Package ‘tofsims’

December 22, 2016

Type Package

Title Import, process and analysis of Time-of-Flight Secondary Ion Mass Spectrometry (ToF-SIMS) imaging data

Version 1.2.0

Author Lorenz Gerber, Viet Mai Hoang

Maintainer Lorenz Gerber <genfys@gmail.com>

Depends R (>= 3.3.0), methods, utils, ProtGenerics

Description This packages offers a pipeline for import, processing and analysis of ToF-SIMS 2D image data. Import of Iontof and Ulvac-Phi raw or preprocessed data is supported. For rawdata, mass calibration, peak picking and peak integration exist. General functionality includes data binning, scaling, image subsetting and visualization. A range of multivariate tools common in the ToF-SIMS community are implemented (PCA, MCR, MAF, MNF). An interface to the bioconductor image processing package EBImage offers image segmentation functionality.

License GPL-3

Imports Rcpp (>= 0.11.2), ALS, ChemometricsWithR, signal, KernSmooth, graphics, grDevices, stats

Suggests EBImage, knitr, rmarkdown, testthat, tofsimsData, BiocParallel, RColorBrewer

Enhances parallel

LinkingTo Rcpp, RcppArmadillo

VignetteBuilder knitr

biocViews Infrastructure, DataImport, MassSpectrometry, ImagingMassSpectrometry, Proteomics, Metabolomics

RoxygenNote 5.0.1

NeedsCompilation yes

R topics documented:

tofsims-package .............................................................. 4
addFixedWidth ............................................................... 4
addPeaks ................................................................. 5
analysis ................................................................. 6
analysisName ........................................................... 7
baseObject ............................................................ 8
<table>
<thead>
<tr>
<th>R topics documented:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>baseObject,PrComp-method</td>
<td>8</td>
</tr>
<tr>
<td>baseObject,PrinComp-method</td>
<td>9</td>
</tr>
<tr>
<td>binning</td>
<td>9</td>
</tr>
<tr>
<td>bwApply</td>
<td>10</td>
</tr>
<tr>
<td>calibPointNew</td>
<td>11</td>
</tr>
<tr>
<td>calibPoints</td>
<td>12</td>
</tr>
<tr>
<td>calibration</td>
<td>13</td>
</tr>
<tr>
<td>changePeakWidth</td>
<td>14</td>
</tr>
<tr>
<td>check.extension</td>
<td>15</td>
</tr>
<tr>
<td>computeMNF</td>
<td>15</td>
</tr>
<tr>
<td>computeNoise</td>
<td>16</td>
</tr>
<tr>
<td>coordToPixel</td>
<td>17</td>
</tr>
<tr>
<td>coordToPixel,MassImage,numeric-method</td>
<td>17</td>
</tr>
<tr>
<td>covDiffCalc</td>
<td>18</td>
</tr>
<tr>
<td>cReadRawPhi</td>
<td>18</td>
</tr>
<tr>
<td>ctable</td>
<td>19</td>
</tr>
<tr>
<td>dim,MassImage-method</td>
<td>19</td>
</tr>
<tr>
<td>dim,MassSpectra-method</td>
<td>20</td>
</tr>
<tr>
<td>EigenDecompose</td>
<td>20</td>
</tr>
<tr>
<td>extract.header.data</td>
<td>21</td>
</tr>
<tr>
<td>findClosestMatch</td>
<td>21</td>
</tr>
<tr>
<td>findPeakWidth</td>
<td>22</td>
</tr>
<tr>
<td>getTOFs</td>
<td>23</td>
</tr>
<tr>
<td>image</td>
<td>23</td>
</tr>
<tr>
<td>imageMatrix</td>
<td>24</td>
</tr>
<tr>
<td>import</td>
<td>25</td>
</tr>
<tr>
<td>import.raw</td>
<td>26</td>
</tr>
<tr>
<td>instrument</td>
<td>26</td>
</tr>
<tr>
<td>iters</td>
<td>27</td>
</tr>
<tr>
<td>iters,MCR-method</td>
<td>28</td>
</tr>
<tr>
<td>itzipName</td>
<td>28</td>
</tr>
<tr>
<td>itzipName&lt;-</td>
<td>29</td>
</tr>
<tr>
<td>LapackGenEigen</td>
<td>29</td>
</tr>
<tr>
<td>legend.col</td>
<td>30</td>
</tr>
<tr>
<td>look.for.itzip.property</td>
<td>30</td>
</tr>
<tr>
<td>MAF</td>
<td>31</td>
</tr>
<tr>
<td>makeTIC</td>
<td>31</td>
</tr>
<tr>
<td>makeTIC,MassSpectra-method</td>
<td>32</td>
</tr>
<tr>
<td>manualSelectPeaks</td>
<td>32</td>
</tr>
<tr>
<td>MassImage</td>
<td>33</td>
</tr>
<tr>
<td>MassSpectra</td>
<td>34</td>
</tr>
<tr>
<td>MCR-class</td>
<td>35</td>
</tr>
<tr>
<td>MNF</td>
<td>36</td>
</tr>
<tr>
<td>mz,MassSpectra-method</td>
<td>37</td>
</tr>
<tr>
<td>nComp</td>
<td>38</td>
</tr>
<tr>
<td>ndim</td>
<td>39</td>
</tr>
<tr>
<td>ndim,MassSpectra-method</td>
<td>39</td>
</tr>
<tr>
<td>nearestNeighbourMean</td>
<td>40</td>
</tr>
<tr>
<td>nnMean</td>
<td>40</td>
</tr>
<tr>
<td>nnMNF</td>
<td>41</td>
</tr>
<tr>
<td>noPlottingData</td>
<td>42</td>
</tr>
<tr>
<td>noPlottingData,PCA-method</td>
<td>42</td>
</tr>
</tbody>
</table>
### R topics documented:

<table>
<thead>
<tr>
<th>Name</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>nPeaks</td>
<td>43</td>
</tr>
<tr>
<td>nz</td>
<td>43</td>
</tr>
<tr>
<td>overlayPlot</td>
<td>44</td>
</tr>
<tr>
<td>parIndicesSearch</td>
<td>45</td>
</tr>
<tr>
<td>PCA-class</td>
<td>46</td>
</tr>
<tr>
<td>pcaLoadings</td>
<td>46</td>
</tr>
<tr>
<td>pcaMAF</td>
<td>47</td>
</tr>
<tr>
<td>PCAnalysis</td>
<td>47</td>
</tr>
<tr>
<td>pcaScores</td>
<td>48</td>
</tr>
<tr>
<td>peakIDs</td>
<td>49</td>
</tr>
<tr>
<td>PeakList</td>
<td>50</td>
</tr>
<tr>
<td>peakMzs</td>
<td>51</td>
</tr>
<tr>
<td>peakPick</td>
<td>52</td>
</tr>
<tr>
<td>peaks2Spectra</td>
<td>53</td>
</tr>
<tr>
<td>peakWidths</td>
<td>53</td>
</tr>
<tr>
<td>plot</td>
<td>54</td>
</tr>
<tr>
<td>plot,MassImage,missing-method</td>
<td>56</td>
</tr>
<tr>
<td>plot,PeakList,missing-method</td>
<td>56</td>
</tr>
<tr>
<td>points</td>
<td>57</td>
</tr>
<tr>
<td>poissonScaling</td>
<td>58</td>
</tr>
<tr>
<td>PrComp-class</td>
<td>59</td>
</tr>
<tr>
<td>PrinComp-class</td>
<td>60</td>
</tr>
<tr>
<td>readBIF</td>
<td>61</td>
</tr>
<tr>
<td>recalibrate</td>
<td>62</td>
</tr>
<tr>
<td>reduceSpectrumResolution</td>
<td>62</td>
</tr>
<tr>
<td>removePeaks</td>
<td>63</td>
</tr>
<tr>
<td>resids</td>
<td>64</td>
</tr>
<tr>
<td>resids,MCR-method</td>
<td>65</td>
</tr>
<tr>
<td>RSS</td>
<td>65</td>
</tr>
<tr>
<td>RSS,MCR-method</td>
<td>66</td>
</tr>
<tr>
<td>scale</td>
<td>66</td>
</tr>
<tr>
<td>show,MassImage-method</td>
<td>67</td>
</tr>
<tr>
<td>show,MassSpectra-method</td>
<td>67</td>
</tr>
<tr>
<td>show,PeakList-method</td>
<td>68</td>
</tr>
<tr>
<td>smootherGolay</td>
<td>68</td>
</tr>
<tr>
<td>smootherSpline</td>
<td>69</td>
</tr>
<tr>
<td>smoothScatter</td>
<td>70</td>
</tr>
<tr>
<td>SNR</td>
<td>71</td>
</tr>
<tr>
<td>subset</td>
<td>72</td>
</tr>
<tr>
<td>unitMassPeaks</td>
<td>72</td>
</tr>
<tr>
<td>validMassImageObject</td>
<td>73</td>
</tr>
<tr>
<td>validMassSpectraObject</td>
<td>74</td>
</tr>
<tr>
<td>validPCAObject</td>
<td>74</td>
</tr>
<tr>
<td>validPeakListObject</td>
<td>75</td>
</tr>
<tr>
<td>xdim</td>
<td>75</td>
</tr>
<tr>
<td>xdim,MassImage-method</td>
<td>76</td>
</tr>
<tr>
<td>xdim,PCA-method</td>
<td>76</td>
</tr>
<tr>
<td>xdim&lt;-</td>
<td>77</td>
</tr>
<tr>
<td>xy</td>
<td>77</td>
</tr>
<tr>
<td>xySpec</td>
<td>78</td>
</tr>
<tr>
<td>ydim</td>
<td>79</td>
</tr>
<tr>
<td>ydim,MassImage-method</td>
<td>79</td>
</tr>
</tbody>
</table>
addFixedWidth

dim,PCA-method .................................................. 80
dim< ................................................................. 80
zdim ................................................................. 81
zdim,MassSpectra-method ........................................ 81

Index 82

tofsims-package ToF-SIMS Toolbox (tofsims)

Description

ToF-SIMS Toolbox

Details

Package: tofsims
Type: Package
Version: 0.99.2
Date: 15-01-2016
License: GPL-3
LazyLoad: yes

Toolbox for Time-of-Flight Secondary Ion Mass-Spectrometry (ToF-SIMS) data processing and analysis. The package facilitates importing of raw data files, loading preprocessed data and a range of multivariate analysis methods that are most commonly applied in the ToF-SIMS community.

Author(s)

Lorenz Gerber <lorenz.gerber@slu.se>
Viet Mai Hoang <hviet.0906@gmail.com>

addFixedWidth Generic method to add/update peak width

Description

This method will update current upper/lower width for all peaks

Usage

addFixedWidth(object, lowerWidth, upperWidth)

## S4 method for signature 'PeakList,numeric,numeric'
addFixedWidth(object, lowerWidth, upperWidth)
addPeaks

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>PeakList object</td>
</tr>
<tr>
<td>lowerWidth</td>
<td>numeric</td>
</tr>
<tr>
<td>upperWidth</td>
<td>numeric</td>
</tr>
</tbody>
</table>

Value

object PeakList with updated/new peak widths

Examples

```r
library(tofsimsData)
data(tofsimsData)
testSpectra<-reduceSpectrumResolution(object = testSpectra, everyN = 4, mode = 'keep')
testSpectra<-smootherSpline(testSpectra, stepsize = 10, spar = 0.3)
testSpectra<-smootherGolay(testSpectra, p = 3, n = 5)
testSpectra<-peakPick(testSpectra, span = 100)
testSpectra<-addFixedWidth(testSpectra, 0.2, 0.2)
plot(testSpectra, , mzRange=c(38.5,40.5), type = 'l')
```

Description

This method will allow user to plot and add peaks manually. This method will take all parameters of PeakList plot method.

