An introduction to erah package
Version 1.1.0

Xavier Domingo–Almenara (Maintainer)

xavier.domingoa@eurecat.org

July 3, 2020

This vignette presents **eRah**, an R package with an integrated design that allows for an innovative deconvolution of GC–MS chromatograms using multivariate techniques based on blind source separation (BSS), alignment of spectra across samples, and automatic identification of metabolites by spectral library matching. eRah outputs a table with compound names, matching scores and the area of the compound for each sample. eRah is designed in an open-structure, where researchers can integrate different algorithms for each step of the pipeline, i.e., compound deconvolution, alignment, identification or statistical analysis. eRah has been tested with GC-TOF/MS and GC-qTOF/MS (using nominal mass) equipment, and is compatible with different spectral databases. Here, we integrate the downloadable version of the MassBank spectral library for a straightforward identification. If you use the package **eRah** in your analysis and publications please cite:

> [1] X. Domingo-Almenara, et al., eRah: a computational tool integrating spectral deconvolution and alignment with quantification and identification of metabolites in GC–MS-based metabolomics. *Analytical Chemistry* 88 **(2016)** 9821–9829. DOI: 10.1021/acs.analchem.6v02927

> [2] X. Domingo-Almenara, et al., Compound identification in gas chromatography/mass spectrometry-based metabolomics by blind source separation. *Journal of Chromatography A* 1409 **(2015)** 226–233. DOI: 10.1016/j.chroma.2015.07.044

Study [1] is also referred to a more technical and detailed explanation about the **eRah** methods.

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

```r
# Install
install.packages("erah")
# Load
library(erah)
```

Support: Any inquiries, bug reports or suggestions are welcome and they should be addressed to xavier.domingoa@eurecat.org.

Contents

1   Introduction

eRah automatically detects and deconvolves the spectra of the compounds appearing in GC–MS chromatograms. eRah processes the raw data files (netCDF or mzXML) of a complete metabolomics experiment in an automated manner.

After that, compounds are aligned by spectral similarity and retention time distance. eRah computes the Euclidean distance between retention time distance and spectral similarity for all compounds in the chromatograms, resulting in compounds appearing across the maximum number of samples and with the least retention time and spectral distance.

Also, an (optional) missing compound recovery step, can be applied to recover those compounds that are missing in some samples. Missing compounds appear as a result of an incorrect deconvolution or alignment - due to a low compound concentration in a sample - , or because it is not present in the sample. This forces the final data table with compound names and compounds area, to not have any missing (zero) values.

Finally, identification of the found metabolites is conducted. A mean spectrum from each group of aligned compounds is compared with a reference library. **eRah** includes a custom version of MassBank repository. Other libraries can be imported with eRah (e.g., Golm Metabolome Database), and eRah's deconvolved spectra can be exported for further comparison with NIST library.

2   GC–MS Data Processing with eRah: a tutorial

In this section, we show the processing of serum samples analyzed through GC–MS, and freely available from MetaboLights (accession number: MTBLS321). 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:

```
library(erah)
help(package = "erah")
```

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

2.1   Creating a new experiment

In the given example, we process only four serum chromatograms, divided into two classes: CONTROL and DISEASE. The experiment has to be 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 experiment folder (Figure 1). eRah also accepts only one class; in that case, only one class-folder has to be created inside an experiment-folder.

In this demo, the experiment folder is the 'PCOS' folder, which contains two class-folders called 'CONTROL' and 'DISEASE' (Figure 1). This experiment has two classes. Here, we used the following control samples: CON_BASA_567795.mzXML, and
CON_BASA_574488.mzXML, and the following disease samples: DIA_BASE_630974.mzXML and
DIA_BASE_635799.mzXML. The user can download the same, others, or all the experiment samples.

