:Before transplantation
## 2                             6968.0000                              1928.0000
## 3                            10206.0000                              9398.0000
## 4                            65635.0000                             32433.0000
## 5                            12402.0000                             20964.0000
## 6                             3472.0000                             10669.0000
##                              LabF_684495                              LabF_684499
## 1 Transplantation:Before transplantation Transplantation:Before transplantation
## 2                            19228.0000                              3029.0000
## 3                             1013.0000                              4190.0000
## 4                             1823.0000                              4429.0000
## 5                            25913.0000                              2709.0000
## 6                              432.0000                              1055.0000
##                              LabF_684503
## 1 Transplantation:Before transplantation
## 2                            23277.0000
```

```
## 3                              11114.0000
## 4                              30427.0000
## 5                              70972.0000
## 6                               1005.0000
```

```r
factors <- data[1, ] # guardamos los factores

# Removemos la fila de factores del dataframe
data <- data[-1, ]
rownames(data) <- NULL

# La primera columna es 'Samples' y el resto son nombres de muestras
colnames(data)[1] <- "Metabolite"
sample_names <- colnames(data)[-1]


head(data)
```

```
##               Metabolite LabF_684508 LabF_684512 LabF_684516 LabF_684520
## 1            1-monoolein   6047.0000   2902.0000   1452.0000   3428.0000
## 2          1-monostearin   9771.0000   6521.0000   1302.0000   2781.0000
## 3 2-hydroxybutanoic acid  13238.0000  29774.0000   4134.0000   4419.0000
## 4 2-hydroxyglutaric acid   7160.0000  11501.0000   3202.0000  17238.0000
## 5  2-ketoisocaproic acid    812.0000   2011.0000    738.0000   2550.0000
## 6         2-monopalmitin   1511.0000    622.0000    883.0000    796.0000
##   LabF_684524 LabF_684528 LabF_684483 LabF_684487 LabF_684491 LabF_684495
## 1   2985.0000  16334.0000 244142.0000   6968.0000   1928.0000  19228.0000
## 2   5789.0000   4338.0000  16848.0000  10206.0000   9398.0000   1013.0000
## 3  13334.0000   2115.0000  11587.0000  65635.0000  32433.0000   1823.0000
## 4  20376.0000   1109.0000   8276.0000  12402.0000  20964.0000  25913.0000
## 5    871.0000    628.0000   2096.0000   3472.0000  10669.0000    432.0000
## 6    623.0000   5716.0000   3405.0000   3196.0000   1457.0000   1416.0000
##   LabF_684499 LabF_684503
## 1   3029.0000  23277.0000
## 2   4190.0000  11114.0000
## 3   4429.0000  30427.0000
## 4   2709.0000  70972.0000
## 5   1055.0000   1005.0000
## 6   1275.0000  14445.0000
```

```r
# Revisamos la estructura de los datos
str(data)
```

```
## 'data.frame':    142 obs. of  13 variables:
##  $ Metabolite : chr  "1-monoolein" "1-monostearin" "2-hydroxybutanoic acid" "2-hydroxyglutaric acid"
##  $ LabF_684508: chr  "6047.0000" "9771.0000" "13238.0000" "7160.0000" ...
##  $ LabF_684512: chr  "2902.0000" "6521.0000" "29774.0000" "11501.0000" ...
##  $ LabF_684516: chr  "1452.0000" "1302.0000" "4134.0000" "3202.0000" ...
##  $ LabF_684520: chr  "3428.0000" "2781.0000" "4419.0000" "17238.0000" ...
##  $ LabF_684524: chr  "2985.0000" "5789.0000" "13334.0000" "20376.0000" ...
##  $ LabF_684528: chr  "16334.0000" "4338.0000" "2115.0000" "1109.0000" ...
##  $ LabF_684483: chr  "244142.0000" "16848.0000" "11587.0000" "8276.0000" ...
##  $ LabF_684487: chr  "6968.0000" "10206.0000" "65635.0000" "12402.0000" ...
```

