Package 'msPurity'

October 18, 2017

Type Package

Title Automated Evaluation of Precursor Ion Purity for Mass Spectrometry Based Fragmentation in Metabolomics

Version 1.2.0

Date 2017-01-10

Author Thomas N. Lawson, Ralf Weber, Martin Jones, Mark Viant, Warwick Dunn

Maintainer Thomas N. Lawson <thomas.nigel.lawson@gmail.com>

Description

Assess the contribution of the targeted precursor in fragmentation acquired or anticipated isolation windows using a metric called ``precursor purity''. Also provides simple processing steps (averaging, filtering,
blank subtraction, etc) for DI-MS data.
Works for both LC-MS(/MS) and DI-MS(/MS) data.

License GPL (>= 2)

LazyData TRUE

Depends Rcpp

Imports plyr, foreach, parallel, doSNOW, stringr, mzR, reshape2, fastcluster, ggplot2, sapa

Suggests testthat, xcms, BiocStyle, knitr, rmarkdown, msPurityData

VignetteBuilder knitr

RoxygenNote 5.0.1

biocViews MassSpectrometry, Metabolomics, Software

Collate 'all-generics.R' 'iw-norm.R' 'pcalc.R' 'purityA-class.R' 'purityA-constructor.R' 'purityA-frag4feature.R' 'purityA-validate.R' 'purityD-class.R' 'purityD-constructor.R' 'purityD-av-spectra.R' 'purityD-dims-purity.R' 'purityD-fileList.R' 'purityD-filterp.R' 'purityD-subtract.R' 'purityD-writeOut.R' 'purityX-class.R' 'purityX-constructor.R' 'spectral-complexity.R' 'splinepurity.R'

NeedsCompilation no

R topics documented:

`assessPuritySingle` *Assess the purity of a single LC-MS/MS or DI-MS/MS file*

Description

Given a filepath to an mzML file the precursor purity for any MS/MS scans will be outputed into a dataframe

Usage

```
assessPuritySingle(filepth, fileid = NA, mostIntense = FALSE,
  nearest = TRUE, offsets = NA, cores = 1, plotP = FALSE,
  plotdir = NULL, interpol = "linear", iwNorm = FALSE, iwNormFun = NULL,
  ilim = 0, mzRback = "pwiz", isotopes = TRUE, im = NULL)
```

Arguments

`filepth`	character = mzML file path for MS/MS spectra
`fileid`	numeric = adds a fileid column (primarily for internal use for msPurity)
`mostIntense`	boolean = True if the most intense peak is used for calculation. False if the centered peak is used
`nearest`	boolean = True if the peak selected is as the nearest MS1 scan. If False then the preceding scan is used
`offsets`	vector = overide the isolation offsets found in the mzML filee.g. c(0.5, 0.5)
`cores`	numeric = number of cores to use

plotP	boolean = if TRUE a plot of the purity is to be saved
plotdir	vector = if plotP is TRUE plots will be saved to this directory
interpol	character = type of interolation to be performed "linear", "spline" or "none"
iwNorm	boolean = if TRUE then the intensity of the isolation window will be normalised based on the iwNormFun function
iwNormFun	function = A function to normalise the isolation window intensity. The default function is very generalised and just accounts for edge effects
ilim	numeric = All peaks less than this percentage of the target peak will be removed from the purity calculation, default is 5% (0.05)
mzRback	character = backend to use for mzR parsing
isotopes	boolean = TRUE if isotopes are to be removed
im	matrix = Isotope matrix, default removes C13 isotopes (single, double and triple bonds)

Value

a dataframe of the purity score of the ms/ms spectra

See Also

purityA

Examples

```
filepth <- system.file("extdata", "lcms", "mzML", "LCMSMS_1.mzML", package="msPurityData")

puritydf <- assessPuritySingle(filepth)
```

---

averageSpectra,purityD-method

Using purityD object, calculates to average mz, intensity and signal-to-noise of multiple scans from multiple MS datafiles (mzML or .csv)

Description

Uses a purityD object with references to multiple MS files. For each file: Averages multiple scans together, see averageSpectraSingle for more information

Usage

```
## S4 method for signature 'purityD'
averageSpectra(Object, rtscn = "all", scanRange = NA,
  timeRange = NA, clustType = "hc", ppm = 1.5, snthr = 3,
  av = "median", missingV = "zero", minfrac = 0.6667, normTIC = FALSE,
  snMeth = "median")
```

