Análisis de datos ómicos: PEC 1

Florian Bouchet

Table of Contents

1. **Abstract**

Caquexia es un síndrome metabólico caracterizado por perdida de masa muscular. En este trabajo usamos datos previamente publicados por Eisner et al. (2010 Metabolomics) y que corresponden a valores brutos de concentración de 63 metabolitos recogidos a partir de muestras de orina de 77 pacientes. De esos pacientes, 47 presentaban la patología Caquexia mientras 30 eran sanos. Los datos fueron descargados desde el repositorio GitHub “metaboData” de la página (usuario) “nutrimetabolomics”. Creamos primero un objeto de clase SummarizedExperiment a partir de esos datos y, a continuación, llevamos a cabo un análisis exploratorio incluyendo un preprocesamiento de los datos y varios análisis univariantes y multivariantes. Los datos fueron log-transformados y estandarizados para análisis posteriores. Los análisis multivariantes muestran en general pocas diferencias entre los dos grupos de pacientes, y, agrupamientos no muy claros de metabolitos. En cambio, evidenciamos un cierto efecto Batch al considerar los tipos de muestras.

2. **Objetivos**

Los objetivos y/o preguntas de este trabajo son los siguientes:

- Justificar la elección de este conjunto de datos
- Explicar las diferencias entre la clase *SummarizedExperiment* y la clase *ExpressionSet*
- Crear un objeto de clase *SummarizedExperiment* a partir de los datos eligidos, explicando el proceso

- Llevar a cabo un análisis exploratorio de los datos
- Interpretar los resultados de este análisis exploratorio desde el punto de vista biológico
- Crear un repositorio de GitHub que contiene el informe, el objeto de clase *SummarizedExperiment* en formato .Rda, el código R correspondiente al análisis exploratorio, los datos en formato .txt, y los metadatos con una breve descripción en formato .md.

3. **Métodos**

El conjunto de datos se descargó a partir de los raw data disponibles en el repositorio GitHub "metaboData" del proyecto "nutrimetabolomics". Para obtener el enlace https a los raw data se tiene que entrar en la página principal del repositorio e ir a los datos haciendo Datasets –> 2024-Cachexia –> human_cachexia.csv. con el enlace se puede leer directamente en archivo .csv en R. El paquete de Bioconductor *SummarizedExperiment* se usó para crear el objeto de clase correspondiente que tras creación contiene, entre otros, la matriz de datos con las variables (los metabolitos) en líneas y las observaciones (las muestras, es decir los pacientes) en columnas. El paquete *POMA* (Bioconductor) fue usado para parte del preprocesamiento de los datos. Este paquete es útil justamente para crear objetos de clase *SummarizedExperiment* (aquí no lo usamos para esto puesto que lo hacemos mediante la función correspondiente de *SummarizedExperiment*), y para preprocesar los datos contenidos en tal objeto. Usamos *POMA* para comprobar la presencia de ceros y valores faltantes y imputarlos, y para aplicar una transformación log. Los datos fueron también estandarizados tras log-transformación. Usamos boxplots para visualizar los datos antes y después de aplicar esas modificaciones. Posteriormente, empleamos funciones del paquete *mixOmics* (originalmente basado en propias funciones de *POMA*) para realizar análisis de componentes principales (ACP) con el fin de explorar la variación entre los pacientes, y funciones del paquete *made4* para hacer un análisis de correspondencias entre metabolitos y pacientes. Por otro lado, calculamos distancias euclidianas entre los pacientes para poder realizar un análisis de escalamiento multidimensional (paquete *MASS*) y de conglomerados (agrupación jerárquica), así permitiendo tener varios resultados sobre agrupamientos y diferencias entre pacientes. Más detalles además de análisis suplementarios se pueden encontrar en Anexos.

4. **Resultados**

4.1. ***Elección de los datos***

El conjunto de datos Cachexia es interesante por tener más muestras que variables (aunque tampoco mucho, 77 pacientes por 63 metabolitos), asegurando que los análisis multivariantes estén bastante robustos. También, uno puede notar que en esos datos los pacientes no solo pertenecen a dos grupos (*cachexic* vs. *control*), pero que también tienen

tipos distintos de identificadores lo que implica tipos distintos de muestras. El interés es entonces de poder testar por un potencial efecto Batch debido a los tipos pacientes. Finalmente, siguiendo la descripción de los datos, sabemos que todas las variables son del mismo tipo (númericos) y independientes, lo que facilitará los análisis.

4.2. ***SummarizedExperiment vs. ExpressionSet***

La primera diferencia mayor entre las clase *SummarizedExperiment* y *ExpressionSet* es que *SummarizedExperiment* es más flexible respecto a lo que representan las líneas. Con *SummarizedExperiment* las líneas de la matriz de datos pueden representar rangos genomicos, con lo cual los metadatos de las líneas serán objetos de clase *GRanges*. Con un objeto de clase *ExpressionSet*, no se puede usar rangos genomicos. La segunda diferencia es que con la clase *SummarizedExperiment* se pueden guardar varias matrices de datos con los metados de cada, en un solo objeto. Con *ExpressionSet* solo se puede tener una sola matriz de datos en un objeto.

4.3. ***Creación del objeto de clase SummarizedExperiment***

Primero, descargamos los datos y exploramos un poco lo que tenemos.

```
## Descarga de los datos
cachexiadata <-
read.csv("https://raw.githubusercontent.com/nutrimetabolomics/metaboData/refs
/heads/main/Datasets/2024-Cachexia/human_cachexia.csv")
```

En esta tabla los pacientes están en líneas y los metabolitos en columnas. La primera columna corresponde al identificador del paciente y la segunda al grupo. El resto de columnas son las concentraciones para cada metabolito. Tenemos bien dos grupos, *cachexic* y *control*. Tenemos bien 63 metabolitos y 77 pacientes. Se nota que hay tres tipos de pacientes: Los *PIF*, los *NETL* y los *NETCR*. Podemos crear una segunda variable de agrupamiento, *identif*, con los tipos pacientes.

```
## Segunda variable de agrupamiento
identif <- strsplit(cachexiadata[, 1], "_")
identif <- unlist(lapply(identif, FUN = function(x) (x[1])))
```

Ahora, podemos pasar a la creación del objeto de clase *SummarizedExperiment* mediante el constructor *SummarizedExperiment()*. Para ello necesitamos la matriz con los valores de concentración, con los pacientes en columnas y los metabolitos en líneas (assays), y, como mínimo, los metadatos de las columnas (colData). En nuestro caso esos metadatos corresponden al grupo al cual pertenece cada paciente. No tenemos información sobre los metabolitos, entonces no habrá metadatos de líneas.

```
## Cargamos el paquete SummarizedExperiment
suppressPackageStartupMessages(library(SummarizedExperiment, quietly = T))
## Matriz con los valores de concentración
valores <- t(as.matrix(data.frame(cachexiadata[, -c(1, 2)],
    row.names = cachexiadata[, 1])))
## Dataframe con los metadatos de las columnas
```

```r
columnasdatos <- data.frame(Group = cachexiadata[, 2],
    row.names = cachexiadata[, 1])
## Ahora podemos crear el objeto de clase SummarizedExperiment
sumexpcachexia <- SummarizedExperiment(assays = valores,
    colData = columnasdatos)
```

4.4. Análisis exploratorio de los datos

4.4.1. análisis univariante y pre-procesamiento

Hay que gestionar los valores faltantes (NAs) y valores que valen 0. Los valores que valen 0 pueden convertirse a NAs. Posteriormente se pueden quitar los NAs, y quitar también columnas que tienen más de 20% de NAs.

```r
## Cargamos el paquete POMA
suppressPackageStartupMessages(library(POMA, quietly = T))
## PomaImpute permite convertir los ceros en NAs, y, por defecto, quitar los
## NAs y quitar las columnas que tienen más de 20% de NAs
sumexpcachexiamod <- PomaImpute(sumexpcachexia, zeros_as_na = T)

## No missing values detected

## 0 features removed.
```

Ahora que hemos gestionado los NAs y los ceros (no había), miramos la distribución de los datos mediante un boxplot, es decir la distribución de las concentraciones en metabolitos para cada paciente (Fig. 1).

