Package ‘CluMSID’

May 8, 2025

Type Package

Title Clustering of MS2 Spectra for Metabolite Identification

Version 1.25.0

Maintainer Tobias Depke <`depke@mailbox.org`>

Description CluMSID is a tool that aids the identification of features in untargeted LC-MS/MS analysis by the use of MS2 spectra similarity and unsupervised statistical methods. It offers functions for a complete and customisable workflow from raw data to visualisations and is interfaceable with the xmcs family of preprocessing packages.

License MIT + file LICENSE

Encoding UTF-8

URL `https://github.com/tdepke/CluMSID`

BugReports `https://github.com/tdepke/CluMSID/issues`

LazyData true

Depends R (>= 3.6)

biocViews Metabolomics, Preprocessing, Clustering

Imports mzR, S4Vectors, dbscan, RColorBrewer, ape, network, GGally, ggplot2, plotly, methods, utils, stats, sna, grDevices, graphics, Biobase, gplots, MSnbase

RoxygenNote 6.1.1

Suggests knitr, rmarkdown, testthat, dplyr, readr, stringr, magrittr, CluMSIDdata, metaMS, metaMSdata, xcms

VignetteBuilder knitr

git_url https://git.bioconductor.org/packages/CluMSID

git_branch devel

git_last_commit 03bd7ec

git_last_commit_date 2025-04-15

Repository Bioconductor 3.22

Date/Publication 2025-05-08

Author Tobias Depke [aut, cre],
Raimo Franke [ctb],
Mark Broenstrup [ths]

Contents

accessors . . . 2
addAnnotations . . . 4
as.MS2spectrum . . . 4
cossim . . . 5
distanceMatrix . . . 6
extractMS2spectra . . . 7
extractPseudospectra . . . 7
featureList . . . 8
findFragment . . . 9
findNL . . . 9
getSimilarities . . . 10
getSpectrum . . . 11
HCplot . . . 12
HCtbl . . . 13
MDSplot . . . 13
mergeMS2spectra . . . 14
mergeSpecList . . . 15
mergeTolerance . . . 16
MS2spectrum-class . . . 16
networkplot . . . 17
neutrallossPatterns . . . 18
OPTICSplot . . . 19
OPTICStbl . . . 20
pseudospectrum-class . . . 20
specplot . . . 21
splitPolarities . . . 21
writeFeaturelist . . . 22

Index **23**

`accessors` *Accessor functions for individual slots of* `MS2spectrum` *and* `pseudospectrum` *objects*

Description

Accessor functions for individual slots of `MS2spectrum` and `pseudospectrum` objects

Usage

```
accessID(x)

accessAnnotation(x)

accessPrecursor(x)

accessRT(x)

accessPolarity(x)
```

```
accessSpectrum(x)

accessNeutralLosses(x)
```

Arguments

`x`	An object of class `MS2spectrum` or `pseudospectrum`

Value

The value of the respective slot of the object (`id`, `annotation`, `precursor`, `rt`, `spectrum`, `neutral_losses`)

Examples

```
load(file = system.file("extdata",
    "annotatedSpeclist.RData",
    package = "CluMSIDdata"))

accessID(annotatedSpeclist[[1]])

load(file = system.file("extdata",
    "annotatedSpeclist.RData",
    package = "CluMSIDdata"))

accessAnnotation(annotatedSpeclist[[1]])

load(file = system.file("extdata",
    "annotatedSpeclist.RData",
    package = "CluMSIDdata"))

accessPrecursor(annotatedSpeclist[[1]])

load(file = system.file("extdata",
    "annotatedSpeclist.RData",
    package = "CluMSIDdata"))

accessRT(annotatedSpeclist[[1]])

load(file = system.file("extdata",
    "annotatedSpeclist.RData",
    package = "CluMSIDdata"))

accessPolarity(annotatedSpeclist[[1]])

load(file = system.file("extdata",
    "annotatedSpeclist.RData",
    package = "CluMSIDdata"))

accessSpectrum(annotatedSpeclist[[1]])

load(file = system.file("extdata",
    "annotatedSpeclist.RData",
    package = "CluMSIDdata"))

accessNeutralLosses(annotatedSpeclist[[1]])
```

`addAnnotations` *Adding external annotations to list of* `MS2spectrum` *objects*

Description

`addAnnotations` is used to add annotations that have been assigned externally, e.g. by library search, to a list of `MS2spectrum` objects as produced by `extractMS2spectra` and `mergeSpecList`.

