Informe PEC1

Alfonso Ramos Torres

Resumen

La caquexia es un síndrome metabólico complejo asociado a una enfermedad subyacente (como el cáncer) y caracterizado por pérdida de masa muscular, con o sin pérdida de grasa. En este estudio se tomaron un total de 77 muestras de orina recogidas de 47 pacientes con caquexia y 30 pacientes control. Estas muestras se han procesado y analizado utilizando diversas técnicas estadísticas, como la construcción de un objeto SummarizedExperiment y la generación de diversos gráficos, llegando a la conclusión de que existen diferencias entre los dos grupos del estudio en cuanto a la distribución de metabolitos.

Objetivos

- Explorar el conjunto de datos de los metabolitos de pacientes con caquexia y pacientes control para identificar posibles diferencias entre ellos.
- Construir un objeto SummarizedExperiment a partir del conjunto de datos seleccionado.
- Realizar un análisis exploratorio de los datos.

Metodología: El conjunto de datos seleccionados es un archivo en formato .csv que contiene en la primera columna el ID de los pacientes, en la segunda columna si presentan o no caquexia y a partir de la tercera columna en adelante los tipos de metabolitos analizados, siendo cada fila un paciente distinto. Se ha utilizado este dataset debido a la facilidad que presenta para trabajar con él, ya que no tiene valores nulos, todos sus valores son numéricos y presenta dos grupos facilmente diferenciables, haciendo que los análisis se ejecuten con facilidad.

Estos datos, aunque son sencillos de analizar, requieren de un procesamiento previo para ajustarlos a las características que tiene un objeto SummarizedExperiment. Esto se debe a que, por defecto, se espera que las filas representen las características (los metabolitos) y las columnas las muestras, por lo que necesitamos transponer los datos para que concuerde con el estándar.

En relación con lo anterior, se ha empleado una serie de herramientas, como los paquetes en R, para realizar el análisis exploratorio del conjunto de datos. Entre ellos destacamos el paquete SummarizedExperiment para la creación del objeto SummarizedExperiment, el paquete ComplexHeatmap para representar un mapa de calor de la expresión de los metabolitos en los distintos pacientes y otras representaciones como el gráfico boxplot.

Todas estas herramientas, así como la justificación de los pasos dados, viene recogida en el archivo del análisis.

Resultados

Instalamos y cargamos el paquete "SummarizedExperiment" tal y como nos indica en la página de Bioconductor. A continuación, cargamos el conjunto de datos seleccionado y mostramos las primeras filas del mismo.

```
if (!require("BiocManager", quietly = TRUE))
    install.packages("BiocManager")
```

```
## Warning: package 'BiocManager' was built under R version 4.4.3
```

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

```
## Bioconductor version 3.20 (BiocManager 1.30.25), R 4.4.2 (2024-10-31 ucrt)
## Warning: package(s) not installed when version(s) same as or greater than current; use
##   `force = TRUE` to re-install: 'SummarizedExperiment'
## Old packages: 'xfun', 'lattice', 'mgcv'
```

```r
library(SummarizedExperiment)
```

```
## Cargando paquete requerido: MatrixGenerics
## Cargando paquete requerido: matrixStats
## Warning: package 'matrixStats' was built under R version 4.4.3
##
## Adjuntando el paquete: '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
## Cargando paquete requerido: GenomicRanges
## Cargando paquete requerido: stats4
## Cargando paquete requerido: BiocGenerics
##
## Adjuntando el paquete: '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, saveRDS, setdiff,
##     table, tapply, union, unique, unsplit, which.max, which.min
## Cargando paquete requerido: S4Vectors
```

```
##
## Adjuntando el paquete: 'S4Vectors'

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

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

## Cargando paquete requerido: IRanges

##
## Adjuntando el paquete: 'IRanges'

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

## Cargando paquete requerido: GenomeInfoDb

## Cargando paquete requerido: Biobase

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

##
## Adjuntando el paquete: 'Biobase'

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

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

```r
# Leemos el archivo CSV
raw_data <- read.csv("human_cachexia.csv")
head(raw_data)
```

