Package 'IDSL.IPA'

January 7, 2023

Type Package

Title Intrinsic Peak Analysis (IPA) for HRMS Data

Version 2.6

Depends R (>= 4.0)

Imports IDSL.MXP (>= 1.8), doParallel, foreach, readxl

Author Sadjad Fakouri-Baygi [cre, aut]
(<https://orcid.org/0000-0002-6864-6911>),
Dinesh Barupal [aut] (<https://orcid.org/0000-0002-9954-8628>)

Maintainer Sadjad Fakouri-Baygi <sadjad.fakouri-baygi@mssm.edu>

Description
A multi-layered untargeted pipeline for high-throughput LC/HRMS data processing to extract signals of organic small molecules. The package performs ion pairing, peak detection, peak table alignment, retention time correction, aligned peak table gap filling, peak annotation and visualization of extracted ion chromatograms (EICs) and total ion chromatograms (TICs).

License MIT + file LICENSE

URL https://github.com/idslme/idsl.ipa

BugReports https://github.com/idslme/idsl.ipa/issues

Encoding UTF-8

LazyData true

Archs i386, x64

NeedsCompilation no

Repository CRAN

Date/Publication 2023-01-07 04:10:09 UTC

R topics documented:

`alignedPeakPropertyTableCorrelationListCalculator`
Aligned Peak Property Table Correlation List Calculator

Description

Aligned Peak Property Table Correlation List Calculator

Usage

```
alignedPeakPropertyTableCorrelationListCalculator(peakPropertyTable,
RTtolerance = 0.05, minFreqDetection = 3, minRatioDetection = 0.01,
method = "pearson", minThresholdCorrelation = 0, number_processing_threads = 1)
```

Arguments

`peakPropertyTable`	peak property table such as 'peak_height', 'peak_area' and 'peak_R13C'
`RTtolerance`	retention time tolerance (min)
`minFreqDetection`	minimum frequency of detection for a (m/z-RT) peak across the peak property table
`minRatioDetection`	minimum ratio of detection for a (m/z-RT) peak across the peak property table. This value should be between (0 - 1).
`method`	a character string indicating which correlation coefficient (or covariance) is to be computed. One of "pearson" (default), or "spearman": can be abbreviated. (from 'cor' function of the 'stats' package)
`minThresholdCorrelation`	minimum threshold for the correlation method
`number_processing_threads`	number of processing threads

Value

A list of related peak IDs for each individual (m/z-RT) pair on the peak property table

analyteRetentionTimeCorrector *analyte retention time corrector*

Description

This function calculates corrected retention times for the peaklists.

Usage

```
analyteRetentionTimeCorrector(referenceMZRTpeaks, inputPathPeaklist, peaklistFileName,
massAccuracy, RTcorrectionMethod, refPeakTolerance = 1, degreePolynomial = 3)
```

Arguments

referenceMZRTpeaks	a matrix of reference peaks for retention time correction.
inputPathPeaklist	input path to peaklist
peaklistFileName	file name peaklist
massAccuracy	mass error to detect common reference peaks.
RTcorrectionMethod	c('RetentionIndex','Polynomial')
refPeakTolerance	number of reference peaks for retention time correction using the 'RetentionIndex' method.
degreePolynomial	polynomial degree for retention time correction using the 'Polynomial' method.

Value

a list of corrected retention times for each peaklist.

chromatogramMatrix *chromatogram builder for m/z = 263.1678 in 003.d from cord blood sample*

Description

This data illustrates a chromatogram and baseline vectors to indicate chromatogram development.

Usage

```
data("chromatogramMatrix")
```

Format

A data frame with 219 observations on the following 6 variables.

`scanNumber` a numeric vector

`retentionTime` a numeric vector

`smoothChromatogram` a numeric vector

`rawChromatogram` a numeric vector

`'12C/13C Isotopologue Pairs'` a numeric vector

`Baseline` a numeric vector

Examples

```
data(chromatogramMatrix)
```

`chromatographicPeakAnalysis`

Chromatography analysis

Description

This function detects individual chromatographic peaks and measures their peak qualification metrics.

Usage

```
chromatographicPeakAnalysis(spectraScanXIC, aggregatedSpectraList, retentionTime,
LretentionTime, massAccuracy, mzTarget, rtTarget = NULL, scanNumberStart,
scanNumberEnd, smoothingWindow, peakResolvingPower, minNIonPair, minPeakHeight,
minRatioIonPair, maxRPW, minSNRbaseline, maxR13CcumulatedIntensity,
maxPercentageMissingScans, nSpline, exportEICparameters  = NULL)
```

Arguments

`spectraScanXIC`	a matrix consists of 5 columns. The column contents are the m/z of 12C isotopologues, intensity of 12C isotopologues, scan number (t), m/z of 13C isotopologues, and intensity of 13C isotopologues, respectively. Redundant scan numbers are not allowed for this module.
`aggregatedSpectraList`	aggregated spectraList and spectra matrix from the 'IPA_spectraListAggregator' module
`retentionTime`	a vector of retention times vs. corresponding scan numbers
`LretentionTime`	length of the retention time vector
`massAccuracy`	mass error to perform chromatography analysis
`mzTarget`	m/z value to perform chromatography analysis

`rtTarget` retention time value for a targeted peak to calculate the ancillary chromatography parameters. When this parameter set at 0, the ancillary chromatography parameters are calculated for the entire detected peaks.