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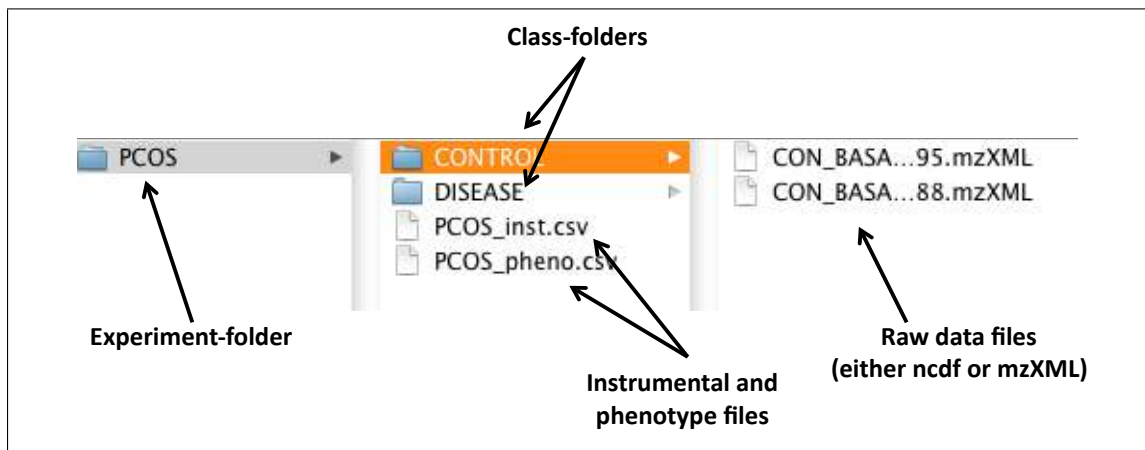
Figure 1: Distribution of the raw data files and the class and experiment folders for the given example.

eRah 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, execute the following command:

```
createdt("experiment_path/PCOS/")
```

where 'experiment path' is the path where the experiment-folder is, and PCOS 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 may raise due to that fact. Also, the folder containing the samples (in this case, the folder 'PCOS') must contain only folders. If the folder 'PCOS' contains files (for example, already created .csv files), eRah will prompt an error.

> ☞ Note that if you have an specific question about a function, you can always access to the help of the function with a question mark before the name of the function:
> ```
> ?createdt
> ```

Next, 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`, `phenoData` and `expClasses` we can retrieve the instrumental data and the experiment classes and the processing status:

```
ex <- newExp(instrumental = "PCOS/PCOS_inst.csv",
+ phenotype = "PCOS/PCOS_pheno.csv", pheno = "PCOS Experiment")
metaData(ex)

        sampleID                         filename        date     time
1 CON_BASA_567795 CONTROL/CON_BASA_567795.mzXML 2015-04-09 15:46:05
2 CON_BASA_574488 CONTROL/CON_BASA_574488.mzXML 2015-04-09 15:47:35
3 DIA_BASE_630974 DISEASE/DIA_BASE_630974.mzXML 2015-04-09 16:09:22
4 DIA_BASE_635799  DISEASE/DIA_BASE_635799.mzXML 2015-04-09 16:10:55

 phenoData(ex)
```

```
         sampleID    class
1 CON_BASA_567795 CONTROL
2 CON_BASA_574488 CONTROL
3 DIA_BASE_630974 DISEASE
4 DIA_BASE_635799 DISEASE
```

```
 expClasses(ex)
```

```
Experiment containing 4 samples in 2 different type of classes named:
 CONTROL, DISEASE.
```

```
         Sample ID Class Type Processing Status
1 CON_BASA_567795    CONTROL      Not processed
2 CON_BASA_574488    CONTROL      Not processed
3 DIA_BASE_630974    DISEASE      Not processed
4 DIA_BASE_635799    DISEASE      Not processed
```

