PEC 1 - Les òmiques

Marc Canela Grimau

2024-10-28

Download and Explore the Data Set

In this report, we are going to use the data used in the paper *"Metabotypes of response to bariatric surgery independent of the magnitude of weight loss"*. The data set is formed by the following files:

- `DataInfo_S013.csv`: Metadata. Information on each column in the "DataValues_S013.csv" file.
- `DataValues_S013.csv`: Clinical and metabolomic values for 39 patients at 5 time points.
- `AAInformation_S006.csv`: Additional information on metabolites in the "DataValues_S013.csv" file.

We will download the files from GitHub and open them in R:

```
setwd(dir = "~/Desktop/Canela-Grimau-Marc-PEC1")
DataInfo_S013 <- read.csv("DataInfo_S013.csv", header=TRUE, row.names=1)
DataValues_S013 <- read.csv("DataValues_S013.csv", header=TRUE, row.names=1)
AAInformation_S006 <- read.csv("AAInformation_S006.csv", header=TRUE, row.names=1)
```

Now we'll briefly explore the files we've downloaded:

```
str(DataInfo_S013)
```

```
## 'data.frame':    695 obs. of  3 variables:
##  $ VarName    : chr  "SUBJECTS" "SURGERY" "AGE" "GENDER" ...
##  $ varTpe     : chr  "integer" "character" "integer" "character" ...
##  $ Description: chr  "dataDesc" "dataDesc" "dataDesc" "dataDesc" ...
```

```
head(DataInfo_S013)
```

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

The `DataInfo_S013` contains information on each column from `DataValues_S013`: the names of each column in `VarName`, the type in `varType`, and the description in `Description`. The `Description` feature is not informative, as it's empty.

```
str(AAInformation_S006)
```

```
## 'data.frame':    188 obs. of  6 variables:
##  $ Class                  : chr  "aminoacids" "aminoacids" "aminoacids" "aminoacids" ...
##  $ Metabolite.abbreviation: chr  "Ala" "Arg" "Asn" "Asp" ...
##  $ Metabolite             : chr  "Alanine" "Arginine" "Asparagine" "Aspartate" ...
##  $ Platform               : chr  "LC-MS/MS" "LC-MS/MS" "LC-MS/MS" "LC-MS/MS" ...
##  $ Data.type              : chr  "Quantified" "Quantified" "Quantified" "Quantified" ...
```

```
##  $ X                      : logi  NA NA NA NA NA NA ...
```

```
head(AAInformation_S006)
```

```
##        Class Metabolite.abbreviation Metabolite Platform  Data.type  X
## 1 aminoacids                     Ala    Alanine LC-MS/MS Quantified NA
## 2 aminoacids                     Arg   Arginine LC-MS/MS Quantified NA
## 3 aminoacids                     Asn Asparagine LC-MS/MS Quantified NA
## 4 aminoacids                     Asp  Aspartate LC-MS/MS Quantified NA
## 5 aminoacids                     Cit Citrulline LC-MS/MS Quantified NA
## 6 aminoacids                     Gln  Glutamine LC-MS/MS Quantified NA
```

The `AAInformation_S006` contains information on the metabolites from `DataValues_S013`: the class of each meatabolite in `Class`, the abbreviation and full name in `Metabolite.abbreviation` and `Metabolite`, respectively, the platform in `Platform`, and the type in `Data.type`.

And now that we know the structure of the data set, we can create the `SummarizedExperiment` class.

Create the SummarizedExperiment Class

We will first prepare the data. In a `SummarizedExperiment` class, rows represent features of interest and columns represent samples, so we may have to transpose the `DataValues_S013`. The `rowData` should have the same amount of rows are the `DataValues_S013`.

```
library(SummarizedExperiment)
```

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

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

```
##
## 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, table,
##     tapply, union, unique, unsplit, which.max, which.min

## Loading required package: S4Vectors

##
## 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

## Loading required package: GenomeInfoDb

## 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
```

```
data_matrix <- t(as.matrix(DataValues_S013))
rowData <- DataFrame(DataInfo_S013[, c("VarName", "varTpe")])
```

And now we'll create the class. We would only include information on each row (i.e., on each variable), and not on each column because the dataset doesn't provide this information.

```
metabo <- SummarizedExperiment(
  assays = list(counts = data_matrix),
  rowData = rowData
)
metabo
```

```
## class: SummarizedExperiment
## dim: 695 39
## metadata(0):
## assays(1): counts
## rownames(695): SUBJECTS SURGERY ... SM.C24.0_T5 SM.C24.1_T5
## rowData names(2): VarName varTpe
## colnames(39): 1 2 ... 38 39
## colData names(0):
```

We can save the data and metadata as `.Rda` files:

```
save(DataInfo_S013, DataValues_S013, AAInformation_S006, metabo, file = "data_and_metadata.rda")
```

Exploration of the data

All the following code can be found in the file `exploration.R`.

