Package ‘LipidMS’

October 21, 2019

Type Package
Title Lipid Annotation for LC-MS/MS DIA Data
Version 1.1.0
Author M Isabel Alcoriza-Balaguer, J Carlos Garcia-Canaveras
Maintainer M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>
Depends R (>= 3.1), enviPick, methods, purrr, CHNOSZ, stats, LipidMSdata
License GPL (>= 2)
LazyData TRUE
RoxygenNote 6.1.1
Suggests knitr, rmarkdown
VignetteBuilder knitr
Encoding UTF-8
NeedsCompilation no
Repository CRAN
Date/Publication 2019-10-21 13:30:08 UTC

R topics documented:

adductsTable ................................................................. 3
assignDB ................................................................. 4
baconjdb ................................................................. 4
badb ................................................................. 5
carnitinesdb ............................................................. 5
CEdb ................................................................. 6
cerdb ................................................................. 6
chainFrags ............................................................... 7
checkClass ......................................................... 8
checkIntensityRules ........................................... 10
cldb ................................................................. 12
coeIutingF rags ................................................... 12
coeIutionScore .................................................... 13
combineChains ..................................................... 14
confLevels .......................................................... 15
cReateLipidDB ..................................................... 16
crossTables ......................................................... 17
dataProcessing ..................................................... 18
dgdb ................................................................. 20
fadb ................................................................. 20
fahfadb ............................................................. 21
findCandidates ..................................................... 21
getInclusionList .................................................. 23
hfadbi .............................................................. 23
idBAneg ............................................................. 24
idCarpos ............................................................. 26
idCEpos .............................................................. 28
idCerneg ............................................................. 30
idCerpos ............................................................. 32
idCLneg ............................................................... 35
idDGpos ............................................................... 37
idFAHFAneg ........................................................ 40
idFAneg ............................................................... 42
idLPCneg ............................................................. 44
idLPCpos ............................................................. 46
idLPEneg ............................................................. 48
idLPEpos ............................................................. 50
idLPGneg ............................................................. 53
idLPIneg ............................................................. 55
idLPSneg ............................................................. 57
idMGpos ............................................................... 59
idNEG ................................................................. 61
idPCneg ............................................................... 62
idPCpos ............................................................... 65
idPEneg ............................................................... 67
idPEpos ............................................................... 70
idPGneg ............................................................... 72
idPlneg ............................................................... 75
idPOS ................................................................. 77
idPSneg ............................................................... 79
idSMpos ............................................................... 81
idSphneg ............................................................. 83
idSphPneg ........................................................... 85
idSphpos ............................................................. 87
idSphPpos .......................................................... 89
idTGpos ............................................................... 91
adductsTable

ly sopadb  ................................................................. 94
ly sopcdb  ................................................................. 95
ly sopedb  ................................................................. 95
ly sopgdb  ................................................................. 96
ly sopidb  ................................................................. 96
ly opsdb  ................................................................. 97
mgdb  ................................................................. 97
nlsphdb  ................................................................. 98
organizeResults  ................................................................. 98
padb  ................................................................. 99
pcdb  ................................................................. 100
pedb  ................................................................. 100
pgdb  ................................................................. 101
pidb  ................................................................. 101
psdb  ................................................................. 102
searchIsotopes  ................................................................. 102
sepByCE  ................................................................. 103
smdb  ................................................................. 104
sphdb  ................................................................. 104
sphPdb  ................................................................. 105
tgdb  ................................................................. 105

Index  106

---

adductsTable  Adducts table

Description

Table of possible adducts to be employed by LipidMS and related information.

Usage

data("adductsTable")

Format

Data frame with 18 observations and the following 4 variables.

adduct  character vector with the adducts names.
mdiff  numeric vector indicating the mass differences.
charge  numeric vector indicating the charge.

n  numeric vector. It indicates if the ion is a monomer (1), a dimer (2), etc.
assignDB: load LipidMS default data bases

**Description**

load all LipidMS default data bases required to run identification functions.

**Usage**

```r
assignDB()
```

**Value**

list of data frames

**Author(s)**

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

**Examples**

```r
dbs <- assignDB()
```

---

baconjdb: Bile acids conjugates database

**Description**

Common bile acids conjugates. It can be modified to look for other BA species.

**Usage**

```r
data("baconjdb")
```

**Format**

Data frame with 2 observations and the following 2 variables.

- **total**: character vector indicating the names of the conjugates.
- **Mass**: numeric vector with the neutral masses of the conjugates fragments.
badb

**Bile acids database**

**Description**

In silico generated database for common bile acids.

**Usage**

```r
data("badb")
```

**Format**

Data frame with 9 observations and the following 5 variables.

- `formula`: character vector with the molecular formulas.
- `total`: character vector containing the names of the BAs (i.e. CA, TDCA, GLCA...).
- `Mass`: numeric vector with the neutral masses.
- `conjugate`: character vector containing the conjugate of each BA.
- `base`: character vector containing the core of each BA.

---

carnitinesdb

**Carnitines database**

**Description**

In silico generated database for common carnitines.

**Usage**

```r
data("carnitinesdb")
```

**Format**

Data frame with 30 observations and the following 3 variables.

- `formula`: character vector containing molecular formulas.
- `total`: character vector indicating the total number of carbons and double bounds of the chains.
- `Mass`: numeric vector with the neutral masses.
**CEdb**

**CEs database**

**Description**

In silico generated database for common CEs.

**Usage**

```r
data("CEdb")
```

**Format**

Data frame with 30 observations and the following 3 variables.

- **formula** character vector containing molecular formulas.
- **total** character vector indicating the total number of carbons and double bounds of the chains.
- **Mass** numeric vector with the neutral masses.

---

**cerdb**

**ceramides database**

**Description**

In silico generated database for common ceramides.

**Usage**

```r
data("cerdb")
```

**Format**

Data frame with 52 observations and the following 3 variables.

- **formula** character vector containing molecular formulas.
- **total** character vector indicating the total number of carbons and double bounds of the chains.
- **Mass** numeric vector with the neutral masses.
### chainFrags

**Search of chain specific fragments**

**Description**

Search of specific fragments that inform about the chains structure.

**Usage**

```r
cleanupFrag(coelfrags, chainfrags, ppm = 10, candidates, f = NULL, dbs)
```

**Arguments**

- `coelfrags`: coeluting fragments for each candidate. Output of `coelutingFrag`.
- `chainfrags`: character vector containing the fragmentation rules for the chain fragments. If it is an empty vector, chains will be calculated based on the difference between the precursor and the other chain. See details.
- `ppm`: m/z tolerance in ppm.
- `candidates`: candidates data frame. If any chain needs to be calculated based on the difference between the precursor and the other chain, this argument will be required. Output of `chainFrag`.
- `f`: known chains. If any chain needs to be calculated based on the difference between the precursor and the other chain, this argument will be required. Output of `chainFrag`.
- `dbs`: list of data bases required for the annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be changed. If data bases have been customized using `createLipidDB`, they also have to be modified here.

**Details**

The chainfrags argument must contain the fragmentation rules which inform about the chains structure. For example, in the case of PG subclass, the chain in sn1 position is identified by the lysophosphatidylglycerol (lysoPG) as M-H resulting from the loss of the FA chain of sn2; and the chain in sn2 position is identified as the free FA chain as M-H. These two fragments need to be searched in two different steps: in the first step we will look for lysoPGs coeluting with the precursor using chainfrags = c("lysopg_M-H"); then, we will look for FA chains using chainfrags = c("fa_M-H"). This information can be combined later using `combineChains` function.

To indicate the fragments to be searched, the class of lipid is written using the same names as the LipidMS databases without the "db" at the end (i.e. pa, dg, lysopa, mg, CE, etc.), and the adduct has to be indicated as it appears in the adductsTable, both parts separated by ".". In case some chain needs to be searched based on a neutral loss, this can be defined using "NL-" prefix, followed by the database and adduct. If this neutral loss is employed to find the remaining chain, "cbdiff-" prefix allows to calculate the difference in carbons and doubles bounds between the precursor and the building block found. For example, "cbdiff-dg_M+H-H2O" will look for DG as M+H-H2O and...
checkClass

Search of class fragments to confirm the lipid class.

Description

Search of characteristic fragments that confirm a given lipid class.

Usage

checkClass(candidates, coelfrags, clfrags, ftype, clrequisites, ppm = 10, 

dbs)
Arguments

candidates: output of `findCandidates` function.

coelfrags: list of peaks coeluting with each candidate. Output of `coelutingFrags`.

clfrags: vector containing the expected fragments for a given lipid class. See details.

ftype: character vector indicating the type of fragments in clfrags. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See details.

clrequisites: logical vector indicating if each class fragment is required or not. If none of the fragment is required, at least one of them must be present within the coeluting fragments. If the presence of any fragment excludes the class, it can be specified by using "excluding".

ppm: m/z tolerance in ppm.

dbs: list of data bases required for the annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be changed. If data bases have been customized using `createLipidDB`, they also have to be modified here. It is employed when some fragment belongs to "BB" ftype.

Details

clfrags, ftype and clrequisites will indicate the rules to confirm a lipid class. All three arguments must have the same length.

This function allows three different types of fragments: fragments with a specific m/z as for example 227.0326 for PG in negative mode, which needs to be defined as clfrags = c(227.0326) and ftype = c("F"); neutral losses such as the head group of some PL (i.e. NL of 74.0359 in PG in negative mode), which will be defined as clfrags = c(74.0359) and ftype = c("NL"); or building blocks resulting from the loss of some groups, as for example, PA as M-H resulting from the loss of the head group (glycerol) in PG in ESI-, which will be defined as clfrags = c("pa_M-H") and ftype = c("BB"). The last two options could define the same fragments. In this case just one of them would be necessary.

When using the third type of fragment ("BB"), the building block will be specified in lower case (i.e. pa, dg, lysopa, mg, etc.) and the adduct will be given as it appears in the adductsTable, both separated by ". Names for the building blocks are the ones used for the LipidMS databases without the "db" at the end.

In case the presence of a fragment indicates that the candidate does not belong to the lipid class (i.e. loss of CH3 in PE, which corresponds to a PC actually), this will be specified by using clrequisites = c("excluding").

Value

List with 2 elements: a matrix with logical values (presence/absense) of each expected fragment (columns) for each candidate (rows), and a logical vector with the confirmation of the lipid class for each candidate.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>
checkIntensityRules

Examples

```
library(LipidMSdata)
dbs <- assignDB()

candidates <- findCandidates(MS1 = MS1_neg$peaklist,
  db = dbs$pgdb, ppm = 10, rt = c(0, 2000), adducts = c("M-H"),
  rttol = 10, rawData = MS1_neg$rawScans, coelCutoff = 0.8)

MSMS <- rbind(MSMS1_neg$peaklist, MSMS2_neg$peaklist)
rawData <- rbind(MS1_neg$rawScans, MSMS1_neg$rawScans,
  MSMS2_neg$rawScans)
coelfrags <- coelutingFrags(candidates$RT, MSMS, rttol = 10, rawData = rawData,
  coelCutoff = 0.8)

classConf <- checkClass(candidates, coelfrags,
  clfrags = c(227.0326, 209.022, 74.0359), clrequisites = c(F, F, F, F),
  ftype = c("F", "F", "NL"), ppm = 10, dbs = dbs)
```

---

checkIntensityRules  Check intensity rules

Description

Check intensity rules to confirm chains position.

Usage

```
checkIntensityRules(intrules, rates, intrequired, nchains, combinations)
```
Arguments

intrules character vector specifying the fragments to compare. See details.
rates character vector with the expected rates between fragments given as a string (i.e. "3/1"). See details.
intrequired logical vector indicating if any of the rules is required. If not, at least one must be verified to confirm the structure.
ncchains number of chains of the targeted lipid class.
combinations output of combineChains.

Details

This function will be employed when the targeted lipid class has more than one chain.

Taking PG subclass as an example, intensities of lysoPG fragments (informative for sn1) can be employed to confirm the chains structure (intrules = c("lysopg_sn1/lysopg_sn1")). In this case, the intensity of the lysoPG resulting from the loss of the FA chain in sn2 is at least 3 times greater (rates = c("3/1")) than the lysoPG resulting from the loss of the FA chain in sn1.

For the intrules argument, "/" will be use to separate the fragments related to each chain (sn1/sn2/etc), and "/" will be use to indicate the list in which they'll be searched. This will depend on the chain fragments rules defined previously. Following the example, as we use lysoPG to define the sn1 position, both fragments will be searched in this list (sn1).

For classes with more than one FA chain, if some intensity rule should be employed to identify their position but they are no defined yet, use "Unknown". If it is not necessary because the fragmentation rules are informative enough to define the position (i.e. sphingolipid species), just leave an empty vector.

Value

List of logical vectors with the confirmation for each combination.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

library(LipidMSdata)
dbs <- assignDB()
candidates <- findCandidates(MS1 = MS1_neg$peaklist, 
db = dbs$pgdb, ppm = 10, rt = c(0, 2000), adducts = c("M-H"), 
rttol = 10, rawData = MS1_neg$rawScans, coelCutoff = 0.8)

MSMS <- rbind(MS1_neg$peaklist, MSMS2_neg$peaklist)
rawData <- rbind(MS1_neg$rawScans, MSMS1_neg$rawScans, 
MSMS2_neg$rawScans)
coelfrags <- coelutingFrags(candidates$RT, MSMS, rttol = 10, rawData = rawData, 
coelCutoff = 0.8)
sn1 <- chainFrags(coelfrags, chainfrags = c("lysopg_M-H"), ppm = 10, dbs = dbs)
sn2 <- chainFrags(coelfrags, chainfrags = c("fa_M-H"), ppm = 10, dbs = dbs)
chainsComb <- combineChains(candidates, nchains=2, sn1, sn2)
intConf <- checkIntensityRules(intrules = c("lysopg_sn1/lysopg_sn1"),
rates = c("2/1"), intrequired = c(T), nchains=2, chainsComb, sn1, sn2)

---

**cldb**

*Cardiolipins database*

**Description**

In silico generated database for commo CLs.

**Usage**

```r
data("cldb")
```

**Format**

Data frame with 714 observations and the following 3 variables.

- `formula` character vector containing molecular formulas.
- `total` character vector indicating the total number of carbons and double bounds of the chains.
- `Mass` numeric vector with the neutral masses.

---

**coelutingFrags**

*Coeluting fragments extraction*

**Description**

Given a RT and a list of peaks, this function subsets all coeluting fragments within a rt windows. It is used by identification functions to extract coeluting fragments from high energy functions for candidate precursor ions.

**Usage**

```r
coenlingFrags(precursors, products, rttol, rawData = data.frame(), coelCutoff = 0)
```
coelutionScore  

Arguments

precursors  candidates data frame. Output of `findCandidates`.
products  peaklist for MS2 function (MSMS).
rttol  rt window in seconds.
rawData  raw scans data. Output of `dataProcessing` function (MSMS$rawData).
coelCutoff  coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied.