Usage

```
addAnnotations(featlist, annolist, annotationColumn = 4)
```

Arguments

`featlist`	A list of `MS2spectrum` objects as produced by `extractMS2spectra` and `mergeSpecList`
`annolist`	A list of annotations, either as a `data.frame` or csv file. The order of features must be the same as in `featlist`. Please see the package vignette for a detailed example!
`annotationColumn`	The column of `annolist` were the annotation is found. Default is 4, which is the case if `writeFeaturelist` followed by manual addition of annotations, e.g. in Excel, is used to generate `annolist`.

Value

A list of `MS2spectrum` objects as produced by `extractMS2spectra` and `mergeSpecList` with external annotations added to the `annotation` slot of each `MS2spectrum` object.

Examples

```
load(file = system.file("extdata",
    "featlist.RData",
    package = "CluMSIDdata"))

addAnnotations(featlist, system.file("extdata",
                "post_anno.csv",
                package = "CluMSIDdata"),
                annotationColumn = 4)
```

`as.MS2spectrum` *Convert spectra from* **MSnbase** *classes*

Description

Convert spectra from **MSnbase** classes

Usage

```
as.MS2spectrum(x)
```

Arguments

`x`	An object of class `Spectrum` or `Spectrum2`

Value

An object of class `MS2spectrum`

Examples

```
#Load a "Spectrum2" object from MSnbase
library(MSnbase)
sp <- itraqdata[["X1"]]
#Convert this object to "MS2spectrum" class
new_sp <- as.MS2spectrum(sp)
#Or alternatively:
new_sp <- as(sp, "MS2spectrum")
```

`cossim` *Calculate cosine similarity between two spectra*

Description

`cossim()` calculates the cosine of the spectral constrast angle as a measure for the similarity of two spectra.

Usage

```
cossim(x, y, type = c("spectrum", "neutral_losses"),
    mzTolerance = 1e-05)

## S4 method for signature 'MS2spectrum,MS2spectrum'
cossim(x, y, type = c("spectrum",
    "neutral_losses"), mzTolerance = 1e-05)

## S4 method for signature 'pseudospectrum,pseudospectrum'
cossim(x, y,
    type = c("spectrum", "neutral_losses"), mzTolerance = 1e-05)
```

Arguments

`x, y`	MS2 spectra, either as `matrix`, `MS2spectrum` or `pseudospectrum` objects. `x` and `y` must have the same class.
`type`	Whether similarity between spectra (`"spectrum"`, default) or neutral loss patterns (`"neutral_losses"`) is to be compared
`mzTolerance`	The m/z tolerance used for merging. If two fragment peaks are within tolerance, they are regarded as the same. Defaults to `1e-5`, i.e. 10ppm.

Value

The cosine similarity of `x` and `y`

Methods (by class)

- `x = MS2spectrum,y = MS2spectrum`: `cossim` method for `MS2spectrum` objects
- `x = pseudospectrum,y = pseudospectrum`: `cossim` method for `pseudospectrum` objects

Examples

```
load(file = system.file("extdata",
    "annotatedSpeclist.RData",
    package = "CluMSIDdata"))

cossim(annotatedSpeclist[[1]], annotatedSpeclist[[2]])
```

`distanceMatrix` *Create distance matrix from list of spectra*

Description

`distanceMatrix()` creates a distance matrix from a list of MS2 spectra, MS1 pseudospectra or neutral loss patterns by pairwise comparison using the specified distance function. This distance matrix is the basis for CluMSID's data mining functions.

Usage

```
distanceMatrix(speclist, distFun = "cossim", type = c("spectrum",
    "neutral_losses"), mz_tolerance = 1e-05)
```

Arguments

`speclist`	A list of `MS2spectrum` or `pseudospectrum` objects as generated by `extractMS2spectra` or `extractPseudospectra`.
`distFun`	The distance function to be used. At the moment, only `cossim` is implemented.
`type`	`"spectrum"` (default) for MS2 spectra or MS1 pseudospectra or `"neutral_losses"` for neutral loss patterns.
`mz_tolerance`	The *m/z* tolerance to be used for merging, default is `1e-5`, i.e. +/- 10ppm. If the mass-to-charge ratios of two peaks differ less than *mz_tolerance*, they are assumed to have the same *m/z*

Value

A numeric `length(speclist)` by `length(speclist)` matrix containing pairwise distances (1 - similarity) between all features in `speclist`. Row and column names are taken from the `id` slot or, if present, pasted from the `id` and `annotation` slots of the `MS2spectrum` or `pseudospectrum` objects.

