IsotopicLabelling R Package: a Practical Guide

Ruggero Ferrazza

2016-04-04

Contents

1 Introduction

The purpose of this document is to explain how to use the *IsotopicLabelling* R package to analyse mass spectrometric isotopic patterns obtained following isotopic labelling experiments.

A typical labelling experiment makes use of substrates enriched in one stable isotope, such as ^{2}H or ^{13}C; consequently, after the growth period, some metabolites are expected to have incorporated the labelling isotope, and therefore its relative distribution within them will be different from its natural occurrence. The *IsotopicLabelling* R package is based on the principle that, since the isotopic patterns obtained in mass spectrometry reflect the isotopic compositions of the elements making up the observed species, the amount of labelling can be assessed by their proper examination.

Worth of note, because there could be overlapping between the isotopic patterns of different species, the isotopic pattern analysis is better suited for LC-MS or GC-MS data rather than for direct-infusion MS, where the chromatographic step prior to MS detection reduces such issues. Therefore, the current implementation of the package only works for LC-MS or GC-MS data.

Below is a step-by-step explanation on how to use the *IsotopicLabelling* package; the example data set that comes with the package will be used, and the involved functions will be properly introduced and discussed.

2 The MS Data Set

The *IsotopicLabelling* package requires the MS data to be a data frame with the first two columns representing *m/z* and retention time (RT) of the mass peaks, and the other columns containing peak intensities or areas (one column for each sample to be analysed).

Since a popular R package for handling MS data is *xcms*, the *IsotopicLabelling* R package can also read in *xcmsSet* objects, which basically contain the peak intensities or areas associated to each of the processed samples, together with their average retention time and *m/z*. In this case, the function `table_xcms` is available for converting such objects to the required data frame.

The example data set included in the package is easily accessible:

```
data("xcms_obj")
```

This is an *xcmsSet* object representing lipid extracts of 8 samples from ^{13}C labelling experiments:

- The first 4 samples are relative to unlabelled cell cultures (natural ^{13}C abundance);
- In the last 4 samples the cells were grown in a substrate where the glucose was replaced by uniformly-labelled ^{13}C glucose (99% ^{13}C labelling).

This LC-MS data was kindly provided by Dr. Jules Griffin and Dr. Nyasha Munjoma (Department of Biochemistry, University of Cambridge - UK).

The conversion of the *xcmsSet* object to the required data frame is achieved through:

```
peak_table <- table_xcms(xcms_obj)
```

Here are a few rows of the obtained data frame:

```
##          mz         rt C12_Sample_1 C12_Sample_2 C12_Sample_3 C12_Sample_4
## 57 157.1582   39.70566   18413.8067    16463.212    19478.077   18099.8879
## 58 158.1609   39.70434    1629.4445     1781.991     1722.311    1755.5812
## 59 158.9439 1077.66724     599.9926           NA           NA     588.8424
##    C13_Sample_1 C13_Sample_2 C13_Sample_3 C13_Sample_4
## 57    15949.496    22425.510    21650.582   24001.7999
## 58     1728.988     2647.263     1868.273    1964.7871
## 59           NA           NA           NA     489.4959
```

In addition to *xcms*, this data frame can be obtained in a number of other independent ways, such as through proprietary software of the vendor of the MS instrument; the important point is for the data frame to be properly formatted:

- its first column, named "mz", should contain the mass-to-charge ratios of the peaks;
- its second column, named "rt", should contain the average retention times of the peaks (in seconds);
- the other columns should be named after the samples (one column for sample), and contain peak intensities or areas.