We will first filter the numeric variables from `DataValues_S013` using the information in `DataInfo_S013`:

```
library(dplyr)
```

```
##
## 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
```

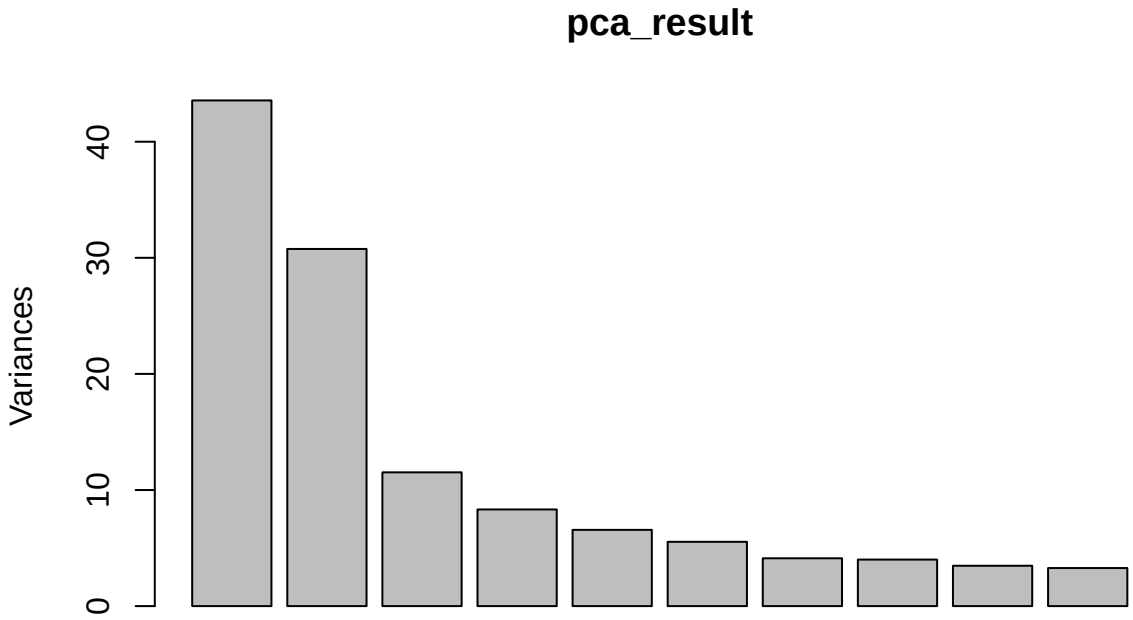
```r
numeric_vars <- DataInfo_S013 %>%
  filter(varTpe %in% c("numeric", "integer")) %>%
  pull(VarName)
DataValues_S013_numeric <- DataValues_S013 %>%
  select(all_of(numeric_vars))
```