Examples

```
load(file = system.file("extdata",
    "annotatedSpeclist.RData",
    package = "CluMSIDdata"))

distanceMatrix(annotatedSpeclist[1:20])
```

`extractMS2spectra` *Extract MS2 spectra from raw data files*

Description

`extractMS2spectra()` is used to extract MS2 spectra from raw data files, e.g. mzXML files.

Usage

```
extractMS2spectra(MSfile, min_peaks = 2, recalibrate_precursor = FALSE,
    RTlims = NULL)
```

Arguments

`MSfile`	An LC-MS/MS raw data file in one of the non-proprietary formats that can be parsed by `mzR`, e.g. mzXML or mzML.
`min_peaks`	Minimum number of peaks in MS2 spectrum, defaults to `2`. Spectra with less than `min_peaks` fragment peaks will be ignored and not extracted.
`recalibrate_precursor`	Logical, defaults to `FALSE`. Applicable only for files that were exported to mzXML using a deprecated version of Bruker Compass Xport (< 3.0.13). If set to `TRUE`, the precursor m/z will be recalculated from the respective fragment m/z in the MS2 spectrum. For details, see Depke et al. 2017.
`RTlims`	Retention time interval for the extraction of spectra. Provide as numeric vector of length 2. Spectra with retention time < `RTlims[1]` or > `RTlims[2]` will be ignored.

Value

A `list` with objects of class `MS2spectrum`, containing MS2 spectra extracted from the raw data.

Examples

```
my_spectra <- extractMS2spectra(MSfile = system.file("extdata",
                                "PoolA_R_SE.mzXML",
                                package = "CluMSIDdata"),
                                min_peaks = 4, RTlims = c(0,10))
```

`extractPseudospectra` *Extract pseudospectra*

Description

`extractPseudospectra()` is used to extract MS1 pseudospectra from **CAMERA** output.

Usage

```
extractPseudospectra(x, min_peaks = 1, intensity_columns = NULL)
```

Arguments

`x`	**CAMERA** output that contains information on pseudospectra. Can either be of class `data.frame` or `xsAnnotate`. It is recommended to use either `xsAnnotate` objects or `data.frames` generated from XCMSonline results tables but other `data.frames` are possible.
`min_peaks`	Minimum number of peaks in pseudospectrum, defaults to `1`. See `extractMS2spectra`.
`intensity_columns`	Numeric, defaults to `NULL`. If a `data.frame` is used as input which has not been generated from an XCMSonline results table, the indices of the columns that contain the peak intensities in the different samples have to be indicated as `intensity_columns`.

Value

A list of pseudospectra, stored as objects of class `pseudospectrum`, analogous to the output of `extractMS2spectra`.

Examples

```
pstable <- readr::read_delim(file = system.file("extdata",
                                   "TD035_XCMS.annotated.diffreport.tsv",
                                   package = "CluMSIDdata"), delim = "\t")

pseudospeclist <- extractPseudospectra(pstable, min_peaks = 2)
```

`featureList` *Generate a* `data.frame` *with feature information from list of* `MS2spectrum` *objects*

Description

`featureList` generates a `data.frame` that contains feature ID, precurosur *m/z* and retention time for all features contained in a list of `MS2spectrum` objects as produced by `extractMS2spectra` and `mergeSpecList`. `featureList` is used internally by `writeFeaturelist`.

Usage

```
featureList(featlist)
```

Arguments

`featlist`	A list of `MS2spectrum` objects as produced by `extractMS2spectra` and `mergeSpecList`

Details

Although originally designed for lists of `MS2spectrum` objects, the function also works with lists of `pseudospectrum` objects. In this case, `NA` is given for precursor *m/z*.

Value

A `data.frame` that contains feature ID, precurosur *m/z* (if available) and retention time

Examples

```
load(file = system.file("extdata",
    "featlist.RData",
    package = "CluMSIDdata"))

pre_anno <- featureList(featlist)
```

`findFragment` *Find spectra that contain a specific fragment*

Description

`findFragment` is used to find spectra that contain a specific fragment ion. Its sister function is `findNL`, which finds specific neutral losses. Both functions work analogous to `getSpectrum`.

Usage

```
findFragment(featlist, mz, tolerance = 1e-05)
```

Arguments

`featlist`	a list that contains only objects of class `MS2spectrum`
`mz`	The mass-to-charge ratio of the fragment ion of interest.
`tolerance`	The *m/z* tolerance for the fragment ion search. Default is `1E-05`, i.e. +/- 10ppm.

Value

If the respective fragment is only found in one spectrum, the output is an object of class `MS2spectrum`; if it is found in more than one spectrum, the output is a list of `MS2spectrum` objects.

