A short tutorial on using *metaX* for high-throughput mass spectrometry-based metabolomic data analysis

Bo Wen

May 15, 2016

Contents

1 Introduction

The *metaX* package provides a integrated pipeline for mass spectrometry-based metabolomic data analysis. It includes the stages peak detection, data preprocessing, normalization, missing value imputation, univariate statistical analysis, multivariate statistical analysis such as PCA and PLS-DA, metabolite identification, pathway analysis, power analysis, feature selection and modeling, data quality assessment and HTML-based report generation. This document describes how to use the function included in the R package *metaX*.

2 Example data

We are going to use two public datasets to show the functions in this tutorial. One is from the reference [1]. This data can be accessed through the *faahKO* package. The samples in this data set can be divided into two groups (group knockout or KO, group wild type or WT) which each group includes six samples. The other is from the reference [2] and can be downloaded from MetaboLights.

3 Using metaX

3.1 Data import

metaX has been designed to accept diverse data types including compound concentration/intensity tables which generated by *XCMS*, MZmine [3], Progenesis QI (csv format) or other software which can be used for peak picking, as well as open file format MS data (such as mzXML, NetCDF).

3.1.1 Input compound intensity table file

metaX accepts several peak intensity/concentration formats.

- Progenesis QI peak picking result file (csv format). It can be imported by the following function:

```
## not run
para <- importDataFromQI(para,file="qi.csv")
```

- *XCMS* peak picking result file (txt or csv format). It can be imported by the following function:

```
## not run
para <- importDataFromXCMS(para,file="xcms.txt")
```

- MetaboAnalyst comatible peak intensity table (csv format). There is an example input file (lcms_table.csv) in the website of MetaboAnalyst. It can be imported by the following function:

```
## not run
para <- importDataFromMetaboAnalyst(para,
 file="http://www.metaboanalyst.ca/resources/data/lcms_table.csv")
```

- *metaX* comatible peak intensity table (txt or csv format), in which both sample or feature names must be unique. In this file, there must be a column named **"name"** which listed the feature IDs. The other columns contains the intensity/concentration data and the names of the columns are sample names. This data file can be imported by the following method:

```
para <- new("metaXpara")
pfile <- system.file("extdata/MTBLS79.txt",package = "metaX")
rawPeaks(para) <- read.delim(pfile,check.names = FALSE)
head(para@rawPeaks[,1:4])
##        name batch01_QC01 batch01_QC02 batch01_QC03
## 1  78.02055      14023.0      13071.0      15270.0
## 2 452.00345      22455.0      10737.0      27397.0
## 3 138.96337       6635.4       8062.3       6294.6
## 4  73.53838      26493.0      26141.0      25944.0
## 5 385.12885      57625.0      56964.0      59045.0
## 6 237.02815     105490.0      90166.0      92315.0
```

After importing the peak intensity data, the user also need to set the sample list file. A very important step in the *metaX* pipeline is the definition of a sample list file, that provides the file names (sample), batch number (batch), sample class (class) and the sample injection order (order). Please note that if the sample list file contains quality control (QC) sample, the value in the column of class must be "NA". The sample list file can be set as below:

```
## set the sample list file path
sfile <- system.file("extdata/MTBLS79_sampleList.txt",package = "metaX")
sampleListFile(para) <- sfile

## print the object:
para

## total samples:

## 172

## sample group information:

##   Group Number of samples
## 1     C                66
## 2    QC                38
```

```
## 3     S                68
```

```
## batch information:
```

```
##   Batch Number of samples
## 1     1                91
## 2     5                17
## 3     6                20
## 4     7                44
```

The content of this sample list is shown below:

```
##         sample batch class order
## 1 batch01_QC01     1  <NA>     1
## 2 batch01_QC02     1  <NA>     2
## 3 batch01_QC03     1  <NA>     3
## 4  batch01_C05     1     C     4
## 5  batch01_S07     1     S     5
## 6  batch01_C10     1     C     6
```

3.1.2 Input MS data files

If the user provides MS data (NetCDF, mzXML and so on), *metaX* uses the *XCMS* to perform peak picking, followed by the CAMERA [4] package to perform peak annotation. In this situation, the MS data must be placed in two subdirectories of a single folder like below:

```
list.files(system.file("cdf", package = "faahKO"),
           recursive = TRUE,full.names = TRUE)
```

```
##  [1] "/home/biocbuild/bbs-3.3-bioc/R/library/faahKO/cdf/KO/ko15.CDF"
##  [2] "/home/biocbuild/bbs-3.3-bioc/R/library/faahKO/cdf/KO/ko16.CDF"
##  [3] "/home/biocbuild/bbs-3.3-bioc/R/library/faahKO/cdf/KO/ko18.CDF"
##  [4] "/home/biocbuild/bbs-3.3-bioc/R/library/faahKO/cdf/KO/ko19.CDF"
##  [5] "/home/biocbuild/bbs-3.3-bioc/R/library/faahKO/cdf/KO/ko21.CDF"
##  [6] "/home/biocbuild/bbs-3.3-bioc/R/library/faahKO/cdf/KO/ko22.CDF"
##  [7] "/home/biocbuild/bbs-3.3-bioc/R/library/faahKO/cdf/WT/wt15.CDF"
##  [8] "/home/biocbuild/bbs-3.3-bioc/R/library/faahKO/cdf/WT/wt16.CDF"
##  [9] "/home/biocbuild/bbs-3.3-bioc/R/library/faahKO/cdf/WT/wt18.CDF"
## [10] "/home/biocbuild/bbs-3.3-bioc/R/library/faahKO/cdf/WT/wt19.CDF"
## [11] "/home/biocbuild/bbs-3.3-bioc/R/library/faahKO/cdf/WT/wt21.CDF"
## [12] "/home/biocbuild/bbs-3.3-bioc/R/library/faahKO/cdf/WT/wt22.CDF"
```

In the *metaX* package, it uses a *metaXpara-class* object to manage the file path information and other parameters for data processing. We can set the input files path like below:

```
## create a metaXpara-class object
#library("metaX")
```

```
para <- new("metaXpara")
## set the MS data path
dir.case(para) <- system.file("cdf/KO", package = "faahKO")
dir.ctrl(para) <- system.file("cdf/WT", package = "faahKO")
```

After setting the MS data file path, the user also need to set the sample list file:

```
## set the sample list file path
sampleFile <- system.file("extdata/faahKO_sampleList.txt",
                          package = "metaX")
sampleListFile(para) <- sampleFile
samList <- read.delim(sampleFile)
print(samList)
```

```
##    sample batch class order
## 1    ko15     1    KO     1
## 2    ko16     1    KO     2
## 3    ko18     1    KO     3
## 4    ko19     1    KO     4
## 5    ko21     1    KO     5
## 6    ko22     1    KO     6
## 7    wt15     1    WT     7
## 8    wt16     1    WT     8
## 9    wt18     1    WT     9
## 10   wt19     1    WT    10
## 11   wt21     1    WT    11
## 12   wt22     1    WT    12
```

In general, the user also needs to set several other parameters for peak picking and annotation. Several parameters related to peak picking and annotation must be set.

