Annotation of LC-MS metabolomics datasets by the "metaMS" package

Pietro Franceschi

October 13, 2015

1 Introduction

metaMS is designed to perform the analysis of LC-MS and GC-MSbased metabolomics assays. For LC-MS data, its major function `runLC()` is a wrapper around the functions and classes of **xcms** and **CAMERA** and it is designed to process a series of data files producing a peak table with the intensity of each feature - an mz,rt couple - in each one of the samples. The functions and classes of **xcms** and **CAMERA** are used to perform the peak picking, grouping and retention correction, peak filling, and CAMERA annotation. The parameters required at each step are collected into a settings object which is passed as an argument to `runLC`. Three examples, fine-tuned for instruments in our own laboratory, are included:

```
> library(metaMS)
> data(FEMsettings)
> Synapt.RP
```

```
Object of class 'metaMSsettings'
Instrument: Synapt.QTOF.RP
Chromatography: LC
```

Consult the manual pages for more details. Package **metaMS** implements a strategy to generate an annotation database by processing the injections of a list chemical standards, which have been analyzed under the same chromatographic conditions as the samples. The injections of the standards are processed by **xcms** and **CAMERA** and the peak lists are matched against a manually validated reference table to identify the set of features associated to each compound. These features are then organized in the database used for the annotation which is included in the output of `runLC()`. In the following, database generation and feature annotation are described in details, while for the discussion of **xcms** and **CAMERA** the reader should refer to their specific documentation.

2 Example Data

A part of the data used to illustrate database creation and feature annotation is included in `metaMS`, while the more heavy experimental data have been included in the **metaMSdata** package. For LC-MS **metaMSdata** contains four CDF files referring to the analysis of a mixture of chemical standards. These files have been converted to the open source CDF format by using proprietary vendor software (Waters, Databridge). The system specific position of the example files can be determined as follows:

```
> library(metaMSdata)
> cdfpath <- system.file("extdata", package = "metaMSdata")
> files <- list.files(cdfpath, "_RP_", full.names=TRUE)
> files
```

```
[1] "/Library/Frameworks/R.framework/Versions/3.2/Resources/library/metaMSdata/extdata/STDmix_RP_pos
[2] "/Library/Frameworks/R.framework/Versions/3.2/Resources/library/metaMSdata/extdata/STDmix_RP_pos
[3] "/Library/Frameworks/R.framework/Versions/3.2/Resources/library/metaMSdata/extdata/STDmix_RP_pos
[4] "/Library/Frameworks/R.framework/Versions/3.2/Resources/library/metaMSdata/extdata/STDmix_RP_pos
```

The construction of the database is performed on the basis of a manually validated reference table which contains the key analytical information for each standard. The example table is part of **metaMS**:

```
> library(metaMS)
> data(exptable)
```

The content of the table can be visualized as follows:

```
> head(exptable)
```

```
  ChemSpiderID                   compound     formula M.ref mz.observed RTman
1         8711             D-(+)-Catechin    C15H14O6    NA    291.0865  18.3
2       391785       malvidin-3-glucoside   C23H25O12    NA    493.1336  23.5
3      4444362                      rutin   C27H30O16    NA    303.0490  28.7
4     10463792 adenosine 2'-monophosphate C10H14N5O7P    NA    348.0700  10.5
              stdFile
1 STDmix_RP_pos01.CDF
2 STDmix_RP_pos01.CDF
3 STDmix_RP_pos01.CDF
4 STDmix_RP_pos01.CDF
```

- `ChemSpiderID`: an unique numeric identifier for a chemical standard from the freeware ChemSpider database (http://www.chemspider.com).
- `compound`: a string with the human readable name of the standard. This name is used to produce the output of the annotation.
- `formula`: the chemical formula of the compound. This field is included in view of future developments.
- `M.ref`: the theoretical mass for the observed ion. This field is included in view of future developments.
- `mz.observed`: the manually validated m/z value which identifies the "main" ion for this specific compound. In the majority of cases, one would choose the most most intense ion of each compound. To avoid wrong assignments the best practice should be to ask to the analyst to "identify" the nature of the ion (protonated/deprotonated, adduct, dimer, common fragment, ...).
- `RTman`: the manually validated retention time for the standard (in minutes).
- `stdFile`: the complete path which points to the raw file of the injection of the standard.