Arguments

Object	object = purityD object
rtscn	character = Whether it is scans or retention time to be filtered. Use "all" if all scans to be used. ['rt', 'scns', 'all']
scanRange	vector = Scan range (if rtscn='scns') e.g. c(40, 69)
timeRange	vector = Time range (if rtscn='rt) e.g. c(10.3, 400.8) (only if using mzML file)
clustType	character = Type of clustering used either Hierarchical or just simple 1dgrouping ['hc', 'simple'], default 'hc'
ppm	numeric = the ppm error to cluster mz together default 1.5
snthr	numeric = Signal to noise ratio threshold, default 0
av	character = What type of averaging to do between peaks
missingV	character = What to do with missing values (zero or ignore)
minfrac	numeric = Min fraction of scans with a grouped peak to be an accepted averaged peak
normTIC	boolean = If TRUE then RSD calculation will use the normalised intensity (intensity divided by TIC) if FALSE will use standard intensity
snMeth	character = Type of snMethod to use

Value

purityD object with averaged spectra

See Also

averageSpectraSingle

Examples

```
datapth <- system.file("extdata", "dims", "mzML", package="msPurityData")
inDF <- Getfiles(datapth, pattern=".mzML", check = FALSE, cStrt = FALSE)
ppDIMS <- purityD(fileList=inDF, cores=1, mzML=TRUE)
ppDIMS <- averageSpectra(ppDIMS)
```

---

averageSpectraSingle *Calculates to average mz, intensity and signal-to-noise of multiple scans from 1 MS datafile (mzML or .csv)*

Description

Averages multiple scans of mass spectrometry data together. Each scan consisting of a minimum of intensity and mz values.

Works for either mzML or a .csv file consisting of mz, i, scanid, (optional: noise, backgroun, snr)

Signal-to-noise (SNR) can be calculated a number of ways. Default is to calculate the SN for every scan as the "Intensity of peak / the median intensity of the scan".

Alternatively if using a .CSV file a precalculated snr can be on of the columns and this can be used.

The function works for LC-MS or DI-MS datasets.

Usage

```
averageSpectraSingle(filePth, rtscn = "all", scanRange = NA,
  timeRange = NA, clustType = "hc", ppm = 1.5, snthr = 3, cores = 1,
  av = "median", missingV = "ignore", minfrac = 0.6667,
  snMeth = "median", MSFileReader = FALSE, normTIC = FALSE,
  mzRback = "pwiz")
```

Arguments

filePth	character = Path of the file to be processed
rtscn	character = Whether it is scans or retention time to be filtered. Use "all" if all scans to be used. ['rt', 'scns', 'all']
scanRange	vector = Scan range (if rtscn='scns') e.g. c(40, 69)
timeRange	vector = Time range (if rtscn='rt) e.g. c(10.3, 400.8) (only if using mzML file)
clustType	character = Type of clustering used either Hierarchical or just simple 1dgrouping ['hc', 'simple'], default 'hc'
ppm	numeric = the ppm error to cluster mz together default 1.5
snthr	numeric = Signal to noise ratio threshold, default 0
cores	numeric = Number of cores used to perform Hierarchical clustering WARNING: memory intensive, default 2
av	character = What type of averaging to do between peaks
missingV	character = What to do with missing values (zero or ignore)
minfrac	numeric = Min fraction of scans with a grouped peak to be an accepted averaged peak
snMeth	character = Type of snMethod to use
MSFileReader	boolean = For thermo files a the MSFileReader API can extract peaklist. This can consist of an .csv file with the following columns c('mz', 'i', 'scanid', 'snr')
normTIC	boolean = If TRUE then RSD calculation will use the normalised intensity (intensity divided by TIC) if FALSE will use standard intensity
mzRback	character = backend to use for mzR parsing

Value

dataframe of the median mz, intensity, signal-to-noise ratio.

Examples

```
mzmlPth <- system.file("extdata", "dims", "mzML", "B02_Daph_TEST_pos.mzML", package="msPurityData")
avP <- averageSpectraSingle(mzmlPth)
```

`dimsPredictPurity,purityD-method`
Using purityD object, assess anticipated purity from a DI-MS run

Description

Assess the precursor purity of anticpated MS/MS spectra. i.e. it 'predicts' the precursor purity of the DI-MS peaks for a future MS/MS run.