Usage

```r
addPeaks(object, mzs, width, ...)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>PeakList object</td>
</tr>
<tr>
<td>mzs</td>
<td>numeric vector M/z’s where peaks shall be added</td>
</tr>
<tr>
<td>width</td>
<td>fixed value to add (m/z)</td>
</tr>
<tr>
<td>...</td>
<td>further args</td>
</tr>
</tbody>
</table>

Value

object updated PeakList object
Examples

```r
library(tofsimsData)
data(tofsimsData)
testPeakList <- PeakList(analysisName = analysisName(testSpectra),
instrument = instrument(testSpectra),
nz = nz(testSpectra),
calibration = calibration(testSpectra),
calibPoints = calibPoints(testSpectra),
mz = mz(testSpectra),
peakIDs = NULL,
peakMzs = NULL)
par(mfcol=c(1,2))
plot(testPeakList, mzRange=c(25,32), type = "l")
testPeakList <- addPeaks(testPeakList, mzs=26:31, width=0.4)
plot(testPeakList, mzRange=c(25,32), type = "l")
```

analysis, slot of MassSpectra class objects

Description

analysis, slot of MassSpectra class objects

Usage

```r
analysis(object, noAccess, ...)  # S4 method for signature 'MassSpectra,missing'
```

Arguments

- `object` object of class MassSpectra
- `noAccess` numeric access number to analysis slot
- `...` additional args
- `value` object to be put in analysis slot

Value

summary or content of analysis slot

See Also

- `object MassSpectra` other slots `mz nz analysisName instrument calibPoints calibration`
Examples

```r
library(tofsimsData)
data(tofsimsData)
testImage<-PCAnalysis(testImage, nComp = 3)
## obtain summary of analysis slot content
analysis(testImage)
```

<table>
<thead>
<tr>
<th>analysisName</th>
<th>analysisName, slot of MassSpectra class objects</th>
</tr>
</thead>
</table>

Description

`analysisName, slot of MassSpectra class objects`

Usage

```r
analysisName(object, ...)
```

```r
analysisName(object) <- value
```

```r
## S4 method for signature 'MassSpectra'
analysisName(object)
```

```r
## S4 replacement method for signature 'MassSpectra'
analysisName(object) <- value
```

Arguments

- `object` : object of class MassSpectra
- `...` : further args
- `value` : character replacement value for slot analysisName

Value

content of analysisName slot

See Also

object `MassSpectra` other slots `mz analysis nz instrument calibPoints calibration`

Examples

```r
library(tofsimsData)
data(tofsimsData)
## access name of analysis
analysisName(testSpectra)
## replace name of analysis
analysisName(testSpectra) <- 'sample001_pos001_settings_default'
analysisName(testSpectra)
```
**baseObject, PrComp-method**

**Description**
generic accessor method `baseObject`

**Usage**

```r
baseObject(object)
```

**Arguments**

- `object` helper for `prcomp` and `princomp` wrappers

**Value**

`baseObject`

---

**baseObject, PrComp-method**

*constructor for PrComp*

**Description**

constructor for `PrComp`

**Usage**

```r
## S4 method for signature 'PrComp'
baseObject(object)
```

**Arguments**

- `object` object of class

**Value**

object of class `PrComp`
Description

constructor for PrinComp

Usage

## S4 method for signature 'PrinComp'
baseObject(object)

Arguments

object object with class

Value

object of class PrinComp

Description

binning

Usage

binning(object, binningFactor, ...)

## S4 method for signature 'MassImage'
binning(object, binningFactor = 2)

Arguments

object object of class MassImage
binningFactor numeric factor for binning (2, 4, etc)
... additional args

Details

binning is used to reduce the resolution/size of MassImage objects. Optionally mclapply from the parallel package is used to speed up processing time.

Value

binned object of class MassImage
Examples

```r
library(BiocParallel)
testImage <- MassImage('dummy')
par(mfcol=c(1,2), oma=c(0,0,0,0), mar=c(0,0,0,0))
image(testImage)
## the following param will cause to run non parallel
register(SerialParam(), default=TRUE)
testImage <- binning(testImage, binningFactor = 4)
image(testImage)
## Not run:
library(tofsimsData)
data(tofsimsData)
par(mfcol=c(1,2), oma=c(0,0,0,0), mar=c(0,0,0,0))
image(testImage)
testImage <- binning(testImage, binningFactor = 4)
image(testImage)
## End(Not run)
```

Description

bwApply allow to get new object from a black / white matrix All NZs at black positions will be taken

Usage

```r
bwApply(object, bwMatrix)
## S4 method for signature 'MassSpectra,matrix'
bwApply(object, bwMatrix)
```

Arguments

- object: object of class MassImage
- bwMatrix: matrix with boolean or numeric 1 and 0

Value

- object of class MassImage multiplied with B/W matrix

Examples

```r
library(tofsimsData)
data(tofsimsData)
testImage <- PCAnalysis(testImage, nComp = 2)
library(EBImage)
mask <- thresh(imageMatrix(analysis(testImage, noAccess = 1), comp = 1), w = 15, h = 15)
# inverse of mask
mask <- (mask - 1)^2
par(mfcol=c(1,2), oma=c(0,0,0,0), mar=c(0,0,0,0))
image(testImage)
image(bwApply(testImage, mask))
```
calibPointNew

Generic method calibPointNew that modifies slot calibPoints

description

Generic method calibPointNew that modifies slot calibPoints

calibPointNew is a method to set a new mass calibration point

Usage

```r
calibPointNew(object, mz, reset = FALSE, value = NULL)
```

## S4 method for signature 'MassSpectra,numeric'

calibPointNew(object, mz, reset = FALSE, value = NULL)

Arguments

- **object**: MassSpectra object
- **mz**: the m/z value to be specified with a TOF value
- **reset**: shall the list of calibration points be reset
- **value**: TOF value to be assigned to mz

Details

CalibPointNew is a method to set a new mass calibration point. When value is not provided as argument, the TOF for the chosen mz value has to be chosen interactively by mouse.

Value

Call by reference, hence MassSpectra object with new calib point

object MassSpectra with added/updated calibration points

Examples

```r
library(tofsimsData)
data(tofsimsData)
testSpectra <- calibPointNew(testSpectra, mz = 15, value = 15.01551)
testSpectra <- calibPointNew(testSpectra, mz = 181, value = 181.0228)
calibPoints(testSpectra)
par(mfcol=c(1,2))
plot(testSpectra, mzRange=c(38.5,40.5), type='l')
testSpectra <- recalibrate(testSpectra)
plot(testSpectra, mzRange=c(38.5,40.5), type='l')
```
**Description**

calibPoints, slot of MassSpectra class objects

**Usage**

calibPoints(object)

calibPoints(object) <- value

## S4 method for signature 'MassSpectra'
calibPoints(object)

## S4 replacement method for signature 'MassSpectra'
calibPoints(object) <- value

**Arguments**

- object: object of class MassSpectra
- value: data.frame replacement values for calibPoints slot

**Value**

contents of slot calibPoints

**See Also**

object MassSpectra other slots mz analysis analysisName instrument nz calibration

**Examples**

```r
library(tofsimsData)
data(tofsimsData)
testSpectra<-calibPointNew(testSpectra, mz = 15, value = 15.0232)
testSpectra<-calibPointNew(testSpectra, mz = 181, value = 181.0569)
## access 'calibPoint' slot of 'MassSpectra' object
calibPoints(testSpectra)
## replacing values in the 'calibPoint' slot
calibPoints(testSpectra)[2,2]<-297000
calibPoints(testSpectra)
```
**calibration**

calibration, *slot of MassSpectra class objects*

**Description**

Generic setter for slot calibration<-  

**Usage**

calibration(object)

calibration(object) <- value

## S4 method for signature 'MassSpectra'
calibration(object)

## S4 replacement method for signature 'MassSpectra'
calibration(object) <- value

**Arguments**

- **object**: object of class MassSpectra
- **value**: data.frame with replacement values for calibration slot

**Value**

content of calibration slot

**See Also**

- object MassSpectra
- other slots mz analysis analysisName instrument calibPoints nz

**Examples**

```r
library(tofsimsData)
data(tofsimsData)
## access calibration slot
 calibration(testSpectra)
## replacing the values of the 'calibration' slot is possible but it makes at the moment no sense as it doesn't change the actual mass calibration. The 'calibration' slot is just used to store the values while 'recalibration' uses the values from 'calibPoints' slot.
calibration(testSpectra) <- data.frame(intercept = 21420, slope = 20480)
calibration(testSpectra)
```
changePeakWidth

Description
method changePeakWidth

Usage
changePeakWidth(object, selectMz, lowerWidth, upperWidth, ...)

## S4 method for signature 'PeakList,missing,missing,missing'
changePeakWidth(object, selectMz, lowerWidth, upperWidth, ...)

## S4 method for signature 'PeakList,numeric,numeric,numeric'
changePeakWidth(object, selectMz, lowerWidth, upperWidth, ...)

Arguments
- object: PeakList object
- selectMz: numeric change width of peak closest to selectMz
- lowerWidth: numeric lower width value in mass units
- upperWidth: numeric upper width value in mass units
- ... additional args

Details
method changePeakWidth is used to modify the peak width of an individual peak it should be called with the argument mzRange to zoom into the region of interest for choosing the peak. Then two further clicks for choosing the (new) lower and upper peak widths.

Value
PeakList object with updated peak widths

Examples
library(tofsimsData)
data(tofsimsData)
testPeakList<-PeakList(analysisName = analysisName(testSpectra),
instrument = instrument(testSpectra),
nz = nz(testSpectra),
calibration = calibration(testSpectra),
calibPoints = calibPoints(testSpectra),
mz = mz(testSpectra),
peakIDs = NULL,
peakMzs = NULL)
par(mfcol=c(1,2))
testPeakList<-addPeaks(testPeakList, mzs=26:31, width=0.4)
check.extension

Description

Function to check file extension

Usage

check.extension(filepath, extension)

Arguments

filepath character
extension character

Details

This function is used for checking file extension

Value

boolean

Author(s)

Lorenz Gerber, Viet Mai Hoang

computeMNF

compute MNF

Description

compute MNF, helper for MNF/nnMNF

Usage

computeMNF(nzData = NULL, noise = NULL, SNR = NULL, ind = NULL, iter = TRUE, limitSNR = NULL, covNoise = NULL)
**computeNoise**

**Arguments**

- `nzData` : matrix
- `noise` : matrix
- `SNR` : numeric
- `ind` : numeric
- `iter` : boolean
- `limitSNR` : numeric
- `covNoise` : matrix

**Details**

This is a helper function for the MNF/nnMNF function and originates from the `mzImage` package.

**Value**

MNF transform

---

**computeNoise**  

**computeNoise**

**Description**

computeNoise determinates the noise by nearest neighbour estimate. This is a helper function for the nnMNF method.

**Usage**