- Peak picking. The peak picking related parameters can be set as below:

```
## set parameters for peak picking
xcmsSet.peakwidth(para) <- c(20,50)
xcmsSet.snthresh(para) <- 10
xcmsSet.prefilter(para) <- c(3,100)
xcmsSet.noise(para) <- 0
xcmsSet.nSlaves(para) <- 4
```

- Peak annotation. The peak annotation related parameters can be set as below:

```
## set parameters for peak annotation
group.bw(para) <- 5
group.minfrac(para) <- 0.3
group.mzwid(para) <- 0.015
group.max(para) <- 1000
```

For the complete parameters, please see the help page of metaXpara-class.

In *metaX*, there is a function `peakFinder()`, which can be used to do the peak picking and annotation.

```
p <- peakFinder(para)
```

For the complete parameters, please see the help page of metaXpara-class.

3.2 Integrated function `metaXpipe()`

The function metaXpipe automates the whole data analysis process. In general, the user only need to use this function to do most of the analysis. It includes the following steps:

1. Peak picking and annotation when input MS data.
2. Data pre-processing: Firstly, if an metabolite feature is detected in <50% of QC samples or detected in <20% of experimental samples, it is removed from the rest data analysis.
3. Missing value imputation.
4. Removal of outliers (sample).
5. Normalization.
6. Feature filter according to the CV (30%) of features in QC sample. It only works when there are QC samples.
7. Univariate statistical analysis, such as **t test**, **Mann-Whitney U test** and **ROC** analysis.
8. PCA (score plot, loading plot)
9. PLS-DA (score plot, loading plot, permutation test, Q2, R2)
10. Assessment of data quality:
    (a) The peak number distribution
    (b) The number of missing value distribution
    (c) The boxplot of peak intensity
    (d) The total peak intensity distribution
    (e) The correlation heatmap of QC samples if available
    (f) Metabolite m/z (or mass) distribution
    (g) Plot of m/z versus retention time
    (h) PCA score or loading plot of all samples

A complete example to show how to run the integrated analysis is shown below:

```
para <- new("metaXpara")
pfile <- system.file("extdata/MTBLS79.txt",package = "metaX")
sfile <- system.file("extdata/MTBLS79_sampleList.txt",package = "metaX")
rawPeaks(para) <- read.delim(pfile,check.names = FALSE)
sampleListFile(para) <- sfile
ratioPairs(para) <- "S:C"
plsdaPara <- new("plsDAPara")
para@outdir <- "output"
p <- metaXpipe(para,plsdaPara=plsdaPara,cvFilter=0.3,remveOutlier = TRUE,
  outTol = 1.2, doQA = TRUE, doROC = TRUE, qcsc = FALSE,
  nor.method = "pqn", pclean = TRUE, t = 1, scale = "uv",)
```

In the above example, the class *plsDAPara* from *metaX* is used to control the parameters for **PLS-DA** analysis. For the complete parameters, please see the help page of *plsDAPara-class*.

After the analysis has completed, the file "index.html" in the output directory can be opened in a web browser to access report generated.

3.3 Function modules

3.3.1 Peak picking and annotation

In metaX, the function `peakFinder()` can be used to do the peak picking and annotation. It uses the *XCMS* to perform peak picking, followed by the CAMERA [4] package to perform peak annotation.

3.3.2 Missing value imputation

Missing value imputation. Missing values is a common phenomenon in a typical quantitative metabolomics dataset. There are several methods provided by *metaX* to process the missing value. Currently, we implemented a variety of methods which enable users to automatically perform missing value imputation by min, Probabilistic PCA (PPCA), Bayesian PCA (BPCA), k nearest-neighbor (KNN), missForest and Singular Value Decomposition Imputation (SVDImpute).

When the user uses the function `metaXpipe` to do the analysis, the following code can be used to set the imputation method:

```
## bpca, svdImpute, knn, rf, min
missValueImputeMethod(para) <- "knn"
```

Also, the user can use the function `missingValueImpute` to perform the missing value imputation:

```
para <- new("metaXpara")
pfile <- system.file("extdata/MTBLS79.txt",package = "metaX")
sfile <- system.file("extdata/MTBLS79_sampleList.txt",package = "metaX")
rawPeaks(para) <- read.delim(pfile,check.names = FALSE)
sampleListFile(para) <- sfile
para <- reSetPeaksData(para)
## bpca, svdImpute, knn, rf, min
para <- missingValueImpute(para,method="knn")

## missingValueImpute:  value

## Sun May 15 22:35:27 2016 Missing value imputation for 'value'

## Missing value in total:  3940

## Missing value in QC sample:  678

## Missing value in non-QC sample:  3262
```

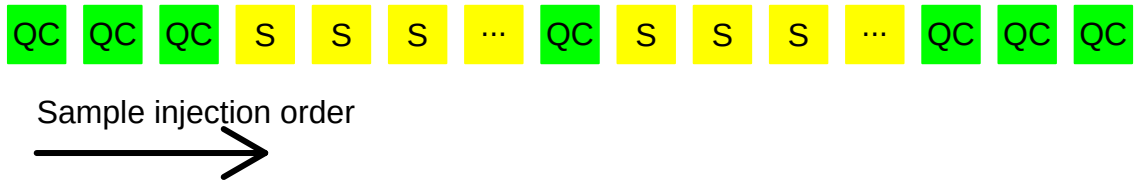

Figure 1: The experiment design which contained QC samples.

```
## Sun May 15 22:35:27 2016 The ratio of missing value:  4.5814%
## <=0:  0
## Missing value in total after missing value inputation:  0
## <=0 value in total after missing value inputation:  0
```

3.3.3 Normalization

Currently, we implemented several methods to perform data normalization, such as the QC-robust spline batch correction (QC-RSC) [5], sum, VSN, probabilistic quotient normalization (PQN) [6], quantiles and robust quantiles.

If there are pooled QC samples in the experiment (the experiment is like figure 1), the function `doQCRLSC` can be used to perform the QC-RSC normalization. This method is implemented to correct data within batch experiment analytical variation, and batch-to-batch variation in large-scale studies.

```
para <- new("metaXpara")
pfile <- system.file("extdata/MTBLS79.txt",package = "metaX")
sfile <- system.file("extdata/MTBLS79_sampleList.txt",package = "metaX")
rawPeaks(para) <- read.delim(pfile,check.names = FALSE)[1:100,]
sampleListFile(para) <- sfile
para <- reSetPeaksData(para)
para <- missingValueImpute(para)
res <- doQCRLSC(para,cpu=1)
```

Except the QC-RSC method, the user can use the other method (PQN, sum et al.) as below:

```
para <- new("metaXpara")
pfile <- system.file("extdata/MTBLS79.txt",package = "metaX")
sfile <- system.file("extdata/MTBLS79_sampleList.txt",package = "metaX")
rawPeaks(para) <- read.delim(pfile,check.names = FALSE)
```

```
sampleListFile(para) <- sfile
para <- reSetPeaksData(para)
para <- missingValueImpute(para)
para <- metaX::normalize(para,method="pqn",valueID="value")
```

3.3.4 Pre-processing of raw peak data

In *metaX*, two functions, `filterQCPeaks()` and `filterPeaks()` can be used to filter features. In general, an metabolite feature is detected in <50% of QC samples (by using `filterQCPeaks()`) or detected in <20% (by using `filterPeaks()`) of experimental samples, it is removed from the rest data analysis.

```
para <- new("metaXpara")
pfile <- system.file("extdata/MTBLS79.txt",package = "metaX")
sfile <- system.file("extdata/MTBLS79_sampleList.txt",package = "metaX")
rawPeaks(para) <- read.delim(pfile,check.names = FALSE)
sampleListFile(para) <- sfile
para <- reSetPeaksData(para)
p <- filterPeaks(para,ratio=0.2)
p <- filterQCPeaks(para,ratio=0.5)
```

