Introduction to the **eRah** Package

Xavier Domingo-Almenara (Maintainer)
Metabolomics Platform.
Departament d'Enginyeria Electronica, Electrica i Automatica (DEEEA),
Universitat Rovira i Virgili. Tarragona, Catalonia, Spain.

1 Introduction

eRah is an R package with an integrated design that allows for an innovative deconvolution of the 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 an straightforward identification.

2 Methods

eRah automatically detects and deconvolves the spectra of the compounds appearing in the 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. A custom clustering algorithm groups the combination that minimize the euclidean distance of the spectra correlation distance and retention time distance between compounds of the different samples.

Also, an (optional) step, called *missing compound recovery* is applied to recover those compounds that are missing in some of the samples. Missing compounds appear as a result of an incorrect deconvolution or alignment - because the compound is in very low concentrations - , 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 spectra from each group of aligned compounds is compared with a reference library. **eRah** includes a custom version of MassBank repository.

3 Installation and use of eRah.

eRah can be installed from the CRAN repository, by:

```
> install.packages('erah')
```

or by downloading from the metabolomics platform webpage:
www.metabolomicsplatform.com/applications, and installing it manually. The webpage lists the updates, fixed bugs, and other issues.

4 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'.

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

Here in the 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. **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:

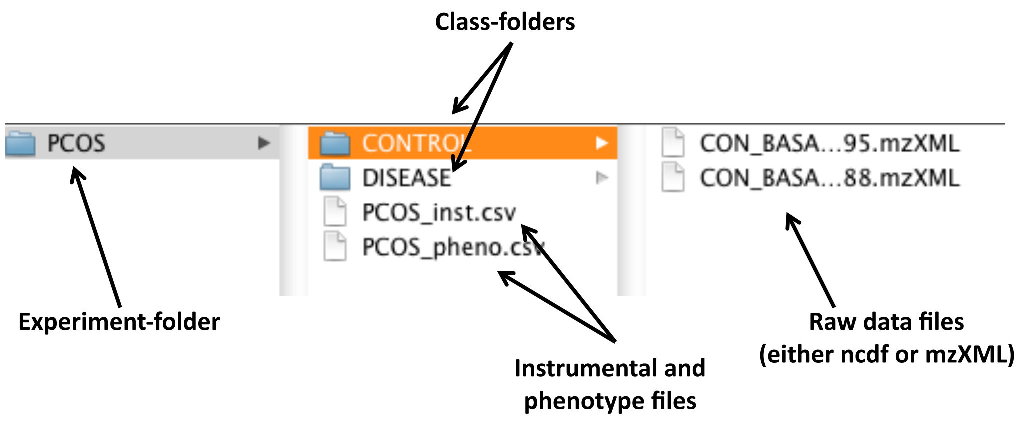

Figure 1: Distribution of the raw data files and the class and experiment folders for the given example.

```
> 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
```

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 processing status:

```
> library(erah)
>
> ex <- newExp(instrumental='PCOS/PCOS_inst.csv',
+ phenotype='PCOS/PCOS_pheno.csv', info='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
```

4.1 Compound deconvolution

The compounds in data 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=0.7, min.peak.height=5000,
+ noise.threshold=500, avoid.processing.mz=c(35:69,73:75,147:149),
+ analysis.time=c(5,15))
> 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 most important are the peak width, the masses to exclude and the analysis time. The minimum peak width (in seconds), is a critical parameter that conditions the efficiency of eRah. Typically, this should be the 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 is set too low the algorithm will generate too false positives compounds, which this usually means that one single compound will detected twice. If the parameter value is increased, the algorithm may fail in separate co-eluted compounds, leading to generate less false positives but loosing capacity of detection.

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

```
> save(ex, file='testPCOS.RData')
> load('testPCOS.RData')
```

These files will be stored at the R working directory.

4.2 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 Minimum spectral correlation value. From 0 (non similar) to 1 (very similar). This value sets how similar two or more compounds have be to be considered for alignment between them. 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 disalignment distance of 3 seconds (max.time.dist). This value (in seconds) sets how far two or more compounds can be to be considered for alignment between them. Mz.range is the range of masses that is considered when comparing spectra. We set that only the masses from 70 to 600 are taken into account.
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 also 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.