```
computeNoise(stat, x, y)
```

**Arguments**

- `stat` : unknown
- `x` : unknown
- `y` : unknown

**Details**

computeNoise determinates the noise by nearest neighbour estimate. This is a helper function for the nnMNF method and originates from the `mzImage` package.

**Value**

- matrix numeric noise
coordToPixel

coordToPixel translates xy coordinates from the locator() function to cell coordinates from the image function. Origo is according to ToF-SIMS images the upper left corner.

Usage

coordToPixel(object, xy)

Arguments

object of class MassImage
xy numeric vector with x/y locator coordinate

Value

xy coordinate of MassImage pixels
covDiffCalc

covDiffCalc calculates a x/y shift covariance matrix of a multispectral image according to Switzer and Green 1984.

Description

covDiffCalc calculates a x/y shift covariance matrix of a multispectral image according to Switzer and Green 1984.

Usage

covDiffCalc(nzData, dataObject)

Arguments

nzData unknown

dataObject unknown

Value

shifted cov matrix

cReadRawPhi

Ulvac phi ToF-SIMS raw data import

Description

Function to read raw data from the ulvac-phi trift TOF-SIMS

Usage

cReadRawPhi(analysisName, mode = c("spectra", "imagepeaks", "image"),
PeakListobj = c(), ...)

Arguments

analysisName character

mode character

PeakListobj object of class PeakList

... additional args

Details

This import function works on data recorded on the ulvac-phi trift ToF-SIMS with WinCadence software version V4.2. Other versions mostl likley will not work. In the current version, data has to be imported with 16bit word length, then converted to 64bit binary and finally converted and read with the word lengths of the respective variables. Currently, the data is unit mass binned with bins of size one from -0.5 to + 0.5.
Value

parsed rawdata for further processing

Author(s)

Lorenz Gerber, Viet Mai Hoang

cTable

cTable is a C++ implementation to make contingency tables

Description

cTable is a C++ implementation to make contingency tables

Usage

cTable(vect)

Arguments

vect NumericVector

Value

vars freqs

dim,MassImage-method

method dim for MassImage

Description

method dim for MassImage

Usage

## S4 method for signature 'MassImage'
dim(x)

Arguments

x object of class MassImage

Value

vector numeric
### dim,MassSpectra-method

*method definition 'dim' for 'MassSpectra' dim is a primitive*

#### Description

method definition 'dim' for 'MassSpectra' dim is a primitive

#### Usage

```r
## S4 method for signature 'MassSpectra'
dim(x)
```

#### Arguments

- `x` object object of type MassSpectra

#### Value

numeric value

### EigenDecompose

*EigenDecompose for the MNF analysis*

#### Description

EigenDecompose for the MNF analysis

#### Usage

```r
EigenDecompose(A, B, startIndex, endIndex)
```

#### Arguments

- `A` NumericMatrix
- `B` NumericMatrix
- `startIndex` int
- `endIndex` int

#### Value

eigval eigvec mA mB
extract.header.data

extract variable names and values from Ulvac-phi ToF-SIMS datafile headers

Description
Extracting the data from a Ulvac-phi Tof-SIMS raw header character string.

Usage
extract.header.data(header)

Arguments
header
header as a raw character string

Details
This function takes a raw header character string read by get.raw.header() as input and extracts variable names and values. values are currently forwarded just as character string. This is a helper function for read.raw.phi.

Value
list with two vectors containing variable names and values as characters

Author(s)
Lorenz Gerber

findClosestMatch
Find single value 'toMatch' in vector 'MatchIn'

Description
Find single value 'toMatch' in vector 'MatchIn'

Usage
findClosestMatch(toMatch, matchIn, twoMatch)

Arguments
toMatch
numeric
matchIn
vector numeric
twoMatch
character 'upper' or 'mean'

Value
numeric ID of match
Description

generic method findPeakWidth

Usage

findPeakWidth(object, p = 3, n = 5, span = 100, widthExtLower = 1.5, widthExtUpper = 1.75, ...)

## S4 method for signature 'PeakList'
findPeakWidth(object, p = 3, n = 199, span = 100, widthExtLower = 1.7, widthExtUpper = 2, ...)

Arguments

object object of class PeakList
p numeric value for savitzky-golay filter on first derivate
n numeric value for savitzky-golay filter on first derivate
span numeric smoothing for determining local minima/maxima values
widthExtLower numeric factor to extend lower peak width
widthExtUpper numeric factor to extend upper peak width
... additional args

Details

This method uses signal processing to determine lower and upper peak width limits based on local max/min detection of the first derivate next to peak center values. The initial code for local min/max detection is adapted from the CRAN package 'ChemometricsWithR'.

Value

object of class PeakList with updated peaks

Examples

library(tofsimsData)
data(tofsimsData)
testPeakList<-PeakList(analysisName = analysisName(testSpectra), instrument = instrument(testSpectra), nz = nz(testSpectra), calibration = calibration(testSpectra), calibPoints = calibPoints(testSpectra), mz = mz(testSpectra), peakIDs = NULL, peakMzs = NULL)
par(mfcol=c(1,2))
getTOFs

generic method to calculate and get TOFs

Description

generic method to calculate and get TOFs

Usage

getTOFs(object)

## S4 method for signature 'MassSpectra'
getTOFs(object)

Arguments

object object of class MassSpectra

Value

vector with ToFs

vector numeric with TOF values

Examples

library(tofsimsData)
data(tofsimsData)
timeOfFlight <- getTOFs(testSpectra)
head(timeOfFlight)

image

set a generic method for image

Description

set a generic method for image

Method to visualize an IMS Mass Image of class MassImage

image for PCA class type loading plots
Usage

image(x, ...)

## S4 method for signature 'MassImage'
image(x, ..., mzSelect = NULL)

## S4 method for signature 'PCA'
image(x, comp, ...)

Arguments

x object object with image data
...
additional args
mzSelect vector, which m/z to combine for visualization. if none are chosen, the TIC is shownhel
comp numeric which component to visualize

Value

graphical output
image plot of the ToF SIMS image data

Examples

testImage<-MassImage('dummy')
image(testImage)
## Not run:
library(tofsimsData)
data(tofsimsData)
image(testImage)
## End(Not run)
library(tofsimsData)
data(tofsimsData)
testImage<-PCAAnalysis(testImage,3)
image(analysis(testImage, 1), comp = 1)
Usage

imageMatrix(object, ...)

## S4 method for signature 'MassImage'
imageMatrix(object)

## S4 method for signature 'PCA'
imageMatrix(object, comp, ...)

Arguments

object object of class MassImage
...
additional args
comp numeric which component

Value

numeric matrix
matrix numeric

Examples

library(tofsimsData)
data(tofsimsData)
## the TIC matrix can be extracted
dataMatrix <- imageMatrix(testImage)
dim(dataMatrix)
## the matrix can be visualized with the
## normal image() function
image(dataMatrix)

import is the C++ code for importing iontof raw data

import

Description

import is the C++ code for importing iontof raw data

Usage

import(rFilename, fType, imageSize, upperMass)

Arguments

rFilename CharacterVector
fType CharacterVector
imageSize int
upperMass int

Value

imported binary raw data
import.raw  

**Raw data import**

**Description**
Function to read raw data.

**Usage**

```r
import.raw(analysisName, mode = c("spectra", "imagepeaks"),
            PeakListobj = c(), untilScan = NULL, ...)
```

**Arguments**
- `analysisName` character
- `mode` character
- `PeakListobj` object of class PeakList
- `untilScan` numeric read data up to which scan number
- `...` additional args

**Details**
This import function works on GRD and ITZIP format

**Value**
parsed rawdata for further processing

**Author(s)**
Lorenz Gerber, Viet Mai Hoang

---

```
instrument

**Description**
instrument, slot of MassSpectra class objects

**Usage**

```r
instrument(object, ...)
iinstrument(object) <- value
```

```r
## S4 method for signature 'MassSpectra'
iinstrument(object)
iinstrument(object) <- value
```
```
### Arguments

- **object** object of class MassSpectra
- ... additional args
- **value** character name of instrument used in the experiment

### Value

content of instrument slot

### See Also

- object MassSpectra other slots mz analysis analysisName nz calibPoints calibration

### Examples

```r
library(tofsimsData)
data(tofsimsData)
## access instrument slot in MassSpectra objects
instrument(testSpectra)
## values for the 'instrument' slot can currently be
## 'iontof' or 'ulvacphi'. It is not advisable to
## change those values manually
```

---

**iters**

**generic accessor for iters slot**

### Description

generic accessor for iters slot

### Usage

```r
iters(object)
```

### Arguments

- **object** object of class MCR

### Value

content of iters slot
MCR accessor `iters`,

**Description**

MCR accessor `iters`.

**Usage**

```r
## S4 method for signature 'MCR'
iters(object)
```

**Arguments**

- `object` object of class MCR

**Value**

`iters` from object

---

**itzipName**

defining generic accessor method for "itzipName"

**Description**

defining generic accessor method for "itzipName"

**Usage**

`itzipName(object)`

**Arguments**

- `object` internal

**Value**

content of itzipName
itzipName<-  
generic for setter itzipName

Description

generic for setter itzipName

Usage

itzipName(object) <- value

Arguments

object  internal
value   internal

Value

object with updated itzipName slot

LapackGenEigen  LapackGenEigen

Description

LapackGenEigen is helper function for MNF and nnMNF

Usage

LapackGenEigen(A, B, IL = 1, IU = 3)

Arguments

A       matrix
B       matrix
IL      int start index
IU      int end index

Details

LapackGenEigen is adapted from the mzImage package. While it initially used dsygvx from the LAPACK library, it is now ported to RcppArmadillo, using the eig_pair function.

Value

list with values, vectors and info
**legend.col**

**Description**

`legend.col` is a helper for the plot function of Scoreplots. It allows to visualize a third component by a color range. `legend.col` plots the color range as legend on the side of the plot.

**Usage**

```r
legend.col(col, lev)
```

**Arguments**

- `col` character color
- `lev` character levels

**Value**

graphical output

---

**look.for.itzip.property**

*Get ITZIP property value*

**Description**

Function to extract value by passing property name.

**Usage**

```r
look.for.itzip.property(itzipName, itzipProperties)
```

**Arguments**

- `itzipName` character
- `itzipProperties` character

**Details**

This function is used to get ITZIP property value by passing its name.

**Value**

character value from itzipProperties corresponding itzipName

**Author(s)**

Lorenz Gerber, Viet Mai Hoang
Class MAF

Class MAF contains methods for Maximum Autocorrelation Factors analysis
MAF is a Maximum Autocorrelation Factor Analysis

Usage

MAF(dataObject, nComp = 10, usePCA = TRUE)

Arguments

dataObject object of type MassImage
nComp integer number of components
usePCA boolean use PCA

Details

Class MAF contains methods for Maximum Autocorrelation Factors analysis
MAF is a Maximum Autocorrelation Factor Analysis. The code is implemented from the publication of

Value

object of type MAF

Examples

library(tofsimsData)
data(tofsimsData)
## Not run: data(tofsimsData)
MAF(testImage,5,TRUE)
image(analysis(testImage,1),comp = 1)
## End(Not run)

makeTIC generic for makeTIC

Description

generic for makeTIC

Usage

makeTIC(object)
**makeTIC, MassSpectra-method**

*Method makeTIC for MassSpectra Class*

**Description**
Method makeTIC sums up all Mass Spectra in the called Mass Spectra object.

**Usage**
```r
## S4 method for signature 'MassSpectra'
makeTIC(object)
```

**Arguments**
- `object` object of class MassSpectra

**Value**
object of class MassSpectra with just one spectra, the TIC

---

**manualSelectPeaks**

*This method is base method for plotting and manual select data*

**Description**
This method is base method for plotting and manual select data.

**Usage**
```r
manualSelectPeaks(object, n = 512, ...)
```

**Arguments**
- `object` object of type PeakList
- `n` numeric
- `...` additional args

**Value**
numeric x coordinates
**Description**

Class MassImage contains the information to shape a number of mass spectra into an image. MassImage is also the call to the class constructor. It is used for importing both BIF/BIF6 and raw image data.

**Usage**