Now we will perform a PCA to see differences:

```r
time_vars <- numeric_vars[grepl("_T[0-9]+$", numeric_vars)]
data_pca <- DataValues_S013_numeric %>%
  select(all_of(c(time_vars, "Group")))
data_pca_clean <- data_pca %>%
  select(where(~ !any(is.na(.)) && sd(.) != 0))
pca_result <- prcomp(data_pca_clean[, -ncol(data_pca_clean)], scale=TRUE)
summary(pca_result)
```

```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     6.5996 5.5464 3.39414 2.88531 2.56252 2.35339 2.02936
## Proportion of Variance 0.2884 0.2037 0.07629 0.05513 0.04349 0.03668 0.02727
## Cumulative Proportion  0.2884 0.4922 0.56846 0.62359 0.66708 0.70375 0.73103
##                           PC8     PC9    PC10    PC11    PC12    PC13   PC14
## Standard deviation     2.00207 1.86421 1.81055 1.73868 1.63255 1.50067 1.4898
## Proportion of Variance 0.02654 0.02301 0.02171 0.02002 0.01765 0.01491 0.0147
## Cumulative Proportion  0.75757 0.78059 0.80230 0.82232 0.83997 0.85488 0.8696
##                           PC15    PC16    PC17    PC18    PC19    PC20    PC21
## Standard deviation     1.41146 1.37684 1.26899 1.21254 1.14539 1.12439 1.06148
## Proportion of Variance 0.01319 0.01255 0.01066 0.00974 0.00869 0.00837 0.00746
## Cumulative Proportion  0.88277 0.89533 0.90599 0.91573 0.92442 0.93279 0.94025
##                           PC22    PC23    PC24    PC25    PC26    PC27    PC28
## Standard deviation     1.03483 1.01411 0.91547 0.87365 0.83736 0.82146 0.74879
## Proportion of Variance 0.00709 0.00681 0.00555 0.00505 0.00464 0.00447 0.00371
## Cumulative Proportion  0.94734 0.95415 0.95970 0.96476 0.96940 0.97387 0.97758
##                           PC29    PC30    PC31   PC32    PC33    PC34    PC35
## Standard deviation     0.71480 0.69738 0.65478 0.6150 0.56550 0.54831 0.52949
## Proportion of Variance 0.00338 0.00322 0.00284 0.0025 0.00212 0.00199 0.00186
## Cumulative Proportion  0.98097 0.98419 0.98703 0.9895 0.99165 0.99364 0.99550
##                           PC36    PC37    PC38     PC39
## Standard deviation     0.4912 0.48573 0.44998 5.17e-15
## Proportion of Variance 0.0016 0.00156 0.00134 0.00e+00
## Cumulative Proportion  0.9971 0.99866 1.00000 1.00e+00
```

```r
plot(pca_result)
```



We will use the first two variables, as they cover the over the 70% of the variance. Let's see which variables explain a higher percentage of the variance of each principal component:

```
loading_scores <- pca_result$rotation

top_pc1 <- loading_scores %>%
  as.data.frame() %>%
  mutate(variable = rownames(loading_scores)) %>%
  arrange(desc(abs(PC1))) %>%
  slice(1:5)
top_pc1['PC1']
```

```
##                       PC1
## PC.aa.C38.3_T0 0.1301553
## PC.aa.C36.1_T0 0.1279939
## PC.aa.C40.5_T0 0.1266381
## PC.ae.C32.2_T0 0.1254043
## PC.aa.C32.3_T0 0.1252324
```

```
top_pc2 <- loading_scores %>%
  as.data.frame() %>%
  mutate(variable = rownames(loading_scores)) %>%
  arrange(desc(abs(PC2))) %>%
  slice(1:5)
top_pc2['PC2']
```

```
##                       PC2
## PC.aa.C40.2_T0 -0.1753303
## PC.ae.C44.3_T0 -0.1732997
## PC.ae.C38.2_T0 -0.1722287
## PC.aa.C42.4_T0 -0.1715662
## PC.ae.C42.1_T0 -0.1707464
```

The top 5 variables correspond in both cases to the *T0* time point. We can now check the information of these variables on the `AAInformation_S006`:

```
pc1 <- sub("_T0$", "", row.names(top_pc1['PC1']))
pc1 <- gsub("\\.", " ", pc1)
pc1 <- sub(" (?!.* )", ":", pc1, perl = TRUE)
pc1_info <- AAInformation_S006 %>%
  filter(Metabolite.abbreviation %in% pc1)
pc1_info
```

```
##                 Class Metabolite.abbreviation  Metabolite  Platform
## 1 glycerophospholipids             PC aa C32:3 PC aa C32:3 FIA-MS/MS
## 2 glycerophospholipids             PC aa C36:1 PC aa C36:1 FIA-MS/MS
## 3 glycerophospholipids             PC aa C38:3 PC aa C38:3 FIA-MS/MS
## 4 glycerophospholipids             PC aa C40:5 PC aa C40:5 FIA-MS/MS
## 5 glycerophospholipids             PC ae C32:2 PC ae C32:2 FIA-MS/MS
##         Data.type  X
## 1 Semi-quantified NA
## 2 Semi-quantified NA
## 3 Semi-quantified NA
## 4 Semi-quantified NA
## 5 Semi-quantified NA
```

```
pc2 <- sub("_T0$", "", row.names(top_pc2['PC2']))
pc2 <- gsub("\\.", " ", pc2)
pc2 <- sub(" (?!.* )", ":", pc2, perl = TRUE)
pc2_info <- AAInformation_S006 %>%
  filter(Metabolite.abbreviation %in% pc2)
pc2_info
```