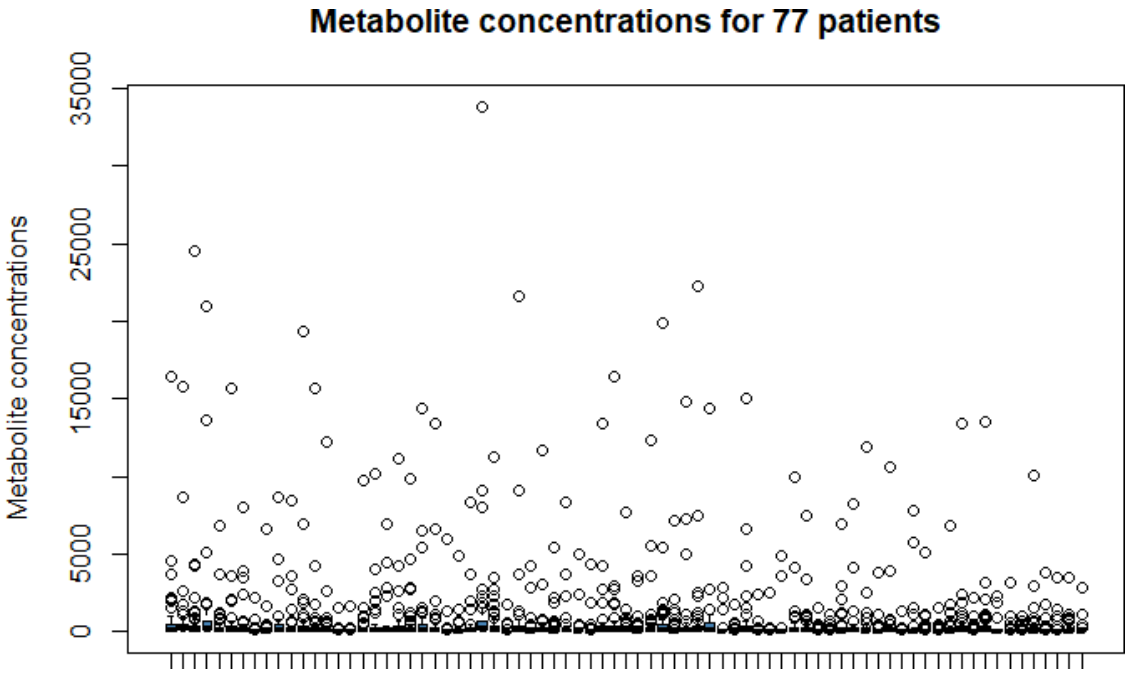


Figure 1. Distribución original de las concentraciones en metabolitos para los 77 pacientes

Hay efectivamente diferencias enormes de valores para un mismo paciente y también de uno a otro (Fig. 1). Habría que transformalos y/o hacer un scaling. Una transformación log parece más adecuada, puesto que los datos son muy asimétricos (Fig. 2). A continuación, se aplica una estandarización (Fig. 2).

```
## Aplicamos la transformación
sumexpcachexia_log <- PomaNorm(sumexpcachexia, method = "log")
## Estandárizamos también los datos
sumexpcachexia_log_norm <- sumexpcachexia_log
assay(sumexpcachexia_log_norm) <- scale(assay(sumexpcachexia_log_norm))
```

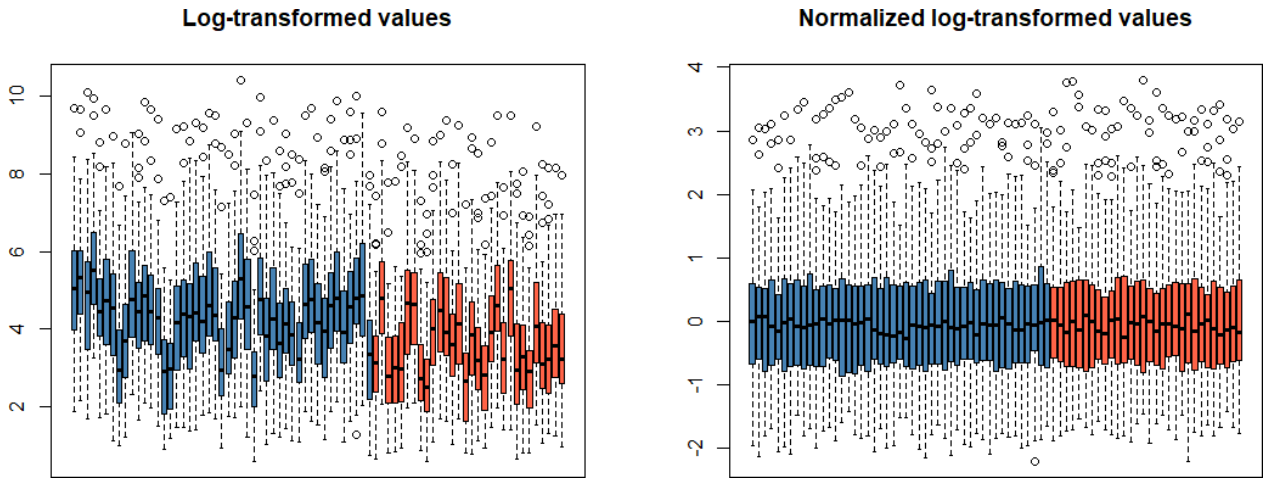


Figure 2. Distribución de las concentraciones en metabolitos para los 77 pacientes después de la transformación log y de la transformación log + estandarización

Hemos acabado con el preprocesamiento y exploración univariante de los datos. última cosa, recordamos los tipos de pacientes y los grupos: podemos mirar la proporción de cada tipo de pacientes por grupo.

```
## Tabla de contingencia
table(identif, colData(sumexpcachexia_log_norm)[, 1])
```

```
##
## identif cachexic control
##   NETCR       13       8
##   NETL        10      13
##   PIF         24       9
```

Los tres tipos están representados en los dos grupos lo que sugiere que el tipo es independiente del grupo.

4.4.2. **Análisis multivariante**

Empezamos la exploración multivariante con un ACP clásico para ver como se distribuyen los pacientes a partir de las concentraciones en metabolitos, y cuanta variabilidad explican esas variables una vez concentradas en componentes (Fig. 3).

```r
## Cargamos el paquete mixOmics
suppressPackageStartupMessages(library(mixOmics, quietly = T))

## Warning: package 'MASS' was built under R version 4.4.3

## PCA con tres componentes. No escalamiento puesto que ya lo hicimos!
acp <- tune.pca(t(assay(sumexpcachexia_log_norm)), scale = F, ncomp = 3)
```

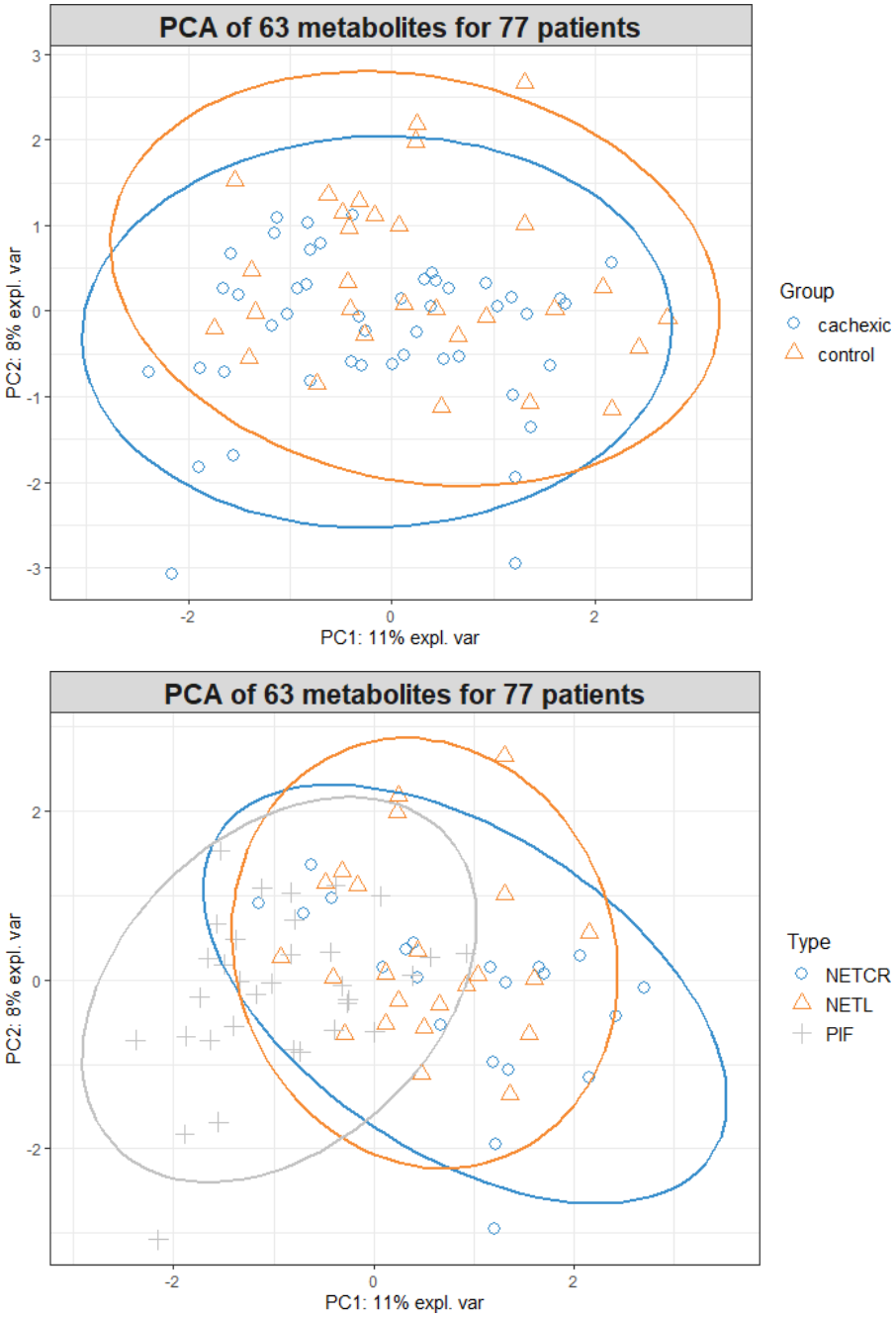


Figure 3. ACP sobre los 77 pacientes considerando los dos grupos y los tres tipos de pacientes

En nuestro caso, la varianza se reparte sobre muchos ejes con lo cual los primeros ejes no explican mucho. Vemos que hay mucho solape cuando consideramos los dos grupos (Fig. 3, arriba). En cambio, la distinción es más marcada al considerar los tipos de pacientes y tenemos los pacientes *PIF* por un lado y los *NETL+NETCR* por otro (Fig.3, abajo). Tenemos un efecto Batch!