3.3.5 Removal of outliers

metaX provides function (RfunctionautoRemoveOutlier()) to automatically remove the outlier samples in the pre-processed data based on expansion of the Hotellings T2 distribution ellipse. A sample within the first and second component PCA score plot beyond the expanded ellipse is removed, then the PCA model is recalculated. In default, three rounds of outlier removal are performed.

```
para <- new("metaXpara")
pfile <- system.file("extdata/MTBLS79.txt",package = "metaX")
sfile <- system.file("extdata/MTBLS79_sampleList.txt",package = "metaX")
rawPeaks(para) <- read.delim(pfile,check.names = FALSE)
sampleListFile(para) <- sfile
para <- reSetPeaksData(para)
para <- missingValueImpute(para)
rs <- autoRemoveOutlier(para,valueID="value")
```

3.3.6 Power analysis and sample size estimation

In *metaX*, the function `powerAnalyst()` can be used to do power analysis and sample size estimation.

```
para <- new("metaXpara")
pfile <- system.file("extdata/MTBLS79.txt",package = "metaX")
sfile <- system.file("extdata/MTBLS79_sampleList.txt",package = "metaX")
rawPeaks(para) <- read.delim(pfile,check.names = FALSE)
sampleListFile(para) <- sfile
para <- reSetPeaksData(para)
para <- missingValueImpute(para)
```

```
## missingValueImpute:  value
```

```
## Sun May 15 22:35:28 2016 Missing value imputation for 'value'
```

```
## Missing value in total:  3940
```

```
## Missing value in QC sample:  678
```

```
## Missing value in non-QC sample:  3262
```

```
## Sun May 15 22:35:28 2016 The ratio of missing value:  4.5814%
```

```
## <=0:  0
```

```
## Missing value in total after missing value inputation:  0
```

```
## <=0 value in total after missing value inputation:  0
```

```
para <- metaX::normalize(para)
```

```
## normalize:  value
```

```
para <- transformation(para,valueID = "value")
para <- preProcess(para,scale = "pareto",valueID="value")
res <- powerAnalyst(para,group=c("C","S"),log=FALSE,
                    maxInd=200,showPlot = TRUE)
```

```
##
##  C  S
## 66 68
```

```
## .//metaX-power.pdf
```

```
print(res)
```

```
## [1] 0.7543686
```

3.3.7 Metabolite correlation network analysis

In *metaX*, the function `plotNetwork()` can be used to do correlation network analysis.

```
para <- new("metaXpara")
pfile <- system.file("extdata/MTBLS79.txt",package = "metaX")
sfile <- system.file("extdata/MTBLS79_sampleList.txt",package = "metaX")
```

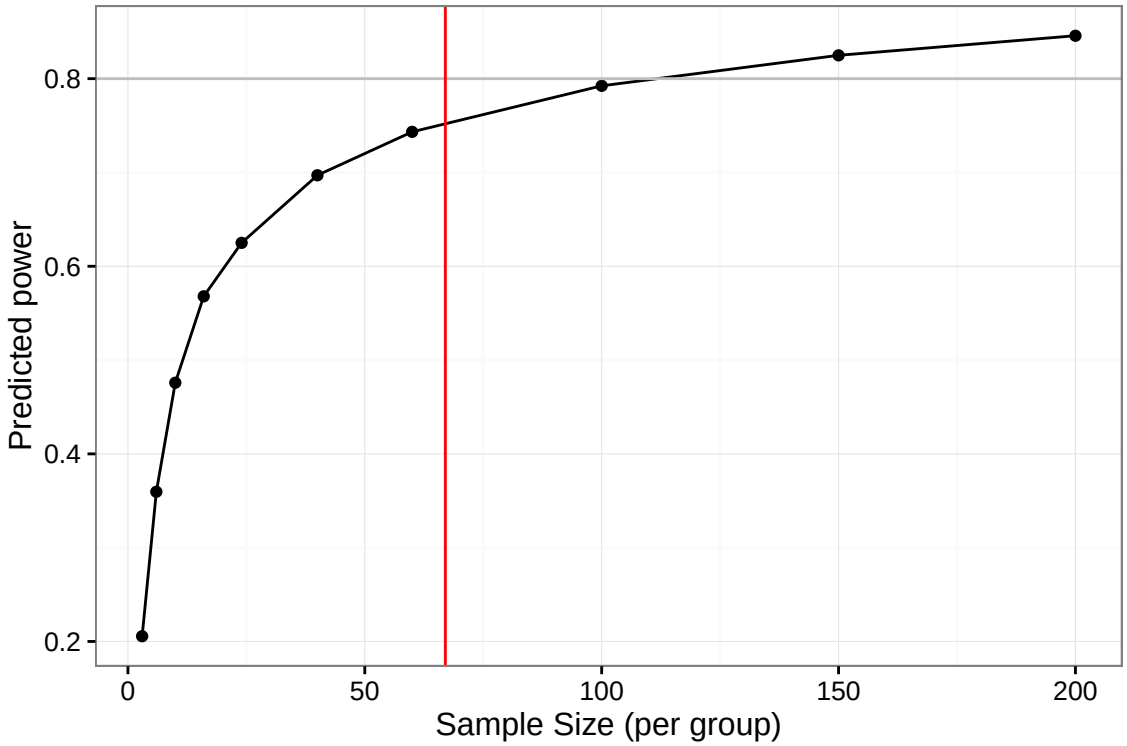

Figure 2: The power and sample size distribution.

```
rawPeaks(para) <- read.delim(pfile,check.names = FALSE)
sampleListFile(para) <- sfile
para <- reSetPeaksData(para)
para <- missingValueImpute(para)
## missingValueImpute:  value
## Sun May 15 22:35:47 2016 Missing value imputation for 'value'
## Missing value in total:  3940
## Missing value in QC sample:  678
## Missing value in non-QC sample:  3262
## Sun May 15 22:35:47 2016 The ratio of missing value:  4.5814%
## <=0:  0
## Missing value in total after missing value inputation:  0
## <=0 value in total after missing value inputation:  0
op <- par()
par(mar=c(0,0,0,0))
gg <- plotNetwork(para,group=c("S","C"),degree.thr = 10,
```