3 A Compact Way of Processing the Data

From the MS data frame, the whole isotopic pattern analysis can be performed through the single, compact function `main_labelling`. As explained in the reference manual, it requires some input parameters:

- **peak_table**, the data frame containing MS peak intensities or areas;
- **compound**, a character vector specifying the chemical formula of the compound of interest. A special notation should be used, whereby the character "X" denotes the element with unknown isotopic distribution. For example, the proton adduct of phosphocholine 32:2, $[PC\ 32{:}2 + H]^+$, has chemical formula C40H77NO8P, but it should be written "X40H77NO8P" for ^{13}C labelling experiments, and "C40X76HNO8P" for ^{2}H experiments, in this last case keeping in mind that one hydrogen atom comes from the solvent, and has therefore fixed natural abundance. Please note that adduct ions should be specified, and not the neutral molecular species;
- **labelling**, a character, either "H" or "C", specifying the labelling isotope;
- **mass_shift**, the maximum difference between measured and true mass. In other words, the mass accuracy;
- **RT**, the expected retention time of the compound of interest;
- **RT_shift**, the maximum difference between true and expected retention time;
- **chrom_width**, an estimate of the chromatographic width of the peaks;
- **initial_abundance**, either NA (the default value) or a numeric vector with length equal to the number of samples, containing the initial estimated percentage isotopic abundances of the labelling isotope. If provided, numbers between 0 and 100.

Using the example data set, the parameters to enter for $[PC\ 32{:}2 + H]^+$ are:

```
fitted_abundances <- main_labelling(peak_table, compound="X40H77NO8P", labelling="C",
                                    mass_shift=0.05, RT=285, RT_shift=20,
                                    chrom_width=7, initial_abundance=NA)
```

The output is an object of class *labelling*, a list containing the results of the analysis.

Further details will be given in the following sections, where each of the steps undertaken by `main_labelling` will be discussed and critically discussed.

4 Step-by-Step Processing

The *IsotopicLabelling* package aims to find the abundance of the isotope used during the labelling experiment within the compounds of interest, based on their MS isotopic patterns. A number of algorithms are available to a-priori compute the isotopic patterns knowing the isotopic abundances; the goal of the *IsotopicLabelling* package is the opposite: starting from measured experimental patterns, this package aims at understanding which is the isotopic distribution giving rise to those patterns. This is basically achieved by a fitting procedure through which the abundance of the labelling isotope is iteratively changed until the best match between predicted and experimental patterns is found.

The `main_labelling` function basically performs three successive steps:

1. It gathers information about the isotopes and all the possible isotopologues arising from the labelling (`isotopic_information` function);
2. It extracts from the experimental data the isotopic patterns of the compound of interest (`isotopic_pattern` function);
3. It analyses the extracted patterns to estimate the percentage isotopic abundance of the labelling isotope.

An alternative to using `main_labelling` is to run the three distinct functions it is made of, and this route is covered below.

4.1 Isotopes and Isotopologues

The first function used by `main_labelling` is `isotopic_information`, which summarizes important isotopic information in a single object, a list required by the subsequent functions. The input parameters are the chemical formula and the type of labelling; for $[PC\ 32{:}2 + H]^+$, the list can be obtained through:

```
info <- isotopic_information(compound="X40H77NO8P", labelling="C")
```

As detailed in the reference manual, the output is a named list:

```
attributes(info)
```

```
## $names
## [1] "compound" "isotopes" "target"   "nX"       "nTOT"
```

In particular, "isotopes" is a table with the natural isotopic abundances (numbers between 0 and 1) of the elements present in the compound. The two isotopes of the labelling element X are given NA values:

```
info$isotopes
```

```
##    element      mass abundance
## 1        H  1.007825  0.999885
## 2        H  2.014102  0.000115
## 3        N 14.003074  0.996360
## 4        N 15.000109  0.003640
## 5        O 15.994915  0.997570
## 6        O 16.999132  0.000380
## 7        O 17.999161  0.002050
## 8        P 30.973762  1.000000
## 9        X 12.000000        NA
## 10       X 13.003355        NA
```

Importantly, "target" is a named vector with the exact masses of all the possible isotopologues arising because of the labelling isotope; in the example, $[PC\ 32{:}2 + H]^+$ has 40 carbon atoms, and therefore the possible isotopologues coming from ^{13}C span a 41 mass range: the lightest one is the monoisotopic species (with 40 ^{12}C atoms), whereas the heaviest is the species with 40 ^{13}C atoms. However, the isotopic patterns also depend on the other elements, and therefore the list of target isotopologues is further extended by two m/z units, enough for small and medium-sized molecules such as lipids and metabolites.