Con el fin de contrastar los resultados del ACP, es interesante realizar otro análisis de reducción dimensional, y esta vez, no lineal. Hacemos un mapping no lineal de Sammon (reduce a dos dimensiones), que requiere distancias entre pacientes, en lugar de la matriz de datos original (Fig. 4).

```
## Distancias euclidianas
dist_euc <- dist(t(assay(sumexpcachexia_log_norm)))
## Cargamos el paquete MASS
suppressPackageStartupMessages(library(MASS, quietly = T))
## Mapping de Sammon
samm <- sammon(dist_euc, trace = FALSE)
```

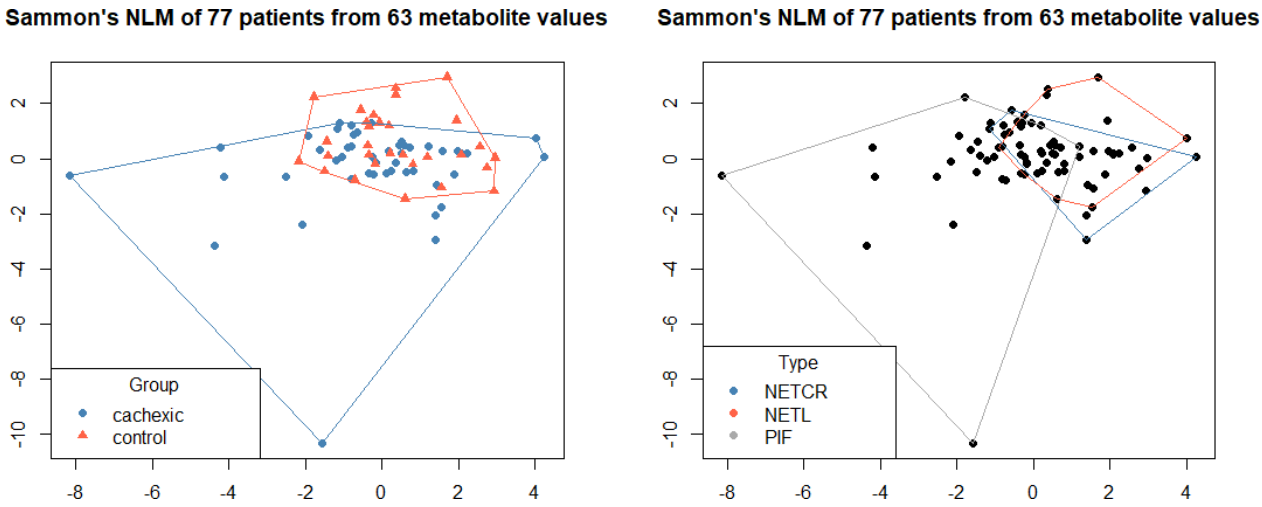


Figure 4. Mapping no lineal de Sammon basado en las distancias euclidianas entre los 77 pacientes

Notamos una vez más el solape marcado entre los dos grupos aunque hay más variabiliad en el grupo *cachexic* (Fig. 4). Considerando los tres tipos de pacientes, vemos que una buena parte de los pacientes *PIF* están separados de una buena parte de los pacientes *NETCR+NETL* (efecto batch).

Un segundo tipo de análisis basado en distancias que podemos realizar es un análisis de conglomerados (clustering). Con un cluster es más fácil ver a la vez la información sobre el grupo y sobre el tipo de los pacientes. Creamos entonces una variable de agrupamiento que reporta a la vez el grupo y el tipo y la usamos como labels a la hora de visualizar el cluster (Fig .5).

```r
## Variable de agrupamiento con doble información
doublegroup <- paste(identif, rep(c("cachexic", "control"), c(47, 30)))
clustering <- hclust(dist_euc, method = "average")
```

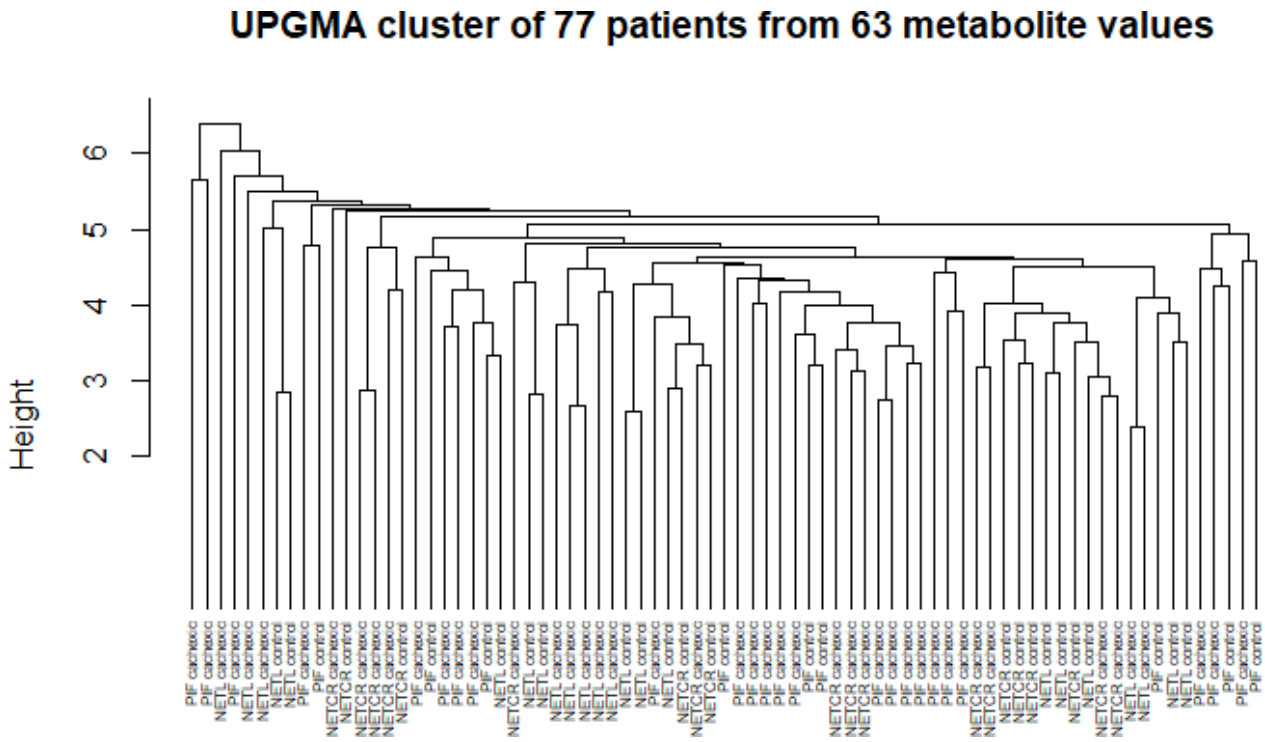


Figure 5. Agrupación jerárquica basada en las distancias euclidianas entre los 77 pacientes

Mirando la topología del árbol vemos que los individuos de un mismo tipo (ej., *PIF*) y de un mismo grupo (ej., *cachexic*) se agrupan entre ellos, y, también, con otros individuos del mismo grupo. Pero se nota también que individuos de un mismo tipo (ej., *NETL*) pueden agruparse entre ellos aunque no son del mismo grupo, en clusteres más grandes. Esto significa que hay una cierta diferenciación entre los dos grupos (*cachexic* vs. *control*) pero que también hay cierto grado de diferenciación ente los tipos de pacientes que puede ocultar parte de las diferencias entre los dos grupos (efecto batch).

Finalmente, podemos hacer un análisis de correspondencias para ir más allá y relacionar los pacientes con los metabolitos. Miramos los resultados usando los tipos de pacientes como variable de agrupamiento puesto que ya estamos al tanto del efecto Batch en nuestro datos (Fig. 6).

```r
## Cargamos el paquete made4
suppressPackageStartupMessages(library(made4, quietly = T))
## Análisis de correspondencias
ca <- ord(assay(sumexpcachexia_log_norm), type = "coa")
```