Value

List of data frames with the coeluting fragments for each candidate.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

```r
library(LipidMSdata)
dbs <- assignDB()
candidates <- findCandidates(MS1 = MS1_neg$peaklist, 
db = dbs$pgdb, ppm = 10, rt = c(0, 2000), adducts = c("M-H"),
rttol = 10, rawData = MS1_neg$rawScans, coelCutoff = 0.8)
MSMS <- rbind(MSMS1_neg$peaklist, MSMS2_neg$peaklist)
rawData <- rbind(MS1_neg$rawScans, MSMS1_neg$rawScans, MSMS2_neg$rawScans)
coelFfrags <- coelutingFrags(candidates$RT, MSMS, rttol = 10, rawData = rawData, 
coelCutoff = 0.8)
```

---

cotelutionScore  

`coelutionScore` calculates coelution score between two peaks.

Description

Calculate coelution score between two peaks.

Usage

`coelutionScore(peak1, peak2, rawData)`
combineChains

Arguments

- **peak1**: character vector specifying the peakID of the first peak.
- **peak2**: character vector specifying the peakID of the second peak.
- **rawData**: data frame with raw data for each scan. It need to have at least 5 columns: m.z, RT, int, Scan (ordinal number for a given MS function) and peakID (peakID to which it has been assigned).

# @keywords internal

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

---

**combineChains**

*Combine chain fragments that could belong to the same precursor.*

**Description**

It calculates combinations of chain fragments that sum up the same number of carbons and double bounds as the precursor.

**Usage**

```r
combineChains(candidates, nchains, sn1, sn2, sn3, sn4)
```

**Arguments**

- **candidates**: candidates data frame. Output of `findCandidates`.
- **nchains**: number of chains of the targeted lipid class.
- **sn1**: list of chain fragments identified for sn1 position. Output of `chainFrags`.
- **sn2**: list of chain fragments identified for sn2 position. Output of `chainFrags`. If required.
- **sn3**: list of chain fragments identified for sn3 position. Output of `chainFrags`. If required.
- **sn4**: list of chain fragments identified for sn4 position. Output of `chainFrags`. If required.

**Value**

List of data frames with candidate chains structures.

**Author(s)**

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>
library(LipidMSdata)
dbs <- assignDB()

candidates <- findCandidates(MS1 = MS1_neg$peaklist,
    db = dbs$pgdb, ppm = 10, rt = c(0, 2000), adducts = c("M-H"),
    rttol = 10, rawData = MS1_neg$rawScans, coelCutoff = 0.8)

MSMS <- rbind(MSMS1_neg$peaklist, MSMS2_neg$peaklist)
rawData <- rbind(MS1_neg$rawScans, MSMS1_neg$rawScans, MSMS2_neg$rawScans)
coelfrags <- coelutingFrags(candidates$RT, MSMS, rttol = 10, rawData = rawData,
    coelCutoff = 0.8)

sn1 <- chainFrags(coelfrags, chainfrags = c("lysopg_M-H"), ppm = 10,
    dbs = dbs)

sn2 <- chainFrags(coelfrags, chainfrags = c("fa_M-H"), ppm = 10, dbs = dbs)

chainsComb <- combineChains(candidates, nchains = 2, sn1, sn2)

intConf <- checkIntensityRules(intrules = c("lysopg_sn1/lysopg_sn1"),
    rates = c("2/1"), intrequired = c(T), nchains = 2, chainsComb, sn1, sn2)

confLevels

<table>
<thead>
<tr>
<th>level</th>
<th>Confidence Annotation Levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>order</td>
<td>Confidence annotation levels and their hierarchy.</td>
</tr>
<tr>
<td>Usage</td>
<td>data(&quot;confLevels&quot;)</td>
</tr>
<tr>
<td>Format</td>
<td>Data frame with 5 observations and 2 variables.</td>
</tr>
</tbody>
</table>

level character vector with the names of the annotation levels.
order numeric vector that indicates the hierarchichal order.
createLipidDB  

Customizable lipid DBs creator

Description
It allows to create easy-customizable lipid DBs for annotation with LipidMS package.

Usage
createLipidDB(lipid, chains, chains2)

Arguments

- **lipid**: character value indicating the class of lipid. See Details.
- **chains**: character vector indicating the FA chains to be employed
- **chains2**: character vector containing the sphingoid bases to be employed if required.

Details


Value
List with the requested dbs (data frames)

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

```r
sph <- c("16:0", "16:1", "18:0", "18:1")
newdb <- createLipidDB(lipid = "PC", chains = fas, chains2 = sph)
```
Cross the original MS1 peaklist with the annotation results.

**Usage**

crossTables(MS1, results, ppm = 10, rttol = 10, dbs)

**Arguments**

- **MS1**: data frame containing all peaks from the full MS function. It must have three columns: m.z, RT (in seconds) and int (intensity).
- **results**: data frame. Output of identification functions.
- **ppm**: mass tolerance in ppm.
- **rttol**: rt tolerance to match peaks in seconds.
- **dbs**: list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See `createLipidDB` and `assignDB`.

**Value**

Data frame with 6 columns: m.z, RT, int, LipidMS_id, adduct and confidence level for the annotation. When multiple IDs are proposed for the same feature, they are sorted based on the annotation level.

**Author(s)**

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

**Examples**

```r
library(LipidMSdata)
results <- idNEG(MS1 = MS1_neg, MSMS1 = MSMS1_neg, MSMS2 = MSMS2_neg)
crossTables(MS1_neg$peaklist, results = results$results,
             ppm = 10, rttol = 10)
```
dataProcessing

Process mzXML files: peakpicking and deisotoping

Description


Usage

dataProcessing(file, mslevel, polarity, dmzgap = 50, drtgap = 25,
ppm = TRUE, minpeak, maxint = 1e+09, dmzdens, drtdens = 20,
merged = FALSE, drtsmall, drtfill = 5, drttotal = 100,
recurs = 4, weight, SB, SN = 2, minint, ended = 2,
removeIsotopes = TRUE, rttolIso = 2, ppmIso = 20)

Arguments

file           path of the mzXML input file.
mslevel        numeric value indicating if data belongs to level 1 (fullMS) or level 2 (MS/MS).
polarity       character value: negative or positive.
dmzgap         enviPick parameter. 50 by default.
drtgap         enviPick parameter. 25 by default.
ppm            logical value. TRUE if dmzdens was set in ppm and FALSE if it was in as an absolute value. TRUE by default.
minpeak        minimum number of measurements required within the RT window of drtsmall. Optional. By default, 5 when mslevel = 1 and 4 when mslevel = 2.
maxint         EIC cluster with measurements above this intensity are kept, even if they do not fulfill minpeak. 1E9 by default.
dmzdens        maximum measurement deviation (+/-) of m/z from its mean within each EIC. Optional. By default, 15 when mslevel = 1 and 30 when mslevel = 2.
drtdens        RT tolerance for clustering. Optional. 20 by default.
merged         merge EIC cluster of comparable m/z. Logical. FALSE by default.
drtsmall       peak definition - RT window of a peak. Optional. By default, 100 when mslevel = 1 and 30 when mslevel = 2.
drtfill        maximum RT gap length to be filled. 5 by default.
drttotal       maximum RT length of a single peak. 100 by default.
recurs         maximum number of peaks within one EIC. 3 by default.
weight         weight for assigning measurements to a peak. Optional. By default, 1 when mslevel = 1 and 2 when mslevel = 2.
SB             signal-to-base ratio. Optional. By default, 3 when mslevel = 1 and 2 when mslevel = 2.
dataProcessing

SN signal-to-noise ratio. 2 by default.

minint minimum intensity of a peak. Optional. By default, 1000 when mslevel = 1 and 100 when mslevel = 2.

ended within the peak detection recursion set by argument recurs, how often can a peak detection fail to end the recursion?. 2 by default.

removeIsotopes logical. If TRUE, only isotopes identified as M+0, are kept when mslevel = 1, and M+0 or unknown when mslevel = 2. TRUE by default. If FALSE, an additional column is added to the peak list to inform about isotopes.

rttolIso numeric. Time windows for isotope matching.

ppmIso numeric. Mass tolerance for isotope matching.

Details

This function executes 2 steps: 1) peak-picking using enviPick package and 2) it searches isotopes using an adaptation of the CAMERA algorithm. If mslevel = 1 and remove isotopes is set as TRUE, only ions with more than 1 isotope are kept.

Value

List with two data frames: peaklist, with 4 columns (m.z, RT, int, and peakID) and rawScan, with all the scans information in 5 columns (m.z, RT, int, peakID and Scan). PeakID columns links both data frames: extracted peaks and raw data. The Scan column indicates the scan number (order) to which each row of the rawScans data frame belong.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

References

https://cran.r-project.org/web/packages/enviPick/index.html


Examples

dataProcessing("input_file.mzXML", mslevel = 1, polarity = "positive")
**dgdb**

*DGs database*

**Description**

In silico generated database for common DGs.

**Usage**

`data("dgdb")`

**Format**

Data frame with 147 observations and the following 3 variables.

- `formula` character vector containing molecular formulas.
- `total` character vector indicating the total number of carbons and double bounds of the chains.
- `Mass` numeric vector with the neutral masses.

---

**fadb**

*FAs database*

**Description**

In silico generated database for common FAs.

**Usage**

`data("fadb")`

**Format**

Data frame with 30 observations and the following 3 variables.

- `formula` character vector containing molecular formulas.
- `total` character vector indicating the total number of carbons and double bounds of the chains.
- `Mass` numeric vector with the neutral masses.
**fahfadb**

FAHFAs database

---

**Description**

In silico generated database for common FAHFAs.

**Usage**

```r
data("fahfadb")
```

**Format**

Data frame with 147 observations and the following 3 variables.

- `formula` character vector containing molecular formulas.
- `total` character vector indicating the total number of carbons and double bounds of the chains.
- `Mass` numeric vector with the neutral masses.

---

**findCandidates**

Search of lipid candidates of a certain class.

**Description**

Search of lipid candidates from a peaklist based on a set of expected adducts.

**Usage**

```r
findCandidates(MS1, db, ppm, rt, adducts, rttol = 3, dbs, 
rawData = data.frame(), coelCutoff = 0)
```

**Arguments**

- **MS1** peaklist of the MS function. Data frame with 3 columns: m.z, RT (in seconds) and int (intensity).
- **db** database (i.e. pcdb, dgdb, etc.). Data frame with at least 2 columns: Mass (exact mass) and total (total number of carbons and double bound of the FA chains, i.e. "34:1").
- **ppm** m/z tolerance in ppm.
- **rt** rt range where the function will look for candidates. By default, it will search within all RT range in MS1.
- **adducts** character vector containing the expected adducts to search for (i.e. "M+H", "M+Na", "M-H", etc.). See details.
- **rttol** rt tolerance in seconds to match adducts.
findCandidates

**dbs** list of data bases required for the annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be changed. If data bases have been customized using `createLipidDB`, they also have to be modified here.

**rawData** raw scans data. Output of `dataProcessing` function (MS1$rawData).

**coelCutoff** coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied.

**Details**

`findCandidates` looks for matches between the m/z of the MS1 peaklist and the expected m/z of the candidates in the database for each adduct. If several adducts are expected, results are combined.

Adducts allowed are contained in `adductsTable` data frame, which can be modified if required (see `adductsTable`).

**Value**

Data frame with the found candidates. It contains 6 columns: m.z, RT, int (from the peaklist data.frame), ppm, cb (total number of carbons and double bounds of the FA chains) and adducts.

**Author(s)**

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

**Examples**

```r
library(LipidMSdata)
dbs <- assignDB()
candidates <- findCandidates(MS1 = MS1_neg$peaklist,
  db = dbs$pgdb, ppm = 10, rt = c(0, 2000), adducts = c("M-H"),
  rttol = 10, rawData = MS1_neg$rawScans, coelCutoff = 0.8)

# If any adduct is not in the adductsTable, it can be added:
adductsTable2 <- rbind(adductsTable,
c(adduct = "M+HCOO", mdiff = 44.9982, n = 1, charge = -1))
dbs <- assignDB()
dbs$adductsTable <- adductsTable2
candidates <- findCandidates(MS1 = MS1_neg$peaklist,
  db = dbs$pgdb, ppm = 10, rt = c(0, 2000), adducts = c("M-H", "M+HCOO"),
  rttol = 10, rawData = MS1_neg$rawScans, coelCutoff = 0.8)
```
getInclusionList

Obtain an inclusion list from the annotation results

Description

Obtain an inclusion list from the annotation results.

Usage

getInclusionList(results, adductsTable = LipidMS::adductsTable)

Arguments

results data frame. Output of identification functions.
adductsTable data frame with the adducts allowed and their mass difference.

Value

Data frame with 6 columns: formula, RT, neutral mass, m/z, adduct and the compound name.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

library(LipidMSdata)
results <- idPOS(MS1_neg, MSMS1_neg, MSMS2_neg)
getInclusionList(results$results)

hfadb

HFAs database

Description

In silico generated database for common HFAs.

Usage

data("hfadb")
**Format**

Data frame with 30 observations and the following 3 variables.

*formula* character vector containing molecular formulas.

*total* character vector indicating the total number of carbons and double bounds of the chains.

*Mass* numeric vector with the neutral masses.

---

**idBAneg Bile Acids (BA) annotation for ESI-**

**Description**

BA identification based on fragmentation patterns for LC-MS/MS AIF data acquired in negative mode.

**Usage**

`idBAneg(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10, rttol = 3, rt, adducts = c("M-H"), conjfrag = c("baconj_M-H"), bafrag = c("ba_M-H-H2O", "ba_M-H-2H2O"), coelCutoff = 0.8, dbs)`

**Arguments**

**MS1**

list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

**MSMS1**

list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

**MSMS2**

list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

**ppm_precursor** mass tolerance for precursor ions. By default, 5 ppm.
**idBAneg**

**ppm_products**  mass tolerance for product ions. By default, 10 ppm.

**rttol**  total rt window for coelution between precursor and product ions. By default, 3 seconds.

**rt**  rt range where the function will look for candidates. By default, it will search within all RT range in MS1.

**adducts**  expected adducts for BA in ESI-. Adducts allowed can be modified in the adductsTable (dbs argument).

**conjfrag**  character vector containing the fragmentation rules for the BA-conjugates. By default just taurine and glycine are considered, but baconjdb can be modified to add more possible conjugates. See chainFrags for details. It can also be an empty vector.

**bafrag**  character vector containing the fragmentation rules for other BA fragments. See chainFrags for details. It can be an empty vector.

**coelCutoff**  coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.

**dbs**  list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See createLipidDB and assignDB.

**Details**

The idBAneg function involves 3 steps. 1) FullMS-based identification of candidate BA as M-H. 2) Search of BA-conjugate fragments if required. 3) Search of fragments coming from the loss of H2O.

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (MS-only if no rules are defined, or Subclass level if they are supported by fragments) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

**Value**

List with BA annotations (results) and some additional information (fragments).

**Note**

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

**Author(s)**

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>
idCarpos

Carnitine annotation for ESI+

Description

Carnitines identification based on fragmentation patterns for LC-MS/MS AIF data acquired in positive mode.

Usage

idCarpos(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10, rttol = 3, rt, adducts = c("M+H", "M+Na"), clfrags = c(60.0807, 85.0295, "fa_M+H-H2O"), clrequired = c(F, F, F), ftype = c("F", "F", "BB"), chainfrags_sn1 = c("fa_M+H-H2O"), coelCutoff = 0.8, dbs)

Arguments

MS1 list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS1 list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS2 list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

ppm_precursor mass tolerance for precursor ions. By default, 5 ppm.
idCarpos function involves 3 steps. 1) FullMS-based identification of candidate carnitines as M+H and M+Na. 2) Search of carnitine class fragments: 60.0807 and 85.0295 or its loss (FA as M+H-H20) coeluting with the precursor ion. 3) Search of specific fragments coming from the FA chain (FA as M+H-H2O).

