Package ‘MALDIquant’

May 12, 2019

Version 1.19.3
Date 2019-05-12
Title Quantitative Analysis of Mass Spectrometry Data
Depends R (>= 3.2.0), methods
Imports parallel
Suggests knitr, testthat (>= 0.8)
Description A complete analysis pipeline for matrix-assisted laser desorption/ionization-time-of-flight (MALDI-TOF) and other two-dimensional mass spectrometry data. In addition to commonly used plotting and processing methods it includes distinctive features, namely baseline subtraction methods such as morphological filters (TopHat) or the statistics-sensitive non-linear iterative peak-clipping algorithm (SNIP), peak alignment using warping functions, handling of replicated measurements as well as allowing spectra with different resolutions.
License GPL (>= 3)
URL http://strimmerlab.org/software/maldiquant/
https://github.com/sgibb/MALDIquant/
BugReports https://github.com/sgibb/MALDIquant/issues/
LazyLoad yes
VignetteBuilder knitr
RoxygenNote 6.1.1
NeedsCompilation yes
Author Sebastian Gibb [aut, cre] (<https://orcid.org/0000-0001-7406-4443>), Korbinian Strimmer [ths] (<https://orcid.org/0000-0001-7917-2056>)
Maintainer Sebastian Gibb <mail@sebastiangibb.de>
Repository CRAN
Date/Publication 2019-05-12 19:50:03 UTC
R topics documented:

- MALDIquant-package ................................................. 2
- AbstractMassObject-class ...................................... 4
- alignSpectra ....................................................... 6
- averageMassSpectra .............................................. 7
- binPeaks .......................................................... 9
- calibrateIntensity-methods .................................. 10
- createMassPeaks .................................................. 12
- createMassSpectrum ............................................. 13
- detectPeaks-methods ........................................... 14
- determineWarpingFunctions .................................. 15
- estimateBaseline-methods ...................................... 18
- estimateNoise-methods ......................................... 21
- fiedler2009subset ............................................... 23
- filterPeaks ....................................................... 24
- findEmptyMassObjects ......................................... 26
- intensityMatrix ................................................... 28
- isMassSpectrum .................................................. 29
- isMassSpectrumList ............................................. 30
- labelPeaks-methods ............................................. 31
- MALDIquant-parallel ........................................... 33
- MassPeaks-class ................................................ 35
- MassSpectrum-class ............................................ 36
- match.closest ..................................................... 37
- mergeMassPeaks ................................................ 39
- monoisotopicPeaks-methods .................................. 40
- msiSlices ........................................................ 41
- plot-methods ..................................................... 43
- plotMsiSlice-methods .......................................... 44
- referencePeaks .................................................. 46
- removeBaseline-methods ...................................... 48
- smoothIntensity-methods ..................................... 49
- transformIntensity-methods .................................. 51
- trim-methods .................................................... 52
- warpMassSpectra ................................................ 53

Index ........................................................................ 55
Description

MALDIquant provides a complete analysis pipeline for matrix-assisted laser desorption/ionization-time-of-flight (MALDI-TOF) and other two-dimensional mass spectrometry data.

In addition to commonly used plotting and processing methods it includes distinctive features, namely baseline subtraction methods such as morphological filters (TopHat) or the statistics-sensitive non-linear iterative peak-clipping algorithm (SNIP), peak alignment using warping functions, handling of replicated measurements as well as allowing spectra with different resolutions.

For a first overview see vignette("MALDIquant-intro", package="MALDIquant") and/or run demo("MALDIquant").

Details

Package: MALDIquant
License: GPL (>= 3)
URL: http://strimmerlab.org/software/maldiquant/

Main classes:
- MassPeaks: Represents a peak list of a single spectrum.

The accompanying website (see below) provides example R scripts to illustrate the functionality of this package, too.

Author(s)

Sebastian Gibb
Maintainer: Sebastian Gibb <mail@sebastiangibb.de>

References

Website: http://strimmerlab.org/software/maldiquant/

See Also

- Introduction: vignette("MALDIquant-intro", package="MALDIquant").
- Run demo files: demo("MALDIquant").
- List all available manual pages: library(help="MALDIquant").
- more MALDIquant examples and complete analyses: https://github.com/sgibb/MALDIquantExamples/.
AbstractMassObject-class

Class "AbstractMassObject"

Description

AbstractMassObject is an abstract (means pure virtual) class. It is the parent class of MassSpectrum and MassPeaks. It shouldn’t create or handle by the user because it is for internal use only.

Derived classes

MassPeaks, MassSpectrum

Slots

mass: numeric, mass or mass-to-charge ratio
intensity: numeric, intensities for measured mass-to-charge ratios
metaData: list, some metadata to describe the spectrum

Methods

signature(x = "AbstractMassObject", i = "numeric"): Extracts a range of an AbstractMassObject object and returns a new one.

as.matrix signature(x = "AbstractMassObject"): Converts an AbstractMassObject object to a matrix with 2 columns (mass, intensity).

coordinates signature(object = "AbstractMassObject"): Accessor function for coordinates stored in object generated from imaging mass spectrometry data.

coordinates<- signature(object = "AbstractMassObject", value = "numeric|matrix"): Replacement function for coordinates used in imaging mass spectrometry datasets.

intensity signature(object = "AbstractMassObject"): Accessor function for slot intensity.

intensity<- signature(object = "AbstractMassObject", value = "numeric"): Replacement function for slot intensity.

isEmpty signature(object = "AbstractMassObject"): Returns TRUE if length of intensity is 0 or all intensity values are 0.

length signature(x = "AbstractMassObject"): Returns length of slot intensity.

lines signature(x = "AbstractMassObject"): Extented function for adding AbstractMassObject object as a line to a specific plot. See lines for details.

mass signature(object = "AbstractMassObject"): Accessor function for slot mass.

mass<- signature(object = "AbstractMassObject", value = "numeric"): Replacement function for slot mass.

mz signature(object = "AbstractMassObject"): Accessor function for slot mass.

mz<- signature(object = "AbstractMassObject", value = "numeric"): Replacement function for slot mass.
**AbstractMassObject-class**

- **metaData** signature(object = "AbstractMassObject"): Accessor function for slot metaData.
- **metaData**<- signature(object = "AbstractMassObject"): Replacement function for slot metaData.
- **points** signature(x = "AbstractMassObject"): Extended function for adding AbstractMassObject object as points to a specific plot. See `points` for details.

**Author(s)**

Sebastian Gibb <mail@sebastiangibb.de>

**See Also**


Website: [http://strimmerlab.org/software/maldiquant/](http://strimmerlab.org/software/maldiquant/)

**Examples**

```r
## load package
library("MALDIquant")

## create example spectrum
s <- createMassSpectrum(mass=1:10, intensity=1:20,
                         metaData=list(name="Example Spectrum"))

## get intensity
intensity(s)

## get mass
mass(s)

## get metaData
metaData(s)

## replace metaData
metaData(s) <- list(name="Spectrum")

## trim spectrum
trim(s, c(2, 9))

## select a range
s[3:6]
```
alignSpectra  

Align MassSpectrum objects.

Description

This function aligns a list of MassSpectrum objects (spectra alignment is also known as warping/phase correction).

Usage

alignSpectra(spectra, halfWindowSize=20, noiseMethod="MAD", SNR=2, 
  reference, tolerance=0.002, warpingMethod="lowess", 
  allowNoMatches=FALSE, emptyNoMatches=FALSE, ...)

Arguments

spectra list, list of MassSpectrum objects.
halfWindowSize numeric, half window size; see detectPeaks.
noiseMethod a noise estimation method; see detectPeaks.
SNR single numeric value. SNR is an abbreviation for signal-to-noise-ratio; see detectPeaks.
reference MassPeaks, reference object to which the samples (1) should be aligned. If missing referencePeaks is used; see determineWarpingFunctions.
tolerance double, maximal relative deviation of a peak position (mass) to be considered as identical. Must be multiplied by 10^-6 for ppm, e.g. use tolerance=5e-6 for 5 ppm; see determineWarpingFunctions.
warpingMethod used basic warping function; see determineWarpingFunctions.
allowNoMatches logical, don’t throw an error if an MassPeaks object could not match to the reference; see determineWarpingFunctions.
emptyNoMatches logical, if TRUE (default: FALSE) the intensity values of MassSpectrum or MassPeaks objects with missing (NA) warping functions are set to zero; see warpMassSpectra.
... arguments to be passed to detectPeaks,MassSpectrum-method.

Details

alignSpectra is a wrapper function around detectPeaks, determineWarpingFunctions and warpMassSpectra. Please call these functions manually if you need finer control (e.g. plotting of warping functions).

Value

Returns a list of aligned MassSpectrum objects.
averageMassSpectra

Author(s)
Sebastian Gibb <mail@sebastiangibb.de>

See Also
demo("warping")
Website: http://strimmerlab.org/software/maldiQuant/

Examples

```r
## load package
library("MALDIquant")

## load example data
data("fiedler2009subset", package="MALDIquant")

## running typical workflow

## transform intensities
spectra <- transformIntensity(fiedler2009subset, method="sqrt")

## smooth spectra
spectra <- smoothIntensity(spectra, method="MovingAverage")

## baseline correction
spectra <- removeBaseline(spectra)

## align spectra
spectra <- alignSpectra(spectra)
```

---


Description

This function averages MassSpectrum objects.

Usage

```r
averageMassSpectra(l, labels, method=c("mean", "median", "sum"), ...)
```

Arguments

- `l` list, list of MassSpectrum objects.
- `labels` list, list of factors (one for each MassSpectrum object) to do groupwise averaging.
averageMassSpectra

method used aggregation function.

... arguments to be passed to underlying functions (currently only mc.cores is supported).

Details

The mass of the averaged MassSpectrum object will be the mass of the first non-empty MassSpectrum object (of each group).

Value

Returns a single (no labels given) or a list (labels given) of averaged MassSpectrum objects.