```
##    Patient.ID Muscle.loss X1.6.Anhydro.beta.D.glucose X1.Methylnicotinamide
## 1     PIF_178    cachexic                       40.85                 65.37
## 2     PIF_087    cachexic                       62.18                340.36
## 3     PIF_090    cachexic                      270.43                 64.72
## 4 NETL_005_V1    cachexic                      154.47                 52.98
## 5     PIF_115    cachexic                       22.20                 73.70
## 6     PIF_110    cachexic                      212.72                 31.82
##   X2.Aminobutyrate X2.Hydroxyisobutyrate X2.Oxoglutarate X3.Aminoisobutyrate
## 1            18.73                 26.05           71.52             1480.30
## 2            24.29                 41.68           67.36              116.75
## 3            12.18                 65.37           23.81               14.30
## 4           172.43                 74.44         1199.91              555.57
## 5            15.64                 83.93           33.12               29.67
## 6            18.36                 80.64           47.94               17.46
##   X3.Hydroxybutyrate X3.Hydroxyisovalerate X3.Indoxylsulfate
```

```
## 1              56.83                10.07            566.80
## 2              43.82                79.84            368.71
## 3               5.64                23.34            665.14
## 4             175.91                25.03            411.58
## 5              76.71                69.41            165.67
## 6              31.82                35.16            183.09
##   X4.Hydroxyphenylacetate Acetate Acetone Adipate Alanine Asparagine Betaine
## 1                  120.30  126.47    9.49   38.09  314.19     159.17  109.95
## 2                  432.68  212.72   11.82  327.01  871.31     157.59  244.69
## 3                  292.95  314.19    4.44  131.63  464.05      89.12  116.75
## 4                  214.86   37.34  206.44  144.03  589.93     273.14  278.66
## 5                   97.51  407.48   44.26   15.03 1118.79      42.52  391.51
## 6                  132.95   81.45   14.44   25.28  237.46     157.59   66.69
##   Carnitine  Citrate Creatine Creatinine Dimethylamine Ethanolamine Formate
## 1    265.07  3714.50   196.37   16481.60        632.70       645.48  441.42
## 2    120.30  2617.57   212.72   15835.35        607.89       487.85  252.14
## 3     25.03   862.64   221.41   24587.66        735.10       407.48  249.64
## 4    200.34 13629.61    85.63   20952.22       1064.22       820.57  468.72
## 5     84.77   854.06   105.64    6768.26        242.26       365.04  114.43
## 6     40.04  1958.63   200.34   15677.78        614.00       459.44  314.19
##   Fucose Fumarate Glucose Glutamine Glycine Glycolate Guanidoacetate Hippurate
## 1 336.97     7.69  395.44    871.31 2038.56    685.40         154.47   4582.50
## 2 198.34    18.92 8690.62    601.85 1107.65    651.97         109.95   1737.15
## 3 186.79     7.10 1352.89    301.87  620.17    141.17         183.09   4315.64
## 4 407.48    96.54  862.64   1685.81 5064.45     70.81         102.51    757.48
## 5  26.05    19.69 6836.29    432.68  395.44     26.58          52.98   1152.86
## 6 123.97     5.05  512.86    298.87  482.99    428.38          57.97   3568.85
##   Histidine Hypoxanthine Isoleucine Lactate Leucine Lysine Methylamine
## 1    925.19        97.51       5.58  106.70   42.10 146.94       52.46
## 2    845.56        82.27       8.17  368.71   77.48 284.29       23.57
## 3    284.29       114.43       9.30  749.95   31.50  97.51       18.73
## 4   1043.15       223.63      37.71  368.71  103.54 290.03       48.91
## 5    327.01        66.69      40.04 3640.95  101.49 122.73       27.94
## 6    459.44        62.80       8.17  113.30   28.79 120.30       36.97
##   Methylguanidine N.N.Dimethylglycine O.Acetylcarnitine Pantothenate
## 1            9.97               23.34             52.98        25.79
## 2            7.69               87.36             50.40       186.79
## 3            4.66               24.53              5.58       145.47
## 4          141.17               40.04            254.68        42.52
## 5            5.31               46.06             45.60        74.44
## 6           43.38               24.29             13.46        35.52
##   Pyroglutamate Pyruvate Quinolinate  Serine Succinate Sucrose Tartrate Taurine
## 1        437.03    21.12      165.67  284.29    154.47   45.15    97.51 1919.85
## 2        437.03    36.97       72.97  391.51    244.69  459.44    32.79 1261.43
## 3        713.37    29.37      192.48  295.89    142.59  160.77    16.28 4272.69
## 4        566.80    64.07       86.49 1248.88    144.03  111.05   837.15 1525.38
## 5        184.93    12.30       38.09  206.44     68.72   75.19     4.53  468.72
## 6        432.68    32.79      112.17  387.61     33.45  336.97    24.05 2059.05
##   Threonine Trigonelline Trimethylamine.N.oxide Tryptophan Tyrosine Uracil
## 1    184.93       943.88               2121.76     259.82   290.03 111.05
## 2    198.34       208.51                639.06      83.10   167.34  46.99
## 3    109.95       192.48               1152.86      82.27    60.34  31.50
## 4    376.15       992.27               1450.99     235.10   323.76  30.57
## 5     64.07        86.49                172.43     103.54   142.59  44.26
```

