Package 'rtms'

May 9, 2026

Title R Toolkit for Mass Spectrometry

Version 0.2.0

Description Quickly imports, processes, analyzes, and visualizes mass-spectrometric data. Includes functions for easily extracting specific data and measurements from large (multi-gigabyte) raw Bruker data files, as well as a set of S3 object classes for manipulating and measuring mass spectrometric peaks and plotting peaks and spectra using the 'ggplot2' package.

Imports stats, ggplot2

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.2.3

Depends R (>= 2.10)

LazyData true

NeedsCompilation no

Author Mary Ashley Rimmer [aut],
Nathaniel Twarog [aut, cre]

Maintainer Nathaniel Twarog <nathaniel.twarog@stjude.org>

Repository CRAN

Date/Publication 2023-06-06 07:00:02 UTC

Contents

`emptySampleSet` *Generate an empty RTMS sample set*

Description

Produces a sample set (of class `rtmsSampleSet`) with no samples but a specific peaks attribute. Useful for building a sample set one sample at a time.

Usage

```
emptySampleSet(peaks)
```

Arguments

`peaks`	A list of objects of class `rtmsPeak`

Value

An empty object of class `rtmsSampleSet`

exampleSpectrum *Simple Example Spectrum*

Description

A subset of an example spectrum depicting turnover from a protein substrate (at 1530.8 m/z) to product (at 1516.8 m/z). While the original spectrum contained nearly 2 million measurements, this spectrum has been trimmed to lie between 1300 m/z and 1600 m/z, with approximately 46000 measurements included.

Usage

```
exampleSpectrum
```

Format

`exampleSpectrum`**:**

An object of class `rtmsSpectrum`

Source

<St. Jude Children's Research Hospital>

getBrukerBAFAllMetadata *Retrieve all metadata values from a Bruker BAF file*

Description

Retrieves a table of all metadata values (including instrument data, acquisition parameters, processing and analysis directives, etc.) from a Bruker single acquisition BAF directory (represented by an `rtmsBrukerBAFReader` object).

Usage

```
getBrukerBAFAllMetadata(reader)
```

Arguments

`reader`	An RTMS reader object of class `rtmsBrukerBAFReader`

Value

A data frame with all metadata parameters for the acquisition. The data frame will have five columns: `rowIndex`, a numeric index for each metadata value; `parameterName`, the internal identifier of the parameter in Bruker software systems; `parameterGroup`, the group of parameters that each value belongs to; `displayName`, the string used to specify the parameter to users (i.e. how the parameter would be labelled in a user interface); and `stringValue`, a character column containing the value of each metadata parameter. Numeric quantities will also be returned as strings, with units if appropriate.

`getBrukerBAFMetadata` *Retrieve specific metadata values from a Bruker BAF file*

Description

Retrieves a list of specific metadata values (including instrument data, acquisition parameters, processing and analysis directives, etc.) from a Bruker single acquisition BAF directory (represented by an `rtmsBrukerBAFReader` object).

Usage

```
getBrukerBAFMetadata(reader, names)
```

Arguments

`reader`	An RTMS reader object of class `rtmsBrukerBAFReader`
`names`	A character vector of metadata names

Value

A named list of values corresponding to the metadata values specified. All values will be returned as a string, including numeric quantities (with units if appropriate).

`getBrukerMCFAllMetadata` *Retrieve all metadata values from a Bruker MCF file*

Description

Retrieves a table of all metadata values (including instrument data, acquisition parameters, processing and analysis directives, etc.) for a specific acquisition from a Bruker multi-acquisition MCF directory (represented by an `rtmsBrukerMCFReader` object).

Usage

```
getBrukerMCFAllMetadata(reader, index)
```

Arguments

reader	An RTMS reader object of class rtmsBrukerMCFReader
index	A single numeric index specifying which acquisition the sample set should be extracted from

Value

A data frame with all metadata parameters for the acquisition. The data frame will have five columns: Index, the numeric index of the acquisition; PermanentName, the internal identifier of the parameter in Bruker software systems; GroupName, the group of parameters that each value belongs to; DisplayName, the string used to specify the parameter to users (i.e. how the parameter would be labelled in a user interface); and Value, a character column containing the value of each metadata parameter. Numeric quantities will also be returned as strings, with units if appropriate.