`scanNumberStart` the first scan number.

`scanNumberEnd` the last scan number.

`smoothingWindow` number of scans for peak smoothing

`peakResolvingPower` a value to represent peak resolving power

`minNIonPair` minimum number of nIsoPair for an individual peak

`minPeakHeight` minimum peak height for an individual peak

`minRatioIonPair` minimum ratio of nIsoPair per number of available scans within an individual peak

`maxRPW` maximum allowed value of ratio of peak width at half-height to baseline (RPW) for an individual peak

`minSNRbaseline` minimum S/N baseline for an individual peak

`maxR13CcumulatedIntensity` maximum allowed value of average R13C for an individual peak

`maxPercentageMissingScans` maximum allowed value of percentage missing scans on the raw chromatogram for an individual peak.

`nSpline` number of points for further smoothing using a cubic spline smoothing method to calculate ancillary chromatographic parameters

`exportEICparameters` When 'NULL', EICs are not plotted. 'exportEICparameters' should contain three variables of 1) an address to save IPA EICs figures, 2) 'HRMS' file name, and 3) a valid string of characters.

Value

a data frame consisting of 24 columns representing chromatography and mass spectrometry parameters. Each row represents an individual separated chromatographic peak.

`chromatographicPeakDetector` *peak detection*

Description

This function detects separated chromatographic peaks on the chromatogram.

Usage

```
chromatographicPeakDetector(int)
```

Arguments

int	a vector of intensities of the chromatogram.

Value

A matrix of 2 columns. Each row indicates peak boundary indices on the 'int' vector.

Examples

```
data(chromatogramMatrix)
int <- chromatogramMatrix$smoothChromatogram
chromatographicPeakDetector(int)
```

---

derivative5pointsStencil

Numerical differentiation by five-point stencil method

Description

This module performs numerical differentiation using the five-point stencil method.

Usage

```
derivative5pointsStencil(x, y, n)
```

Arguments

x	a vector of values for x.
y	a vector of values for y.
n	order of numerical differentiation (n=1-4).

Value

A matrix of 2 columns. The first column represents x and the second column represents numerical differentiation values. This matrix has four rows (two rows from the beginning and 2 rows from the end) less than length of x or y.

Examples

```
data(peak_spline)
rt <- peak_spline[, 1]
int <- peak_spline[, 2]
n <- 2 # second order derivative
derivative5pointsStencil(rt, int, n)
```

gapFillingCore *Gap-Filling Core Function*

Description

Gap-Filling Core Function

Usage

```
gapFillingCore(input_path_hrms, peakXcol, massAccuracy, RTtolerance, scanTolerance,
retentionTimeCorrectionCheck = FALSE, listCorrectedRTpeaklists = NULL,
inputPathPeaklist = NULL, ionMassDifference = 1.003354835336,
number_processing_threads = 1)
```

Arguments

input_path_hrms	input_path_hrms
peakXcol	peakXcol
massAccuracy	massAccuracy
RTtolerance	RTtolerance
scanTolerance	a scan tolerance to extend the chromatogram for better calculations.
retentionTimeCorrectionCheck	retentionTimeCorrectionCheck
listCorrectedRTpeaklists	listCorrectedRTpeaklists
inputPathPeaklist	inputPathPeaklist
ionMassDifference	ionMassDifference
number_processing_threads	number of processing threads

Value

A list of gap-filled data

IPA_aggregate *aggregation method for the IDSL.IPA modules*

Description

This module is to optimize the 'indexVec' variable by removing elements that have redundant 'idVec' numbers.

Usage

```
IPA_aggregate(idVec, variableVec, indexVec, targetVar)
```

Arguments

idVec	a vector of id numbers. Repeated id numbers are allowed.
variableVec	a vector of variable of the interest such as RT, m/z, etc.
indexVec	a vector of indices
targetVar	the targeted value in 'variableVec'

Value

a clean indexVec after removing repeated 'idVec'.

IPA_baselineDeveloper *Develop a baseline for the chromatogram using local minima*

Description

This function generates a vector of baselines for the chromatogram using local minima. It also is capable of excluding outlier local minima to generate a realistic baseline including true baseline regions. This baseline may represent the local noise levels for the chromatogram.

Usage

```
IPA_baselineDeveloper(segment, int)
```

Arguments

segment	a matrix or a vecotr of adjusted scan number of local minima w/ or w/o redundant local minima. Adjusted scan numbers are the scan numbers but adjusted to start at 1.
int	a vector of intensities of the chromatogram.

Value

A vector of baselines in the same size of the "int" vector.

Examples

```
data(segment)
data(chromatogramMatrix)
int <- chromatogramMatrix$smoothChromatogram
IPA_baselineDeveloper(segment, int)
```

`IPA_CompoundsAnnotation` *Compound-centric peak annotation*

Description

This function performs compound-centric peak annotation.

Usage

```
IPA_CompoundsAnnotation(PARAM)
```

Arguments

`PARAM`	a data frame from IPA_xlsxAnalyzer function containing the IPA parameters.

Value

This function saves individual .csv files for each compound in the "compound_centric_annotation" folder.