Examples

```
load(file = system.file("extdata",
    "annotatedSpeclist.RData",
    package = "CluMSIDdata"))
putativeAQs <- findFragment(annotatedSpeclist, 159.068)
```

`findNL` *Find spectra that contain a specific neutral loss*

Description

`findNL` is used to find spectra that contain a specific neutral loss. Its sister function is `findFragment`, which finds specific fragment ions. Both functions work analogous to `getSpectrum`.

Usage

```
findNL(featlist, mz, tolerance = 1e-05)
```

Arguments

`featlist`	a list that contains only objects of class `MS2spectrum`
`mz`	The mass-to-charge ratio of the neutral loss of interest.
`tolerance`	The *m/z* tolerance for the neutral loss search. Default is `1E-05`, i.e. +/- 10ppm.

Value

If the respective neutral loss is only found in one spectrum, the output is an object of class `MS2spectrum`; if it is found in more than one spectrum, the output is a list of `MS2spectrum` objects.

Examples

```
load(file = system.file("extdata",
    "annotatedSpeclist.RData",
    package = "CluMSIDdata"))
findNL(annotatedSpeclist, 212.009)
```

`getSimilarities` *Match one spectrum against a set of spectra*

Description

`getSimilarities` calculates the similarities of one spectrum or neutral loss pattern to a set of other spectra or neutral loss patterns.

Usage

```
getSimilarities(spec, speclist, type = c("spectrum", "neutral_losses"),
    hits_only = FALSE)
```

Arguments

`spec`	The spectrum to be compared to other spectra. Can be either an object of class `MS2spectrum` or a two-column numerical matrix that contains fragment mass-to-charge ratios in the first and intensities in the second column.
`speclist`	The set of spectra to which `spec` is to be compared. Must be a list where every entry is an object of class `MS2spectrum`. Can be generated from an mzXML file with `extractMS2spectra` and `mergeMS2spectra` or constructed using `new("MS2spectrum", ...)` for every list entry (see vignette for details).
`type`	Specifies whether MS2 spectra or neutral loss patterns are to be compared. Must be either `'spectrum'` (default) or `'neutral_losses'`.
`hits_only`	Logical that indicates whether the result should contain only similarities greater than zero.

Value

A named vector with similarities of `spec` to all spectra or neutral loss patterns in `speclist`.

Examples

```
load(file = system.file("extdata",
    "annotatedSpeclist.RData",
    package = "CluMSIDdata"))
getSimilarities(annotatedSpeclist[[137]],
                annotatedSpeclist, hits_only = TRUE)
```

`getSpectrum` *Access individual spectra from a list of spectra by various slot entries*

Description

As accessing S4 objects within lists is not trivial, `getSpectrum` can be used to access individual or several `MS2spectrum` objects by their slot entries.

Usage

```
getSpectrum(featlist, slot, what, mz.tol = 1e-05, rt.tol = 30)
```

Arguments

`featlist`	a list that contains only objects of class `MS2spectrum`
`slot`	The slot to be searched (invalid slot arguments will produce errors). Possible values are: • `'id'` • `'annotation'` • `'precursor'` (*m/z* of precursor ion) • `'rt'` (retention time of precursor)
`what`	the search term or number, must be character for `'id'` and `'annotation'` and numeric for `'precursor'` and `'rt'` See vignette for examples.
`mz.tol`	the tolerance used for precursor ion *m/z* searches, defaults to `1E-05` (+/-10ppm)
`rt.tol`	the tolerance used for precursor ion retention time searches, defaults to 30s; high values can be used to specify retention time ranges (see vignette for example)

Value

If the only one spectrum matches the search criteria, the output is an object of class `MS2spectrum`; if more than one spectrum matches, the output is a list of `MS2spectrum` objects.

Examples

```
load(file = system.file("extdata",
    "annotatedSpeclist.RData",
    package = "CluMSIDdata"))

getSpectrum(annotatedSpeclist, "annotation", "pyocyanin")

getSpectrum(annotatedSpeclist, "id", "M244.17T796.4")
```

```
getSpectrum(annotatedSpeclist, "precursor", 286.18, mz.tol = 1E-03)

six_eight <- getSpectrum(annotatedSpeclist, "rt", 420, rt.tol = 60)
```

`HCplot` *Generate cluster dendrogram or heatmap from spectral similarity data*

Description

`HCplot()` performs hierarchical clustering of spectral similarity data using average linkage as agglomeration criterion like `HCtbl` and generates either a circular dendrogram or a combination of dendrogram and heatmap.