The naming of the target masses follows this convention:

- **M+0** is the monoisotopic mass, the sum of the masses of the atoms using the lightest isotope for each element, X included;
- **M+1** is the mass where one light isotope (either X or any other element) is replaced by its heaviest counterpart;
- **M+i** is the mass where there have been "i" replacements.

The underlying assumption is that the MS resolution is not high enough to resolve the isotopic fine structure; consequently, the replacement of, for example, ^1H with ^2H is indistinguishable from ^{12}C with ^{13}C. This is true for most of the instruments currently used in LC-MS measurements.

4.2 Extraction of the Experimental Isotopic Patterns

Once the target masses are known, the experimental isotopic patterns are extracted from the MS data through the function `isotopic_pattern`. Keeping on with the example of $[PC\ 32{:}2 + H]^+$, this can be achieved through the command:

```
experimental_patterns <- isotopic_pattern(peak_table, info, mass_shift=0.05,
                                          RT=285, RT_shift=20, chrom_width=7)
```

In addition to the table of peaks and the list obtained above, the used also has to provide the parameters `mass_shift`, `RT`, `RT_shift` and `chrom_width` (already discussed above and further detailed in the reference manual).

The output is a matrix containing the extracted signals, with its first two columns reserved for the exact m/z and the retention times of the peaks:

```
head(experimental_patterns, n=3)
```

```
##           mz       rt C12_Sample_1 C12_Sample_2 C12_Sample_3 C12_Sample_4
## M+0 730.5387 282.8571    2033683.5    1504910.9    1702717.5    1556318.5
## M+1 731.5420 282.8571     918440.7     682545.9     772604.1     706638.7
## M+2 732.5454 282.8571     219390.3     169205.2     189325.5     169515.6
##     C13_Sample_1 C13_Sample_2 C13_Sample_3 C13_Sample_4
## M+0            0            0            0            0
## M+1            0            0            0            0
## M+2            0            0            0            0
```

Each of its columns, therefore, represents the extracted experimental pattern for that sample. To get this result, the `isotopic_pattern` function does the following:

1. For all the masses in the "target" vector, it finds and stores the indices of the peaks in the experimental data frame that are within the specified m/z and RT ranges;
2. It considers the retention times of the obtained indices, and compares them across isotopologues in order to group them: peaks that differ in RT within the specified chromatographic width are assumed to be two isotopologues;
3. More groups may have been identified: those containing less than two isotopologues are discarded and, if still more groups are left, the one closer in retention time to the expected value is chosen.

Two of the patterns extracted for $[PC\ 32{:}2 + H]^+$ are shown in Figure 1: the first (to the left) is relative to an unlabelled sample, whereas the second one (to the right) is relative to a labelled sample (99% ^{13}C labelling).

In this simple case, the difference is straightforward: in the labelled sample the most intense signal is shifted 40 mass units upwards with respect to the monoisotopic peak, indicating that the most abundant species is the one where all 40 carbon atoms have been replaced by the labelling isotope, ^{13}C.

4.3 Isotopic Pattern Analysis

The extracted patterns are then analysed by the function `find_abundance`, which takes each of them and finds the best theoretical pattern that reproduces it by iteratively changing the relative abundance of the labelling isotope.