`IPA_GapFiller` *IPA GapFiller*

Description

This function fills the gaps on the peak table.

Usage

```
IPA_GapFiller(PARAM)
```

Arguments

`PARAM`	a data frame from the ‘IPA_xlsxAnalyzer‘ function containing the IPA parameters.

Value

This function saves individual .csv and .Rdata files for the gap-filled peak tables for peak height, area, and R13C properties in the "peak_alignment" folder.

`IPA_IonPairing` *IPA Ion Pairing*

Description

This function pairs two ions with a fixed distance in high-resolution mass spectral datasets

Usage

```
IPA_IonPairing(spectraList, minSpectraNoiseLevel, massAccuracyIonPair = 0.015,
ionMassDifference = 1.003354835336)
```

Arguments

`spectraList`	list of mass spectra in each chromatogram scan
`minSpectraNoiseLevel`	intensity threshold at each chromatogram scan
`massAccuracyIonPair`	mass error to detect pair ions
`ionMassDifference`	mass difference to pair ions. (Default = DeltaC = 13C - 12C = 1.003354835336), or DeltaS = 34S - 32S = 1.9957958356, or any numerical value.

Value

A matrix consists of 5 columns. The column contents are the m/z of 12C isotopologues, intensity of 12C isotopologues, scan number (t), m/z of 13C isotopologues, and intensity of 13C isotopologues, respectively.

`IPA_logRecorder` *IPA_logRecorder*

Description

IPA_logRecorder

Usage

```
IPA_logRecorder(messageQuote, printMessage = TRUE)
```

Arguments

messageQuote	messageQuote
printMessage	printMessage

IPA_MSdeconvoluter *MS deconvoluter*

Description

This function deconvolutes mass spectrometry files into a list of mass spectra and a vector of retention times.

Usage

```
IPA_MSdeconvoluter(inputHRMSfolderPath, MSfileName, MSlevel = 1)
```

Arguments

inputHRMSfolderPath	address of the mass spectrometry file
MSfileName	mass spectrometry file.
MSlevel	MS level to extract information.

Value

spectraList	a list of mass spectra.
retentionTime	a vector of retention times for scan numbers.
MS_polarity	mass spectrometry ionization mode (+/-)

IPA_PeakAlignment *IPA peak alignment*

Description

This function produces an aligned peak table from individual peaklists.

Usage

```
IPA_PeakAlignment(PARAM)
```

Arguments

PARAM	a data frame from the ‘IPA_xlsxAnalyzer‘ function.

Value

This function saves individual .csv and .Rdata files for the aligned peak tables for peak height, area, and R13C properties in the "peak_alignment" folder.

`IPA_PeakAnalyzer` *IPA Peak Analyzer*

Description

This function performs the IPA peak detection module.

Usage

```
IPA_PeakAnalyzer(PARAM)
```

Arguments

`PARAM`	is a data frame from IPA_xlsxAnalyzer function.

Value

This function saves individual peaklist files in '.csv' and '.Rdata' formats for HRMS files in the 'peaklists' folder.

`IPA_PeaklistAnnotation` *IPA Peaklist Annotation*

Description

This function performs sample-centric peak annotation.

Usage

```
IPA_PeaklistAnnotation(PARAM)
```

Arguments

`PARAM`	a data frame from IPA_xlsxAnalyzer function.

Value

This function saves individual .csv files for peak height, area, and R13C properties in the "sample_centric_annotation" folder.

IPA_peak_alignment_folder_xlsxAnalyzer

IPA peak alignment folder xlsxAnalyzer

Description

IPA peak alignment folder xlsxAnalyzer

Usage

```
IPA_peak_alignment_folder_xlsxAnalyzer(PARAM, PARAM_ID, checkpoint_parameter,
correctedRTcheck = FALSE, CSAcheck = FALSE, allowedVerbose = TRUE)
```

Arguments

PARAM	PARAM
PARAM_ID	PARAM_ID
checkpoint_parameter	checkpoint_parameter
correctedRTcheck	correctedRTcheck
CSAcheck	CSAcheck
allowedVerbose	c(TRUE, FALSE). A 'TRUE' allowedVerbose provides messages about the flow of the function.

Value

PARAM	PARAM
checkpoint_parameter	checkpoint_parameter

IPA_spectraListAggregator

spectraList filtering

Description

This module stacks the spectraList object and creates a list of ions for a rapid spectra query.

Usage

```
IPA_spectraListAggregator(spectraList)
```

Arguments

spectraList	a list of mass spectra in each chromatogram scan.

Value

aggregatedSpectraList	aggregated spectraList
spectraListMatrix	matrix of row bounded spectraList

IPA_targeted *IPA Targeted Analysis*

Description

This function plots extracted ion chromatogram (EIC) figures in the targeted mode.

Usage

```
IPA_targeted(PARAM_targeted, allowedVerbose = TRUE)
```

Arguments

PARAM_targeted	IPA parameters to feed the 'IPA_targeted' module. This variable can be produced using the 'IPA_targeted_xlsxAnalyzer' module.
allowedVerbose	c(TRUE, FALSE). A 'TRUE' allowedVerbose provides messages about the flow of the function.