Usage

```
HCplot(distmat, h = 0.95, type = c("dendrogram", "heatmap"), ...)
```

Arguments

`distmat`	A distance matrix as generated by `distanceMatrix`.
`h`	Height where the tree is to be cut, defaults to `0.95`. See `cutree` for details.
`type`	Specifies which visualisation is to be generated: `"dendrogram"` (default) for a circular dendrogram or `"heatmap"` for a combination of dendrogram and heatmap.
`...`	Additional graphical parameters passed to `plot.phylo` (for `type = "dendrogram"`) or `gplots::heatmap.2` (for `type = "heatmap"`)

Value

A plot as specified by `type`.

Examples

```
load(file = system.file("extdata",
    "distmat.RData",
    package = "CluMSIDdata"))

HCplot(distmat[1:50,1:50], h = 0.8, type = "heatmap")
```

`HCtbl` *Hierarchical clustering of spectral similarity data*

Description

`HCtbl()` performs hierarchical clustering of spectral similarity data using average linkage as agglomeration criterion.

Usage

```
HCtbl(distmat, h = 0.95)
```

Arguments

`distmat`	A distance matrix as generated by `distanceMatrix`.
`h`	Height where the tree is to be cut, defaults to `0.95`. See `cutree` for details.

Value

A `data.frame` with name and cluster ID for each feature in `distmat`.

See Also

`HCplot`

Examples

```
load(file = system.file("extdata",
    "distmat.RData",
    package = "CluMSIDdata"))

my_HCtbl <- HCtbl(distmat[1:50,1:50], h = 0.8)
```

`MDSplot` *Multidimensional scaling of spectral similarity data*

Description

`MDSplot()` is used to generate multidimensional scaling plots from spectral similarity data. An interactive visualisation can be produced using **plotly**.

Usage

```
MDSplot(distmat, interactive = FALSE, highlight_annotated = FALSE, ...)
```

Arguments

`distmat`	A distance matrix as generated by `distanceMatrix`.
`interactive`	Logical, defaults to `FALSE`. If `TRUE`, an interactive visualisation is generated using **plotly**.
`highlight_annotated`	Logical, defaults to `FALSE`. If `TRUE`, points for features for which an annotation was added before using `distanceMatrix` are highlighted by red colour, while other points are grey in the MDS plot.
`...`	Additional arguments passed to `geom_point()`, e.g. `pch`, `size` or `alpha`.

Value

An MDS plot generated with the help of `cmdscale`, `ggplot` and, if interactive, `ggplotly`.

Examples

```
load(file = system.file("extdata",
    "distmat.RData",
    package = "CluMSIDdata"))

MDSplot(distmat, highlight_annotated = TRUE)
```

`mergeMS2spectra` *Merge MS2 spectra with or without external peak table*

Description

`mergeMS2spectra` is used to merge MS2 spectra that come from the same precursor. It does so either by grouping spectra of the same precursor *m/z* that fall into a defined retention time window (`rt_tolerance`) or by grouping spectra to peaks from an externally supplied peak table. Please see the vignette for more details.

Usage

```
mergeMS2spectra(ms2list, mz_tolerance = 1e-05, rt_tolerance = 30,
    peaktable = NULL, exclude_unmatched = FALSE)
```

Arguments

`ms2list`	A `list` of `MS2spectrum` objects to be merged.
`mz_tolerance`	The *m/z* tolerance to be used for merging, default is `1e-5`, i.e. +/- 10ppm. If the mass-to-charge ratios of two peaks differ less than *mz_tolerance*, they are assumed to have the same *m/z*
`rt_tolerance`	The retention time tolerance used for merging features. If used without a peak table, `rt_tolerance` is the maximum retention time difference between to subsequent spectra of the same precursor *m/z* with which they are still assumed to belong to the same feature If used with an external peak table, `rt_tolerance` is the maximum retention time difference between a spectrum and a peak in the peak table with which the spectrum is still considered to belong to that peak.

`peaktable`	An external peak table, e.g. from XCMS, that serves as a template for grouping spectra. The peaktable must be a three-column `data.frame` with feature ID, *m/z* and retention time for each peak/feature.
`exclude_unmatched`	If an external peak table is used: Should spectra that do not match to any peak/feature in the peak table be exclude from the resulting list?

Value

A merged list of `MS2spectrum` objects.