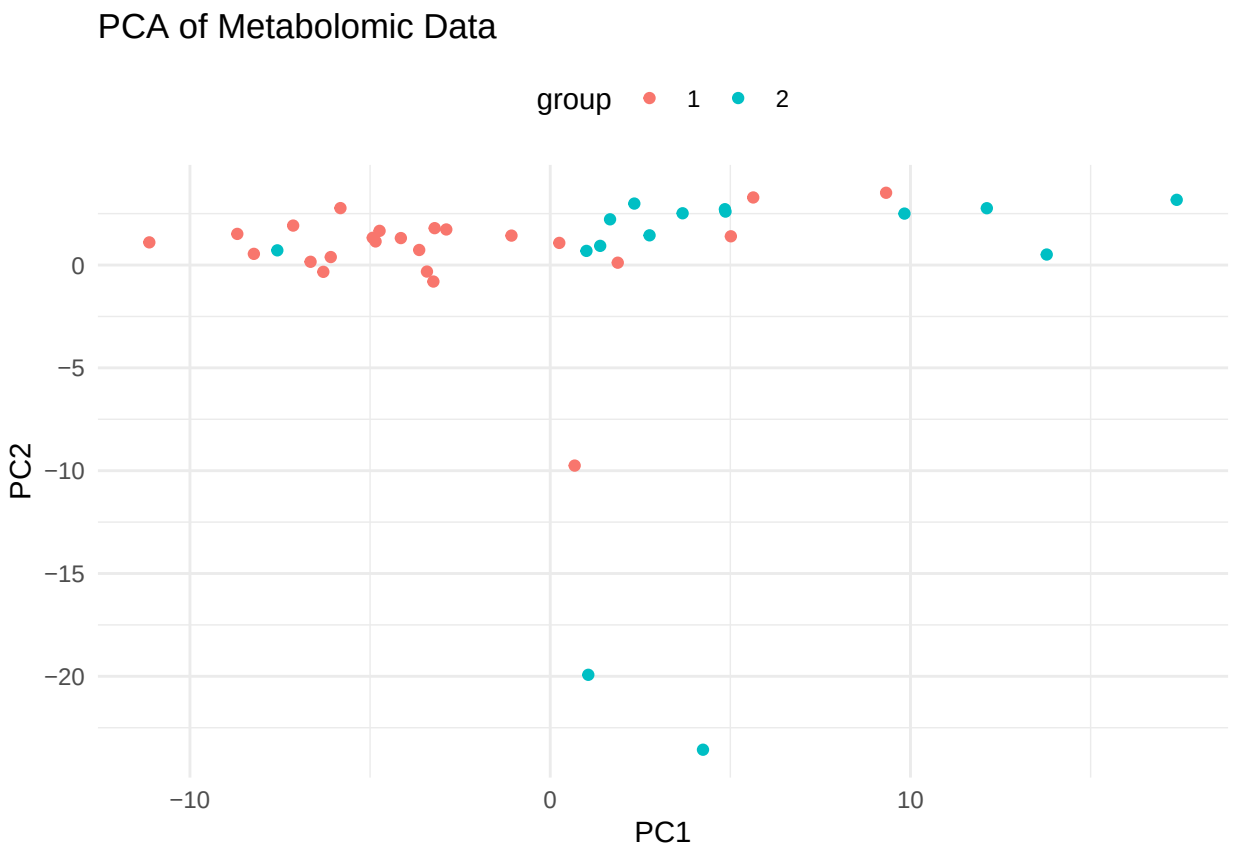
```
##                 Class Metabolite.abbreviation  Metabolite  Platform
## 1 glycerophospholipids             PC aa C40:2 PC aa C40:2 FIA-MS/MS
## 2 glycerophospholipids             PC aa C42:4 PC aa C42:4 FIA-MS/MS
## 3 glycerophospholipids             PC ae C38:2 PC ae C38:2 FIA-MS/MS
## 4 glycerophospholipids             PC ae C42:1 PC ae C42:1 FIA-MS/MS
## 5 glycerophospholipids             PC ae C44:3 PC ae C44:3 FIA-MS/MS
##         Data.type  X
## 1 Semi-quantified NA
## 2 Semi-quantified NA
## 3 Semi-quantified NA
## 4 Semi-quantified NA
## 5 Semi-quantified NA
```