In the example, this information can be achieved through:

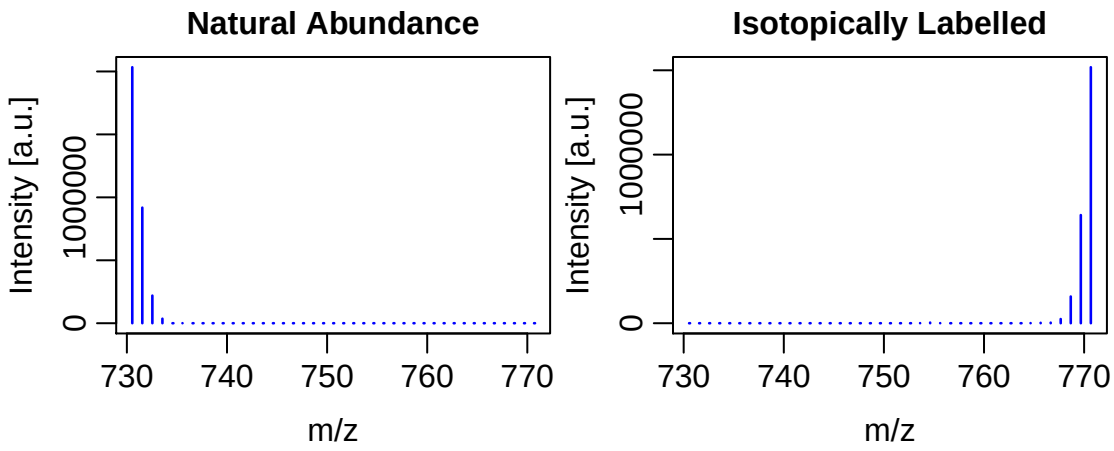

Figure 1: An example showing two of the patterns extracted from the experimental data; to the left is an unlabelled sample, to the right a labelled sample (99% ^{13}C).

```
fitted_abundances <- find_abundance(patterns=experimental_patterns, info=info,
                                    initial_abundance=NA)
```

The output is a class *labelling* object, which is a list containing the results of the isotopic pattern analysis:

```
attributes(fitted_abundances)
```

```
## $names
## [1] "compound"      "best_estimate" "std_error"     "dev_percent"
## [5] "x_scale"       "y_exp"         "y_theor"       "residuals"
## [9] "warnings"
##
## $class
## [1] "labelling"
```

The estimated percentage abundances of the labelling isotope are in "best_estimate" (numbers between 0 and 100), the standard errors from the fitting procedure are in "std_error", whereas the percentage deviations between best fitted and experimental patterns are in "dev_percent". The *m/z* values are in "x_scale", while in "y_exp" are their normalised intensities and in "y_theor" are those of the best fitted theoretical patterns. The differences between the two are in "residuals".

The `find_abundance` function performs the following:

1. It takes each experimental pattern and normalises it to its highest signal, set to 100;
2. If the initial estimates are not provided, it looks for the *m/z* position of the most intense signal, and uses it to get a first rough estimate of the X isotopic abundance;
3. A fitting procedure ensues, whereby the single unkwown variable "isotopic abundance of X" is iteratively changed starting from its initial estimate, and the resulting theoretical patterns are compared to the experimental pattern. The final value is the one that minimizes the sum of squares of the difference between the two normalised patterns. In order to account for noise, a weight is given to the signals, proportional to the square root of their intensities. The theoretical patterns are computed using the *ecipex* R package, which exploits Fourier transforms of simplex-based elemental models.

5 Overview of the Results

There are a number of ways to look at and save the results of the isotopic pattern analysis:

1. The generic function `summary` allows to quickly glance at the estimated percentage abundances:

```
summary(fitted_abundances)
```

```
##                     C12_Sample_1 C12_Sample_2 C12_Sample_3 C12_Sample_4
## Best Estimate [%]   1.074287762  1.080694206  1.080582514  1.080180140
## Standard Error [%]  0.006673445  0.003860987  0.004485302  0.005093723
##                     C13_Sample_1 C13_Sample_2 C13_Sample_3 C13_Sample_4
## Best Estimate [%]  98.942141509 98.947395995 98.936148771 98.945395245
## Standard Error [%]  0.008717031  0.008215919  0.006310484  0.007579631
```

   In this example, the average value is 1.08% for unlabelled samples (close to the natural ^{13}C abundance, 1.07%), whereas it is 98.94% for labelled samples.