3 Database Construction

In order to create the database, the `stdFile` column in `exptable` has to be updated with the complete path pointing to the correct CDF file:

```
> exptable$stdFile <- sapply(exptable$stdFile, grep, files, value = TRUE)
> exptable$stdFile
```

```
[1] "/Library/Frameworks/R.framework/Versions/3.2/Resources/library/metaMSdata/extdata/STDmix_RP_pos
[2] "/Library/Frameworks/R.framework/Versions/3.2/Resources/library/metaMSdata/extdata/STDmix_RP_pos
[3] "/Library/Frameworks/R.framework/Versions/3.2/Resources/library/metaMSdata/extdata/STDmix_RP_pos
[4] "/Library/Frameworks/R.framework/Versions/3.2/Resources/library/metaMSdata/extdata/STDmix_RP_pos
```

The annotation database is constructed on the basis of `exptable` with the following workflow:

1. Peak picking. The injections of the standards are processed to produce a feature list by using the parameters specified in the `PeakPicking` slot of the `settings` list. For each feature, the maximum value of the signal on the chromatographic peak (`maxo`) is extracted. For a detailed description of the single settings refer to the package documentation.

   ```
   > metaSetting(Synapt.RP, "PeakPicking")

   $method
   [1] "matchedFilter"

   $step
   [1] 0.05

   $fwhm
   [1] 20

   $snthresh
   [1] 4

   $max
   [1] 50
   ```

2. Feature grouping. `CAMERA` with its default settings is used to group the features in pseudospectra and annotate them looking for isotopes and common adducts. These pseudospectra can be therefore considered as mass spectrometric fingerprints of compounds eluting at a specific retention time. It is important to remember that co-eluting compounds are likely to be grouped together, in particular where chromatographic separation is not optimal (e.g. at the extreme ends of a chromatographic run).

3. Ref. Table Matching. The full list of feature is matched with `exptable` looking for features compatible with `M.ref` and `RTman`. The m/z and retention time tolerances are fixed and specified in the `DBconstruction` element of the `settings` list.

   ```
   > metaSetting(Synapt.RP, "DBconstruction")

   $minfeat
   [1] 3

   $rttol
   [1] 0.3

   $mztol
   [1] 0.01
   ```

   The retention time tolerance is specified in minutes, the m/z one in dalton. The `minfeat` parameter is used to prevent the inclusion in the database of standards with a very low number of associated features. This unfortunate situation can happen, for example, when the signal is too low either because a chemical is not efficiently ionized or because it has been injected with low concentration. The absence of a good matching for a specific compound is notified on the output on the screen.

4. Database Creation The features assigned to each standard are collected into a dataframe which can be used for annotation. At this stage an additional filter on the feature intensities is applied: only the ones with an intensity bigger than `Ithr` are kept. This is done to avoid inserting in the DB low intensity features coming from the noise.

Consider as an example the construction of the database included in the `metaMS` package. It contains four chemical standards:

```
> exptable$compound
```

```
[1] "D-(+)-Catechin"             "malvidin-3-glucoside"
[3] "rutin "                     "adenosine 2'-monophosphate"
```

The reference table has been already described. The data can be loaded as follows:

```
> library(metaMSdata)
> cdfpath <- system.file("extdata", package = "metaMSdata")
> files <- list.files(cdfpath, "_RP_", full.names=TRUE)
> exptable$stdFile <- sapply(exptable$stdFile,
+                            function(x)
+                            files[grep(x,files)])
```

For this example the `minfeat` parameter is set to 2:

```
> metaSetting(Synapt.RP, "DBconstruction.minfeat")  <- 2
```

The database is constructed by using the`createSTDdbLC` function:

```
> LCDBtest <- createSTDdbLC(stdInfo=exptable,
+                           settings = Synapt.RP,
+                           polarity = "positive",
+                           Ithr = 20)
```

The messages on the screen can be used to monitor the progress of the analysis. In practice, the `createSTDdbLC` function is all that users need to use.

The example DB is also available as data object `LCDBtest` and can be loaded with

```
> data(LCDBtest)
```

The example database is a list of three elements:

```
> names(LCDBtest)
```

```
[1] "Reftable" "Info"     "DB"
```

The first contains the reference table:

```
> head(LCDBtest$Reftable)
```

```
  ChemSpiderID                   compound     formula M.ref mz.observed RTman
1         8711             D-(+)-Catechin    C15H14O6    NA    291.0865  18.3
2       391785       malvidin-3-glucoside   C23H25O12    NA    493.1336  23.5
3      4444362                      rutin    C27H30O16    NA    303.0490  28.7
4     10463792 adenosine 2'-monophosphate C10H14N5O7P    NA    348.0700  10.5
                                                                      stdFile
1 /home/pietro/R/x86_64-pc-linux-gnu-library/3.0/metaMSdata/CDF_LC/STDmix_RP_pos01.CDF
2 /home/pietro/R/x86_64-pc-linux-gnu-library/3.0/metaMSdata/CDF_LC/STDmix_RP_pos01.CDF
3 /home/pietro/R/x86_64-pc-linux-gnu-library/3.0/metaMSdata/CDF_LC/STDmix_RP_pos01.CDF
4 /home/pietro/R/x86_64-pc-linux-gnu-library/3.0/metaMSdata/CDF_LC/STDmix_RP_pos01.CDF
```