Usage

```
## S4 method for signature 'purityD'
dimsPredictPurity(Object, ppm = 1.5, minOffset = 0.5,
  maxOffset = 0.5, iwNorm = FALSE, iwNormFun = NULL, ilim = 0.05,
  sampleOnly = FALSE, isotopes = TRUE, im = NULL)
```

Arguments

`Object`	object = purityD object
`ppm`	numeric = tolerance for target mz value in each scan
`minOffset`	numeric = isolation window minimum offset
`maxOffset`	numeric = isolation window maximum offset
`iwNorm`	boolean = if TRUE then the intensity of the isolation window will be normalised based on the iwNormFun function
`iwNormFun`	function = A function to normalise the isolation window intensity. The default function is very generalised and just accounts for edge effects
`ilim`	numeric = All peaks less than this percentage of the target peak will be removed from the purity calculation, default is 5% (0.05)
`sampleOnly`	boolean = if TRUE will only calculate purity for sample peaklists
`isotopes`	boolean = TRUE if isotopes are to be removed
`im`	matrix = Isotope matrix, default removes C13 isotopes (single, double and triple bonds)

Value

purityD object with predicted purity of peaks

purityD object

See Also

`dimsPredictPuritySingle`

Examples

```
datapth <- system.file("extdata", "dims", "mzML", package="msPurityData")
inDF <- Getfiles(datapth, pattern=".mzML", check = FALSE, cStrt = FALSE)
ppDIMS <- purityD(fileList=inDF, cores=1, mzML=TRUE)
ppDIMS <- averageSpectra(ppDIMS)
ppDIMS <- filterp(ppDIMS)
ppDIMS <- subtract(ppDIMS)
ppDIMS <- dimsPredictPurity(ppDIMS)
```

`dimsPredictPuritySingle` *Predict the precursor purity from a DI-MS dataset*

Description

Given a an DI-MS dataset (either mzML or .csv file) calculate the predicted purity for a vector of mz values.

Calculated at a given offset e.g. for 0.5 +/- Da the minOffset would be 0.5 and the maxOffset of 0.5.

A ppm tolerance is used to find the target mz value in each scan.

Usage

```
dimsPredictPuritySingle(mztargets, filepth, minOffset = 0.5,
  maxOffset = 0.5, ppm = 2.5, mzML = TRUE, iwNorm = FALSE,
  iwNormFun = NULL, ilim = 0.05, mzRback = "pwiz", isotopes = TRUE,
  im = NULL)
```

Arguments

`mztargets`	vector = mz targets to get predicted purity for
`filepth`	character = mzML file path or .csv file path
`minOffset`	numeric = isolation window minimum offset
`maxOffset`	numeric = isolation window maximum offset
`ppm`	numeric = tolerance for target mz value in each scan
`mzML`	boolean = Whether an mzML file is to be used or .csv file (TRUE == mzML)
`iwNorm`	boolean = if TRUE then the intensity of the isolation window will be normalised based on the iwNormFun function
`iwNormFun`	function = A function to normalise the isolation window intensity. The default function is very generalised and just accounts for edge effects
`ilim`	numeric = All peaks less than this percentage of the target peak will be removed from the purity calculation, default is 5% (0.05)
`mzRback`	character = backend to use for mzR parsing
`isotopes`	boolean = TRUE if isotopes are to be removed
`im`	matrix = Isotope matrix, default removes C13 isotopes (single, double and triple bonds)

Value

a dataframe of the target mz values and the predicted purity score

Examples

```
mzmlPth <- system.file("extdata", "dims", "mzML", "B02_Daph_TEST_pos.mzML", package="msPurityData")
predicted <- dimsPredictPuritySingle(c(173.0806, 216.1045), filepth=mzmlPth , minOffset=0.5, maxOffset=0.5,
```

---

filterp,purityD-method

Filter out peaks based on intensity and RSD criteria

Description

Uses a purityD object remove peaks from either (or both) samples and blanks that are either below an intensity threshold or greater than a Relative Standard Deviation (RSD) threshold