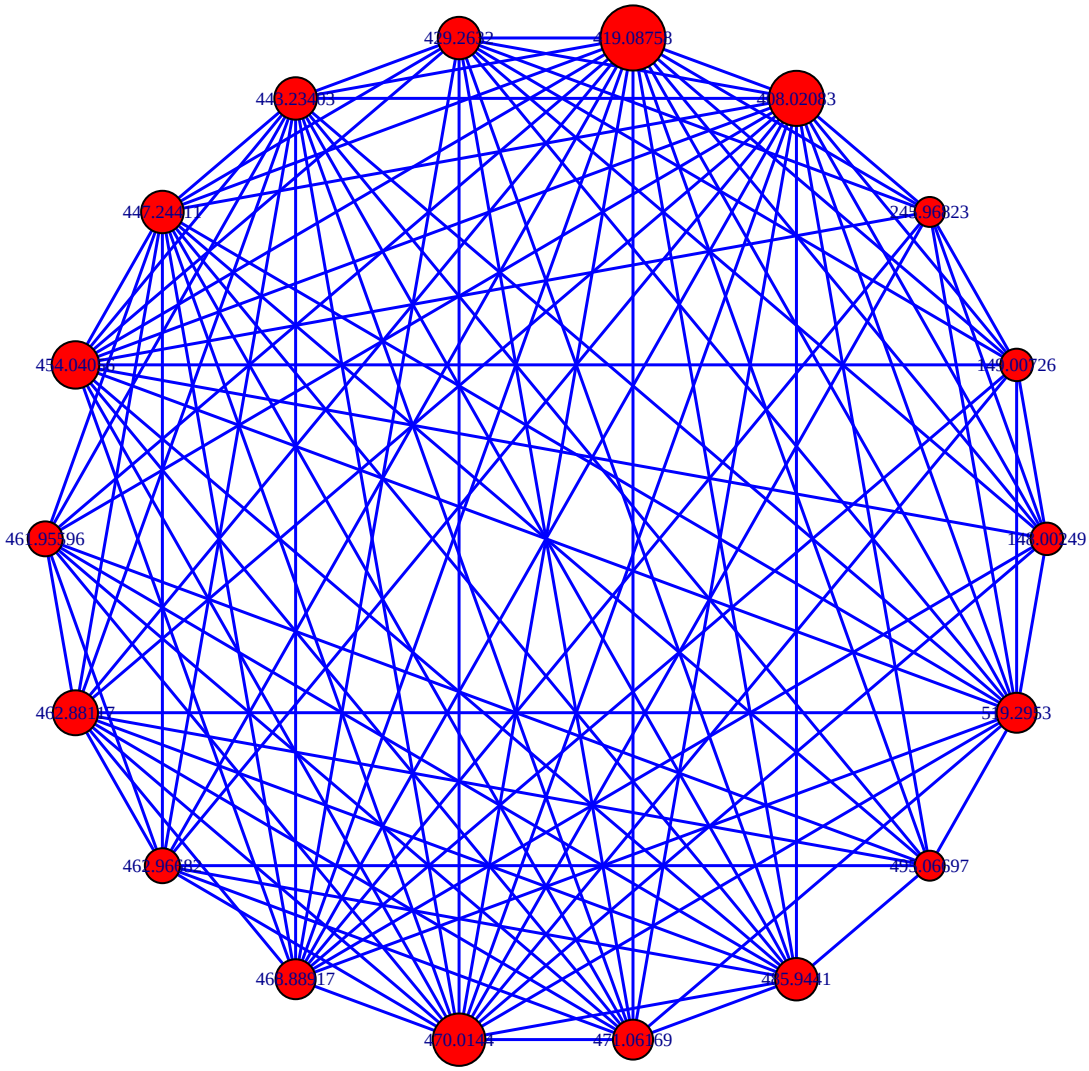

Figure 3: The correlation network.

```
                    cor.thr = 0.85,showPlot=TRUE)
## V: 18
## E: 111
## .//metaX-network.pdf
```

```
par(op)
```

3.3.8 Metabolite identification

In *metaX*, the function `metaboliteAnnotation()` can be used to do the metabolite identification. Currently, only HMDB [7] is supported.

3.3.9 Functional analysis

At the present, metaX supports the function for metabolite pathway analysis based on IMPaLA [8]. The function `pathwayAnalysis()` can be used to do the pathway analysis.

```
res <- pathwayAnalysis(id=c("HMDB00060","HMDB00056","HMDB00064"),
                       outfile="pathway.csv")

## Save result to file:  pathway.csv
```

Part of the pathway analysis result is shown below:

```
xtable::xtable(head(res[,2:4]))
```

	pathway_source	num_overlapping_metabolites	overlapping_metabolites
1	KEGG	2	HMDB00056;HMDB00060
2	Reactome	2	HMDB00056;HMDB00060
3	Reactome	1	HMDB00060
4	HumanCyc	1	HMDB00064
5	Reactome	1	HMDB00056
6	KEGG	1	HMDB00060

3.3.10 Biomarker analysis

metaX uses the functions from the R package "caret" to perform the biomarker selection, model creation and performance evaluation.

```
para <- new("metaXpara")
pfile <- system.file("extdata/MTBLS79.txt",package = "metaX")
sfile <- system.file("extdata/MTBLS79_sampleList.txt",package = "metaX")
rawPeaks(para) <- read.delim(pfile,check.names = FALSE)[1:100,]
sampleListFile(para) <- sfile
para <- reSetPeaksData(para)
res <- featureSelection(para, group=c("S","C"), method = "rf",
                        valueID = "value", fold = 5)
```

The biomarker selection result is shown below:

```
xtable::xtable(head(res$results))
```

The best feature(s) is below:

	Variables	Accuracy	Kappa	AccuracySD	KappaSD
1	1.00	0.98	0.95	0.02	0.04
2	2.00	0.99	0.98	0.02	0.03
3	3.00	1.00	1.00	0.00	0.00
4	4.00	1.00	1.00	0.00	0.00
5	5.00	1.00	1.00	0.00	0.00
6	6.00	1.00	1.00	0.00	0.00

```
print(res$optVariables)
```

[1] "461.95596" "204.13419" "295.04729"

3.3.11 Data transformation

There are two methods which can be used to do transformation, log transformation and cube root transformation. The function `transformation()` can be used to do transformation like below:

```
para <- new("metaXpara")
pfile <- system.file("extdata/MTBLS79.txt",package = "metaX")
sfile <- system.file("extdata/MTBLS79_sampleList.txt",package = "metaX")
rawPeaks(para) <- read.delim(pfile,check.names = FALSE)
sampleListFile(para) <- sfile
para <- reSetPeaksData(para)
para <- missingValueImpute(para)
```

```
## missingValueImpute:  value
## Sun May 15 22:36:38 2016 Missing value imputation for 'value'
## Missing value in total:  3940
## Missing value in QC sample:  678
## Missing value in non-QC sample:  3262
## Sun May 15 22:36:38 2016 The ratio of missing value:  4.5814%
## <=0:  0
## Missing value in total after missing value inputation:  0
## <=0 value in total after missing value inputation:  0
```

```
para <- transformation(para,valueID = "value",method=1)
```

3.3.12 Data scaling

There are three methods which can be used to do scaling, "pareto", "vector", "uv". The function `preProcess()` can be used to do scaling like below:

```
para <- new("metaXpara")
pfile <- system.file("extdata/MTBLS79.txt",package = "metaX")
sfile <- system.file("extdata/MTBLS79_sampleList.txt",package = "metaX")
rawPeaks(para) <- read.delim(pfile,check.names = FALSE)
sampleListFile(para) <- sfile
para <- reSetPeaksData(para)
para <- missingValueImpute(para)
```

```
## missingValueImpute:  value
```

```
## Sun May 15 22:36:40 2016 Missing value imputation for 'value'
```

```
## Missing value in total:  3940
```

```
## Missing value in QC sample:  678
```

```
## Missing value in non-QC sample:  3262
```

```
## Sun May 15 22:36:40 2016 The ratio of missing value:  4.5814%
```

```
## <=0:  0
```

```
## Missing value in total after missing value inputation:  0
```

```
## <=0 value in total after missing value inputation:  0
```

```
para <- metaX::preProcess(para,valueID = "value",scale="uv")
```