Value

This module saves extracted ion chromatograms (EICs) in .png format in the "Targeted_EICs" folder and saves a table of peak properties.

IPA_targeted_xlsxAnalyzer *IPA Targeted xlsxAnalyzer*

Description

This function processes the spreadsheet of the IPA parameters to ensure the parameter inputs are in agreement with the 'IPA_targeted' function.

Usage

```
IPA_targeted_xlsxAnalyzer(spreadsheet)
```

Arguments

spreadsheet contains the IPA parameters.

Value

'PARAM_targeted' which is the IPA parameters to feed the 'IPA_targeted' function.

Examples

```
## To generate the IPA spreadsheet parameters
s_path <- system.file("extdata", package = "IDSL.IPA")
SSh1 <- paste0(s_path,"/IPA_parameters.xlsx")
spreadsheet <- readxl::read_xlsx(SSh1, sheet = 'IPA_targeted')
temp_wd <- tempdir()
temp_wd_zip <- paste0(temp_wd,"/idsl_ipa_test_files.zip")
download.file(paste0("https://github.com/idslme/IDSL.IPA/blob/main/",
"IPA_educational_files/idsl_ipa_test_files.zip?raw=true"),
destfile = temp_wd_zip, mode = "wb")
unzip(temp_wd_zip, exdir = temp_wd)
spreadsheet[3, 4] <- temp_wd
spreadsheet[7, 4] <- temp_wd
spreadsheet[8, 4] <- "53.01853, 61.00759"
spreadsheet[9, 4] <- "0.951, 0.961"
##
PARAM_targeted <- IPA_targeted_xlsxAnalyzer(spreadsheet)
```

IPA_workflow *IPA Workflow*

Description

This function executes the IPA workflow in order.

Usage

```
IPA_workflow(spreadsheet)
IPA_Workflow(spreadsheet)
```

Arguments

spreadsheet IPA spreadsheet

Value

This function organizes the IPA file processing for a better performance using the template spreadsheet.

See Also

https://ipa.idsl.me/home

Examples

```
library(IDSL.IPA)
s_path <- system.file("extdata", package = "IDSL.IPA")
SSh1 <- paste0(s_path,"/IPA_parameters.xlsx")
## To see the results, use a known folder instead of the `tempdir()` command
temp_wd <- tempdir()
temp_wd_zip <- paste0(temp_wd, "/idsl_ipa_test_files.zip")
spreadsheet <- readxl::read_xlsx(SSh1)
download.file(paste0("https://github.com/idslme/IDSL.IPA/blob/main/",
"IPA_educational_files/idsl_ipa_test_files.zip?raw=true"),
destfile = temp_wd_zip, mode = "wb")
unzip(temp_wd_zip, exdir = temp_wd)
spreadsheet[7, 4] <- temp_wd
spreadsheet[43, 4] <- s_path
spreadsheet[10, 4] <- temp_wd
IPA_workflow(spreadsheet)
```

IPA_xlsxAnalyzer *IPA xlsx Analyzer*

Description

This function processes the spreadsheet of the IPA parameters to ensure the parameter inputs are in agreement with the IPA requirements.

Usage

```
IPA_xlsxAnalyzer(spreadsheet)
```

Arguments

spreadsheet	IPA spreadsheet

Value

This function returns the IPA parameters to feed the IPA_Workflow, IPA_CompoundsAnnotation, IPA_GapFiller, IPA_PeakAlignment, IPA_PeakAnalyzer, and IPA_PeaklistAnnotation functions.

Examples

```
s_path <- system.file("extdata", package = "IDSL.IPA")
SSh1 <- paste0(s_path, "/IPA_parameters.xlsx")
temp_wd <- tempdir()
temp_wd_zip <- paste0(temp_wd,"/idsl_ipa_test_files.zip")
spreadsheet <- readxl::read_xlsx(SSh1)
download.file(paste0("https://github.com/idslme/IDSL.IPA/blob/main/",
"IPA_educational_files/idsl_ipa_test_files.zip?raw=true"),
destfile = temp_wd_zip)
unzip(temp_wd_zip, exdir = temp_wd)
spreadsheet[7, 4] <- temp_wd
spreadsheet[40, 4] <- s_path # reference file location
spreadsheet[10, 4] <- temp_wd # output data location
PARAM <- IDSL.IPA::IPA_xlsxAnalyzer(spreadsheet)
```

`islocalminimum` *islocalminimum*

Description

This function returns indices of local minimum points on a curve.

Usage

```
islocalminimum(y)
```

Arguments

y	is a vector of y values.

Value

A vector in the same size of the vector 'y'. Local minimum arrays represented by -1.

Examples

```
data(chromatogramMatrix)
int <- chromatogramMatrix$smoothChromatogram
islocalminimum(int)
```

islocaloptimum *islocaloptimum*

Description

This function returns indices of local minimum and maximum points on a curve.

Usage

```
islocaloptimum(y)
```

Arguments

y	is a vector of y values.

Value

A vector in the same size of the vector 'y'. Local minimum and maximum arrays represented by -1 and +1, respectively.

Examples

```
data(chromatogramMatrix)
int <- chromatogramMatrix$smoothChromatogram
islocaloptimum(int)
```

loadRdata *loadRdata*

Description

This function loads .Rdata files into a variable.