---

getBrukerMCFIntensities

Retrieve peak intensities directly from an MCF file

Description

The size of mass spectrometric data in general, and Bruker MCF directories specifically, makes the extraction of data a resource intensive and time consuming process. rtms as a package is designed to reduce this burden, but pulling a sample set from an MCF file can (in the event of compressed spectra) requires reading nearly all data out of a file, which could take an extremely long time over a network connection. Since peak intensity (calculated as the sum of local intensity maxima within a given peak width) is one of the most common measurements used in evaluating spectra, and because this measure can be extracted without extracting the full spectra, this function aims to avoid expensive reading time by skipping the creation of a sample set object and calculating peak intensity directly. Other measurements are not possible, but full spectra do not have to be read, even when spectra are compressed, as local maxima are pre-processed and stored separately in a Bruker MCF file.

Usage

```
getBrukerMCFIntensities(reader, peaks, indices)
```

Arguments

reader	An RTMS reader object of class rtmsBrukerMCFReader
peaks	A list of peak objects of class rtmsPeak
indices	A vector of numeric indices specifying which acquisitions the measurements should be taken from

Value

A data frame containing columns specifying the index of each acquisition, the name of each acquistion (if `indices` is a named vector), the peak value of the peak measure, the peak name (if `peaks` is a named list), the measure name (which will always be "PeakIntensity"), and the measured value for that sample and peak. Format matches the output of `measureSampleSet`

`getBrukerMCFMetadata` *Retrieve specific metadata values from a Bruker BAF file*

Description

Retrieves a list of specific metadata values (including instrument data, acquisition parameters, processing and analysis directives, etc.) for a specific acquisition from from a Bruker multi-acquisition BAF directory (represented by an `rtmsBrukerBAFReader` object).

Usage

```
getBrukerMCFMetadata(reader, names, index)
```

Arguments

`reader`	An RTMS reader object of class `rtmsBrukerMCFReader`
`names`	A character vector of metadata names
`index`	A single numeric index specifying which acquisition the sample set should be extracted from

Value

A named list of values corresponding to the metadata values specified. All values will be returned as a string, including numeric quantities (with units if appropriate).

`getBrukerMCFSpots` *Get spot names and indices from a Bruker MCF file*

Description

Assembles a table of all acquisitions in a Bruker MCF file; Bruker measurements are often identified by the metadata parameter "Spot Number", so this function extracts that specific metadata value and joins it with the indices used to pick out spectra in other functions. Also retrieves the timestamp at which acquisition was taken, if acquisitions must be identified by order.

Usage

```
getBrukerMCFSpots(reader)

getBrukerMCFIndices(reader)
```

Arguments

reader	An openRTMS reader object of class rtmsBrukerMCFReader

Value

A data.frame with an Index column containing the indices of each acquisition (used by other functions such as getSpectrum or getSample), a SpotNumber column containing the "Spot Number" metadata value for each acquisition, and a Timestampe column containing the time at which each acquisition was collected by the instrument.

Functions

- getBrukerMCFIndices(): Retrieves a vector of all the indices (beginning with zero) of the acquisitions in the MCF file. Faster than getBrukerMCFSpots but contains no metadata or spot names

---

getSample *Get an RTMS Sample*

Description

Fetches an object of class rtmsSample from the specified object.

Usage

```
getSample(x, peaks, ...)
```

Arguments

x	The object from which the sample should be retrieved
peaks	A list of objects of class rtmsPeak
...	Other possible arguments to specify a particular sample to be retrieved

Value

A sample object of class rtmsSample

`getSample.rtmsBrukerBAFReader`

Extract a sample from a Bruker BAF directory

Description

Extracts an RTMS sample object (of class `rtmsSample`) from a single acquisition Bruker BAF directory opened using an RTMS reader object (of class `rtmsBrukerBAFReader`). Because a BAF directory only contains one spectrum, no additional parameters are needed to specify the spectrum from which to take the sample.