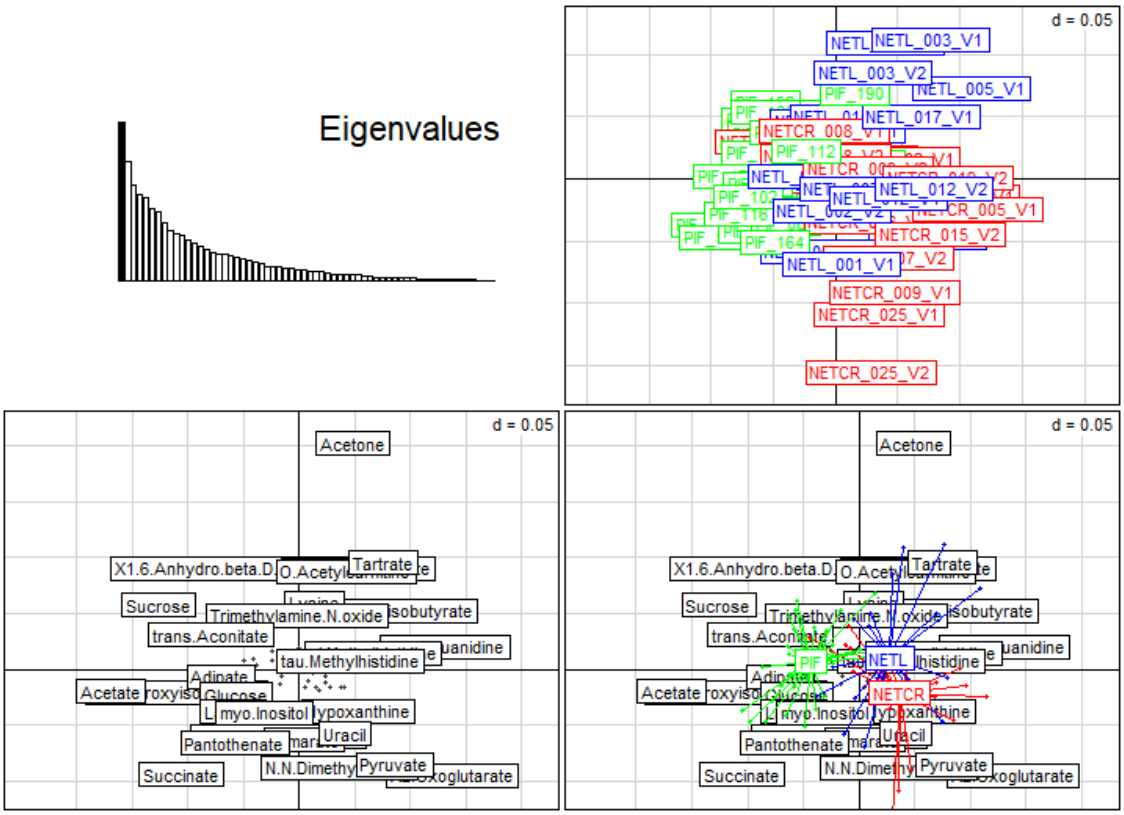


Figure 6. Análisis de correspondencias mostrando las relaciones entre los metabolitos y los pacientes (coloridos por tipos)

Notamos que los primeros componentes explican más varianza que los primeros PCs del ACP que hicimos más arriba pero tampoco explican mucho (Fig. 6 arriba izquierda). El efecto Batch se confirma con este análisis (Fig. 6 arriba derecha). La distribución de los metabolitas no muestra agrupamientos muy claros pero podemos decir que, más o menos, los metabolitos más a la izquierda (Sucrose, Acetate, Succinate etc.) corresponden a los pacientes *PIF* y los más a la derecha (Acetone, Tartrate, Pyruvate etc.) corresponden a los pacientes *NETL+NETCR* (Fig. 6 abajo).

5. **Discusión**

El conjunto de datos Cachexia reporta concentraciones en metabolitos para dos grupos de pacientes. La distribución de los datos original muestra que distintos metabolitos se encuentran en concentraciones distintas en el cuerpo y que las diferencias de concentraciones de un individuo a otro pude variar considerablemente. Esto demuestra la importancia de siempre verificar los datos, especificamente para comprobar si hay valores nulos (ceros), valores faltantes (NAs), valores atípicos, y, distribuciones distintas de una muestra a otra. Preprocesar los datos asegura más control sobre los datos a la hora de analizarlos y entonces evitar de encontrarse en situaciones en que los resultados no son representativos y son, en realidad, debidos a artefactos. En este estudio

realizamos una serie de análisis multivariantes para explorar patrones de distribución y agrupación de los pacientes y metabolitos. Un ACP muestra que la variabilidad en las concentraciones de metabolitos entre los pacientes se reparte sobre muchos componentes y no se concentra en los primeros ejes. Mirando los dos primeros componentes (el tercero no es muy útil, ver Anexos), hay un solape evidente entre los pacientes que presentan la afección y los que son sanos. En cambio, si diferenciamos los pacientes según el tipo (tipo de pacientes = tipo de muestras), se nota menos solape y una cierta distincción entre los pacientes *PIF* y los pacientes *NETL+NETCR*, evidenciando un efecto Batch n mapping no lineal de Sammon basado en las distancias euclidianas entre los pacientes confirma esos resultados (ver también análisis basadas en distancias de Pearson en Anexos). Un análisis de conglomerados, basado también en las distancias euclidianas entre los pacientes (ver Anexos para distancias de Pearson), resulta en clusters en que se agrupan a menudo pacientes del mismo grupo aunque no sean del mismo tipo. Sin embargo, en muchas ocasiones se agrupan también pacientes del mismo tipo que no sean del mismo grupo. La topología del árbol apoya a la afirmación de que hay un efecto Batch relacionado con los tipos de pacientes que oculta seguramente parte de las diferencias reales entre los pacientes que tienen Caquexia y los que no. Finalmente, un análisis de correspondencias muestra otra vez este efecto Batch, y podría sugerir una relación entre la concentración de ciertos metabolitos y la diferenciación entre uno (*PIF*) o otros (*NETL+NETCR*) de los tipos de pacientes, aunque no esté muy claro, . Este trabajo era solamente un estudio exploratorio del conjunto de datos seleccionado. A partir de ahí sería interesante profundizar en las correspondencias metabolitos-pacientes y obtener más detalles sobre como se definieron exactamente los tipos de muestras (no se ha encontrado información al respecto). Sobre todo, haría falta intentar corregir por este efecto Batch en el caso en que uno quisiera estudiar especificamente las diferencias entre los dos grupos de pacientes.

6. **Conclusiones**

En este trabajo nos focalizamos en un conjunto de datos sobre el síndrome metabólico Caquexia, que corresponde a valores de concentración de 63 metabolitos para 77 pacientes. Los pacientes se reparten en dos grupos, los que presentan la patología y los sanos. Se nota también la presencia de tres tipos de muestras (tipos de pacientes). Un preprocesamiento de los datos permite corregir por diferencias marcadas en las distribuciones de esas concentraciones en metabolitos entre pacientes. Una serie de análisis multivariantes muestra que hay en general pocas diferencias entre los dos grupos de pacientes, y, agrupamientos no muy claros de los metabolitos. En cambio, evidenciamos un efecto Batch debido a los tipos de muestras, destacando dos clusters que son independientes del hecho de tener la patología o de ser sano.

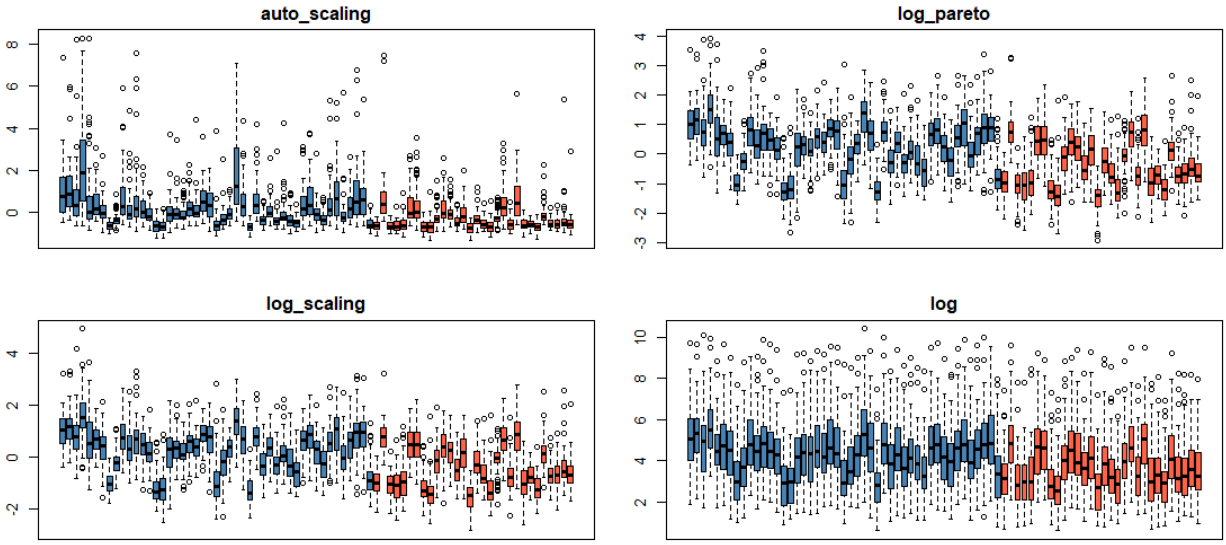
7. **Referencias**

https://github.com/floriangbcht/Bouchet-Florian-PEC1

Anexos

1. Boxplots comparativos según la modificación aplicada a los datos

Estandarización (normalization) clásica vs. escalamiento log Pareto (por defecto en PomaNorm) vs. escalamiento log clásico vs. transformación log.



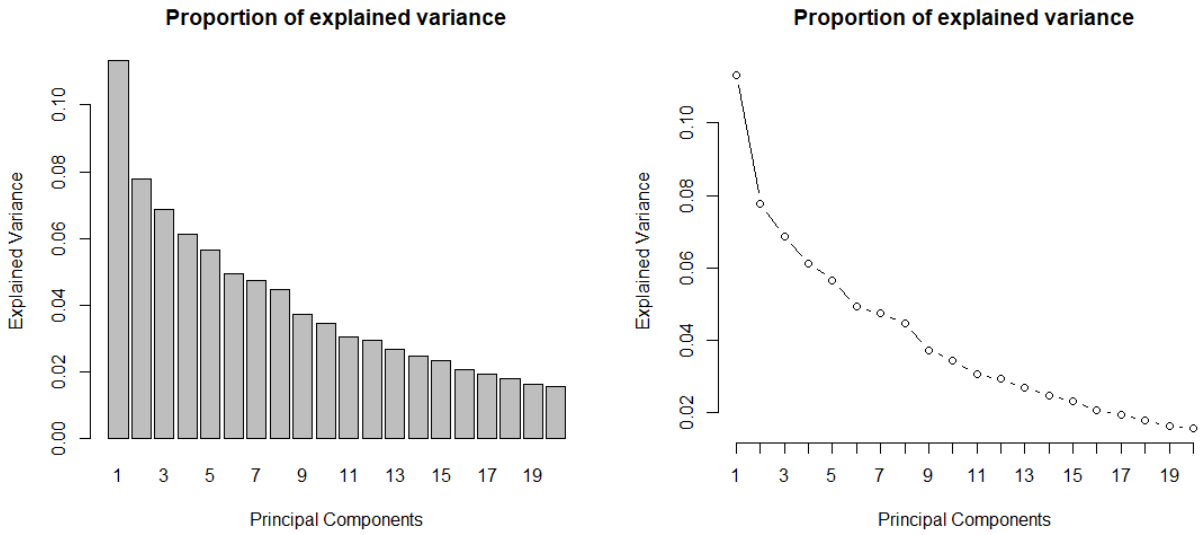
Efectivamente, parece más adecuado empezar con una transformación log.