```
## 6    105.64       862.64                880.07     239.85   127.74  29.67
##   Valine  Xylose cis.Aconitate myo.Inositol trans.Aconitate pi.Methylhistidine
## 1  86.49   72.24        237.46       135.64           51.94             157.59
## 2 109.95  192.48        333.62       376.15          217.02             307.97
## 3  59.15 2164.62        330.30        86.49           58.56             145.47
## 4 102.51  125.21       1863.11       247.15           75.94             249.64
## 5 160.77  186.79        101.49       749.95           98.49              84.77
## 6  36.97   89.12        287.15       129.02          121.51             399.41
##   tau.Methylhistidine
## 1              160.77
## 2              130.32
## 3               83.93
## 4              254.68
## 5               79.84
## 6               68.72
```

En primer lugar, realizamos la construcción del objeto SummarizedExperiment:

Como podemos observar, las dos primeras columnas (Patient.ID y Muscle.loss) muestran el identificador único de cada paciente y si pertenecen al grupo control o al grupo que presentan caquexia, mientras que las columnas posteriores muestran las mediciones de los distintos metabolitos. Por tanto, procedemos a extraer las dos primeras columnas (los metadatos), utilizando el ID como etiqueta única para cada fila (como nombre de fila):

```r
# Extraemos los metadatos
colData <- raw_data[, 1:2]

# Asignamos los ID a los nombres de las filas
rownames(colData) <- colData[,"Patient.ID"]

# Eliminamos la columna redundante para que no aparezca información duplicada y
# mantenemos el formato de data frame con la opción "drop = FALSE"
colData <- colData[, -1, drop = FALSE]

head(colData)
```

```
##             Muscle.loss
## PIF_178        cachexic
## PIF_087        cachexic
## PIF_090        cachexic
## NETL_005_V1    cachexic
## PIF_115        cachexic
## PIF_110        cachexic
```

Ahora eliminamos las dos primeras columnas para construir una matriz que contenga solo las mediciones de los distintos metabolitos:

```r
# Extraemos las mediciones de los metabolitos
assay_data <- as.matrix(raw_data[, -c(1, 2)])

# Comprobamos que hemos extraído correctamente las dos primeras columnas
dim(raw_data)
```

```
## [1] 77 65
```

```r
dim(assay_data)
```

```
## [1] 77 63
```

Por defecto, en un objeto SummarizedExperiment se espera que las filas representen las características (los metabolitos) y las columnas las muestras, por lo que necesitamos transponer nuestros datos:

```
# Transponemos la matriz
assay_data_t <- t(assay_data)

# Asignamos los nombres de las filas a partir de los nombres de los metabolitos
rownames(assay_data_t) <- colnames(raw_data)[-c(1,2)]