Usage

```
## S4 method for signature 'purityD'
filterp(Object, thr = 5000, rsd = 20,
  sampleOnly = TRUE)
```

Arguments

Object	object = purityD object
thr	numeric = intensity threshold
rsd	numeric = rsd threshold
sampleOnly	boolean = if only the sample (not blanks) should be filtered

Value

purityD object

Examples

```
datapth <- system.file("extdata", "dims", "mzML", package="msPurityData")
inDF <- Getfiles(datapth, pattern=".mzML", check = FALSE, cStrt = FALSE)

ppDIMS <- purityD(inDF, cores=1)
ppDIMS <- averageSpectra(ppDIMS)
ppDIMS <- filterp(ppDIMS, thr = 5000)
```

`frag4feature,purityA-method`
Assign precursor purity scored fragmentation spectra to XCMS features

Description

Assign fragmentation spectra (MS/MS) scored via msPurity package to features from an XCMS set object.

Allows the user to filter out spectra below a certain threshold for purity.

Usage

```
## S4 method for signature 'purityA'
frag4feature(pa, xset, ppm = 5, plim = 0,
  intense = TRUE, convert2RawRT = TRUE)
```

Arguments

pa	= purityA object
xset	xcms object = XCMS object derived from the same files as the puritydf
ppm	numeric = ppm tolerance between precursor mz and feature mz
plim	numeric = min purity of precursor to be included
intense	boolean = If the most intense precursor or the centered precursor is used
convert2RawRT	boolean = If retention time correction has been used in XCMS set this to TRUE

Value

a dataframe of the purity score of the ms/ms spectra

Examples

```
msmsPths <- list.files(system.file("extdata", "lcms", "mzML", package="msPurityData"), full.names = TRUE, pa
xset <- xcms::xcmsSet(msmsPths, nSlaves = 1)
xset <- xcms::group(xset)
xset <- xcms::retcor(xset)
xset <- xcms::group(xset)

pa  <- purityA(msmsPths, interpol = "linear")
```

Getfiles *Get files for DI-MS processing*

Description

Takes in a folder path and outputs the a data frame structure for purityD. Function modified from mzmatch.

Usage

```
Getfiles(projectFolder = NULL, recursive = FALSE, pattern = ".csv",
  check = TRUE, raw = FALSE, peakout = NA, cStrt = TRUE,
  mzml_out = FALSE)
```

Arguments

projectFolder	character: directory path
recursive	boolean: recursively check for files
pattern	character file suffix to check for
check	boolean check with a GUI the files
raw	(REDUNDANT)
peakout	(REDUNDANT)
cStrt	boolean use the first word as the class name for files
mzml_out	(REDUNDANT)

Value

dataframe of files

Examples

```
datapth <- system.file("extdata", "dims", "mzML", package="msPurityData")
inDF <- Getfiles(datapth, pattern=".mzML", check = FALSE, cStrt = FALSE)
```

getP,purityD-method *Get peaklist for a purityD object*

Description

output peak list for a purityD object

Usage

```
## S4 method for signature 'purityD'
getP(x)
```

Arguments

x	object = purityD object

Value

peaks

Examples

```
datapth <- system.file("extdata", "dims", "mzML", package="msPurityData")
inDF <- Getfiles(datapth, pattern=".mzML", check = FALSE, cStrt = FALSE)
ppDIMS <- purityD(fileList=inDF, cores=1, mzML=TRUE)
peaks <- getP(ppDIMS)
```

---

groupPeaks,purityD-method

Using purityD object, group multiple peaklists by similar mz values (mzML or .csv)

Description

Uses a purityD object to group all the peaklists in the 'avPeaks$processing' slot

Usage

```
## S4 method for signature 'purityD'
groupPeaks(Object, ppm = 3, sampleOnly = FALSE,
  clustType = "hc")
```

Arguments

Object	object = purityD object
ppm	numeric = The ppm tolerance to group peaklists
sampleOnly	= if TRUE the sample peaks will only be grouped
clustType	= if 'hc' the hierarchical clustering, if 'simple' the mz values will just be grouped using a simple 1D method

