Package ‘peakPantheR’

April 10, 2025

Title Peak Picking and Annotation of High Resolution Experiments

Version 1.21.0

Date 2024-02-26

Description An automated pipeline for the detection, integration and reporting of predefined features across a large number of mass spectrometry data files.
It enables the real time annotation of multiple compounds in a single file, or the parallel annotation of multiple compounds in multiple files.
A graphical user interface as well as command line functions will assist in assessing the quality of annotation and update fitting parameters until a satisfactory result is obtained.

Depends R (>= 4.2)

Imports foreach (>= 1.4.4), doParallel (>= 1.0.11), ggplot2 (>= 3.5.0), gridExtra (>= 2.3), MSnbase (>= 2.4.0), mzR (>= 2.12.0), stringr (>= 1.2.0), methods (>= 3.4.0), XML (>= 3.98.1.10), minpack.lm (>= 1.2.1), scales(>= 0.5.0), shiny (>= 1.0.5), bslib, shinycssloaders (>= 1.0.0), DT (>= 0.15), pracma (>= 2.2.3), utils, lubridate, svglite (>= 2.1.1)

biocViews MassSpectrometry, Metabolomics, PeakDetection

License GPL-3

BugReports `https://github.com/phenomecentre/peakPantheR/issues/new`

URL `https://github.com/phenomecentre/peakPantheR`

Encoding UTF-8

LazyData false

RoxygenNote 7.3.1

Suggests testthat, devtools, faahKO, msdata, knitr, rmarkdown, pander, BiocStyle

VignetteBuilder knitr

git_url https://git.bioconductor.org/packages/peakPantheR

git_branch devel

git_last_commit 4d6e977

git_last_commit_date 2024-10-29

Repository Bioconductor 3.21

Date/Publication 2025-04-09

Author Arnaud Wolfer [aut, cre] (ORCID:
<`https://orcid.org/0000-0001-5856-3218`>),
Goncalo Correia [aut] (ORCID: <`https://orcid.org/0000-0001-8271-9294`>),
Jake Pearce [ctb],
Caroline Sands [ctb]

Maintainer Arnaud Wolfer <`adwolfer@gmail.com`>

Contents

`acquisitionTime,peakPantheRAnnotation-method`

acquisitionTime accessor returns value as.POSIXct

Description

acquisitionTime accessor returns value as.POSIXct

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
acquisitionTime(object)
```

Arguments

`object`	peakPantheRAnnotation

Value

(POSIXct) A character vector of acquisition date-time (converted from POSIXct) or NA

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                c('cpdID', 'cpdName', 'rtMin', 'rt', 'rtMax', 'mzMin', 'mz',
                'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

## acquisitionTime can only be extracted from NetCDF files
acquisitionTime(annotation)
# [1] NA NA NA
```

```
}
```

`annotationDiagnosticMultiplot`

Generate a multiplot of all diagnostic plots

Description

Generate a multiplot of all diagnostic plots (as generated by `annotationDiagnosticPlots()`) for each compound

Usage

```
annotationDiagnosticMultiplot(annotationDiagnosticPlotList)
```

Arguments

`annotationDiagnosticPlotList`	(list) List of (one per compound) of list of diagnostic plots as generated by `annotationDiagnosticPlots()`

Value

A list of multiplots (one per compound)

`annotationDiagnosticPlots,peakPantheRAnnotation-method`

Generate fit diagnostic plots

Description

Generate fit diagnostic plots for each ROI: `EICFit` the raw data and detected feature fit, `rtPeakwidthVert` detected peaks retention time apex and peakwidth (vertical and no run order), `rtPeakwidthHorzRunOrder` detected peaks retention time apex and peakwidth by run order, `mzPeakwidthHorzRunOrder` detected peaks m/z apex and peakwidth by run order, `areaRunOrder` detected peaks area by run order, `rtHistogram` histogram of detected peaks retention time, `mzHistogram` histogram of detected peaks m/z, `areaHistogram` histogram of detected peaks area.