Usage

```
loadRdata(fileName)
```

Arguments

fileName	is an '.Rdata' file.

Value

The called variable into the new assigned variable name.

mzClusteringRawXIC *m/z clustering raw XIC*

Description

This function clusters related 12C m/z values.

Usage

```
mzClusteringRawXIC(spectraScan123, massAccuracy, minNIonPair, minPeakHeightXIC)
```

Arguments

spectraScan123	a matrix consists of 3 columns. The column contents are the m/z of 12C isotopologues, intensity of 12C isotopologues, and scan number (t).
massAccuracy	mass accuracy to detect related 12C m/z values.
minNIonPair	minimum number of nIsoPair for an individual peak.
minPeakHeightXIC	minimum peak height for an individual raw EIC

Value

This function returns an list on index numbers of EICs for the "spectraScan" variable.

mzRTindexer *m/z - RT Indexer*

Description

This function locate the closest pair of a reference (m/z - RT) pair in a 2-D grid of ‘m/z‘ and ‘RT‘ vectors.

Usage

```
mzRTindexer(MZvec, RTvec, MZref, RTref, massAccuracy, RTtolerance)
```

Arguments

MZvec	m/z vector
RTvec	RT vector
MZref	a reference m/z
RTref	a reference RT
massAccuracy	m/z tolerance
RTtolerance	RT tolerance

Value

index of closest pair to the reference (m/z - RT) pair

Note

This function returns NULL in case no match is detected.

`opendir` *opendir*

Description

This function opens the directory.

Usage

```
opendir(dir)
```

Arguments

`dir`	full address of the directory.

Value

This function opens its input directory for the user.

`peakAlignmentCore` *Peak Alignment Core*

Description

This function aligns peaks from multiple peaklists and produces an aligned table of common peaks among multiple samples.

Usage

```
peakAlignmentCore(peaklistInputFolderPath, peaklistFileNames, listCorrectedRTpeaklists,
massAccuracy, RTtolerance, number_processing_threads = 1)
```

Arguments

peaklistInputFolderPath	path to directory of peaklists.
peaklistFileNames	name of peaklists for peak table production.
listCorrectedRTpeaklists	a list of corrected or uncorrected retention times for each peaklist.
massAccuracy	mass error to detect common peaks.
RTtolerance	retention time tolerance to detect common peaks.
number_processing_threads	number of processing threads

Value

This function returns an aligned peak table with index numbers from individual peaklists for each peak.

---

peakAreaCalculator *peak area*

Description

This function calculates area under the curve using a trapezoid method.

Usage

```
peakAreaCalculator(x, y)
```

Arguments

x	is a vector of x values.
y	is a vector of y values.

Value

A number for the integrated peak area.

Examples

```
data("peak_spline")
rt <- peak_spline[, 1]
int <- peak_spline[, 2]
peakAreaCalculator(rt, int)
```

peakAsymmetryFactorCalculator

Asymmetry factor for a chromatographic peak

Description

This function calculates an asymmetry factor for a chromatographic peak.

Usage

```
peakAsymmetryFactorCalculator(rt, int)
```

Arguments

rt	a vector of retention times for the chromatographic peak.
int	a vector of intensities corresponding to the vector of retention times for the chromatographic peak.

Value

asymmetry of the chromatographic peak. 1 is for very symmetric peak.

Examples

```
data(peak_spline)
rt <- peak_spline[, 1]
int <- peak_spline[, 2] - peak_spline[, 3]
peakAsymmetryFactorCalculator(rt, int)
```

peakDerivativeSkewnessCalculator

Peak Derivative Skewness Calculator

Description

This function calculates skewness of a chromatographic peak using first order degree of numerical differentiation.

Usage

```
peakDerivativeSkewnessCalculator(rt, int)
```

Arguments

rt	a vector representing retention times of the chromatographic peak.
int	a vector representing intensities of the chromatographic peak.

Value

Skewness of a chromatographic peak. 1 is for very symmetric peak. Minimum is 0 from this function.

Examples

```
data(peak_spline)
rt <- peak_spline[, 1]
int <- peak_spline[, 2]
peakDerivativeSkewnessCalculator(rt, int)
```

---

peakFrontingTailingResolver

Fronting and tailing peaks resolver

Description

This function attempts to resolve peak tailings or frontings into the main peak in case they were detected as seperate peaks.

Usage

```
peakFrontingTailingResolver(segment, int, maxScanDifference, peakResolvingPower = 0.025)
```

Arguments

segment	a matrix or a vector of peak boundaries.
int	a vector of intensities of the entire chromatogram.
maxScanDifference	maximum scan number difference between peak tailing or fronting and the main peak.
peakResolvingPower	power of peak resolving tool.

Value

A matrix of 2 columns. Each row indicates peak boundary indices on the 'int' vector after resolving fronting and tailing peaks.

Examples

```
data(segment)
data(chromatogramMatrix)
int <- chromatogramMatrix$smoothChromatogram
maxScanDifference <- 7
peakResolvingPower <- 0.2
peakFrontingTailingResolver(segment, int, maxScanDifference, peakResolvingPower)
```

peakGaussianityCalculator
Peak Gaussianity Calculator

Description

This module measures gaussianity of chromatographic peak using Pearson correlation coefficients (ρ) at top 80 percent of peak.