Value

data.frame of peaklists grouped together by mz

Examples

```
datapth <- system.file("extdata", "dims", "mzML", package="msPurityData")
inDF <- Getfiles(datapth, pattern=".mzML", check = FALSE, cStrt = FALSE)
ppDIMS <- purityD(fileList=inDF, cores=1, mzML=TRUE)
ppDIMS <- averageSpectra(ppDIMS)
grpedP <- groupPeaks(ppDIMS)
```

groupPeaksEx *Group peaklists from a list of dataframes*

Description

Group a list of dataframes by their m/z values

Usage

```
groupPeaksEx(peak_list, cores = 1, clustType = "hc", ppm = 2)
```

Arguments

peak_list	list = A list (named) of dataframes consiting of a least the following columns ['peakID', 'mz']
cores	= number of cores used for calculation
clustType	= if 'hc' the hierarchical clustering, if 'simple' the mz values will just be grouped using a simple 1D method
ppm	numeric = The ppm tolerance to group peaklists

Value

data.frame of peaklists grouped together by mz

Examples

```
datapth <- system.file("extdata", "dims", "mzML", package="msPurityData")
inDF <- Getfiles(datapth, pattern=".mzML", check = FALSE, cStrt = FALSE)
ppDIMS <- purityD(fileList=inDF, cores=1, mzML=TRUE)
ppDIMS <- averageSpectra(ppDIMS)
grpedP <- groupPeaks(ppDIMS)
```

initialize,purityD-method

Constructor for S4 class to represent a DI-MS purityD

Description

The class used to predict purity from an DI-MS dataset.

Usage

```
## S4 method for signature 'purityD'
initialize(.Object, fileList, cores = 1, mzML = TRUE,
  mzRback = "pwiz")
```

Arguments

.Object	object = purityD object
fileList	data.frame = created using GetFiles, data.frame with filepaths and sample class information
cores	numeric = Number of cores used to perform Hierarchical clustering WARNING: memory intensive, default 1
mzML	boolean = TRUE if mzML to be used FALSE if .csv file to be used
mzRback	character = backend to use for mzR parsing

Value

purityD object

Examples

```
datapth <- system.file("extdata", "dims", "mzML", package="msPurityData")
inDF <- Getfiles(datapth, pattern=".mzML", check = FALSE, cStrt = FALSE)
ppDIMS <- purityD(fileList=inDF, cores=1, mzML=TRUE)
```

---

pcalc *Perform purity calculation on a peak matrix*

Description

This is the main purity calculation that is performed in purityX, purityD and purityA.

- Takes in a matrix of peaks
- gets isolation window based on mzmin mzmax
- locates the mz target in the peak matrix
- removes isotopic peaks
- removes any peaks below limit (percentage of target peak intensity)
- normalises
- Calculates purity: Divides the target peak intensity by the total peak intensity for the isolation window

Usage

```
pcalc(peaks, mzmin, mzmax, mztarget, ppm = NA, iwNorm = FALSE,
  iwNormFun = NULL, ilim = 0, targetMinMZ = NA, targetMaxMZ = NA,
  isotopes = FALSE, im = NULL)
```

Arguments

peaks	matrix = matrix of peaks consisting of 2 columns: mz and i
mzmin	numeric = isolation window (min)
mzmax	numeric = isolation window (max)
mztarget	numeric = the mz window to target in the isolation window
ppm	numeric = ppm tolerance for the target mz value. If NA will presume targetMinMZ and targetMaxMZ will be used
iwNorm	boolean = if TRUE then the intensity of the isolation window will be normalised based on the iwNormFun function
iwNormFun	function = A function to normalise the isolation window intensity. The default function is very generalised and just accounts for edge effects
ilim	numeric = All peaks less than this percentage of the target peak will be removed from the purity calculation, default is 5% (0.05)
targetMinMZ	numeric = range to look for the mztarget (min)
targetMaxMZ	numeric = range to look for the mztarget (max)
isotopes	boolean = TRUE if isotopes are to be removed
im	matrix = Isotope matrix, default removes C13 isotopes (single, double and triple bonds)

Value

a vector of the purity score and the number of peaks in the window e.g c(purity, pknm)

Examples

```
pm <- rbind(c(100, 1000),c(101.003, 10))
pcalc(pm, mzmin = 98, mzmax = 102, mztarget=100, ppm=5)
pcalc(pm, mzmin = 98, mzmax = 102, mztarget=100, ppm=5, isotopes = TRUE)
```

---

purityA *Assess the purity of multiple LC-MS/MS or DI-MS/MS files (constructor)*

Description

Constructor for the purityA class.