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (in this case, as Carnitines only have one chain, only Subclass and FA level are possible) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

Value

list with Carnitine annotations (results) and some additional information (class fragments and chain fragments).

Note

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.
idCEpos

Author(s)
M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

```r
library(LipidMSdata)
idCarpos(MS1 = MS1_pos, MSMS1 = MSMS1_pos, MSMS2 = MSMS2_pos)
```

---

**idCEpos**  
*Cholesterol Esthers (CE) annotation for ESI+*

**Description**

CE identification based on fragmentation patterns for LC-MS/MS AIF data acquired in positive mode.

**Usage**

```r
idCEpos(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10, 
        rttol = 3, rt, adducts = c("2M+NH4", "2M+Na", "M+NH4", "M+Na"), 
        clfrags = c(369.3516, "fa_M+H-H2O"), clrequired = c(T, F), 
        ftype = c("F", "BB"), chainfrags_sn1 = c("fa_M+H-H2O"), 
        coelCutoff = 0.8, dbs)
```

**Arguments**

- **MS1**: list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

- **MSMS1**: list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

- **MSMS2**: list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). In case no coelution score needs to be applied, this argument can be just the peaklist data frame.
idCEpos function involves 3 steps. 1) FullMS-based identification of candidate CE as 2M+NH4, 2M+Na, M+NH4 and M+Na. 2) Search of CE class fragments: 369.3516 or its loss (FA as M+H-H2O) coeluting with the precursor ion. 3) Search of specific fragments that confirm chain composition (FA as M+H-H2O).

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (in this case, as CE only have one chain, only Subclass and FA level are possible) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

**Value**

list with CE annotations (results) and some additional information (class fragments and chain fragments).
Note

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

idCEpos(MS1 = MS1_pos, MSMS1 = MSMS1_pos, MSMS2 = MSMS2_pos)

idCerneg

Ceramides (Cer) annotation for ESI-

Description

Cer identification based on fragmentation patterns for LC-MS/MS AIF data acquired in negative mode.

Usage

idCerneg(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10, rttol = 3, rt, adducts = c("M-H", "M+CH3COO"), clfrags = c(), clrequired = c(), ftype = c(), chainfrags_sn1 = c("NL-nlsph_M-H", "sph_M-H-2H2O", "sph_M-H-H2O"), chainfrags_sn2 = c("fa_Mn-1.9918"), intrules = c(), rates = c(), intrequired = c(), coelCutoff = 0.8, dbs)

Arguments

MS1

list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS1

list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID...
"rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

`MSMS2` list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

`ppm_precursor` mass tolerance for precursor ions. By default, 5 ppm.

`ppm_products` mass tolerance for product ions. By default, 10 ppm.

`rttol` total rt window for coelution between precursor and product ions. By default, 3 seconds.

`rt` rt range where the function will look for candidates. By default, it will search within all RT range in MS1.

`adducts` expected adducts for Cer in ESI-. Adducts allowed can be modified in `adductsTable` (dbs argument).

`clfrags` vector containing the expected fragments for a given lipid class. See `checkClass` for details.

`clrequired` logical vector indicating if each class fragment is required or not. If any of them is required, at least one of them must be present within the coeluting fragments. See `checkClass` for details.

`ftype` character vector indicating the type of fragments in `clfrags`. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See `checkClass` for details.

`chainfrags_sn1` character vector containing the fragmentation rules for the chain fragments in sn1 position. See `chainFrags` for details.

`chainfrags_sn2` character vector containing the fragmentation rules for the chain fragments in sn2 position. See `chainFrags` for details. If empty, it will be estimated based on the difference between precursors and sn1 chains.

`intrules` character vector specifying the fragments to compare. See `checkIntensityRules`.

`rates` character vector with the expected rates between fragments given as a string (i.e. "3/1"). See `checkIntensityRules`.

`intrequired` logical vector indicating if any of the rules is required. If not, at least one must be verified to confirm the structure.

`coelCutoff` coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.

`dbs` list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See `createLipidDB` and `assignDB`. 
Details

The `idCerpos` function involves 5 steps. 1) FullMS-based identification of candidate Cer as M-H and M+CH3COO. 2) Search of Cer class fragments: there are no class fragment by default. 3) Search of specific fragments that inform about the sphingoid base (Sph as M-H-2H2O resulting from the loss of the FA chain or loss of part of the sphingoid base) and the FA chain (FA as M-H but with a N instead of an O, what means a mass difference of 1.9918 from the exact mass of the FA). 4) Look for possible chains structure based on the combination of chain fragments. 5) Check intensity rules to confirm chains position. In this case, there are no intensity rules by default.

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (Subclass, FA level, where chains are known but not their positions, or FA position level) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

Value

List with Cer annotations (results) and some additional information (class fragments and chain fragments).

Note

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

```r
library(LipidMSdata)
idCerpos(MS1 = MS1_neg, MSMS1 = MSMS1_neg, MSMS2 = MSMS2_neg)
```

---

Ceramides (Cer) annotation for ESI+

Description

Cer identification based on fragmentation patterns for LC-MS/MS AIF data acquired in positive mode.
idCerpos

Usage

idCerpos(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10,
rrtol = 3, rt, adducts = c("M+H-H2O", "M+Na", "M+H"),
c1frags = c(), c1required = c(), ftype = c(),
chainfrags_sn1 = c("sph_M+H-2H2O"), chainfrags_sn2 = c(""),
intrules = c(), rates = c(), intrequired = c(), coelCutoff = 0.8,
dbs)

Arguments

MS1 list with two data frames containing all peaks from the full MS function ("peak-list" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS1 list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS2 list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

ppm_precursor mass tolerance for precursor ions. By default, 5 ppm.

ppm_products mass tolerance for product ions. By default, 10 ppm.

rrtol total rt window for coelution between precursor and product ions. By default, 3 seconds.

rt rt range where the function will look for candidates. By default, it will search within all RT range in MS1.

adducts expected adducts for Cer in ESI+. Adducts allowed can be modified in adductsTable (dbs argument).

c1frags vector containing the expected fragments for a given lipid class. See checkClass for details.

c1required logical vector indicating if each class fragment is required or not. If any of them is required, at least one of them must be present within the coeluting fragments. See checkClass for details.

ftype character vector indicating the type of fragments in clfrags. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See checkClass for details.
idCerpos

chainfrags_sn1 character vector containing the fragmentation rules for the chain fragments in sn1 position. See chainFrags for details.

chainfrags_sn2 character vector containing the fragmentation rules for the chain fragments in sn2 position. See chainFrags for details. If empty, it will be estimated based on the difference between precursors and sn1 chains.

intrules character vector specifying the fragments to compare. See checkIntensityRules.

rates character vector with the expected rates between fragments given as a string (i.e. "3/1"). See checkIntensityRules.

intrequired logical vector indicating if any of the rules is required. If not, at least one must be verified to confirm the structure.

coeIlCutoff coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.

dbs list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See createLipidDB and assignDB.

Details

idCerpos function involves 5 steps. 1) FullMS-based identification of candidate Cer as M+H, M+H-H2O and M+Na. 2) Search of Cer class fragments: there isn’t any class fragment by default. 3) Search of specific fragments that inform about the sphingoid base (Sph as M+H-2H2O resulting from the loss of the FA chain) and the FA chain (by default it is calculated using the difference between precursor and sph fragments). 4) Look for possible chains structure based on the combination of chain fragments. 5) Check intensity rules to confirm chains position. In this case, there are no intensity rules by default.

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (Subclass, FA level, where chains are known but not their positions, or FA position level) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

Value

list with Cer annotations (results) and some additional information (class fragments and chain fragments).

Note

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>
Examples

```r
library(LipidMSdata)
idCerpos(MS1 = MS1_pos, MSMS1 = MSMS1_pos, MSMS2 = MSMS2_pos)
```

Description

CL identification based on fragmentation patterns for LC-MS/MS AIF data acquired in negative mode.

Usage

```r
idCLneg(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10,
       rttol = 5, rt, adducts = c("M-H", "M+Na-2H"), clfrags = c(),
       c1required = c(), ftype = c(),
       chainfrags_sn1 = c("lysopa_M-H-H2O"),
       chainfrags_sn2 = c("lysopa_M-H-H2O"),
       chainfrags_sn3 = c("lysopa_M-H-H2O"),
       chainfrags_sn4 = c("lysopa_M-H-H2O"), intrules = c("Unknown"),
       rates = c(), intrequired = c(), coelCutoff = 0.8, dbs)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MS1</td>
<td>List with two data frames containing all peaks from the full MS function (&quot;peaklist&quot; data frame) and the raw MS scans data (&quot;rawScans&quot; data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). &quot;rawScans&quot; data frame also needs an extra column named &quot;Scan&quot;, which indicates the scan order number. Output of <code>dataProcessing</code> function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.</td>
</tr>
<tr>
<td>MSMS1</td>
<td>List with two data frames containing all peaks from the high energy function (&quot;peaklist&quot; data frame) and the raw MS scans data (&quot;rawScans&quot; data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). &quot;rawScans&quot; data frame also needs an extra column named &quot;Scan&quot;, which indicates the scan order number. Output of <code>dataProcessing</code> function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.</td>
</tr>
<tr>
<td>MSMS2</td>
<td>List with two data frames containing all peaks from a second high energy function (&quot;peaklist&quot; data frame) and the raw MS scans data (&quot;rawScans&quot; data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID</td>
</tr>
</tbody>
</table>
(link between both data frames). "rawScans" data frame also needs a extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

- `ppm_precursor` mass tolerance for precursor ions. By default, 5 ppm.
- `ppm_products` mass tolerance for product ions. By default, 10 ppm.
- `rttol` total rt window for coelution between precursor and product ions. By default, 3 seconds.
- `rt` rt range where the function will look for candidates. By default, it will search within all RT range in MS1.
- `adducts` expected adducts for CL in ESI-. Adducts allowed can be modified in adductsTable (dbs argument).
- `clfrags` vector containing the expected fragments for a given lipid class. See `checkClass` for details.
- `clrequired` logical vector indicating if each class fragment is required or not. If any of them is required, at least one of them must be present within the coeluting fragments. See `checkClass` for details.
- `ftype` character vector indicating the type of fragments in clfrags. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See `checkClass` for details.
- `chainfrags_sn1` character vector containing the fragmentation rules for the chain fragments in sn1 position. See `chainFrags` for details.
- `chainfrags_sn2` character vector containing the fragmentation rules for the chain fragments in sn2 position. See `chainFrags` for details.
- `chainfrags_sn3` character vector containing the fragmentation rules for the chain fragments in sn3 position. See `chainFrags` for details.
- `chainfrags_sn4` character vector containing the fragmentation rules for the chain fragments in sn4 position. See `chainFrags` for details.
- `intrules` character vector specifying the fragments to compare. See `checkIntensityRules`. If some intensity rules should be employed to identify the chains position but they aren’t known yet, use "Unknown". If it isn’t required, leave an empty vector.
- `rates` character vector with the expected rates between fragments given as a string (i.e. "3/1"). See `checkIntensityRules`.
- `intrquired` logical vector indicating if any of the rules is required. If not, at least one must be verified to confirm the structure.
- `coelCutoff` coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.
- `dbs` list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See `createLipidDB` and `assignDB`.

**Details**

`idCLneg` function involves 5 steps. 1) FullMS-based identification of candidate CL as M-H or M-2H. 2) Search of CL class fragments: no class fragments are searched by defaults as they use to
have bad coelution scores. 3) Search of specific fragments that inform about chain composition at
sn1 (lysoPA as M-H-H2O), sn2 (lysoPA as M-H-H2O), sn3 (lysoPA as M-H-H2O) and sn4 (lysoPA
as M-H-H2O). 4) Look for possible chains structure based on the combination of chain fragments.
5) Check intensity rules to confirm chains position. For CL there are no intensity rules by default.
Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA
composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity,
which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (Subclass, FA
level, where chains are known but not their positions, or FA position level) and PFCS (parent-
fragment coelution score mean of all fragments used for the identification).

Value

List with CL annotations (results) and some additional information (class fragments and chain frag-
ments).

Note

Isotopes should be removed before identification to avoid false positives. This function has been
written based on fragmentation patterns observed for two different platforms (QTOF 6550 from
Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or
acquisition settings.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

library(LipidMSdata)
idCL(MS1 = MS1_neg, MSMS1 = MSMS1_neg, MSMS2 = MSMS2_neg, coelCutoff = 0)

idDGpos

Diacylglycerols (DG) annotation for ESI+

Description

DG identification based on fragmentation patterns for LC-MS/MS AIF data acquired in positive
mode.

Usage

idDGpos(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10,
rttol = 3, rt, adducts = c("M+H-H2O", "M+NH4", "M+Na"),
clfrags = c(), clrequired = c(), ftype = c(),
chainfrags_sn1 = c("mg_M+H-H2O"), chainfrags_sn2 = c("mg_M+H-H2O"),
intrules = c("mg_sn1/mg_sn2"), rates = c("1"), intrequired = c(T),
coelCutoff = 0.8, dbs)
Arguments

**MS1**
List with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

**MSMS1**
List with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

**MSMS2**
List with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

**ppm_precursor**
Mass tolerance for precursor ions. By default, 5 ppm.

**ppm_products**
Mass tolerance for product ions. By default, 10 ppm.

**rttol**
Total rt window for coelution between precursor and product ions. By default, 3 seconds.

**rt**
Rt range where the function will look for candidates. By default, it will search within all RT range in MS1.

**adducts**
Expected adducts for DG in ESI+. Adducts allowed can be modified in adductsTable (dbs argument).

**clfrags**
Vector containing the expected fragments for a given lipid class. See checkClass for details.

**clrequired**
Logical vector indicating if each class fragment is required or not. If any of them is required, at least one of them must be present within the coeluting fragments. See checkClass for details.

**ftype**
Character vector indicating the type of fragments in clfrags. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See checkClass for details.

**chainfrags_sn1**
Character vector containing the fragmentation rules for the chain fragments in sn1 position. See chainFrags for details.

**chainfrags_sn2**
Character vector containing the fragmentation rules for the chain fragments in sn2 position. See chainFrags for details. If empty, it will be estimated based on the difference between precursors and sn1 chains.

**intrules**
Character vector specifying the fragments to compare. See checkIntensityRules.
idDGpos

rates character vector with the expected rates between fragments given as a string (i.e. "3/1"). See checkIntensityRules.

intrequired logical vector indicating if any of the rules is required. If not, at least one must be verified to confirm the structure.

coeIcutoff coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.

dbs list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See createLipidDB and assignDB.

Details

idDGpos function involves 5 steps. 1) FullMS-based identification of candidate DG as M+H-H2O, M+NH4 and M+Na. 2) Search of DG class fragments: there are no class fragment by default. 3) Search of specific fragments that inform about the FA chains (MGs as M+H-H2O resulting from the loss of the FA chains). 4) Look for possible chains structure based on the combination of chain fragments. 5) Check intensity rules to confirm chains position: MG coming from the loss of the sn2 chain is more intense than the one coming from the loss of sn1.

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (Subclass, FA level, where chains are known but not their positions, or FA position level) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

Value

List with DG annotations (results) and some additional information (class fragments and chain fragments).

Note

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

library(LipidMSdata)
idDGpos(MS1 = MS1_pos, MSMS1 = MSMS1_pos, MSMS2 = MSMS2_pos)
idFAHFAneg

FAHFA annotation for ESI-

Description
FAHFA identification based on fragmentation patterns for LC-MS/MS AIF data acquired in negative mode.

Usage
idFAHFAneg(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10,
rttol = 3, rt, adducts = c("M-H"), clfrags = c(),
clrequired = c(), ftype = c(), chainfrags_sn1 = c("hfa_M-H"),
chainfrags_sn2 = c("fa_M-H"), intrules = c("hfa_sn1/fa_sn2"),
rates = c("3/1"), intrequired = c(T), coelCutoff = 0.8, dbs)

Arguments
MS1 list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS1 list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS2 list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

ppm_precursor mass tolerance for precursor ions. By default, 5 ppm.

ppm_products mass tolerance for product ions. By default, 10 ppm.

rttol total rt window for coelution between precursor and product ions. By default, 3 seconds.

rt rt range where the function will look for candidates. By default, it will search within all RT range in MS1.
adducts: expected adducts for FAHFA in ESI-. Adducts allowed can be modified in adductsTable (dbs argument).

c1frags: vector containing the expected fragments for a given lipid class. See checkClass for details.

c1required: logical vector indicating if each class fragment is required or not. If any of them is required, at least one of them must be present within the coeluting fragments. See checkClass for details.

ftype: character vector indicating the type of fragments in c1frags. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See checkClass for details.

chainfrags_sn1: character vector containing the fragmentation rules for the chain fragments in sn1 position. See chainFrags for details.

chainfrags_sn2: character vector containing the fragmentation rules for the chain fragments in sn2 position. See chainFrags for details. If empty, it will be estimated based on the difference between precursors and sn1 chains.

intrules: character vector specifying the fragments to compare. See checkIntensityRules.

rates: character vector with the expected rates between fragments given as a string (i.e. "3/1"). See checkIntensityRules.

intrrequired: logical vector indicating if any of the rules is required. If not, at least one must be verified to confirm the structure.

coe1Cutoff: coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.

dbs: list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See createLipidDB and assignDB.

Details

idFAHFAneg function involves 5 steps. 1) FullMS-based identification of candidate FAHFA as M-H. 2) Search of FAHFA class fragments: there is’t any class fragment by default. 3) Search of specific fragments that inform about chain composition in sn1 (HFA as M-H resulting from the loss of the FA chain) and sn2 (FA chain as M-H). 4) Look for possible chains structure based on the combination of chain fragments. 5) Check intensity rules to confirm chains position. In this case, HFA intensity has to be higher than FA.

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (Subclass, FA level, where chains are known but not their positions, or FA position level) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

Value

list with FAHFA annotations (results) and some additional information (class fragments and chain fragments).
Note
Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

Author(s)
M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