```
##  $ LabF_684491: chr  "1928.0000" "9398.0000" "32433.0000" "20964.0000" ...
##  $ LabF_684495: chr  "19228.0000" "1013.0000" "1823.0000" "25913.0000" ...
##  $ LabF_684499: chr  "3029.0000" "4190.0000" "4429.0000" "2709.0000" ...
##  $ LabF_684503: chr  "23277.0000" "11114.0000" "30427.0000" "70972.0000" ...
```

```r
# Creamos una función para detectar valores no numéricos
non_numeric_indices <- list()

for (i in 2:ncol(data)) {
# Identificamos los valores que no pueden ser convertidos a numéricos
  non_numeric <- !grepl("^[0-9\\.]+$", data[[i]])

  if (any(non_numeric, na.rm = TRUE)) {
    non_numeric_indices[[colnames(data)[i]]] <- which(non_numeric)
  }
}

# Mostramos las columnas y filas donde hay valores no numéricos
non_numeric_indices
```

```
## list()
```

```r
# Miramos los valores no numéricos
for (col_name in names(non_numeric_indices)) {
  indices <- non_numeric_indices[[col_name]]
  cat("Columna:", col_name, "\n")
  print(data[indices, c("Metabolite", col_name)])
  cat("\n")
}

# Reemplazamos valores no numéricos por NA
for (i in 2:ncol(data)) {
  data[[i]] <- as.numeric(as.character(data[[i]]))
}
# Calculamos el número de NAs en cada columna
na_counts <- sapply(data[, -1], function(x) sum(is.na(x)))

# Mostrar las columnas con NAs y el número de NAs aaaaaa
na_counts[na_counts > 0]
```

```
## named integer(0)
```

```r
na_counts
```

```
## LabF_684508 LabF_684512 LabF_684516 LabF_684520 LabF_684524 LabF_684528
##           0           0           0           0           0           0
## LabF_684483 LabF_684487 LabF_684491 LabF_684495 LabF_684499 LabF_684503
##           0           0           0           0           0           0
```

Una vez tenemos los datos de los experimentos limpios vamos a crear los datos de Mertadatos de las muestras, que son las columnas.

```r
#Creamos el dataframe de los metadatos de las muestras (colData)
# Extrarmos los nombres de las muestras (columnas)
sample_names <- colnames(data)[-1]  # Sacamos la primera columna

# Extraemos los factores correspondientes
group_labels <- as.character(factors[-1])

# Limpiamos los nombres de los grupos
group_labels <- gsub("Transplantation:", "", group_labels)
group_labels <- factor(group_labels)

# Creamos el dataframe de metadatos de las muestras
colData <- data.frame(
  SampleID = sample_names,
  Group = group_labels,
  stringsAsFactors = FALSE
)

# Establecemos los nombres de las filas
rownames(colData) <- colData$SampleID

colData
```

```
##                SampleID                  Group
## LabF_684508 LabF_684508  After transplantation
## LabF_684512 LabF_684512  After transplantation
## LabF_684516 LabF_684516  After transplantation
## LabF_684520 LabF_684520  After transplantation
## LabF_684524 LabF_684524  After transplantation
## LabF_684528 LabF_684528  After transplantation
## LabF_684483 LabF_684483 Before transplantation
## LabF_684487 LabF_684487 Before transplantation
## LabF_684491 LabF_684491 Before transplantation
## LabF_684495 LabF_684495 Before transplantation
## LabF_684499 LabF_684499 Before transplantation
## LabF_684503 LabF_684503 Before transplantation
```

Una vez tenemos los metadatos de las columnas ahora sacaremos los metadatos de las filas. Que serian los metadatos de los distintos tipos de metabolito.

