An introduction to Baitmet package

Version 1.0.1

Xavier Domingo-Almenara (Maintainer)
xdomingo@scripps.edu

May 17, 2017

This vignette presents **Baitmet**, an R package for library driven compound profiling in full scan GC–MS. Baitmet allows an automated quantification of metabolites by targeting a mass spectral library into the chromatograms. **Baitmet** outputs a table with compound names, spectral matching score, general across-samples RI error, and the area of the compound for each sample. Baitmet is compatible with the use of internal standards, but it is also capable of automatically determine which is the compounds retention index without co-injecting internal standards. If you use the package **Baitmet** in your analysis and publications please cite:

Domingo-Almenara X, Brezmes J, Venturini G, Vivo-Truyols G, Perera A, Vinaixa M. Baitmet, a computational approach for GC–MS library-driven metabolite profiling. Metabolomics (**2017**) *Submitted*

This study is also referred for a more technical and detailed explanation about the **Baitmet** methods.

Installation: **Baitmet** can be installed from any CRAN repository, by:

```
# Install
install.packages("baitmet")
# Load
library(baitmet)
```

Support: Any enquiries, bug reports or suggestions are welcome and they should be addressed to xdomingo@scripps.edu.

Contents

1 Introduction

Baitmet is an R package that allows a high-throughput search of an entire mass spectral (MS) library into full-scan gas chromatography – mas spectrometry (GC–MS) data, using the library as a bait, to quantify metabolites (met) and thus performing a library-driven compound profiling. **Baitmet** can quantify compounds with (i) multivariate methods and without any prior information about the selective masses or (ii) by the integration of a previously defined selective mass for each compound. Also, internal standards (IS) for RT standardization are not needed in each sample, instead, only a single separated analysis of the IS is needed. **Baitmet** automatically determines which is the instrumental retention time variation of each samples with respect to a fixed retention index/retention time curve. **Baitmet** is however, also compatible with the use of internal standards, using them to compute the RI. **Baitmet** outputs a table with compounds name, spectral matching score, RI error, and the peak intensity or integrated area - relative concentration - of the compound in each sample.

2 Importing and customizing mass spectral libraries: using the Golm Metabolome Database

Baitmet comes with the free and downloadable version of the MassBank repository containing a set of ∼500 EI GC-TOF mass spectra. To use it simply execute:

```
data(mslib)
```

Users might import their own mass spectral libraries. **Baitmet** is recommended to be used with the Golm Metabolome Database (GMD). To use the GMD, first, we have to download it from its webpage[1], by downloading the file ”GMD_20111121_VAR5_ALK_MSP.txt” or ”GMD_20111121_MDN35_ALK_MSP.txt”, depending on which type of chromatographic columns (VAR5 or MDN35) are we using. Then, we can load the library with the function `importGMD()`:

```
g.info <- "
GOLM Metabolome Database
------------------------
Kopka, J., Schauer, N., Krueger, S., Birkemeyer, C., Usadel, B., Bergmuller, E., Dor-
mann, P., Weckwerth, W., Gibon, Y., Stitt, M., Willmitzer, L., Fernie, A.R. and Stein-
hauser, D. (2005) GMD.CSB.DB: the Golm Metabolome Database, Bioinformatics, 21, 1635-
1638."
golm.database <- importGMD(filename="GMD_20111121_VAR5_ALK_MSP.txt", DB.name=
"GMD", DB.version="GMD_20111121", DB.info= g.info, type="VAR5.ALK")
# The library in R format can now be stored for a posterior faster loading
save(golm.database, file= "golmdatabase.rda")
```

Even if using FAMEs, we can download and use the ALK version of the GMD. The ALK version is recommended as it contains more RI than the FAME version. Baitmet operates with relative retention indexes, then, both FAME and ALK RI might be used indistinctly. The critical step is to download and use the correct type of chromatographic column (VAR5 or MDN35).