2.2 Compound deconvolution

Compounds are deconvolved with `deconvolveComp` function. This function needs a 'Deconvolution parameters' object, that can be created with `setDecPar` function, containing the parameters of the algorithm as shown as follows:

```
ex.dec.par <- setDecPar(min.peak.width = 1,  +
avoid.processing.mz = c(35:69,73:75,147:149))
ex <- deconvolveComp(ex, ex.dec.par )
 Extracting factors from CONTROL/CON_BASA_567795.mzXML ... Processing 1 / 4
   |=================================================| 100%
 Extracting factors from CONTROL/CON_BASA_574488.mzXML ... Processing 2 / 4
   |=================================================| 100%
 Extracting factors from DISEASE/DIA_BASE_630974.mzXML ... Processing 3 / 4
   |=================================================| 100%
 Extracting factors from DISEASE/DIA_BASE_635799.mzXML ... Processing 4 / 4
   |=================================================| 100%
Compounds deconvolved
```

The peak width value (in seconds) is a critical parameter that conditions the efficiency of eRah, and also the masses to exclude have an important role in GC–MS-based metabolomics experiments.

> ☞ **Peak width parameter:** 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 we selected 1 second peak width. The lower this parameter is set to, the more sensibility to deconvolve co-eluted compounds, but it also may increase the number of false positive compounds. If it is set too low the algorithm will generate too false positives compounds, meaning that one single compound will be detected twice. If the parameter value is increased, the algorithm may fail in separate co-eluted compounds, leading to generate less false positives but losing the capacity of detection.

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

```
# Save
save(ex, file = "testPCOS.rda")
# Load
load("testPCOS.rda")
```

For low run-time performance in the deconvolution stage, please read the 'Kwon troubleshooting' section.

> ☞ **Masses to exclude:** masses 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 moiety. If the samples have been 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, eRah will automatically set the used library's masses to zero, so it does not affect spectral matching and identification.

2.3 Alignment

Alignment is executed with `alignComp` function. The parameters have to be also previously set.

```
ex.al.par <- setAlPar(min.spectra.cor = 0.90, max.time.dist = 3, +
mz.range = 70:600)
ex <- alignComp(ex, alParameters = ex.al.par)

  |=================================================| 100%
```

The parameters are min.spectra.cor, max.time.dist and mz.range. The min.spectra.cor (Minimum spectral correlation) value - from 0 (not similar) to 1 (very similar) - sets how similar two or more compounds have to be to be considered for alignment. We can be restrictive with this parameter, as if one compound is not detected in some samples, we can retrieve it later by the 'missing compound recovery' step. Also, we impose a maximum alignment distance of 3 seconds (max.time.dist). This value (in seconds) sets how far two or more compounds can be across samples to be considered the same compound, and thus being aligned. Mz.range is the range of masses that are considered when comparing spectra. We set that only the masses from 70 to 600 are taken into account, for the reasons commented above in the 'Masses to exclude' warning box (☞).

We can decide to execute the missing compound recovery step (and retrieve the compounds that have missing values - have not been found - in certain samples) or to identify the compounds without applying the `MissRecComp` function. In other words, the missing compound recovery step is optional. Here, we apply the missing recovery step to later identify the compounds.

> ☞ **Aligning large ammout of samples:** For experiments containing more than 100 (Windows) or 1000 (Mac or Linux) samples, alignment could lead to errors or show a poor run-time performance. In those cases alignment can be conducted by block segmentation. This issue can be overcome by using the parameter `blocks.size` in the function `alignComp`. For more details access to the `alignComp` help through `?alignComp`

2.4 Missing compound recovery

The missing compound recovery step only requires to indicate the number of minimum values for which a compound wants to be 're-searched' in the samples. If a compound appears in at least the same or more samples than the minimum samples value (min.samples), then, this compound is searched in the rest of the samples where its concentration has not been registered. To do so:

```
ex <- recMissComp(ex, min.samples = 3)
```

```
   |================================================| 100%
Updating alignment table...
Model fitted!
```

> ☞ **Missing compound recovery:** The `min.samples` parameter sets the number of
> samples from the total number of samples. Also, this parameter should be large, in
> alignment with the dimension of the number of samples in the experiment. If set too
> low, a higher number of false positives are expected. A recommended strategy is to
> first evaluate the average number of samples where the compounds appear, by execut-
> ing alignList or idList - after being identified - functions (explained in the following
> sections). **Warning:** if we have already identified the compounds, we always have
> to re-identify the compounds after executing the missing compound recovery step, by
> `identifyComp`, as explained in the following sections.