# Asiignamos los nombres de las columnas usando los ID
colnames(assay_data_t) <- rownames(colData)

# Verificamos que se muestra todo correctamente
assay_data_t[1:5, 1:5]
```

```
##                             PIF_178 PIF_087 PIF_090 NETL_005_V1 PIF_115
## X1.6.Anhydro.beta.D.glucose   40.85   62.18  270.43      154.47   22.20
## X1.Methylnicotinamide         65.37  340.36   64.72       52.98   73.70
## X2.Aminobutyrate              18.73   24.29   12.18      172.43   15.64
## X2.Hydroxyisobutyrate         26.05   41.68   65.37       74.44   83.93
## X2.Oxoglutarate               71.52   67.36   23.81     1199.91   33.12
```

Verificamos que los nombres de las columnas en la matriz transpuesta (los ID de los pacientes) coincidan con los nombres de las filas en "colData", ya que usamos estos IDs para etiquetar las muestras:

```
# Mostramos los nombres de las columnas en assay_data_t
print(colnames(assay_data_t))
```

```
##  [1] "PIF_178"      "PIF_087"      "PIF_090"      "NETL_005_V1"  "PIF_115"
##  [6] "PIF_110"      "NETL_019_V1"  "NETCR_014_V1" "NETCR_014_V2" "PIF_154"
## [11] "NETL_022_V1"  "NETL_022_V2"  "NETL_008_V1"  "PIF_146"      "PIF_119"
## [16] "PIF_099"      "PIF_162"      "PIF_160"      "PIF_113"      "PIF_143"
## [21] "NETCR_007_V1" "NETCR_007_V2" "PIF_137"      "PIF_100"      "NETL_004_V1"
## [26] "PIF_094"      "PIF_132"      "PIF_163"      "NETCR_003_V1" "NETL_028_V1"
## [31] "NETL_028_V2"  "NETCR_013_V1" "NETL_020_V1"  "NETL_020_V2"  "PIF_192"
## [36] "NETCR_012_V1" "NETCR_012_V2" "PIF_089"      "NETCR_002_V1" "PIF_179"
## [41] "PIF_114"      "NETCR_006_V1" "PIF_141"      "NETCR_025_V1" "NETCR_025_V2"
## [46] "NETCR_016_V1" "PIF_116"      "PIF_191"      "PIF_164"      "NETL_013_V1"
## [51] "PIF_188"      "PIF_195"      "NETCR_015_V1" "PIF_102"      "NETL_010_V1"
## [56] "NETL_010_V2"  "NETL_001_V1"  "NETCR_015_V2" "NETCR_005_V1" "PIF_111"
## [61] "PIF_171"      "NETCR_008_V1" "NETCR_008_V2" "NETL_017_V1"  "NETL_017_V2"
## [66] "NETL_002_V1"  "NETL_002_V2"  "PIF_190"      "NETCR_009_V1" "NETCR_009_V2"
## [71] "NETL_007_V1"  "PIF_112"      "NETCR_019_V2" "NETL_012_V1"  "NETL_012_V2"
## [76] "NETL_003_V1"  "NETL_003_V2"
```

```
# Mostramos los nombres de las filas de colData
print(rownames(colData))
```

```
##  [1] "PIF_178"      "PIF_087"      "PIF_090"      "NETL_005_V1"  "PIF_115"
##  [6] "PIF_110"      "NETL_019_V1"  "NETCR_014_V1" "NETCR_014_V2" "PIF_154"
## [11] "NETL_022_V1"  "NETL_022_V2"  "NETL_008_V1"  "PIF_146"      "PIF_119"
## [16] "PIF_099"      "PIF_162"      "PIF_160"      "PIF_113"      "PIF_143"
## [21] "NETCR_007_V1" "NETCR_007_V2" "PIF_137"      "PIF_100"      "NETL_004_V1"
## [26] "PIF_094"      "PIF_132"      "PIF_163"      "NETCR_003_V1" "NETL_028_V1"
## [31] "NETL_028_V2"  "NETCR_013_V1" "NETL_020_V1"  "NETL_020_V2"  "PIF_192"
## [36] "NETCR_012_V1" "NETCR_012_V2" "PIF_089"      "NETCR_002_V1" "PIF_179"
## [41] "PIF_114"      "NETCR_006_V1" "PIF_141"      "NETCR_025_V1" "NETCR_025_V2"
```