```r
#Creamos el dataframe de metadatos de los metabolitos (las filas)
# Extraemos los nombres de los metabolitos
metabolite_names <- data$Metabolite

# Creamos el dataframe de metadatos de los metabolitos
rowData <- data.frame(
  Metabolite = metabolite_names,
  stringsAsFactors = FALSE
)

# Establecer los nombres de las filas
rownames(rowData) <- metabolite_names
```

```r
head(rowData)
```

```
##                                  Metabolite
## 1-monoolein                      1-monoolein
## 1-monostearin                  1-monostearin
## 2-hydroxybutanoic acid 2-hydroxybutanoic acid
## 2-hydroxyglutaric acid 2-hydroxyglutaric acid
## 2-ketoisocaproic acid   2-ketoisocaproic acid
## 2-monopalmitin                2-monopalmitin
```

```r
#Preparamos la matriz de expresión
# Removemos la columna Metabolite del dataframe "data"
expr_data <- data[, -1]

# Convertimos los valores a numéricos
expr_matrix <- as.matrix(sapply(expr_data, as.numeric))

# Ponemos  los nombres de las filas y columnas
rownames(expr_matrix) <- metabolite_names
colnames(expr_matrix) <- sample_names

expr_matrix[1:5, 1:5]
```

```
##                        LabF_684508 LabF_684512 LabF_684516 LabF_684520
## 1-monoolein                   6047        2902        1452        3428
## 1-monostearin                 9771        6521        1302        2781
## 2-hydroxybutanoic acid       13238       29774        4134        4419
## 2-hydroxyglutaric acid        7160       11501        3202       17238
## 2-ketoisocaproic acid          812        2011         738        2550
##                        LabF_684524
## 1-monoolein                   2985
## 1-monostearin                 5789
## 2-hydroxybutanoic acid       13334
## 2-hydroxyglutaric acid       20376
## 2-ketoisocaproic acid          871
```

```r
# Creamos una columna con las etiquetas "B" o "A" según el grupo
colData$Label <- ifelse(colData$Group == "Before transplantation", "B", "A")
# Crear una nueva columna para los nombres modificados sin alterar SampleID
colData$SampleID_modified <- paste0(colData$SampleID, "_", colData$Label)

# Actualizamo los nombres de las filas en colData
rownames(colData) <- colData$SampleID_modified
# Actualizamos los nombres de las columnas en expr_matrix
colnames(expr_matrix) <- colData$SampleID_modified
colData
```

```
##                   SampleID                  Group Label SampleID_modified
## LabF_684508_A LabF_684508  After transplantation     A     LabF_684508_A
## LabF_684512_A LabF_684512  After transplantation     A     LabF_684512_A
## LabF_684516_A LabF_684516  After transplantation     A     LabF_684516_A
## LabF_684520_A LabF_684520  After transplantation     A     LabF_684520_A
```

```
## LabF_684524_A LabF_684524  After transplantation     A     LabF_684524_A
## LabF_684528_A LabF_684528  After transplantation     A     LabF_684528_A
## LabF_684483_B LabF_684483 Before transplantation     B     LabF_684483_B
## LabF_684487_B LabF_684487 Before transplantation     B     LabF_684487_B
## LabF_684491_B LabF_684491 Before transplantation     B     LabF_684491_B
## LabF_684495_B LabF_684495 Before transplantation     B     LabF_684495_B
## LabF_684499_B LabF_684499 Before transplantation     B     LabF_684499_B
## LabF_684503_B LabF_684503 Before transplantation     B     LabF_684503_B
```

```
#Mostramos los nombres de las columnas de expr_matrix
colnames(expr_matrix)
```

```
##  [1] "LabF_684508_A" "LabF_684512_A" "LabF_684516_A" "LabF_684520_A"
##  [5] "LabF_684524_A" "LabF_684528_A" "LabF_684483_B" "LabF_684487_B"
##  [9] "LabF_684491_B" "LabF_684495_B" "LabF_684499_B" "LabF_684503_B"
```

```
#Comparamos que los nombres coinciden.
all(colnames(expr_matrix) == rownames(colData))
```

```
## [1] TRUE
```

Ahora ya tenemos todo listo para crear el SummarizedExperiment, poruqé tenemos tanto la matriz de expresión con los datos del experimento. Los datos que pueden tener los metabolitos y los metadatos de los experimentos.

Creamos el objeto "se" y lo ejecutamos para ver que todo encaja y guardamos el objeto .rds.