Author(s)

Sebastian Gibb <mail@sebastiangibb.de>

See Also

MassSpectrum, mergeMassPeaks

Website: http://strimmerlab.org/software/maldiquant/

Examples

```R
## load package
library("MALDIquant")

## create four MassSpectrum objects and add them to a list
s <- list(createMassSpectrum(mass=1:5, intensity=1:5),
         createMassSpectrum(mass=1:5, intensity=1:5),
         createMassSpectrum(mass=1:5, intensity=6:10),
         createMassSpectrum(mass=1:5, intensity=6:10))

## average all four MassSpectrum objects into a single new one
## by sum their intensities
## (no labels, returns only one new MassSpectrum object)
summedSpectra <- averageMassSpectra(s, method="sum")

## only average MassSpectrum objects in a group
## (e.g. useful for technical replicates)
## (two different labels, returns a list of two new MassPeaks objects)
groups <- factor(c("a", "a", "b", "b"), levels=c("a", "b"))
averagedSpectra <- averageMassSpectra(s, labels=groups, method="mean")
```
**binPeaks**

**Align Peaks into discrete bins.**

**Description**

This function looks for similar peaks (mass) across MassPeaks objects and equalizes their mass.

**Usage**

```r
binPeaks(l, method=c("strict", "relaxed"), tolerance=0.002)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>l</td>
<td>list, list of MassPeaks objects.</td>
</tr>
<tr>
<td>method</td>
<td>bin creation rule. &quot;strict&quot; creates bins never containing two or more peaks of the same sample. &quot;relaxed&quot; allows multiple peaks of the same sample in one bin.</td>
</tr>
<tr>
<td>tolerance</td>
<td>double, maximal relative deviation of a peak position (mass) to be considered as identical. Must be multiplied by 10^-6 for ppm, e.g. use tolerance=5e-6 for 5 ppm.</td>
</tr>
</tbody>
</table>

**Details**

The algorithm is based on the following workflow:

1. Put all mass in a sorted vector.
2. Calculate differences between each neighbor.
3. Divide the mass vector at the largest gap (largest difference) and form a left and a right bin.
4. Rerun step 3 for the left and/or the right bin if they don’t fulfill the following criteria:
   - All peaks in a bin are near to the mean \( \text{abs(mass - meanMass)/meanMass < tolerance} \).
   - method = "strict": The bin doesn’t contain two or more peaks of the same sample.

**Value**

Returns a list of mass adjusted MassPeaks objects.

**Author(s)**

Sebastian Gibb <mail@sebastiangibb.de>
calibrateIntensity-methods

Calibrates intensities of a MassSpectrum object.

Description
This function calibrates (normalize) intensities of MassSpectrum objects.

See Also
intensityMatrix.MassPeaks
Website: http://strimmerlab.org/software/maldiQuant/

Examples
```r
## load package
library("MALDIquant")

## create two MassPeaks objects
p <- list(createMassPeaks(mass=seq(100, 500, 100), intensity=1:5),
          createMassPeaks(mass=seq(100.2, 300.2, 100), 395), intensity=1:4))

binnedPeaks <- binPeaks(p, tolerance=0.002)

## compare result
iM1 <- intensityMatrix(p)
iM2 <- intensityMatrix(binnedPeaks)

all(dim(iM1) == c(2, 9)) # TRUE
all(dim(iM2) == c(2, 6)) # TRUE

show(iM2)

## increase tolerance
binnedPeaks <- binPeaks(p, tolerance=0.1)

iM3 <- intensityMatrix(binnedPeaks)

all(dim(iM3) == c(2, 5)) # TRUE

show(iM3)

## differences between "strict" and "relaxed"
p <- c(createMassPeaks(mass=c(1, 1.01, 3), intensity=c(2, 1, 1)),
       createMassPeaks(mass=c(0.99, 3), intensity=rep(1, 2)),
       createMassPeaks(mass=c(1.02, 3), intensity=rep(1, 2)))

intensityMatrix(binPeaks(p, method="strict", tolerance=0.05))
intensityMatrix(binPeaks(p, method="relaxed", tolerance=0.05))
```
Usage

```r
## S4 method for signature 'MassSpectrum'
calibrateIntensity(object, method=c("TIC", "PQN", "median"), range, ...)
## S4 method for signature 'list'
calibrateIntensity(object, method=c("TIC", "PQN", "median"), range, ...)
```

Arguments

- **object**: `MassSpectrum` object or a list of `MassSpectrum` objects.
- **method**: the calibration method to be used. This should be one of "TIC", "PQN" or "median". See 'Details' section.
- **range**: numeric of length 2, if given the scaling factor is calculated on the mass range from range[1L] to range[2L] and applied to the whole spectrum.
- **...**: arguments to be passed to other functions. Currently only `mc.cores` is supported if object is a list.

Details

A number of different calibration methods are provided:

"TIC": The TIC (Total Ion Current) of a `MassSpectrum` object is set to one. If range is given the TIC is only calculated for the intensities in the specified mass range.

"PQN": The PQN (Probabilistic Quotient Normalization) is described in Dieterle et al 2006. `calibrateIntensity` uses the following algorithm:
1. Calibrate all spectra using the "TIC" calibration.
2. Calculate a median reference spectrum.
3. Calculate the quotients of all intensities of the spectra with those of the reference spectrum.
4. Calculate the median of these quotients for each spectrum.
5. Divide all intensities of each spectrum by its median of quotients.

"median": The median of intensities of a `MassSpectrum` object is set to one.

Value

Returns a modified `MassSpectrum` object with calibrated intensities.

Author(s)

Sebastian Gibb <mail@sebiangibb.de>

References

createMassPeaks

See Also

MassSpectrum
Website: http://strimmerlab.org/software/maldiQuant/

Examples

```r
## load package
library("MALDIquant")

## load example data
data("fiedler2009subset", package="MALDIquant")

## baseline correction
b <- removeBaseline(fiedler2009subset)

## calibrate intensity values
calibrateIntensity(b, method="TIC")

## calibrate intensity values using TIC for a specific mass range
calibrateIntensity(b, method="TIC", range=c(3000, 5000))
```

createMassPeaks Creates a MassPeaks object.

Description

This function creates a MassPeaks object. Normally it shouldn’t be called by the user. Try detectPeaks, MassSpectrum-methods instead.

Usage

```r
createMassPeaks(mass, intensity, snr=rep.int(NA_real_, length(intensity)), metaData=list())
```

Arguments

- `mass`: vector, mass or mass-to-charge ratio.
- `intensity`: vector, intensities for measured mass-to-charge ratios.
- `snr`: vector, signal-to-noise ratios for intensity values.
- `metaData`: list, some metadata to describe the peaks.

Value

Returns a MassPeaks object.

Author(s)

Sebastian Gibb <mail@sebastiangibb.de>


createMassSpectrum

See Also

detectPeaks.MassSpectrum-method, MassPeaks

Website: http://strimmerlab.org/software/maldiquant/

Examples

```r
## load package
library("MALDIquant")

## create a MassPeaks object by default constructor
s <- createMassPeaks(mass=1:100, intensity=rnorm(100)^2,
                      metaData=list(name="example peaks"))

## show some details
s
```

createMassSpectrum  

*Creates a MassSpectrum object.*

Description

This function creates a *MassSpectrum* object.

Usage

```r
createMassSpectrum(mass, intensity, metaData=list())
```

Arguments

- **mass**: vector, mass or mass-to-charge ratio
- **intensity**: vector, intensities for measured mass-to-charge ratios
- **metaData**: list, some metadata to describe the spectrum

Value

Returns a *MassSpectrum* object.

Author(s)

Sebastian Gibb <mail@sebastiangibb.de>

See Also

*MassSpectrum*

Website: http://strimmerlab.org/software/maldiquant/*
detectPeaks-methods

**Examples**

```r
## load package
library("MALDIquant")

## create a MassSpectrum object by default constructor
s <- createMassSpectrum(mass=1:100, intensity=rnorm(100)^2,
                         metaData=list(name="example spectrum"))

## show some details
s
```

---

detectPeaks-methods  

**Detects peaks in a MassSpectrum object.**

**Description**

This method looks for peaks in mass spectrometry data (represented by a MassSpectrum object). A peak is a local maximum above a user defined noise threshold.

**Usage**

```r
## S4 method for signature 'MassSpectrum'
detectPeaks(object,
             halfWindowSize=20, method=c("MAD", "SuperSmoother"), SNR=2,
             ...)  
## S4 method for signature 'list'
detectPeaks(object, ...)
```

**Arguments**

- `object`  
  MassSpectrum object or a list of MassSpectrum objects.

- `halfWindowSize`  
  numeric, half window size. The resulting window reaches from `mass[currentIndex-halfWindowSize]` to `mass[currentIndex+halfWindowSize]`. A local maximum have to be the highest one in the given window to be recognized as peak.

- `method`  
  a noise estimation function; see `estimateNoise,MassSpectrum-method`.

- `SNR`  
  single numeric value. SNR is an abbreviation for signal-to-noise-ratio. A local maximum has to be higher than `SNR*noise` to be recognize as peak.

- `...`  
  arguments to be passed to `estimateNoise,MassSpectrum-method`. If object is a list `mc.cores` is also supported.

**Value**

Returns a MassPeaks object.
determineWarpingFunctions

Determine warping functions of MassPeaks objects.

Author(s)

Sebastian Gibb <mail@sebastiangibb.de>

See Also

MassPeaks, MassSpectrum, estimateNoise, MassSpectrum-method
demo("peaks")
Website: http://strimmerlab.org/software/maldiquant/

Examples

```r
## load package
library("MALDIquant")