We can now select only a subset of the library with the function `subSetLib()`:

```
# Create a object id.list containing the identification list
load("golmdatabase.rda")
# Select those with KEGG number:
```

[1]http://gmd.mpimp-golm.mpg.de/download/

```
kegg.ind <- which(lapply(golm.database@database, function(x) x$KEGG)!="")
golm.kegg <- subSetLib(database = golm.database, indexes = kegg.ind)
# We can also add the FAME into the library by:
db.names <- unlist(lapply(golm.database@database, function(x) x$Name))
fame.ind <- grep("FAME", db.names, ignore.case = TRUE)
golm.kegg.fames <- subSetLib(database = golm.database, indexes = unique(c(kegg.ind,
fame.ind)))
```

Finally, we can modify and customize the database:

```
# We can find a certain compound by:
findComp(name = "phenol", id.database = golm.kegg.fames)
  DB.Id            Compound Name CAS     Formula
1    13            Phenol (1TMS)  NA
2   181 Phenol, 2-amino- (2TMS)  NA C12H23NOSi2
3   263 Phenol, 2-amino- (3TMS)  NA C15H31NOSi3

# and access to its information (DB.Id=13):
golm.kegg.fames@database[[13]]$Name
"Phenol (1TMS)"

# or modify (for example) its selective masses:
golm.kegg.fames@database[[13]]$SelMZ <- c(120,154)

# To see all the slots available, execute:
str(golm.kegg.fames@database[[13]],1)
```

Baitmet uses by default in all its functions the library stored in the *mslib*. If we want to use Baitmet functions with another library, we have two options: i) to set the new library (in this case *golm.kegg.fames*) in all the functions that requiere it, for example:

```
findComp(name = "phenol", id.database = golm.kegg.fames)
DB.Id            Compound Name CAS     Formula
1    13            Phenol (1TMS)  NA
2   181 Phenol, 2-amino- (2TMS)  NA C12H23NOSi2
3   263 Phenol, 2-amino- (3TMS)  NA C15H31NOSi3
```

or ii) to substitute the default Baitmet database object *mslib*, for our custom database, by the following code:

```
mslib <- golm.kegg.fames
# and then:
findComp(name = "phenol")
DB.Id            Compound Name CAS     Formula
1    13            Phenol (1TMS)  NA
2   181 Phenol, 2-amino- (2TMS)  NA C12H23NOSi2
3   263 Phenol, 2-amino- (3TMS)  NA C15H31NOSi3
```

This allows executing all the functions without the need of always setting the library parameter. For the rest of the tutorial though, we will always declare the library, despite being not necessary.

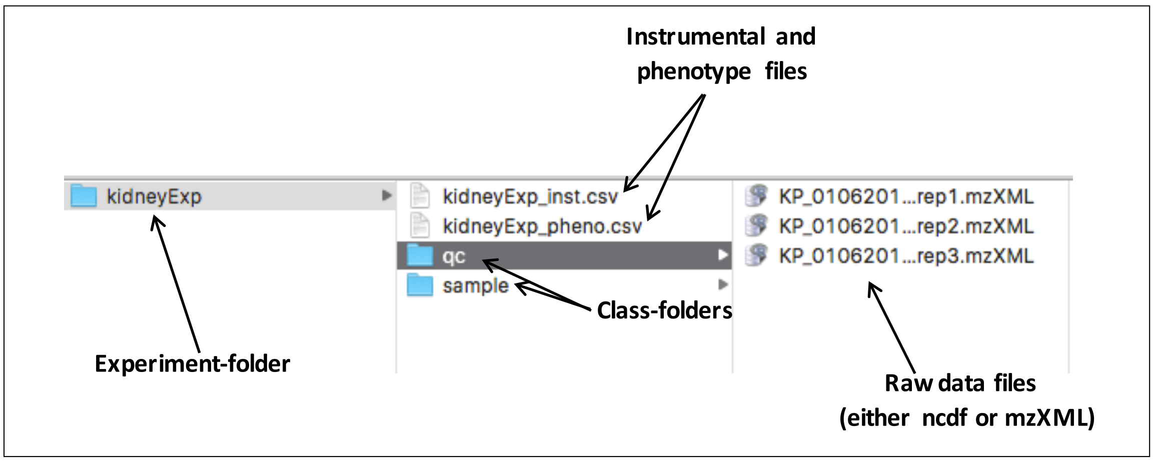

Figure 1: Distribution of the raw data files and the class and experiment folders for the given example.