3.3.13 Univariate statistical analysis

For univariate statistical analysis, the parametric statistical test (t test), non-parametric statistical test (Mann-Whitney U test), and classical univariate receiver operating characteristic (ROC) curve analysis are implemented. The function `peakStat()` can be used to do these analysis.

```
para <- new("metaXpara")
pfile <- system.file("extdata/MTBLS79.txt",package = "metaX")
sfile <- system.file("extdata/MTBLS79_sampleList.txt",package = "metaX")
rawPeaks(para) <- read.delim(pfile,check.names = FALSE)[1:50,]
sampleListFile(para) <- sfile
para <- reSetPeaksData(para)
para <- missingValueImpute(para)
ratioPairs(para) <- "S:C"
addValueNorm(para) <- para
plsdaPara <- new("plsDAPara")
plsdaPara@nperm <- 10
```

```
res <- peakStat(para,plsdaPara,doROC = TRUE)
```

The part of the analysis result is shown below:

```
head(res@quant)
```

```
##          ID     ratio t.test_p.value wilcox.test_p.value t.test_p.value_BHcorrect
## 1 102.09132 1.0686877   4.474695e-01        5.643959e-01             5.736789e-01
## 2 133.06077 0.7184348   2.718336e-09        1.975226e-08             1.235607e-08
## 3 133.07219 1.0000180   9.179260e-01        7.135023e-01             9.179260e-01
## 4  136.0474 1.0632009   4.299989e-01        6.134752e-01             5.657880e-01
## 5 138.96337 0.8847494   1.941816e-03        3.882242e-05             4.623372e-03
## 6 144.10058 0.9765783   2.594286e-01        2.555067e-01             3.706123e-01
##   wilcox.test_p.value_BHcorrect       roc    lowROC     upROC        VIP sample
## 1                  6.882876e-01 0.4710339 0.3776181 0.5626337 0.19444924    S:C
## 2                  7.597022e-08 0.7811943 0.6967357 0.8523173 1.41948421    S:C
## 3                  7.927803e-01 0.5184938 0.4194352 0.6122995 0.02918377    S:C
## 4                  7.303277e-01 0.4745989 0.3811107 0.5738525 0.19682177    S:C
## 5                  1.021643e-04 0.7061052 0.6138369 0.7970254 0.75880701    S:C
## 6                  3.650096e-01 0.5570410 0.4661514 0.6550858 0.33906390    S:C
```

3.3.14 PCA

The PCA analysis can be performed by the function `plotPCA()` in *metaX*. The score plot (as shown in figure 4) for 2D plot, figure 5) for 3D plot) and loading plot (as shown in figure 6)) are outputed.

```
para <- new("metaXpara")
pfile <- system.file("extdata/MTBLS79.txt",package = "metaX")
sfile <- system.file("extdata/MTBLS79_sampleList.txt",package = "metaX")
rawPeaks(para) <- read.delim(pfile,check.names = FALSE)
sampleListFile(para) <- sfile
para <- reSetPeaksData(para)
para <- missingValueImpute(para)
para <- transformation(para,valueID = "value")
metaX::plotPCA(para,valueID="value",scale="pareto",center=TRUE,rmQC = FALSE)
```

```
## $fig
## [1] ".//metaX-PCA.png"
##
## $highfig
## [1] ".//metaX-PCA.pdf"
##
## $pca
## svdImpute calculated PCA
## Importance of component(s):
```

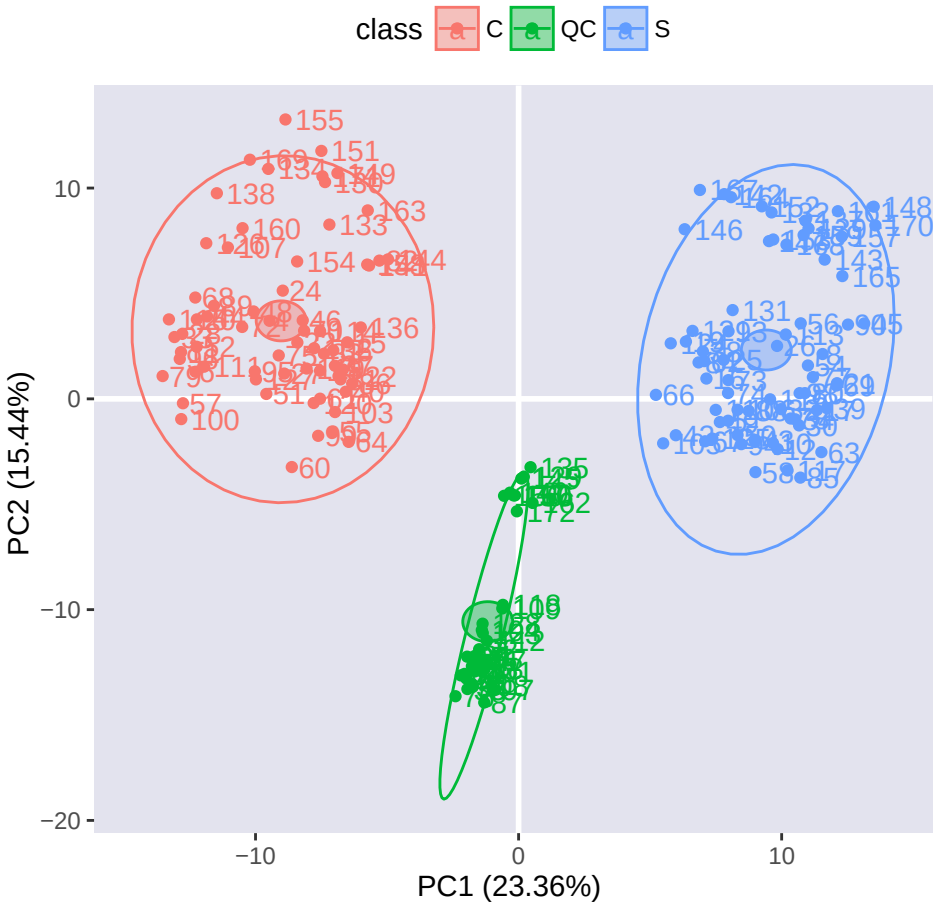

Figure 4: The PCA score plot.

```
##                  PC1    PC2    PC3
## R2            0.2336 0.1544 0.1026
## Cumulative R2 0.2336 0.3880 0.4905
## 500  Variables
## 172  Samples
## 0  NAs ( 0 %)
## 3  Calculated component(s)
## Data was mean centered before running PCA
## Data was scaled before running PCA
## Scores structure:
## [1] 172   3
## Loadings structure:
## [1] 500   3
```