```
idFAHFAneg(MS1 = MS1_neg, MSMS1 = MSMS1_neg, MSMS2 = MSMS2_neg)
```

```
Fatty Acids (FA) annotation for ESI-
```

Description
FA identification based on fragmentation patterns for LC-MS/MS AIF data acquired in negative mode.

Usage

```
idFAneg(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10, rttol = 3, rt, adducts = c("M-H", "2M-H"), clfrags = c("fa_M-H", "fa_M-H-H2O"), clrequired = c(F, F), ftype = c("BB", "BB"), coelCutoff = 0.8, dbs)
```

Arguments

- **MS1**: list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

- **MSMS1**: list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.
MSMS2 list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

ppm_precursor mass tolerance for precursor ions. By default, 5 ppm.
ppm_products mass tolerance for product ions. By default, 10 ppm.
rttol total rt window for coelution between precursor and product ions. By default, 3 seconds.
rt rt range where the function will look for candidates. By default, it will search within all RT range in MS1.
adducts expected adducts for FA in ESI-. Adducts allowed can be modified in adductsTable (dbs argument).
c1frags vector containing the expected fragments for a given lipid class. See checkClass for details.
c1required logical vector indicating if each class fragment is required or not. If any of them is required, at least one of them must be present within the coeluting fragments. See checkClass for details.
ftype character vector indicating the type of fragments in c1frags. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See checkClass for details.
coelCutoff coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.
dbs list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See createLipidDB and assignDB.

Details

idfAneg function involves 2 steps. 1) FullMS-based identification of candidate FA as M-H or 2M-H. 2) Search of FA class fragments: neutral loss of H2O coeluting with the precursor ion or the molecular ion.

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), m/z, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (in this case, just MS-only or Subclass level (if any class fragment is defined) are possible) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

Value

List with FA annotations (results) and some additional information (class fragments and chain fragments).
Note

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

library(LipidMSdata)
idFAneg(MS1 = MS1_neg, MSMS1 = MSMS1_neg, MSMS2 = MSMS2_neg)

---

**idLPCneg**

Lysophosphocholines (LPC) annotation for ESI-

Description

LPC identification based on fragmentation patterns for LC-MS/MS AIF data acquired in negative mode.

Usage

```
library(LipidMSdata)
idLPCneg(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10, 
        rttol = 3, rt, adducts = c("M+CH3COO", "M-CH3", "M+CH3COO-CH3"),
        clfrags = c(168.0426, 224.0688, "lysopa_M-H", "lysopc_M-CH3"),
        clrequired = c(F, F, F, F), ftype = c("F", "F", "BB", "BB"),
        chainfrags_sn1 = c("fa_M-H"), coelCutoff = 0.8, dbs)
```

Arguments

- **MS1**: list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

- **MSMS1**: list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID
idLPCneg function involves 3 steps. 1) FullMS-based identification of candidate LPC as M+CH3COO, M-CH3 and M+CH3COO-CH3. To avoid incorrect annotations of PE as PC, candidates which are present just as M-CH3 will be ignored. 2) Search of LPC class fragments: 168.0426, 224.0688, lysoPA as M-H or lysoPC as M-CH3 coeluting with the precursor ion. 3) Search of specific fragments that confirm chain composition (FA as M-H).

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (in this case, as LPC only have one chain, only Subclass and FA level are possible) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).
idLPCpos

Value
list with LPC annotations (results) and some additional information (class fragments and chain fragments).

Note
Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

Author(s)
M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

```r
library(LipidMSdata)
idLPCneg(MS1 = MS1_neg, MSMS1 = MSMS1_neg, MSMS2 = MSMS2_neg)
```

---

idLPCpos  
Lysosphosphocolines (LPC) annotation for ESI+

Description
LPC identification based on fragmentation patterns for LC-MS/MS AIF data acquired in positive mode.

Usage
```
idLPCpos(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10, rttol = 3, rt, adducts = c("M+H", "M+Na"), clfrags = c(104.1075, 184.0739), cIrequired = c(F, F), ftype = c("F", "F"), chainfrags_sn1 = c("ng_M+H-H2O"), coelCutoff = 0.8, dbs)
```

Arguments
- **MS1** list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.
**MSMS1** list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of **dataProcessing** function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

**MSMS2** list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of **dataProcessing** function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

**ppm_precursor** mass tolerance for precursor ions. By default, 5 ppm.

**ppm_products** mass tolerance for product ions. By default, 10 ppm.

**rttol** total rt window for coelution between precursor and product ions. By default, 3 seconds.

**rt** rt range where the function will look for candidates. By default, it will search within all RT range in MS1.

**adducts** expected adducts for LPC in ESI+. Adducts allowed can be modified in adductsTable (dbs argument).

**clfrags** vector containing the expected fragments for a given lipid class. See **checkClass** for details.

**clrequired** logical vector indicating if each class fragment is required or not. If any of them is required, at least one of them must be present within the coeluting fragments. See **checkClass** for details.

**ftype** character vector indicating the type of fragments in clfrags. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See **checkClass** for details.

**chainfrags_sn1** character vector containing the fragmentation rules for the chain fragments. See **chainFrags** for details.

**coelCutoff** coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.

**dbs** list of databases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See **createLipidDB** and **assignDB**.

**Details**

**idLPCpos** function involves 3 steps. 1) FullMS-based identification of candidate LPC as M+H and M+Na. 2) Search of LPC class fragments: 104.1075 and 184.0739 coeluting with the precursor ion. 3) Search of specific fragments that confirm chain composition (MG as M+H-H2O).

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), m.z, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (in this case, as
LPC only have one chain, only Subclass and FA level are possible) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

Value

list with LPC annotations (results) and some additional information (class fragments and chain fragments).

Note

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

```
library(LipidMSdata)
idLPCpos(MS1 = MS1_pos, MSMS1 = MSMS1_pos, MSMS2 = MSMS2_pos)
```

```
idLPEneg(MS1 = MS1, MSMS1 = MSMS1, MSMS2 = MSMS2, ppm_precursor = 5, ppm_products = 10, rttol = 3, rt, adducts = c("M-H"), clfrags = c(140.0115, 196.038, 214.048, "lysope_M-CH3"), clrequired = c(F, F, F, "excluding"), ftype = c("F", "F", "F", "BB"), chainfrags_sn1 = c("fa_M-H"), coelCutoff = 0.8, dbs)
```

Description

LPE identification based on fragmentation patterns for LC-MS/MS AIF data acquired in negative mode.

Usage

```
idLPEneg(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10, rttol = 3, rt, adducts = c("M-H"), clfrags = c(140.0115, 196.038, 214.048, "lysope_M-CH3"), clrequired = c(F, F, F, "excluding"), ftype = c("F", "F", "F", "BB"), chainfrags_sn1 = c("fa_M-H"), coelCutoff = 0.8, dbs)
```
**Arguments**

**MS1**
list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

**MSMS1**
list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

**MSMS2**
list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

**ppm_precursor**
mass tolerance for precursor ions. By default, 5 ppm.

**ppm_products**
mass tolerance for product ions. By default, 10 ppm.

**rttol**
total RT window for coelution between precursor and product ions. By default, 3 seconds.

**rt**
RT range where the function will look for candidates. By default, it will search within all RT range in MS1.

**adducts**
expected adducts for LPE in ESI-. Adducts allowed can be modified in adductsTable (dbs argument).

**clfrags**
vector containing the expected fragments for a given lipid class. See checkClass for details.

**clrequired**
logical vector indicating if each class fragment is required or not. If any of them is required, at least one of them must be present within the coeluting fragments. See checkClass for details.

**ftype**
character vector indicating the type of fragments in clfrags. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See checkClass for details.

**chainfrags_sn1**
character vector containing the fragmentation rules for the chain fragments. See chainFrags for details.

**coelCutoff**
coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.

**dbs**
list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See createLipidDB and assignDB.
Details

`idLPEneg` function involves 3 steps. 1) FullMS-based identification of candidate LPE as M-H. 2) Search of LPE class fragments: 140.0115, 196.038 and 214.048 coeluting with the precursor ion. If a loss of CH3 group is found coeluting with any candidate, this will be excluded as it is a characteristic fragment of LPC. 3) Search of specific fragments that confirm chain composition (FA as M-H).

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (in this case, as LPE only have one chain, only Subclass and FA level are possible) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

Value

List with LPE annotations (results) and some additional information (class fragments and chain fragments).

Note

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

```r
library(LipidMSdata)
idLPEneg(MS1 = MS1_neg, MSMS1 = MSMS1_neg, MSMS2 = MSMS2_neg)
```

<table>
<thead>
<tr>
<th><code>idLPEpos</code></th>
<th>Lysosphoethanolamines (LPE) annotation for ESI+</th>
</tr>
</thead>
</table>

Description

LPE identification based on fragmentation patterns for LC-MS/MS AIF data acquired in positive mode.
idLPEpos

Usage

idLPEpos(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10,
rttol = 3, rt, adducts = c("M+H", "M+Na"), clfrags = c(141.01909),
clrequired = c(F), ftype = c("NL"),
chainfrags_sn1 = c("mg_M+H-H2O"), coelCutoff = 0.8, dbs)

Arguments

MS1 list with two data frames containing all peaks from the full MS function ("peak-
list" data frame) and the raw MS scans data ("rawScans" data frame). They
must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link
between both data frames). "rawScans" data frame also needs a extra column
named "Scan", which indicates the scan order number. Output of dataProcess-
ing function. In case no coelution score needs to be applied, this argument can
be just the peaklist data frame.

MSMS1 list with two data frames containing all peaks from the high energy function
("peaklist" data frame) and the raw MS scans data ("rawScans" data frame).
They must have four columns: m.z, RT (in seconds), int (intensity) and peakID
(link between both data frames). "rawScans" data frame also needs a extra col-
umn named "Scan", which indicates the scan order number. Output of dataPro-
cessing function. In case no coelution score needs to be applied, this argument can
be just the peaklist data frame.

MSMS2 list with two data frames containing all peaks from a second high energy func-
tion ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame).
They must have four columns: m.z, RT (in seconds), int (intensity) and peakID
(link between both data frames). "rawScans" data frame also needs a extra col-
umn named "Scan", which indicates the scan order number. Output of dataPro-
cessing function. In case no coelution score needs to be applied, this argument can
be just the peaklist data frame. Optional.

ppm_precursor mass tolerance for precursor ions. By default, 5 ppm.

ppm_products mass tolerance for product ions. By default, 10 ppm.

rttol total rt window for coelution between precursor and product ions. By default, 3
seconds.

rt rt range where the function will look for candidates. By default, it will search
within all RT range in MS1.

adducts expected adducts for LPE in ESI+. Adducts allowed can be modified in adducts-
Table (dbs argument).

clfrags vector containing the expected fragments for a given lipid class. See checkClass
for details.

clrequired logical vector indicating if each class fragment is required or not. If any of them
is required, at least one of them must be present within the coeluting fragments.
See checkClass for details.

ftype character vector indicating the type of fragments in clfrags. It can be: "F" (frag-
ment), "NL" (neutral loss) or "BB" (building block). See checkClass for details.
chainfrags_sn1 character vector containing the fragmentation rules for the chain fragments. See chainFrags for details.

coe1Cutoff coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.

dbs list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See createLipidDB and assignDB.

Details

idLPEpos function involves 3 steps. 1) FullMS-based identification of candidate LPE as M+H and M+Na. 2) Search of LPE class fragments: neutral loss of 141.01909 coeluting with the precursor ion. 3) Search of specific fragments that confirm chain composition in sn1 (MG as M+H-H2O).

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (in this case, as LPE only have one chain, only Subclass and FA level are possible) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

Value

List with LPE annotations (results) and some additional information (class fragments and chain fragments).

Note

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

library(LipidMSdata)
idLPEpos(MS1 = MS1_pos, MSMS1 = MSMS1_pos, MSMS2 = MSMS2_pos)
Lysophosphoglycerols (LPG) annotation for ESI-

Description

LPG identification based on fragmentation patterns for LC-MS/MS AIF data acquired in negative mode.

Usage