From the available information from `AAInformation_S006`, we can see that all the top 5 variables from PC1 and PC2 correspond to glycerophospholipids, all of them semi-quantified using FIA-MS/MS. Now we are going to plot all the individuals and divide it by group, gender, and surgery:

```
library(ggplot2)
plot_pca_data <- as.data.frame(pca_result$x)
plot_pca_data$group <- as.character(data_pca_clean$Group)

pca_plot <- ggplot(plot_pca_data, aes(x = PC1, y = PC2, color = group)) +
    geom_point() +
    labs(title = "PCA of Metabolomic Data",
         x = "PC1",
         y = "PC2") +
    theme_minimal() +
    theme(legend.position = "top")
```

```
pca_plot
```



From the plot of the PCA by group, we can observe that the PC1 clearly separates quite well both groups. We can perform a statistical text with the PC1 to observe if the differences are significant:

```
var.test(plot_pca_data$PC1[plot_pca_data$group == '1'],
         plot_pca_data$PC1[plot_pca_data$group == '2'])
```

```
##
##  F test to compare two variances
##
## data:  plot_pca_data$PC1[plot_pca_data$group == "1"] and plot_pca_data$PC1[plot_pca_data$group == "2
## F = 0.61555, num df = 23, denom df = 14, p-value = 0.2926
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.2197726 1.5367965
## sample estimates:
## ratio of variances
##          0.6155541
```

```
t.test(PC1 ~ group, data = plot_pca_data, var.equal = TRUE)
```

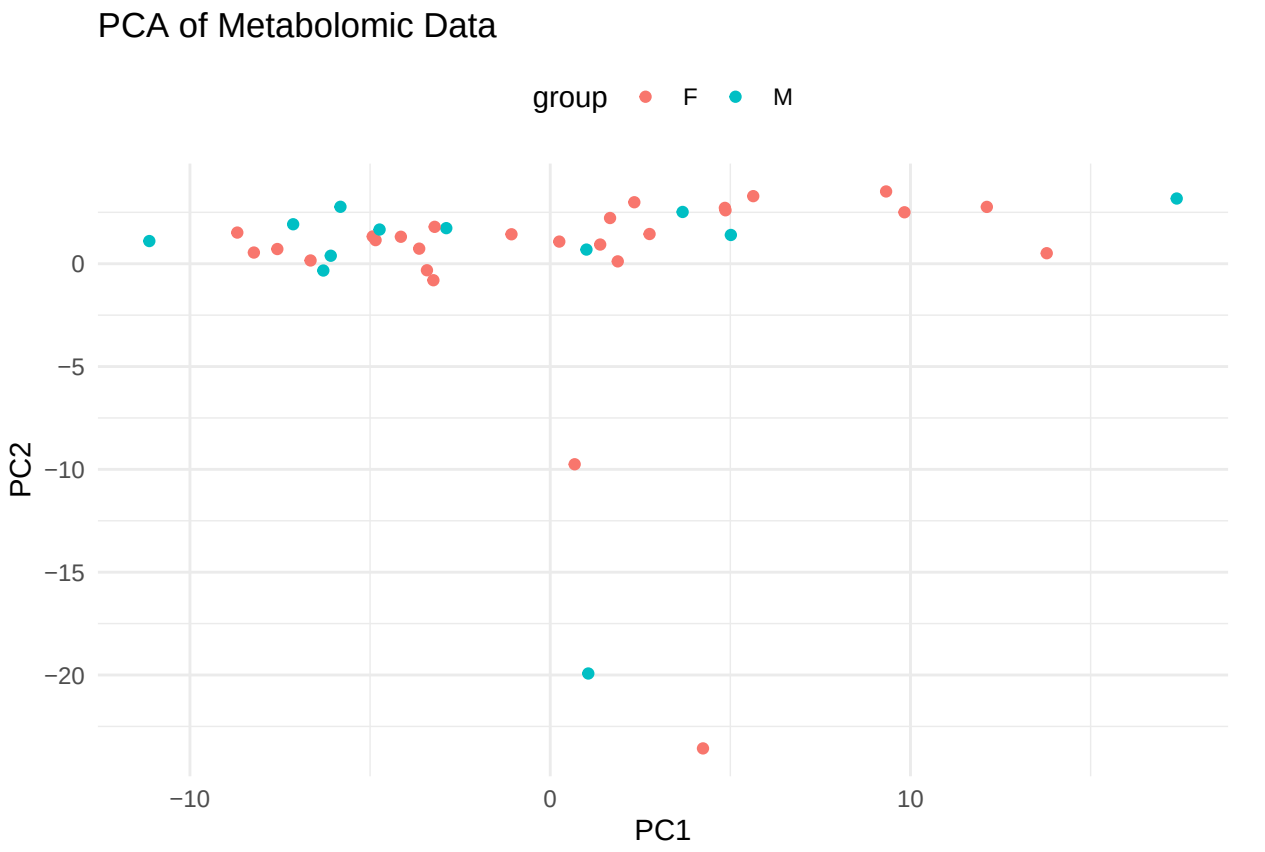
```
##
##  Two Sample t-test
##
## data:  PC1 by group
## t = -4.4879, df = 37, p-value = 6.776e-05
## alternative hypothesis: true difference in means between group 1 and group 2 is not equal to 0
```

```
## 95 percent confidence interval:
##  -11.539007  -4.360634
## sample estimates:
## mean in group 1 mean in group 2
##       -3.057623        4.892197
```