2.5   Identification

The final processing step is to identify the previously aligned compounds and assign them a pu-
tative name. **eRah** compares all the spectra found against a reference database. This package
includes a custom version of the MassBank MS library, which is selected as the default database
for all the functions. However, we strongly encourage the use of the Golm Metabolome Database
(GMD). GMD importation is described in following sections.

Identification can be executed by `identifyComp`, and accessed by `idList` as follows:

```
ex <- identifyComp(ex)

Constructing matrix database...
Comparing spectra...
Done!
 id.list <- idList(ex)
head(id.list[,1:4], n = 8)

  AlignID   tmean FoundIn                      Name.1 MatchFactor.1 DB.Id.1
1       1  5.0226       4               Ketovaline          68.57     274
2       2  5.1443       4  2'-Deoxyadenosine (3TMS)          61.39     372
3       3  5.3317       4    N-Carbamoyl-L-Aspartate          43.03     409
4       4  5.3567       4            2-Hydroxypyridine          99.61      25
5       5  5.4284       4    Pentachlorophenol (1TMS)          51.82      77
6       7  5.5401       4 Methylmalonic acid (2TMS)          20.06     250
7       8  5.6267       4          5-Aminovaleric acid          79.24     194
8      10  5.7076       4            L-(+)-Lactic acid          95.59     482
```

2.6   Retention Index (RI) error

eRah allows computing the retention index error. Both external and internal calibrations are
compatible. To perform an internal calibration, we need to AlignID values of the compounds to
be used as a reference. To do so, once metabolites are identified, use the output from `idList`
function to find the AlignID value of the compounds to be used as a reference. These compounds
can be FAME or any other metabolites which we know that their identity is correct. In other
words, naturally occuring metabolites that we know that their identity are correct, can be used
to perform an internal calibration curve. First of all, we have to define the Reference List. As an
example, we have 6 metabolites retrieved from `idList` that we known their identity:

```
reference.list <- list(AlignID = c(34,56,38,123,465))
```

We can manually define the RT and RI. This is the procedure to be used when performing an external calibration (The internal standards are not co-injected with the samples).

```
reference.list <- list(RT = c(5.43,6.78,7.86,9.12,11.15,13.45),  +
RI = c(1081.68, 1251.31, 1346.8, 1550.82, 1614.59, 1633.29))
```

To look for the RI of a specific metabolite we can make use of `findComp` and `compInfo` functions. For instance, if we want to know the RI of phenolic acid (in Golm Database):

```
findComp(name = "phenol", id.database = golm.database)
DB.Id          Compound Name CAS     Formula
1    23           Phenol (1TMS)  NA
2   364 Phenol, 2-amino- (2TMS)  NA C12H23NOSi2
3   263 Phenol, 2-amino- (3TMS)  NA C15H31NOSi3
```

Its DB.Id is 23, and now we retrieve the RI by:

```
compInfo(comp.id = 23, id.database = golm.database)
Name:  Phenol (1TMS)
Synonyms:  (Aminomethyl)cyclopropane , acide carbolique , acide phenique , ...
KEGG: C00146
RI (FAME & Var5):  0
RI (ALK & Var5):  1049.38
```

which is 1049.38. Finally, we can compute the RI error by:

```
ex <- computeRIerror(ex, id.database = golm.database, +
reference.list =  reference.list, ri.error.type =  "relative")
```

Keep in mind! The use of the Golm Metabolome Database is strongly encouraged when computing RI errors.

