Metabolite Identification with MetFrag in R

C. Ruttkies, S. Neumann, A. Helmchen

10. September 2016

Table of Contents

1 Introduction

In this section the MetFragR package is introduced and described. For a better understanding the functionality of a mass spectrometer (MS/MS) was outlined. Initially, the package shall be loaded.

```
> library(metfRag)
```

For the understanding of the metabolism of several organisms different analysis methods are necessary. One of these methods is referred as mass spectrometry to detect and identify compounds in samples [**Dunn2008**]. These compounds are partially unknown and have to be determined and annotated to understand other metabolic circles and pathways. On the other hand mass spectrometry can help to identify known proteins or posttranslational modifications (PTMs) like alteration caused by phosphate.

Actually, to obtain applicable results of an unknown molecule tandem mass spectrometry MS/MS or MS^2 is used, because additional structural hints and the exact mass of fragments of a compound are delivered [**Wolf2010**]. Hence, two mass spectrometers are combined. In the first MS based on a given mass a single precursor ion is selected. The molecule is colliding with a neutral gas (collision-induced dissociation (CID)). This compound is then cleaved into several fragments which can be distinguished by a given mass [**Jeol2006**].

This results in a MS/MS spectrum which contains only product ions from selected precursor ion. The spectra indicate the mass-to-charge ratio (m/z ratio) which used by the MS/MS to deflect specific ions to the analysator. The ion detector data produce a fragmentation spectrum with the m/z ratio against the abundance of each compound.

Hence, to use the statistical power of R we write the package *metfRag* which expand the features of MetFrag by the capabilities of R. Metfrag performs an in silico refragmentation of compounds. Therefor, compound libraries are compared with spectra of unknown metabolites [**Wolf2010**]. Metfrag is written in Java so that it was integrated in metfRag by using *rJava* [**Urbanek2013**] The package 'metfRag' uses the precursor mass, the compound id or the formula to obtain a candidate list of compounds from KEGG or PubChem. Additionally, it provides functions to process the obtained candidate list and calculates ranking parameters like the optimistic, the pessimistic or the relative rank.

2 Methods of the MetFragR package

In this section the functions of this package are described. Firstly, the loading of compounds and the processing are explained. Then a ranking procedure is introduced to generate own ranking methods by using own properties or functions. Additionally, an example illustrates the functional principle on mentioned methods.

2.1 Structure file preparation

To load compounds from a chemical table file like SDF you can directly use the function from *rcdk*. In the example below a metfRag system file is used. The included SDF is the result of MetFusion processing of the MS/MS spectrum of the CASMI 2014 challenge 5 [**Guha2007**].

```
> file <- "sdf/MetFrag_Candidates.sdf"
> sdf <- system.file(file, package = "metfRag")
```

2.2 Scoring structures with MetFrag

To score received compounds from a database MetFragR uses two options. On the one hand scoring of a chemical table format SDF or mol file. For this option metfRag needs a path. Obligatory this package needs the mass-to-charge ratio values from the peak list, the intensity values and the exact mass of the precursor ion.

```
> file <- "sdf/MetFrag_Peaklist.mf"
> queryfile <- system.file(file, package = "metfRag")
> peaktable <- read.table(queryfile, sep=" ", col.names=c("mz", "int"))
> scored.molecules <- score.molecules.from.sdf(sdf,
+                                              mzs=peaktable[,"mz"],
+                                              ints=peaktable[,"int"],
+                                              exact.mass=290.0646,
+                                              number.threads = 1,
+                                              mz.abs=0.001,
+                                              mz.ppm=5,
+                                              search.ppm=5,
+                                              pos.charge=TRUE,
+                                              mode=1,
+                                              tree.depth=2)
```