Usage

```
peakGaussianityCalculator(RT, Int, BL, gauge = 0.8)
```

Arguments

RT	a vector of retention times of the chromatographic peak.
Int	a vector of intensities of the chromatographic peak.
BL	a vector of baseline of the chromatographic peak.
gauge	represents the gauge height of peak for Gaussianity measurement.

Value

Gaussianity of the chromatographic peak.

Examples

```
data("peak_spline")
RT <- peak_spline[, 1]
Int <- peak_spline[, 2]
BL <- peak_spline[, 3]
peakGaussianityCalculator(RT, Int, BL, gauge = 0.8)
```

peakPropertyTableFreqCalculator
Peak Property Table Frequency Calculator

Description

Peak Property Table Frequency Calculator

Usage

```
peakPropertyTableFreqCalculator(peakPropertyTable, startColumnIndex = 3,
number_processing_threads = 1, allowedVerbose = TRUE)
```

Arguments

peakPropertyTable	peakPropertyTable
startColumnIndex	startColumnIndex
number_processing_threads	number_processing_threads
allowedVerbose	c(TRUE, FALSE). A 'TRUE' allowedVerbose provides messages about the flow of the function.

Value

a vector of frequency of detection.

---

peakPropertyTableMedianCalculator

Peak Property Table Median Calculator

Description

Peak Property Table Median Calculator

Usage

```
peakPropertyTableMedianCalculator(peakPropertyTable, falggingVector = NULL,
number_processing_threads = 1, allowedVerbose = TRUE)
```

Arguments

peakPropertyTable	peakPropertyTable
falggingVector	falggingVector
number_processing_threads	number_processing_threads
allowedVerbose	c(TRUE, FALSE). A 'TRUE' allowedVerbose provides messages about the flow of the function.

Value

updated peak property table

peakPseudomomentsSymmetryCalculator

Peak Pseudomoments Symmetry Calculator

Description

This function measures peak symmetry and skewness using the inflection points of the peak on both sides.

Usage

```
peakPseudomomentsSymmetryCalculator(rt, int)
```

Arguments

rt	a vector of retention times for the chromatographic peak.
int	a vector of intensities corresponding to the vector of retention times for the chromatographic peak.

Value

PeakSymmetry	peak symmetry for the chromatographic peak.
Skewness	skewness for the chromatographic peak.

Examples

```
data("peak_spline")
rt <- peak_spline[, 1]
int <- peak_spline[, 2] - peak_spline[, 3]
peakPseudomomentsSymmetryCalculator(rt, int)
```

peakSharpnessCalculator

Peak Sharpness Calculator

Description

This function measures sharpness of a chromatographic peak

Usage

```
peakSharpnessCalculator(int)
```

Arguments

int	a vector of intensities of the chromatographic peak.

Value

A number representing peak sharpness. The higher values indicate higher sharpness.

Examples

```
data("peak_spline")
int <- peak_spline[, 2]
peakSharpnessCalculator(int)
```

---

peakUSPtailingFactorCalculator

Peak USP Tailing Factor Calculator

Description

This function calculates USP tailing factor at above 10 percent of the height.

Usage

```
peakUSPtailingFactorCalculator(rt, int)
```

Arguments

rt	a vector of retention times for the chromatographic peak.
int	a vector of intensities corresponding to the vector of retention times for the chromatographic peak.

Value

USP tailing factor for the chromatographic peak.

Examples

```
data(peak_spline)
rt <- peak_spline[, 1]
int <- peak_spline[, 2] - peak_spline[, 3]
peakUSPtailingFactorCalculator(rt, int)
```

peakWidthCalculator *peak width measuement*

Description

This function measures peak width at different peak heights.

Usage

```
peakWidthCalculator(rt, int, gauge)
```

Arguments

rt	a vector of retention times of the chromatographic peak.
int	a vector of intensities of the chromatographic peak.
gauge	a height gauge to measure the peak width. This parameter should be between 0-1.

Value

A peak width at the guaged height.

Examples

```
data("peak_spline")
rt <- peak_spline[, 1]
int <- peak_spline[, 2] - peak_spline[, 3]
gauge <- 0.5
peakWidthCalculator(rt, int, gauge)
```

peakXcolFiller *Peak table producer*

Description

This function fills the peak table from individual peaklists.

Usage

```
peakXcolFiller(peakXcol, inputPathPeaklist)
```

Arguments

peakXcol	a matrix of index numbers in individual peaklists for each peak (m/z-RT).
inputPathPeaklist	address of the peaklists.

Value

peak_height	peak table for height values
peak_area	peak table for area values
peak_R13C	peak table for R13C values

---

peakXcolFlagger *PeakXcol Flagger*

Description

PeakXcol Flagger

Usage

```
peakXcolFlagger(mzPeakXcol, rtPeakXcol, freqPeakXcol, massAccuracy, RTtolerance,
maxRedundantPeakFlagging)
```

Arguments

mzPeakXcol	mzPeakXcol
rtPeakXcol	rtPeakXcol
freqPeakXcol	freqPeakXcol
massAccuracy	massAccuracy
RTtolerance	RTtolerance
maxRedundantPeakFlagging	maxRedundantPeakFlagging

Value

a vector with flagged numbers

---

peak_spline *peak spline*

Description

illusterates a smoothe peak using cubic spline smoothing method

Usage

```
data("peak_spline")
```

Format

A data frame with 100 observations on the following 3 variables.