3 GC–MS Data Processing with Baitmet: a tutorial

In this section we show the processing of pure standards samples analyzed through GC–MS. This tutorial shows how to deconvolve, align and identify the compounds contained in these four samples.

All the listed commands (script) to reproduce the following demo can by found by executing:

```
help(package="baitmet")
```

and then click on *User guides, package vignettes and other documentation* and on *source* from the ’Baitmet Manual’.

3.1 Compound profiling: data processing

3.1.1 Setting up the experiment files

The experiment has to been organized as follows: all the samples related to each class have to be stored in the same folder (one folder = one class), and all the class-folders in one folder, which is the so-called *experiment folder*. **Baitmet** also accepts only one class; in that case, only one class-folder has to be created inside an experiment-folder. As an illustrative example, Figure 1 shows the disposition of the files in an example case where we aim to process some samples divided into two classes: *qc* and *sample*. In this example, the experiment folder is the ’kidneyExp’ folder, which contains two class-folders called ’qc’ and ’sample’.

For the rest of this tutorial, we will use instead a set of four pure standards chromatograms, which form a single class called *Std* in an *experiment folder* called *DEMO*. The following code will automatically create a folder ’DEMO’ and a subfolder ’Std’ in the current R working directory and download the raw data from a GitHub repository:

```
# Create the directory:
dir.create("DEMO/Std", recursive=T)
# Download raw data
download.file("https://github.com/xdomingoal/baitmet/raw/master/
131114aMRsa04_1.rdata", "DEMO/Std/131114aMRsa04_1.rdata")
download.file("https://github.com/xdomingoal/baitmet/raw/master/
131114aMRsa06_1.rdata", "DEMO/Std/131114aMRsa06_1.rdata")
download.file("https://github.com/xdomingoal/baitmet/raw/master/
131114aMRsa11_1.rdata", "DEMO/Std/131114aMRsa11_1.rdata")
download.file("shttps://github.com/xdomingoal/baitmet/raw/master/
131114aMRsa12_1.rdata", "DEMO/Std/131114aMRsa12_1.rdata")
```

To create a new experiment we have to create first a .csv type file containing the name of the raw data files to process. The raw data files have to be in the same directory as the instrumental file. **Baitmet** also admits a phenotypic table which contains the classes of the samples. The instrumental data file is always needed but the phenotype file is optional. The instrumental table can have as many columns as desired, but it has to contain at least two columns named 'sampleID' and 'filename'. The same is applicable to the phenotypic table, in this case the two necessary columns are 'sampleID' and 'class'. Please note that capital letters of the column names must be respected and that 'sampleID' is the column that relates the instrumental and phenotypic tables. These files can also be created automatically by executing the following code:

```
createdt("DEMO")
```

where DEMO is the *experiment path*, corresponding the the folder 'DEMO' in the current working directory. The *experiment path* is the path where the experiment-folder is, and DEMO is the experiment-folder. Two things have to be considered at this step: .csv files are different when created by American and European computers, so errors might raise due to that fact. Also, the folder containing the samples (in this case, the folder 'DEMO', must contain only folders. If the folder 'DEMO' contains files (for example, an already created .csv file), **Baitmet** will prompt an error.

3.1.2 Data processing

We load the new experiment to the R workspace using the function 'newExp', where we introduce the path of the .csv file containing the instrumental data and the phenotypic data, along with a description of the experiment. We name the new experiment as 'ex'. With `metaData` and `phenoData` we can retrieve the instrumental data and the experiment classes:

```
ex <- newExp(instrumental="DEMO/DEMO_inst.csv", phenotype="DEMO/DEMO_pheno.csv")
# Accessing metadata:
metaData(ex)
         sampleID                  filename       date     time
1 131114aMRsa04_1 Std/131114aMRsa04_1.rdata 2016-07-11 13:43:40
2 131114aMRsa06_1 Std/131114aMRsa06_1.rdata 2016-07-11 13:40:48
3 131114aMRsa11_1 Std/131114aMRsa11_1.rdata 2016-07-11 13:40:56
4 131114aMRsa12_1 Std/131114aMRsa12_1.rdata 2016-07-11 13:41:06
 # Accessing phenodata:
phenoData(ex)
Experiment containing 4 samples in 1 different type of classes named:  Std.

sampleID          class
1 131114aMRsa04_1   Std
2 131114aMRsa06_1   Std
3 131114aMRsa11_1   Std
4 131114aMRsa12_1   Std
```