Given a vector of LC-MS/MS or DI-MS/MS mzML file paths calculate the precursor purity of each MS/MS scan

Will automatically determine the isolation widths offsets from the mzML file. For some vendors though this is not recorded (Agilent). In these cases the offsets should be given as a parameter.

In the case of Agilent only the "narrow" isolation is supported. This roughly equates to +/- 0.65 Da (depending on the instrument). If the file is detected as originating from an Agilent instrument the isolation widths will automatically be set as +/- 0.65 Da.

Usage

```
purityA(fileList, cores = 1, mostIntense = FALSE, nearest = TRUE,
  offsets = NA, plotP = FALSE, plotdir = NULL, interpol = "linear",
  iwNorm = FALSE, iwNormFun = NULL, ilim = 0.05, mzRback = "pwiz",
  isotopes = TRUE, im = NULL)
```

Arguments

fileList	vector = mzML file paths for MS/MS spectra
cores	numeric = number of cores to use
mostIntense	boolean = True if the most intense peak is used for calculation. False if the centered peak is used
nearest	boolean = True if the peak selected is from either the preceding scan or the nearest.
offsets	vector = overide the isolation offsets found in the mzML filee.g. c(0.5, 0.5)
plotP	boolean = if TRUE a plot of the purity is to be saved
plotdir	vector = if plotP is TRUE plots will be saved to this directory
interpol	character = type of interolation to be performed "linear" or "spline"
iwNorm	boolean = if TRUE then the intensity of the isolation window will be normalised based on the iwNormFun function
iwNormFun	function = A function to normalise the isolation window intensity. The default function is very generalised and just accounts for edge effects
ilim	numeric = All peaks less than this percentage of the target peak will be removed from the purity calculation, default is 5% (0.05)
mzRback	character = backend to use for mzR parsing
isotopes	boolean = TRUE if isotopes are to be removed
im	matrix = Isotope matrix, default removes C13 isotopes (single, double and triple bonds)

Value

a dataframe of the purity score of the ms/ms spectra

See Also

assessPuritySingle

Examples

```
filepths <- system.file("extdata", "lcms", "mzML", "LCMSMS_1.mzML", package="msPurityData")
pa <- purityA(filepths)
```

purityD-class *An S4 class to represent a DI-MS purityD*

Description

The class used to assess anticipated purity from a DI-MS run

Arguments

.Object	object = purityD object
fileList	data.frame = created using GetFiles, data.frame with filepaths and sample class information
cores	numeric = Number of cores used to perform Hierarchical clustering WARNING: memory intensive, default 1
mzML	boolean = TRUE if mzML to be used FALSE if .csv file to be used

Value

purityD object

Examples

```
datapth <- system.file("extdata", "dims", "mzML", package="msPurityData")
inDF <- Getfiles(datapth, pattern=".mzML", check = FALSE, cStrt = FALSE)
ppDIMS <- purityD(fileList=inDF, cores=1, mzML=TRUE)
```

purityX *Assessing anticipated purity of XCMS features from an LC-MS run*

Description

Constructor for the purityX class.

Given an XCMS object get the anticipated precursor purity of the grouped peaks

Usage

```
purityX(xset, purityType = "purityFWHMmedian", offsets = c(0.5, 0.5),
  fileignore = NULL, cores = 1, xgroups = NULL, iwNorm = FALSE,
  iwNormFun = NULL, ilim = 0, plotP = FALSE, mzRback = "pwiz",
  isotopes = FALSE, im = NULL)
```

Arguments

xset	object = xcms object
purityType	character = Area and average used for the purity predictions. Options are "purityFWHMmedian", "purityFWmedian", "purityFWHMmean", "purityFWmean"
offsets	vector = vector of the isolation window upper and lower offsets
fileignore	vector = vector of files to ignore for the prediction calculation
cores	numeric = number of cores to use
xgroups	vector = vector of xcms groups to perform prediction on
iwNorm	boolean = if TRUE then the intensity of the isolation window will be normalised based on the iwNormFun function
iwNormFun	function = A function to normalise the isolation window intensity. The default function is very generalised and just accounts for edge effects
ilim	numeric = All peaks less than this percentage of the target peak will be removed from the purity calculation, default is 5% (0.05)
plotP	boolean = TRUE if plot of the EIC of feature and associated contamination is the be save to the working directory
mzRback	character = backend to use for mzR parsing
isotopes	boolean = TRUE if isotopes are to be removed
im	matrix = Isotope matrix, default removes C13 isotopes (single, double and triple bonds)