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
annotationDiagnosticPlots(
    object,
    sampleColour,
    sampling,
    verbose
)
```

Arguments

`object`	(peakPantheRAnnotation) Annotated peakPantheRAnnotation object
`sampleColour`	(str) NULL or vector colour for each sample
`sampling`	(int) Number of points to employ when plotting fittedCurve
`verbose`	(bool) if TRUE message the plot generation progress

Value

A list (one list per compound) of diagnostic plots: `result[[i]]$EICFit`, `result[[i]]$rtPeakwidthVert`, `result[[i]]$rtPeakwidthHorzRunOrder`, `result[[i]]$mzPeakwidthHorzRunOrder`, `result[[i]]$areaRunOrder`, `result[[i]]$rtHistogram`, `result[[i]]$mzHistogram`, `result[[i]]$areaHistogram`, `result[[i]]$title`

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

emptyAnnotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                        targetFeatTable=targetFeatTable)

annotationDiagnosticPlots(emptyAnnotation)
# Warning: the object has not been annotated, return an empty diagnostic plot
# list
# [[1]]
# NULL
# [[2]]
# NULL
}
```

---

annotationParamsDiagnostic,peakPantheRAnnotation-method

Set uROI and FIR based on annotation results

Description

Set updated ROI (uROI) and Fallback Integration Regions (FIR) based on the annotation results. If the object is not annotated, it is returned untouched. ROI is not modified. If uROI exist they are left untouched, otherwise they are set as the minimum and maximum found peaks limits (+/-5% of ROI in retention time). If FIR are used they are left untouched, otherwise they are set as the median of the found limits (rtMin, rtMax, mzMin, mzMax).

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
annotationParamsDiagnostic(
    object,
    verbose
)
```

Arguments

`object`	(peakPantheRAnnotation) Annotated peakPantheRAnnotation object
`verbose`	(bool) If TRUE message progress of uROI and FIR calculation

Value

(peakPantheRAnnotation) object with updated ROI and FIR set from annotation results

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
```

```
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

emptyAnnotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                        targetFeatTable=targetFeatTable)

annotationParamsDiagnostic(emptyAnnotation, verbose=TRUE)
# Warning: the object has not been annotated, return the object untouched
# An object of class peakPantheRAnnotation
#  2 compounds in 3 samples.
#   updated ROI do not exist (uROI)
#   does not use updated ROI (uROI)
#   does not use fallback integration regions (FIR)
#   is not annotated
}
```

`annotationTable,peakPantheRAnnotation-method`
annotationTable accessor

Description

annotationTable returns a dataframe (row samples, col compounds) filled with a specific peakTable column

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
annotationTable(object, column)
```

Arguments

`object`	peakPantheRAnnotation
`column`	a peakTable columns

Value

(data.frame) (row samples, col compounds) filled with a specific peakTable column

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
```

```
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

## default values without annotation
annotationTable(annotation)
#                                              ID-1 ID-2
# C:/R/R-3.6.0/library/faahKO/cdf/KO/ko15.CDF   NA   NA
# C:/R/R-3.6.0/library/faahKO/cdf/KO/ko16.CDF   NA   NA
# C:/R/R-3.6.0/library/faahKO/cdf/KO/ko18.CDF   NA   NA
}
```

`annotation_diagnostic_multiplot_UI_helper` *UI diagnostic plot helper - single feature multiplot*

Description

Return a ggplot object of a feature diagnostic multiplot

Usage

```
annotation_diagnostic_multiplot_UI_helper(
    cpdNb,
    annotation,
    splNum = NULL,
    splColrColumn = NULL,
    ...
)
```

Arguments

`cpdNb`	(int) position of the feature to extract (1 to nbCpd)
`annotation`	(peakPantheRAnnotation) Annotation object
`splNum`	(int) NULL or number of spectra to plot, chosen randomly from all spectra. If NULL or equal to the total number of spectra, plot all spectra
`splColrColumn`	(str) NULL, None or a spectraMetadata column for colouring each sample
`...`	Additional parameters for plotting

Value

(ggplotObject) Diagnostic multiplot for a feature

Examples

```
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
spectraPaths <- c('./path/file1', './path/file2', './path/file3')

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

emptyAnnotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                        targetFeatTable=targetFeatTable)

# Plot of an empty annotation
annotation_diagnostic_multiplot_UI_helper(cpdNb = 2,
                                            annotation = emptyAnnotation,
                                            splNum = NULL,
                                            splColrColumn = NULL)
# Warning: the object has not been annotated, return an empty diagnostic
# plot list
```

---

annotation_fit_summary_UI_helper

UI diagnostic table - fit summary

Description

Return a table of fit statistic (ratio of peaks found, ratio of peaks filled, ppm error, RT deviation)