Examples

```
my_spectra <- extractMS2spectra(MSfile = system.file("extdata",
                                "PoolA_R_SE.mzXML",
                                package = "CluMSIDdata"),
                                min_peaks = 4, RTlims = c(0,5))

my_merged_spectra <- mergeMS2spectra(my_spectra, rt_tolerance = 20)
```

`mergeSpecList` *Merge list of spectra*

Description

`mergeSpecList()` is an accessory function used only inside `mergeMS2spectra`.

Usage

```
mergeSpecList(speclist, tolerance = 1e-05)
```

Arguments

`speclist`	A `list` of `MS2spectrum` objects to be merged.
`tolerance`	The m/z tolerance to be used for merging.

Value

A `list` of the same length as `speclist` containing merged spectra as `MS2spectrum` objects. If multiple spectra contribute to one consensus spectrum, than this consensus spectrum is contained in the list multiple times at the respective positions of the contributing spectra.

`mergeTolerance` *Merge spectra with m/z tolerance*

Description

`mergeTolerance()` merges two spectra by identifying common peaks with a given m/z tolerance. It can be used with `Reduce()` to merge more than two spectra.

Usage

```
mergeTolerance(x, y, tolerance = 1e-05)
```

Arguments

`x, y`	MS2 spectra as objects of class `matrix` with m/z in the first column and intensity in the second.
`tolerance`	The m/z tolerance used for merging. If two peaks are within tolerance, they are regarded as the same. Defaults to `1e-5`, i.e. 10ppm.

Value

A matrix with m/z in the first column and separate columns for intensities in the respective spectra. If peaks were merged, their m/z corresponds to the mean of the two original m/z.

`MS2spectrum-class` *A custom S4 class for MS2 spectra, neutral loss patterns and respective metainformation*

Description

A custom S4 class for MS2 spectra, neutral loss patterns and respective metainformation

Usage

```
## S4 method for signature 'MS2spectrum'
show(object)

## S4 method for signature 'MS2spectrum'
precursorMz(object)

## S4 method for signature 'MS2spectrum'
rtime(object)

## S4 method for signature 'MS2spectrum'
intensity(object)

## S4 method for signature 'MS2spectrum'
mz(object)

## S4 method for signature 'MS2spectrum,ANY'
peaksCount(object)
```

Arguments

`object`	An object of class `MS2spectrum`

Value

Prints information from the object slots with exception of 'spectrum' and 'neutral_losses' where only a summary is given.

Methods (by generic)

- `show`: A show generic for `MS2spectra`.
- `precursorMz`: Method for`MSnbase::precursorMz` for `MS2spectrum` objects. Accesses `precursor` slot and returns precursor *m/z* as a numeric.
- `rtime`: Method for`MSnbase::rtime` for `MS2spectrum` objects. Accesses `rt` slot and returns retention time as a numeric.
- `intensity`: Method for`MSnbase::intensity` for `MS2spectrum` objects. Accesses `spectrum` slot and returns the intensity column as a numeric vector.
- `mz`: Method for`MSnbase::mz` for `MS2spectrum` objects. Accesses `spectrum` slot and returns the *m/z* column as a numeric vector.
- `peaksCount`: Method for`MSnbase::mz` for `MS2spectrum` objects. Accesses `spectrum` slot and returns the number of peaks as a numeric.

Slots

`id` a character string similar to the ID used by XCMSonline or the ID given in a predefined peak list

`annotation` a character string containing a user-defined annotation, defaults to empty

`precursor` (median) *m/z* of the spectrum's precursor ion

`rt` (median) retention time of the spectrum's precursor ion

`polarity` the ionisation polarity, `"positive"` or `"negative"`

`spectrum` the actual MS2 spectrum as two-column matrix (column 1 is (median) *m/z*, column 2 is (median) intensity of the product ions)

`neutral_losses` a neutral loss pattern generated by subtracting the product ion mass-to-charge ratios from the precursor *m/z* in a matrix format analogous to the `spectrum` slot

---

`networkplot` *Correlation network from spectral similarity data*

Description

`networkplot()` is used to generate correlation networks from spectral similarity data. An interactive visualisation can be produced using **plotly**.