2.7   Results and visualization

Now, 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 urea is appearing at minute 8.13 with an AlignID number #41. Let us have a look to its profile with the function `plotProfile`:

> ☞ Execute `?alignList`, to access to the help of `alignList` function with a detailed explanation of each column in an align list.

```
plotProfile(ex, 41)
```

which displays Figure 2.7. Its spectra can be also be plotted and compared with the reference spectra using the function `plotSprectra`, which displays Figure 3:

```
plotSpectra(ex, 41)
```

The `plotSpectra` function has a lot of possibilities for plotting, to know more access to its particular help by executing `?plotSpectra`. For example, eRah allows a rapid assessment for visualizing the second hit returned in the case of compound align ID #41 (Urea). To do so:
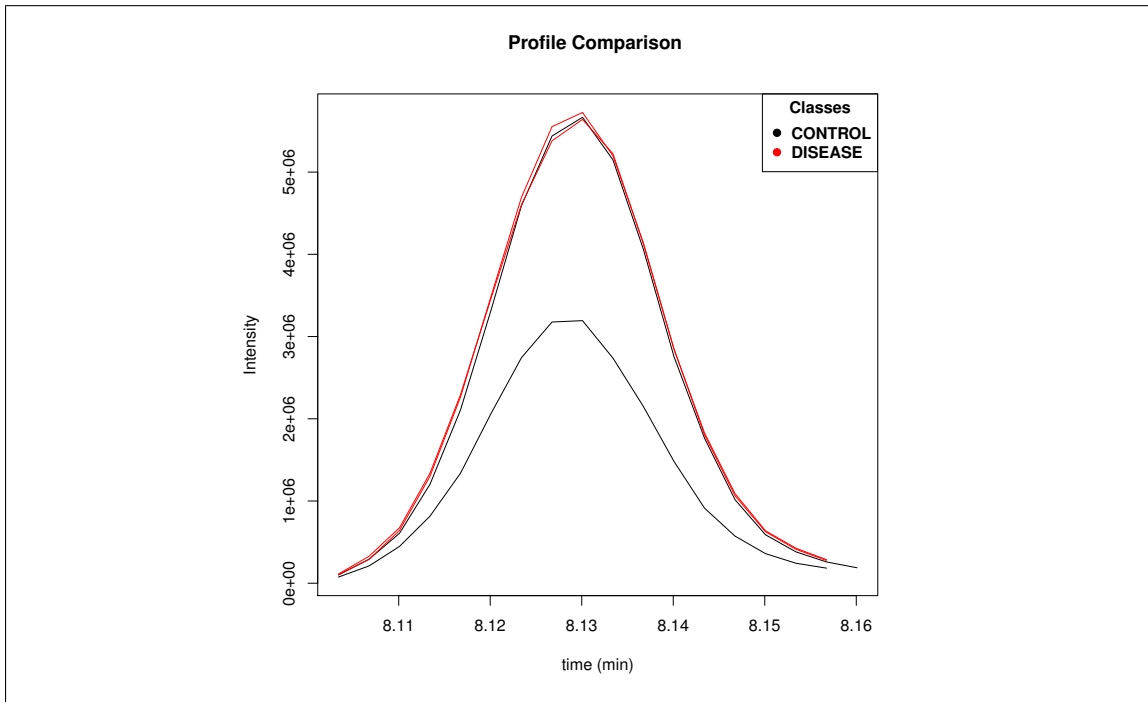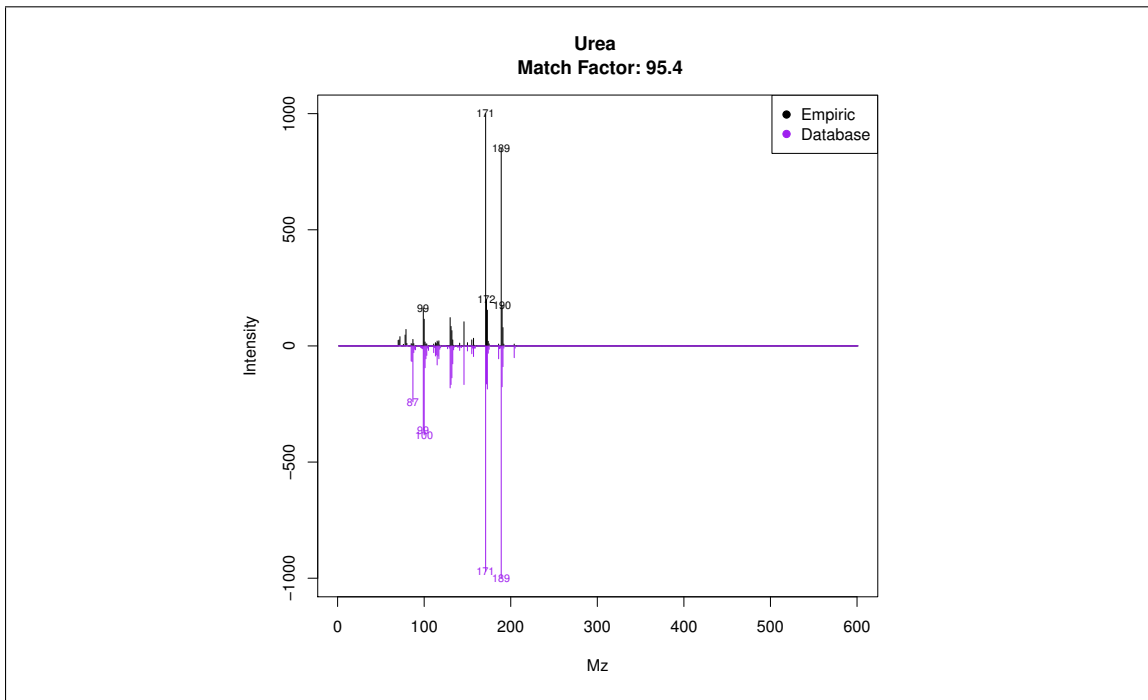
Figure 2: Image from plotProfile(ex,41).