## load example data
data("fiedler2009subset", package="MALDIquant")

## choose only the first mass spectrum
s <- fiedler2009subset[[1]]

## transform intensities
s <- transformIntensity(s, method="sqrt")

## smoothing spectrum
s <- smoothIntensity(s, method="MovingAverage")

## remove baseline
s <- removeBaseline(s)

## plot spectrum
plot(s)

## call peak detection
p <- detectPeaks(s)

## draw peaks on the plot
points(p)

## label 10 highest peaks
top10 <- intensity(p) %in% sort(intensity(p), decreasing=TRUE)[1:10]
labelPeaks(p, index=top10)
```
determineWarpingFunctions

Description

This function determines a warping function for a list of AbstractMassObject objects (warping is also known as phase correction/spectra alignment).

Usage

determineWarpingFunctions(l, reference, tolerance=0.002,
  method=c("lowess", "linear", "quadratic", "cubic"),
  allowNoMatches=FALSE,
  plot=FALSE, plotInteractive=FALSE, ...)

Arguments

l
  list, list of MassPeaks objects.
reference
  MassPeaks, reference object to which the samples (l) should be aligned. If missing referencePeaks is used.
tolerance
  double, maximal relative deviation of a peak position (mass) to be considered as identical. Must be multiplied by 10^{-6} for ppm, e.g. use tolerance=5e-6 for 5 ppm.
method
  used basic warping function.
allowNoMatches
  logical, don't throw an error if an MassPeaks object could not match to the reference.
plot
  logical, if TRUE a warping plot is drawn for each sample.
plotInteractive
  logical, if FALSE a non-interactive device (e.g. pdf) is used for warping plots.
  ...
  arguments to be passed to warpingFunction

Details

warpingFunction: determineWarpingFunctions estimates a warping function to overcome the difference between mass in reference and in the current sample. To calculate the differences each reference peak would match with the highest sample peak in the nearer neighborhood (defined by mass of reference peak*tolerance).
allowNoMatches: If allowNoMatches is TRUE a warning instead of an error is thrown if an MassPeaks object could not match to the reference. The returned list of warping functions will contain NA for this object (same index in the list).
plotInteractive: If plot is TRUE a lot of output is created (each sample in l gets its own plot). That's why an non-interactive devices is recommended:

## create a device
pdf()
## calculate warping functions
w <- determineWarpingFunctions(p, plot=TRUE)
## close device
dev.off()
**determineWarpingFunctions**

**Value**

Returns a list of individual warping functions. The attribute `nmatch` contains the number of matches of each `MassPeaks` element in `l` against reference.

**Author(s)**

Sebastian Gibb <mail@sebastiangibb.de>

**See Also**

`referencePeaks`, `warpMassPeaks`, `warpMassSpectra`, `MassPeaks`  
`demo("warping")`  
Website: [http://strimmerlab.org/software/maldiquant/](http://strimmerlab.org/software/maldiquant/)

**Examples**

```r
## load package
library("MALDIquant")

## create a reference MassPeaks object
r <- createMassPeaks(mass=1:5, intensity=1:5)

## create test samples
p <- list(createMassPeaks(mass=((1:5)*1.01), intensity=1:5),
          createMassPeaks(mass=((1:5)*0.99), intensity=1:5))

## create an interactive device with 2 rows
par(mfrow=c(2, 1))
## calculate warping function
## (using a linear function as basic warping function)
## and show warping plot
w <- determineWarpingFunctions(p, tolerance=0.02, method="linear",
                               plot=TRUE, plotInteractive=TRUE)
par(mfrow=c(1, 1))

## access number of matches
attr(w, "nmatch")

## w contains the individual warping functions
warpedPeaks <- warpMassPeaks(p, w)

## compare results
all(mass(r) == mass(warpedPeaks[[1]])) # TRUE
all(mass(r) == mass(warpedPeaks[[2]])) # TRUE

## realistic example

## load example data
data("fiedler2009subset", package="MALDIquant")
```
## estimateBaseline-methods

Estimates the baseline of a MassSpectrum object.

### Description

This method estimates the baseline of mass spectrometry data (represented by a MassSpectrum object).
Usage

## S4 method for signature 'MassSpectrum'

```
estimateBaseline(object,  
  method=c("SNIP", "TopHat", "ConvexHull", "median"),  
  ...)
```

Arguments

- **object**: `MassSpectrum` object
- **method**: used baseline estimation method, one of "SNIP", "TopHat", "ConvexHull" or "median".
- **...**: arguments to be passed to method

Details

"SNIP": This baseline estimation is based on the Statistics-sensitive Non-linear Iterative Peak-clipping algorithm (SNIP) described in Ryan et al 1988.

The algorithm based on the following equation:

\[
y_i(k) = \min\{y_i, \frac{(y_{i-k} + y_{i+k})}{2}\}
\]

It has two additional arguments namely iterations and decreasing. iterations controls the window size \(k\); similar to halfWindowSize in "TopHat", "Median" of the algorithm. The resulting window reaches from \(\text{mass}[	ext{cur_index} - \text{iterations}]\) to \(\text{mass}[	ext{cur_index} + \text{iterations}]\).

decreasing: In Morhac 2009 a decreasing clipping window is suggested to get a smoother baseline. For decreasing = TRUE (decreasing = FALSE) \(k\)=iterations is decreased (increased) by one until zero (iterations) is reached. The default setting is decreasing = TRUE.

"TopHat": This algorithm applies a moving minimum (erosion filter) and subsequently a moving maximum (dilation filter) filter on the intensity values. The implementation is based on van Herk 1996. It has an additional halfWindowSize argument determining the half size of the moving window for the TopHat filter. The resulting window reaches from \(\text{mass}[	ext{cur_index} - \text{halfWindowSize}]\) to \(\text{mass}[	ext{cur_index} + \text{halfWindowSize}]\).

"ConvexHull": The baseline estimation is based on a convex hull constructed below the spectrum.

"median": This baseline estimation uses a moving median. It is based on runmed. The additional argument halfWindowSize corresponds to the \(k\) argument in runmed \((k = 2 * \text{halfWindowSize} + 1)\) and controls the half size of the moving window. The resulting window reaches from \(\text{mass}[	ext{cur_index} - \text{halfWindowSize}]\) to \(\text{mass}[	ext{cur_index} + \text{halfWindowSize}]\).

Value

Returns a two column matrix (first column: mass, second column: intensity) of the estimated baseline.

Author(s)

Sebastian Gibb <mail@sebastiangibb.de>
References

"SNIP":


"TopHat":


"ConvexHull":

See Also

MassSpectrum, removeBaseline, MassSpectrum-method
demo("baseline")
Website: http://strimmerlab.org/software/maldi/quant/

Examples

## load package
library("MALDIquant")

## load example data
data("fiedler2009subset", package="MALDIquant")

## choose only the first mass spectrum
s <- fiedler2009subset[[1]]

## SNIP
plot(s)

## estimate baseline
b <- estimateBaseline(s, method="SNIP", iterations=100)

## draw baseline on the plot
lines(b, col="red")

## TopHat
plot(s)
```
## estimate baseline (try different parameters)
b1 <- estimateBaseline(s, method="TopHat", halfWindowSize=75)
b2 <- estimateBaseline(s, method="TopHat", halfWindowSize=150)

## draw baselines on the plot
lines(b1, col=2)
lines(b2, col=3)

## draw legend
legend(x="topright", lwd=1, legend=paste0("halfWindowSize=", c(75, 150)),
       col=c(2, 3))

## ConvexHull
plot(s)

## estimate baseline
b <- estimateBaseline(s, method="ConvexHull")

## draw baseline on the plot
lines(b, col="red")

## Median
plot(s)

## estimate baseline
b <- estimateBaseline(s, method="median")

## draw baseline on the plot
lines(b, col="red")
```

**estimateNoise-methods**  
Estimates the noise of a MassSpectrum object.

**Description**
This method estimates the noise of mass spectrometry data (represented by a MassSpectrum object).

**Usage**
```
## S4 method for signature 'MassSpectrum'
estimateNoise(object,
      method=c("MAD", "SuperSmoother"),
      ...)
```
estimateNoise-methods

Arguments

- **object**: MassSpectrum object
- **method**: used noise estimation method, one of "MAD" or "SuperSmoother".
- **...**: arguments to be passed to method.

Details

"MAD": This function estimates the noise of mass spectrometry data by calculating the median absolute deviation, see also `mad`.

"SuperSmoother": This function estimates the noise of mass spectrometry data using Friedman’s Super Smoother. Please refer `supsmu` for details and additional arguments.

Value

Returns a two column matrix (first column: mass, second column: intensity) of the estimated noise.

Author(s)

Sebastian Gibb <mail@sebastiangibb.de>

See Also

MassSpectrum, detectPeaks, MassSpectrum-method, mad, supsmu

Website: [http://strimmerlab.org/software/maldiquant/](http://strimmerlab.org/software/maldiquant/)

Examples

```r
## load package
library("MALDImquant")