2. The generic function `plot` can be used to produce three types of plots, depending on the parameter "type". By default (type="patterns") a series of plots is returned, one for each sample, showing the normalised experimental patterns superimposed to their fitted theoretical patterns:

```
plot(fitted_abundances, type="patterns", saveplots=F)
```

   Two of the plots obtained in this example are in Figure 2.

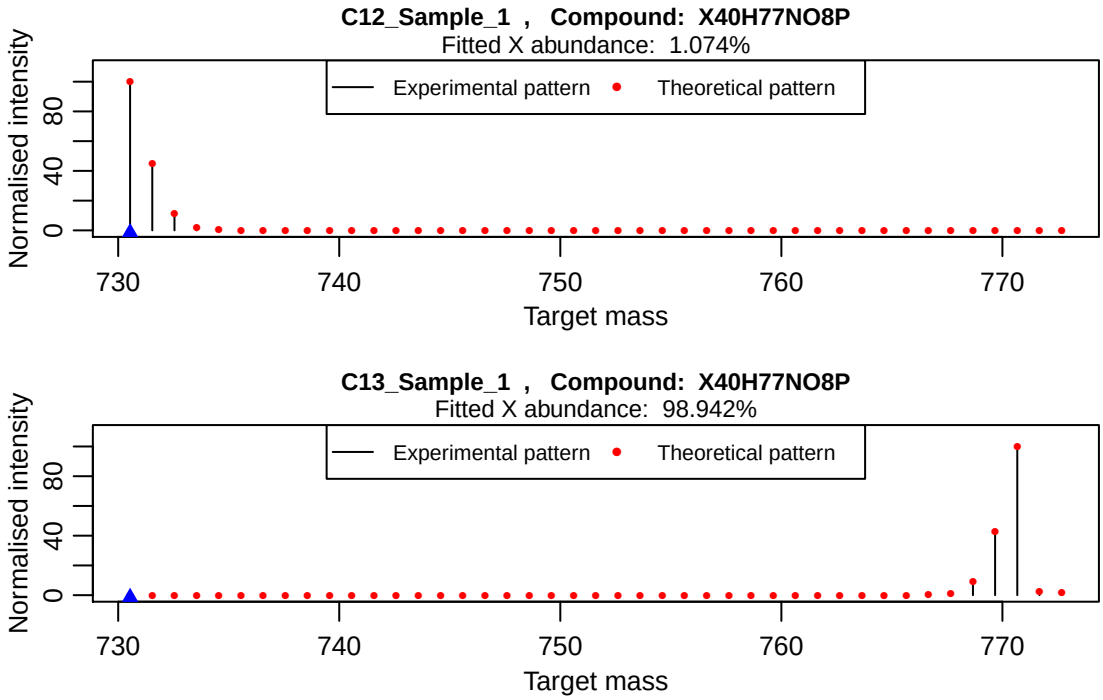

Figure 2: Graphical summary of the isotopic pattern analysis for an unlabelled (top) and a labelled (bottom) sample.

   If "type" is set to "residuals", the residuals are plotted:

```
plot(fitted_abundances, type="residuals", saveplots=F)
```

This is shown in Figure 3.