Next, we create the chromatographic method by the function `setChrmMethod`, where we input the retention times and indexes of our references. In the given example dataset, 13 FAME are co-injected with the samples. The RI of those FAME are known, and we have manually measured their retention times. Then, we can setup the chromatographic method for this example by:

```
Chrm.STD <- setChrmMethod(rt=c(4.415, 5.145, 5.836, 7.11, 8.26, 9.30, 10.263,
11.14, 11.94, 12.691, 13.455, 14.485, 15.968), ri=c(1120.62, 1225.27, 1326.95,
1526.31, 1729.57, 1928.52, 2131.7, 2342.06, 2548.56, 2739.86, 2941.17, 3140.52,
3343.88), name="Chrm STD")
```

Next, compounds are detected with `decBaitmet()` function [2]. This function needs a 'Deconvolution parameters' object, that can be created with `setBaitPar()` function, containing the parameters of the algorithm as shown as follows:

```
ext.par <- setBaitPar(ri.error = 0.05, min.peak.width = 1, min.peak.height = 1000,
noise.threshold = 100, avoid.processing.mz = c(1:69,73:75,147:149))
ex <- decBaitmet(Experiment = ex, BaitParameters = ext.par, ms.library = mslib,
chrom.method = Chrm.STD)
Searching metabolites in Std/131114aMRsa04_1.rdata ...  Processing 1 / 4
|==================================================| 100%
Searching metabolites in Std/131114aMRsa06_1.rdata ...  Processing 2 / 4
|==================================================| 100%
Searching metabolites in Std/131114aMRsa11_1.rdata ...  Processing 3 / 4
|==================================================| 100%
Searching metabolites in Std/131114aMRsa12_1.rdata ...  Processing 4 / 4
|==================================================| 100%
Grouping metabolites across samples...

Computing spectral match scores...

Done!
```

The most important parameter is the retention index error (ri.error) in which compounds are going to be searched. A value of 0.05 corresponds to a 5 percent. Also, the min.peak.width plays an important role, as an incorrect value will cause an incorrect deconvolution of the compounds. typically, this value should be less than half of the mean compound width. For this experiment, the average peak width is between 2 and 2.5 seconds, so an optional value is between $1-2$ seconds peak width. The lower this parameter is set to, the more narrow the Region of Interest, which might lead to an incorrect quantification of some metabolites with wider chromatographic peaks. If the parameter value is increased, the regions of interest are wider and so are the spectral variance captured by the algorithm, which might lead to an incorrect deconvolution of the compound.

Another important parameter is the `avoid.processing.mz`, that defines the masses to exclude. The m/z 73, 74, 75, 147, 148, 149 are recommended to be excluded in the processing and subsequent steps, since these are ubiquitous mass fragments typically generated from compounds carrying a trimethylsilyl (TMS) moiety. If the samples have ben derivatized, including these masses will only hamper the deconvolution process; this is because an important number of compounds will share these masses leading to a poorer selectivity between compounds. Also, in GC–MS-based metabolomics samples, we recommend excluding all masses from 1 to 69 Da, for the same reasons. Those masses are generated from compounds with groups that are very common for a large number of compounds in metabolomics, leading to a poorer selectivity between compounds. Although those masses are also the most intense m/z in the compounds spectra, Baitmet automatically sets the used library's masses to zero, so it does not affect spectral matching and identification.

A detailed explanation of the parameters can be accessed by executing: `?setBaitPar`

[2] If when executing decBaitmet() function an error of the type: "Error Reading from Connection" appears, that means that the demo data used for this example has not been downloaded correctly. Please, download it again

☞ **Masses to exclude:** It is important to know the range of the spectral library used, and set the `avoid.processing.mz` parameter accordingly. For instance, when using the Golm Metabolome Database, these aforementioned masses (from 1 to 69, and 73, 74, 75, 147, 148, 149) are not included in the library, so it is important to exclude all these masses. Otherwise, Baitmet will fail in correctly 'target' these spectra.