`rt_spline` a numeric vector

`int_spline` a numeric vector

`bl_approx` a numeric vector

Examples

```
data(peak_spline)
```

`plot_mz_eic` *plot_mz_eic*

Description

plot_mz_eic

Usage

```
plot_mz_eic(filelist, filelocation, mzTarget, massAccuracy,
number_processing_threads = 1, rtstart = 0, rtend = 0, plotTitle = "")
```

Arguments

`filelist`	filelist
`filelocation`	filelocation
`mzTarget`	mzTarget
`massAccuracy`	massAccuracy
`number_processing_threads`	number of processing threads
`rtstart`	rtstart
`rtend`	rtend
`plotTitle`	plotTitle

Value

plot_mz_eic

plot_simple_tic *plot_simple_tic*

Description

plot_simple_tic

Usage

```
plot_simple_tic(filelist, filelocation, number_processing_threads = 1,
plotTitle = "Total Ion Chromatogram")
```

Arguments

filelist	filelist
filelocation	filelocation
number_processing_threads	number of processing threads
plotTitle	plotTitle

Value

plot_simple_tic

primaryXICdeconvoluter *Primary peak analyzer*

Description

This function performs the first round of the chromatography analysis.

Usage

```
primaryXICdeconvoluter(spectraScan, scanTolerance, indexXIC, aggregatedSpectraList,
retentionTime, massAccuracy, smoothingWindow, peakResolvingPower, minNIonPair,
minPeakHeight, minRatioIonPair, maxRPW, minSNRbaseline, maxR13CcumulatedIntensity,
maxPercentageMissingScans, nSpline, exportEICparameters = NULL)
```

Arguments

spectraScan	a matrix consists of 5 columns. The column contents are the m/z of 12C isotopologues, intensity of 12C isotopologues, scan number (t), m/z of 13C isotopologues, and intensity of 13C isotopologues.
scanTolerance	a scan tolerance to extend the chromatogram for better calculations.
indexXIC	a list of indices of candidate 12C m/z values from spectraScan matrix.
aggregatedSpectraList	aggregated spectraList and spectra matrix from the 'IPA_spectraListAggregator' module
retentionTime	a vector of retention times vs. corresponding scan numbers.
massAccuracy	a m/z value to perform chromatography analysis.
smoothingWindow	number of scans for peak smoothing.
peakResolvingPower	a value to represent peak resolving power.
minNIonPair	minimum number of nIsoPair for an individual peak.
minPeakHeight	minimum peak height for an individual peak.
minRatioIonPair	minimum ratio of nIsoPair per number of available scans within an individual peak.
maxRPW	maximum allowed value of ratio of peak width at half-height to baseline (RPW) for an individual peak.
minSNRbaseline	minimum S/N baseline for an individual peak.
maxR13CcumulatedIntensity	maximum allowed value of average R13C for an individual peak.
maxPercentageMissingScans	maximum allowed value of percentage missing scans on the raw chromatogram for an individual peak.
nSpline	number of points for further smoothing using a cubic spline smoothing method.
exportEICparameters	When 'NULL', EICs are not plotted. 'exportEICparameters' should contain three variables of 1) an address to save IPA EICs figures, 2) 'HRMS' file name, and 3) a valid string of characters.

Value

a data frame consisting of 24 columns representing chromatography and mass spectrometry parameters. Each row represents an individual separated chromatographic peak.

`recursiveMZpeaklistCorrector` *recursive mass correction*

Description

This function performs recursive mass correction.

Usage

```
recursiveMZpeaklistCorrector(peaklist, spectraScan, scanTolerance,
aggregatedSpectraList, retentionTime, massAccuracy, smoothingWindow,
peakResolvingPower, minNIonPair, minPeakHeight, minRatioIonPair, maxRPW,
minSNRbaseline, maxR13CcumulatedIntensity, maxPercentageMissingScans, nSpline,
exportEICparameters = NULL)
```

Arguments

`peaklist`	an IPA peaklist from 'primaryXICdeconvoluter' function.
`spectraScan`	a matrix consists of 5 columns. The column contents are the m/z of 12C isotopologues, intensity of 12C isotopologues, scan number (t), m/z of 13C isotopologues, and intensity of 13C isotopologues.
`scanTolerance`	a scan tolerance to extend the chromatogram for better calculations.
`aggregatedSpectraList`	aggregated spectraList and spectra matrix from the 'IPA_spectraListAggregator' module
`retentionTime`	a vector of retention times for corresponding scan numbers.
`massAccuracy`	an m/z value to perform chromatography analysis.
`smoothingWindow`	a number of scans for peak smoothing.
`peakResolvingPower`	a value to represent peak resolving power.
`minNIonPair`	minimum number of nIsoPair for an individual peak.
`minPeakHeight`	minimum peak height for an individual peak.
`minRatioIonPair`	minimum ratio of nIsoPair per number of available scans within an individual peak.
`maxRPW`	maximum allowed value of ratio of peak width at half-height to baseline (RPW) for an individual peak.
`minSNRbaseline`	minimum S/N baseline for an individual peak.
`maxR13CcumulatedIntensity`	maximum allowed value of average R13C for an individual peak.

maxPercentageMissingScans
: maximum allowed value of percentage missing scans on the raw chromatogram for an individual peak.

nSpline
: number of points for further smoothing using a cubic spline smoothing method to calculate ancillary chromatographic parameters.

exportEICparameters
: When 'NULL', EICs are not plotted. 'exportEICparameters' should contain three variables of 1) an address to save IPA EICs figures, 2) 'HRMS' file name, and 3) a valid string of characters.