```r
idlPGneg(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10,
        rttol = 3, rt, adducts = c("M-H"), clfrags = c(152.9958, 227.0326,
        209.022, 74.0359), clrequired = c(F, F, F, F), ftype = c("F", "F",
        "F", "NL"), chainfrags_sn1 = c("fa_M-H"), coelCutoff = 0.8, dbs)
```

Arguments

- **MS1**
  - list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

- **MSMS1**
  - list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

- **MSMS2**
  - list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

- **ppm_precursor**
  - mass tolerance for precursor ions. By default, 5 ppm.

- **ppm_products**
  - mass tolerance for product ions. By default, 10 ppm.

- **rttol**
  - total rt window for coelution between precursor and product ions. By default, 3 seconds.

- **rt**
  - rt range where the function will look for candidates. By default, it will search within all RT range in MS1.
idLPGneg

**adducts**
expected adducts for LPG in ESI-. Adducts allowed can be modified in adductsTable (dbs argument).

**clfrags**
vector containing the expected fragments for a given lipid class. See checkClass for details.

**clrequired**
logical vector indicating if each class fragment is required or not. If any of them is required, at least one of them must be present within the coeluting fragments. See checkClass for details.

**ftype**
character vector indicating the type of fragments in clfrags. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See checkClass for details.

**chainfrags_sn1**
character vector containing the fragmentation rules for the chain fragments. See chainFrags for details.

**coelCutoff**
coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.

**dbs**
list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See createLipidDB and assignDB.

**Details**

idLPGneg function involves 3 steps. 1) FullMS-based identification of candidate LPG as M-H. 2) Search of LPG class fragments: 152.9958, 227.0326, 209.022 and neutral loss of 74.0359 coeluting with the precursor ion. 3) Search of specific fragments that confirm chain composition (FA as M-H).

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (in this case, as LPG only have one chain, only Subclass and FA level are possible) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

**Value**

List with LPG annotations (results) and some additional information (class fragments and chain fragments).

**Note**

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

**Author(s)**

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>
idLPIneg

Examples

```r
library(LipidMSdata)
idLPNeg(MS1 = MS1_neg, MSMS1 = MSMS1_neg, MSMS2 = MSMS2_neg)
```

idLPIneg  

Lysophosphoinositols (LPI) annotation for ESI-

Description

LPI identification based on fragmentation patterns for LC-MS/MS AIF data acquired in negative mode.

Usage

```r
idLPNeg(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10, 
        rttol = 3, rt, adducts = c("M-H"), clfrags = c(241.0115, 223.0008, 
        259.0219, 297.0375), clrequired = c(F, F, F, F), ftype = c("F", "F", "F", "F"), 
        chainfrags_sn1 = c("fa_M-H"), coelCutoff = 0.8, dbs)
```

Arguments

- **MS1**
  - list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

- **MSMS1**
  - list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

- **MSMS2**
  - list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

- **ppm_precursor**
  - mass tolerance for precursor ions. By default, 5 ppm.
idLPIneg

ppm_products mass tolerance for product ions. By default, 10 ppm.

rttol total rt window for coelution between precursor and product ions. By default, 3 seconds.

rt rt range where the function will look for candidates. By default, it will search within all RT range in MS1.

adducts expected adducts for LPI in ESI-. Adducts allowed can be modified in adductsTable (dbs argument).

c1frags vector containing the expected fragments for a given lipid class. See checkClass for details.

clrequired logical vector indicating if each class fragment is required or not. If any of them is required, at least one of them must be present within the coeluting fragments. See checkClass for details.

ftype character vector indicating the type of fragments in clfrags. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See checkClass for details.

chainfrags_sn1 character vector containing the fragmentation rules for the chain fragments. See chainFrags for details.

coeCutoff coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.

dbs list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See createLipidDB and assignDB.

Details

idLPIneg function involves 3 steps. 1) FullMS-based identification of candidate LPI as M-H. 2) Search of LPI class fragments: 241.0115, 223.0008, 259.0219 and 297.0375 coeluting with the precursor ion. 3) Search of specific fragments that confirm chain composition (FA as M-H).

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (in this case, as LPI only have one chain, only Subclass and FA level are possible) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

Value

List with LPI annotations (results) and some additional information (class fragments and chain fragments).

Note

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.
Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

library(LipidMSdata)
idLPlines(MS1 = MS1_neg, MSMS1 = MSMS1_neg, MSMS2 = MSMS2_neg)

Description

LPS identification based on fragmentation patterns for LC-MS/MS AIF data acquired in negative mode.

Usage

idLPSneg(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10, rttol = 3, rt, adducts = c("M-H", "M+Na-2H"), clfrags = c(87.032), clrequired = c(F), ftype = c("NL"), chainfrags_sn1 = c("fa_M-H"), coelCutoff = 0.8, dbs)

Arguments

MS1 list with two data frames cointaining all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs a extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS1 list with two data frames cointaining all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs a extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS2 list with two data frames cointaining all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID
idLPSneg function involves 3 steps. 1) FullMS-based identification of candidate LPS as M-H and M+Na-2H. 2) Search of LPS class fragments: neutral loss of 87.032 coeluting with the precursor ion. 3) Search of specific fragments that confirm chain composition (FA as M-H).

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (in this case, as LPS only have one chain, only Subclass and FA level are possible) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

Value

List with LPS annotations (results) and some additional information (class fragments and chain fragments).
Note

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

```r
library(LipidMSdata)
idLPSneg(MS1 = MS1_neg, MSMS1 = MSMS1_neg, MSMS2 = MSMS2_neg)
```

---

**idMGpos**  
*Monoacylglycerol (MG) annotation for ESI+

Description

MG identification based on fragmentation patterns for LC-MS/MS AIF data acquired in positive mode.

Usage

```r
idMGpos(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10,
        rttol = 3, rt, adducts = c("M+H-H2O", "M+NH4", "M+Na"),
        clfrags = c(), clrequired = c(), ftype = c(), coelCutoff = 0.8,
        dbs)
```

Arguments

- **MS1**
  list with two data frames cointaining all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs a extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

- **MSMS1**
  list with two data frames cointaining all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID
idMGpos

(idMGpos) function involves 2 steps. 1) FullMS-based identification of candidate MG as M+H-H2O, M+NH4 and M+Na. 2) Search of MG class fragments if any is assigned.

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input). Adducts, ppm (m/z error), confidenceLevel (in this case, just MS-only or Subclass level (if any class fragment is defined) are possible) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

Value

List with MG annotations (results) and some additional information (class fragments and chain fragments).
Note

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

```r
library(LipidMSdata)
idMGpos(MS1 = MS1_pos, MSMS1 = MSMS1_pos, MSMS2 = MSMS2_pos)
```

---

**idNEG**

*Lipids annotation for ESI-

**Description**

Lipids annotation based on fragmentation patterns for LC-MS/MS all-ions data acquired in negative mode. This function compiles all functions written for ESI- annotations.

**Usage**

```r
idNEG(MS1, MSMS1, MSMS2, ppm_precursor = 10, ppm_products = 10, rttol = 10, coelCutoff = 0.8, dbs)
```

**Arguments**

- **MS1** list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs a extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

- **MSMS1** list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs a extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.
idPCneg

**MSMS2** list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

**ppm_precursor** mass tolerance for precursor ions. By default, 5 ppm.

**ppm_products** mass tolerance for product ions. By default, 10 ppm.

**rttol** total rt window for coelution between precursor and product ions. By default, 3 seconds.

**coelCutoff** coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.

**dbs** list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See `createLipidDB` and `assignDB`.

**Value**

The output is a list with 2 elements: 1) a data frame that shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), m/z, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (Subclass, FA level, where chains are known but not their positions, or FA position level) and PFCS (parent-fragment coelution score mean of all fragments used for the identification); and 2) the original MS1 peaklist with the annotations on it.

**Author(s)**

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

**Examples**

```r
library(LipidMSdata)
idNEG(MS1 = MS1_neg, MSMS1 = MSMS1_neg, MSMS2 = MSMS2_neg)
```

---

**idPCneg** *Phosphocholines (PC) annotation for ESI-

**Description**

PC identification based on fragmentation patterns for LC-MS/MS AIF data acquired in negative mode.
idPCneg

Usage

idPCneg(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10, rttol = 3, rt, adducts = c("M+CH3COO", "M-CH3", "M+CH3COO-CH3"), clfrags = c(168.0426, 224.0688, "pc_M-CH3"), clrequired = c(F, F, F), ftype = c("F", "F", "BB"), chainfrags_sn1 = c("lysopc_M-CH3"), chainfrags_sn2 = c("fa_M-H", "lysopc_M-CH3"), intrules = c("lysopc_sn1/lysopc_sn2"), rates = c("3/1"), intrequired = c(T), coelCutoff = 0.8, dbs)

Arguments

MS1

list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS1

list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS2

list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

ppm_precursor

mass tolerance for precursor ions. By default, 5 ppm.

ppm_products

mass tolerance for product ions. By default, 10 ppm.

rttol

total rt window for coelution between precursor and product ions. By default, 3 seconds.

rt

rt range where the function will look for candidates. By default, it will search within all RT range in MS1.

adducts

expected adducts for PC in ESI-. Adducts allowed can be modified in adductsTable (dbs argument).

clfrags

vector containing the expected fragments for a given lipid class. See checkClass for details.

clrequired

logical vector indicating if each class fragment is required or not. If any of them is required, at least one of them must be present within the coeluting fragments. See checkClass for details.
idPCneg

**ftype** character vector indicating the type of fragments in clfrags. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See checkClass for details.

**chainfrags_sn1** character vector containing the fragmentation rules for the chain fragments in sn1 position. See chainFrags for details.

**chainfrags_sn2** character vector containing the fragmentation rules for the chain fragments in sn2 position. See chainFrags for details. If empty, it will be estimated based on the difference between precursors and sn1 chains.

**intrules** character vector specifying the fragments to compare. See checkIntensityRules.

**rates** character vector with the expected rates between fragments given as a string (i.e. "3/1"). See checkIntensityRules.

**intrequired** logical vector indicating if any of the rules is required. If not, at least one must be verified to confirm the structure.

**coelCutoff** coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.

**dbs** list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See createLipidDB and assignDB.

**Details**

idPCneg function involves 5 steps. 1) FullMS-based identification of candidate PC as M+CH3COO, M-CH3 or M+CH3COO-CH3. To avoid incorrect annotations of PE as PC, candidates which are present just as M-CH3 will be ignored. 2) Search of PC class fragments: 168.0426, 224.0688 or loss of CH3 coeluting with the precursor ion. 3) Search of specific fragments that inform about chain composition in sn1 (lysoPC as M-CH3 resulting from the loss of the FA chain at sn2) and sn2 (lysoPC as M-CH3 resulting from the loss of sn1 or FA as M-H). 4) Look for possible chains structure based on the combination of chain fragments. 5) Check intensity rules to confirm chains position. In this case, lysoPC from sn1 is at least 3 times more intense than lysoPC from sn2.

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (Subclass, FA level, where chains are known but not their positions, or FA position level) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

**Value**

List with PC annotations (results) and some additional information (class fragments and chain fragments).

**Note**

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.
idPCpos

Author(s)
M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

library(LipidMSdata)
idPCneg(MS1 = MS1_neg, MSMS1 = MSMS1_neg, MSMS2 = MSMS2_neg)

idPCpos
Phosphocholines (PC) annotation for ESI+

Description
PC identification based on fragmentation patterns for LC-MS/MS AIF data acquired in positive mode.

Usage

idPCpos(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10, rttol = 3, rt, adducts = c("M+H", "M+Na"), clfrags = c(104.1075, 184.0739, 183.06604), clrequired = c(F, F, F), ftype = c("F", "F", "NL"), chainfrags_sn1 = c("lysopc_M+H", "lysopc_M+H-H2O"), chainfrags_sn2 = c("lysopc_M+H", "lysopc_M+H-H2O", ""), intrules = c("lysopc_sn1/lysopc_sn2"), rates = c("2/1"), intrequired = c(T), coelCutoff = 0.8, dbs)

Arguments

MS1
list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS1
list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.
idPCpos

MSMS2 list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

ppm_precursor mass tolerance for precursor ions. By default, 5 ppm.

ppm_products mass tolerance for product ions. By default, 10 ppm.

rttol total rt window for coelution between precursor and product ions. By default, 3 seconds.

rt rt range where the function will look for candidates. By default, it will search within all RT range in MS1.

adducts expected adducts for PC in ESI+. Adducts allowed can be modified in adductsTable (dbs argument).

clfrags vector containing the expected fragments for a given lipid class. See checkClass for details.

clrequired logical vector indicating if each class fragment is required or not. If any of them is required, at least one of them must be present within the coeluting fragments. See checkClass for details.

ftype character vector indicating the type of fragments in clfrags. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See checkClass for details.

chainfrags_sn1 character vector containing the fragmentation rules for the chain fragments in sn1 position. See chainFrags for details.

chainfrags_sn2 character vector containing the fragmentation rules for the chain fragments in sn2 position. See chainFrags for details. If empty, it will be estimated based on the difference between precursors and sn1 chains.

intrules character vector specifying the fragments to compare. See checkIntensityRules.

rates character vector with the expected rates between fragments given as a string (i.e. "3/1"). See checkIntensityRules.

intrequired logical vector indicating if any of the rules is required. If not, at least one must be verified to confirm the structure.

coeLCutoff coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.

dbs list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See createLipidDB and assignDB.

Details

idPCpos function involves 5 steps. 1) FullMS-based identification of candidate PC as M+H and M+Na. 2) Search of PC class fragments: 104.1075, 184.0739 and neutral loss of 183.06604 coeluting with the precursor ion. 3) Search of specific fragments that inform about chain composition in sn1 (lysoPC as M+H or M+H-H2O resulting from the loss of the FA chain at sn2) and sn2 (lysoPC
as M+H or M+H-H2O resulting from the loss of the FA chain at sn1 or the difference between precursor and sn1 chain fragments). 4) Look for possible chains structure based on the combination of chain fragments. 5) Check intensity rules to confirm chains position. In this case, lysoPC from sn1 is at least twice more intense than lysoPC from sn2.

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (Subclass, FA level, where chains are known but not their positions, or FA position level) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

Value

List with PC annotations (results) and some additional information (class fragments and chain fragments).

Note

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

library(LipidMSdata)
idPCpos(MS1 = MS1_pos, MSMS1 = MSMS1_pos, MSMS2 = MSMS2_pos)

---

**idPEneg**

*Phosphoethanolamines (PE) annotation for ESI-

Description

PE identification based on fragmentation patterns for LC-MS/MS AIF data acquired in negative mode.
Usage