```r
MassImage(select = c("ulvacbif", "iontofbif", "iontofgrdpeaks", "ulvacrawpeaks", "dummy"), analysisName, PeakListobj = c(), untilScan = NULL, ...)
```

**Arguments**

- `select` character, 'ulvacbif', 'iontofbif', 'iontofgrdpeaks', 'ulvacrawpeaks', 'dummy'
- `analysisName` character, name of analysis
- `PeakListobj` PeakList class object, used as peaklist for rawdata import
- `untilScan` integer or NULL to determine number of ToF-SIMS scans to import
- `...` additional args

**Details**

Class MassImage inherits from the classes MassAnalysis and MassSpectra. It contains the information to shape a number of mass spectra into an image.

MassImage is the user class constructor to obtain a MassImage object. Data can be imported from BIF or raw data files (Iontof or Ulvacphi). To import raw data, a MassSpectra object with a valid PeakList object has to be provided as argument.

**Value**

object of class MassImage

**Slots**

- `xy` vector giving the pixel dimension of the image

**Author(s)**

Lorenz Gerber <lorenz.gerber@slu.se>
Examples

```r
# creating dummy data
testImage <- MassImage('dummy')
image(testImage)

## Not run:
# import of rawdata
# first a PeakList object has to be created
library(tofsimsData)
data(tofsimsData)
testSpectra <- calibPointNew(testSpectra, mz = 15, value = 15.01551)
testSpectra <- calibPointNew(testSpectra, mz = 181, value = 181.0228)
testSpectra <- recalibrate(testSpectra)
testSpectra <- unitMassPeaks(testSpectra, mzRange = c(1,200), widthAt = c(15, 181),
factor = c(0.4, 0.6), lower = c(14.97, 15.05), upper = c(180.84, 181.43))

# obtaining the path to the raw data file in 'tofsims' package
importFile <- system.file("rawdata", "trift_test_001.RAW", package = "tofsimsData")
rawImportedImage <- MassImage('ulvacrawpeaks', importFile,
PeakListobj = testSpectra)
image(rawImportedImage)

## End(Not run)
```

MassSpectra Class

MassSpectra is the main data container in the tofsims package as it contains the individual mass spectra.

MassSpectra is also the call to class constructor. It is used for importing high-resolution mass spectra from raw data.

Usage

```r
MassSpectra(select = c("ulvacraw", "iontofgrd", "dummy"), analysisName, ...)
MassSpectra(select = c("ulvacraw", "iontofgrd", "dummy"), analysisName, ...)
```

Arguments

- `select` character, 'ulvacraw', 'iontofgrd', 'dummy'
- `analysisName` character, the (file)name of the dataset
- `...` additional args

Details

Class MassSpectra is the main data container of the tofsims package, containing the individual mass spectra in the slot nz. Additional metadata about the analysis can be found in the slots analysisName and instrument. Values for slope and intercept of the linear mass calibration equation are stored in the slot calibration. The M/z values can be found in mz. calibration allows calculating from M/z values back to times-of-flight. The slot calibPoints is used to recalibrate the dataset. It contains a data.frame with the columns mz and TOF. The slot analysis of type list,
is used as a container for data analysis objects. Typically, object of the class MassSpectra are constructed during data import using the user constructor function with the same name as the class, MassSpectra.

MassSpectra is also the call to class constructor. It is used for importing high-resolution mass spectra from raw data.

Value

object of class MassSpectra

Slots

analysisName character vector with the import filename
instrument character vector type of instrument used in the experiment
calibration data frame for numerics slope and intercept of the mass calibration
calibPoints data frame for time of flight to mass to charge calibration
nz matrix with rows of ion counts and columns as toftimes or mass to charge ratios
mz vector same length as columns in nz for mass to charge values

Author(s)

Lorenz Gerber <lorenz.gerber@slu.se>

Examples

## Not run:
## access rawdata in tofsims package
library(tofsimsData)
importFile <- system.file("rawdata", "trift_test_001.RAW", package = "tofsimsData")
MassSpectra('ulvacraw', importFile)

## End(Not run)
## create dummy MassSpectra object
MassSpectra('dummy')

MCR-class

Class MCR

Description

Class MCR contains methods for ’Multivariate Curve Resolution by Alternate Least Squares’
opAMCR is a MCR-ALS function using the Orthogonal Projection Approach from

Usage

opaMCR(dataObject, opaComps, maxiter = 10)

Arguments

dataObject object of class MassImage
opaComps numeric number of components for the opa method
maxiter numeric how many iterations
Details

Class MCR contains methods for 'Multivariate Curve Resolution by Alternate Least Squares'
opAMCR uses the function ChemometricsWithR::opa() (Orthogonal Projection Approach, CRAN
package 'ChemometricsWithR') for start estimates of pure spectra and ALS::als() (CRAN pack-
age 'ALS') as MCR-ALS implementation. This method is doing fine with images up to 256x256
pixels. For larger images, memory usage becomes unreasonably high.

Value

object of class MCR

Slots

RSS numeric residual sum of squares
resids matrix with residuals
iters numeric number of iterations

Author(s)

Lorenz Gerber <lorenz.gerber@slu.se>

Examples

testImage<-MassImage(’dummy’)
testImage<-opaMCR(testImage, 2, 2)
image(analysis(testImage,1), comp = 1)
## Not run:
library(tofsimsData)
data(tofsimsData)
testImage<-MCR(testImage, 5, 5)
image(analysis(testImage,1), comp = 1)
## End(Not run)

MNF

Description

Class MNF contains methods for Maximum Autocorrelation Factors analysis
This method calculates MNF transform using the diagonal shift method from Switzer and Green
(1984) to estimate the noise.

Usage

MNF(dataObject)

Arguments

dataObject object of type massImage
mz, MassSpectra-method

Details

Class MNF contains methods for Maximum Autocorrelation Factors analysis.

Minimum Noise Fraction according Green et al. (1988) using diagonal shift method from Switzer and Green (1984) to estimate the noise. As the original package mzImage from Stone et al. 2012 is no longer maintained, we use it as code base for the present version. The C code was implemented through Rcpp (Eddelbuettel and Francois, 2011). Practically, this method uses covDiffCalc from the MAF method. The present function is a user constructor that will create a new analysis slot in the chosen MassSpectra/MassImage object.

Value

object of class MNF

Examples

testImage<-MassImage('dummy')
testImage<-MNF(testImage)
image(analysis(testImage,1), comp = 1)
## Not run:
library(tofsimsData)
data(tofsimsData)
MNF(testImage)
image(analysis(testImage,1), comp = 1)
## End(Not run)

mz, MassSpectra-method  mz getter method

Description

mz getter method

mz setter method

Usage

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

## S4 replacement method for signature 'MassSpectra'
mz(object) <- value

Arguments

object of type MassSpectra
value double mass to charge ratio

Value

MassSpectra object with updated mz slot
Examples

```r
library(tofsimsData)
data(tofsimsData)
## access the mz values fo each spectra point
mz(testSpectra)[1:100]
## replace a mz value
mz(testSpectra)[1] <- 0.000025
mz(testSpectra)[1:100]
```

---

**nComp**

generic accessor method for slot nComp

Description

generic accessor method for slot nComp

PCA accessor nComp, number of component

Usage

```r
nComp(object)

## S4 method for signature 'PCA'
nComp(object)
```

Arguments

- `object` object of class PCA

Value

- contents of nComp slot
- numeric number of components

Examples

```r
library(tofsimsData)
data(tofsimsData)
testImage<-PCAnalysis(testImage,4)
nComp(analysis(testImage,1))
```
**Description**

generic accessor method for slot ndim

**Usage**

```r
da.im(object)
```

**Arguments**

- `object`: object of class MassSpectra

**Value**

contents of slot ndim

---

**ndim,MassSpectra-method**

method definition 'ndim' on 'MassSpectra'

**Description**

method definition 'ndim' on 'MassSpectra'

**Usage**

```r
## S4 method for signature 'MassSpectra'
da.im(object)
```

**Arguments**

- `object`: object of type MassSpectra

**Value**

numeric value
Description

nearestNeighbourMean helper for nnMNF

Usage

nearestNeighbourMean(x)

Arguments

x unknown see mzimage

Details

function from mzimage to calculate nearest neighbour means

Value

matrix numeric nearest neighbours

---

**nnMean**

nnMean is C++ code for calculating nearest neighbour means in a 2D matrix

Description

nnMean is C++ code for calculating nearest neighbour means in a 2D matrix

Usage

nnMean(y, nrows, ncols)

Arguments

y NumericVector

nrows int

ncols int

Value

eY
nnMNF

Class nnMNF

Description
Class nnMNF contains methods for Maximum Autocorrelation Factors analysis
This method calculates MNF transform using an nearest neighbour estimate as implemented in
mzImage from Stone et al. (2012).

Usage
nnMNF(dataObject, limitSNR = 1.5)
nnMNF(dataObject, limitSNR = 1.5)

Arguments
dataObject object of type MassImage
limitSNR numeric

Details
Class nnMNF contains methods for Maximum Autocorrelation Factors analysis
Minimum Noise Fraction according Green et al. (1988) but using a nearest neighbour estimate for
the noise determination as seen in the package mzImage from Stone et al. (2012). As the mentioned
package is no longer maintained, we used an archived version as code base for a new version. The
C code was implemented through Rcpp (Eddelbuettel and Francois, 2011). The present function is
a user constructor that will create a new analysis slot in the chosen MassSpectra/MassImage object.

Value
object of class MNF

Examples

testImage<-MassImage('dummy')
testImage<-MNF(testImage)
image(analysis(testImage,1), comp = 1)
## Not run:
library(tofsimsData)
data(tofsimsData)
testImage<-nnMNF(testImage)
image(analysis(testImage,1), comp = 1)
## End(Not run)
noPlottingData, PCA-method

generic method for 'noPlottingData' aka 'is.null'

Description

generic method for 'noPlottingData' aka 'is.null'

Usage

noPlottingData(object)

Arguments

object object of class PCA

Value

boolean validity check of PCA object

---

noPlottingData, PCA-method

Check NULL PCA object

Description

Check NULL PCA object

Usage

## S4 method for signature 'PCA'
noPlottingData(object)

Arguments

object object of class PCA

Value

boolean validity check of class PCA object
nPeaks

generic method for nPeaks

nPeaks accessor/getter nPeaks for PeakList Class

Usage

nPeaks(object)

## S4 method for signature 'PeakList'
nPeaks(object)

Arguments

object object of class PeakList

Value

integer value for number of peaks

Examples

library(tofsimsData)
data(tofsimsData)
testSpectra<-calibPointNew(testSpectra, mz = 15, value = 15.01551)
testSpectra<-calibPointNew(testSpectra, mz = 181, value = 181.0228)
testSpectra<-recalibrate(testSpectra)
testSpectra<-unitMassPeaks(testSpectra, mzRange = c(1,200), widthAt = c(15, 181), factor = c(0.4, 0.6), lower = c(14.97, 15.05), upper = c(180.84, 181.43))
nPeaks(testSpectra)

---

nz

nz, slot of MassSpectra class objects

Description

nz, slot of MassSpectra class objects

Usage

nz(object, mzRange = NULL)
nz(object) <- value

## S4 method for signature 'MassSpectra,missing'
nz(object, mzRange = NULL)

---
### overlayPlot

#### Arguments

- **object** object of class MassSpectra
- **mzRange** vector numeric mass values for nz matrix
- **value** matrix replacement values for nz

#### Value

numeric matrix, content of nz

#### See Also

- object: MassSpectra
- other slots: mz analysis, analysisName, instrument, calibPoints, calibration

#### Examples

```r
library(tofsimsData)
data(tofsimsData)
## access main data slot
testSpectra <- nz(testSpectra)[,1:1000]
```

### overlayPlot

#### Description

This function takes as input a list with objects of type MassSpectra. The easiest way to obtain the input data, is to use mclapply from the parallel package.

#### Usage