```
# Creamos el objeto SummarizedExperiment
se <- SummarizedExperiment(
  assays = list(counts = expr_matrix),
  rowData = rowData,
  colData = colData
)
# Resumen del objeto SummarizedExperiment
se
```

```
## class: SummarizedExperiment
## dim: 142 12
## metadata(0):
## assays(1): counts
## rownames(142): 1-monoolein 1-monostearin ... xanthine xylose
## rowData names(1): Metabolite
## colnames(12): LabF_684508_A LabF_684512_A ... LabF_684499_B
##   LabF_684503_B
## colData names(4): SampleID Group Label SampleID_modified
```

```
# Guardem l'objecte SummarizedExperiment
saveRDS(se, "metabolomics_se.rds")
```

A continuación haremos un rápido Analisis con el objeto de SummaryExpression creado anteriormente para ver que funciona como deseamo usando Componentes Principales.

```r
# Instlamos y cargamos paquete de ggplot para graficar

if (!requireNamespace("ggplot2", quietly = TRUE))
    install.packages("ggplot2")
if (!requireNamespace("ggfortify", quietly = TRUE))
    install.packages("ggfortify")
library(ggplot2)
```

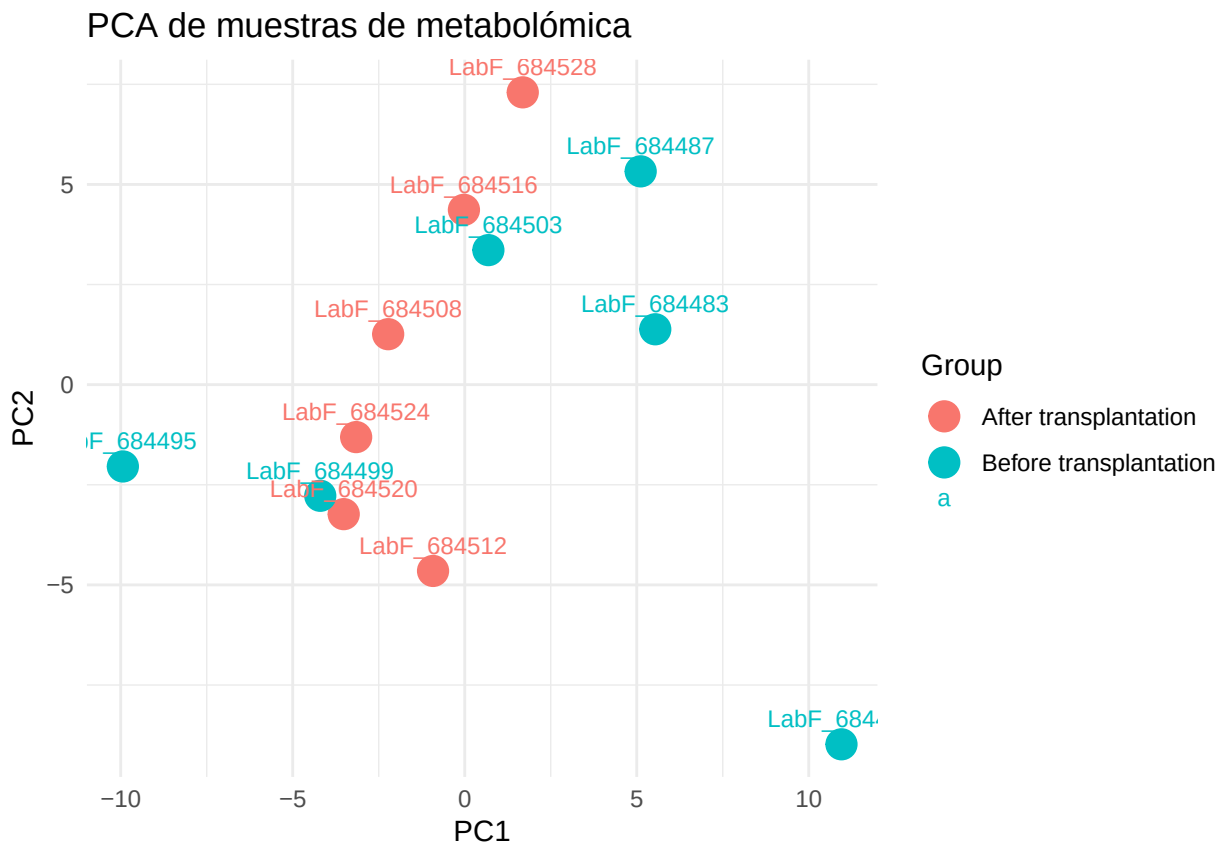
```
## Warning: package 'ggplot2' was built under R version 4.3.3
```