2. Análisis de componentes principales

Miramos lo que da el ACP seleccionando primero un máximo de 20 PCs. Así podemos ver como cambia la varianza explicada a lo largo de los componentes.

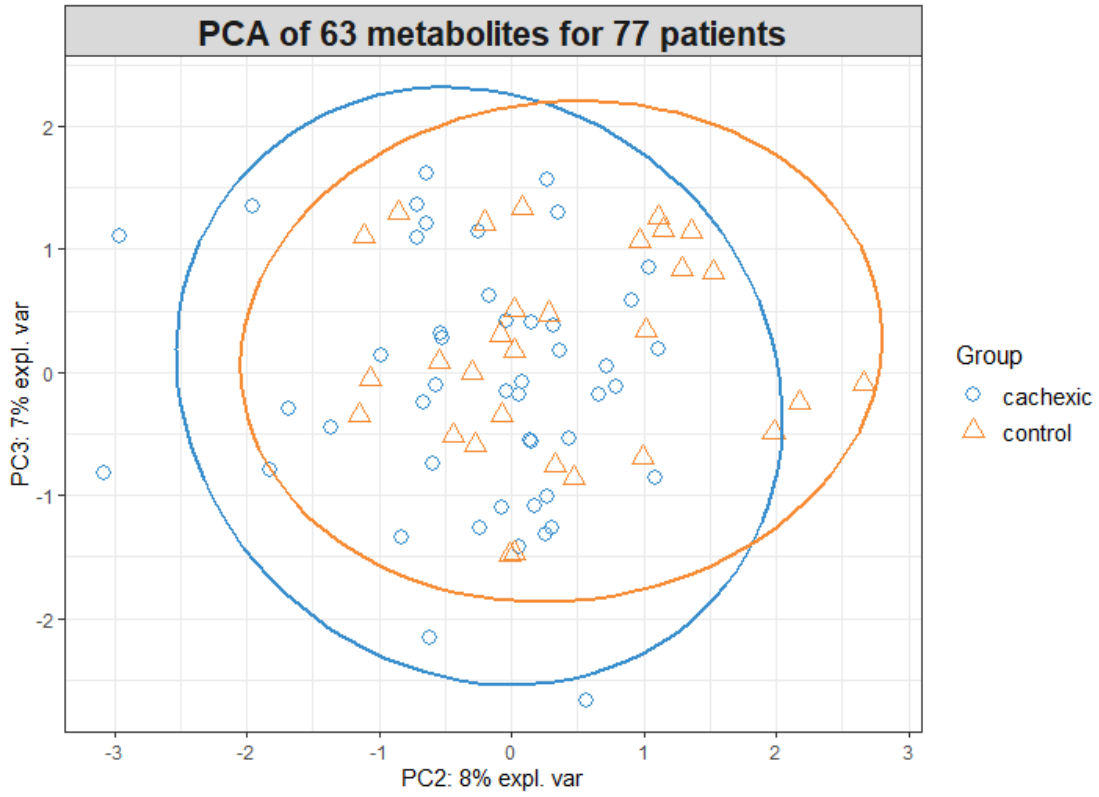
```
## ACP (seleccionamos de antemano un máximo de 20 PCs):
acp_explo <- tune.pca(t(assay(sumexpcachexia_log_norm)), scale = F, ncomp =
20)
## Proporción de varianza explicada por cada PC
round(acp_explo$prop_expl_var$X * 100, 2)

##   PC1   PC2   PC3   PC4   PC5   PC6   PC7   PC8   PC9  PC10  PC11  PC12
PC13
## 11.33  7.77  6.86  6.13  5.66  4.93  4.74  4.47  3.74  3.45  3.06  2.93
2.69
##  PC14  PC15  PC16  PC17  PC18  PC19  PC20
##  2.48  2.33  2.06  1.94  1.79  1.63  1.57
```



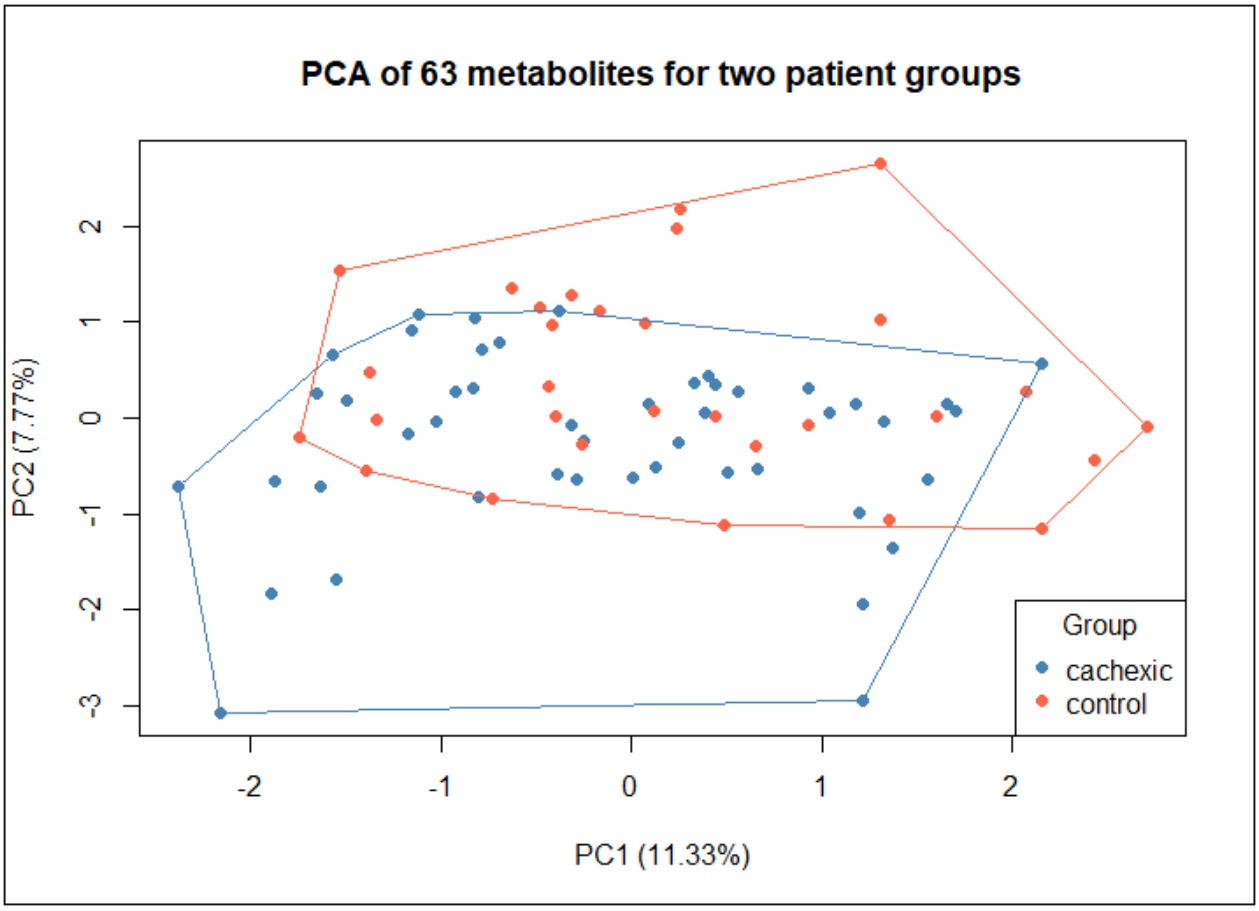
El drop en la varianza es continuo, no es muy marcado. La varianza se reparte en muchos PCs y por tanto los primeros PCs no explican mucha varianza. En todo caso solo considerar los primeros PCs es suficiente.

Volvemos al acp que usamos en este trabajo, y miramos el gráfico PC2 vs. PC3.