The second contains the settings and the date of creation of the DB:

```
> names(LCDBtest$Info)
```

```
[1] "Modified" "settings"
```

The third is the true database:

```
> head(LCDBtest$DB)
```

```
  ChemSpiderID       compound                 adduct   isotopes       mz
1         8711 D-(+)-Catechin [M+2H-HCOOH]2+ 290.08             123.0447
2         8711 D-(+)-Catechin                          [11][M]+ 139.0397
3         8711 D-(+)-Catechin                          [15][M]+ 165.0559
4         8711 D-(+)-Catechin     [M+H-H20]+ 290.08             273.0757
5         8711 D-(+)-Catechin         [M+H]+ 290.08   [25][M]+ 291.0865
6         8711 D-(+)-Catechin                        [25][M+1]+ 292.0910
        rt      maxo validated
1 18.23447  83.31494 automatic
2 18.23447 135.40686 automatic
3 18.25618  52.00934 automatic
4 18.23447  30.37975 automatic
5 18.23447 341.33984 automatic
6 18.23447  62.35062 automatic
```

Each line of this `data.frame` is a feature detected at `mz` and `rt` with an intensity `maxo`. The output of **CAMERA** annotation are presented in the `adduct` and `isotopes` fields. `ChemSpiderID` and `compound` identify the compound which a feature is associated to. The `validate` column is set to `automatic` to indicate that the feature has been assigned to the neutral by using an automatic algorithm, without performing any manual validation of the results.

It is interesting to see how many features are included in the DB and their association to the four chemical standards included in the reference table:

```
> table(LCDBtest$DB$compound)
```

```
           D-(+)-Catechin       malvidin-3-glucoside
                        7                          2
                    rutin  adenosine 2'-monophosphate
                       11                          4
```

4 Annotation

In the previous section, we have illustrated how to create a database from a series of injections of standards. This DB can then be used to annotate the results of the analysis of a complete metabolomic experiment by passing it to the main `runLC` function. It is important to remember that this type of annotation relies very much on the retention time, so it gives its best results when the standards and the samples have been analyzed under the same chromatographic and mass-spectrometric conditions.

Considering that in LC-MS experiments co-elution of different compounds is the rule rather than the exception, the annotation is performed feature-wise (each feature is independently matched with the DB) and a subsequent validation step is performed. The idea is to retain annotations only if more than one feature associated to a specific compound is found in the peak list. How many "validation" features are requested is an adjustable parameter included in the (`minfeat` slot of the settings object). For matching and validation it is necessary to specify mass and m/z tolerances, accounting for mass and retention time shifts. For retention time the tolerance is fixed and it is specified in the settings. For m/z, the

package implements either a *fixed tolerance* or an *adaptive tolerance*. The use of an *adaptive tolerance* to optimize the results of the annotation for Q-TOF spectrometers has been proposed by the authors in [1]. With this approach, the optimal m/z tolerance is calculated taking into account the m/z value for a specific ion and its intensity: these two parameters are indeed affecting the accuracy of this specific class of analyzers. A more detailed description of the approach can be found in the manual pages of the package and in the specific reference.

The implemented annotation strategy can be broken down in the following steps:

1. Feature wise Annotation Each feature detected by `runLC` is matched against the database. If the mass error function is provided, the appropriate m/z tolerance is calculated, otherwise a fixed tolerance is used (`mzdiff`). The retention time tolerance is fixed and should be selected on the bases of the characteristics of each chromatographic method (`rtdiff`). Multiple annotations - i.e. features which are associated to more than one compound - are possible. This outcome does not indicate a problem *per se*, but is an inherent drawback of co-elution.

2. Annotation Validation The annotated features are organized in "pseudospectra" collecting all the experimental features which are assigned to a specific compound. A specific annotation is confirmed only if more than `minfeat` features which differ in retention time less than `rtval` are present in a pseudospectrum. As a general rule `rtval` should be narrower than `rtdiff`. The latter, indeed, accounts for shifts in retention time between the injection of the standards and the metabolomics experiment under investigation. This time can be rather long, considering that the standards are not commonly re-analyzed each time. On the other hand,`rtval` represents the shift between the ions of the same compound within the same batch of injections and therefore it has only to account for the smaller shifts occurring during peak picking and alignment.