## load example data
data("fiedler2009subset", package="MALDImquant")

## choose only the first mass spectrum
s <- fiedler2009subset[[1]]

## transform intensities
s <- transformIntensity(s, method="sqrt")

## remove baseline
s <- removeBaseline(s)

## plot spectrum
plot(s)

## estimate noise
nm <- estimateNoise(s, method="MAD")
nss <- estimateNoise(s, method="SuperSmoother")

## draw noise on the plot
```

Example Mass Spectra (raw)

Description

This dataset contains 16 example mass spectra. It is used to demonstrate the usage of `MALDIquant-package`.

Usage

`fiedler2009subset`

Format

A list containing 16 `MassSpectrum-class` objects.

Details

The dataset is a subset of data used in `Fiedler et al 2009`.
It contains spectra of 8 different patients (each one has 2 technical replicates).

<table>
<thead>
<tr>
<th>list_index</th>
<th>laboratory</th>
<th>patient_id</th>
<th>sex</th>
<th>age</th>
<th>type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Leipzig</td>
<td>LC77</td>
<td>male</td>
<td>37</td>
<td>control</td>
</tr>
<tr>
<td>2</td>
<td>Leipzig</td>
<td>LC77</td>
<td>male</td>
<td>37</td>
<td>control</td>
</tr>
<tr>
<td>3</td>
<td>Leipzig</td>
<td>LC213</td>
<td>female</td>
<td>51</td>
<td>control</td>
</tr>
<tr>
<td>4</td>
<td>Leipzig</td>
<td>LC213</td>
<td>female</td>
<td>51</td>
<td>control</td>
</tr>
<tr>
<td>5</td>
<td>Leipzig</td>
<td>LT178</td>
<td>male</td>
<td>58</td>
<td>cancer</td>
</tr>
<tr>
<td>6</td>
<td>Leipzig</td>
<td>LT178</td>
<td>male</td>
<td>58</td>
<td>cancer</td>
</tr>
<tr>
<td>7</td>
<td>Leipzig</td>
<td>LT157</td>
<td>male</td>
<td>60</td>
<td>cancer</td>
</tr>
<tr>
<td>8</td>
<td>Leipzig</td>
<td>LT157</td>
<td>male</td>
<td>60</td>
<td>cancer</td>
</tr>
<tr>
<td>9</td>
<td>Heidelberg</td>
<td>HC49</td>
<td>male</td>
<td>43</td>
<td>control</td>
</tr>
<tr>
<td>10</td>
<td>Heidelberg</td>
<td>HC49</td>
<td>male</td>
<td>43</td>
<td>control</td>
</tr>
<tr>
<td>11</td>
<td>Heidelberg</td>
<td>HC54</td>
<td>female</td>
<td>71</td>
<td>control</td>
</tr>
<tr>
<td>12</td>
<td>Heidelberg</td>
<td>HC54</td>
<td>female</td>
<td>71</td>
<td>control</td>
</tr>
<tr>
<td>13</td>
<td>Heidelberg</td>
<td>HT151</td>
<td>male</td>
<td>53</td>
<td>cancer</td>
</tr>
<tr>
<td>14</td>
<td>Heidelberg</td>
<td>HT151</td>
<td>male</td>
<td>53</td>
<td>cancer</td>
</tr>
<tr>
<td>15</td>
<td>Heidelberg</td>
<td>HT429</td>
<td>female</td>
<td>58</td>
<td>cancer</td>
</tr>
<tr>
<td>16</td>
<td>Heidelberg</td>
<td>HT429</td>
<td>female</td>
<td>58</td>
<td>cancer</td>
</tr>
</tbody>
</table>
References

“Serum Peptidome Profiling Revealed Platelet Factor 4 as a Potential Discriminating Peptide Associated with Pancreatic Cancer”
ISSN 1557-3265; doi:10.1158/1078-0432.CCR-08-2701
http://clincancerres.aacrjournals.org/content/15/11/3812

See Also

MassSpectrum-class
Website: http://strimmerlab.org/software/maldiquant/

filterPeaks

Removes less frequent peaks.

Description

This function removes infrequently occuring peaks in a list of Mass Peaks objects.

Usage

filterPeaks(l, minFrequency, minNumber, labels, mergeWhitelists=FALSE)

Arguments

l list, list of Mass Peaks objects.
minFrequency double, remove all peaks which occur in less than minFrequency*length(l) Mass Peaks objects. It is a relative threshold.
minNumber double, remove all peaks which occur in less than minNumber Mass Peaks objects. It is an absolute threshold.
labels factor, (one for each Mass Peaks object) to do groupwise filtering. The levels of the factor label define the groups. If not specified a single group is assumed.
mergeWhitelists logical, if FALSE the filtering criteria are applied groupwise. If TRUE peaks that survive the filtering in one group (level of labels) these peaks are also kept in other groups even if their frequencies are below minFrequency.

Details

For mergeWhitelists=FALSE the filtering uses a separate peak whitelist for each group specified by labels, and is done independently in each group. For mergeWhitelists=TRUE the peak whitelists are combined, which means that peaks that occur frequently in at least one group are also kept in all other groups.

If both minFrequency and minNumber arguments are specified the more stringent threshold is used.
filterPeaks

Value

Returns a list of filtered MassPeaks objects.

Author(s)

Sebastian Gibb <mail@sebastiangibb.de>

See Also

intensityMatrix, MassPeaks

Website: http://strimmerlab.org/software/maldiquant/

Examples

```r
## load package
library("MALDIquant")

## create four MassPeaks objects and add them to the list
p <- list(createMassPeaks(mass=1:2, intensity=1:2),
          createMassPeaks(mass=1:3, intensity=1:3),
          createMassPeaks(mass=1:4, intensity=1:4),
          createMassPeaks(mass=1:5, intensity=1:5))

## only keep peaks which occur in all MassPeaks objects
filteredPeaks <- filterPeaks(p, minFrequency=1)

## compare result
intensities <- intensityMatrix(filteredPeaks)

## peaks at mass 3,4,5 are removed
all(dim(intensities) == c(4, 2)) # TRUE
all(intensities[,1] == 1) # TRUE
all(intensities[,2] == 2) # TRUE

## only keep peaks which occur in all MassPeaks objects in a group
## (e.g. useful for technical replicates)
groups <- factor(c("a", "a", "b", "b"), levels=c("a", "b"))
filteredPeaks <- filterPeaks(p, minFrequency=1, labels=groups)

## peaks at mass 3 were removed in group "a"
filteredPeaks[groups == "a"]

## peaks at mass 5 were removed in group "b"
filteredPeaks[groups == "b"]

## only keep peaks which occur at least twice in a group
## groups <- factor(c("a", "a", "b", "b", "b"), levels=c("a", "b"))
filteredPeaks <- filterPeaks(c(p, p[[3]]), minNumber=2, labels=groups)

## peaks at mass 3 were removed in group "a"
filteredPeaks[groups == "a"]
```
findEmptyMassObjects

Finds or removes empty AbstractMassObject objects in a list.

Description

These functions look for empty AbstractMassObject objects in a list.

Usage

findEmptyMassObjects(l)
findEmptyMassObjects

removeEmptyMassObjects(l)

Arguments

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>list, list of AbstractMassObject where empty objects should be found or removed.</td>
</tr>
</tbody>
</table>

Value

findEmptyMassObjects: Returns a vector of indices referring to empty AbstractMassObject objects.
removeEmptyMassObjects: Returns a list of AbstractMassObject objects but without empty ones.

Author(s)

Sebastian Gibb <mail@sebastiangibb.de>

See Also

isEmpty, AbstractMassObject-method, AbstractMassObject

Website: http://strimmerlab.org/software/maldiquant/

Examples

```r
## load package
library("MALDIquant")

## create list
peakList <- list()

## create two MassPeaks objects and add them to the list
peakList[[1]] <- createMassPeaks(mass=1:100, intensity=1:100,
                                   metaData=list(name="example 1"))
peakList[[2]] <- createMassPeaks(mass=1:100, intensity=1:100,
                                   metaData=list(name="example 2"))

## find empty objects (there should not be any one)
findEmptyMassObjects(peakList)

## add an empty MassPeaks object to the list
peakList[[3]] <- createMassPeaks(mass=double(), intensity=double(),
                                   metaData=list(name="empty MassPeaks object"))

## look for empty objects (isEmptyIdx == 3)
(isEmptyIdx <- findEmptyMassObjects(peakList))

## to remove all empty MassObjects from a list
length(peakList) # 3
peakList <- removeEmptyMassObjects(peakList)
```
intensityMatrix

Converts a list of MassPeaks objects into a matrix.

Description

This function converts a list of MassPeaks objects into a matrix.

Usage

intensityMatrix(peaks, spectra)

Arguments

peaks list, list of MassPeaks objects.
spectra list, list of MassSpectrum objects. If a peak is missing the corresponding intensity value of the spectrum is used. If spectra is missing NA is used instead.

Details

peaks have to be binned by binPeaks before calling intensityMatrix.

Value

Returns a matrix containing intensities of all MassPeaks objects of peaks and interpolated intensity values for missing peaks if spectra was given or NA otherwise.

The matrix has length(peaks) rows (one row for each sample) and length(unique(mass)) columns. There is an additional attribute "mass" that stores the mass values.

Author(s)

Sebastian Gibb <mail@sebastiangibb.de>

See Also

binPeaks, MassPeaks, MassSpectrum

Website: http://strimmerlab.org/software/maldiquant/

Examples

```r
# load package
library("MALDIquant")

# create example MassPeaks objects
p <- list(createMassPeaks(mass=1:4,
   intensity=11:14,
   metaData=list(name="test mass peaks 1")))
```
isMassSpectrum

createMassPeaks(mass=2:5,
    intensity=22:25,
    metaData=list(name="test mass peaks 2")))

## converts MassPeaks objects into a matrix
intensityMatrix(p)

## realistic example
## load example data
data("fiedler2009subset", package="MALDIquant")

## transform intensities
s <- transformIntensity(fiedler2009subset, method="sqrt")

## smoothing spectrum
s <- smoothIntensity(s, method="MovingAverage")

## remove baseline
s <- removeBaseline(s)

## call peak detection
p <- detectPeaks(s)

## bin peaks
p <- binPeaks(p)

## convert MassPeaks objects into a matrix with missing intensity
## values
intensityMatrix(p)

## convert MassPeaks and MassSpectrum objects into a matrix without
## missing intensity values
intensityMatrix(p, s)

---

**isMassSpectrum**

Tests for MassSpectrum or MassPeaks object.

**Description**

These functions test for a MassSpectrum or MassPeaks object.

**Usage**

isMassSpectrum(x)

isMassPeaks(x)
Arguments

x  object to be tested.

Value

Returns TRUE or FALSE depending on whether its argument is an MassSpectrum or MassPeaks object.

Author(s)

Sebastian Gibb <mail@sebastiangibb.de>

See Also

MassPeaks, MassSpectrum, AbstractMassObject

Website: http://strimmerlab.org/software/maldiquant/

Examples