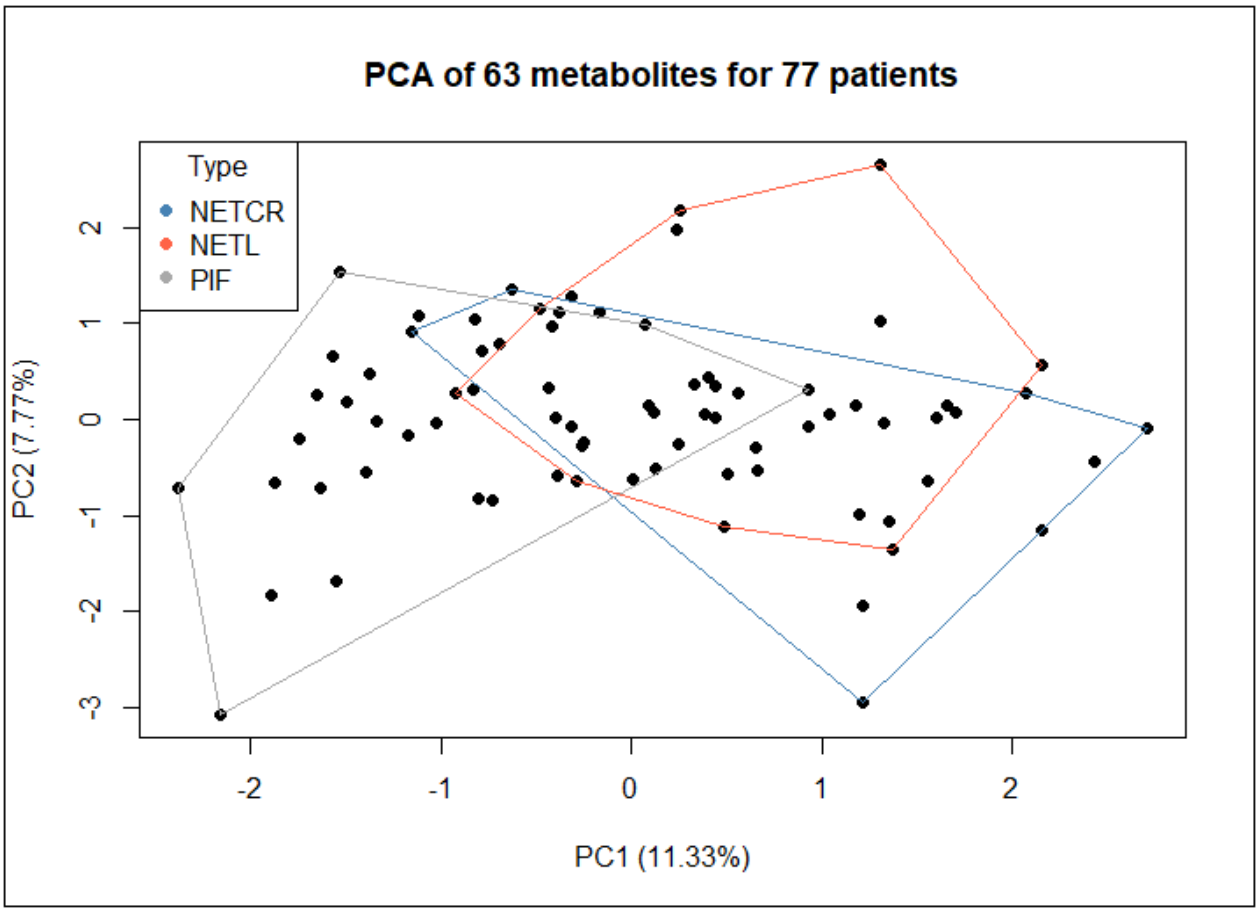
Hay mucho solape. Y vemos que hay menos solape en el plot PC1 vs. PC2 que en el plot PC2 vs. PC3. El PC3 no sirve mucho para separar los dos grupos.

Las elipses pueden ser un poco confundientes visualmente, entonces vamos a mirar el plot PC1 vs. PC2 de otra manera, usando convex hulls.



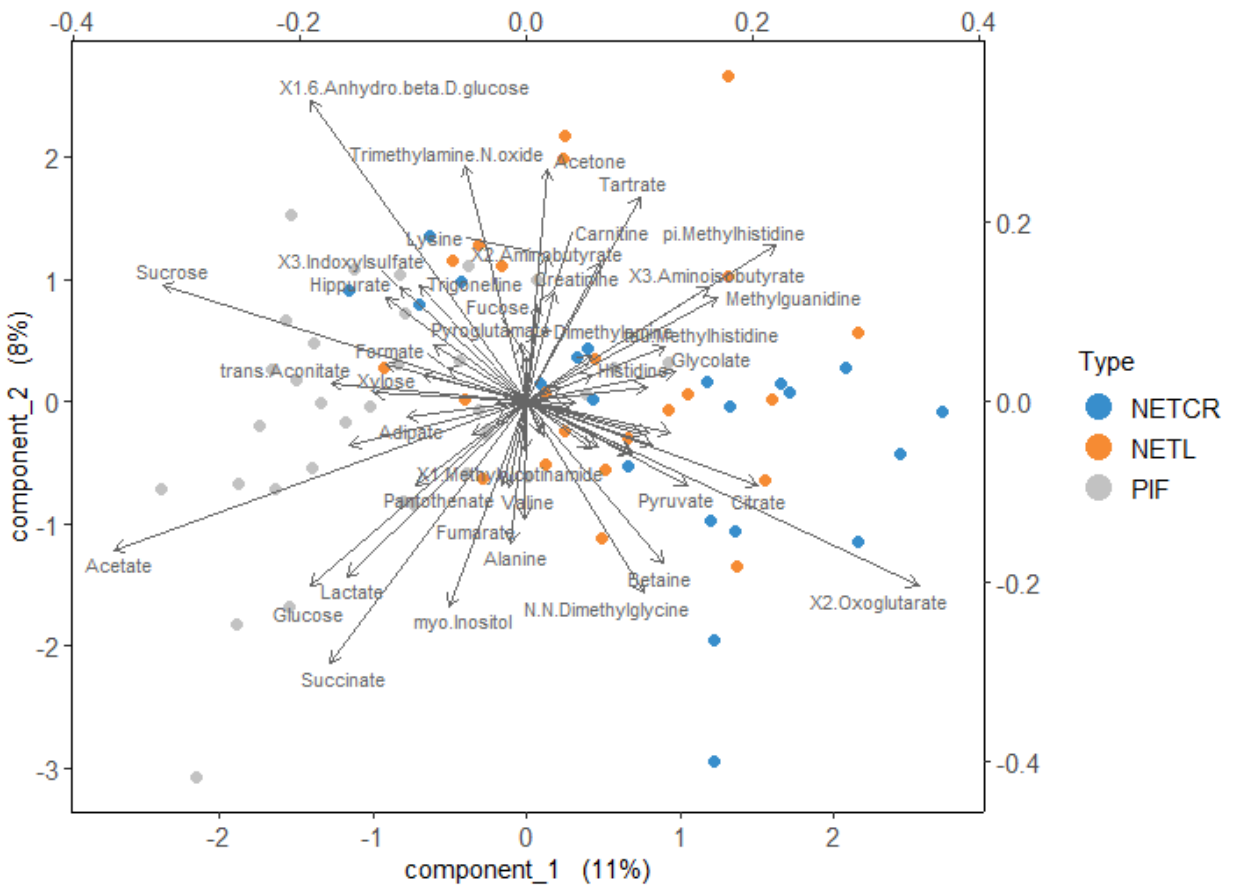
Más comprensible así. Pero confirmamos que hay mucho solape si consideramos los dos grupos.

Ahora miramos también este ACP diferenciando los pacientes por tipos.



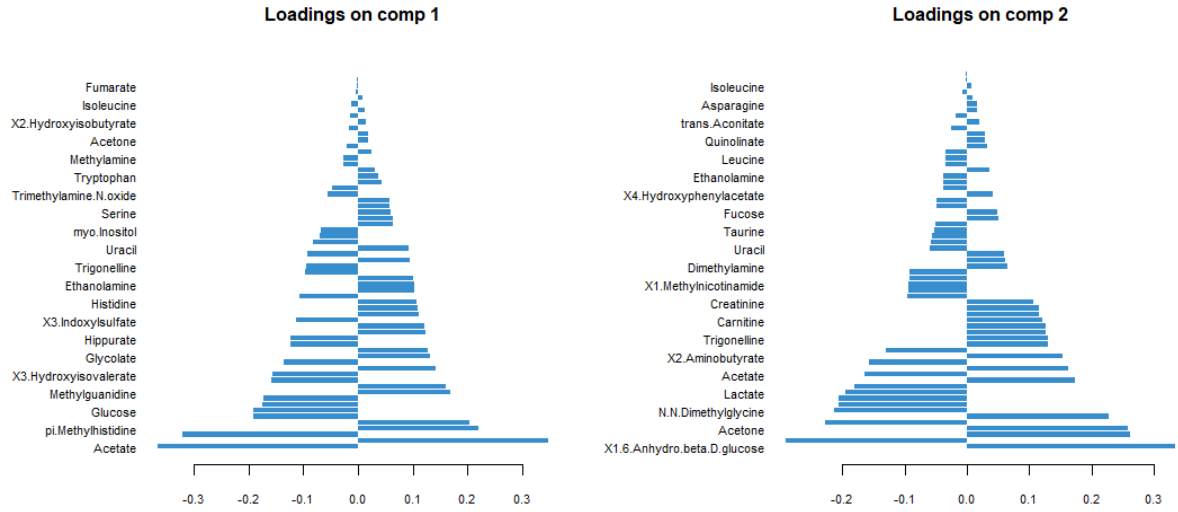
Más comprensible también. Hay un efecto Batch.

Ahora miramos como las variables se correlan entre ellas, y con los PCs, diferenciando los pacientes por tipo puesto que ya sabemos que tenemos un efecto batch.