Usage

```
annotation_fit_summary_UI_helper(annot)
```

Arguments

`annot`	(peakPantheRAnnotation) Annotation object

Value

(data.frame) Fit statistics

Examples

```
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
spectraPaths <- c('./path/file1', './path/file2', './path/file3')

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

emptyAnnotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                        targetFeatTable=targetFeatTable)

# statistics of an empty annotation
annotation_fit_summary_UI_helper(emptyAnnotation)
#              Ratio peaks found (%) Ratio peaks filled (%) ppm error
# ID-1 - Cpd 1                    NA                      0       NaN
# ID-2 - Cpd 2                    NA                      0       NaN
#               RT deviation (s)
# ID-1 - Cpd 1               NaN
# ID-2 - Cpd 2               NaN
```

`annotation_showMethod_UI_helper`

UI show annotation helper - list of properties

Description

show method specific to the UI, that returns each field in a named list to ease display

Usage

```
annotation_showMethod_UI_helper(annotation)
```

Arguments

`annotation`	(peakPantherAnnotation) Object to describe

Value

(list) Named list of annotation properties

Examples

```
# Initialise an empty annotation, no uROI, no use of FIR
annotInit     <- peakPantheRAnnotation()

# return properties
annotation_showMethod_UI_helper(annotInit)
# $nbCompounds
# [1] 0
#
# #$nbSamples
# [1] 0
#
# #$uROIExist
# [1] FALSE
#
# $useUROI
# [1] FALSE
#
# $useFIR
# [1] FALSE
#
# $isAnnotated
# [1] FALSE
```

`annotation_showText_UI_helper`

UI show annotation helper - UI sidebar string

Description

Return a text description of an annotation for UI presentation

Usage

```
annotation_showText_UI_helper(annotProp)
```

Arguments

`annotProp`	(list) Named list of annotation properties as created by `annotation_showMethod_UI_helper()`

Value

(str) Textual description of the annotation to show on UI

Examples

```
# Input
properties_default <- list(nbCompounds = 0,
                            nbSamples = 0,
                            uROIExist = FALSE,
                            useUROI = FALSE,
                            useFIR = FALSE,
                            isAnnotated = FALSE)

# Generate description
annotation_showText_UI_helper(properties_default)
# [[1]]
# [1] "Not annotated"
#
# [[2]]
# [1] "0 compounds"
#
# [[3]]
# [1] "0 samples"
#
# [[4]]
# [1] "updated ROI do not exist (uROI)"
#
# [[5]]
# [1] "does not use updated ROI (uROI)"
#
# [[6]]
# [1] "does not use fallback integration regions (FIR)"
```

`cpdID,peakPantheRAnnotation-method`
cpdID accessor

Description

cpdID accessor

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
cpdID(object)
```

Arguments

`object`	peakPantheRAnnotation

Value

(str) A character vector of compound IDs, of length number of compounds

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                        c('cpdID','cpdName','rtMin','rt','rtMax','mzMin',
                        'mz', 'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

cpdID(annotation)
# [1] 'ID-1' 'ID-2'
}
```

`cpdMetadata,peakPantheRAnnotation-method`

cpdMetadata accessor

Description

cpdMetadata accessor

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
cpdMetadata(object)
```

Arguments

`object`	peakPantheRAnnotation

Value

(data.frame) A data.frame of compound metadata, with compounds as row and metadata as columns

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

## default values not initialised
cpdMetadata(annotation)
# data frame with 0 columns and 2 rows
}
```

`cpdName,peakPantheRAnnotation-method`

cpdName accessor

Description

cpdName accessor

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
cpdName(object)
```

Arguments

`object`	peakPantheRAnnotation

Value

(str) A character vector of compound names, of length number of compounds

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                c('cpdID', 'cpdName', 'rtMin', 'rt', 'rtMax', 'mzMin', 'mz',
                'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

cpdName(annotation)
# [1] 'Cpd 1' 'Cpd 2'
}
```

`dataPoints,peakPantheRAnnotation-method`
dataPoints accessor

Description

dataPoints accessor

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
dataPoints(object)
```

Arguments

`object`	peakPantheRAnnotation

Value

A list of length number of spectra files. Each list element is a *ROIsDataPoint* list of `data.frame` of raw data points for each ROI/uROI (retention time 'rt', mass 'mz' and intensity 'int' (as column) of each raw data points (as row))

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

## default values without annotation
dataPoints(annotation)
# [[1]]
# NULL
# [[2]]
# NULL
# [[3]]
# NULL
}
```