Usage

```
networkplot(distmat, interactive = FALSE, show_labels = FALSE,
    label_size = 1.5, highlight_annotated = FALSE,
    min_similarity = 0.1, exclude_singletons = FALSE)
```

Arguments

`distmat`	A distance matrix as generated by `distanceMatrix`.
`interactive`	Logical, defaults to `FALSE`. If `TRUE`, an interactive visualisation is generated using **plotly**.
`show_labels`	Logical, defaults to `FALSE`. If `TRUE`, feature IDs are printed as labels in the network plot. Argument has no effect if `interactive` is `TRUE` (because in this case, labels are displayed on mouse-over).
`label_size`	Numeric, defaults to `1.5`. If `show_labels` is `TRUE` and `interactive` is `FALSE`, `label_size` defines the size of labels in the plot.
`highlight_annotated`	Logical, defaults to `FALSE`. If `TRUE`, points for features for which an annotation was added before using `distanceMatrix` are highlighted by red colour, while other points are grey in the network plot.
`min_similarity`	Numeric, defaults to `0.1`. The minimum spectral contrast angle (see`cossim`) that is considered a spectral similarity and hence a connection in the network.
`exclude_singletons`	Logical, defaults to `FALSE`. If `TRUE`, features that have no connection to any other feature will not be displayed in the network plot.

Value

A network plot generated with the help of `network`, `ggnet2` and, if interactive, `ggplotly`. Edge weights correspond to spectral similarities.

Examples

```
load(file = system.file("extdata",
    "distmat.RData",
    package = "CluMSIDdata"))

networkplot(distmat[1:50,1:50], show_labels = TRUE,
                exclude_singletons = TRUE)
```

`neutrallossPatterns` *Generate neutral loss patterns from MS2 spectra*

Description

`neutrallossPatterns` generates neutral loss patterns from MS2 spectra and adds them to `MS2spectrum` objects in the slot `neutral_losses`.

Usage

```
neutrallossPatterns(x)
```

Arguments

`x`	an object of class `MS2spectrum` that contains an MS2 spectrum in the `spectrum` slot

Value

an object of class `MS2spectrum` with a neutral loss pattern in the `neutral_losses` slot

---

`OPTICSplot` *Visualisation of density-based clustering of spectral similarity data*

Description

`OPTICSplot()` performs density-based clustering of spectral similarity data using the OPTICS algorithm like `OPTICStbl` and creates a reachability distance plot.

Usage

```
OPTICSplot(distmat, eps = 10000, minPts = 3, eps_cl = 0.5, ...)
```

Arguments

`distmat`	A distance matrix as generated by `distanceMatrix`.
`eps`	OPTICS parameters, see `optics`.
`minPts`	OPTICS parameters, see `optics`.
`eps_cl`	The reachability distance used for cluster determination, see `extractDBSCAN`.
`...`	Additional graphical parameters to be passed to `plot()`

Details

The function internally uses `optics` and `extractDBSCAN` from the **dbscan** package.

Value

A reachability distance plot as visualisation of OPTICS clustering, see codeextractDBSCAN.

See Also

OPTICStbl

Examples

```
load(file = system.file("extdata",
    "distmat.RData",
    package = "CluMSIDdata"))

OPTICSplot(distmat[1:50,1:50], eps_cl = 0.7)
```

`OPTICStbl` *Density-based clustering of spectral similarity data*

Description

`OPTICStbl()` performs density-based clustering of spectral similarity data using the OPTICS algorithm.

Usage

```
OPTICStbl(distmat, eps = 10000, minPts = 3, eps_cl = 0.5)
```

Arguments

`distmat`	A distance matrix as generated by `distanceMatrix`.
`eps`, `minPts`	OPTICS parameters, see `optics`.
`eps_cl`	The reachability distance used for cluster determination, see `extractDBSCAN`.

Details

The function internally uses `optics` and `extractDBSCAN` from the **dbscan** package.

Value

A `data.frame` with feature name, cluster ID and OPTICS order for each feature in `distmat`.

See Also

OPTICSplot

Examples

```
load(file = system.file("extdata",
    "distmat.RData",
    package = "CluMSIDdata"))

my_OTPICStbl <- OPTICStbl(distmat[1:50,1:50], eps_cl = 0.7)
```

`pseudospectrum-class` *A custom S4 class for MS1 pseudospectra and respective metainformation*

Description

A custom S4 class for MS1 pseudospectra and respective metainformation

Slots

- `id` a the `"pcgroup"` number assigned by **CAMERA**
- `annotation` a character string containing a user-defined annotation, defaults to empty
- `rt` (median) retention time of the ions contained in the pseudospectrum
- `spectrum` the actual MS1 pseudospectrum as two-column matrix (column 1 is (median) *m/z*, column 2 is (median) intensity of the ions)

`specplot` *Create a basic plot of MS2 spectra*

Description

`specplot` creates a very basic plot of MS2 spectra from `MS2spectrum` or `pseudospectrum` objects.

Usage

```
specplot(spec, ...)
```

Arguments

`spec`	An object of class `MS2spectrum` or `pseudospectrum`
`...`	Additional graphical parameters to be passed to `plot()`

Value

A plot of the MS2 spectrum saved in the `spectrum` slot of `spec`.