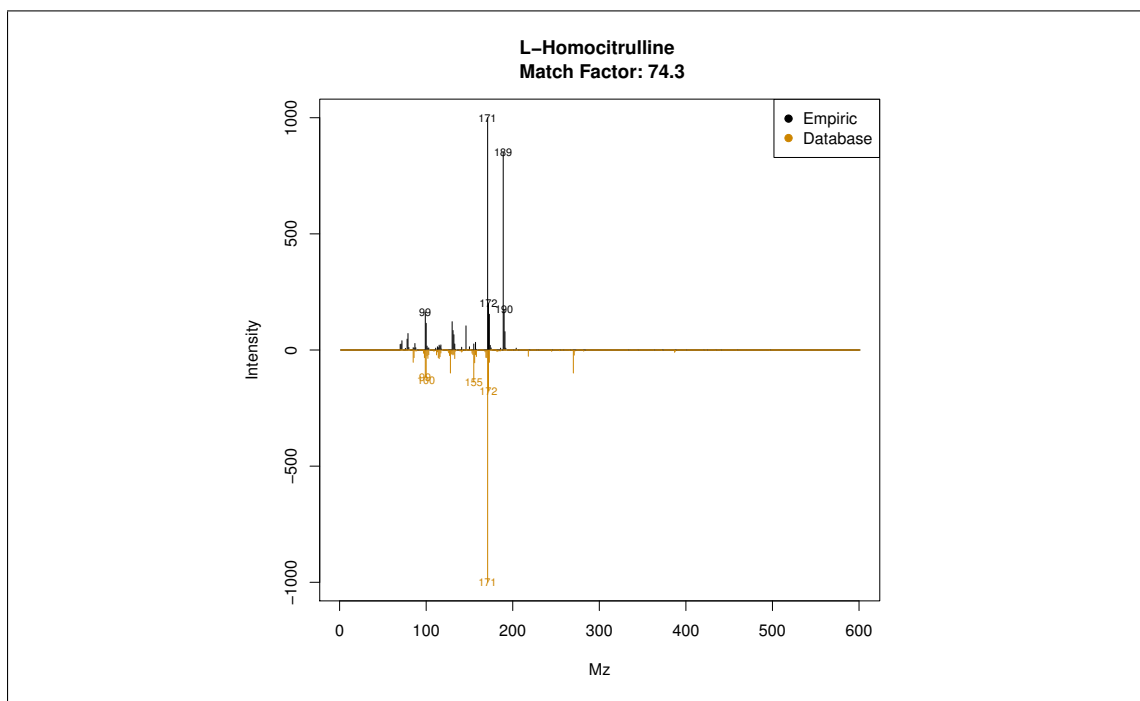**Figure 3:** Image from plotSpectra(ex,41).