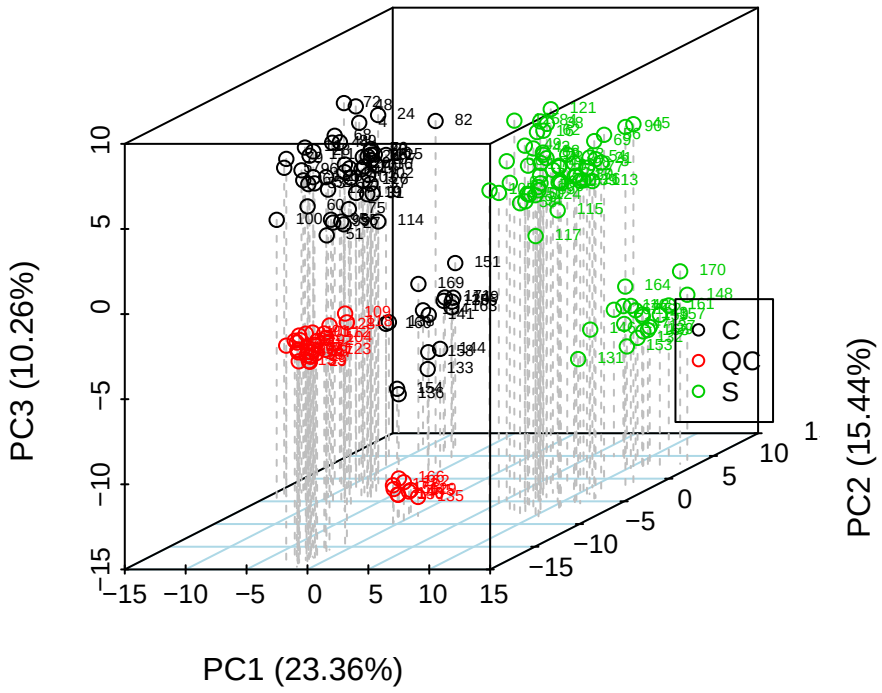

Figure 5: The 3D PCA score plot.

3.3.15 PLS-DA

The PLS-DA analysis can be performed by the function `runPLSDA()` in *metaX*. The R2, Q2 and the permutation test p-value are calculated and outputed. The permutation test plot is shown in figure 7. The score plot and loading plot are shown in figure 8 and figure 9, respectively.

```
para <- new("metaXpara")
pfile <- system.file("extdata/MTBLS79.txt",package = "metaX")
sfile <- system.file("extdata/MTBLS79_sampleList.txt",package = "metaX")
rawPeaks(para) <- read.delim(pfile,check.names = FALSE)
sampleListFile(para) <- sfile
para <- reSetPeaksData(para)
para <- missingValueImpute(para)
para <- metaX::normalize(para,method="pqn")
plsdaPara <- new("plsDAPara")
plsdaPara@nperm <- 100
plsda.res <- runPLSDA(para = para,plsdaPara = plsdaPara,
```

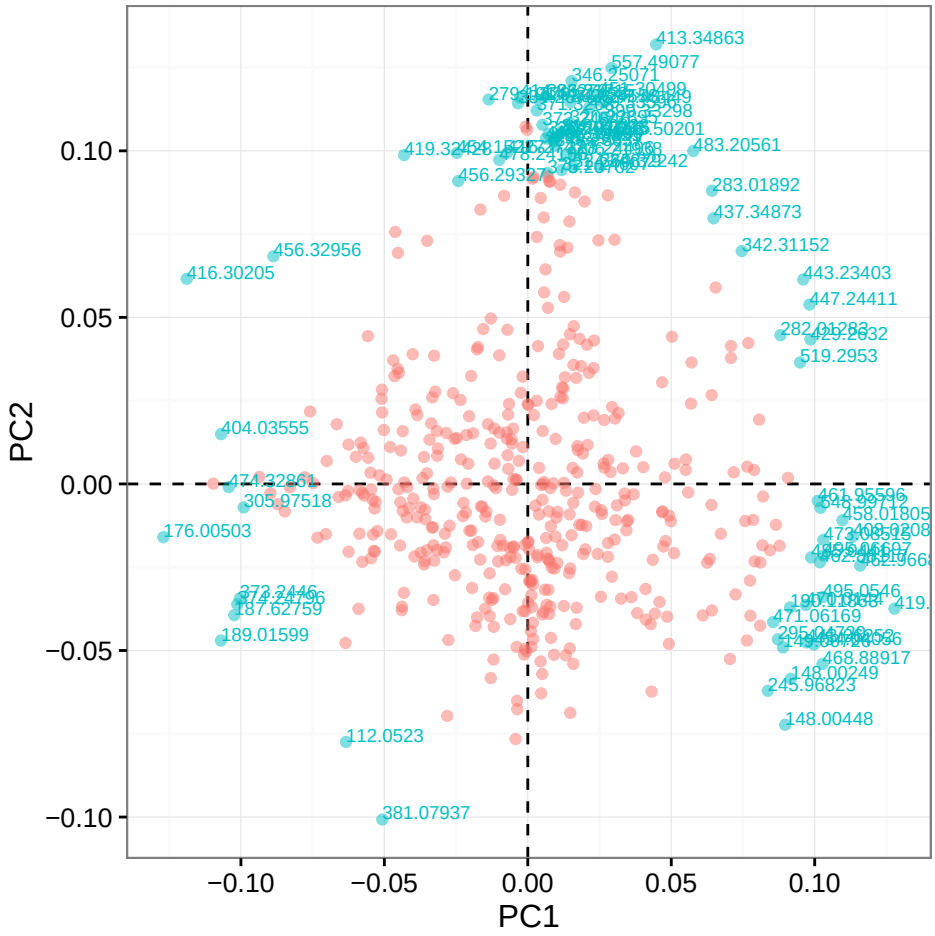

Figure 6: The PCA loading plot.

```
                    sample = c("S","C"),valueID="value")

cat("R2Y: ",plsda.res$plsda$res$R2,"\n")

## R2Y:  0.9847925

cat("Q2Y: ",plsda.res$plsda$res$Q2,"\n")

## Q2Y:  0.9803359

## permutation test
cat("P-value R2Y: ",plsda.res$pvalue$R2,"\n")

## P-value R2Y:  0

cat("P-value Q2Y: ",plsda.res$pvalue$Q2,"\n")

## P-value Q2Y:  0
```

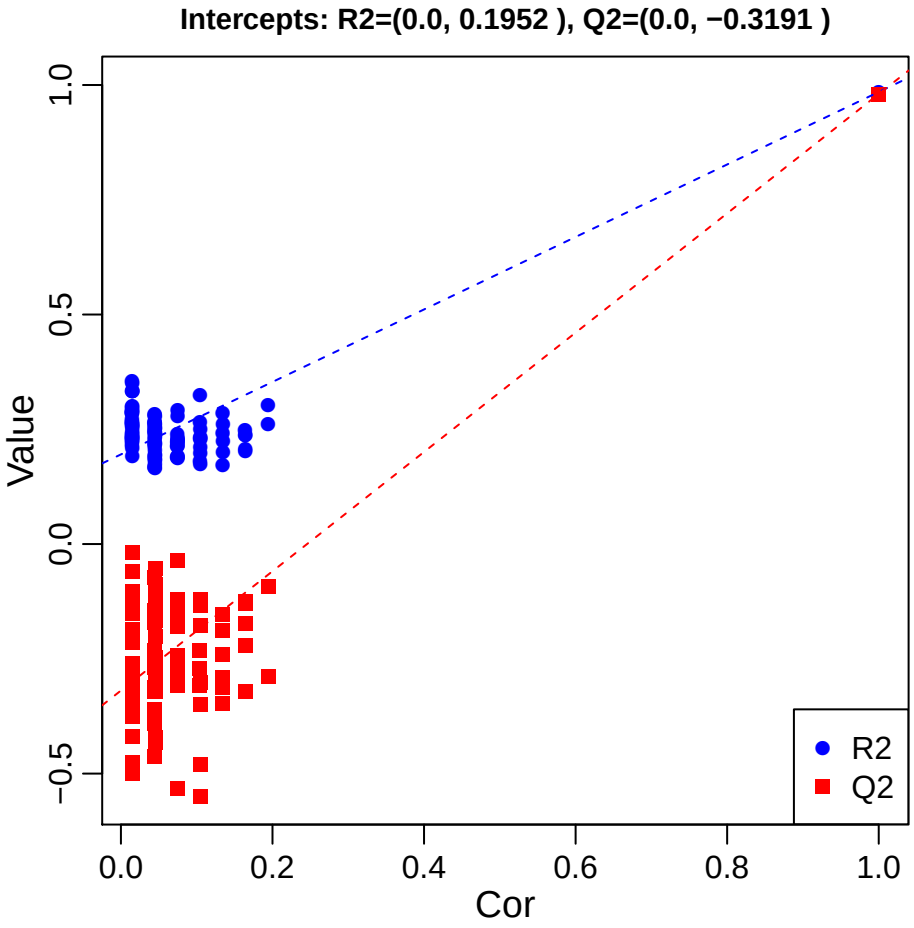

Figure 7: The permutation test plot for PLS-DA.