```r
idPEneg(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10, rttol = 5, rt, adducts = c("M-H"), c1frags = c(140.0118, 196.038, 214.048, "pe_M-CH3"), c1required = c(F, F, F, "excluding"), ftype = c("F", "F", "F", "BB"), chainfrags_sn1 = c("lysope_M-H"), chainfrags_sn2 = c("lysope_M-H", "fa_M-H"), intrules = c("lysope_sn1/lysope_sn2"), rates = c("3/1"), intrequired = c(T), coelCutoff = 0.8, dbs)
```

Arguments

**MS1**
- list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

**MSMS1**
- list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

**MSMS2**
- list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

**ppm_precursor**
- mass tolerance for precursor ions. By default, 5 ppm.

**ppm_products**
- mass tolerance for product ions. By default, 10 ppm.

**rttol**
- total rt window for coelution between precursor and product ions. By default, 3 seconds.

**rt**
- rt range where the function will look for candidates. By default, it will search within all RT range in MS1.

**adducts**
- expected adducts for PE in ESI-. Adducts allowed can be modified in `adductsTable` (dbs argument).

**c1frags**
- vector containing the expected fragments for a given lipid class. See `checkClass` for details.

**c1required**
- logical vector indicating if each class fragment is required or not. If any of them is required, at least one of them must be present within the coeluting fragments. See `checkClass` for details.
idPEneg

**ftype** character vector indicating the type of fragments in clfrags. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See checkClass for details.

**chainfrags_sn1** character vector containing the fragmentation rules for the chain fragments in sn1 position. See chainFrags for details.

**chainfrags_sn2** character vector containing the fragmentation rules for the chain fragments in sn2 position. See chainFrags for details. If empty, it will be estimated based on the difference between precursors and sn1 chains.

**intrules** character vector specifying the fragments to compare. See checkIntensityRules.

**rates** character vector with the expected rates between fragments given as a string (i.e. "3/1"). See checkIntensityRules.

**intrquired** logical vector indicating if any of the rules is required. If not, at least one must be verified to confirm the structure.

**coelCutoff** coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.

**dbs** list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See createLipidDB and assignDB.

**Details**

idPEneg function involves 5 steps. 1) FullMS-based identification of candidate PE as M-H. 2) Search of PE class fragments: 140.0115, 196.038, 214.048 ion coeluting with the precursor ion. If a loss of CH3 group is found coeluting with any candidate, this will be excluded as it is a characteristic fragment of PC. 3) Search of specific fragments that inform about chain composition in sn1 (lysoPE as M-H resulting from the loss of the FA chain at sn2) and sn2 (lysoPE as M-H resulting from the loss of the FA chain at sn1 or FA chain as M-H). 4) Look for possible chains structure based on the combination of chain fragments. 5) Check intensity rules to confirm chains position. In this case, lysoPE from sn1 is at least 3 times more intense than lysoPE from sn2.

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (Subclass, FA level, where chains are known but not their positions, or FA position level) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

**Value**

List with PE annotations (results) and some additional information (class fragments and chain fragments).

**Note**

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.
Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

library(LipidMSdata)
idPEneg(MS1 = MS1_neg, MSMS1 = MSMS1_neg, MSMS2 = MSMS2_neg)

<table>
<thead>
<tr>
<th>idPEneg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phosphoethanolamines (PE) annotation for ESI+</td>
</tr>
</tbody>
</table>

Description

PE identification based on fragmentation patterns for LC-MS/MS AIF data acquired in positive mode.

Usage

idPEneg(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10, rttol = 3, rt, adducts = c("M+H", "M+Na"), clfrags = c("dg_M+H-H2O"), clrequired = c(F), ftype = c("BB"), chainfrags_sn1 = c("lysope_M+H-H2O", "mg_M+H-H2O"), chainfrags_sn2 = c("mg_M+H-H2O"), intrules = c("lysope_sn1/lysope_sn1", "mg_sn1/mg_sn2"), rates = c("3/1", "1/2"), intrequired = c(F, F), coelCutoff = 0.8, dbs)

Arguments

<table>
<thead>
<tr>
<th>MS1</th>
</tr>
</thead>
<tbody>
<tr>
<td>list with two data frames containing all peaks from the full MS function (&quot;peaklist&quot; data frame) and the raw MS scans data (&quot;rawScans&quot; data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). &quot;rawScans&quot; data frame also needs an extra column named &quot;Scan&quot;, which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MSMS1</th>
</tr>
</thead>
<tbody>
<tr>
<td>list with two data frames containing all peaks from the high energy function (&quot;peaklist&quot; data frame) and the raw MS scans data (&quot;rawScans&quot; data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). &quot;rawScans&quot; data frame also needs an extra column named &quot;Scan&quot;, which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.</td>
</tr>
</tbody>
</table>
**MSMS2**

list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of *dataProcessing* function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

**ppm_precursor**

mass tolerance for precursor ions. By default, 5 ppm.

**ppm_products**

mass tolerance for product ions. By default, 10 ppm.

**rttol**

total RT window for coelution between precursor and product ions. By default, 3 seconds.

**rt**

RT range where the function will look for candidates. By default, it will search within all RT range in MS1.

**adducts**

expected adducts for PE in ESI+. Adducts allowed can be modified in adductsTable (dbs argument).

**clfrags**

vector containing the expected fragments for a given lipid class. See *checkClass* for details.

**clrequired**

logical vector indicating if each class fragment is required or not. If any of them is required, at least one of them must be present within the coeluting fragments. See *checkClass* for details.

**ftype**

character vector indicating the type of fragments in clfrags. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See *checkClass* for details.

**chainfrags_sn1**

character vector containing the fragmentation rules for the chain fragments in sn1 position. See *chainFrags* for details.

**chainfrags_sn2**

character vector containing the fragmentation rules for the chain fragments in sn2 position. See *chainFrags* for details. If empty, it will be estimated based on the difference between precursors and sn1 chains.

**intrules**

character vector specifying the fragments to compare. See *checkIntensityRules*.

**rates**

character vector with the expected rates between fragments given as a string (i.e. "3/1"). See *checkIntensityRules*.

**intrequired**

logical vector indicating if any of the rules is required. If not, at least one must be verified to confirm the structure.

**coelCutoff**

tolerance score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.

**dbs**

list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See *createLipidDB* and *assignDB*.

### Details

The *idPEpos* function involves 5 steps. 1) FullMS-based identification of candidate PE as M+H and M+Na. 2) Search of PE class fragments: loss of head group (DG as M+H-H2O) coeluting with the precursor ion. 3) Search of specific fragments that inform about chain composition at sn1 (MG as M+H-H2O resulting from the loss of the FA chain at sn2 and the head group or LPE as M+H-H2O...
resulting just from the loss of the FA chain) and sn2 (FA or MG chain from sn2 as M+H-H2O or the difference between precursor and sn1 chain fragments). 4) Look for possible chains structure based on the combination of chain fragments. 5) Check intensity rules to confirm chains position. LPE or MG from sn1 is at least 3 times more intense than the ones from sn2.

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (Subclass, FA level, where chains are known but not their positions, or FA position level) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

Value

List with PE annotations (results) and some additional information (class fragments and chain fragments).

Note

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

library(LipidMSdata)
idPEpos(MS1 = MS1_pos, MSMS1 = MSMS1_pos, MSMS2 = MSMS2_pos)

---

idPGneg

Phosphoglycerols (PG) annotation for ESI-

Description

PG identification based on fragmentation patterns for LC-MS/MS AIF data acquired in negative mode.
Usage

idPGneg(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10,
rttol = 3, rt, adducts = c("M-H"), clfrags = c(152.9958, 227.0326,
209.022, 74.0359), clrequired = c(F, F, F, F), ftype = c("F", "F",
"F", "NL"), chainfrags_sn1 = c("lysopg_M-H"),
chainfrags_sn2 = c("lysopg_M-H", "fa_M-H"),
intrules = c("lysopg_sn1/lysopg_sn2"), rates = c("2/1"),
intrequired = c(T), coelCutoff = 0.8, dbs)

Arguments

MS1 list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs a extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS1 list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs a extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS2 list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs a extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

ppm_precursor mass tolerance for precursor ions. By default, 5 ppm.

ppm_products mass tolerance for product ions. By default, 10 ppm.

rttol total rt window for coelution between precursor and product ions. By default, 3 seconds.

rt rt range where the function will look for candidates. By default, it will search within all RT range in MS1.

adducts expected adducts for PG in ESI-. Adducts allowed can be modified in adductsTable (dbs argument).

clfrags vector containing the expected fragments for a given lipid class. See checkClass for details.

clrequired logical vector indicating if each class fragment is required or not. If any of them is required, at least one of them must be present within the coeluting fragments. See checkClass for details.
ftype  character vector indicating the type of fragments in clfrags. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See checkClass for details.

chainfrags_sn1  character vector containing the fragmentation rules for the chain fragments in sn1 position. See chainFrags for details.

chainfrags_sn2  character vector containing the fragmentation rules for the chain fragments in sn2 position. See chainFrags for details. If empty, it will be estimated based on the difference between precursors and sn1 chains.

intrules  character vector specifying the fragments to compare. See checkIntensityRules.

rates  character vector with the expected rates between fragments given as a string (i.e. "3/1"). See checkIntensityRules.

intrequired  logical vector indicating if any of the rules is required. If not, at least one must be verified to confirm the structure.

coe1cutoff  coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.

dbs  list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See createLipidDB and assignDB.

Details

idPGneg function involves 5 steps. 1) FullMS-based identification of candidate PG as M-H. 2) Search of PG class fragments: 152.9958, 227.0326, 209.022 and neutral loss of 74.0359 coeluting with the precursor ion. 3) Search of specific fragments that inform about chain composition at sn1 (lysoPG as M-H resulting from the loss of the FA chain at sn2) and sn2 (lysoPG as M-H resulting from the loss of the FA chain at sn1 or FA chain as M-H). 4) Look for possible chains structure based on the combination of chain fragments. 5) Check intensity rules to confirm chains position. In this case, lysoPG from sn1 is at least 3 times more intense than lysoPG from sn2.

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (Subclass, FA level, where chains are known but not their positions, or FA position level) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

Value

List with PG annotations (results) and some additional information (class fragments and chain fragments).

Note

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>
idPIneg

Examples

```r
library(LipidMSdata)
idPIneg(MS1 = MS1_neg, MSMS1 = MSMS1_neg, MSMS2 = MSMS2_neg)
```

---

idPIneg  
Phosphoinositols (PI) annotation for ESI-

---

Description

PI identification based on fragmentation patterns for LC-MS/MS AIF data acquired in negative mode.

Usage

```r
idPIneg(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10,  
rttol = 3, rt, adducts = c("M-H"), clfrags = c(241.0115, 223.0008,  
259.0219, 297.0375), clrequired = c(F, F, F, F), ftype = c("F", "F",  
"F", "F"), chainfrags_sn1 = c("lysopi_M-H", "lysopa_M-H"),  
chainfrags_sn2 = c("lysopi_M-H", "lysopa_M-H", "fa_M-H"),  
intrules = c("lysopi_sn1/lysopi_sn2", "lysopa_sn1/lysopa_sn2"),  
rates = c("3/1", "3/1"), intrequired = c(F, F), coelCutoff = 0.8,  
dbs)
```

Arguments

- **MS1**: list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

- **MSMS1**: list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

- **MSMS2**: list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.
idPIneg function involves 5 steps. 1) FullMS-based identification of candidate PI as M-H. 2) Search of PI class fragments: 241.0115, 223.0008, 259.0219 and 297.0375 coeluting with the precursor ion. 3) Search of specific fragments that inform about chain composition at sn1 (lysoPI as M-H resulting from the loss of the FA chain at sn2 or lysoPA as M-H if it also losses the head group) and sn2 (lysoPI or lysoPA as M-H resulting from the loss of the FA chain at sn1 or FA chain as M-H). 4) Look for possible chains structure based on the combination of chain fragments. 5)
Check intensity rules to confirm chains position. In this case, lysoPI or lysoPA from sn1 is at least 3 times more intense than lysoPI or lysoPA from sn2.

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (Subclass, FA level, where chains are known but not their positions, or FA position level) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

Value

List with PI annotations (results) and some additional information (class fragments and chain fragments).

Note

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

library(LipidMSdata)
idPIneg(MS1 = MS1_neg, MSMS1 = MSMS1_neg, MSMS2 = MSMS2_neg)

idPOS

Lipids annotation for ESI+

Description

Lipids annotation based on fragmentation patterns for LC-MS/MS all-ions data acquired in positive mode. This function compiles all functions written for ESI+ annotations.

Usage

idPOS(MS1, MSMS1, MSMS2, ppm_precursor = 10, ppm_products = 10, rttol = 10, coelCutoff = 0.8, dbs)
Arguments

MS1 list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS1 list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS2 list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

ppm_precursor mass tolerance for precursor ions. By default, 5 ppm.

ppm_products mass tolerance for product ions. By default, 10 ppm.

rttol total rt window for coelution between precursor and product ions. By default, 3 seconds.

coelCutoff coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.

dbs list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See createLipidDB and assignDB.

Value

The output is a list with 2 elements: 1) a data frame that shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), m/z, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m/z error), confidenceLevel (Subclass, FA level, where chains are known but not their positions, or FA position level) and PFCS (parent-fragment coelution score mean of all fragments used for the identification); and 2) the original MS1 peaklist with the annotations on it.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

library(LipidMSdata)
idPOS(MS1_pos, MSMS1_pos, MSMS2_pos)
idPSneg

Phosphoserines (PS) annotation for ESI-

Description

PS identification based on fragmentation patterns for LC-MS/MS AIF data acquired in negative mode.

Usage

idPSneg(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10, rttol = 3, rt, adducts = c("M-H", "M+Na-2H"), clfrags = c(87.032, 152.9958), clrequired = c(F, F), ftype = c("NL", "F"), chainfrags_sn1 = c("lysopa_M-H", "lysopa_M-H-H2O"), chainfrags_sn2 = c("lysopa_M-H", "lysopa_M-H-H2O", "fa_M-H"), intrules = c("lysopa_sn1/lysopa_sn2"), rates = c("3/1"), intrequired = c(T), coelCutoff = 0.8, dbs)

Arguments

MS1 list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS1 list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS2 list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

ppm_precursor mass tolerance for precursor ions. By default, 5 ppm.

ppm_products mass tolerance for product ions. By default, 10 ppm.

rttol total rt window for coelution between precursor and product ions. By default, 3 seconds.
idPSneg

rt range where the function will look for candidates. By default, it will search within all RT range in MS1.

adducts expected adducts for PS in ESI-. Adducts allowed can be modified in adductsTable (dbs argument).

clf frags vector containing the expected fragments for a given lipid class. See checkClass for details.

cl required logical vector indicating if each class fragment is required or not. If any of them is required, at least one of them must be present within the coeluting fragments. See checkClass for details.

ftype character vector indicating the type of fragments in clfrags. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See checkClass for details.

chainfrags_sn1 character vector containing the fragmentation rules for the chain fragments in sn1 position. See chainFrags for details.

chainfrags_sn2 character vector containing the fragmentation rules for the chain fragments in sn2 position. See chainFrags for details. If empty, it will be estimated based on the difference between precursors and sn1 chains.

intrules character vector specifying the fragments to compare. See checkIntensityRules.

rates character vector with the expected rates between fragments given as a string (i.e. "3/1"). See checkIntensityRules.

int required logical vector indicating if any of the rules is required. If not, at least one must be verified to confirm the structure.

coe lCutoff coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.

dbs list of databases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See createLipidDB and assignDB.

Details

idPSneg function involves 5 steps. 1) FullMS-based identification of candidate PS as M-H or M+Na-2H. 2) Search of PS class fragments: neutral loss of 87.032 (serine) coeluting with the precursor ion. 3) Search of specific fragments that inform about chain composition at sn1 (lysoPA as M-H or M-H-H2O resulting from the loss of the FA chain at sn2 and the head group) and sn2 (lysoPA as M-H or M-H-H2O resulting from the loss of the FA chain at sn1 and the head group or FA chain as M-H). 4) Look for possible chains structure based on the combination of chain fragments. 5) Check intensity rules to confirm chains position. In this case, lysoPA from sn1 is at least 3 times more intense than lysoPA from sn2.

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (Subclass, FA level, where chains are known but not their positions, or FA position level) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

Value

list with PS annotations (results) and some additional information (class fragments and chain fragments).
**Note**

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

**Author(s)**

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

**Examples**