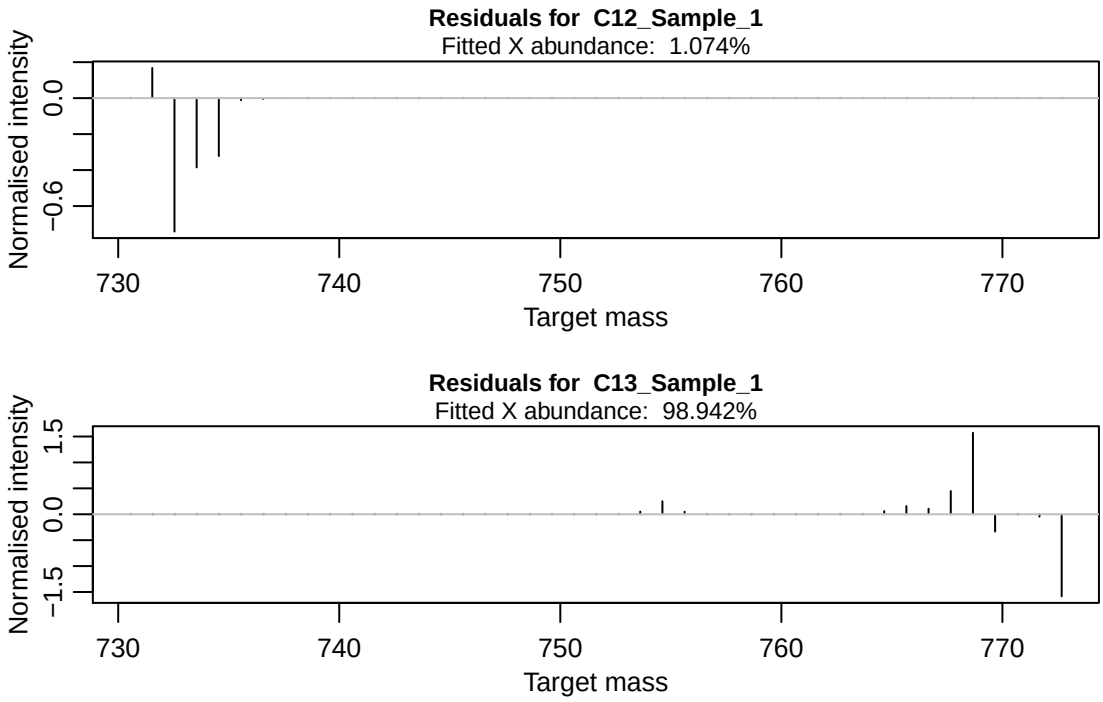

Figure 3: Plot of the residuals for an unlabelled (top) and a labelled (bottom) sample.

Finally, with type="summary", a summary plot with the estimated percentage abundances is provided (see Figure 4).

If the parameter "saveplots" is set to `TRUE`, the plots are saved as a *.pdf file in the working directory.

3. The `save_labelling` function allows to save the results to a *.csv file in the working directory:

```
save_labelling(fitted_abundances)
```

For each sample, this file reports:

a. The estimated percentage abundance of the labelling isotope;
b. The related standard error;
c. The percentage deviation between theoretical and experimental isotopic patterns;
d. The outcome message from the fitting procedure.

6 Average the Estimates Within Groups

Since it is usual in biochemistry to investigate different sample groups and to have more biological replicates within the same group, the *IsotopicLabelling* package provides an additional function that allows to condense the results from the individual fits (carried out on a sample-to-sample basis) into a single estimate for each of the sample groups.

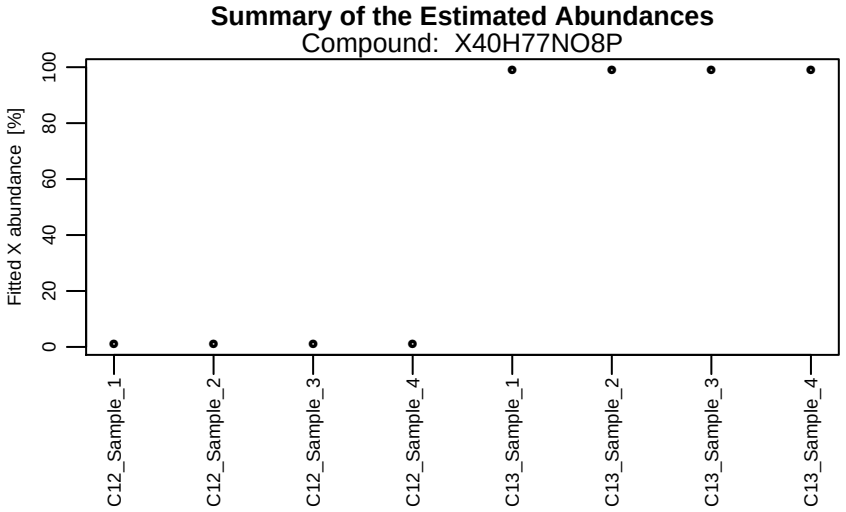

Figure 4: Graphical summary of the estimated percentage abundances and related standard errors, following the isotopic pattern analysis.