Now, we can access to the identification list, alignment list and final list respectively by idList() (identification results), alignList() (relative quantification results) and dataList() (both identification and quantification results). For example:

```
head(idList(object = ex, id.database = mslib ))
AlignID tmean FoundIn                                   Name.1 MatchFactor.1
1      5 4.042       4        Acetoacetic acid (1MEOX) (1TMS) MP         77.28
2     10 4.043       4   Butanoic acid, 2-oxo- (1MEOX) (1TMS) MP         48.51
3      4 4.044       4        Acetoacetic acid (1MEOX) (1TMS) BP         90.96
4     11 4.047       4 Isovaleric acid, 2-oxo- (1MEOX) (1TMS) MP         95.63
5     23 4.047       4 Isovaleric acid, 2-oxo- (1MEOX) (1TMS) BP         79.84
6      1 4.069       3                      Glycolic acid (2TMS)         83.42
```

For more details on how display **Baitmet** results can be found in section 4 Results and visualization.

Data can be saved and loaded at any stage of the process by:

```
save(ex, file="DEMO_baitmet.rda")
load("DEMO_baitmet.rda")
```

3.2 Computing retention indexes

3.2.1 Withouth using internal standards

In large datasets, RI/RT curve calculated using the calibration file containing RI standards (representative of the chromatographic method), might suffer variation due to instrumental error. To evaluate these variations, and automatically determine the empirical compounds RI by using the compounds found in the samples as natural reference, use:

```
ex <- computeRI(Experiment = ex, ms.library = mslib)
```

Now, if we access to the identification list by `idList()` function, the table will be updated with a RI error, in %. For more details, please see the original **Baitmet** study.

3.2.2 Using internal standards

Alternatively, **Baitmet** is also compatible with the use of FAME or ALK, or any other compound which the user might consider as a reference. Let us first illustrate the steps to follow for an user that has co-injected FAME with the samples, as is the case of the given example. Once the samples are processed, users might access to the compounds list through `idList()` function. The user has to localize all the FAME in the list. To do it automatically, execute the following lines:

```
# Create a object id.list containing the identification list
id.list <- idList(object = ex, id.database = mslib)
# Select their names
id.names <- id.list$Name.1
# Search and select only those names containing the word FAME
fame.indexes <- grep("FAME", id.names, ignore.case = TRUE)
id.list[fame.indexes,]
```

```
AlignID tmean FoundIn                                   Name.1 MatchFactor.1
349      4.418       4      Octanoic acid methyl ester (FAME MIX)        99.42
350      5.153       4      Nonanoic acid methyl ester (FAME MIX)        98.97
351      5.850       4     Decanoic acid, methyl ester (FAME MIX)        98.87
352      7.121       4    Dodecanoic acid methyl ester (FAME MIX)        98.5
353      8.271       4 Tetradecanoic acid methyl ester (FAME MIX)        98.73
354      9.316       4  Hexadecanoic acid methyl ester (FAME MIX)        99.2
355     10.275       4  Octadecanoic acid methyl ester (FAME MIX)        98.79
356     11.152       4    Eicosanoic acid methyl ester (FAME MIX)        98.7
357     11.960       4    Docosanoic acid methyl ester (FAME MIX)        99.69
358     12.710       4 Tetracosanoic acid methyl ester (FAME MIX)        99.27
359     13.477       4  Hexacosanoic acid methyl ester (FAME MIX)        99.16
360     14.506       4  Octacosanoic acid methyl ester (FAME MIX)        98.76
361     15.990       4 Triacontanoic acid methyl ester (FAME MIX)        98.02
```

To use those FAME as internal reference, the user has to introduce their AlignID to the function `computeRI()`, therefore:

```
FAME.AlignID <- as.numeric(as.vector(id.list[fame.indexes, "AlignID"]))
ex <- computeRI(Experiment = ex, ms.library = mslib, IS.alignid = FAME.AlignID)
```

Now, if we access to the identification list by `idList()` function, the table will be updated with a RI error, in %. In fact, the user might select other compounds to be used as reference. If the user is confident of the identity of a set of metabolites (e.g., urea, glycerol, myo-inositol), the user might use a set of them to compute the RI, by introducing their AlignID into the `computeRI()` function. However, this should be use at own risk or if the user has a good background on chromatography, as those metabolites selected should elute along all the chromatogram (as FAME or ALK are), and their identity should be priorly confirmed.