```r
overlayPlot(objectList, ...)
```

#### Arguments

- **objectList** list with object of type MassSpectra
- **...** additional args
- **type** character type of plot, usually 'l'
- **mzRange** vector numeric lower and upper range for plotting the spectra
- **PeakListObj** object a PeakList object can be provided to plot peaks
- **cex.legend** numeric text size

```r
overlayPlot(objectList, ..., type = "l", mzRange = c(1, 200), PeakListObj = NULL, cex.legend = 0.5)
```
**parIndicesSearch**

**Value**

graphical output

**Author(s)**

Lorenz Gerber <lorenz.gerber@slu.se>

**Examples**

```r
library(tofsimsData)
data('tofsimsData')
overlayPlot(list(testImage, testSpectra))
```

**Description**

helper function for parallel processing in rawdata import routines

**Usage**

```r
parIndicesSearch(rawVector, mzs, mzsOrder, startOrEnd = "start")
```

**Arguments**

- `rawVector`: unknown
- `mzs`: unknown
- `mzsOrder`: unknown
- `startOrEnd`: character 'start' or 'end'

**Value**

numeric indices of time of flight
**Description**

Class PCA is a virtual class for PCA that will be inherited.

**Details**

Class PCA is a virtual class for PCA that will be inherited.

**Slots**

- `pcaLoadings` matrix that holds the loadings of a principal component like analysis
- `pcaScores` matrix that holds the scores of a principal component like analysis
- `nComp` numeric number of components in the principal component like analysis
- `imageDim` vector x and y values of the image dimension
- `classOfData` character a more detailed description of the analysis type

**Usage**

```r
pcaLoadings(object, comps = c(1, 2))
```

## S4 method for signature 'PCA,missing'

```r
pcaLoadings(object)
```

## S4 method for signature 'PCA,numeric'

```r
pcaLoadings(object, comps = c(1, 2))
```

**Arguments**

- `object` object of class PCA
- `comps` numeric number of components

**Value**

contents of slot pcaLoadings
matrix numeric with loadings
vector or matrix numeric with loadings according `comps`
Examples

```r
library(tofsimsData)
data(tofsimsData)
testImage <- PCAnalysis(testImage, 4)
plot(pcaLoadings(analysis(testImage, 1), comps = c(1, 2)))
```

description

**helper function for MAF calculation**

**Usage**

```
pcaMAF(X, nComp)
```

**Arguments**

- `X` matrix numeric, matrix to calculate PCA from
- `nComp` number of components

**Value**

principal component analysis

---

**PCAnalysis**

**Class** `PCAnalysis`

**Description**

Class `PCAnalysis` contains methods for simple PCA analysis

`PCAnalysis` is a PCA constructor function

**Usage**

```
PCAnalysis(dataObject, nComp, ...)
```

**Arguments**

- `dataObject` object of type `MassImage`
- `nComp` integer number of components
- `...` further args
Details

Class PCAnalysis contains methods for simple PCA analysis.

PCAnalysis constructor function uses call by reference. The new object is put into the analysis slot of the dataObject on which PCA was calculated.

Value

PCAnalysis class object

Author(s)

Lorenz Gerber <lorenz.gerber@slu.se>

Examples

testImage<-MassImage('dummy')
testImage<-PCAnalysis(testImage, 4)
image(analysis(testImage, 1), comp = 1)
## Not run:
library(tofsimsData)
data(tofsimsData)
testImage<-PCAnalysis(testImage, nComp = 4)
image(analysis(testImage, 1), comp = 1)
## End(Not run)

---

pcaScores generic accessor for slot pcaScores

Description

generic accessor for slot pcaScores

PCA accessor pcaScores, pcaScores matrix

PCA accessor pcaScores, pcaScores matrix

Usage

pcaScores(object, comps = c(1, 2))

## S4 method for signature 'PCA,ANY'
pcaScores(object)

## S4 method for signature 'PCA,numeric'
pcaScores(object, comps = c(1, 2))

Arguments

object object of class PCA
comps numeric number of components
peakIDs

Value
contents of slot pcaScores
vector or matrix numeric with scores according comps

Examples
library(tofsimsData)
data(tofsimsData)
testImage<-PCAnalysis(testImage,4)
plot(pcaScores(analysis(testImage,1), comps = c(1,2)))

peakIDs

Description
peakIDs, slot of PeakList class objects

Usage
peakIDs(object)
peakIDs(object) <- value

## S4 method for signature 'PeakList'
peakIDs(object)

## S4 replacement method for signature 'PeakList'
peakIDs(object) <- value

Arguments
object object of class PeakList
value data.frame

Value
content of slot peakIDs

Examples
library(tofsimsData)
data(tofsimsData)
testSpectra<-calibPointNew(testSpectra, mz = 15, value = 15.01551)
testSpectra<-calibPointNew(testSpectra, mz = 181, value = 181.0228)
testSpectra<-recalibrate(testSpectra)
testSpectra<-unitMassPeaks(testSpectra, mzRange = c(1,200), widthAt = c(15, 181),
factor = c(0.4, 0.6), lower = c(14.97, 15.05), upper = c(180.84, 181.43))
peakIDs(testSpectra)[,1:10]
### Description

Class `PeakList` is an extension of TIC class that can hold information about peaks.

Class `PeakList` inherits from the classes `MassAnalysis`, `MassSpectra` and `TIC`.

#### PeakList class constructor

**Usage**

```r
PeakList(analysisName = NULL, instrument = NULL, nz = NULL, calibration = NULL, calibPoints = NULL, mz = NULL, peakIDs = NULL, peakMzs = NULL, ...)```

**Arguments**

- `analysisName` character vector with the import filename
- `instrument` character vector type of instrument used in the experiment
- `nz` matrix numeric containing ion counts, rows are image points, column toftimes/mass to charge ratios
- `calibration` data frame for numerics slope and intercept of the mass calibration
- `calibPoints` data frame for time of flight to mass to charge calibration
- `mz` vector same length as columns in `nz` for mass to charge values
- `peakIDs` matrix integer ID for peaks
- `peakMzs` matrix with mass to charge values for lower, middle and upper peak values
- `...` additional args

**Details**

The `PeakList` class constructor is used to construct a new `PeakList` object. Input are currently all needed variables.

**Value**

object of class `PeakList`

**Slots**

- `peakIDs` matrix integer ID for peaks
- `peakMzs` matrix with mass to charge values for lower, middle and upper peak values
Examples

# The typical way to obtain a PeakList object is by applying some peak picking method to a MassSpectra
# below an example using the 'unitMassPeaks' method
library(tofsimsData)
data(tofsimsData)
testSpectra<-calibPointNew(testSpectra, mz = 15, value = 15.01551)
testSpectra<-calibPointNew(testSpectra, mz = 181, value = 181.0228)
testSpectra<-recalibrate(testSpectra)
testSpectra<-unitMassPeaks(testSpectra, mzRange = c(1,200), widthAt = c(15, 181), factor = c(0.4, 0.6), lower = c(14.97, 15.05), upper = c(180.84, 181.43))
show(testSpectra)

Description

peakMzs, slot of PeakList class objects

Usage

peakMzs(object)

peakMzs(object) <- value

## S4 method for signature 'PeakList'
peakMzs(object)

## S4 replacement method for signature 'PeakList'
peakMzs(object) <- value

Arguments

object object of class PeakList
value data.frame

Value

contents of slot peakMzs
Examples

```r
library(tofsimsData)
data(tofsimsData)

testSpectra<-calibPointNew(testSpectra, mz = 15, value = 15.01551)
testSpectra<-calibPointNew(testSpectra, mz = 181, value = 181.0228)
testSpectra<-recalibrate(testSpectra)
testSpectra<-unitMassPeaks(testSpectra, mzRange = c(1,200), widthAt = c(15, 181),
factor = c(0.4, 0.6), lower = c(14.97, 15.05), upper = c(180.84, 181.43))
peakMzs(testSpectra)[,1:10]
```

peakPick  
**generic method peak.pick**

Description

generic method peak.pick

method peakPick

Usage

```r
peakPick(object, span = 100, ...)
```

## S4 method for signature 'MassSpectra'

```r
peakPick(object, span = 100, ...)
```

Arguments

- `object` object of class MassSpectra
- `span` numeric parameter for local max/min detection
- `...` additional args

Details

Method peakPick for MassSpectra class, works as a constructor for PeakList class. The local
min/max detection implementation is adapted from the CRAN package 'ChemometricsWithR'.

Value

object of class PeakList with updated slots PeakIDs and peakMzs

object of class PeakList

Examples

```r
library(tofsimsData)
data(tofsimsData)

testSpectra <- reduceSpectrumResolution(object = testSpectra, everyN = 4, mode = 'keep')
testSpectra <- smootherSpline(testSpectra, stepsize = 10, spar = 0.3)
testSpectra <- smootherGolay(testSpectra, p = 3, n = 5)
testSpectra <- peakPick(testSpectra, span = 100)
plot(testSpectra, , mzRange=c(38.5,40.5), type = 'l')
```
peaks2Spectra

**Description**

peaks2Spectra allows to transfer the peaks from a PeakList object onto a MassSpectra object. By this, the MassSpectra object is promoted into a PeakList object.

**Usage**

```r
peaks2Spectra(objectPeaks, objectSpectra)
```

**Arguments**

- `objectPeaks`: object of class PeakList
- `objectSpectra`: object of class MassSpectra

**Value**

object of class PeakList

**Examples**

```r
library(tofsimsData)
data(tofsimsData)
testSpectra<-reduceSpectrumResolution(testSpectra,everyN = 4, mode = 'keep')
peakPickSpectra<-testSpectra
peakPickSpectra<-calibPointNew(peakPickSpectra, mz = 15, value = 15.01551)
peakPickSpectra<-calibPointNew(peakPickSpectra, mz = 181, value = 181.0228)
peakPickSpectra<-recalibrate(peakPickSpectra)
peakPickSpectra<-unitMassPeaks(peakPickSpectra, mzRange = c(1,200), widthAt = c(15, 181), factor = c(0.4, 0.6), lower = c(14.97, 15.05), upper = c(180.84, 181.43))
par(mfcol = c(1,2))
plot(testSpectra, mzRange = c(38.5, 40.5), type = 'l')
testSpectra<-peaks2Spectra(peakPickSpectra, testSpectra)
plot(testSpectra, mzRange = c(38.5, 40.5), type = 'l')
```

peakWidths

**Generic method peakWidths**

**Description**

Generic method peakWidths

peakWidths
Usage

peakWidths(object, plot = FALSE)

## S4 method for signature 'PeakList'
peakWidths(object, plot = FALSE)

Arguments

object   PeakList object
plot     boolean should there be graphical output

Details

This method will calculate peak widths (m/z) based on lower and upper widths.

Method to return the peakWidth values of all peaks. On plot=TRUE the width values are plotted against the M/z of the corresponding peak.

Value

vector of peak widths

Examples

library(tofsimsData)
data(tofsimsData)
testPeakList<-PeakList(analysisName = analysisName(testSpectra),
instrument = instrument(testSpectra),
nz = nz(testSpectra),
calibration = calibration(testSpectra),
calibPoints = calibPoints(testSpectra),
mz = mz(testSpectra),
peakIDs = NULL,
peakMzs = NULL)
testPeakList<-addPeaks(testPeakList, mzs=26:31, width=0.4)
testPeakList<-findPeakWidth(testPeakList, p = 3, n = 199,
span = 100, widthExtLower = 2, widthExtUpper = 2)
testPeakList<-peakWidths(testPeakList, plot = FALSE)

plot

Generic method for plot

Description

Generic method for plot

Method defining plot() for the MassSpectra class plot has no generic by default
plot

Usage

plot(x, y, ...)