3.3.16 Assessment of data quality

In `metaXpipe`, pre- and post-normalization, the data quality is visually assessed in several aspects:

1. The peak number distribution
2. The number of missing value distribution
3. The boxplot of peak intensity
4. The total peak intensity distribution
5. The correlation heatmap of QC samples if available
6. The metabolite m/z (or mass) distribution
7. The plot of m/z versus retention time
8. The PCA score or loading plot of all samples (only available for post-normalization).

The example figures can be viewed in website http://wenbostar.github.io/metaX/.

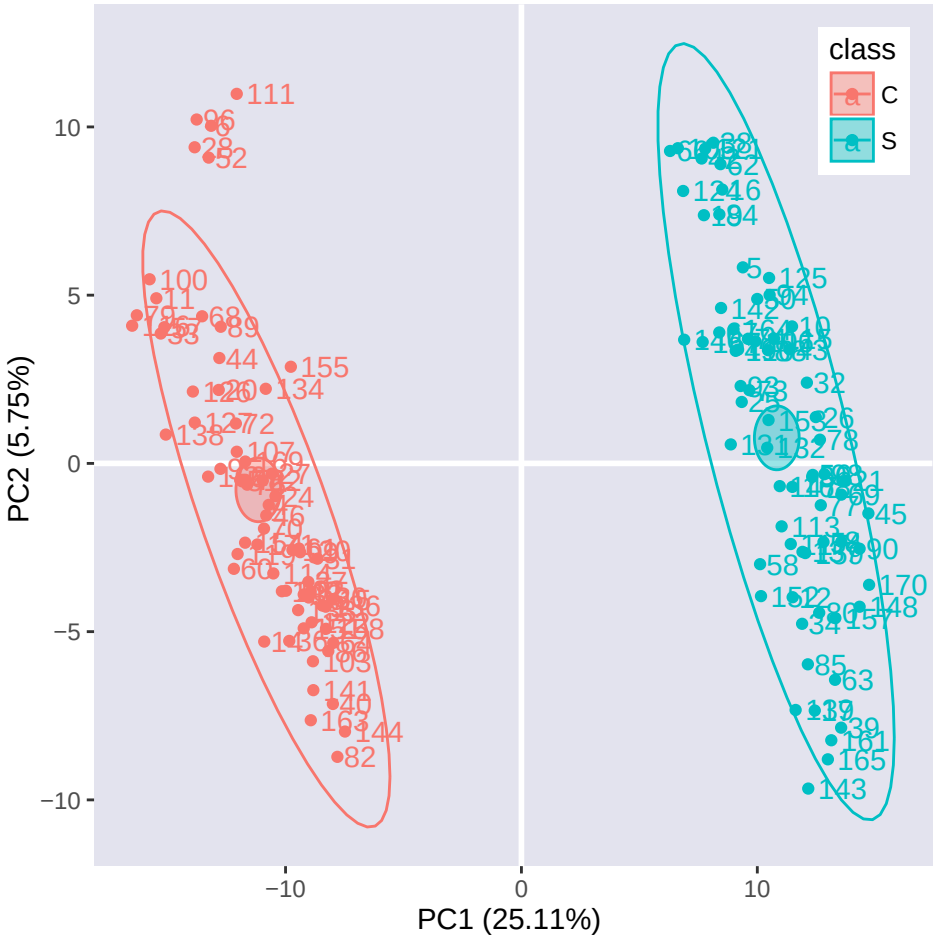

Figure 8: The score plot for PLS-DA.

4 Frequently asked questions

4.1 How to select the best number of components for PLS-DA?

The function `selectBestComponent` can be used to select the best number of components for PLS-DA. This function calculates the R2 and Q2 for each component.

```
para <- new("metaXpara")
pfile <- system.file("extdata/MTBLS79.txt",package = "metaX")
sfile <- system.file("extdata/MTBLS79_sampleList.txt",package = "metaX")
rawPeaks(para) <- read.delim(pfile,check.names = FALSE)
sampleListFile(para) <- sfile
para <- reSetPeaksData(para)
para <- missingValueImpute(para)
para <- metaX::normalize(para,method="pqn",valueID="value")
selectBestComponent(para,np=10,sample=c("S","C"),scale="pareto",valueID="value",k=5)
```

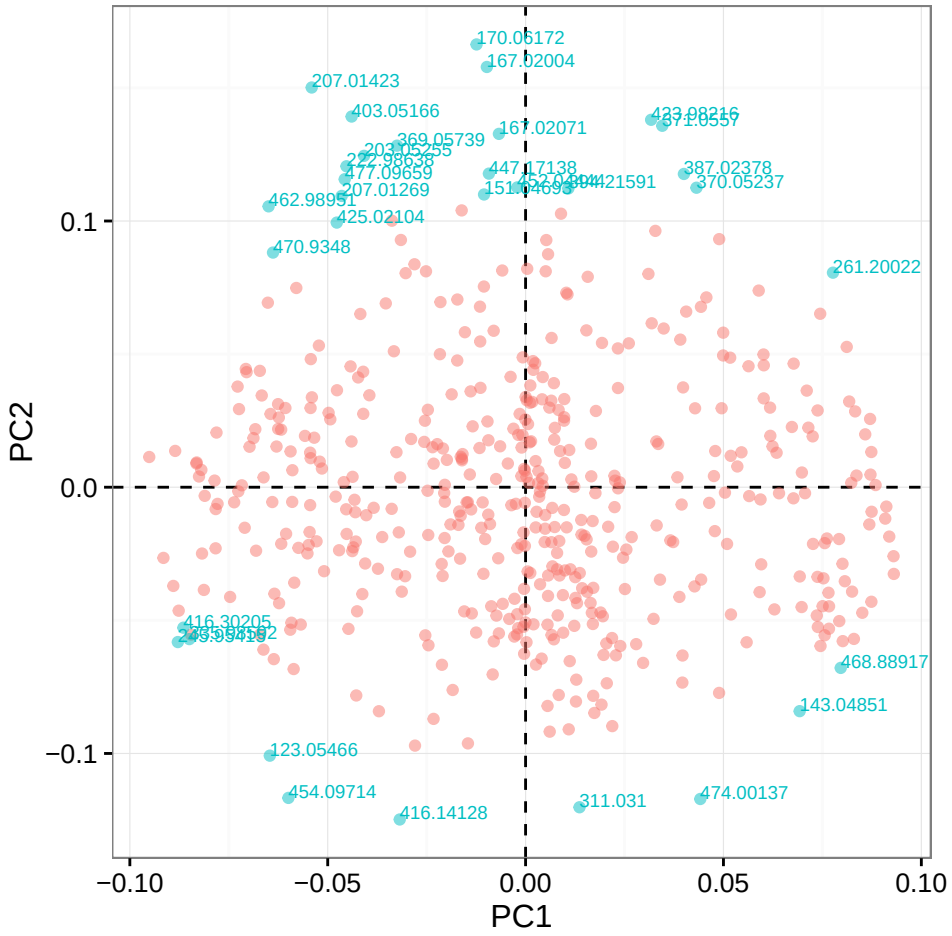

Figure 9: The loading plot for PLS-DA.