```r
## load package
library("MALDIquant")

## create a MassPeaks object
peaks <- createMassPeaks(mass=1:100, intensity=1:100,
                           metaData=list(name="example 1"))

## test
isMassPeaks(peaks)  # returns TRUE
isMassSpectrum(peaks)  # returns FALSE
isMassPeaks(double())  # returns FALSE
```

isMassSpectrumList Tests a list of MassSpectrum or MassPeaks objects.

Description

These functions test a list whether containing MassSpectrum or MassSpectrum objects.

Usage

isMassSpectrumList(x)

Arguments

x  object to be tested.
**Value**

Returns **TRUE** or **FALSE** depending on whether its argument is a list of `MassSpectrum` or `MassPeaks` objects.

**Author(s)**

Sebastian Gibb <mail@sebastiangibb.de>

**See Also**

`MassPeaks`, `MassSpectrum`, `AbstractMassObject`

Website: [http://strimmerlab.org/software/maldiquant/](http://strimmerlab.org/software/maldiquant/)

**Examples**

```r
## load package
library("MALDIquant")

## create list
p <- list()

## test list
isMassPeaksList(p) # returns FALSE

## create two MassPeaks objects and add them to the list
p <- createMassPeaks(mass=1:100, intensity=1:100,
                    metaData=list(name="example 1"))
p <- createMassPeaks(mass=1:100, intensity=1:100,
                    metaData=list(name="example 2"))

## test list
isMassPeaksList(p) # returns TRUE
isMassSpectrumList(p) # returns FALSE
```

---

**Description**

`labelPeaks` draws the corresponding mass values on top of the peaks stored in a `MassPeaks` object to a plot.

**Usage**

```r
## S4 method for signature 'MassPeaks'
labelPeaks(object,
           index,
           mass,
```
labels, digits=3, underline=TRUE,
verticalOffset=abs(diff(par("usr")[3:4]))*0.01,
absoluteVerticalPos,
adj=c(0.5, 0), cex=0.7, srt=0,
avoidOverlap=FALSE,
arrowLength=0, arrowLwd=0.5, arrowCol=1,
...

Arguments

object | MassPeaks object.
index | integer/logical, indices of peaks to label.
mass | numeric, mass of peaks to label.
labels | character, use labels instead of mass values as peak label.
digits | integer, number of decimal places.
underline | logical, underline peak values?
verticalOffset | numeric, move label vertically (relative to peak height).
absoluteVerticalPos
  numeric, absolute y value for the label. If missing verticalOffset is used.
adj | numeric, adjust text to the left, center, right and top, center, bottom; see text.
cex | numeric, font size, see par.
srt | numeric, the label rotation in degrees.
avoidOverlap | logical, try to find label coordinates to avoid overlap.
arrowLength, arrowLwd, arrowCol
  arrow parameters, possible vectors. NA values in arrowCol cause the arrow to be omitted, see arrows.
... | arguments to be passed to text.

Details

Please note that avoidOverlap = TRUE is just supported for srt %% 90 == 0 (means srt has to be a multiple of 90 degree).

Author(s)

Sebastian Gibb

See Also

MassPeaks, plot, AbstractMassObject, missing-method

Website: http://strimmerlab.org/software/maldiquant/
## Examples

```r
## load package
library("MALDIquant")

## create a MassPeaks object from scratch
p <- createMassPeaks(mass=1:20, intensity=sample(x=100:10000, size=20),
                     metaData=List(name="example"))

## plot peaks
plot(p)

## label the first 5 peaks
labelPeaks(p, index=1:5)

## label all peaks in mass range 15 to 20
labelPeaks(p, mass=15:20, underline=FALSE)

## label highest peaks (top 5)
top5 <- intensity(p) %in% sort(intensity(p), decreasing=TRUE)[1:5]
labelPeaks(p, index=top5, col="red")

## real example
data("fiedler2009subset")

## a simplified preprocessing
r <- removeBaseline(fiedler2009subset[[1]])
p <- detectPeaks(r)
plot(p)

## label highest peaks (top 10) and avoid label overlap
top10 <- sort(intensity(p), decreasing=TRUE, index.return=TRUE)$ix[1:10]
labelPeaks(p, index=top10, avoidOverlap=TRUE, digits=1)

## use own labels and rotate by 90 degree
plot(p)
labelPeaks(p, index=top10, labels=paste("TOP", 1:10), underline=FALSE,
           srt=90, adj=c(0, 0.5), col=2)
```

## Description

**MALDIquant** offers multi-core support using `mclapply` and `mcmapply`. This approach is limited to unix-based platforms.

Please note that not all functions benefit from parallelisation. Often the overhead to create/copy objects outrun the time saving of parallel runs. This is true for functions that are very fast to compute (e.g. sqrt-transformation). That's why the default value for the `mc.cores` argument in all
functions is 1L. It depends on the size of the dataset which step (often only removeBaseline and detectPeaks) benefits from parallelisation. In general it is faster to encapsulate the complete workflow into a function and parallelise it using mclapply instead of using the mc.cores argument of each method. The reason is the reduced overhead for object management (only one split/combine is needed instead of doing these operations in each function again and again).

Details

The following functions/methods support the mc.cores argument:

- trim.list.numeric-method
- transformIntensity.list-method
- smoothIntensity.list-method
- removeBaseline.list-method
- calibrateIntensity.list-method
- detectPeaks.list-method
- alignSpectra
- averageMassSpectra
- mergeMassPeaks

See Also

mclapply, mcmapply

Examples

```r
## load package
library("MALDIquant")

## load example data
data("fiedler2009subset", package="MALDIquant")

## run single-core baseline correction
print(system.time(
  b1 <- removeBaseline(fiedler2009subset, method="SNIP")
))

if(.Platform$OS.type == "unix") {
  ## run multi-core baseline correction
  print(system.time(
    b2 <- removeBaseline(fiedler2009subset, method="SNIP", mc.cores=2)
  ))
  stopifnot(all.equal(b1, b2))
}

## parallelise complete workflow
workflow <- function(spectra, cores) {
  s <- transformIntensity(spectra, method="sqrt", mc.cores=cores)
  s <- smoothIntensity(s, method="SavitzkyGolay", halfWindowSize=10,
                        mc.cores=cores)
```
MassPeaks-class

Description

MassPeaks represents extracted peaks of a single spectrum of a MALDI-TOF mass spectrometry measurement.

Objects from the Class

createMassPeaks: Creates a MassPeaks object.

Extends

Class AbstractMassObject, directly.

Slots

- snr: vector, signal-to-noise ratio

Methods

labelPeaks signature(x = "MassPeaks"): Draws peak labels to plot. See labelPeaks, MassPeaks-method for details.

monoisotopicPeaks signature(x = "MassPeaks"): Finds monoisotopic peaks in peak lists. See monoisotopicPeaks, MassPeaks-method for details.

Author(s)

Sebastian Gibb <mail@sebastiangibb.de>
MassSpectrum-class

See Also


Website: http://strimmerlab.org/software/maldiquant/

---

MassSpectrum-class  

Class "MassSpectrum"

Description

MassSpectrum represents a single spectrum of a MALDI-TOF mass spectrometry measurement. It provides an easy framework for doing some preprocessing steps like peak detection, baseline correction and much more.

Objects from the Class

createmassspectrum: Creates a MassSpectrum object.

Extends

Class AbstractMassObject, directly.

Methods

calibrateIntensity signature(x = "MassSpectrum"): Calibrates the intensity of a MassSpectrum object. See calibrateIntensity,MassSpectrum-method for details.
detectPeaks signature(x = "MassSpectrum"): Look for local maxima and estimate noise to extract peaks out of a MassSpectrum object. See detectPeaks,MassSpectrum-method for details.
isRegular signature(object = "MassSpectrum"): Returns FALSE if the frequency of mass values with irregular intervals is greater than threshold (because object was measured in centroid mode or some intensity values were filtered).
removeBaseline signature(x = "MassSpectrum"): Estimates and removes the baseline of a MassSpectrum object. See removeBaseline,MassSpectrum-method for details.
totalIonCurrent signature(object = "MassSpectrum"): Accessor function for Total Ion Current (TIC, area under the curve).
match.closest

Author(s)
Sebastian Gibb <mail@sebastiangibb.de>

See Also

Website: http://strimmerlab.org/software/maldiquant/

Examples

```r
## load package
library("MALDIquant")

## create a MassSpectrum object by default constructor
s <- createMassSpectrum(mass=1:100, intensity=rnorm(100)^2,
                           metaData=list(name="example"))

## show some details
s

## plot spectrum
plot(s)

## get TIC
totalIonCurrent(s)

## modify intensity and metaData
intensity(s)[1:50] <- 0
metaData(s) <- list(name="modified example")

## plot again
plot(s)
```

match.closest

**Relaxed Value Matching**

**Description**

match.closest returns a vector of the positions of (first) matches its first arguments in its second. In contrast to the similar match it just accept numeric arguments but has an additional tolerance argument that allows relaxed matching.

**Usage**

```r
match.closest(x, table, tolerance = Inf, nomatch = NA_integer_)
```
Arguments

- **x**: numeric, the values to be matched.
- **table**: numeric, the values to be matched against. In contrast to `match`, `table` has to be sorted in increasing order.
- **tolerance**: numeric, accepted tolerance. Use `Inf` to match without restrictions. Could be of length one or the same length as `table`.
- **nomatch**: numeric, if the difference between the value in `x` and `table` is larger than `tolerance`, `nomatch` is returned. Has to be of length one.

Value

An integer vector of the same length as `x` giving the closest position in `table` of the first match or `nomatch` if there is no match.

See Also

- `match`

Examples

```r
library("MALDIquant")
match.closest(c(1.1, 1.4, 9.8), 1:10)
# [1] 1 1 10
match.closest(c(1.1, 1.4, 9.8), 1:10, tolerance=0.25)
# [1] 1 NA 10
match.closest(c(1.1, 1.4, 9.8), 1:10, tolerance=0.25, nomatch=0)
# [1] 1 0 10