```
## [46] "NETCR_016_V1" "PIF_116"      "PIF_191"      "PIF_164"      "NETL_013_V1"
## [51] "PIF_188"      "PIF_195"      "NETCR_015_V1" "PIF_102"      "NETL_010_V1"
## [56] "NETL_010_V2"  "NETL_001_V1"  "NETCR_015_V2" "NETCR_005_V1" "PIF_111"
## [61] "PIF_171"      "NETCR_008_V1" "NETCR_008_V2" "NETL_017_V1"  "NETL_017_V2"
## [66] "NETL_002_V1"  "NETL_002_V2"  "PIF_190"      "NETCR_009_V1" "NETCR_009_V2"
## [71] "NETL_007_V1"  "PIF_112"      "NETCR_019_V2" "NETL_012_V1"  "NETL_012_V2"
## [76] "NETL_003_V1"  "NETL_003_V2"
```

```
# Construimos el objeto SummarizedExperiment
se <- SummarizedExperiment(
  assays = list(counts = assay_data_t),
  colData = colData
)

# Mostramos la estructura del SummarizedExperiment
se
```

```
## class: SummarizedExperiment
## dim: 63 77
## metadata(0):
## assays(1): counts
## rownames(63): X1.6.Anhydro.beta.D.glucose X1.Methylnicotinamide ...
##   pi.Methylhistidine tau.Methylhistidine
## rowData names(0):
## colnames(77): PIF_178 PIF_087 ... NETL_003_V1 NETL_003_V2
## colData names(1): Muscle.loss
```

Las principales diferencias entre SummarizedExperiment y ExpressionSet es que este último maneja una única matriz de datos y los metadatos se almacenan en un único objeto, mientras que el SummarizedExperiment permite almacenar múltiples ensayos, es más moderno y se integra mejor con las herramientas actuales y puede distinguir entre la información de las muestras y de las variables.

A continuación, procedemos a realizar un análisis exploratorio del conjunto de datos.En primer lugar vamos a extraer la matriz de ensayos (las mediciones de los metabolitos) y observar sus estadísticas básicas:

```
# Extraemos la matriz de datos del objeto SummarizedExperiment
# Debemos recordar que las filas corresponden con los metabolitos y las columnas
# con las muestras (los IDs)
assay_data <- assay(se, "counts")

# Comprobamos las dimensiones de la matriz, que nos dirá el número de metabolitos (filas) y muestras (c
dim(assay_data)
```

```
## [1] 63 77
```

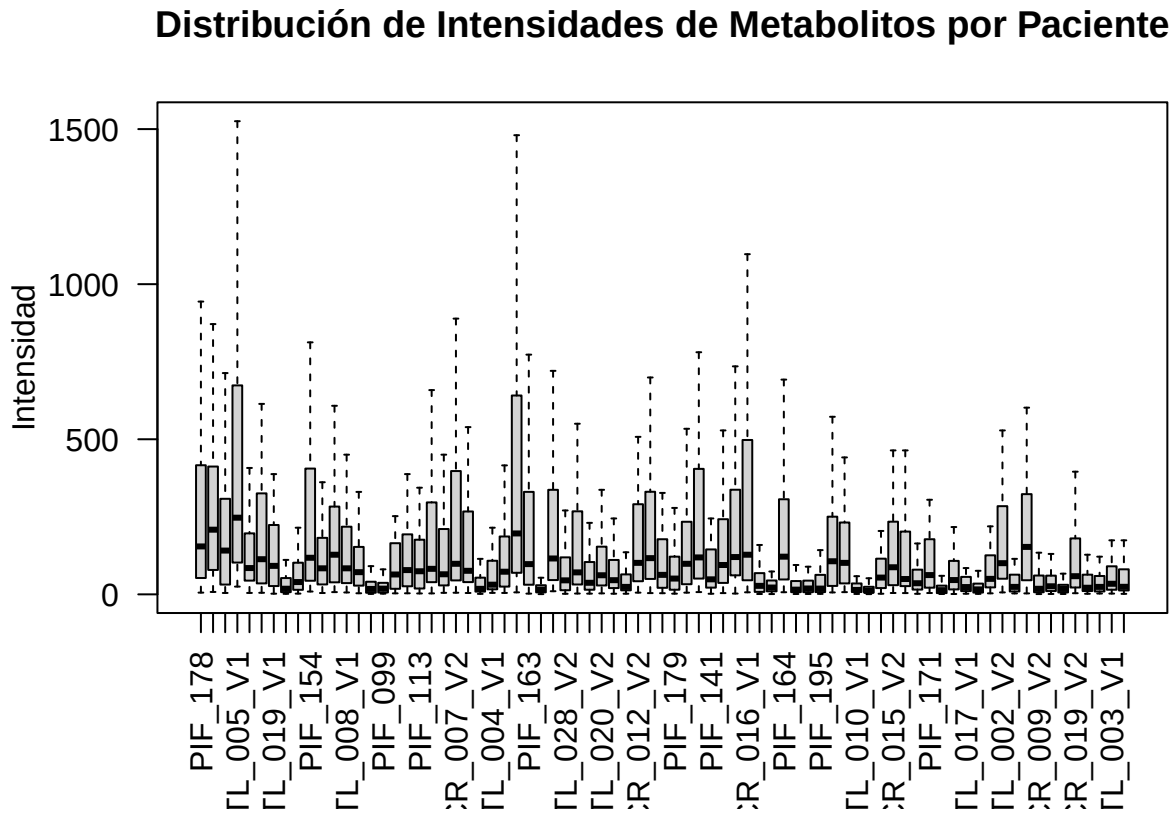
```
# Mostramos un resumen general de todos los valores
summary(as.vector(assay_data))
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##     0.79    17.46    51.42   347.37   160.77 33860.35
```

Para tener una idea general de cómo es la distribución de metabolitos por paciente realizamos un gráfico de boxplot, en el cuál representaremos los valores de las mediciones de metabolitos de cada paciente, observando de esta forma si siguen o no una distribución similar:

```
# Generamos un gráfico de boxplot
boxplot(assay_data,
        outline = FALSE, # Eliminamos la visualización de los valores atípicos para que se pueda observ
        las = 2, # Giramos las etiquetas para visualizar mejor los nombres
```

```
        main = "Distribución de Intensidades de Metabolitos por Paciente",
        ylab = "Intensidad")
```



Como podemos observar, las medianas se distribuyen de forma parecida salvo en ciertos puntos, y que el rango intercuartil es muy diverso, por lo que las muestras no comparten la misma consistencia en la dispersión de sus valores.

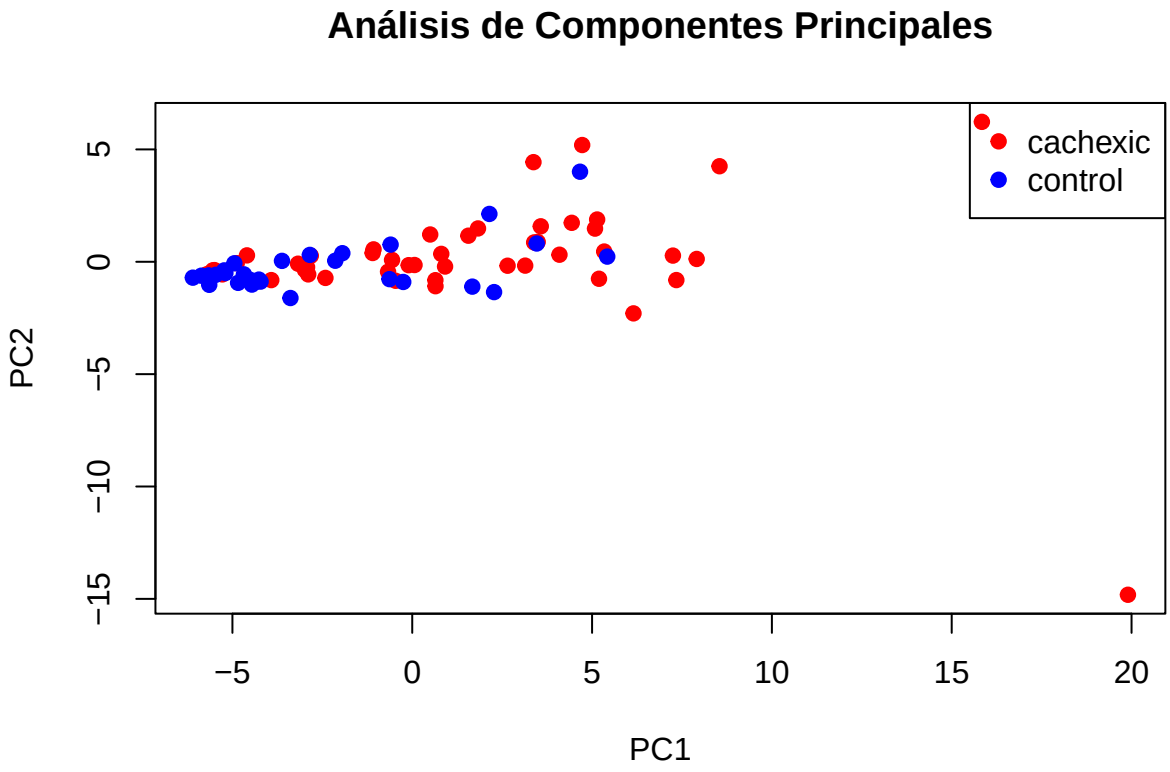
A continuación, procedemos con el análisis de componentes principales (PCA):