## S4 method for signature 'MassSpectra,missing'
plot(x, y, ..., mzRange = c(0, 200),
     normalize = FALSE)

## S4 method for signature 'PCA,ANY'
plot(x, ..., comps = c(1, 2), pcType = "pcaLoadings",
     label = FALSE, labelThreshold = 1)

Arguments

x          object of type MassSpectra
y          missing
...         further args
mzRange     vector or length two, indicating the mz range to be plotted
normalize   boolean should the mass spectra be normalized
comps       numeric vector of length two denoting the components to be plotted
pcType      character 'pcaLoadings' or pcaScores'
label       boolean plot label
labelThreshold numeric threshold on which values to plot a label

Details

The output of this method is adapted for plotting mass spectra. Uncalibrated data is plotted as xy plot while uncalibrated data is plotted as barplot. The parameter mzRange allows choosing the plot range directly according to the mz number (when calibrated). The argument lineplot, TRUE by default, allows to switch between line and barplot.

Value

graphical output
plot of mass spectra
scatter loading/score plot

Examples

## plot method for MassSpectra objects
library(tofsimsData)
data(tofsimsData)
plot(testSpectra, mzRange=c(1,300),type='l')
**plot,MassImage,missing-method**

*Method plot() for MassImage*

### Description
Method defining `plot()` for the `MassImage` class plot has no generic by default.

### Usage
```r
## S4 method for signature 'MassImage,missing'
plot(x, y, ..., mzRange = c(0, 200),
     normalize = FALSE)
```

### Arguments
- **x**: object of type `MassImage`
- **y**: missing
- **...**: additional args
- **mzRange**: vector or length two, indicating the mz range to be plotted
- **normalize**: should the mass spectra be normalized

### Details
This method will call `plot` method of `MassSpectra` class.

### Value
scatter plot with loading or scores

---

**plot,PeakList,missing-method**

*Method plot() for MassSpectra*

### Description
Method defining `plot()` for the `MassSpectra` class plot has no generic by default.

### Usage
```r
## S4 method for signature 'PeakList,missing'
plot(x, y, ..., mzRange = c(0, 200),
     plotDeriv = FALSE, plotPeaks = TRUE, plotWidths = TRUE)
```
points

Arguments

- **x**: object of type PeakList
- **y**: missing
- **...**: further args
- **mzRange**: vector or length two, indicating the mz range to be plotted
- **plotDeriv**: boolean plot derivate if available
- **plotPeaks**: boolean plot peaks if available
- **plotWidths**: boolean plot peak widths if available

Details

The output of this method is adapted for plotting mass spectra. Uncalibrated data is plotted as xy plot while uncalibrated data is plotted as barplot. The parameter mzRange allows choosing the plot range directly according to the mz number (when calibrated).

Value

plot spectra with peaks and peak widths

Description

generic method points generic method points

Method defining points() for the MassSpectra class points has no generic by default

Usage

```r
points(x, ...)
```

```r
## S4 method for signature 'MassSpectra'
points(x, y, ..., mzRange = c(0, 200),
       normalize = FALSE)
```

Arguments

- **x**: vector with mz for mass spectra plot
- **...**: additional args
- **y**: vector with ion counts for mass spectra plot
- **mzRange**: vector of length 2, indicating the mz range to be plotted
- **normalize**: boolean should the mass spectra be normalized

Details

This function can be used to visualize several spectra in the same plot.
poissonScaling

Value

- graphical output
- graphic output

Examples

```r
library(tofsimsData)
data("tofsimsData")
plot(testImage, type='l', normalize = TRUE, col = 'blue')
points(testSpectra, type = 'l', normalize = TRUE, col = 'red')
```

poissonScaling generic method for "poissonScaling"

Description

generic method for "poissonScaling"
Possion scaling for data matrices.

Usage

```r
poissonScaling(object, offset = 1, ...)
```

### S4 method for signature 'MassSpectra'
```r
poissonScaling(object, offset = 1, ...)
```

Arguments

- `object` object of class MassSpectra
- `offset` numeric value for poisson scaling
- `...` further args

Details

Possion scaling is proposed as the method of choice for ToF-SIMS data see Keenan and Kotula 2004. This implementation was done according to a description in Multivariate Analysis of SIMS spectra in ToF-SIMS: Materials Analysis by Mass Spectrometry, Vickerman and Briggs 2013 and the eigenvector wiki. The offset is described in the eigenvector wiki.

Value

- object of class MassSpectra with poission scaled mass spectra in slot nz
- object of class MassSpectra

Author(s)

Lorenz Gerber <lorenz.gerber@slu.se>
Examples

```r
## poisson scaling of MassSpectra objects
testImage <- MassImage('dummy')
testImage <- poissonScaling(testImage)

## Not run:
# poission scaling on real data
library(tofsimsData)
data(tofsimsData)
par(mfcol=c(2,2))
plot(testImage,type='l')
image(testImage)
testImage <- poissonScaling(testImage)
plot(testImage,type='l')
image(testImage)

## End(Not run)
```

---

**PrComp-class**

**Class PrComp**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class PrComp is a wrapper for the S3 function prcomp</td>
</tr>
<tr>
<td>PrComp is a PCA constructor function</td>
</tr>
</tbody>
</table>

**Usage**

```
prComp(dataObject, ...)
```

**Arguments**

- `dataObject` object of class MassSpectra
- `...` additional args for prcomp

**Details**

Class PrComp is a wrapper for the S3 function prcomp

PrComp constructor function uses call by reference. The new object is put into the analysis slot of the dataObject on which PCA was calculated.

**Value**

object of class PrComp

**Slots**

- `scale` logical see description of stats::prcomp
- `center` vector see description of stats::prcomp
- `sdev` vector see description of stats::prcomp
Author(s)

Lorenz Gerber <lorenz.gerber@slu.se>

Examples

testImage<-MassImage('dummy')
testImage<-prComp(testImage)
image(analysis(testImage, 1), comp = 1)
## Not run:
library(tofsimsData)
data(tofsimsData)
testImage<-prComp(testImage)
image(analysis(testImage, 1), comp = 1)
## End(Not run)

PrinComp-class

Class PrinComp

Description

Class PrinComp is a wrapper for the S3 function princomp
PrinComp is a PCA constructor function

Usage

prinComp(dataObject, ...)

Arguments

dataObject object of class MassSpectra
... additional args

Details

Class PrinComp is a wrapper for the S3 function princomp
PrinComp constructor function uses call by reference. The new object is put into the analysis slot
of the dataObject on which PCA was calculated.

Value

object of class prinComp

Slots

scale vector see description of stats::princomp
n.obs numeric see description of stats::princomp
call language see description of stats::princomp
center center see description of stats::princomp
sdev vector see description of stats::princomp
**readBIF**

ToF-SIMS BIF/BIF6 file import

Description

Function to read ToF-SIMS data in the form of preprocessed BIF files

Usage

```r
readBIF(analysisName, instrument = c("iontof", "ulvacphi"),
        mode = c("spectra", "image"))
```

Arguments

- `analysisName` : filename of BIF/BIF6 file to read
- `instrument` : character, ‘iontof’ or ‘ulvacphi’
- `mode` : ’spectra’ or ’image’

Details

This function imports BIF files from IONTOF Surface Lab or ULVAC-PHI’s WinCadence. This function reads the data sequential directly from the binary stream. Therefore it’s rather slow, but uses less memory than the `readBIFParallel` function.

Value

object of type MassImage or MassSpectra

Author(s)

Lorenz Gerber
### recalibrate

**Generic method recalibrate**

**Description**

Generic method recalibrate

**Usage**

```r
recalibrate(object)
```

```r
# S4 method for signature 'MassSpectra'
recalibrate(object)
```

**Arguments**

- `object`: object of class `MassSpectra`

**Value**

- object of class `MassSpectra`, recalibrated using the data from slots `calibPoints`
- object of class `MassSpectra`, recalibrated mass values

**Examples**

```r
library(tofsimsData)
data(tofsimsData)
testSpectra <- calibPointNew(testSpectra, mz = 15, value = 15.01551)
testSpectra <- calibPointNew(testSpectra, mz = 181, value = 181.0228)
calibPoints(testSpectra)
par(mfcol=c(1,2))
plot(testSpectra, mzRange=c(38.5,40.5), type="l")
testSpectra <- recalibrate(testSpectra)
plot(testSpectra, mzRange=c(38.5,40.5), type="l")
```

### reduceSpectrumResolution

**generic method reduceSpectrumResolution**

**Description**

generic method reduceSpectrumResolution

**Usage**

```r
reduceSpectrumResolution(object, everyN = 2, mode = "remove")
```

```r
# S4 method for signature 'MassSpectra'
reduceSpectrumResolution(object, everyN = 2, mode = "remove")
```
removePeaks

Arguments

- **object**: object of class MassSpectra
- **everyN**: numeric act on every nth spectra point
- **mode**: character 'remove' or 'keep'

Details

The method reduceSpectrumResolution for MassSpectra is used sometimes for performance reasons.

Value

- object of class MassSpectra with reduced spectral resolution
- object of class MassSpectra

Examples

```r
library(tofsimsData)
data(tofsimsData)
par(mfcol=c(1,2))
plot(testSpectra, mzRange = c(40,50), type='l')
testSpectra <- reduceSpectrumResolution(object = testSpectra, everyN = 2, mode = 'remove')
plot(testSpectra, mzRange = c(40,50), type='l')
```

---

removePeaks

generic method removePeaks

Description

generic method removePeaks

removePeaks for PeakList Class allows removing peaks below a certain threshold of ioncounts. The threshold is not calculated as area, but just from the peak height (ion count at peak center)

removePeaks for PeakList Class allows removing peaks manually

Usage

```r
removePeaks(object, mzs, operator, limit, nLocator, ...)
```

## S4 method for signature 'PeakList,missing,missing,numerics,missing'
```r
removePeaks(object, mzs, operator, limit, nLocator, ...)
```

## S4 method for signature 'PeakList,missing,missing,missing,numerics'
```r
removePeaks(object, mzs, operator, limit, nLocator, ...)
```

## S4 method for signature 'PeakList,numerics,missing,missing,missing'
```r
removePeaks(object, mzs, operator, limit, nLocator, ...)
```
# S4 method for signature 'PeakList,missing,character,numeric,missing'
removePeaks(object, mzs, 
  operator, limit, nLocator, ...)

Arguments
  object       object of class PeakList
  mzs          M/z's of peaks to be removed
  operator     Accept ">", "<", "==", "<=" or ">=" or "!="
  limit        numeric limit for peaks to be removed
  nLocator     numeric how many peaks to remove with visual selection
  ...          additional args

Value
  object of class PeakList with removed/updated peaks

Examples
  library(tofsimsData)
  data(tofsimsData)
  testPeakList<-PeakList(analysisName = analysisName(testSpectra),
    instrument = instrument(testSpectra),
    nz = nz(testSpectra),
    calibration = calibration(testSpectra),
    calibPoints = calibPoints(testSpectra),
    mz = mz(testSpectra),
    peakIDs = NULL,
    peakMzs = NULL)
  par(mfcol=c(1,2))
  testPeakList<-addPeaks(testPeakList, mzs = 26:31, width=0.4)
  plot(testPeakList, mzRange = c(25,32), type = 'l')
  testPeakList<-removePeaks(testPeakList, mzs = 27)
  plot(testPeakList, mzRange = c(25,32), type = 'l')

resids               generic accessor method for resids

Description
  generic accessor method for resids

Usage
  resids(object)

Arguments
  object       object of class MCR

Value
  content of slot resids
Description
MCR accessor resids,

Usage
## S4 method for signature 'MCR'
resids(object)

Arguments
object object of class MCR

Value
resids from object

RSS  generic accessor for RSS

Description
generic accessor for RSS

Usage
RSS(object)

Arguments
object object of class MCR

Value
content of slot RSS
**Description**

MCR accessor RSS.