Usage

```
## S3 method for class 'rtmsBrukerBAFReader'
getSample(x, peaks, ...)

getBrukerBAFSample(reader, peaks)
```

Arguments

`x`	The BAF reader object
`peaks`	A list of peak objects of class `rtmsPeak`
`...`	Additional parameters
`reader`	An RTMS reader object of class `rtmsBrukerBAFReader`

Value

An RTMS sample object of class `rtmsSample`

Functions

- `getSample(rtmsBrukerBAFReader)`: The S3 method `getSample` for objects of class `rtmsBrukerBAFReader`; calls `getBrukerBAFSample`

`getSample.rtmsBrukerMCFReader`

Extract a sample from a Bruker MCF directory

Description

Extracts an RTMS sample object (of class `rtmsSample`) from a multi- acquisition Bruker MCF directory opened using an RTMS reader object (of class `rtmsBrukerMCFReader`). A numeric index is used to identify which acquisition the sample should be extracted from.

Usage

```
## S3 method for class 'rtmsBrukerMCFReader'
getSample(x, peaks, ...)

getBrukerMCFSample(reader, peaks, index)
```

Arguments

x	The MCF reader object
peaks	A list of peak objects of class rtmsPeak
...	Additional parameters
reader	An RTMS reader object of class rtmsBrukerMCFReader
index	A single numeric index specifying which acquisition the sample set should be extracted from

Value

An RTMS sample object of class rtmsSample

Functions

- getSample(rtmsBrukerMCFReader): The S3 method getSample for objects of class rtmsBrukerMCFReader; calls getBrukerMCFSample

---

getSample.rtmsSpectrum

Extract a sample from an RTMS spectrum object

Description

Extracts a sample object of class rtmsSample from a single spectrum object of class rtmsSample usin g a list of peaks

Usage

```
## S3 method for class 'rtmsSpectrum'
getSample(x, peaks, ...)

getSampleFromSpectrum(spectrum, peaks, freqSpacing = TRUE, threshold = NULL)
```

Arguments

x	The spectrum object
peaks	A list of peak objects of class rtmsPeak
...	Additional parameters
spectrum	A full spectrum of class rtmsSpectrum
freqSpacing	If TRUE (the default), local maxima (estimated via quadratic interpolation) are calculated in inverse m/z (or frequency) space, as in FTMS spectra. If FALSE, maxima are calculated directy in m/z space
threshold	If NULL, all local maxima will be returned for each subsample; if set to particular value, only those maxima above that threshold will be returned.

Value

An RTMS sample object of class rtmsSample

Functions

- getSample(rtmsSpectrum): The S3 method getSample for objects of class rtmsSpectrum; calls getSampleFromSpectrum

Examples

```
peaks <- rtmsPeakList(c(1516.83,1530.84),peakWidth=0.2,windowWidth = c(5,10))
names(peaks) <- c("Product","Substrate")
sample <- getSample(exampleSpectrum,peaks)
```

---

getSampleSet *Get an RTMS Sample Set*

Description

Fetches an object of class rtmsSampleSet from the specified object.

Usage

```
getSampleSet(x, peaks, ...)
```

Arguments

x	The object from which the sample set should be retrieved
peaks	A list of objects of class rtmsPeak
...	Other possible arguments to specify a particular sample set to be retrieved

Value

A sample set object of class rtmsSampleSet

getSampleSet.rtmsBrukerMCFReader

Extract a sample set from a Bruker MCF directory

Description

Extracts an RTMS sample object (of class `rtmsSampleSet`) from a multi- acquisition Bruker MCF directory opened using an RTMS reader object (of class `rtmsBrukerMCFReader`). A vector numeric indices is used to identify which acquisitions the sample set should be extracted from.

Usage

```
## S3 method for class 'rtmsBrukerMCFReader'
getSampleSet(x, peaks, ...)

getBrukerMCFSampleSet(reader, peaks, indices)
```

Arguments

`x`	The MCF reader object
`peaks`	A list of peak objects of class `rtmsPeak`
`...`	Additional parameters
`reader`	An RTMS reader object of class `rtmsBrukerMCFReader`
`indices`	A vector of numeric indices specifying which acquisitions the sample set should be extracted from

Value

An RTMS sample set object of class `rtmsSampleSet`

Functions

- `getSampleSet(rtmsBrukerMCFReader)`: The S3 method `getSample` for objects of class `rtmsBrukerMCFReader`; calls `getBrukerMCFSampleSet`

getSpectrum

Get an RTMS Spectrum

Description

Fetches an object of class `rtmsSpectrum` from the specified object.