EICs,peakPantheRAnnotation-method

EICs accessor

Description

EICs accessor

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
EICs(object, aggregationFunction)
```

Arguments

`object`	peakPantheRAnnotation
`aggregationFunction`	(str) Function to use in order to aggregate intensities across mz in each scan. One of `sum`, `max`, `min`, `mean`

Value

(float) Extracted Ion Chromatogram aggregated across mz in each scan

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

## default values without annotation
EICs(annotation)
# [[1]]
# list()
# [[2]]
# list()
# [[3]]
# list()
}
```

`emgGaussian_guess` *Guess function for initial exponentially modified gaussian parameters and bounds*

Description

Guess function for initial exponentially modified gaussian parameters and bounds, at the moment only checks the x position

Usage

```
emgGaussian_guess(x, y)
```

Arguments

`x`	(numeric) x values (e.g. retention time)
`y`	(numeric) y observed values (e.g. spectra intensity)

Value

A list of guessed starting parameters `list()$init_params`, lower `list()$lower_bounds` and upper bounds `list()$upper_bounds` (`$gamma`, `$center`, `$sigma`, `$amplitude`)

`emgGaussian_minpack.lm` *Implementation of the Exponentially Modified Gaussian (EMG) peak shape for use with minpack.lm*

Description

Implementation of the Exponentially Modified Gaussian (EMG) peak shape for use with minpack.lm

Usage

```
emgGaussian_minpack.lm(params, xx)
```

Arguments

`params`	(list) exponentiall modified gaussian parameters (`params$gamma`, `params$center`, `params$sigma`, `params$amplitude`)
`xx`	(numeric) values at which to evalute the exponentially modified gaussian

Value

value of the exponentially modified gaussian evaluated at xx

`emgGaussian_minpack.lm_objectiveFun` *Exponentially Modified Gaussian minpack.lm objective function*

Description

Exponentially Modified Gaussian (EMG) minpack.lm objective function, calculates residuals using the EMG Peak Shape

Usage

```
emgGaussian_minpack.lm_objectiveFun(params, observed, xx)
```

Arguments

`params`	(list) exponentially modified gaussian parameters (`params$gamma`, `params$center`, `params$sigma`, `params$amplitude`)
`observed`	(numeric) observed y value at xx
`xx`	(numeric) value at which to evalute the exponentially modified gaussian

Value

difference between observed and expected exponentially modified gaussian value evaluated at xx

---

`extractSignalRawData` *Extract signal in a multiple defined mz rt window from a raw data file*

Description

Extract all signal from multiple defined mz rt window from raw data and returns a data.frame. If no rt-mz window is provided, all signal in the raw data file are returned

Usage

```
extractSignalRawData(rawSpec, rt, mz, msLevel = 1L, verbose = TRUE)
```

Arguments

`rawSpec`	an `OnDiskMSnExp-class`
`rt`	(numeric(2) or two-column matrix) the lower and upper retention time range from which the data should be extracted. If a matrix is passed, each row corresponds to a different window. If not provided, the full retention time range will be extracted.

`mz`	(numeric(2) or two-column matrix) the lower and upper mass range from which the data should be extracted. If a matrix is passed, each row corresponds to a different window. If not provided, the full mass range will be extracted.
`msLevel`	(int) the MS level at which the data should be extracted (default to MS level 1)
`verbose`	(bool) If TRUE message progress and warnings

Details

Examples cannot be computed as the function is not exported: ## Use a file form the faahKO package and extract data from a region of ## interest library(faahKO) rawSpec <- MSnbase::readMSData(system.file(’cdf/KO/ko15.CDF’,package=’faahKO’), centroided=TRUE, mode=’onDisk’) dataPoints <- extractSignalRawData(rawSpec, rt = c(3290., 3410.), mz = c(522.194778, 522.205222), verbose=TRUE) # Reading data from 1 windows