Figure 4: Image from plotSpectra(ex,41, 2, draw.color='orange3').

```
plotSpectra(ex, 41, draw.color = "orange")
```

Figure 4 is a comparison of the empirical spectrum found, with the second most similar metabolite from the database (in this case Homocitrulline). From the figure, it is clear that eRah returned the first hit correctly, as this spectrum is more similar to urea than to homocitrulline.

3 Importing and customizing mass spectral libraries

3.1 Using the Golm Metabolome Database

Users may import their own mass spectral libraries. We strongly recommend using the Golm Metabolome Database (GMD) with eRah. 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. If you are not interested in using any retention index information, then both files can be used indistinctly. Then, we can load the library with the function `importMSP()`:

```
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=
```

---

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

```
"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")
```

We can substitute the default eRah database object `mslib`, for our custom database, by the following code:

```
load("golmdatabase.rda")
mslib <- golm.database
```

This allows executing all the functions without the need of always setting the library parameter. If we do not replace the `mslib` object as shown before, we have to use the new library (in this case `golm.database`) in all the functions, for example:

```
findComp(name = "phenol", id.database = golm.database)
DB.Id          Compound Name CAS      Formula
1    23            Phenol (1TMS)  NA
2   364 Phenol, 2-amino- (2TMS)  NA C12H23NOSi2
3   537 Phenol, 2-amino- (3TMS)  NA C15H31NOSi3
```

3.2 Using in-house libraries

Other MSP-formatted libraries can be also imported. The procedure is the same as for the GMD database, with the only exception is that the function is `importMSP` instead of `importGMD`. Access to specific `importMSP` help in R (`?importMSP`) for details on database MSP input format.

4 Exporting spectra: comparison with NIST

Users may export their results 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 `export2MSP` and `export2CEF` functions help for more details.

Of note, the `export2MSP` function has two algorithms implemented (see alg.version parameter). By default, alg.version is set to 1. If it not works with NIST MS search, try setting alg.version to 2.

5 Final Summary

To complement the given tutorial, the user may access to the particular help for each function, as shown before. Also, and for more details, please read the original article.