```r
library(ggfortify)
```

```
## Warning: package 'ggfortify' was built under R version 4.3.3
```

```r
# Realizamos PCA.
pca_result <- prcomp(t(assay(se)), scale. = TRUE)
# Crearmos el dataframe para el gráfico plot
pca_data <- data.frame(pca_result$x, Group = colData(se)$Group, SampleID = colData(se)$SampleID)

# Generamos Gráfico del PCA con etiquetas
ggplot(pca_data, aes(x = PC1, y = PC2, color = Group)) +
  geom_point(size = 5) +
  geom_text(aes(label = SampleID), hjust = 0.5, vjust = -1, size = 3) +
  theme_minimal() +
  labs(title = "PCA de muestras de metabolómica", x = "PC1", y = "PC2")
```



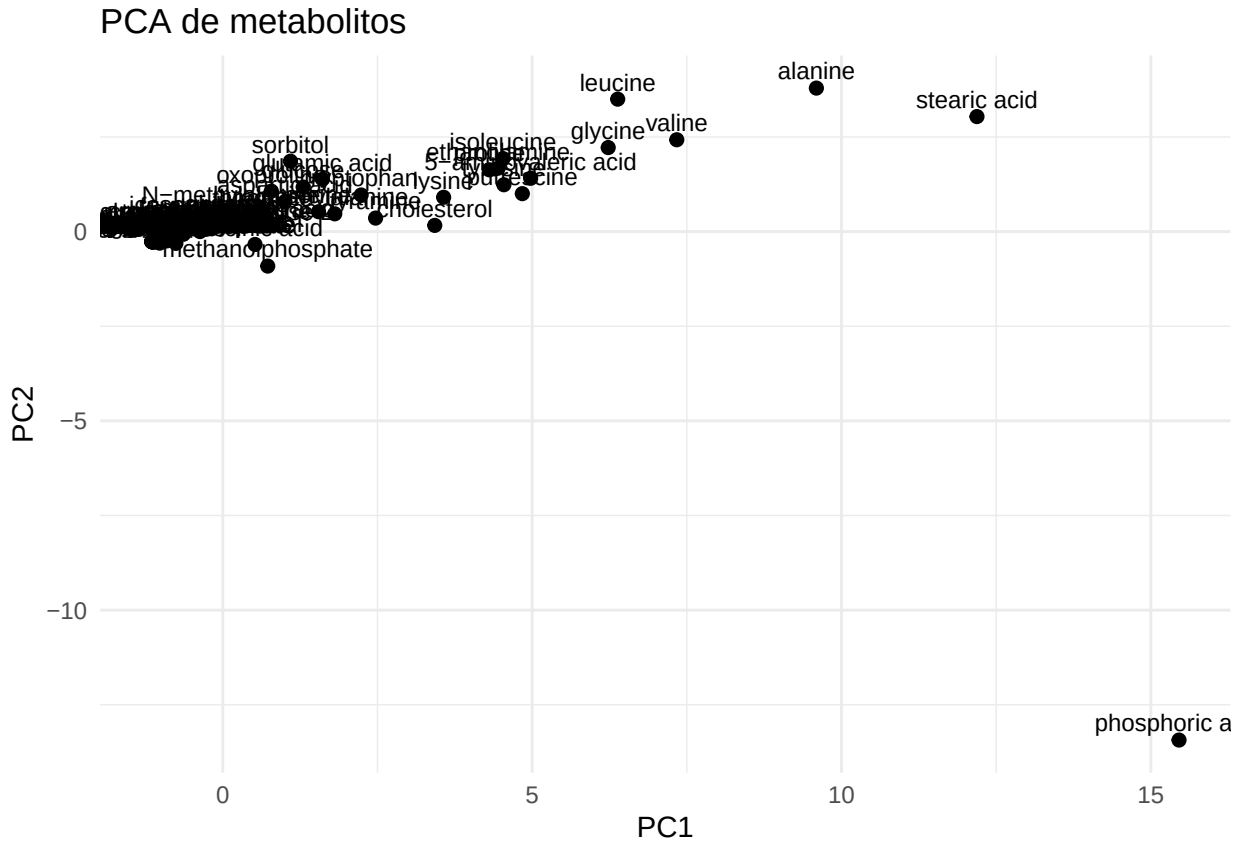
Se pueden observar en el gráfico anterior como las muestras se agrupan y se diferencian entre sí en función del perfil metabolómico

A continuación haremos el mismo analiss de PCA para los metabolitos para los metabolitos