4.3 Missing compound recovery

To apply it, we just have to select 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!
```

4.4 Identification

The final processing step is to identify the previously aligned compounds and assign them a putative 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 default database for all the functions. 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
```

4.5 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()':

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

which displays Figure 4.5.
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:

```
> ?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:

```
> plotSpectra(ex,41, 2, draw.color='orange3')
```

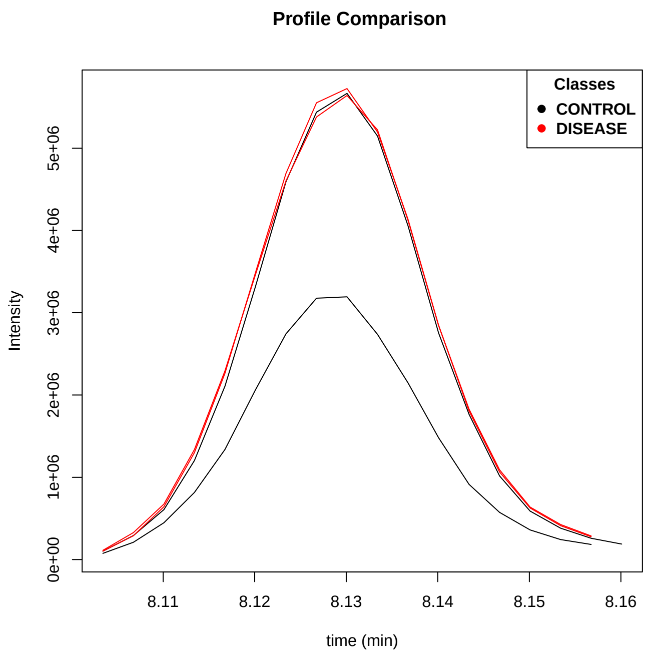

Figure 2: Image from plotProfile(ex,41).

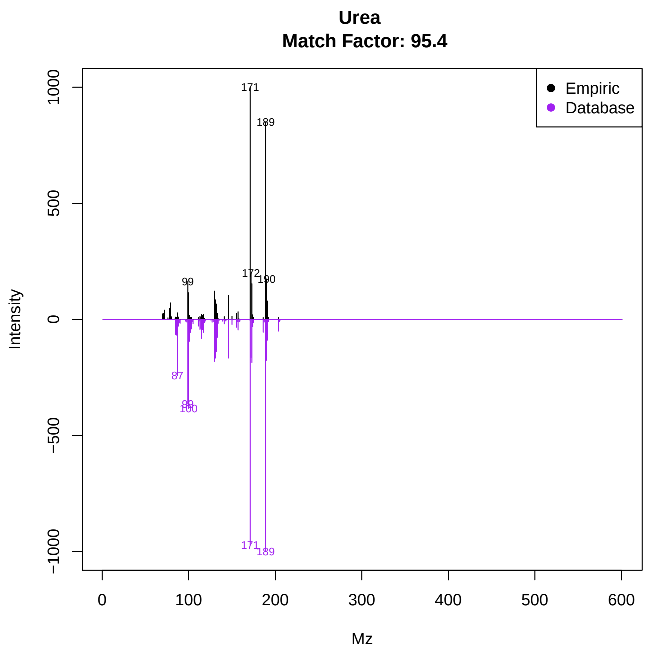

Figure 3: Image from plotSpectra(ex,41).

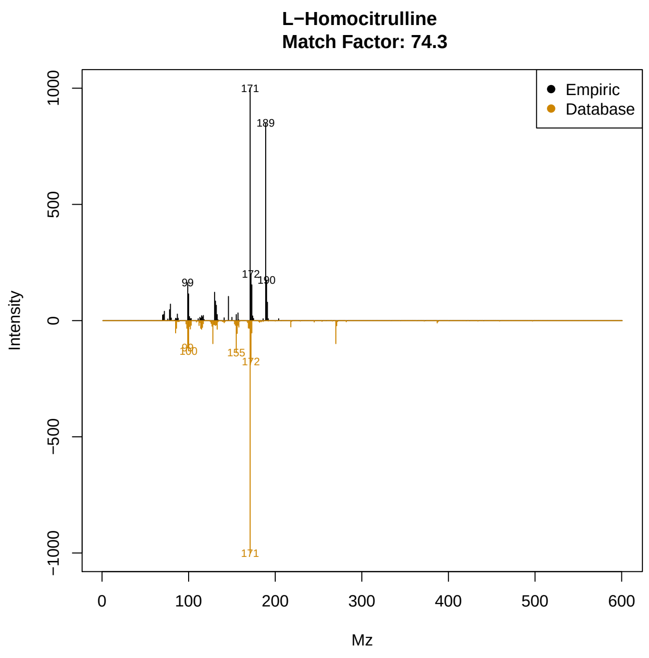

Figure 4: Image from plotSpectra(ex,41, 2, draw.color='orange3').

This plots Figure 4, which 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 spectra is more similar to Urea than to Homocitrulline.

4.6 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.

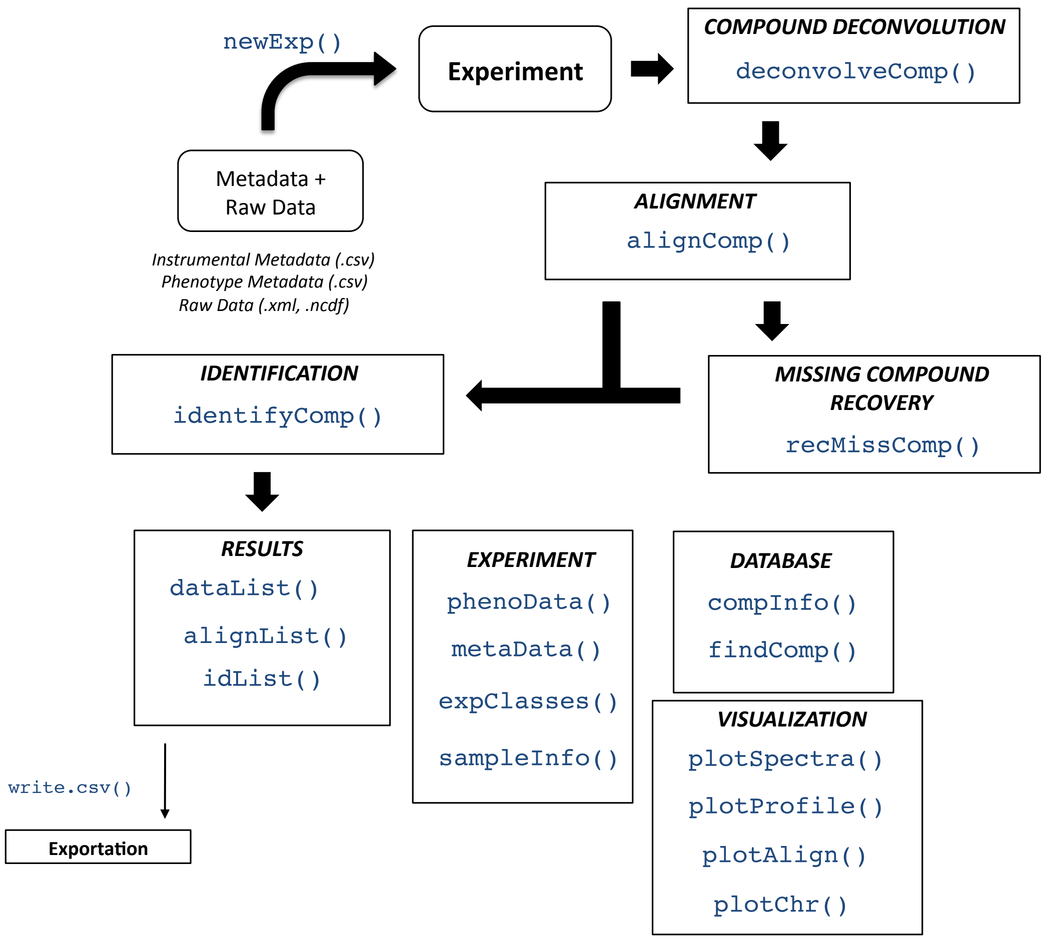

Figure 5: **eRah** summary of functions.