```r
library(LipidMSdata)
idPSneg(MS1 = MS1_neg, MSMS1 = MSMS1_neg, MSMS2 = MSMS2_neg)
```

---

**idSMpos**  
*Sphingomyelins (SM) annotation for ESI+*

**Description**

SM identification based on fragmentation patterns for LC-MS/MS AIF data acquired in positive mode.

**Usage**

```r
idSMpos(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10, rttol = 3, rt, adducts = c("M+H", "M+Na"), clfrags = c(104.1075, 184.0739, 183.06604), clrequired = c(F, F, F), ftype = c("F", "F", "NL"), chainfrags_sn1 = c("sph_M+H-2H2O"), chainfrags_sn2 = c(""), intrules = c(), rates = c(), intrequired = c(), coelCutoff = 0.8, dbs)
```

**Arguments**

- **MS1** list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.
idSMpos

MSMS1 list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS2 list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

ppm_precursor mass tolerance for precursor ions. By default, 5 ppm.

ppm_products mass tolerance for product ions. By default, 10 ppm.

rttol total rt window for coelution between precursor and product ions. By default, 3 seconds.

rt rt range where the function will look for candidates. By default, it will search within all RT range in MS1.

adducts expected adducts for SM in ESI+. Adducts allowed can be modified in adductsTable (dbs argument).

c1frags vector containing the expected fragments for a given lipid class. See checkClass for details.

c1required logical vector indicating if each class fragment is required or not. If any of them is required, at least one of them must be present within the coeluting fragments. See checkClass for details.

ftype character vector indicating the type of fragments in c1frags. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See checkClass for details.

chainfrags_sn1 character vector containing the fragmentation rules for the chain fragments in sn1 position. See chainFrags for details.

chainfrags_sn2 character vector containing the fragmentation rules for the chain fragments in sn2 position. See chainFrags for details. If empty, it will be estimated based on the difference between precursors and sn1 chains.

intrules character vector specifying the fragments to compare. See checkIntensityRules.

rates character vector with the expected rates between fragments given as a string (i.e. "3/1"). See checkIntensityRules.

intrequired logical vector indicating if any of the rules is required. If not, at least one must be verified to confirm the structure.

coelCutoff coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.

dbs list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See createLipidDB and assignDB.
Details

idSMpos function involves 5 steps. 1) FullMS-based identification of candidate SM as M+H and M+Na. 2) Search of SM class fragments: 104.1075, 184.0739 and neutral loss of 183.06604 coeluting with the precursor ion. 3) Search of specific fragments that inform about the composition of the sphingoid base (Sph as M+H-2H2O resulting from the loss of the FA chain) and the FA chain (by default it is calculated using the difference between precursor and sph chain fragments). 4) Look for possible chains structure based on the combination of chain fragments. 5) Check intensity rules to confirm chains position. In this case, there are no intensity rules by default as FA chain is unlikely to be detected.

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (Subclass, FA level, where chains are known but not their positions, or FA position level) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

Value

List with SM annotations (results) and some additional information (class fragments and chain fragments).

Note

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

library(LipidMSdata)
idSMpos(MS1 = MS1_pos, MSMS1 = MSMS1_pos, MSMS2 = MSMS2_pos)

idSphneg  Sphingoid bases (Sph) annotation for ESI-

Description

Sph identification based on fragmentation patterns for LC-MS/MS AIF data acquired in negative mode.
Usage

idSphneg(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10,
rttol = 3, rt, adducts = c("M-H"), clfrags = c("sph_M-H-H2O",
"sph_M-H-2H2O"), clrequired = c(F, F), ftype = c("BB", "BB"),
coelCutoff = 0.8, dbs)

Arguments

MS1 list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS1 list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS2 list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

ppm_precursor mass tolerance for precursor ions. By default, 5 ppm.

ppm_products mass tolerance for product ions. By default, 10 ppm.

rttol total rt window for coelution between precursor and product ions. By default, 3 seconds.

rt rt range where the function will look for candidates. By default, it will search within all RT range in MS1.

adducts expected adducts for Sph in ESI-. Adducts allowed can be modified in adductsTable (dbs argument).

clfrags vector containing the expected fragments for a given lipid class. See checkClass for details.

clrequired logical vector indicating if each class fragment is required or not. If any of them is required, at least one of them must be present within the coeluting fragments. See checkClass for details.

ftype character vector indicating the type of fragments in clfrags. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See checkClass for details.
**Details**

`idSphneg` function involves 2 steps. 1) FullMS-based identification of candidate Sph as M-H. 2) Search of Sph class fragments: neutral loss of 1 or 2 H2O molecules.

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (in this case, as Sph only have one chain, only Subclass and FA level are possible) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

**Value**

List with Sph annotations (results) and some additional information (class fragments and chain fragments).

**Note**

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

**Author(s)**

M Isabel Alcoriza-Balaguer &lt;maialba@alumni.uv.es&gt;

**Examples**

```r
library(LipidMSdata)
idSphneg(MS1 = MS1_neg, MSMS1 = MSMS1_neg, MSMS2 = MSMS2_neg)
```

**Description**

SphP annotation for ESI-
Usage

idSphPneg(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10,
   rttol = 3, rt, adducts = c("M-H"), clfrags = c(78.9585, 96.9691,
   "sphP_M-H-H2O"), clrequired = c(F, F, F), ftype = c("F", "F", "BB"),
   coelCutoff = 0.8, dbs)

Arguments

MS1  list with two data frames containing all peaks from the full MS function ("peak-
      list" data frame) and the raw MS scans data ("rawScans" data frame). They
      must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link
      between both data frames). "rawScans" data frame also needs an extra column
      named "Scan", which indicates the scan order number. Output of dataProcessing
      function. In case no coelution score needs to be applied, this argument can
      be just the peaklist data frame.

MSMS1 list with two data frames containing all peaks from the high energy function
      ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame).
      They must have four columns: m.z, RT (in seconds), int (intensity) and peakID
      (link between both data frames). "rawScans" data frame also needs an extra col-
      umn named "Scan", which indicates the scan order number. Output of dataProcess-
      ing function. In case no coelution score needs to be applied, this argument can
      be just the peaklist data frame.

MSMS2 list with two data frames containing all peaks from a second high energy func-
      tion ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame).
      They must have four columns: m.z, RT (in seconds), int (intensity) and peakID
      (link between both data frames). "rawScans" data frame also needs an extra col-
      umn named "Scan", which indicates the scan order number. Output of dataPro-
      cessing function. In case no coelution score needs to be applied, this argument can
      be just the peaklist data frame. Optional.

ppm_precursor mass tolerance for precursor ions. By default, 5 ppm.

ppm_products mass tolerance for product ions. By default, 10 ppm.

rttol total rt window for coelution between precursor and product ions. By default, 3
      seconds.

rt rt range where the function will look for candidates. By default, it will search
      within all RT range in MS1.

adducts expected adducts for SphP in ESI-. Adducts allowed can be modified in adduct-
      sTable (dbs argument).

clfrags vector containing the expected fragments for a given lipid class. See checkClass
      for details.

clrequired logical vector indicating if each class fragment is required or not. If any of them
      is required, at least one of them must be present within the coeluting fragments.
      See checkClass for details.

ftype character vector indicating the type of fragments in clfrags. It can be: "F" (frag-
      ment), "NL" (neutral loss) or "BB" (building block). See checkClass for details.
coelCutoff: coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.

dbs: list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See createLipidDB and assignDB.

Details

idSphpos function involves 2 steps. 1) FullMS-based identification of candidate SphP as M-H. 2) Search of SphP class fragments: 78.9585, 96.969 or neutral loss of 1 H2O molecule.

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (in this case, as SphP only have one chain, only Subclass and FA level are possible) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

Value

list with SphP annotations (results) and some additional information (class fragments and chain fragments).

Note

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

library(LipidMSdata)
idSphPneg(MS1 = MS1_neg, MSMS1 = MSMS1_neg, MSMS2 = MSMS2_neg)
Usage

```r
idSphpos(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10, rttol = 3, rt, adducts = c("M+H"), clfrags = c("sph_M+H-H2O", "sph_M+H-2H2O"), clrequired = c(F, F), ftype = c("BB", "BB"), coelCutoff = 0.8, dbs)
```

Arguments

**MS1**
list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

**MSMS1**
list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

**MSMS2**
list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of `dataProcessing` function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

**ppm_precursor** mass tolerance for precursor ions. By default, 5 ppm.

**ppm_products** mass tolerance for product ions. By default, 10 ppm.

**rttol** total RT window for coelution between precursors and product ions. By default, 3 seconds.

**rt** RT window where the function will look for candidates. By default, it will search within all RT range in MS1.

**adducts** expected adducts for Sph in ESI+. Adducts allowed can be modified in adductsTable (dbs argument).

**clfrags** vector containing the expected fragments for a given lipid class. See `checkClass` for details.

**clrequired** logical vector indicating if each class fragment is required or not. If any of them is required, at least one of them must be present within the coeluting fragments. See `checkClass` for details.

**ftype** character vector indicating the type of fragments in clfrags. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See `checkClass` for details.
**idSphPpos**

- **coelCutoff**: coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.
- **dbs**: list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See createLipidDB and assignDB.

**Details**

The idSphpos function involves 2 steps. 1) FullMS-based identification of candidate Sph as M+H. 2) Search of Sph class fragments: neutral loss of 1 or 2 H2O molecules.

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (in this case, as Sph only have one chain, only Subclass and FA level are possible) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

**Value**

A list with Sph annotations (results) and some additional information (class fragments and chain fragments).

**Note**

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for a Q-TOF 6550 from Agilent.

**Author(s)**

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

**Examples**

```r
library(LipidMSdata)
idSphpos(MS1 = MS1_pos, MSMS1 = MSMS1_pos, MSMS2 = MSMS2_pos)
```

---

**Description**

*Sphingoid bases phosphate (SphP) annotation for ESI+*

SphP identification based on fragmentation patterns for LC-MS/MS AIF data acquired in positive mode.
Usage

idSphPpos(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10, 
rttol = 3, rt, adducts = c("M+H"), clfrags = c("sphP_M+H-H2O", 
"sphP_M+H-2H2O", "sphP_M+H-H2O-NH4"), clrequired = c(F, F, F), 
ftype = c("BB", "BB", "BB"), coelCutoff = 0.7, dbs)

Arguments

MS1 list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS1 list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

MSMS2 list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs an extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

ppm_precursor mass tolerance for precursor ions. By default, 5 ppm.

ppm_products mass tolerance for product ions. By default, 10 ppm.

rttol total rt window for coelution between precursor and product ions. By default, 3 seconds.

rt rt range where the function will look for candidates. By default, it will search within all RT range in MS1.

adducts expected adducts for SphP in ESI+. Adducts allowed can be modified in adductsTable (dbs argument).

clfrags vector containing the expected fragments for a given lipid class. See checkClass for details.

clrequired logical vector indicating if each class fragment is required or not. If any of them is required, at least one of them must be present within the coeluting fragments. See checkClass for details.

ftype character vector indicating the type of fragments in clfrags. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See checkClass for details.
idTGpos

coelCutoff coelution score threshold between parent and fragment ions. Only applied if rawData info is supplied. By default, 0.8.

dbs list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See createLipidDB and assignDB.

Details

idSphpos function involves 2 steps. 1) FullMS-based identification of candidate SphP as M+H. 2) Search of SphP class fragments: neutral loss of 1 or 2 H2O molecules, or H2O and NH4.

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bounds), FA composition (specific chains composition if it has been confirmed), m/z, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m.z error), confidenceLevel (in this case, as SphP only have one chain, only Subclass and FA level are possible). and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

Value

list with SphP annotations (results) and some additional information (class fragments and chain fragments).

Note

Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

library(LipidMSdata)
idSphpos(MS1 = MS1_pos, MSMS1 = MSMS1_pos, MSMS2 = MSMS2_pos)

idTGpos Triacylglycerols (TG) annotation for ESI+

Description

TG identification based on fragmentation patterns for LC-MS/MS AIF data acquired in positive mode.
Usage

idTGpos(MS1, MSMS1, MSMS2, ppm_precursor = 5, ppm_products = 10, 
rttol = 3, rt, adducts = c("M+NH4", "M+Na"), clfrags = c(), 
clrequired = c(), ftype = c(), 
chainfrags_sn1 = c("cbdiff-dg_M+H-H2O"), 
chainfrags_sn2 = c("cbdiff-dg_M+H-H2O"), 
chainfrags_sn3 = c("cbdiff-dg_M+H-H2O"), 
intrules = c("cbdiff-dg_sn2/cbdiff-dg_sn1", "cbdiff-dg_sn2/cbdiff-dg_sn3", "cbdiff-dg_sn1/cbdiff-dg_sn3"), 
rates = c("1", "1", "1"), intrequired = c(T, T, T), 
coelCutoff = 0.8, dbs)

Arguments

- **MS1**: list with two data frames containing all peaks from the full MS function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs a extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

- **MSMS1**: list with two data frames containing all peaks from the high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs a extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame.

- **MSMS2**: list with two data frames containing all peaks from a second high energy function ("peaklist" data frame) and the raw MS scans data ("rawScans" data frame). They must have four columns: m.z, RT (in seconds), int (intensity) and peakID (link between both data frames). "rawScans" data frame also needs a extra column named "Scan", which indicates the scan order number. Output of dataProcessing function. In case no coelution score needs to be applied, this argument can be just the peaklist data frame. Optional.

- **ppm_precursor**: mass tolerance for precursor ions. By default, 5 ppm.

- **ppm_products**: mass tolerance for product ions. By default, 10 ppm.

- **rttol**: total rt window for coelution between precursor and product ions. By default, 3 seconds.

- **rt**: rt range where the function will look for candidates. By default, it will search within all RT range in MS1.

- **adducts**: expected adducts for TG in ESI+. Adducts allowed can be modified in adductsTable (dbs argument).

- **clfrags**: vector containing the expected fragments for a given lipid class. See checkClass for details.
clrequired logical vector indicating if each class fragment is required or not. If any of them is required, at least one of them must be present within the coeluting fragments. See checkClass for details.

ftype character vector indicating the type of fragments in clfrags. It can be: "F" (fragment), "NL" (neutral loss) or "BB" (building block). See checkClass for details.

chainfrags_sn1 character vector containing the fragmentation rules for the chain fragments in sn1 position. See chainFrags for details.

chainfrags_sn2 character vector containing the fragmentation rules for the chain fragments in sn2 position. See chainFrags for details. If empty, it will be estimated based on the difference between precursors and sn1 chains.

chainfrags_sn3 character vector containing the fragmentation rules for the chain fragments in sn3 position. See chainFrags for details. If empty, it will be estimated based on the difference between precursors and sn2 chains.

intrules character vector specifying the fragments to compare. See checkIntensityRules. If some intensity rules should be employed to identify the chains position but they are not known yet, use "Unknown". If it isn’t required, leave an empty vector.

rates character vector with the expected rates between fragments given as a string (i.e. "3/1"). See checkIntensityRules.

intrequired logical vector indicating if any of the rules is required. If not, at least one must be verified to confirm the structure.

coe1Cutoff score threshold between parent and fragment ions. Only applied if raw data is supplied. By default, 0.8.

dbs list of data bases required for annotation. By default, dbs contains the required data frames based on the default fragmentation rules. If these rules are modified, dbs may need to be supplied. See createLipidDB and assignDB.

Details

idTGpos function involves 5 steps. 1) FullMS-based identification of candidate TG as M+NH4 and M+Na. 2) Search of TG class fragments: there are no class fragment by default. 3) Search of specific fragments that inform about the FA chains: DGs resulting from the loss of FA chains as M+H-H2O. 4) Look for possible chains structure based on the combination of chain fragments. 5) Check intensity rules to confirm chains position. In the case of TG, DG resulting from the loss of sn2 if the most intense, followed by the loss of sn1 and sn3, but this FA position level still needs to be improved due to the high level of coelution for TG.

Results data frame shows: ID, class of lipid, CDB (total number of carbons and double bonds), FA composition (specific chains composition if it has been confirmed), mz, RT (in seconds), I (intensity, which comes directly from de input), Adducts, ppm (m/z error), confidenceLevel (Subclass, FA level, where chains are known but not their positions, or FA position level) and PFCS (parent-fragment coelution score mean of all fragments used for the identification).

Value

list with TG annotations (results) and some additional information (class fragments and chain fragments).
Note
Isotopes should be removed before identification to avoid false positives. This function has been written based on fragmentation patterns observed for two different platforms (QTOF 6550 from Agilent and Sinapt G2-Si from Waters), but it may need to be customized for other platforms or acquisition settings.

Author(s)
M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

Examples