## this function is most useful if you want to subset an intensityMatrix
## by a few (reference) peaks

## create an example intensityMatrix
im <- matrix(1:10, nrow=2, dimnames=list(NULL, 1:5))
attr(im, "mass") <- 1:5
im
#   1 2 3 4 5
# [1,] 1 3 5 7 9
# [2,] 2 4 6 8 10
# attr(,"mass")
# [1] 1 2 3 4 5

## reference peaks
ref <- c(2.2, 4.8)

im[, match.closest(ref, attr(im, "mass"), tolerance=0.25, nomatch=0)]
#   2 5
# [1,] 3 9
# [2,] 4 10
```
mergeMassPeaks

Merges MassPeaks objects.

Description

This function merges MassPeaks objects.

Usage

mergeMassPeaks(l, labels, method=c("mean", "median", "sum"), ignore.na=TRUE, 
                  ...)  

Arguments

l       list, list of MassPeaks objects.
labels  list, list of factors (one for each MassPeaks object) to do groupwise merging.
method  used merge method.
ignore.na  Should NA (positions where a peak is missing) ignored (ignore.na=TRUE) or treated as zero (ignore.na=FALSE)?
...      arguments to be passed to underlying functions (currently only mc.cores is supported).

Value

Returns a single (no labels given) or a list (labels given) of merged MassPeaks objects.

Author(s)

Sebastian Gibb <mail@sebastiangibb.de>

See Also

MassPeaks, averageMassSpectra

Website: http://strimmerlab.org/software/maldiquant/

Examples

## load package
library("MALDIquant")

## create four MassPeaks objects and add them to the list
p <- list(createMassPeaks(mass=1:2, intensity=1:2),
          createMassPeaks(mass=1:3, intensity=1:3),
          createMassPeaks(mass=1:4, intensity=1:4),
          createMassPeaks(mass=1:5, intensity=1:5))

## merge all four MassPeaks objects into a single new one
by sum their intensities
(no labels, returns only one new MassPeaks object)
mergedPeaks <- mergeMassPeaks(p, method="sum")

only merge MassPeaks objects in a group
(two different labels, returns a list of two new MassPeaks objects)
groups <- factor(c("a", "a", "b", "b"), levels=c("a", "b"))
mergedPeaks <- mergeMassPeaks(p, labels=groups, method="mean")

the same, but treat NA as zero
mergedPeaks <- mergeMassPeaks(p, labels=groups, method="mean", ignore.na=FALSE)

---

Finds monoisotopic peaks in a MassPeaks object.

Description

This method looks for monoisotopic peaks in peak list data (represented by a MassPeaks objects). It is based on the poisson model for isotopic patterns described in Breen et al 2000.

Usage

```r
## S4 method for signature 'MassPeaks'
monoisotopicPeaks(object, 
  minCor=0.95, tolerance=1e-4, distance=1.00235, size=3L:10L)
## S4 method for signature 'list'
monoisotopicPeaks(object, ...)
```

Arguments

- **object**: MassPeaks object or a list of MassPeaks objects.
- **minCor**: double, minimal correlation between the peak pattern generated by the model and the experimental peaks in the MassPeaks object to be recognized as isotopic pattern.
- **tolerance**: double, maximal relative deviation of peaks position (mass) to be considered as isotopic distance (abs(((mass[i]+distance)-mass[i+1])/mass[i]) < tolerance).
- **distance**: double, distance between two consecutive peaks in an isotopic pattern (default value taken from Park et al 2008). Could contain more than one value, e.g. distance=(1:3)^-1 to find isotopic patterns for multiple charged patterns (e.g. 1+, 2+, and 3+). Please note that the order matters here if there is a monoisotopic peak for charge state 1 and 3 it would be reported as charge 1 for distance=(1:3)^-1 and as 3 for distance=(3:1)^-1 respectively.
- **size**: double, size (length) of isotopic pattern, longer patterns are preferred over shorter ones.
- **...**: arguments to be passed to monoisotopicPeaks, MassPeaks-method. If object is a list mc.cores is also supported.
msiSlices

Value

Returns a MassPeaks object with monoisotopic peaks only.

Author(s)

Sebastian Gibb <mail@sebastiangibb.de>

References


See Also

MassPeaks, detectPeaks, MassSpectrum-method

Website: http://strimmerlab.org/software/maldiquant/

Examples

```r
## load package
library("MALDIquant")

## create example peaks
p <- createMassPeaks(mass=995:1005,
                      intensity=c(100, 10, 30, 10, 40, # noise
                                  550, 330, 110, 10,   # isotopic pattern
                                  5, 15))            # more noise
m <- monoisotopicPeaks(p)
as.matrix(m)

## plot the peaks and mark the monoisotopic one
plot(p)
points(m, col=2, pch=4)
```

```r
msiSlices

Turn a list of AbstractMassObjects into a mass spectrometry imaging slice.

Description

This function turns a mass spectrometry imaging dataset represented by a list of AbstractMassObject objects into an intensityMatrix for each slice (stored in an array).```
Usage

msiSlices(x, center, tolerance, method=c("sum", "mean", "median"), adjust=TRUE)

Arguments

  x      a list of MassSpectrum/MassPeaks objects.
  center double, the center mas value of each slice.
  tolerance double, specifies the thickness of the slices (center + c(-tolerance, tolerance)).
  method used aggregation function.
  adjust logical, if TRUE the lowest coordinates of the mass spectrometry imaging dataset
           are set to c(x=1, y=1) to avoid NA values at the borders.

Details

Each MassSpectrum/MassPeaks object in x must contain a list named imaging with an element
pos that stores the x and y value of the spectrum, e.g.:

> metaData(spectra[[1]])$imaging$pos
  x   y
  1   5

Value

Returns an array of three dimensions. The first and second dimensions contains the x and y coor-
dinates of the image. The third dimension represents the index of the center of each slice. There
are two additional attributes, namely "center" and "tolerance" which store the original center
and tolerance information.

Author(s)

Sebastian Gibb <mail@sebastiangibb.de>

See Also

AbstractMassObject, MassSpectrum, MassPeaks, coordinates, AbstractMassObject-method,
plotMsiSlice, list-method

Please find real examples on:

Website: http://strimmerlab.org/software/maldiquant/


Shiny: https://github.com/sgibb/ims-shiny/
Examples

```r
## load package
library("MALDIquant")

## load example data
data("fiedler2009subset", package="MALDIquant")

## please note: this is NOT a MSI data set
## we just add some coordinates for demonstration
coordinates(fiedler2009subset) <- cbind(x=rep(1:4, 2), y=rep(1:2, each=4))

slices <- msiSlices(fiedler2009subset, center=c(5864.49, 8936.97),
                     tolerance=0.25)
slices
```

---

**plot-methods**

Plots an AbstractMassObject object.

---

**Description**

This is an overloaded method to allow plotting of an AbstractMassObject object.

**Usage**

```r
## S4 method for signature 'AbstractMassObject,missing'
plot(x, col="black",
     xlab=expression(italic(m/z)), ylab="intensity",
     type=ifelse(isMassPeaks(x), "h", "l"),
     xlim=c(ifelse(length(x@mass), min(x@mass, na.rm=TRUE), 0),
            ifelse(length(x@mass), max(x@mass, na.rm=TRUE), 1)),
     ylim=c(0, ifelse(length(x@intensity), max(x@intensity, na.rm=TRUE), 1)),
     main=x@metaData$FileName, sub=x@metaData$File, 
     cex=0.75, col.sub="#808080", ...)
```

**Arguments**

- `x` MassSpectrum object.
- `col` line colour, see `par`.
- `xlab` title for the x-axis, see `title`.
- `ylab` title for the y-axis, see `title`.
- `type` type of plot: see `plot`.
- `xlim` the x limits (x1, x2) of the plot, see `plot.default`.
- `ylim` the y limits (y1, y2) of the plot, see `plot.default`.
- `main` title for the plot, see `title`. 
sub

sub title for the plot, see `title`.

cex.sub

sub title font size, see `par`.

col.sub

sub title color, see `par`.

... arguments to be passed to `plot`.

Author(s)

Sebastian Gibb <mail@sebastiangibb.de>

See Also

`AbstractMassObject`

Website: [http://strimmerlab.org/software/maldiquant/](http://strimmerlab.org/software/maldiquant/)

Examples

```r
## load package
library("MALDIquant")

## create a MassSpectrum object by default constructor
s <- createMassSpectrum(mass=1:100, intensity=rnorm(100)^2, 
                          metaData=list(name="example"))

## show some details
s

## plot spectrum
plot(s)
```

plotMsiSlice-methods  

Plots a Mass Spectrometry Imaging dataset.

Description

This function allows to plot a slice of a mass spectrometry imaging dataset represented by a list of `AbstractMassObject` objects or an array or a matrix.