```
# Realizamos PCA sobre los metabolitos
pca_metabolites <- prcomp(assay(se), scale. = TRUE)

# Creamos dataframe para el PCA de metabolitos y añadimos los nombres
pca_metabolites_data <- data.frame(pca_metabolites$x)
pca_metabolites_data$Metabolite <- rownames(pca_metabolites$x)

ggplot(pca_metabolites_data, aes(x = PC1, y = PC2)) +
  geom_point(size = 2) +
  geom_text(aes(label = Metabolite), hjust = 0.5, vjust = -0.5, size = 3) +
  theme_minimal() +
  labs(title = "PCA de metabolitos", x = "PC1", y = "PC2")
```



Como hemos visto que hay un metablito que esta muy alejado del resto vamos a sacarlo acotando la visualización

```
expr_matrix <- assay(se)


metabolite_info <- rowData(se)
metabolite_info
```

```
## DataFrame with 142 rows and 1 column
##                                    Metabolite
##                                   <character>
## 1-monoolein                         1-monoolein
## 1-monostearin                     1-monostearin
## 2-hydroxybutanoic acid 2-hydroxybutanoic acid
## 2-hydroxyglutaric acid 2-hydroxyglutaric acid
## 2-ketoisocaproic acid   2-ketoisocaproic acid
## ...                                       ...
## uric acid                           uric acid
## uridine                               uridine
## valine                                 valine
## xanthine                             xanthine
## xylose                                 xylose
```

Ahora vamos a añadir más metadatos que no hemos añadido con anterioridad a los metabolitos, de esta manera tenemos la info más completa.

```
# LeeMOS todas las líneas del archivo de origen por si a caso.
lines <- readLines(data_url)

# Encontrar¡mos los índices de inicio y fin de la sección de metadatos de metabolitos
metabolites_start <- grep("METABOLITES_START", lines)
metabolites_end <- grep("METABOLITES_END", lines)
# Extraemos las líneas de la sección de metadatos de los metabolitos
metabolites_lines <- lines[(metabolites_start + 1):(metabolites_end - 1)]

# Leemos los metadatos de los metabolitos en un dataframe
metabolites_meta <- read.table(text = metabolites_lines, sep = "\t", header = TRUE, check.names = FALSE

# Mostramos las primeras filas del dataframe con los nuevos datos
head(metabolites_meta)
```

```
##          metabolite_name moverz_quant     ri ri_type pubchem_id inchi_key
## 1            1-monoolein          129 952993   Fiehn    5283468        NA
## 2          1-monostearin          399 959625   Fiehn     107036        NA
## 3 2-hydroxybutanoic acid          131 258175   Fiehn      11266        NA
## 4 2-hydroxyglutaric acid          129 506359   Fiehn         43        NA
## 5  2-ketoisocaproic acid          200 310629   Fiehn         70        NA
## 6         2-monopalmitin          218 889972   Fiehn     123409        NA
##   kegg_id other_id other_id_type
## 1          213963       BinBase
## 2  D01947   202835       BinBase
## 3  C05984   199800       BinBase
## 4  C02630   214409       BinBase
## 5  C00233   213388       BinBase
## 6          233239       BinBase
```