```r
library(LipidMSdata)
idTGpos(MS1 = MS1_pos, MSMS1 = MSMS1_pos, MSMS2 = MSMS2_pos)
```

---

**lysopadb**  
*LPAs database*

Description
In silico generated database for common LPAs.

Usage
`data("lysopadb")`

Format
Data frame with 30 observations and the following 3 variables.

- **formula** character vector containing molecular formulas.
- **total** character vector indicating the total number of carbons and double bounds of the chains.
- **Mass** numeric vector with the neutral masses.
**lysopcdb**  
*LPCs database*

**Description**

In silico generated database for common LPCs.

**Usage**

```r
data("lysopcdb")
```

**Format**

Data frame with 30 observations and the following 3 variables.

- `formula`: character vector containing molecular formulas.
- `total`: character vector indicating the total number of carbons and double bounds of the chains.
- `Mass`: numeric vector with the neutral masses.

---

**lysopedb**  
*LPEs database*

**Description**

In silico generated database for common LPEs.

**Usage**

```r
data("lysopedb")
```

**Format**

Data frame with 30 observations and the following 3 variables.

- `formula`: character vector containing molecular formulas.
- `total`: character vector indicating the total number of carbons and double bounds of the chains.
- `Mass`: numeric vector with the neutral masses.
lysogpdb  
*LPGs database*

**Description**

In silico generated database for common LPGs.

**Usage**

```r
data("lysogpdb")
```

**Format**

Data frame with 30 observations and the following 3 variables.

- `formula` character vector containing molecular formulas.
- `total` character vector indicating the total number of carbons and double bounds of the chains.
- `Mass` numeric vector with the neutral masses.

lysopidb  
*LPIs database*

**Description**

In silico generated database for common LPIs.

**Usage**

```r
data("lysopidb")
```

**Format**

Data frame with 30 observations and the following 3 variables.

- `formula` character vector containing molecular formulas.
- `total` character vector indicating the total number of carbons and double bounds of the chains.
- `Mass` numeric vector with the neutral masses.
lysopsdb

**LPSs database**

**Description**

In silico generated database for common LPSs

**Usage**

```r
data("lysopsdb")
```

**Format**

Data frame with 30 observations and the following 3 variables.

- `formula` character vector containing molecular formulas.
- `total` character vector indicating the total number of carbons and double bounds of the chains.
- `Mass` numeric vector with the neutral masses.

mgdb

**MGs database**

**Description**

In silico generated database for common MGs.

**Usage**

```r
data("mgdb")
```

**Format**

Data frame with 30 observations and the following 3 variables.

- `formula` character vector containing molecular formulas.
- `total` character vector indicating the total number of carbons and double bounds of the chains.
- `Mass` numeric vector with the neutral masses.
organizeResults

**nlsphdb**  
*Neutral losses db for sphingoid bases. It is employed by idCerneg function.*

**Description**

In silico generated database for neutral losses of sphingoid bases in ESI-.

**Usage**

```r
data("nlsphdb")
```

**Format**

Data frame with 4 observations and the following 3 variables.

- `formula` character vector containing molecular formulas.
- `total` character vector indicating the total number of carbons and double bounds of the chains.
- `Mass` numeric vector with the neutral masses.

organizeResults  
*Prepare output for LipidMS annotation functions*

**Description**

Prepare a readable output for LipidMS identification functions.

**Usage**

```r
organizeResults(candidates, clfrags, classConf, chainsComb, intrules, intConf, nchains, class)
```

**Arguments**

- `candidates` candidates data frame. Output of `findCandidates`.
- `clfrags` vector containing the expected fragments for a given lipid class.
- `classConf` output of `checkClass`
- `chainsComb` output of `combineChains`
- `intrules` character vector specifying the fragments to compare. See `checkIntensityRules`.
- `intConf` output of `checkIntensityRules`
- `nchains` number of chains of the targeted lipid class.
- `class` character value. Lipid class (i.e. PC, PE, DG, TG, etc.).
library(LipidMSdata)
dbs <- assignDB()

candidates <- findCandidates(MS1 = MS1_neg$peaklist,
   db = dbs$pgdb, ppm = 10, rt = c(0, 2000), adducts = c("M-H"),
   rttol = 10, rawData = MS1_neg$rawScans, coelCutoff = 0.8)

MSMS <- rbind(MSMS1_neg$peaklist, MSMS2_neg$peaklist)
rawData <- rbind(MS1_neg$rawScans, MSMS1_neg$rawScans, MSMS2_neg$rawScans)
coelfrags <- coelutingFrags(candidates$RT, MSMS, rttol = 10, rawData = rawData,
   coelCutoff = 0.8)

classConf <- checkClass(candidates, coelfrags,
   clfrags = c(227.0326, 209.022, 74.0359), clrequisites = c(F, F, F, F),
   ftype = c("F", "F", "NL"), ppm = 10, dbs = dbs)

sn1 <- chainFrags(coelfrags, chainfrags = c("lysopg_M-H"), ppm = 10,
   dbs = dbs)

sn2 <- chainFrags(coelfrags, chainfrags = c("fa_M-H"), ppm = 10, dbs = dbs)

chainsComb <- combineChains(candidates, nchains=2, sn1, sn2)

intConf <- checkIntensityRules(intrules = c("lysopg_sn1/lysopg_sn1"),
   rates = c("2/1"), intrequired = c(T), nchains=2, chainsComb, sn1, sn2)

res <- organizeResults(candidates, clfrags = c(227.0326, 209.022, 74.0359),
   classConf, chainsComb, intrules = c("lysopg_sn1/lysopg_sn1"), intConf,
   nchains = 2, class="PG")
**Format**

Data frame with 147 observations and the following 3 variables.

*formula* character vector containing molecular formulas.

*total* character vector indicating the total number of carbons and double bounds of the chains.

*Mass* numeric vector with the neutral masses.

---

**Description**

In silico generated database for common PCs.

**Usage**

```r
data("pcdb")
```

---

**Format**

Data frame with 147 observations and the following 3 variables.

*formula* character vector containing molecular formulas.

*total* character vector indicating the total number of carbons and double bounds of the chains.

*Mass* numeric vector with the neutral masses.

---

**Description**

In silico generated database for common PEs.

**Usage**

```r
data("pedb")
```
**pgdb**

*PGs database*

**Description**

In silico generated database for common PGs.

**Usage**

```r
data("pgdb")
```

**Format**

Data frame with 147 observations and the following 3 variables.

- **formula** character vector containing molecular formulas.
- **total** character vector indicating the total number of carbons and double bounds of the chains.
- **Mass** numeric vector with the neutral masses.

---

**pidb**

*PIs database*

**Description**

In silico generated database for common PIs.

**Usage**

```r
data("pidb")
```

**Format**

Data frame with 147 observations and the following 3 variables.

- **formula** character vector containing molecular formulas.
- **total** character vector indicating the total number of carbons and double bounds of the chains.
- **Mass** numeric vector with the neutral masses.
psdb

**Description**

In silico generated database for common PSs.

**Usage**

```r
data("psdb")
```

**Format**

Data frame with 147 observations and the following 3 variables.

- *formula*: character vector containing molecular formulas.
- *total*: character vector indicating the total number of carbons and double bounds of the chains.
- *Mass*: numeric vector with the neutral masses.

---

**searchIsotopes**

**Target isotopes search**

**Description**

This function uses annotation results of an unlabelled sample to search for labelled compounds in a labelled sample.

**Usage**

```r
searchIsotopes(results, MS1, label, adductsTable = LipidMS::adductsTable, rttol = 10, ppm = 10)
```

**Arguments**

- **results**: annotation results for an unlabelled sample. Output of identification functions (i.e. idPOS$results).
- **MS1**: Data frame with at least three columns: m.z, RT, int. Peak list for the labelled sample. Output of `dataProcessing` function (MS$peaklist).
- **label**: isotope employed for the experiment. It can be "13C" or "D".
- **adductsTable**: adducts table employed for lipids annotation.
- **rttol**: rt window in seconds.
- **ppm**: mass error tolerance.
Value

List with the isotopes for each compound in the results data frame.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>

---

Description

Separation of .mzXML files from all-ions data by collision energy to work with them separately.

Usage

sepByCE(file, output)

Arguments

- file: path of the input .mzXML file
- output: a unique character value indicating the name of the output files. The energy employed and .mzXML will be added automatically to each file.

Details

This function has been designed based on mzXML files obtained from .d files (Agilent) using msConvert tool, in which we can find the collision energy information. In addition to separate files by collision energies, this function also changes the MS level of the high energy scans from 2 to 1 allowing their treatment (peak-picking for each collision energy, alignment, i.e) with common software (xcms, mzMine2, enviPick, etc).

Value

As many .mzXML files as different collision energies employed.

Note

Be careful with input and output arguments. For example, "file.mzXML" would be the input argument and "file_sep" could be the output.

Author(s)

M Isabel Alcoriza-Balaguer <maialba@alumni.uv.es>
Examples

```r
## Not run:
sepByCE("input_file.mzXML", "output_file")
## End(Not run)
```

---

**smdb**  
*SMs database*

**Description**

In silico generated database for common SMs.

**Usage**

```r
data("smdb")
```

**Format**

Data frame with 52 observations and the following 3 variables.

- **formula** character vector containing molecular formulas.
- **total** character vector indicating the total number of carbons and double bounds of the chains.
- **Mass** numeric vector with the neutral masses.

---

**sphdb**  
*Sphingoid bases database*

**Description**

In silico generated database for common sphingoid bases.

**Usage**

```r
data("sphdb")
```

**Format**

Data frame with 4 observations and the following 3 variables.

- **formula** character vector containing molecular formulas.
- **total** character vector indicating the total number of carbons and double bounds of the chains.
- **Mass** numeric vector with the neutral masses.
sphPdb  

**Sphingoid bases phosphate database**

**Description**
In silico generated database for common sphingoid bases phosphate.

**Usage**
```r
data("sphPdb")
```

**Format**
Data frame with 4 observations and the following 3 variables.

- `formula` character vector containing molecular formulas.
- `total` character vector indicating the total number of carbons and double bounds of the chains.
- `Mass` numeric vector with the neutral masses.

---

tgdb  

**TGs database**

**Description**
In silico generated database for common TGs.

**Usage**
```r
data("tgdb")
```

**Format**
Data frame with 376 observations and the following 3 variables.

- `formula` character vector containing molecular formulas.
- `total` character vector indicating the total number of carbons and double bounds of the chains.
- `Mass` numeric vector with the neutral masses.
Index

*Topic datasets
  adductsTable, 3
  baconjdb, 4
  badb, 5
  carnitinesdb, 5
  CEdb, 6
  cerdb, 6
  cldb, 12
  confLevels, 15
  dgdb, 20
  fadb, 20
  fahfadb, 21
  hfadb, 23
  lysopadb, 94
  lysopcdb, 95
  lysopedb, 95
  lysopgdb, 96
  lysopidb, 96
  lysopsdb, 97
  mgdb, 97
  nlsphdb, 98
  padb, 99
  pcdb, 100
  pedb, 100
  pgdb, 101
  pidb, 101
  psdb, 102
  smdb, 104
  spdb, 104
  spPhpdb, 105
  tgd, 105

  adductsTable, 3, 22
  assignDB, 4, 17, 25, 27, 29, 31, 34, 36, 39, 41, 43, 45, 47, 49, 52, 54, 56, 58, 60, 62,
  64, 66, 69, 71, 74, 76, 78, 80, 82, 85, 87, 89, 91, 93
  baconjdb, 4
  badb, 5
  carnitinesdb, 5
  CEdb, 6
  cerdb, 6
  chainFrags, 7, 7, 14, 25, 27, 29, 31, 34, 36, 38, 41, 43, 45, 47, 49, 52, 54, 56, 58, 64,
  66, 69, 71, 74, 76, 80, 82, 93
  checkClass, 8, 27, 29, 31, 33, 36, 38, 41, 43, 45, 47, 49, 51, 54, 56, 58, 60, 63, 64,
  66, 68, 69, 71, 73, 74, 76, 80, 82, 84, 86, 88, 90, 92, 93, 98
  checkIntensityRules, 10, 31, 34, 36, 38, 39, 41, 64, 66, 69, 71, 74, 76, 80, 82, 93, 98
  cldb, 12
  coelutingFrags, 7, 9, 12
  coelutionScore, 13
  combineChains, 7, 11, 14, 98
  confLevels, 15
  createLipidDB, 7, 9, 16, 17, 22, 25, 27, 29, 31, 34, 36, 39, 41, 43, 45, 47, 49, 52, 54, 56,
  58, 60, 62, 64, 66, 69, 71, 74, 76, 78, 80, 82, 85, 87, 89, 91, 93
  crossTables, 17
  dataProcessing, 13, 18, 22, 24, 26, 28–31, 33, 35, 36, 38, 40–47, 49, 51, 53, 55, 57–63,
  65, 66, 68, 70, 71, 73, 75, 76, 78, 81, 82, 84, 86, 88, 90, 92, 102
  dgdb, 20
  fadb, 20
  fahfadb, 21
  findCandidates, 9, 13, 14, 21, 22, 98
  getInclusionList, 23
  hfadb, 23
  idBAneg, 24
  idCarpos, 26
idCEpos, 28
idCerneg, 30
idCerpos, 32
idCLneg, 35
idDGpos, 37
idFAHFAneg, 40
idFAneg, 42
idLPCneg, 44
idLPEneg, 46
idLPEpos, 48
idLPEneg, 50
idLPNneg, 53
idLPIneg, 55
idLPSneg, 57
idMGpos, 59
idNEG, 61
idPCneg, 62
idPCpos, 65
idPEneg, 67
idPEpos, 70
idPGneg, 72
idPINeg, 75
idPOS, 77
idPSneg, 79
idSMpos, 81
idSphneg, 83
idSphPneg, 85
idSphpos, 87
idSphPpos, 89
idTGpos, 91

lysopadb, 94
lysocpdb, 95
lysopedb, 95
lysopgdb, 96
lysopidb, 96
lysopsdb, 97

mgdb, 97
nlsphdb, 98

organizeResults, 98

padb, 99
pcdb, 100
pedb, 100
pgdb, 101
pidb, 101