Usage

```r
## S4 method for signature 'list'
plotMsiSlice(x, center, tolerance, 
             colRamp=colorRamp(c("black", "blue", "green", "yellow", "red")),
             interpolate=FALSE, legend=TRUE, alignLabels=FALSE, combine=FALSE, 
             ...)

## S4 method for signature 'array'
plotMsiSlice(x, 
             colRamp=colorRamp(c("black", "blue", "green", "yellow", "red")),
```

Arguments

x  The mass spectrometry imaging dataset. It could be a list of MassSpectrum/MassPeaks objects or an array (e.g. generated by msiSlices) or a matrix.
center  double, if x is a list of MassSpectrum/MassPeaks objects this argument represent the center mass value of the slices, see msiSlices for details.
tolerance  double, if center is given tolerance specifies the thickness of the slices (center + c(tolerance, see msiSlices for details).
colRamp  colours as colorRamp function, see colorRamp for details. If combine=TRUE multiple colour functions must be applied as list with an length that equals the number of given centers.
interpolate  logical, use linear interpolation when drawing the image, see rasterimage for details.
scale  logical, if TRUE all values are divided by the maximal value of the slice to get values between 0 and 1.
legend  logical, if TRUE a reference color gradient is plotted on the right hand side of the plot. The upper color represents the highest value in the slice and the lower color the lowest value respectively. The legend is disabled if scale=FALSE.
alignLabels  logical, if combine=TRUE and alignLabels=TRUE the center positions below the legend are aligned on the right margin otherwise the aligned to their corresponding gradient.
combine  logical, if TRUE multiple centers are plotted in one image. Therefore it would be necessary to apply a list of colRamp functions (one function for each center). The intensity values for each center of each pixel are compared against each other and the highest scaled intensity determines the center (and the corresponding colRamp).
plotInteractive  logical, if the slice array contains multiple centers, combine=FALSE and an interactive plotting device is used a warning is thrown and only the first center would be plotted. Use plotInteractive=TRUE to overwrite this behaviour and to plot multiple centers on an interactive device.

... arguments to be passed to plot, e.g. main.

Details

Each MassSpectrum/MassPeaks object in x must contain a list named imaging with an element pos that stores the x and y value of the spectrum, e.g.:

```r
> metaData(spectra[[1]])$imaging$pos
x  y
1 5
```
**referencePeaks**

**Author(s)**

Sebastian Gibb <mail@sebastiangibb.de>

**See Also**


Please find real examples on:

Website: [http://strimmerlab.org/software/maldiQuant/](http://strimmerlab.org/software/maldiQuant/)


Shiny: [https://github.com/sgibb/ims-shiny/](https://github.com/sgibb/ims-shiny/)

**Examples**

```r
## load package
library("MALDIquant")

## load example data
data("fiedler2009subset", package="MALDIquant")

## please note: this is NOT a MSI data set
## we just add some coordinates for demonstration
coordinates(fiedler2009subset) <- cbind(x=rep(1:4, 2), y=rep(1:2, each=4))

plotMsiSlice(fiedler2009subset, center=8936.97, tolerance=0.25)

plotMsiSlice(fiedler2009subset, center=c(5864.49, 8936.97), tolerance=0.25, 
  combine=TRUE, 
  colRamp=list(colorRamp(c("#000000", "#FF00FF")), 
               colorRamp(c("#000000", "#00FF00"))))
```

**referencePeaks**

*Creates a reference MassPeaks object.*

**Description**

This function creates a reference MassPeaks object (also called Anchor Peaks) from a list of MassPeaks objects.

Generally it is a combination of binPeaks and filterPeaks

**Usage**

```r
referencePeaks(l, method=c("strict", "relaxed"), minFrequency=0.9, 
  tolerance=0.002)
```
Arguments

1  list, list of `MassPeaks` objects.
method  bin creation rule (see `binPeaks`).
minFrequency  double, remove all peaks which occur in less than minFrequency*length(l) `MassPeaks` objects.
tolerance  double, maximal relative deviation of a peak position (mass) to be considered as identical. Must be multiplied by 10^-6 for ppm, e.g. use tolerance=5e-6 for 5 ppm.

Value

Returns a new `MassPeaks` objects.
The `intensity` slot of the returned `MassPeaks` represents the frequency of this mass position in all samples.

Author(s)

Sebastian Gibb <mail@sebastiangibb.de>

See Also

`binPeaks`, `filterPeaks`, `MassPeaks`

Website: http://strimmerlab.org/software/maldiquant/

Examples

```r
## load package
library("MALDIquant")

## create four MassPeaks objects and add them to the list
p<- list(createMassPeaks(mass=1:2, intensity=1:2),
         createMassPeaks(mass=1:3, intensity=1:3),
         createMassPeaks(mass=1:4, intensity=1:4),
         createMassPeaks(mass=1:5, intensity=1:5))

## only use peaks which occur in all MassPeaks objects as reference peaks
refPeaks <- referencePeaks(p, minFrequency=1)

mass(refPeaks)  # 1:2
intensity(refPeaks)  # c(1, 1)
```
removeBaseline-methods

Removes the baseline of a MassSpectrum object.

Description

This method removes the baseline of mass spectrometry data (represented by a Mass Spectrum object). The intensity of the mass spectrometry data would be reduced by baseline.

Usage

```r
## S4 method for signature 'Mass Spectrum'
removeBaseline(object, method=c("SNIP", "TopHat", "ConvexHull", "median"), ...)
## S4 method for signature 'list'
removeBaseline(object, ...)
```

Arguments

- `object` Mass Spectrum object or a list of Mass Spectrum objects.
- `method` used baseline estimation method, one of "SNIP", "TopHat", "ConvexHull" or "median". See estimateBaseline,Mass Spectrum-method for details.
- `...` arguments to be passed to estimateBaseline,Mass Spectrum-method. If object is a list mc.cores is also supported.

Value

Returns a modified Mass Spectrum object with reduced intensities.

Author(s)

Sebastian Gibb <mail@sebiangibb.de>

See Also

Mass Spectrum, estimateBaseline,Mass Spectrum-method
demo("baseline")

Website: http://strimmerlab.org/software/maldiQuant/
Examples

```r
## load package
library("MALDIquant")

## load example data
data("fiedler2009subset", package="MALDIquant")

## choose only the first mass spectrum
s <- fiedler2009subset[[1]]

## plot spectrum
plot(s)

## subtract baseline
b <- removeBaseline(s, method="SNIP")

## draw modified spectrum on the plot
lines(b, col="blue")
```

smoothIntensity-methods

Smoothes intensities of a MassSpectrum object.

Description

This method smoothes the intensity values of a MassSpectrum object.

Usage

```r
## S4 method for signature 'MassSpectrum'
smoothIntensity(object, 
               method=c("SavitzkyGolay", "MovingAverage"), halfWindowSize, 
               ...)
```

Arguments

- **object**: AbstractMassObject object or a list of AbstractMassObject objects.
- **method**: used smoothing method, one of "SavitzkyGolay" or "MovingAverage".
- **halfWindowSize**: half window size. The resulting window reaches from mass[currentIndex-halfWindowSize] to mass[currentIndex+halfWindowSize] (window size is 2*halfWindowSize+1). The best size differs depending on the selected method.
- **...**: arguments to be passed to method. SavitzkyGolay has an additional polynomialOrder argument (default: 3) to control the order of the filter. MovingAverage has an additional weighted argument (default: FALSE) to indicate if the average should be equal weight (default) or if it should have weights depending on the distance from the center as calculated as 1/2^abs(-halfWindowSize:halfWindowSize) with the sum of all weights normalized to 1.
Details

halfWindowSize: Depends on the selected method. For the SavitzkyGolay the halfWindowSize should be smaller than FWHM of the peaks (full width at half maximum; please find details in Bromba and Ziegler 1981). In general the halfWindowSize for the MovingAverage has to be much smaller than for SavitzkyGolay to conserve the peak shape.

Author(s)

Sebastian Gibb <mail@sebastiangibb.de>

Weighted moving average: Sigurdur Smarason

References


See Also

MassSpectrum

Website: http://strimmerlab.org/software/maldiquant/

Examples

```r
## load package
library("MALDIquant")

## load example data
data("fiedler2009subset", package="MALDIquant")

## smooth spectra
s <- smoothIntensity(fiedler2009subset, method="MovingAverage", halfWindowSize=2)
## or
s <- smoothIntensity(fiedler2009subset, method="MovingAverage", halfWindowSize=2, weighted=TRUE)
## or
s <- smoothIntensity(fiedler2009subset, method="SavitzkyGolay", halfWindowSize=10)
```
Transforms intensities of an AbstractMassObject object.

Description

This method performs a transformation (e.g., sqrt-transformation) on the intensities of an AbstractMassObject object.

Usage

```r
## S4 method for signature 'AbstractMassObject'
transformIntensity(object, 
    method=c("sqrt", "log", "log2", "log10"))
## S4 method for signature 'list'
transformIntensity(object, ...)
```

Arguments

- `object`: AbstractMassObject object or a list of AbstractMassObject objects.
- `method`: used transformation method.
- `...`: arguments to be passed to underlying functions. If `object` is a list `mc.cores` is also supported.

Author(s)

Sebastian Gibb <mail@sebastiangibb.de>

See Also

AbstractMassObject, MassSpectrum

Website: http://strimmerlab.org/software/maldiquant/

Examples

```r
## load package
library("MALDIquant")

## load example data
data("fiedler2009subset", package="MALDIquant")

## choose only the first mass spectrum
s <- fiedler2009subset[[1]]

## transform spectrum
t <- transformIntensity(s, method="sqrt")

## plot spectrum
```
trim-methods

Trim an AbstractMassObject object.

Description

This method trims an AbstractMassObject object. That is useful if some mass ranges should be excluded from further analysis.

Usage

```r
## S4 method for signature 'AbstractMassObject, numeric'
trim(object, range)
## S4 method for signature 'list, numeric'
trim(object, range, ...)
## S4 method for signature 'list, missing'
trim(object, range, ...)
```

Arguments

- `object`: AbstractMassObject object or a list of AbstractMassObject objects.
- `range`: numeric, limits of trimming (left/minimal mass, right/maximal mass). If missing it is automatically determined (largest overlapping mass range) for a list of AbstractMassObject.
- `...`: arguments to be passed to underlying functions (currently only mc.cores is supported).