**Usage**

```r
## S4 method for signature 'MCR'
RSS(object)
```

**Arguments**

- `object`: object of type MCR

**Value**

RSS from object

---

**scale**

**generic for scale**

**Description**

generic for scale

scale autoscaling method for MassSpectra object. Scaling is along the mass channels. Therefore more than one spectra is needed for scaling.

**Usage**

```r
scale(x, center = TRUE, scale = TRUE)
```

**Arguments**

- `x`: object object of class MassSpectra
- `center`: boolean should data be centered
- `scale`: boolean should data be scaled

**Value**

- object of class MassSpectra with scaled mass spectra
- object of class MassSpectra
Examples

```r
## autoscaling of dummy image data
testImage <- MassImage('dummy')
par(mfcol=c(2,2))
plot(testImage, type='l')
image(testImage)
testImage <- scale(testImage)
plot(testImage, type='l')
image(testImage)
## Not run:
## autoscaling of real spectral data
library(tofsimsData)
data(tofsimsData)
par(mfcol=c(2,2))
plot(testImage, type='l')
image(testImage)
testImage <- scale(testImage)
plot(testImage, type='l')
image(testImage)
## End(Not run)
```

Description

method definition `show()` on `MassImage` show has a generic by default

Usage

```r
## S4 method for signature 'MassImage'
show(object)
```

Arguments

- `object`: object of class `MassImage`

Value

data.frame character

Description

method defining `show()` for the `MassSpectra` class show has a generic by default
Usage

## S4 method for signature 'MassSpectra'
show(object)

Arguments

object object of class MassSpectra

Value
data.frame character

Description

method defining show() for the MassSpectra class show has a generic by default

Usage

## S4 method for signature 'PeakList'
show(object)

Arguments

object object of class PeakList

Value
data.frame character

smootherGolay

generic method smootherGolay

Description

generic method smootherGolay

Method smootherGolay for MassSpectra class

Usage

smootherGolay(object, p = 3, n = 5, ...)

## S4 method for signature 'MassSpectra'
smootherGolay(object, p = 3, n = 5, ...)
**smootherSpline**

**Arguments**

- `object`: object of class `MassSpectra`
- `p`: numeric parameter for savitzky-golay filter
- `n`: numeric parameter for savitzky-golay filter
- `...`: additional args

**Value**

- object of class `MassSpectra` with updated mass spectra
- object of class `MassSpectra` with smoothed TIC

**Examples**

```r
library(tofsimsData)
data(tofsimsData)
testSpectraSmooth <- smootherGolay(testSpectra, p = 3, n = 9)
overlayPlot(list(testSpectra, testSpectraSmooth), mzRange = c(38.5, 40.5), type = 'l')
```

**Description**

generic smootherSpline

method smootherSpline for TIC

**Usage**

```r
smootherSpline(object, stepsize = 5, spar = 0.3, ...)
```

## S4 method for signature 'MassSpectra'

```r
smootherSpline(object, stepsize = 5, spar = 0.3, ...)
```

**Arguments**

- `object`: MassSpectra
- `stepsize`: numeric arg for spline smoother
- `spar`: numeric arg for spline smoother
- `...`: additional args

**Value**

- object of class `MassSpectra` with updated mass spectra
- object of class `MassSpectra`
smoothScatter

Examples

```r
library(tofsimsData)
data(tofsimsData)

testSpectraSmooth <- smootherSpline(testSpectra)
overlayPlot(list(testSpectra, testSpectraSmooth), mzRange = c(38.5, 40.5), type = 'l')
```

smoothScatter

**generic for smoothScatter**

**Description**

generic for smoothScatter

smoothScatter method for PCA class

**Usage**

```r
smoothScatter(x, y = NULL, nbin = 128, bandwidth,
colramp = colorRampPalette(c("white", blues9)), nrpoints = 100,
ret.selection = FALSE, pch = ".", cex = 1, col = "black",
transformation = function(x) x^0.25, postPlotHook = box, xlab = NULL,
ylab = NULL, xlim, ylim, xaxs = par("xaxs"), yaxs = par("yaxs"), ...)
```

```r
## S4 method for signature 'PCA'
smoothScatter(x, y = NULL, nbin = 128, bandwidth,
colramp = colorRampPalette(c("white", blues9)), nrpoints = 100,
ret.selection = FALSE, pch = ".", cex = 1, col = "black",
transformation = function(x) x^0.25, postPlotHook = box, xlab = NULL,
ylab = NULL, xlim, ylim, xaxs = par("xaxs"), yaxs = par("yaxs"), ...
comps = c(1, 2), pcType = "pcaScores", label = FALSE,
labelThreshold = 1)
```

**Arguments**

- `x` object of class PCA
- `y` numeric usually NULL
- `nbin` numeric
- `bandwidth` numeric vector length 1 or 2
- `colramp` numeric
- `nrpoints` numeric
- `ret.selection` logical
- `pch` character
- `cex` numeric
- `col` character
- `transformation` function
- `postPlotHook` box
- `xlab` NULL
- `ylab` NULL
SNR

Signal-to-Noise Ratio (SNR)

Description

SNR function for MNF to calculate Signal to Noise Ratio

Usage

`SNR(stat, x, y)`

Arguments

- `stat`: unknown
- `x`: unknown
- `y`: unknown

Details

function from mzimage to calculate signal-to-noise ratio function

Value

matrix numeric with signal-to-noise ratios
subset

**Generic method for subset**

**Description**
Generic method for subset
Subset method for objects of class MassImage

**Usage**

```r
subset(x, ...)  # S4 method for signature 'MassImage'
subset(x, ..., xyUpperLeft = NULL, xyLowerRight = NULL)
```

**Arguments**

- `x`: object of class MassImage
- `...`: additional args
- `xyUpperLeft`: vector of length two with x and y for the upper left subset corner
- `xyLowerRight`: vector of length two with x and y for the lower right subset corner

**Value**
object of class MassImage a subest of the in-object
object of class MassImage

**Examples**

```r
library(tofsimsData)
data(tofsimsData)
subsetTestImage<-subset(testImage, xyUpperLeft = c(1,1), xyLowerRight = c(50,50))
image(subsetTestImage)
```

unitMassPeaks

**Generic method for unitMassPeaks**

**Description**
Generic method for unitMassPeaks

**Usage**

```r
unitMassPeaks(object, mzRange, widthAt, factor, upper = NULL, lower = NULL, ...)
# S4 method for signature 'MassSpectra,numeric,numeric'
unitMassPeaks(object, mzRange, widthAt, factor, upper = NULL, lower = NULL, ...)
```
validMassImageObject

Arguments

object  object of class MassSpectra

mzRange  vector numeric with lower and upper mass range limit for which to set unit mass peaks

widthAt  vector numeric two mass values at which to sample for peak width

factor  vector numeric two values summing up to 1 for setting asymmetric peak width limits

upper  vector numeric upper peak width limits

lower  vector numeric lower peak width limits

...  additional args

Value

object of class PeakList with unit mass peaks

Examples

library(tofsimsData)
data(tofsimsData)
testSpectra <- calibPointNew(testSpectra, mz = 15, value = 15.01551)
testSpectra <- calibPointNew(testSpectra, mz = 181, value = 181.0228)
testSpectra <- recalibrate(testSpectra)
testSpectra <- unitMassPeaks(testSpectra, mzRange = c(1,200), widthAt = c(15, 181),
factor = c(0.4, 0.6), lower = c(14.97, 15.05), upper = c(180.84, 181.43))
plot(testSpectra, mzRange = c(1,200), type = 'l')

validMassImageObject  Validation method function for class MassImage objects

Description

Validation method function for class MassImage objects

Usage

validMassImageObject(object)

Arguments

object  object of class MassImage

Value

boolean class validity test
validMassSpectraObject

Validation method function for class MassImage objects

Description

Validation method function for class MassImage objects

Usage

validMassSpectraObject(object)

Arguments

object object of class MassSpectra

Value

boolean class validity test

validPCAObject

Validation method function for class PCA objects

Description

Validation method function for class PCA objects

Usage

validPCAObject(object)

Arguments

object object of class PCA

Value

boolean class validity test
validPeakListObject  

Validation method function for class PeakList objects

Description

Validation method function for class PeakList objects

Usage

validPeakListObject(object)

Arguments

object object of class PeakList

Value

boolean class validity test

xdim  
generic accessor method for "xdim"

Description

generic accessor method for "xdim"

Usage

xdim(object)

Arguments

object object of class MassImage

Value

numeric value x dimension of mass image
Description

Getter, method definition "xdim" on "MassImage"

Usage

## S4 method for signature 'MassImage'
xdim(object)

Arguments

object : objet of class MassImage

Value

numeric x dimension of slot xy

Description

method xdim() for PCA class object

Usage

## S4 method for signature 'PCA'
xdim(object)

Arguments

object : object of class PCA

Value

numeric x dimension of image
xdim<-  

generic setter method for "xdim"

Description

generic setter method for "xdim"

Usage

xdim(object) <- value

Arguments

| object    | object of class MassImage |
| value     | numeric x dimension of image |

Value

object of class MassImage with updated x dimension

xy

xy, slot of MassImage class objects

Description

xy, slot of MassImage class objects

Usage

xy(object)

xy(object) <- value

## S4 method for signature 'MassImage'

xy(object)

## S4 replacement method for signature 'MassImage'

xy(object) <- value

Arguments

| object | object of class MassImage |
| value  | vector numeric two values for x and y dimension of image |

Value

vector numeric with xy dimensions of image
Examples

```r
library(tofsimsData)
data(tofsimsData)
xy(testImage)
```

---

**xySpec**

Generic method `xySpec`

**Description**

Selection of Spectra

Method `xySpec` extracts the mass spectra of position x/y and puts them in a MassSpectra class object.

**Usage**

```r
xySpec(object, x = NULL, y = NULL)
## S4 method for signature 'MassImage'
xySpec(object, x = NULL, y = NULL)
```

**Arguments**

- `object`: object of class MassImage
- `x`: numeric x coordinate from where to sample a mass spectra
- `y`: numeric y coordinate from where to sample a mass spectra

**Details**

Selection of mass spectra by vectors of equal length for x and y.