Usage

```
getSpectrum(x, ...)
```

Arguments

x	The object from which the spectrum should be retrieved
...	Other possible arguments to specify a particular spectrum to be retrieved

Value

An spectrum object of class `rtmsSpectrum`

`getSpectrum.rtmsBrukerBAFReader` *Extract a spectrum from a Bruker BAF directory*

Description

Extracts an RTMS spectrum object (of class `rtmsSpectrum`) from a single acquisition Bruker BAF directory opened using an RTMS reader object (of class `rtmsBrukerBAFReader`). Because a BAF directory only contains one spectrum, no additional parameters are needed to specify the spectrum to be extracted.

Usage

```
## S3 method for class 'rtmsBrukerBAFReader'
getSpectrum(x, ...)

getBrukerBAFSpectrum(reader)
```

Arguments

x	The BAF reader object
...	Additional parameters
reader	An RTMS reader object of class `rtmsBrukerBAFReader`

Value

An RTMS spectrum object of class `rtmsSpectrum`

Functions

- `getSpectrum(rtmsBrukerBAFReader)`: The S3 method `getSpectrum` for objects of class `rtmsBrukerBAFReader`; calls `getBrukerBAFSpectrum`

getSpectrum.rtmsBrukerMCFReader *Extract a spectrum from a Bruker MCF directory*

Description

Extracts an RTMS spectrum object (of class `rtmsSpectrum`) from a multi- acquisition Bruker MCF directory opened using an RTMS reader object (of class `rtmsBrukerMCFReader`). A numeric index is used to identify which spectrum should be extracted.

Usage

```
## S3 method for class 'rtmsBrukerMCFReader'
getSpectrum(x, ...)

getBrukerMCFSpectrum(reader, index)
```

Arguments

`x`	The MCF reader object
`...`	Additional parameters
`reader`	An RTMS reader object of class `rtmsBrukerMCFReader`
`index`	A single numeric index specifying which acquisition should be extracted

Value

An RTMS spectrum object of class `rtmsSpectrum`

Functions

- `getSpectrum(rtmsBrukerMCFReader)`: The S3 method `getSpectrum` for objects of class `rtmsBrukerMCFReader`; calls `getBrukerMCFSpectrum`

measureSample *Measure peaks in an RTMS sample*

Description

`measureSample()` extracts one or more measurements for every peak in an RTMS sample object (of class `rtmsSample`).

Usage

```
measureSample(sample, measure = "PeakIntensity")
```

Arguments

sample	An object of class rtmsSample
measure	A character vector of named measurements, or a list of custom measurement functions. Supported measurement names are "PeakIntensity", which takes the total of any local maxima within the peak width, "PeakArea", which takes the area under the intensity curve within the peak width, and "NumPeaks", which counts the local maxima in the peak window. If measure is a list of functions, each function must take an object of class rtmsSubsample, and return a single numeric value. If the functions are named, those names will be returned in the "measure" column of the resulting data frame; otherwise they will be identified as "Measure1", "Measure2", etc.

Value

A data frame with one row for each peak and measurement in the sample. The data.frame will have a column named "peakName" with the name of the relevant peak (if the "peaks" attribute of sample is a named list); a column named "peakValue" containing the m/z value at the center of the relevant peak; a column named "measure" containing the name of the relevant measure; and a column named "value" containing the numeric value of the particular measure for that peak.

Examples

```
peaks <- rtmsPeakList(c(1516.83,1530.84),peakWidth=0.2,windowWidth = c(5,10))
names(peaks) <- c("Product","Substrate")
sample <- getSample(exampleSpectrum,peaks)

measure <- measureSample(sample,c("PeakArea","PeakIntensity"))

myFunctions <- list(PeakRawIntensity = function(s) max(s$peakPiece$intensity))
myMeasures <- measureSample(sample,myFunctions)
```

---

measureSampleSet *Measure peaks and samples in an RTMS sample set*

Description

measureSampleSet() extracts one or more measurements for every peak in every sample in an RTMS sample set object (of class rtmsSampleSet).