Author(s)

Sebastian Gibb <mail@sebastiangibb.de>

See Also

AbstractMassObject, MassPeaks, MassSpectrum

Website: [http://strimmerlab.org/software/maldiquant/](http://strimmerlab.org/software/maldiquant/)

Examples

```r
## load package
library("MALDIquant")

## load example data
data("fiedler2009subset", package="MALDIquant")
```
warpMassSpectra

```r
## select only one spectrum
s <- fiedler2009subset[[1]]

## remove all mass lower 3000
trim(s, range=c(3000, Inf))

## remove all mass higher 8000
trim(s, range=c(0, 8000))

## remove all mass lower 3000 and higher 8000
trim(s, range=c(3000, 8000))

## choose largest overlapping mass range for all spectra
trim(fiedler2009subset)
```

---

**warpMassSpectra**

*Run warping functions on AbstractMassObject objects.*

**Description**

These functions run warping functions on AbstractMassObject objects (warping is also known as phase correction).

**Usage**

```r
warpMassPeaks(l, w, emptyNoMatches=FALSE)
warpMassSpectra(l, w, emptyNoMatches=FALSE)
```

**Arguments**

- `l` - list, list of MassPeaks or MassSpectrum objects.
- `w` - a list of warping functions determined by determineWarpingFunctions. Has to be of the same length as `l`.
- `emptyNoMatches` - logical, if TRUE (default: FALSE) the intensity values of MassSpectrum or MassPeaks objects with missing (NA) warping functions are set to zero.

**Details**

The warping function `w` is called in the following way:

```r
newMass = oldMass + w(oldMass)
```

**Value**

Returns a list of warped MassPeaks or MassSpectrum objects.
Author(s)
Sebastian Gibb <mailto@sebastiangibb.de>

See Also
determineWarpingFunctions, MassPeaks, MassSpectrum

Website: http://strimmerlab.org/software/maldiquant/

Examples

```r
## load package
library("MALDIquant")

## create a MassPeaks object
p <- createMassPeaks(mass=1:5, intensity=1:5)

## stupid warping function for demonstration
## (please use determineWarpingFunctions in real life applications)
simpleWarp <- function(x) { return(1) }

## run warping function
w <- warpMassPeaks(list(p), list(simpleWarp))[[1]]

## compare results
all(mass(w) == mass(p)+1) # TRUE

## no warping (MassPeaks object is not changed)
warpMassPeaks(list(p), list(NA))

## no warping (intensity values of MassPeaks object are set to zero)
warpMassPeaks(list(p), list(NA), emptyNoMatches=TRUE)
```
Index

*Topic classes
  AbstractMassObject-class, 4
  MassPeaks-class, 35
  MassSpectrum-class, 36

*Topic datasets
  fiedler200subset, 23

*Topic methods
  alignSpectra, 6
  averageMassSpectra, 7
  binPeaks, 9
  calibrateIntensity-methods, 10
  createMassPeaks, 12
  createMassSpectrum, 13
  detectPeaks-methods, 14
  determineWarpingFunctions, 15
  estimateBaseline-methods, 18
  estimateNoise-methods, 21
  filterPeaks, 24
  findEmptyMassObjects, 26
  intensityMatrix, 28
  isMassSpectrum, 29
  isMassSpectrumList, 30
  labelPeaks-methods, 31
  mergeMassPeaks, 39
  monoisotopicPeaks-methods, 40
  plot-methods, 43
  referencePeaks, 46
  removeBaseline-methods, 48
  smoothIntensity-methods, 49
  transformIntensity-methods, 51
  trim-methods, 52
  warpMassSpectra, 53

*Topic misc
  MALDIquant-parallell, 33
  [,AbstractMassObject,logical,missing-method
   (AbstractMassObject-class), 4]
  [,AbstractMassObject,numerical,missing-method
   (AbstractMassObject-class), 4]
  [,MassPeaks,logical,missing-method
   (AbstractMassObject-class), 4]

AbstractMassObject, 4, 5, 16, 26, 27, 30, 31,
  35–37, 41–44, 46, 49, 51–53

AbstractMassObject-class, 4
  alignSpectra, 6, 34
  array, 42
  arrows, 32
  as.matrix,AbstractMassObject-method
  (AbstractMassObject-class), 4
  averageMassSpectra, 7, 34, 39

  binPeaks, 9, 28, 46, 47

  calibrateIntensity
    (calibrateIntensity-methods),
    10
  calibrateIntensity,list-method
    (calibrateIntensity-methods),
    10
  calibrateIntensity,MassSpectrum-method
    (calibrateIntensity-methods),
    10
  calibrateIntensity-methods, 10
  colorRamp, 45
  coordinates
    (AbstractMassObject-class),
    4
  coordinates,AbstractMassObject-method
    (AbstractMassObject-class), 4
  coordinates,list-method
    (AbstractMassObject-class), 4
  coordinates<-
    (AbstractMassObject-class), 4
  coordinates<-,AbstractMassObject,matrix-method
    (AbstractMassObject-class), 4
  coordinates<-,AbstractMassObject,numerical-method
    (AbstractMassObject-class), 4
coordinates<-, list, matrix-method
(AbstractMassObject-class), 4
createMassPeaks, 12, 35, 36
createMassSpectrum, 13, 36, 37
detectPeaks, 6, 7, 34
detectPeaks (detectPeaks-methods), 14
detectPeaks, list-method
(detectPeaks-methods), 14
detectPeaks, MassSpectrum-method
(detectPeaks-methods), 14
detectPeaks-methods, 14
determineWarpingFunctions, 6, 7, 15, 53, 54

estimateBaseline
(estimateBaseline-methods), 18
estimateBaseline, MassSpectrum-method
(estimateBaseline-methods), 18
estimateBaseline-methods, 18
estimateNoise (estimateNoise-methods), 21
estimateNoise, MassSpectrum-method
(estimateNoise-methods), 21
estimateNoise-methods, 21
false, 30, 31
findEmptyMassObjects, 26
isRegular, MassSpectrum-method
(MassSpectrum-class), 36
isRegular
(MassSpectrum-class), 36
isMassPeaks, 3–6, 9, 10, 12–17, 24, 25, 28–32, 35, 39–42, 45–47, 52–54
isMassPeaks (isMassSpectrum), 29
isMassPeaksList (isMassSpectrumList), 30
isMassSpectrum, 29
isMassSpectrumList, 30
isRegular
(MassSpectrum-class), 36
mad, 22
MALDIquant, 33
MALDIquant (MALDIquant-package), 2
MALDIquant-package, 2
MALDIquant-parallel, 33
mass
(AbstractMassObject-class), 4
mass, AbstractMassObject-method
(AbstractMassObject-class), 4
mass<-(AbstractMassObject-class), 4
mass<-, AbstractMassObject, numeric-method
(AbstractMassObject-class), 4
MassPeaks, 3–8, 10–15, 18–22, 28–31, 36, 42, 43, 45, 46, 48–54
MassSpectrum (MassSpectrum-class), 35
MassSpectrum-class, 35
Match, 37, 38
match, 37, 38
match.closest, 37
metadata, 47
metadata
(AbstractMassObject-class), 4
metadata<-(AbstractMassObject-class), 4
metadata<-, AbstractMassObject, numeric-method
(AbstractMassObject-class), 4
mclapply, 33, 34
mcmapply, 33, 34
mergeMassPeaks, 8, 34, 39
metaData
(AbstractMassObject-class), 4
metaData, AbstractMassObject-method
(AbstractMassObject-class), 4
metaData<-(AbstractMassObject-class), 4
metaData<-, AbstractMassObject-method
(AbstractMassObject-class), 4
monoisotopicPeaks
(monomoisotopicPeaks-methods), 40
**INDEX**

monoisotopicPeaks, list-method
  (monoisotopicPeaks-methods), 40
monoisotopicPeaks, MassSpectrum-method
  (monoisotopicPeaks-methods), 40
monoisotopicPeaks-methods, 40
msiSlices, 41, 45, 46
mz (AbstractMassObject-class), 4
mz, AbstractMassObject-method
  (AbstractMassObject-class), 4
mz<-(AbstractMassObject-class), 4
mz<-, AbstractMassObject-method
  (AbstractMassObject-class), 4
par, 32, 43, 44
plot, 43–45
plot, AbstractMassObject, missing-method
  (plot-methods), 43
plot, MassSpectrum, missing-method
  (plot-methods), 43
plot-methods, 43
plot.default, 43
plotMsISlice (plotMsISlice-methods), 44
plotMsISlice, array-method
  (plotMsISlice-methods), 44
plotMsISlice, list-method
  (plotMsISlice-methods), 44
plotMsISlice, matrix-method
  (plotMsISlice-methods), 44
plotMsISlice-methods, 44
points, 5
points, AbstractMassObject-method
  (AbstractMassObject-class), 4
rasterImage, 45
referencePeaks, 6, 7, 16, 17, 46
removeBaseline, 34
removeBaseline
  (removeBaseline-methods), 48
removeBaseline, list-method
  (removeBaseline-methods), 48
removeBaseline, MassSpectrum-method
  (removeBaseline-methods), 48
removeBaseline-methods, 48
removeEmptyMassObjects
  (findEmptyMassObjects), 26
runmed, 19
smoothIntensity, list-method
  (smoothIntensity-methods), 49
smoothIntensity, MassSpectrum-method
  (smoothIntensity-methods), 49
smoothIntensity-methods, 49
snr (MassPeaks-class), 35
snr, MassPeaks-method (MassPeaks-class), 35
supsmu, 22
text, 32
title, 43, 44
totalIonCurrent (MassSpectrum-class), 36
totalIonCurrent, MassSpectrum-method
  (MassSpectrum-class), 36
transformIntensity
  (transformIntensity-methods), 51
transformIntensity, AbstractMassObject-method
  (transformIntensity-methods), 51
transformIntensity, list-method
  (transformIntensity-methods), 51
transformIntensity-methods, 51
trim (trim-methods), 52
trim, AbstractMassObject, numeric-method
  (trim-methods), 52
trim, list, missing-method
  (trim-methods), 52
trim, list, numeric-method
  (trim-methods), 52
trim-methods, 52
TRUE, 30, 31
vector, 27
warpMassPeaks, 17
warpMassPeaks (warpMassSpectra), 53
warpMassSpectra, 6, 7, 17, 53