Hay muchos metabolitos, y es dificil comentar las correlaciones entre ellos. Ningúna variable tiene correlación estricta con uno o otro PC: Es decir, ningún vector propio es bien parallelo a uno o otro PC. Aunque no hayan agrupamientos claros de los metabolitos podemos decir que los más a la izquierda pueden contribuir a tener scores del PC1 más negativos y los más a la dercha a tener scores del PC2 más positivos. Básicamente, esos metabolitos más a la izquierda y más a la derecha pueden contribuir a la separación entre los pacientes *PIF* y los pacientes *NETL+NETCR*. Pero otra vez, tampoco está muy claro. De hecho notamos que el patrón de las variables (es decir como se reparten los metabolitos en el plot) es similar al que vimos previamente en el plot del análisis de correspondencias.

Miramos directamente los valores de los vectores propios, mediante un gráfico.



No hay variables (metabolitos) que se destacan marcademente más que las demás. Pero quizá valdría la pena seleccionar un subconjunto de variables y ver si los vectores propios cambian (y por tanto, últimamente, la variabilidad explicada por cada PC). Sería coma una filtración de las variables, que al final puede formar parte del procesamiento de los datos...

Con *mixOmics* podemos testar si hay un número óptimo de variables a incluir que podría maximizar la variabilidad explicada por los PCs (aquí vamos con 3 PCs). Con la función *tune.spca* podemos testar incluyendo todas las variables (para 3 PCs), y repetir 25 veces la validación cruzada (máximo 50 pero ya con 30 deberíamos obtener un resultado coherente). Básicamente, esta función *tune.spca()* permite obtener el número de variables a incluir que maximiza la correlación entre ellas y los PCs, para cada PC, empezado con el número más bajo posible de variables y añadiendo cada vez más.

El código seriá el siguiente (no lo ejecutamos aquí porque puede llevar un poco de tiempo):

#acp_mod <- tune.spca(t(assay(sumexpcachexia_log_norm)), scale = F, ncomp = 3, folds = 5, test.keepX = 1:nrow(assay(sumexpcachexia_log_norm)), nrepeat = 30)
#acp_mod$choice.keepX

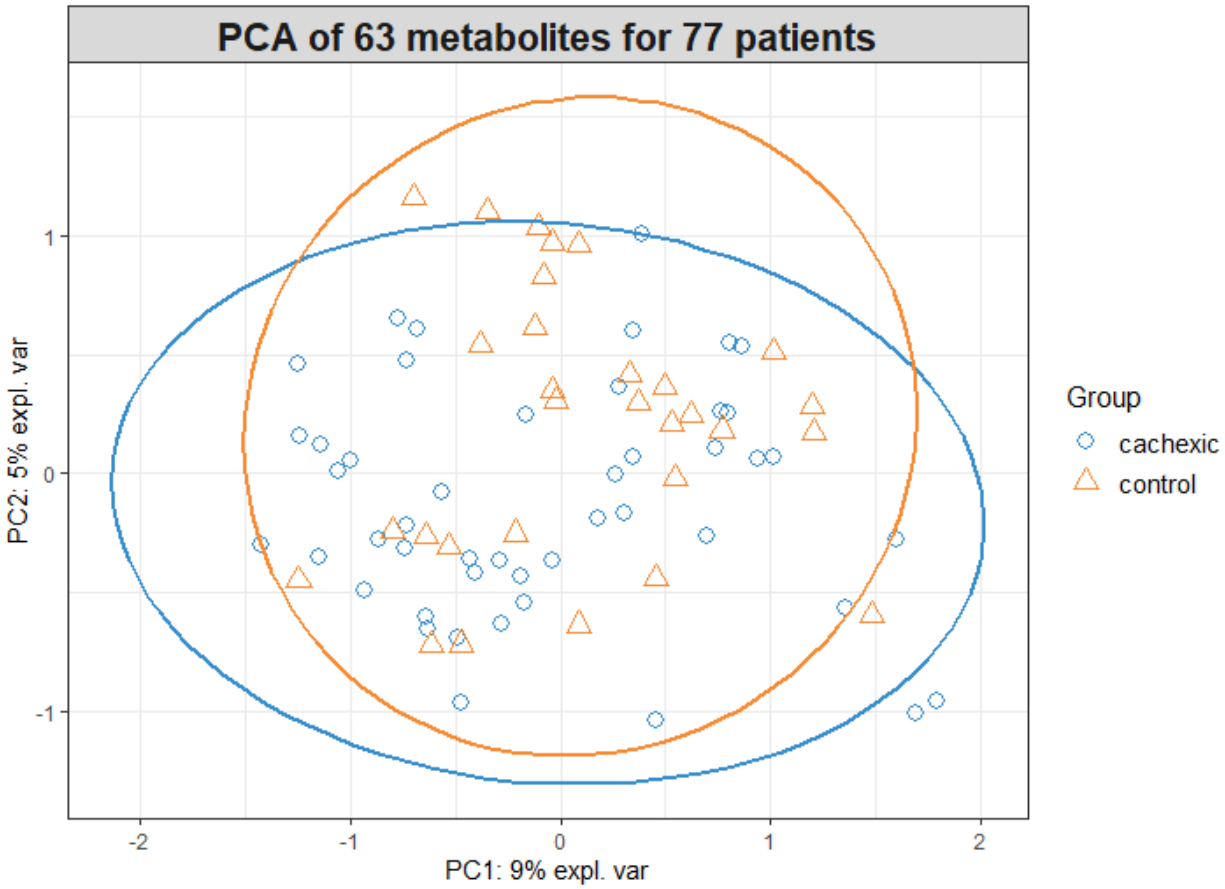
Si ejecutamos esta función (podemos repetir para asegurarse del resultado), el resultado nos da un número muy alto de variables, muy cerca del número original de variables que tenemos. Es que quitar variables (metabolitos) no “mejorará” el análisis. A esta altura podríamos directamente seleccionar nosotros los metabolitos a incluir, por ejemplo los que muestran un loading (= vector propio = contribución de la variable al componente) superior a 0.3.

```
## Selección de metabolitos
select1 <- length((abs(selectVar(acp, comp = 1)$value) >
    0.3)[(abs(selectVar(acp, comp = 1)$value) > 0.3) == T])
```

```
select2 <- length((abs(selectVar(acp, comp = 2)$value) >
    0.3)[(abs(selectVar(acp, comp = 2)$value) > 0.3) == T])

## ACP con los metabolitos seleccionados
final.spca.multi <- spca(t(assay(sumexpcachexia_log_norm)), scale = F,
    keepX = c(select1, select2))
final.spca.multi$prop_expl_var$X * 100

##       PC1       PC2
## 9.477482 5.079856
```



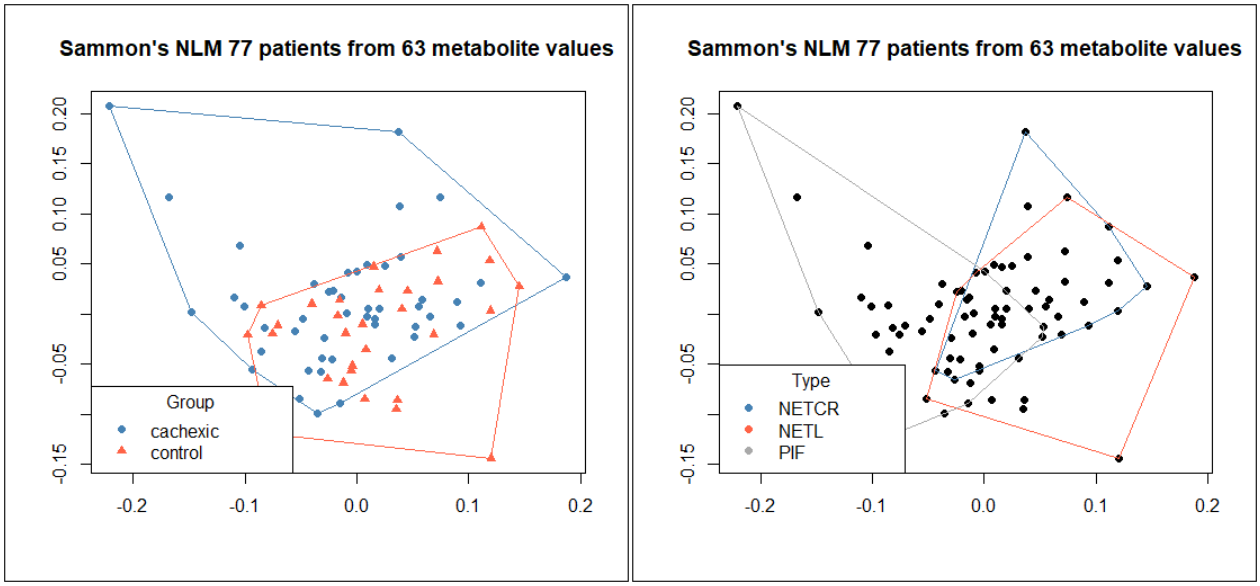
Tampoco mejorá el ACP (lo que se podía esperar). La varianza explicada es aún más baja para cada PC (tiene sentido)... Ya no hacemos nada con el ACP y pasamos a otros análisis.

3. Análisis basados en distancias entre pacientes

Además de distancias euclidianas, creamos también un set de distancias basadas en correlaciones de Pearson. Distancias basadas en la correlación se usan bastante en genomica (datos de expresión), y quizá puedan aportar algo aquí...