Usage

```
measureSampleSet(sampleset, measure = "PeakIntensity")
```

Arguments

sampleset	An object of class rtmsSampleSet
measure	A character vector of named measurements, or a list of custom measurement functions. Supported measurement names are "PeakIntensity", which takes the total of any local maxima within the peak width, "PeakArea", which takes the area under the intensity curve within the peak width, and "NumPeaks", which counts the local maxima in the peak window. If measure is a list of functions, each function must take an object of class rtmsSubsample, and return a single numeric value. If the functions are named, those names will be returned in the "measure" column of the resulting data frame; otherwise they will be identified as "Measure1", "Measure2", etc.

Value

A data frame with one row for each sample, peak, and measurement. The data.frame will have a character column named "sample", containing either the name of the sample (if the samples in sampleset are named) or the index of the sample if they are not (but it will always be a character column); a column named "peakName" with the name of the relevant peak (if the "peaks" attribute of sampleset is a named list); a column named "peakValue" containing the m/z value at the center of the relevant peak; a column named "measure" containing the name of the relevant measure; and a column named "value" containing the numeric value of the particular measure for that sample and peak.

Examples

```
peaks <- rtmsPeakList(c(1516.83,1530.84),peakWidth=0.2,windowWidth = c(5,10))
names(peaks) <- c("Product","Substrate")
sample <- getSample(exampleSpectrum,peaks)
sampleSet <- rep(sample,3)
names(sampleSet) <- c("A","B","C")

measures <- measureSampleSet(sampleSet)
```

---

newBrukerBAFReader *Open a Bruker single-acquisition BAF directory*

Description

Creates an RTMS reader object (of class rtmsBrukerBAFReader) which can extract data from a Bruker single acquisition directory (extension ".d")

Usage

```
newBrukerBAFReader(bafdir)
```

Arguments

bafdir	A directory (usually with the extension ".d") containing data from a single Bruker acquisition. This directory will contain a file with extension ".baf" that holds the primary raw data, as well as an index and calibration file (see Details).

Details

Currently, RTMS can create reader objects for two binary Bruker data formats, BAF (presumably standing for "Bruker acquisition format") holding data from a single spectrum acquisition, and MCF (probably "multiacquisition container format") containing data from multiple spectra acquired in a single run. Both formats hold data in a directory marked with the extension ".d". The single acquisition BAF format directory contains three essential data files: the main raw data file with extension ".baf", an index file with extension ".baf_idx", and a calibration data file with extension ".baf_xtr". This function processes the index and calibration files so that raw data can be extracted quickly on demand from the ".baf" file.

An important note: when a MCF multi-acquistion reader is created, it creates an open connection to the raw data file, which allows for quicker processing of many spectra in a single file. However, because a BAF file contains only a single spectrum, there is little advantage to maintaining an open connection, so the connection is re-opened every time data is read. Thus, while it is important to close an MCF reader object when all data is extracted, it is not necessary to close an object of class `rtmsBrukerBAFReader`.

Value

An object of class `rtmsBrukerBAFReader` which can extract raw data from the specified directory

---

newBrukerMCFReader *Open a Bruker multi-acquisition MCF directory*

Description

Creates an RTMS reader object (of class `rtmsBrukerMCFReader`) which can extract data from a Bruker multi-acquisition directory (extension ".d")

Usage

```
newBrukerMCFReader(mcfdir)
```

Arguments

mcfdir	A directory (usually with the extension ".d") containing data from a Bruker multi-acquisition run. This directory will contain a files with extenssion ".mcf" and matching index files with extension ".mcf_idx" (see Details).

Details

Currently, RTMS can create reader objects for two binary Bruker data formats, BAF (presumably standing for "Bruker acquisition format") holding data from a single spectrum acquisition, and MCF (probably "multi-acquisition container format") containing data from multiple spectra acquired in a single run. Both formats hold data in a directory marked with the extension ".d". The multi-acquisition MCF format directory contains four essential data files: the main raw data file ending in "_1" with extension ".mcf", a matching index file with extension ".mcf_idx", and a calibration data file ending in "_2" with extension ".mcf", and the matching calibration index file with extension ".mcf_idx". This function preprocesses the main index file and calibration files so that raw data can be extracted quickly on demand from the main ".mcf" file.

An important note: when a MCF multi-acquisition reader is created, it creates an open connection to the raw data file, which allows for quicker processing of many spectra in a single file. This connection will remain open until the reader is closed with the `close` function.

Value

A reader object of class `rtmsBrukerMCFReader` with an open connection to the main ".mcf" data file.