To illustrate the procedure consider the results of the annotation of the example data included in `metaMSdata` with the `LCDBtest` db. Here we set a fixed mass tolerance and we use the settings for a Reverse Phase chromatography.

```
> LC <- runLC(files, settings = Synapt.RP, DB = LCDBtest$DB)
```

The progress of the analysis can be followed from the messages on the screen. As before, the results from this vignette are available in data object `LCresults` – this is used here to demonstrate the structure of the output without having to create it on the fly, which simply takes too much time.

```
> data(LCresults)
```

A summary of the results of the annotation can be found in the `Annotation` element of `LCresults`:

```
> head(LCresults$Annotation$annotation.table)
```

```
  feature db_position ChemSpiderID        mz       rt          I
1     238          21     10463792  137.0651 10.29328   35.58127
2     241          22     10463792  348.0699 10.30414 3754.53906
3     244          23     10463792  696.1375 10.31496   34.43039
4     255          21     10463792  137.0649 10.75478   53.04349
5     260          23     10463792  696.1368 10.76082   68.55536
6     261          24     10463792 1042.1931 10.76082   39.59879
                   compound     db_mz    db_rt       db_I    db_ann mz.err
1 adenosine 2'-monophosphate  137.0647 10.75965   33.41772 automatic  0.005
2 adenosine 2'-monophosphate  348.0716 10.78127 4478.66016 automatic  0.005
3 adenosine 2'-monophosphate  696.1342 10.75965   58.21411 automatic  0.005
4 adenosine 2'-monophosphate  137.0647 10.75965   33.41772 automatic  0.005
5 adenosine 2'-monophosphate  696.1342 10.75965   58.21411 automatic  0.005
6 adenosine 2'-monophosphate 1042.1933 10.75965   26.32912 automatic  0.005
  clid
```

```
1    1
2    1
3    1
4    2
5    2
6    2
```

This dataframe contains the complete results of the annotation.

- `feature`: the index of the annotated feature in the peak table.
- `ChemSpiderID`: the Chem Spider ID of the neutral the feature is associated to.
- `dbposition`: the position inside the DB of the matching entry.
- `mz, rt, I`: mass, retention time and intensity of the feature.
- `compound`: the (human) readable name of the standard.
- `db_mz, db_rt, db_I, db_ann`: information relative to the corresponding DB entry.
- `mz.err`: the m/z error used in the matching.
- `clid`: the results of a hierarchical clustering of the retention times of the annotated features. This can be used to identify the presence of sub groupings of the features which are assigned to the same standard, thus suggesting the presence of co-eluting isomers/compounds. The HC tree is cut at a hight of `rtval`.

```
> LCresults$Annotation$compounds
```

```
[1] "adenosine 2'-monophosphate" "D-(+)-Catechin"
[3] "malvidin-3-glucoside"       "rutin "
```

```
> LCresults$Annotation$ChemSpiderIDs
```

```
[1] 10463792     8711   391785  4444362
```

These are the list of standards found in the peaklist.

```
> LCresults$Annotation$multiple.annotations
```

```
numeric(0)
```

This is the list of features which show multiple annotations

```
> LCresults$Annotation$ann.features
```

```
 [1] 238 241 244 255 260 261 265 462 463 464 465 469 472 474 616 618 765 766 768
[20] 769 770 771 772 773 775 785 791
```

The list of the features with annotation.

Inside the results,the outputs of the annotation are also included in a more compact form as part of the peak table in the `ChemSpiderID` and `compound` columns:

```
> head(LCresults$PeakTable)
```

	ChemSpiderID	compound	pcgroup	adduct	isotopes	mz	rt
1			115			252.9734	0.6283927
2			135			233.0649	0.9481111
3			135			221.0198	0.9535292
4			135			282.0654	0.9535292
5			135			339.6040	0.9535292
6			135			213.9858	0.9565840

	STDmix_RP_pos01	STDmix_RP_pos02	STDmix_RP_pos03	STDmix_RP_pos04
1	144.50919	343.37668	198.53423	207.49843
2	32.63088	54.49682	47.94568	25.78470
3	17.02691	34.96705	30.16131	31.07106
4	30.43845	56.92614	47.80102	55.25565
5	35.44303	28.47058	53.00211	41.54268
6	58.50821	79.08769	119.44610	80.88001

References

[1] Nir Shahaf, Pietro Franceschi, Panagiotis Arapitsas, Ilana Rogachev, Urska Vrhovsek, and Ron Wehrens. Constructing a mass measurement error surface to improve automatic annotations in liquid chromatography/mass spectrometry based metabolomics. *Rapid Communications in Mass Spectrometry*, 27(21):2425–2431, 2013.