```
# Realizamos el Análisis de Componentes Principales
pca <- prcomp(t(assay_data), scale. = TRUE) # Escalamos los datos

# Generamos un vector para asignar colores según al grupo que pertenezcan
# El grupo con caquexia tendrá color rojo y el grupo control azul
group_colors <- ifelse(colData(se)$Muscle.loss == "cachexic", "red", "blue")

# Generamos el gráfico del PCA
plot(pca$x[, 1:2], # Extraemos las coordenadas de las muestras en los dos
                   # componentes principales (PC1 y PC2)
     col = group_colors, # asignamos los colores a los puntos
     pch = 19, # Representamos los puntos con un círculo relleno
     xlab = "PC1",
     ylab = "PC2",
     main = "Análisis de Componentes Principales")
legend("topright",
       legend = c("cachexic", "control"), # añadimos una leyenda
       col = c("red", "blue"),
```

```
       pch = 19)
```



Utilizamos el PCA para reducir la elevada dimensionalidad del conjunto de datos empleado, extrayendolos patrones de variación más relevantes. Representamos los dos componentes principales y observamos como los puntos que representan los grupos con caquexia y control (representados con los diferentes colores) se alejan entre sí, denotando que existe un patrón distinto en la composición metabólica de los grupos estudiados.

Realizamos un "Heatmap" para observar patrones de expresión entre metabolitos:

```
library(ComplexHeatmap)
```

```
## Cargando paquete requerido: grid
```

```
## ========================================
## ComplexHeatmap version 2.22.0
## Bioconductor page: http://bioconductor.org/packages/ComplexHeatmap/
## Github page: https://github.com/jokergoo/ComplexHeatmap
## Documentation: http://jokergoo.github.io/ComplexHeatmap-reference
##
## If you use it in published research, please cite either one:
## - Gu, Z. Complex Heatmap Visualization. iMeta 2022.
## - Gu, Z. Complex heatmaps reveal patterns and correlations in multidimensional
##     genomic data. Bioinformatics 2016.
##
##
## The new InteractiveComplexHeatmap package can directly export static
## complex heatmaps into an interactive Shiny app with zero effort. Have a try!
##
```

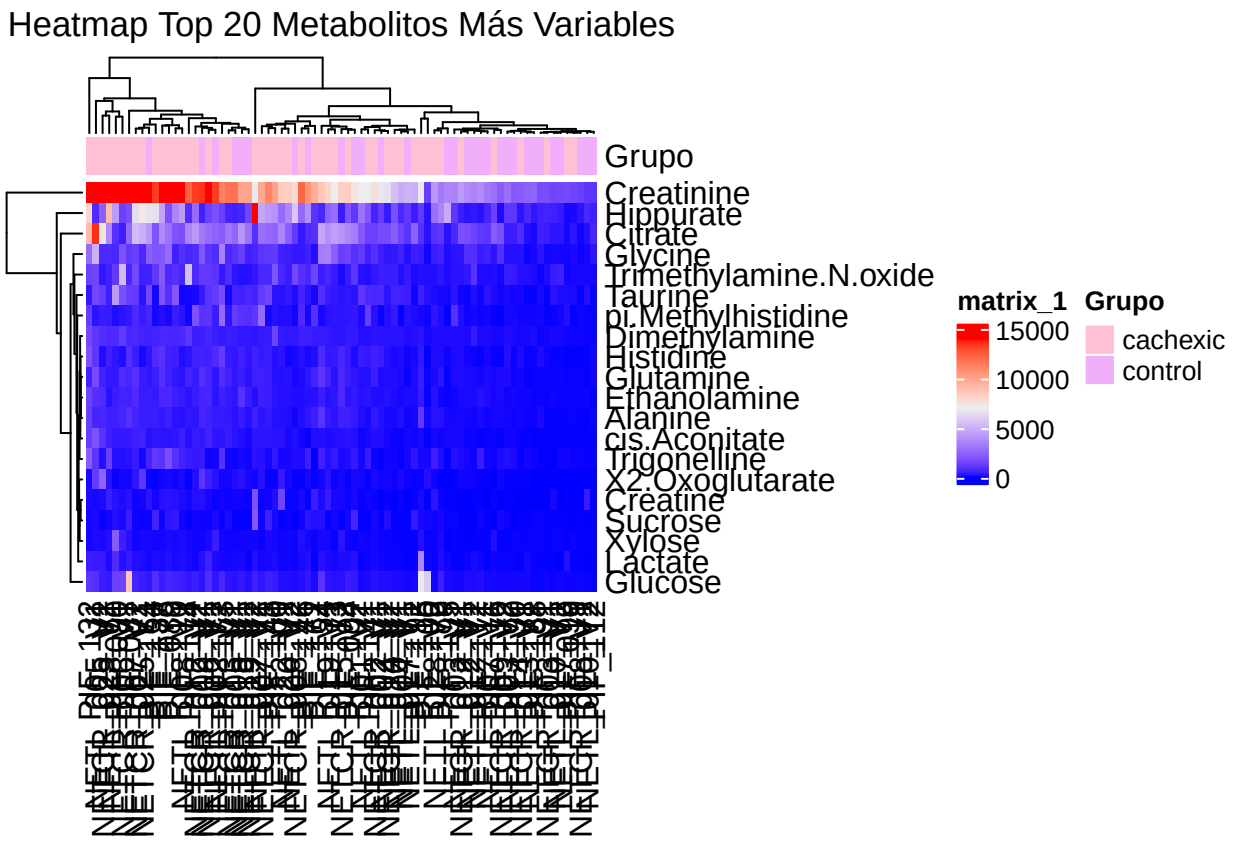
```
## This message can be suppressed by:
##   suppressPackageStartupMessages(library(ComplexHeatmap))
## ========================================
```

```
varianza <- apply(assay_data, 1, var)
top20_metabolites <- names(sort(varianza, decreasing = TRUE))[1:20]

anotacion_columnas <- data.frame(Grupo = colData(se)$Muscle.loss)
rownames(anotacion_columnas) <- colnames(assay_data)


Heatmap(assay_data[top20_metabolites, ],
        top_annotation = HeatmapAnnotation(df = anotacion_columnas),
        column_title = "Heatmap Top 20 Metabolitos Más Variables")
```

```
## The automatically generated colors map from the 1^st and 99^th of the
## values in the matrix. There are outliers in the matrix whose patterns
## might be hidden by this color mapping. You can manually set the color
## to `col` argument.
##
## Use `suppressMessages()` to turn off this message.
```



Debido a la dificultad de visualización del heatmap, lo generamos como un pdf para poder visualizarlo mejor (Ver archivo "heatmap_gande.pdf).

Este mapa de calor nos permite visualizar de forma conjunta la expresión de los 20 metabolitos más variables a lo largo de las muestras. La escala de color indica la intensidad de cada metabolito, representando los tonos más intensos a valores más elevados y los tonos menos intensos a valores más bajos. El Heatmap nos

revela que, como podemos observar, existen distintos patrones de intensidad en las distintas muestras para un mismo metabolito, lo que nos indica que hay diferencias entre los grupos “caquexia” y “control”.

Referencias

Este es el enlace del repositorio de GitHub: https://github.com/Alframtor/Ramos-Torres-Alfonso-PEC1