3.3 Validating results: using selective masses for compound quantification

The user might confirm the quantitative results provided by **Baitmet** by the quantification of selective masses of some (or all) compounds. For example, we want to quantify some m/z from Benzoic acid (AlignID 38) and Fumaric acid (AlignID 61) [3]. The function `quantSM` will first see if those compounds have their most selective masses defined in the library selected, if not, the function will take the mzN (parameter in quantSM) most intense masses of the empirical spectra. By default, the function selects the 5 most intense masses of the compound (mzN=5).

```
ex <- quantSM(Experiment = ex, ms.library = mslib, AlignID = c(38,61))
```

and we can access to the results by:

```
mzList(Experiment = ex, by.area = TRUE)
AlignID  MZ    RT 131114aMRsa04_1 131114aMRsa06_1 131114aMRsa11_1 131114aMRsa12_1
1     38 122 5.349        61255151        56364467        51502905        54833166
2     38  76 5.347          365098          343833          313597          344531
3     38  91 5.349         4337608         4177261         3689513         3940886
4     38 105 5.349        60301812        55618941        50945494        54151536
5     38  77 5.349          980967          952414          860949          919944
```

[3]The AlignID is a dynamic variable that changes from one to another experiment. This means that if the user has reproduced this DEMO exactly with the same parameters, the AlignID should coincide, but the use of other parameters will lead to another AlignID number for each metabolite.

```r
plotProfile(Experiment = ex, AlignId = 61)
```

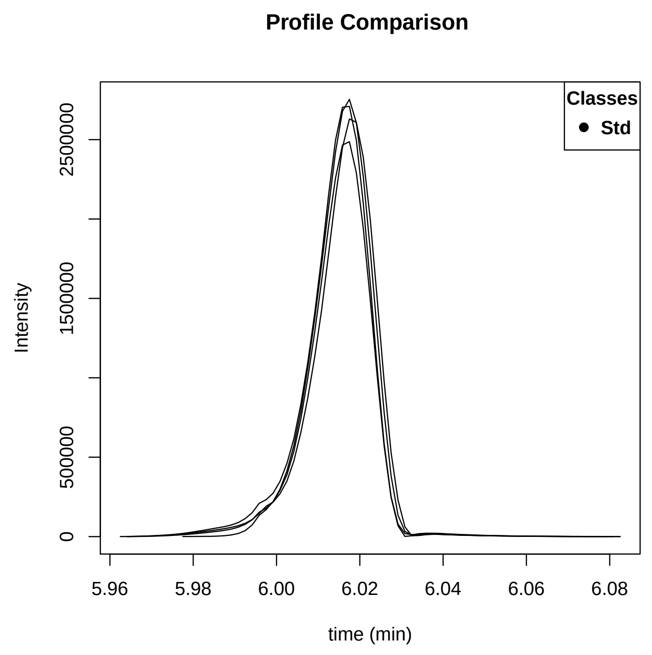

Figure 2: Image from plotProfile(ex,61).

```
6     61  245  6.018           7656873          7661698          6903457          7188981
7     61   83  6.018           9186852          8713769          8099637          8686076
8     61  143  6.018          29964360         27890251         26534381         28433705
9     61  133  6.018           6094245          5690635          5403206          5785183
10    61  246  6.018          15830972         15093192         13977989         15037326
```

We can visualize the extracted masses by the `plotMZ()` function, as explained in Section 4.

4 Results and visualization

We can access to the identification list, alignment list and final list by idList(), alignList() and dataList() respectively. From the idList(ex), we see that Fumaric acid is appearing at minute 6.01 with an AlignID number #61. Let us have a look to its profile with the function `plotProfile()`, which displays Figure 2.

Its spectra can be also be plotted and compared with the reference spectra using the function `plotSpectra()` which displays Figure 3:

The plotSpectra() function has a lot of possibilities for plotting, to know more access to its particular help by `?plotSpectra`. Alternatively, if we have quantified the selective masses with the `quantSM()` function, we can visually those extracted masses by `plotMZ()`, which displays Figure 4.