Indeed, the PC1 shows significant differences between groups 1 and 2 (p-value = 6.776e-05).

```
plot_pca_data$group <- as.character(DataValues_S013$GENDER)

pca_plot <- ggplot(plot_pca_data, aes(x = PC1, y = PC2, color = group)) +
    geom_point() +
    labs(title = "PCA of Metabolomic Data",
         x = "PC1",
         y = "PC2") +
    theme_minimal() +
    theme(legend.position = "top")
pca_plot
```




```
var.test(plot_pca_data$PC1[plot_pca_data$group == 'F'],
         plot_pca_data$PC1[plot_pca_data$group == 'M'])
```

```
##
##  F test to compare two variances
##
## data:  plot_pca_data$PC1[plot_pca_data$group == "F"] and plot_pca_data$PC1[plot_pca_data$group == "M
## F = 0.65389, num df = 26, denom df = 11, p-value = 0.3609
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
```

```
##  0.2074817 1.6584814
## sample estimates:
## ratio of variances
##          0.6538909
```

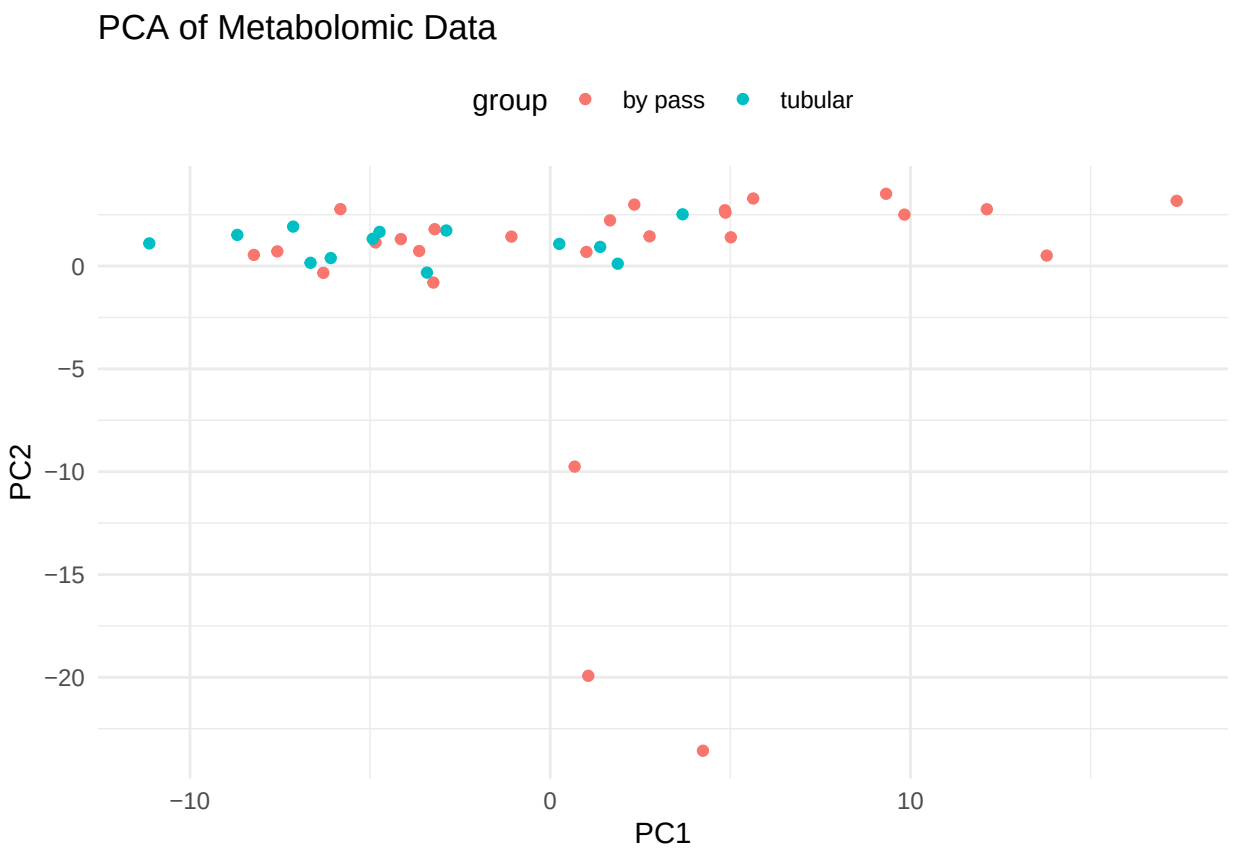
```r
t.test(PC1 ~ group, data = plot_pca_data, var.equal = TRUE)
```

```