4.2 How to set the comparison groups?

We can use the following method to set the comparison groups:

```
ratioPairs(para) <- "KO:WT"
```

If multiple comparison groups must be set in a single analysis, the user can set the "para@ratioPairs" like "A:B;C:B;D:B", each comparison group is separated by semicolon.

```
ratioPairs(para) <- "A:B;C:B;D:B"
```

4.3 How to set the output file directory?

The user can set the output directory and the prefix of the output files as below:

```
## set the output parameters
outdir(para) <- "test"
```

```
prefix(para) <- "metaX"
```

Session information

All software and respective versions used to produce this document are listed below.

- R version 3.3.0 (2016-05-03), x86_64-pc-linux-gnu
- Locale: LC_CTYPE=en_US.UTF-8, LC_NUMERIC=C, LC_TIME=en_US.UTF-8, LC_COLLATE=C, LC_MONETARY=en_US.UTF-8, LC_MESSAGES=en_US.UTF-8, LC_PAPER=en_US.UTF-8, LC_NAME=C, LC_ADDRESS=C, LC_TELEPHONE=C, LC_MEASUREMENT=en_US.UTF-8, LC_IDENTIFICATION=C
- Base packages: base, datasets, grDevices, graphics, grid, methods, parallel, stats, utils
- Other packages: Biobase 2.32.0, BiocGenerics 0.18.0, ProtGenerics 1.4.0, Rcpp 0.12.5, SSPA 2.12.0, VennDiagram 1.6.17, caret 6.0-68, data.table 1.9.6, dplyr 0.4.3, futile.logger 1.4.1, ggplot2 2.1.0, lattice 0.20-33, limma 3.28.4, metaX 1.4.2, mzR 2.6.2, pROC 1.8, pls 2.5-0, plyr 1.8.3, qvalue 2.4.2, randomForest 4.6-12, reshape2 1.4.1, scatterplot3d 0.3-36, xcms 1.48.0
- Loaded via a namespace (and not attached): BBmisc 1.9, BiocInstaller 1.22.2, BiocStyle 2.0.2, CAMERA 1.28.0, DBI 0.4-1, DiffCorr 0.4.1, DiscriMiner 0.1-29, Formula 1.2-1, Hmisc 3.17-4, MASS 7.3-45, Matrix 1.2-6, MatrixModels 0.4-1, Nozzle.R1 1.1-1, R6 2.1.2, RBGL 1.48.0, RColorBrewer 1.1-2, RCurl 1.95-4.8, SparseM 1.7, acepack 1.3-3.3, affy 1.50.0, affyio 1.42.0, ape 3.4, assertthat 0.1, backports 1.0.2, bitops 1.0-6, boot 1.3-18, bootstrap 2015.2, car 2.1-2, checkmate 1.7.4, chron 2.3-47, class 7.3-14, cluster 2.0.4, codetools 0.2-14, colorspace 1.2-6, compiler 3.3.0, corpcor 1.6.8, digest 0.6.9, doParallel 1.0.10, e1071 1.6-7, ellipse 0.3-8, evaluate 0.9, faahKO 1.12.0, fdrtool 1.2.15, foreach 1.4.3, foreign 0.8-66, formatR 1.4, futile.options 1.0.0, graph 1.50.0, gridExtra 2.2.1, gtable 0.2.0, highr 0.6, igraph 1.0.1, impute 1.46.0, iterators 1.0.8, itertools 0.1-3, knitr 1.13, labeling 0.3, lambda.r 1.1.7, latticeExtra 0.6-28, lazyeval 0.1.10, lme4 1.1-12, magrittr 1.5, mgcv 1.8-12, minqa 1.2.4, missForest 1.4, mixOmics 5.2.0, multtest 2.28.0, munsell 0.4.3, nlme 3.1-128, nloptr 1.0.4, nnet 7.3-12, pbkrtest 0.4-6, pcaMethods 1.64.0, pheatmap 1.0.8, preprocessCore 1.34.0, quantreg 5.24, rgl 0.95.1441, rpart 4.1-10, scales 0.4.0, splines 3.3.0, stats4 3.3.0, stringi 1.0-1, stringr 1.0.0, survival 2.39-4, tidyr 0.4.1, tools 3.3.0, vsn 3.40.0, xtable 1.8-2, zlibbioc 1.18.0

References

[1] Alan Saghatelian, Sunia A Trauger, Elizabeth J Want, Edward G Hawkins, Gary Siuzdak, and Benjamin F Cravatt. Assignment of endogenous substrates to enzymes by global metabolite profiling. *Biochemistry*, 43(45):14332–14339, 2004.

[2] Jennifer A Kirwan, Ralf JM Weber, David I Broadhurst, and Mark R Viant. Direct infusion mass spectrometry metabolomics dataset: a benchmark for data processing and quality control. *Scientific data*, 1, 2014.

[3] Mikko Katajamaa, Jarkko Miettinen, and Matej Orešič. Mzmine: toolbox for processing and visualization of mass spectrometry based molecular profile data. *Bioinformatics*, 22(5):634–636, 2006.

[4] Carsten Kuhl, Ralf Tautenhahn, Christoph Bttcher, Tony R Larson, and Steffen Neumann. Camera: an integrated strategy for compound spectra extraction and annotation of liquid chromatography/mass spectrometry data sets. *Analytical chemistry*, 84(1):283–289, 2011.

[5] Warwick B Dunn, David Broadhurst, Paul Begley, Eva Zelena, Sue Francis-McIntyre, Nadine Anderson, Marie Brown, Joshau D Knowles, Antony Halsall, John N Haselden, et al. Procedures for large-scale metabolic profiling of serum and plasma using gas chromatography and liquid chromatography coupled to mass spectrometry. *Nature protocols*, 6(7):1060–1083, 2011.

[6] Frank Dieterle, Alfred Ross, Götz Schlotterbeck, and Hans Senn. Probabilistic quotient normalization as robust method to account for dilution of complex biological mixtures. application in 1h nmr metabonomics. *Analytical chemistry*, 78(13):4281–4290, 2006.

[7] David S Wishart, Timothy Jewison, An Chi Guo, Michael Wilson, Craig Knox, Yifeng Liu, Yannick Djoumbou, Rupasri Mandal, Farid Aziat, Edison Dong, et al. Hmdb 3.0the human metabolome database in 2013. *Nucleic acids research*, page gks1065, 2012.

[8] Atanas Kamburov, Rachel Cavill, Timothy MD Ebbels, Ralf Herwig, and Hector C Keun. Integrated pathway-level analysis of transcriptomics and metabolomics data with impala. *Bioinformatics*, 27(20):2917–2918, 2011.