Users might export their spectra to MSP format or CEF format for comparison with NIST MS Search software (MSP), or to compare spectra with NIST through the MassHunter workstation (CEF). Users are referred to `exportMSP` and `exportCEF` functions help for more details.

```
plotSpectra(Experiment = ex, AlignId = 61, id.database = mslib)
```

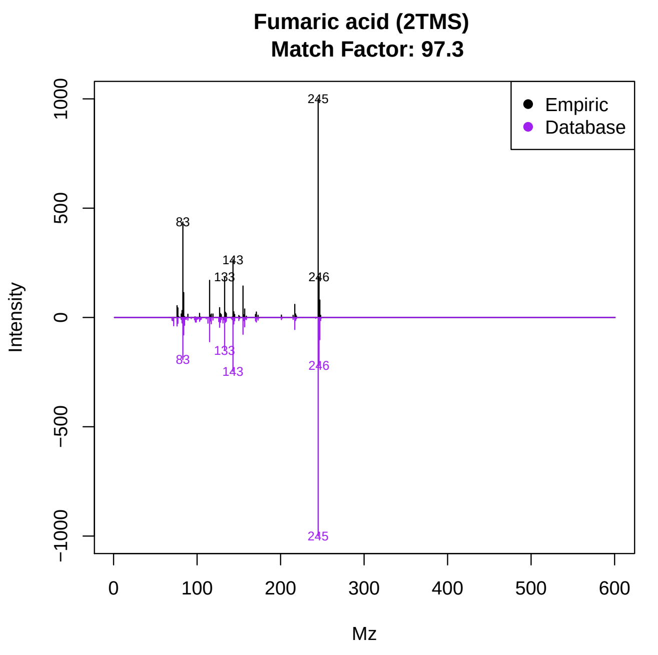

Figure 3: Image from plotSpectra(ex,61).

```
plotMZ(Experiment = ex, AlignId = 61)
```

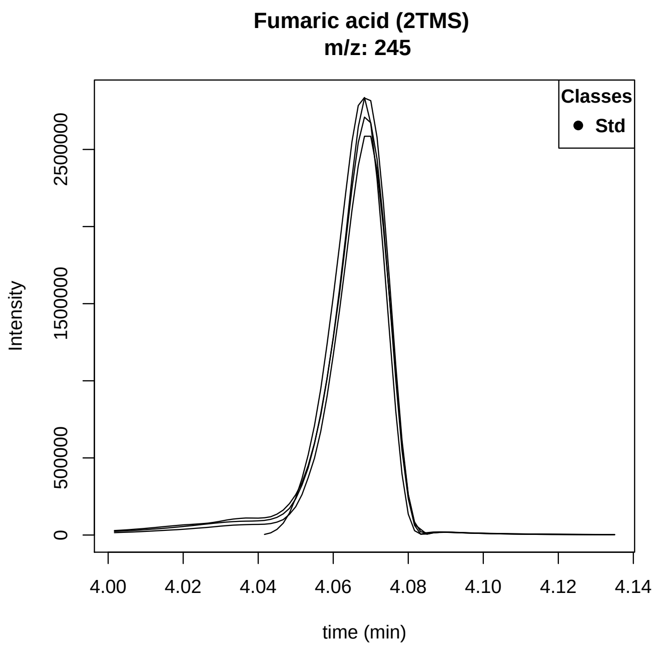

Figure 4: Image from plotMZ(ex,61).

5 Kwon troubleshootings

5.1 I have not used Proteowizard software to convert the raw data into .cdf or .mzXML

The use of Proteowizard is strongly recommended. Vendor software is usually not designed to convert their file formats into .cdf or .mzXML files, resulting in corrupted files or poor data quality (with the exepction of ChromaTOF software, which performance is good). Please, download and use Proteowizard (http://proteowizard.sourceforge.net).

5.2 I can not import the libraries from MSP files

MSP files have to be correctly formatted. For more details access to the `importMSP` help through `?importMSP`. To load the Golm Metabolome Database files please use `importGMD`

5.3 Other issues

In case of finding an undescribed issue or error, please contact the maintainer at xdomingo@scripps.edu.