Value

a purityX object containing a dataframe of predicted purity scores

Examples

```
msPths <- list.files(system.file("extdata", "lcms", "mzML", package="msPurityData"), full.names = TRUE, patt
xset <- xcms::xcmsSet(msPths)
xset <- xcms::group(xset)
xset <- xcms::retcor(xset)
xset <- xcms::group(xset)
ppLCMS <- purityX(xset, cores = 1, xgroups = c(1, 2))
```

---

show,purityA-method *Show method for purityA class*

Description

print statement for purityA class

Usage

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

Arguments

object	object = purityA object

Value

a print statement of regarding object

---

show,purityD-method *Show method for purityD*

Description

Show method for purityD object

Usage

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

Arguments

object	= purityD object

Value

a print statement of regarding object

---

show,purityX-method *Show method for purityX*

Description

Show method for purityX object

Usage

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

Arguments

object	= purityX object

Value

a print statement of regarding object

subtract,purityD-method *Using Subtract MZ values based on ppm tolerance and noise ratio*

Description

Uses a purityD object with references to multiple MS files. Subtract blank peaks from the sample peaks see subtractMZ for more information

Usage

```
## S4 method for signature 'purityD'
subtract(Object, byClass = TRUE, mapping = c("sample",
  "blank"), ppm = 5, s2bthres = 10)
```

Arguments

Object	= purityD object
byClass	boolean = subtract within each class
mapping	parameter not functional (TODO)
ppm	numeric = ppm tolerance
s2bthres	numeric = threshold for the samp2blank (i1/i2)

Value

purityD object with averaged spectra

See Also

subtractMZ

Examples

```
datapth <- system.file("extdata", "dims", "mzML", package="msPurityData")
inDF <- Getfiles(datapth, pattern=".mzML", check = FALSE, cStrt = FALSE)

ppDIMS <- purityD(inDF, cores=1)
ppDIMS <- averageSpectra(ppDIMS)
ppDIMS <- filterp(ppDIMS, thr = 5000)
ppDIMS <- subtract(ppDIMS)
```

subtractMZ *Subtract MZ values based on ppm tolerance and noise ratio*

Description

This function is intended for blank subtraction of mz values from two peaklists. It takes in 2 vectors of mz values and 2 coresponding vectors of Intensity values.

The second mz values are subtracted from the first set within an MZ tolerance.

However, if the mz match but the intensity is above a defined threshold then they are not subtracted

Usage

```
subtractMZ(mz1, mz2, i1, i2, ppm = 5, s2bthres = 10)
```

Arguments

mz1	vector = mz values to start with
mz2	vector = mz values to subtract
i1	vector = i values for mz1
i2	vector = i values for mz2
ppm	numeric = ppm tolerance
s2bthres	numeric = threshold for the samp2blank (i1/i2)

Value

a vector of the remaining mz values

Examples

```
mz1 <- c(100.001, 200.002, 300.302)
mz2 <- c(100.004, 200.003, 500.101)
i1 <- c(100, 100, 100)
i2 <- c(100, 10000, 100)

subtractMZ(mz1, mz2, i1, i2, ppm=5, s2bthres =10)
```

validate,purityA-method *Validate precursor purity predictions using LC-MS and LC-MS/MS dataset*

Description

The method is used to validate the precursor purity predictions made from an LC-MS dataset

Usage

```
## S4 method for signature 'purityA'
validate(pa, ppLCMS)
```

Arguments

pa	= purityA object
ppLCMS	= purityX object

Value

purityA object

---

writeOut,purityD-method

Using purityD object, save peaks as text files

Description

Uses a purityD object with references to multiple MS files. Predicts the purity of the processed sample files

Usage

```
## S4 method for signature 'purityD'
writeOut(Object, outDir, original)
```

Arguments

Object	object = purityD object
outDir	character = Directory to save text files
original	boolean = if the original (unprocessed) files are to be saved to text files

Value

purityD object

Index