```
# Cargar la librería dplyr
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.3.2
```

```
##
## Attaching package: 'dplyr'

## The following object is masked from 'package:Biobase':
##
##     combine

## The following objects are masked from 'package:GenomicRanges':
##
##     intersect, setdiff, union

## The following object is masked from 'package:GenomeInfoDb':
##
##     intersect

## The following objects are masked from 'package:IRanges':
##
##     collapse, desc, intersect, setdiff, slice, union

## The following objects are masked from 'package:S4Vectors':
##
##     first, intersect, rename, setdiff, setequal, union

## The following objects are masked from 'package:BiocGenerics':
##
##     combine, intersect, setdiff, union

## The following object is masked from 'package:matrixStats':
##
##     count

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
rowData(se) <- DataFrame(Metabolite = rownames(se))

# Función para limpiar y estandarizar nombres
clean_names <- function(x) {
  x <- tolower(x)
  x <- trimws(x)
  x <- gsub("[^a-z0-9]", "", x)
  return(x)
}

# Limpiamos los nombres de los metabolitos en ambos conjuntos de datos
metabolites_meta$clean_name <- clean_names(metabolites_meta$metabolite_name)
```

```r
rowData(se)$clean_name <- clean_names(rownames(se))

# Verificamos coincidencias después de limpiar los nombres
common_metabolites <- intersect(rowData(se)$clean_name, metabolites_meta$clean_name)
cat("Número de metabolitos coincidentes:", length(common_metabolites), "\n")
```

```
## Número de metabolitos coincidentes: 142
```

```r
# Si hay metabolitos que no coinciden, podemos revisarlos
metabolites_no_match <- setdiff(rowData(se)$clean_name, metabolites_meta$clean_name)
if (length(metabolites_no_match) > 0) {
  cat("Metabolitos en 'se' que no coinciden con metadatos:", metabolites_no_match, "\n")
}
# Seleccionamos lascolumnas necesarias de metabolites_meta, excluyendo 'metabolite_name' original
metabolites_meta_subset <- metabolites_meta %>%
  select(-metabolite_name)

# Añadir una columna auxiliar para mantener el orden
rowData(se)$order <- seq_len(nrow(rowData(se)))
rowData_merged <- rowData(se) %>%
  as.data.frame() %>%
  left_join(metabolites_meta_subset, by = "clean_name")

# Se Ordena el resultado según la columnaorder y remover columnas auxiliares
rowData_merged <- rowData_merged %>%
  arrange(order) %>%
  select(-clean_name, -order)

# Actualizar el rowData del SummarizedExperiment
rowData(se) <- rowData_merged

# Aseguramos que los nombres de las filas se mantienen correctamente
rownames(rowData(se)) <- rowData(se)$Metabolite

# Se muestan las primeras filas del rowData actualizado
head(rowData(se))
```

```
## DataFrame with 6 rows and 9 columns
##                                   Metabolite moverz_quant        ri
##                                  <character>    <integer> <integer>
## 1-monoolein                      1-monoolein          129    952993
## 1-monostearin                  1-monostearin          399    959625
## 2-hydroxybutanoic acid 2-hydroxybutanoic acid          131    258175
## 2-hydroxyglutaric acid 2-hydroxyglutaric acid          129    506359
## 2-ketoisocaproic acid   2-ketoisocaproic acid          200    310629
## 2-monopalmitin                2-monopalmitin          218    889972
##                            ri_type pubchem_id inchi_key     kegg_id  other_id
##                          <character>  <integer> <logical> <character> <integer>
## 1-monoolein                    Fiehn    5283468        NA                213963
## 1-monostearin                  Fiehn     107036        NA      D01947    202835
## 2-hydroxybutanoic acid         Fiehn      11266        NA      C05984    199800
## 2-hydroxyglutaric acid         Fiehn         43        NA      C02630    214409
## 2-ketoisocaproic acid          Fiehn         70        NA      C00233    213388
```