---

plotRtmsSample *Plot an RTMS sample object*

Description

`plotRtmsSample()` takes an RTMS sample object and produces a `ggplot` object depicting all extracted peaks, and their context windows if included.

Usage

```
plotRtmsSample(sample, usePeakNames = TRUE, freey = TRUE)
```

Arguments

`sample`	An object of class `rtmsSample`.
`usePeakNames`	If the list of peaks used to create the sample was a named list, then setting this to TRUE (the default) will use those names to label the facets of the plotted sample. If set to FALSE, the facets will be labelled with the m/z values of each peak. This parameter will be ignored if the peaks are unnamed.
`freey`	If TRUE (the default) the y-axes of each peak's facet will be allowed to vary freely, so different peaks will be plotted on different scales. Setting this to FALSE will fix all peaks with in a sample on the same y-axis scale.

Value

A `ggplot` object depicting the RTMS sample.

Examples

```
peaks <- rtmsPeakList(c(1516.83,1530.84),peakWidth=0.2,windowWidth = c(5,10))
names(peaks) <- c("Product","Substrate")
sample <- getSample(exampleSpectrum,peaks)

plot1 <- plotRtmsSample(sample)
plot2 <- plotRtmsSample(sample,freey=FALSE)
```

`plotRtmsSampleSet` *Plot an RTMS sample set object*

Description

`plotRtmsSampleSet()` takes an RTMS sample set object and produces a `ggplot` object depicting all extracted peaks, and their context windows if included.

Usage

```
plotRtmsSampleSet(sampleset, usePeakNames = TRUE, freey = TRUE)
```

Arguments

`sampleset`	An object of class `rtmsSampleSet`.
`usePeakNames`	If the list of peaks used to create the sample set was a named list, then setting this to TRUE (the default) will use those names to label the facets of the plotted sample set. If set to FALSE, the facets will be labelled with the m/z values of each peak. This parameter will be ignored if the peaks are unnamed.
`freey`	If TRUE (the default) the y-axes of each sample and peak's facet will beallowed to vary freely, so different facets will be plotted on different scales. Setting this to FALSE will fix all peaks and samples on the same y-axis scale.

Value

A `ggplot` object depicting the RTMS sample set.

Examples

```
peaks <- rtmsPeakList(c(1516.83,1530.84),peakWidth=0.2,windowWidth = c(5,10))
names(peaks) <- c("Product","Substrate")
sample <- getSample(exampleSpectrum,peaks)
sampleSet <- rep(sample,3)
names(sampleSet) <- c("A","B","C")

plot1 <- plotRtmsSampleSet(sampleSet)
plot2 <- plotRtmsSampleSet(sampleSet,freey=FALSE) + ggplot2::theme_bw()
```

plotRtmsSpectrum *Plot an RTMS spectrum object*

Description

plotRtmsSpectrum() takes an RTMS spectrum object and produces a ggplot object depicting the spectrum

Usage

```
plotRtmsSpectrum(spectrum, limits = NULL)
```

Arguments

spectrum	An object of class rtmsSpectrum.
limits	An optional parameter to control the bounds of the m/z x axis. If set to NULL, the default, the full spectrum will be plotted. Otherwise, limits should be a two element numeric vector; if one element is NA, then only the other boundary will be enforced on the x-axis.

Details

Unlike a sample object, an RTMS spectrum is actually quite simple; just a vector of m/z values and vector of intensities. Ordinarily, this could be done using standard ggplot2 functions, such as geom_line. However, mass spectra can often be quite large (on the order of millions of measurements), and sending all that data to be plotted can be computationally intractable. plotRtmsSpectrum() therefore selects a subset of up to 10000 m/z-intensity pairs from the original spectrum to produce a representative plot without rendering millions of points. Any points that are sufficiently larger than their local surroundings (including all relevant peaks) will be included in this subset, as well as a random sampling of points closer to the baseline. This ensures that the peaks plotted will always be present. However, there will be slight differences from one plot to the next in terms of baseline points plotted. This can be eliminated by fixing the random seed using set.seed before plotting.

We also strongly discourage using xlim or seetting the x-coordinate boundaries using standard ggplot2 methods, as these will only be applied after the data has been down-sampled. If you would like to plot a particular subset of the spectrum, it is recommended that you use the limits parameter of this function instead.