##
##  Two Sample t-test
##
## data:  PC1 by group
## t = 0.83529, df = 37, p-value = 0.4089
## alternative hypothesis: true difference in means between group F and group M is not equal to 0
## 95 percent confidence interval:
##  -2.737718  6.578116
## sample estimates:
## mean in group F mean in group M
##       0.5908305      -1.3293687
```

There are no significant differences between genders (p-value = 0.4089).

```r
plot_pca_data$group <- as.character(DataValues_S013$SURGERY)

pca_plot <- ggplot(plot_pca_data, aes(x = PC1, y = PC2, color = group)) +
    geom_point() +
    labs(title = "PCA of Metabolomic Data",
         x = "PC1",
         y = "PC2") +
    theme_minimal() +
    theme(legend.position = "top")
pca_plot
```




```
var.test(plot_pca_data$PC1[plot_pca_data$group == 'by pass'],
         plot_pca_data$PC1[plot_pca_data$group == 'tubular'])
```

```
##
##  F test to compare two variances
##
## data:  plot_pca_data$PC1[plot_pca_data$group == "by pass"] and plot_pca_data$PC1[plot_pca_data$group
## F = 2.3284, num df = 25, denom df = 12, p-value = 0.1278
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.7741328 5.8556423
## sample estimates:
## ratio of variances
##           2.328389
```