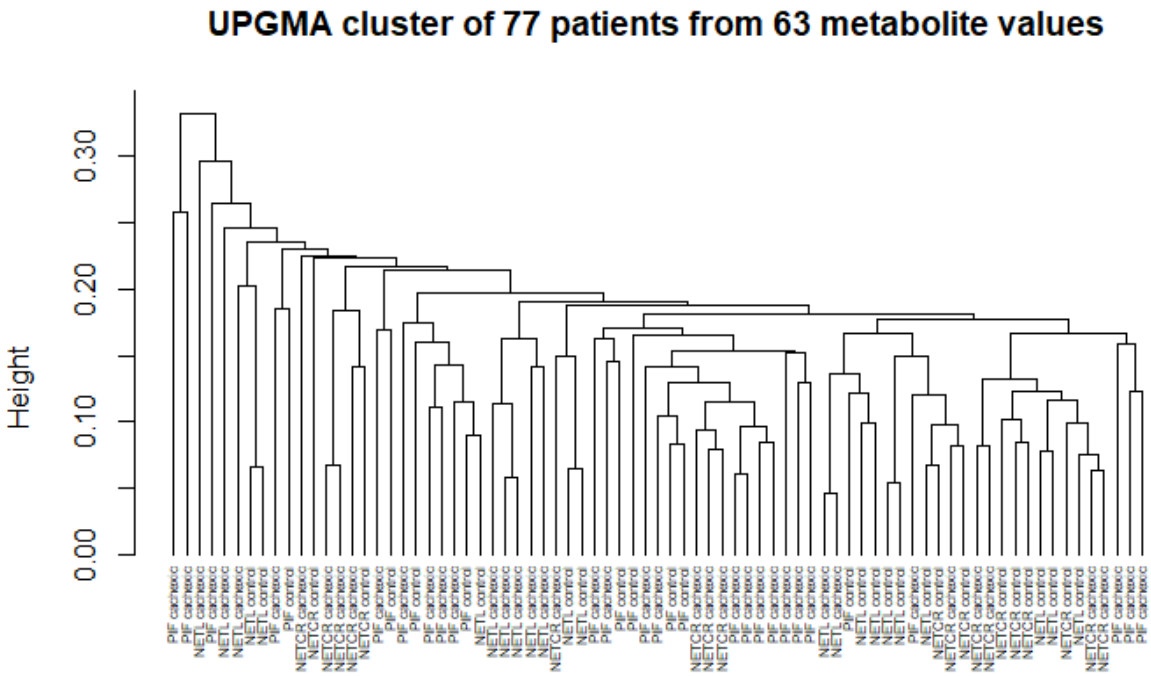
```r
## Distancias de Pearson
dist_cor <- as.dist(1 - cor(assay(sumexpcachexia_log_norm)))
```

En el mapping no lineal de Sammon basado en distancias euclidianas tenemos más variabilidad en los pacientes *PIF*. Añadimos que el paciente con coordenadas aprox (-2, -10) podría representar un outlier. Ahora podemos realizar de nuevo el análisis pero con distancias de Pearson.



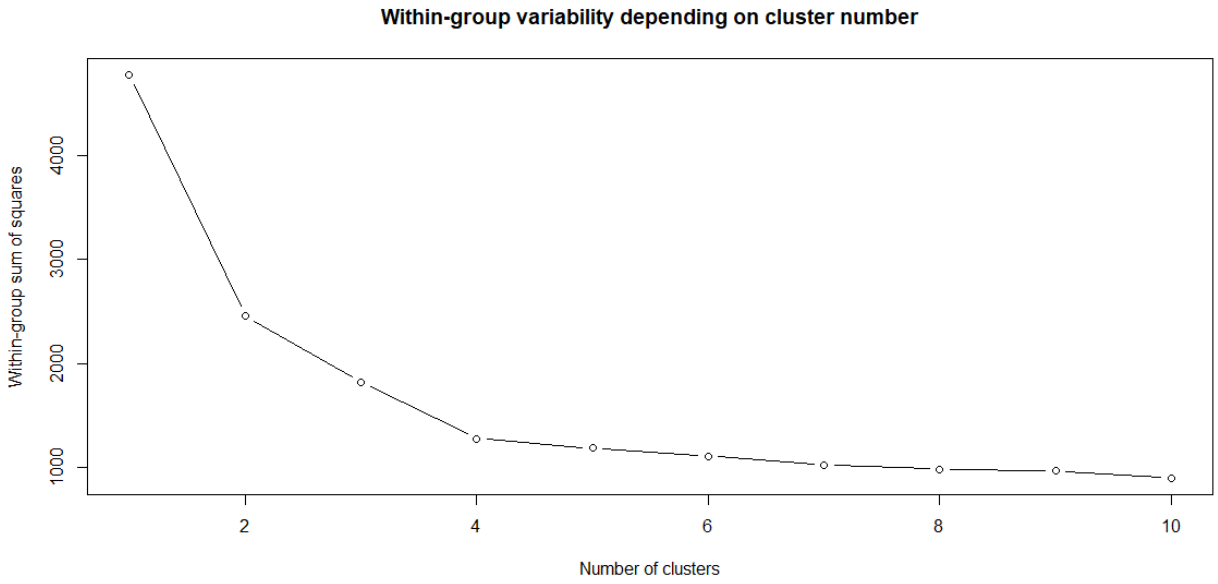
El solape aún es marcado pero la variabilidad es más parecida ente los dos grupos. Aún tenemos una buena parte de los pacientes *PIF* que son separados de una buena parte de los *NETCR*+*NETL* (efecto batch). El paciente con coordenadas aprox (-0.2, 0.20) podría representar el mismo outlier que en el análisis basado en distancias euclidianas.

Podemos también repetir el análisis de conglomerados con distancias de Pearson.



Las topologías de los dos clusteres son bastante similares. Esto confirma que hay una cierta diferenciación entre los dos grupos (*cachexic* vs. *control*) pero que también hay cierto grado de diferenciación ente los tipos de pacientes y que por tanto oculta una parte de las diferencias entre los dos grupos (efecto batch).

Para acabar, podemos intentar ver si existen realmente grupos (clusteres) de metabolitos en nuestros datos usando K-means. Primero miramos cual es el número óptimo de clusteres usando el método del codo (para ver con cual número de clusteres la variabilidad intragrupos es miníma).



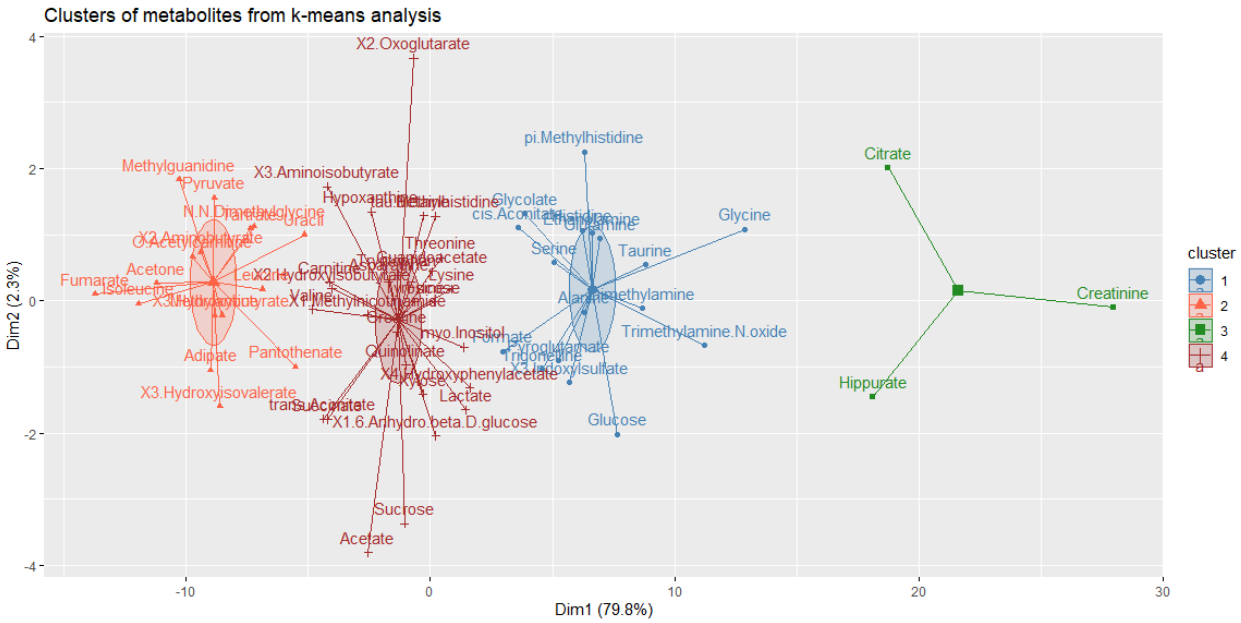
Parece que es 4. Si repetimos el proceso varias veces siempre saldrá que con K = 4 tenemos el valor de variabilidad intragrupos miníma (o sea, a partir de este K este valor ya no baja significativamente).

Ahora aplicamos la función para K-means y hacemos un gráfico con los clusters usando el paquete *factoextra*.

```
## K-means
k_means <- kmeans(assay(sumexpcachexia_log_norm), 4, iter.max = 1000)

## Gráfico
suppressPackageStartupMessages(library(factoextra, quietly = T))
fviz_cluster(k_means, data = assay(sumexpcachexia_log_norm),
    palette = c("steelblue", "tomato", "forestgreen",
        "brown"), ellipse.type = "euclid", star.plot = TRUE,
    main = "Clusters of metabolites from k-means analysis")

## Too few points to calculate an ellipse
```



Efectivamente tenemos 4 clusteres con poco solape aunque hay cercanía, el último siendo muy pequeño y más separado de los otros. Podría ser interesante a partir de este resultado intentar ver como el grupo y el tipo de pacientes explica esta variabilidad (hacerlo al reves que en el análisis de correspondencias previo).