Examples

```
load(file = system.file("extdata",
    "annotatedSpeclist.RData",
    package = "CluMSIDdata"))

specplot(annotatedSpeclist[[1]])
```

`splitPolarities` *Separate spectra with different polarities from the same run*

Description

Using `splitPolarities`, spectra with different polarities from the same run can be separated, e.g. when processing spectra recorded with polarity-switching.

Usage

```
splitPolarities(ms2list, polarity = c("positive", "negative"))
```

Arguments

`ms2list`	A list of `MS2spectrum` objects as produced by `extractMS2spectra`.
`polarity`	The polarity of spectra to be analysed, must be `"positive"` or `"negative"`.

Value

A list of `MS2spectrum` objects that contains only spectra with the given `polarity`.

Examples

```
my_spectra <- extractMS2spectra(MSfile = system.file("extdata",
                                "PoolA_R_SE.mzXML",
                                package = "CluMSIDdata"),
                                min_peaks = 4, RTlims = c(0,5))

my_positive_spectra <- splitPolarities(my_spectra, "positive")
```

`writeFeaturelist` *Write feature information from list of* `MS2spectrum` *objects*

Description

`writeFeaturelist` uses `featureList` to generate a `data.frame` that contains feature ID, precurosur *m/z* and retention time for all features contained in a list of `MS2spectrum` objects as produced by `extractMS2spectra` and `mergeSpecList` and writes it to a csv file.

Usage

```
writeFeaturelist(featlist, filename = "pre_anno.csv")
```

Arguments

`featlist`	A list of `MS2spectrum` objects as produced by `extractMS2spectra` and `mergeSpecList`
`filename`	The desired file name of the csv file, default is `"pre_anno.csv"`

Details

Although originally designed for lists of `MS2spectrum` objects, the function also works with lists of `pseudospectrum` objects. In this case, `NA` is given for precursor *m/z*.

Value

A csv file that contains feature ID, precurosur *m/z* and retention time. The file has a header but no row names and is separated by `','`.

Examples

```
load(file = system.file("extdata",
    "featlist.RData",
    package = "CluMSIDdata"))

writeFeaturelist(featlist, filename = "pre_anno.csv")
```

Index

∗ **internal**
    mergeSpecList, 15
    mergeTolerance, 16
    neutrallossPatterns, 18

accessAnnotation (accessors), 2
accessID (accessors), 2
accessNeutralLosses (accessors), 2
accessors, 2
accessPolarity (accessors), 2
accessPrecursor (accessors), 2
accessRT (accessors), 2
accessSpectrum (accessors), 2
addAnnotations, 4
as.MS2spectrum, 4

cmdscale, *14*
cossim, 5, *6*, *18*
cossim,MS2spectrum,MS2spectrum-method (cossim), 5
cossim,pseudospectrum,pseudospectrum-method (cossim), 5
cutree, *12, 13*

distanceMatrix, 6, *12–14*, *18–20*

extractDBSCAN, *19, 20*
extractMS2spectra, *6*, 7, *8*, *10*, *22*
extractPseudospectra, *6*, 7

featureList, 8, *22*
findFragment, *9*, 9
findNL, *9*, 9

getSimilarities, 10
getSpectrum, *9*, 11
ggnet2, *18*
ggplot, *14*
ggplotly, *14*, *18*

HCplot, 12, *13*
HCtbl, *12*, 13

intensity,MS2spectrum-method (MS2spectrum-class), 16

MDSplot, 13
mergeMS2spectra, *10*, 14
mergeSpecList, 15
mergeTolerance, 16
MS2spectrum, *2, 3, 5, 6*, *9–11*, *15*, *17–19*, *21, 22*
MS2spectrum-class, 16
mz,MS2spectrum-method (MS2spectrum-class), 16

network, *18*
networkplot, 17
neutrallossPatterns, 18

optics, *19, 20*
OPTICSplot, 19
OPTICStbl, *19*, 20

peaksCount,MS2spectrum,ANY-method (MS2spectrum-class), 16
precursorMz,MS2spectrum-method (MS2spectrum-class), 16
pseudospectrum, *2, 3, 5, 6*, *8*, *21*
pseudospectrum-class, 20

rtime,MS2spectrum-method (MS2spectrum-class), 16

show,MS2spectrum-method (MS2spectrum-class), 16
specplot, 21
Spectrum, *5*
Spectrum2, *5*
splitPolarities, 21

writeFeaturelist, *4*, *8*, 22

xsAnnotate, *8*