For each group, the estimated isotopic abundance of the labelling isotope is calculated by simply averaging the estimates for each sample in that group.

As for the standard error of the mean, this is obtained by taking into account both the individual standard errors coming from the fittings (one for each sample) and the variability of the estimates across samples. According to the law of total variance, the overall variance is given by the sum of the variances of each estimate (the "within-sample variance"), plus the variance caused by the distribution of the obtained estimates (the "across-sample variance"). The standard error of the mean is simply the square root of this amount divided by the number of samples.

This additional information can be achieved starting from the class *labelling* object; for example:

```
grouped_estimates <- group_labelling(fitted_abundances,
                                     groups=factor(c(rep("C12",4), rep("C13",4))))
```

The second input object (`groups`) is a factor specifying the groupings. The output is a data frame containing the results for each group:

```
grouped_estimates
```

```
##     N      Mean     SE mean   t_crit Lower 95% CI Upper 95% CI
## C12 4  1.078936 0.003001192 3.182446     1.069385     1.088487
## C13 4 98.942770 0.004592550 3.182446    98.928155    98.957386
```

The last three columns of the data frame contain the critical value for a 95% confidence interval of the t distribution with N-1 degrees of freedom, and the lower and upper values for the 95% confidence interval.

This data frame contains all the information needed for additional statistical analysis, such as the comparison across sample groups in order to understand whetheher or not two (or more) groups have statistically different estimates.

7 Conclusion

In this document we introduced the main functions of the *IsotopicLabelling* R package, explaining their basic working principles. Using the provided data set, we illustrated how to use the package in practice and how to quickly assess the results, with the hope that this package will prove to be a useful tool to researchers dealing with labelling experiments.

The package, in its current implementation, allows to deal with either ^{2}H or ^{13}C enrichments in the whole range 0-100%. A note of caution is here necessary, though: during the isotopic pattern analysis, the package assumes that for each sample and analyte there is a single (unknown) isotopic distribution of the label. Consequently, *IsotopicLabelling* cannot deal with samples containing multiple sources of label abundances, such as biological samples that were spiked with a labelled analyte, or samples resulting from the pooling of several ones with different label enrichments. Therefore, pooled samples should be avoided in the isotopic pattern analysis.

After having discussed the principles behind this package, our final aim here is to provide a compact summary that may be used as a script for analysing LC-MS data relative to labelling experiments.

```
# Load the package
library("IsotopicLabelling")

# Load the xcmsSet object
data(xcms_obj)

# Convert the object into the required data frame
peak_table <- table_xcms(xcms_obj)

# Process the data
fitted_abundances <- main_labelling(peak_table, compound="X40H77NO8P", labelling="C",
                                    mass_shift=0.05, RT=285, RT_shift=20,
                                    chrom_width=7, initial_abundance=NA)

# Quickly look at the results
summary(fitted_abundances)

# Plot the patterns
plot(fitted_abundances, type="patterns", saveplots=F)

# Plot the residuals
plot(fitted_abundances, type="residuals", saveplots=F)

# Plot the overall results
plot(fitted_abundances, type="summary", saveplots=F)

# Save the results to a *.csv file
save_labelling(fitted_abundances)

# Group the samples and obtain grouped estimates
grouped_estimates <- group_labelling(fitted_abundances,
                                     groups=factor(c(rep("C12",4), rep("C13",4))))
```