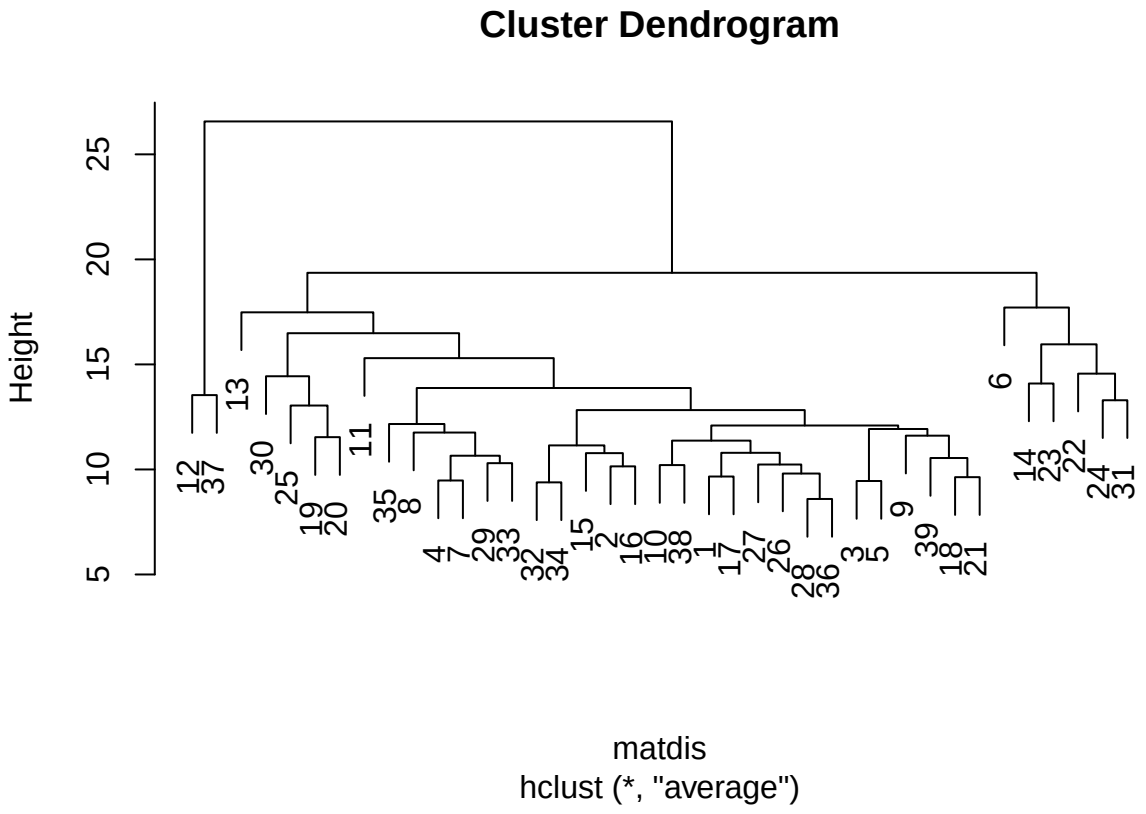
```
t.test(PC1 ~ group, data = plot_pca_data, var.equal = TRUE)
```

```
##
##  Two Sample t-test
##
## data:  PC1 by group
## t = 2.6922, df = 37, p-value = 0.0106
## alternative hypothesis: true difference in means between group by pass and group tubular is not equal
## 95 percent confidence interval:
##  1.383647 9.802373
## sample estimates:
## mean in group by pass mean in group tubular
##              1.864337             -3.728673
```

When we consider the type of surgery, we observe that "by pass" group is much more scattered than the "tubular" one. Indeed, both types of surgery are significantly different considering the PC1 (p-value= 0.0106).

Another interesting technique is applying clustering techniques. We'll start by using a hierarchical technique and draw a dendrogram:

```
data_pca_clean$Group <- NULL
hierarchical_data <- scale(data_pca_clean)
matdis <- dist(hierarchical_data)
hierarchical_data.hc <- hclust(matdis, method="average")
plot(hierarchical_data.hc)
```




```
cor(matdis, cophenetic(hierarchical_data.hc))
```

```
## [1] 0.8676329
```

The cophenetic correlation is quite high (0.8676329), which indicates that the dendrogram reflects the actual distances in your data quite well, capturing about 86.76% of the original distance structure of the data. There are two individuals (12 and 37) which cluster apart from the rest, and indeed correspond to the outlier point from Group 2 that we observed in the PCA.

GitHub repository

The first step is checking that we have Git installed through a `bash` terminal cell:

```
which git
git --version
```

```
## /usr/bin/git
## git version 2.39.5 (Apple Git-154)
```

To create a new Git repository in RStudio, we go to `File` > `New Project...` > `New Directory` > `New Project`. Then we give it the name `Canela-Grimau-Marc-PEC1` and we also select `Create a git repository`.

To commit the data, we go the the tab named `Git` and we press `Commit`. Then we select the files to commit and enter a message.

Finally, we can upload the code to GitHub creating a new repository:

```
library(usethis)
usethis::use_github()
```

We can now push any changes. In the tab `Git`, we first commit any changes. Then, we can click on `Push` to upload the changes into GitHub. The repository can be found in:

https://github.com/marccanela/Canela-Grimau-Marc-PEC1