**Value**

object of class MassSpectra with selected mass spectra

**Author(s)**

Lorenz Gerber <lorenz.gerber@slu.se>

**Examples**

```r
library(tofsimsData)
data(tofsimsData)
spectra100100<-xySpec(testImage, 100,100)
plot(spectra100100, type = 'l')
```
ydim

generic accessor method for "ydim"

Description

generic accessor method for "ydim"

Usage

ydim(object)

Arguments

object object of class MassImage

Value

numeric integer, y dimension of image

ydim,MassImage-method Getter, method definition "ydim" on "MassImage"

Description

Getter, method definition "ydim" on "MassImage"

Usage

## S4 method for signature 'MassImage'
ydim(object)

Arguments

object object of class MassImage

Value

numeric y dimension of slot xy
Description

method ydim() for PCA class object

Usage

## S4 method for signature 'PCA'
ydim(object)

Arguments

object object of class PCA

Value

numeric y dimension of image

dim<- generic setter method for "ydim"

Description

generic setter method for "ydim"

Usage

ydim(object) <- value

Arguments

object object of class MassImage
value numeric y dimension of image

Value

updated object of type MassImage
**zdim**  

**generic accessor method for "zdim"**

**Description**

generic accessor method for "zdim"

**Usage**

```
zdim(object)
```

**Arguments**

- `object` object of class MassImage

**Value**

numeric, number of mass channels / peaks

---

**zdim,MassSpectra-method**

*method definition 'zdim' on 'MassSpectra'*

**Description**

method definition 'zdim' on 'MassSpectra'

**Usage**

```  
## S4 method for signature 'MassSpectra'
zdım(object)
```  

**Arguments**

- `object` object of class MassSpectra

**Value**

numeric value
Index

°Topic package
  tofsims-package, 4
addFixedWidth, 4
addFixedWidth, PeakList, numeric, numeric-method (addFixedWidth), 4
addPeaks, 5
addPeaks, PeakList, missing, numeric-method (addPeaks), 5
addPeaks, PeakList, numeric, numeric-method (addPeaks), 5
analysis, 6, 7, 12, 13, 27, 44
analysis, MassSpectra, missing-method (analysis), 6
analysis, MassSpectra, numeric-method (analysis), 6
analysis<-(analysis), 6
analysis<-, MassSpectra-method (analysis), 6
analysisName, 6, 7, 12, 13, 27, 44
analysisName, MassSpectra-method (analysisName), 7
analysisName<-(analysisName), 7
analysisName<-, MassSpectra-method (analysisName), 7

dim, MassSpectra-method, 20
dim, MassImage-method, 19
eigenDecompose, 20
extract.header.data, 21
findClosestMatch, 21
findPeakWidth, 22
findPeakWidth, PeakList-method (findPeakWidth), 22
getoToF, 23
getoToF, MassSpectra-method (getoToF), 23

calibration, 6, 7, 12, 13, 27, 44
calibration, MassSpectra-method (calibration), 13
calibration<-(calibration), 13
changePeakWidth, 14
changePeakWidth, PeakList, missing, missing, missing-method (changePeakWidth), 14
changePeakWidth, PeakList, numeric, numeric, numeric-method (changePeakWidth), 14
check.extension, 15
computeMNF, 15
calculateNoise, 16
coordToPixel, 17
coordToPixel, MassImage, numeric-method, 17
covDiffCalc, 18
readRawPhi, 18
cTable (cTable), 19
cTable, 19
dim, MassImage-method, 19
dim, MassSpectra-method, 20

eigenDecompose, 20
extract.header.data, 21
findClosestMatch, 21
findPeakWidth, 22
findPeakWidth, PeakList-method (findPeakWidth), 22
getoToF, 23
getoToF, MassSpectra-method (getoToF), 23

calibration, 6, 7, 12, 13, 27, 44
calibration, MassSpectra-method (calibration), 13
calibration<-(calibration), 13
changePeakWidth, 14
changePeakWidth, PeakList, missing, missing, missing-method (changePeakWidth), 14
changePeakWidth, PeakList, numeric, numeric, numeric-method (changePeakWidth), 14
check.extension, 15
computeMNF, 15
calculateNoise, 16
coordToPixel, 17
coordToPixel, MassImage, numeric-method, 17
covDiffCalc, 18
readRawPhi, 18
cTable (cTable), 19
cTable, 19
dim, MassImage-method, 19
dim, MassSpectra-method, 20

eigenDecompose, 20
extract.header.data, 21
findClosestMatch, 21
findPeakWidth, 22
findPeakWidth, PeakList-method (findPeakWidth), 22
getoToF, 23
getoToF, MassSpectra-method (getoToF), 23

calibration, 6, 7, 12, 13, 27, 44
calibration, MassSpectra-method (calibration), 13
calibration<-(calibration), 13
changePeakWidth, 14
changePeakWidth, PeakList, missing, missing, missing-method (changePeakWidth), 14
changePeakWidth, PeakList, numeric, numeric, numeric-method (changePeakWidth), 14
check.extension, 15
computeMNF, 15
calculateNoise, 16
coordToPixel, 17
coordToPixel, MassImage, numeric-method, 17
covDiffCalc, 18
readRawPhi, 18
cTable (cTable), 19
cTable, 19
dim, MassImage-method, 19
dim, MassSpectra-method, 20

eigenDecompose, 20
extract.header.data, 21
findClosestMatch, 21
findPeakWidth, 22
findPeakWidth, PeakList-method (findPeakWidth), 22
getoToF, 23
getoToF, MassSpectra-method (getoToF), 23

calibration, 6, 7, 12, 13, 27, 44
calibration, MassSpectra-method (calibration), 13
calibration<-(calibration), 13
changePeakWidth, 14
changePeakWidth, PeakList, missing, missing, missing-method (changePeakWidth), 14
changePeakWidth, PeakList, numeric, numeric, numeric-method (changePeakWidth), 14
check.extension, 15
computeMNF, 15
calculateNoise, 16
coordToPixel, 17
coordToPixel, MassImage, numeric-method, 17
covDiffCalc, 18
readRawPhi, 18
cTable (cTable), 19
cTable, 19
dim, MassImage-method, 19
dim, MassSpectra-method, 20

eigenDecompose, 20
extract.header.data, 21
findClosestMatch, 21
findPeakWidth, 22
findPeakWidth, PeakList-method (findPeakWidth), 22
getoToF, 23
getoToF, MassSpectra-method (getoToF), 23

calibration, 6, 7, 12, 13, 27, 44
calibration, MassSpectra-method (calibration), 13
calibration<-(calibration), 13
changePeakWidth, 14
changePeakWidth, PeakList, missing, missing, missing-method (changePeakWidth), 14
changePeakWidth, PeakList, numeric, numeric, numeric-method (changePeakWidth), 14
check.extension, 15
computeMNF, 15
calculateNoise, 16
coordToPixel, 17
coordToPixel, MassImage, numeric-method, 17
covDiffCalc, 18
readRawPhi, 18
cTable (cTable), 19
cTable, 19
dim, MassImage-method, 19
dim, MassSpectra-method, 20

eigenDecompose, 20
extract.header.data, 21
findClosestMatch, 21
findPeakWidth, 22
findPeakWidth, PeakList-method (findPeakWidth), 22
getoToF, 23
getoToF, MassSpectra-method (getoToF), 23

calibration, 6, 7, 12, 13, 27, 44
calibration, MassSpectra-method (calibration), 13
calibration<-(calibration), 13
changePeakWidth, 14
changePeakWidth, PeakList, missing, missing, missing-method (changePeakWidth), 14
changePeakWidth, PeakList, numeric, numeric, numeric-method (changePeakWidth), 14
check.extension, 15
computeMNF, 15
calculateNoise, 16
coordToPixel, 17
coordToPixel, MassImage, numeric-method, 17
covDiffCalc, 18
readRawPhi, 18
cTable (cTable), 19
cTable, 19
dim, MassImage-method, 19
dim, MassSpectra-method, 20

eigenDecompose, 20
extract.header.data, 21
findClosestMatch, 21
findPeakWidth, 22
findPeakWidth, PeakList-method (findPeakWidth), 22
getoToF, 23
getoToF, MassSpectra-method (getoToF), 23

82
INDEX

instrument, 6, 7, 12, 13, 26, 44
instrument, MassSpectra-method (instrument), 26
instrument<- (instrument), 26
instrument<-, MassSpectra-method (instrument), 26
iters, 27
iters, MCR-method, 28
itzipName, 28
itzipName<-, 29
LapackGenEigen, 29
legend.col, 30
look.for.itzip.property, 30
MAF, 31
makeTIC, 31
makeTIC, MassSpectra-method, 32
manualSelectPeaks, 32
MassImage, 33
MassSpectra, 6, 7, 12, 13, 27, 34, 44
MCR (MCR-class), 35
MCR-class, 35
MNF, 36
mz, 6, 7, 12, 13, 27, 44
mz, MassSpectra-method, 37
mz<-, MassSpectra-method (mz, MassSpectra-method), 37
nComp, 38
nComp, PCA-method (nComp), 38
ndim, 39
ndim, MassSpectra-method, 39
nearestNeighbourMean, 40
nnMean, 40
nnMNF, 41
noPlottingData, 42
noPlottingData, PCA-method, 42
nPeaks, 43
nPeaks, PeakList-method (nPeaks), 43
nz, 6, 7, 12, 13, 27, 43
nz, MassSpectra, missing-method (nz), 43
nz, MassSpectra, numeric-method (nz), 43
nz<-, (nz), 43
nz<-, MassSpectra-method (nz), 43
opaMCR (MCR-class), 35
overlayPlot, 44
overlayPlot, list-method (overlayPlot), 44
parIndicesSearch, 45
PCA (PCA-class), 46
PCA-class, 46
calaLoadings, 46
calaLoadings, PCA, missing-method (calaLoadings), 46
calaLoadings, PCA, numeric-method (calaLoadings), 46
calaMAF, 47
PCAAnalysis, 47
calaScores, 48
calaScores, PCA, ANY-method (calaScores), 48
calaScores, PCA, numeric-method (calaScores), 48
peakIDs, 49
peakIDs, PeakList-method (peakIDs), 49
peakIDs<-, (peakIDs), 49
peakIDs<-, PeakList-method (peakIDs), 49
PeakList, 50
peakMzs, 51
peakMzs, PeakList-method (peakMzs), 51
peakMzs<-, (peakMzs), 51
peakMzs<-, PeakList-method (peakMzs), 51
peakPick, 52
peakPick, MassSpectra-method (peakPick), 52
peaks2Spectra, 53
peaks2Spectra, PeakList, MassSpectra-method (peaks2Spectra), 53
peakWidths, 53
peakWidths, PeakList-method (peakWidths), 53
plot, 54
plot, MassImage, missing-method, 56
plot, MassSpectra, missing-method (plot), 54
plot, PCA, ANY-method (plot), 54
plot, PeakList, missing-method, 56
points, 57
points, MassSpectra-method (points), 57
poissonScaling, 58
poissonScaling, MassSpectra-method (poissonScaling), 58
PrComp (PrComp-class), 59
prComp (PrComp-class), 59
PrComp-class, 59
PrinComp (PrinComp-class), 60
prinComp (PrinComp-class), 60
PrinComp-class, 60
readBIF, 61
recalibrate, 62
recalibrate, MassSpectra-method (recalibrate), 62
reduceSpectrumResolution, 62
reduceSpectrumResolution, MassSpectra-method (reduceSpectrumResolution), 62
removePeaks, 63
removePeaks, PeakList, missing, character, numeric, missing-method (removePeaks), 63
removePeaks, PeakList, missing, missing, missing-method (removePeaks), 63
removePeaks, PeakList, missing, missing, numeric, missing-method (removePeaks), 63
removePeaks, PeakList, numeric, missing, missing, missing-method (removePeaks), 63
resids, 64
resids, MCR-method, 65
RSS, 65
RSS, MCR-method, 66
scale, 66
scale, MassSpectra-method (scale), 66
show, MassImage-method, 67
show, MassSpectra-method, 67
show, PeakList-method, 68
smootherGolay, 68
smootherGolay, MassSpectra-method (smootherGolay), 68
smootherSpline, 69
smootherSpline, MassSpectra-method (smootherSpline), 69
smoothScatter, 70
smoothScatter, PCA-method (smoothScatter), 70
SNR, 71
subset, 72
subset, MassImage-method (subset), 72
tofsims-package, 4
unitMassPeaks, 72
unitMassPeaks, MassSpectra, numeric, numeric-method (unitMassPeaks), 72
validMassImageObject, 73
validMassSpectraObject, 74
validPCAObject, 74
validPeakListObject, 75
xdim, 75
xdim, MassImage-method, 76
xdim, PCA-method, 76
xdim <-, 77
xy, 77
xy, MassImage-method (xy), 77
xy <- (xy), 77
xy <-, MassImage-method (xy), 77