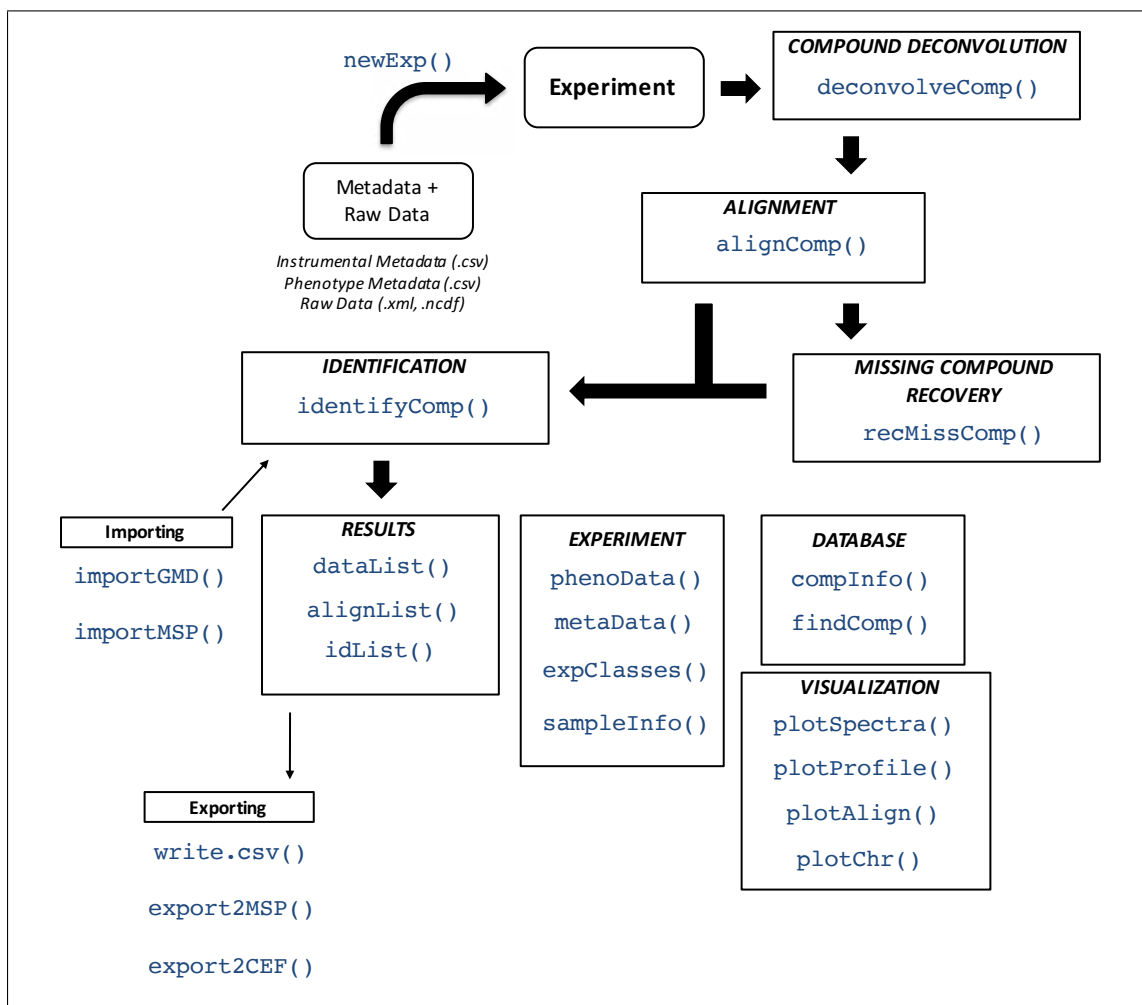Here, we show a figure (Figure 5) with all the available functions.

Figure 5: eRah summary of functions.

6 Kwon troubleshootings

6.1 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 exception of ChromaTOF software, which performance is good). Please, download and use Proteowizard (http://proteowizard.sourceforge.net).

6.2 eRah run-time performance is low on the deconvolution or alignment

eRah might experience low run-time performance in the deconvolution step if the chromatograms are oversampled (wider peaks with high scans per second values). If for example, the chromatogram has been acquired at 5 scans per second and peaks are 20 seconds or wider, please set the parameter `down.sample=TRUE` when using `deconvolveComp`. This is due that in those cases chromatograms are over-sampled. By having wider peaks and acquiring at 5 scans/s, a 30 s peak will be defined by more than 150 scans (points). This is similar to have 1 s peak width and scan at 150 scans/s. eRah uses multivariate methods which run-time performance depend on the amount of data to

be analyzed, and since each compound has a lot of data (150 scans or more), eRah run-time performance is expected to be low. A normal scans per second acquisition rate should that that is able to define the minimum peak width with 5-10 points (scans). Therefore, the way to optimize the deconvolution in these cases is the use of the `down.sample=TRUE` option.

In the same way, the alignment might experience errors or poor run-time performance for experiments containing more than 100 (Windows) or 1000 (Mac or Linux) samples. In these cases, alignment can be conducted by block segmentation. This issue can be overcome by using the parameter `blocks.size` in the function `alignComp`. For more details access to the `alignComp` help through `?alignComp`.

6.3   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`

6.4   Other issues

In case of finding an undescribed issue or error, please contact the maintainer at xavier.domingoa@eurecat.org.