Value

A ggplot object depicting the RTMS spectrum.

Examples

```
plot1 <- plotRtmsSpectrum(exampleSpectrum)
plot2 <- plotRtmsSpectrum(exampleSpectrum,limits=c(1500,1550)) +
ggplot2::geom_vline(xintercept=c(1516.83,1530.84),
colour="red",linetype=2)
```

reopen *Reopen a reader object*

Description

An S3 generic function for reopening reader objects that have been close

Usage

```
reopen(x, ...)
```

Arguments

x	The object to be re-opened
...	Other possible arguments for specific object types

Value

An object of the same type as x

reopen.rtmsBrukerMCFReader *Manage an MCF reader file connection*

Description

Closes the open connection to the main data file in a Bruker MCF reader object.

Usage

```
## S3 method for class 'rtmsBrukerMCFReader'
reopen(x, ...)

## S3 method for class 'rtmsBrukerMCFReader'
close(con, ...)

closeBrukerMCFReader(reader)

reopenBrukerMCFReader(reader)
```

Arguments

x	The reader object to reopen
...	Included for S3 compatibility
con	The reader object to be closed
reader	An RTMS reader object of class rtmsBrukerMCFReader

Details

Because Bruker MCF directories contain a potentially large number of spectra, reopening a connection to the main data file when reading many spectra or samples from it is inefficient and slow, especially if the file is being accessed over a network connection. The `rtmsBrukerMCFReader` object therefore maintains an open connection to the main binary data file until it is closed by the user. Of course, the reader object still maintains all the index and calibration data, making it possible to reopen a connection to the MCF directory without all the preprocessing required when first opening. Unfortunately, taking advantage of this fact is a little tricky.

In most cases, R is a functional language, with limited side effects on R objects; so it is difficult to alter the state of reader object without returning it explicitly. However, one of the few cases where side-effects are quite important is R's management of open file connections, with functions like `close`. Thus, calling `closeBrukerMCFReader` (and the S3 function `close` which wraps it) will in fact close the connection, but will not return an altered copy of the reader object reflecting that it is closed. So if the user wishes to close a reader object with the possibility of reopening it, they must close the reader AND assign the returned reader object to the relevant name. This will store the fact the connection has been closed, and allow the `reopen` function to operate correctly.

Value

The same reader object with a closed connection

Functions

- `reopen(rtmsBrukerMCFReader)`: The S3 method `close` for objects of class `rtmsBrukerMCFReader`; calls `closeBrukerMCFReader`
- `close(rtmsBrukerMCFReader)`: The S3 method `reopen` for objects of class `rtmsBrukerMCFReader`; calls `reopenBrukerMCFReader`
- `reopenBrukerMCFReader()`: Reopens a file connection to the main binary data file in a Bruker MCF directgry so that data can be extracted

---

`rtmsPeak` *Create an RTMS m/z Peak Object*

Description

Generates an object of class `rtmsPeak` which contains the m/z values bounding a spectrometric peak to be measured. A peak object specifies not only the m/z value at the cetner of the peak, but the upper and lower bounds within which the peak is to be quantified; it also may optionally include wider upper and lower bounds used to plot the peak in a wider context of the spectrum.

Usage

```
rtmsPeak(
  value,
  peakWidth = 0.1,
  windowWidth = NULL,
```

```
  bounds = NULL,
  window = NULL
)
```

Arguments

value	The m/z value that the peak is intended to measure
peakWidth	The width of the peak centered on value. If a single numeric value, the lower bound of the peak will lie peakWidth/2 below value,and the upper bound will lie peakWidth/2 above value. If a vector of two numeric values, the first value specifies how far below value the lower bound lies, and the second value specifies how far above value the upper bound lies. If parameter bounds is not null, this parameter will be ignored.
windowWidth	The width of the optional wider window around value used to show the peak in context. Operates by the same principles as peakWidth with a single value split evenly between lower and upper bounds, and two values specifying how far below and above value each bound lies. If parameter window is not null, this parameter will be ignored.
bounds	If not null, a two-value numeric vector specifying the lower and upper m/z bounds of the measured peak. One of bounds or peakWidth must be not null, and if bounds is not null, then peakWidth will be ignored.
window	If not null, a two-value numeric vector specifying the lower and upper m/z bounds of the wider context window of the peak. If window is not null, then windowWidth will be ignored.