```
## 2-monopalmitin               Fiehn     123409        NA               233239
##                         other_id_type
##                           <character>
## 1-monoolein                   BinBase
## 1-monostearin                 BinBase
## 2-hydroxybutanoic acid        BinBase
## 2-hydroxyglutaric acid        BinBase
## 2-ketoisocaproic acid         BinBase
## 2-monopalmitin                BinBase
```

```
# Acceder a la matriz de expresión de metabolitos
expr_matrix <- assay(se)


metabolite_info <- rowData(se)


# Acceder a los metadatos de las muestras
sample_info <- colData(se)
sample_info
```

```
## DataFrame with 12 rows and 4 columns
##                   SampleID                 Group       Label SampleID_modified
##                <character>              <factor> <character>       <character>
## LabF_684508_A LabF_684508  After transplantation           A     LabF_684508_A
## LabF_684512_A LabF_684512  After transplantation           A     LabF_684512_A
## LabF_684516_A LabF_684516  After transplantation           A     LabF_684516_A
## LabF_684520_A LabF_684520  After transplantation           A     LabF_684520_A
## LabF_684524_A LabF_684524  After transplantation           A     LabF_684524_A
## ...                   ...                    ...         ...               ...
## LabF_684487_B LabF_684487 Before transplantation           B     LabF_684487_B
## LabF_684491_B LabF_684491 Before transplantation           B     LabF_684491_B
## LabF_684495_B LabF_684495 Before transplantation           B     LabF_684495_B
## LabF_684499_B LabF_684499 Before transplantation           B     LabF_684499_B
## LabF_684503_B LabF_684503 Before transplantation           B     LabF_684503_B
```

```
# Guardar el objeto para uso futuro
saveRDS(se, "datos_metabolomicos_se.rds")

# Cargar el objeto en otra sesión
se <- readRDS("datos_metabolomicos_se.rds")

se
```

```
## class: SummarizedExperiment
## dim: 142 12
## metadata(0):
## assays(1): counts
## rownames(142): 1-monoolein 1-monostearin ... xanthine xylose
## rowData names(9): Metabolite moverz_quant ... other_id other_id_type
## colnames(12): LabF_684508_A LabF_684512_A ... LabF_684499_B
##   LabF_684503_B
## colData names(4): SampleID Group Label SampleID_modified
```

A continuación subiremos los archivos a GIT: Primero configuraremos para que Rstudio se conecte con mi cuenta de GIT:. Después guardaremos en un repositorio local los archivos necesarios: - El informe al apretar .knit ya se guarda junto con el archivo rmd. en este caso en pdf - Guardaremos el archivo se en formato Rda:

```
save(se, file = "metabolomics_se.Rda")
```

-Generamos el script con solo el código R a través del Rmd.

```
library(knitr)
```

```
## Warning: package 'knitr' was built under R version 4.3.3
```

```
# Extraer el código R del archivo Rmd
purl("PEC 1 Datos Ómicos DZ.Rmd", output = "PEC 1 Datos Ómicos DZ.R", documentation = 0)
```

```
##
##
## processing file: PEC 1 Datos Ómicos DZ.Rmd
```

```
##   |                                                                      |
```

```
## output file: PEC 1 Datos Ómicos DZ.R
```

```
## [1] "PEC 1 Datos Ómicos DZ.R"
```

- Generamos el texto plano de los datos de entrada

Y una vez tenemos todo guardaremos nuestro proyecto en GIT desde Rstudio creando un nuevo proyecto. Enlazandolo con el repositorio de Git y cargando nuestros archivos.

Subiendolo al siguiente repositorio: https://github.com/dzamoraso/zamora_sole_dani_PEC1/