Value

a dataframe consisting of 24 columns representing chromatography and mass spectrometry parameters. Each row represents an individual separated chromatographic peak.

---

referenceRetentionTimeDetector

Reference retention time detector

Description

This module detects recurring reference peaks (m/z-RT) for retention time correction.

Usage

```
referenceRetentionTimeDetector(inputPathPeaklist, refPeaklistFileNames,
minFrequencyRefPeaks, massAccuracy, RTtolerance, number_processing_threads = 1)
```

Arguments

inputPathPeaklist
: path to directory of peaklists.

refPeaklistFileNames
: name of peaklists files to detect recurring reference peaks (m/z-RT).

minFrequencyRefPeaks
: minimum frequency of the recurring reference peaks (m/z-RT) in the reference files.

massAccuracy
: mass error to detect common peaks.

RTtolerance
: retention time tolerance to detect common peaks.

number_processing_threads
: number of processing threads

Value

referenceMZRTpeaks
: a matrix of two columns of m/z and RT of common peaks in the reference samples.

listRefRT
: a list of corrected or uncorrected retention times for each peaklist.

segment *segment*

Description

This data illustrates an output matrix of chromatogram peak detection module from the "chromatogramMatrix.rda" object.

Usage

```
data("segment")
```

Format

The format is: num [1:16, 1:2] 7 15 23 33 38 46 67 86 102 118 ...

Examples

```
data(segment)
```

SNRbaseline *SNR baseline*

Description

This function calculates S/N using local noise levels from baseline,

Usage

```
SNRbaseline(int, baseline)
```

Arguments

int	a vector of intensities corresponding to the vector of retention times for the chromatographic peak.
baseline	a vector of baseline of the chromatographic peak.

Value

S/N value

Examples

```
data("peak_spline")
int <- peak_spline[, 2]
baseline <- peak_spline[, 3]
SNRbaseline(int, baseline)
```

SNRrms *SNR RMS*

Description

This function calculates signal-to-noise ratio using root mean square.

Usage

```
SNRrms(int, baseline, gauge = 0.80)
```

Arguments

int	is the vector of intensities corresponding to the vector of retention times for the chromatographic peak.
baseline	is a vector of baseline of the chromatographic peak.
gauge	represents the gauge height of peak for gaussianity measurement.

Value

S/N value

Examples

```
data("peak_spline")
int <- peak_spline[, 2]
baseline <- peak_spline[, 3]
SNRrms(int, baseline)
```

SNRxcms *SNR xcms*

Description

This function calculates S/N values using a method suggested in the xcms paper (Tautenhahn, 2008).

Usage

```
SNRxcms(int)
```

Arguments

int	a vector of intensities corresponding to the vector of retention times for the chromatographic peak.

Value

S/N value

References

Tautenhahn, R., Böttcher, C. and Neumann, S. (2008). Highly sensitive feature detection for high resolution LC/MS. *BMC bioinformatics*, 9(1), 1-16, doi: 10.1186/147121059504.

Examples

```
data(peak_spline)
int <- peak_spline[, 2]
SNRxcms(int)
```

---

targetedIonPairing *Targeted Ion Pairing*

Description

This module only pairs ‘mzTarget‘ values across ‘scanNumberStart‘ through ‘scanNumberEnd‘ scan numbers.

Usage

```
targetedIonPairing(spectraList, scanNumberStart, scanNumberEnd, mzTarget,
massAccuracy, ionMassDifference = 1.003354835336, massAccuracyIonPair = massAccuracy*1.5)
```

Arguments

`spectraList`	spectraList which is a list of mass spectra
`scanNumberStart`	the first scan number.
`scanNumberEnd`	the last scan number.
`mzTarget`	m/z value to perform chromatography analysis
`massAccuracy`	mass accuracy to select the dominant ion
`ionMassDifference`	mass difference to pair ions. (Default = DeltaC = 13C - 12C = 1.003354835336), or DeltaS = 34S - 32S = 1.9957958356, or any numerical value.
`massAccuracyIonPair`	mass accuracy to select the second ion

Value

A targeted ion paired spectra and their scan numbers

XIC *XIC*

Description

XIC

Usage

```
XIC(aggregatedSpectraList, scanNumberStart, scanNumberEnd, mzTarget, massAccuracy)
```

Arguments

aggregatedSpectraList	aggregated spectraList and spectra matrix from the 'IPA_spectraListAggregator' module
scanNumberStart	the first scan number.
scanNumberEnd	the last scan number.
mzTarget	an m/z value to perform XIC analysis.
massAccuracy	a mass error to perform XIC analysis.

Value

A matrix of three columns representing scan number, m/z, and intensity.

Index