Value

An object of class rtmsPeak, used by RTMS functions to extract and measure peaks from mass spectra.

Examples

```
peaks <- rtmsPeak(1516.83,peakWidth=0.2,windowWidth = c(5,10))
```

---

rtmsPeakList *Create a list of RTMS m/z peak objects*

Description

Generates a list of objects of class rtmsPeak which can be used to extract a sample or sample set from other RTMS objects.

Usage

```
rtmsPeakList(values, peakWidth = 0.1, windowWidth = NULL)
```

Arguments

values	The m/z values that the peaks is intended to measure
peakWidth	The width of each peak centered on values. If a single numeric value, the lower bound of each peak will lie peakWidth/2 below the given m/z value ,and the upper bound will lie peakWidth/2 above it. If a vectorof two numeric values, the first value specifies how far below each given m/z value the lower bounds lie, and the second value specifies how far above each value the upper bounds lie.
windowWidth	The width of each optional wider window around the m/z values used to show the peaks in context. Operates by the same principles as peakWidth with a single value split evenly between lower and upper bounds, and two values specifying how far below and above the m/z values each bound lies.

Value

A list of objects of class rtmsPeak

Examples

```
peaks <- rtmsPeakList(c(1516.83,1530.84),peakWidth=0.2,windowWidth = c(5,10))
names(peaks) <- c("Product","Substrate")
```

---

rtmsSpectrum *Create a new RTMS spectrum*

Description

Generates an RTMS spectrum object (of class rtmsSpectrum) from a given vector of m/z and intensity values.

Usage

```
rtmsSpectrum(mz, intensity)
```

Arguments

mz	A numeric vector of m/z values
intensity	A numeric vector of intensity values

Value

An object of class rtmsSpectrum with the given m/z and intensity values

sampleAndSampleSet *Functions for creating and manipulating samples and sample sets*

Description

Select a subset of a sample set (returns an rtmsSampleSet)

Usage

```
## S3 method for class 'rtmsSampleSet'
x[i, ...]

## S3 method for class 'rtmsSampleSet'
x[[i, ...]]

## S3 replacement method for class 'rtmsSampleSet'
x[[i]] <- value

## S3 method for class 'rtmsSampleSet'
rep(x, ...)

## S3 method for class 'rtmsSample'
rep(x, ...)
```

Arguments

x	An object of class rtmsSample
i	A single numeric index of the sample set
...	Included for S3 compatibility
value	An object of class rtmsSample

Details

The sample (class rtmsSample) and sample set (class rtmsSampleSet) objects are the core structures used to extract meaningful data from mass spectographic data. In general, samples and sample sets will be created automatically from other RTMS objects (such as readers or spectra) but in the event that one wishes to manipulate them directly, it is important to understand several details about how they work.

In terms of the data it contains, an object of class rtmsSample is just a list of smaller objects (of class rtmsSubsample); however, each of these subsamples corresponds to an rtmsPeak object that was used to extract it; the rtmsSample object therefor has a "peaks" attribute, which is a list of objects of class rtmsPeak corresponding to the subsamples in the rtmsSample object. This attribute is used to determine how measurements of the sample are reported and how the sample is plotted.

Similarly, the data contained in an object of class rtmsSampleSet is just a list of rtmsSample objects but with an important difference. If many rtmsSample objects were arranged into a list, there would be no guarantee that they contain measurements of the same peaks; such guarantees

are essential for plotting sample sets together or constructing extracted ion chromatograms. The `rtmsSampleSet` therefore strips the "peaks" attribute from its individual members, and applies a single shared "peaks" attribute to the entire sample set. Further samples can only be added to the sample set if their peaks attributes are deemed compatible.

Value

An object of class `rtmsSampleSet`

An object of class `rtmsSample`

An object of class `rtmsSampleSet`

An object of class `rtmsSampleSet`

An object of class `rtmsSampleSet`

Functions

- `[[`: Select a single element of a sample set (returns an `rtmsSample`)
- `` `[[`(rtmsSampleSet) <- value ``: Insert a sample into a sample set
- `rep(rtmsSampleSet)`: Repeat a sample set multiple times
- `rep(rtmsSample)`: Create a sample set by repeating a single sample (returns an `rtmsSampleSet`)

Index