dataPoints # [[1]] # rt mz int # 1 3290.115 522.2 1824 # 2 3291.680 522.2 1734 # 3 3293.245 522.2 1572 # 4 3294.809 522.2 1440 # 5 3299.504 522.2 1008 # 6 3301.069 522.2 871 # 7 3302.634 522.2 786 # 8 3304.199 522.2 802 # 9 3305.764 522.2 834 # 10 3307.329 522.2 839 # 11 3315.154 522.2 2187 # 12 3316.719 522.2 3534 # 13 3318.284 522.2 6338 # 14 3319.849 522.2 11718 # 15 3321.414 522.2 21744 # 16 3322.979 522.2 37872 # 17 3324.544 522.2 62424 # 18 3326.109 522.2 98408 # 19 3327.673 522.2 152896 # 20 3329.238 522.2 225984 # 21 3330.803 522.2 308672 # 22 3332.368 522.2 399360 # 23 3333.933 522.2 504000 # 24 3335.498 522.2 614656 # 25 3337.063 522.2 711872 # 26 3338.628 522.2 784704 # 27 3340.193 522.2 836608 # 28 3341.758 522.2 866304 # 29 3343.323 522.2 882304 # 30 3344.888 522.2 889280 # 31 3346.453 522.2 888256 # 32 3348.018 522.2 866816 # 33 3349.583 522.2 827392 # 34 3351.148 522.2 777728 # 35 3352.713 522.2 727040 # 36 3354.278 522.2 678464 # 37 3355.843 522.2 629120 # 38 3357.408 522.2 578048 # 39 3358.973 522.2 524288 # 40 3360.538 522.2 471040 # 41 3362.102 522.2 416320 # 42 3363.667 522.2 360064 # 43 3365.232 522.2 302400 # 44 3366.797 522.2 249152 # 45 3368.362 522.2 202560 # 46 3369.927 522.2 161024 # 47 3371.492 522.2 123520 # 48 3373.057 522.2 93160 # 49 3374.622 522.2 71856 # 50 3376.187 522.2 58392 # 51 3377.752 522.2 51072 # 52 3379.317 522.2 48376 # 53 3380.882 522.2 49168 # 54 3382.447 522.2 53120 # 55 3384.012 522.2 62488 # 56 3385.577 522.2 78680 # 57 3387.142 522.2 102840 # 58 3388.707 522.2 134656 # 59 3390.272 522.2 173440 # 60 3391.837 522.2 217088 # 61 3393.402 522.2 268864 # 62 3394.966 522.2 330496 # 63 3396.531 522.2 395776 # 64 3398.096 522.2 453376 # 65 3399.661 522.2 499072 # 66 3401.226 522.2 537024 # 67 3402.791 522.2 570304 # 68 3404.356 522.2 592512 # 69 3405.921 522.2 598912 # 70 3407.486 522.2 595008 # 71 3409.051 522.2 588416

Value

a list (one entry per window) of data.frame with signal as row and retention time (’rt’), mass (’mz’) and intensity (’int) as columns.

`filename,peakPantheRAnnotation-method`

filename accessor by spliting filepath

Description

filename accessor by spliting filepath

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
filename(object)
```

Arguments

`object`	peakPantheRAnnotation

Value

(str) filename

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

filename(annotation)
# [1] 'ko15' 'ko16' 'ko18'
}
```

`filepath,peakPantheRAnnotation-method`
filepath accessor

Description

filepath accessor

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
filepath(object)
```

Arguments

`object`	peakPantheRAnnotation

Value

(str) A character vector of file paths, of length number of spectra files

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                c('cpdID', 'cpdName', 'rtMin', 'rt', 'rtMax', 'mzMin', 'mz',
                'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)
filepath(annotation)
# [1] 'C:/R/R-3.6.0/library/faahKO/cdf/KO/ko15.CDF'
# [2] 'C:/R/R-3.6.0/library/faahKO/cdf/KO/ko16.CDF'
# [3] 'C:/R/R-3.6.0/library/faahKO/cdf/KO/ko18.CDF'
}
```

`findTargetFeatures` *Find and integrate target features in each ROI*

Description

For each ROI, fit a curve and integrate the largest feature in the box. Each entry in `ROIsDataPoints` must match the corresponding row in `ROI`. The curve shape to employ for fitting can be changed with `curveModel` while fitting parameters can be changed with `params` (list with one param per ROI window). `rtMin` and `rtMax` are established at 0.5 outward (the window is the ROI width); if after 8 iterations `rtMin` or `rtMax` is not found, NA is returned and the peak fit rejected. `peakArea` is calculated from `rtMin` to `rtMax`. `peakAreaRaw` is calculated from `rtMin` to `rtMax` but using the raw data points instead of the modelled line-shape. `mz` is the weighted (by intensity) average mz of datapoints falling into the `rtMin` to `rtMax` range, `mzMin` and `mzMax` are the minimum and maxmimum mass in these range. If `rtMin` or `rtMax` falls outside of ROI (extracted scans), `mzMin` or `mzMax` are returned as the input ROI limits and `mz` is an approximation on the datapoints available (if no scan of the ROI fall between rtMin/rtMax, mz would be NA, the peak is rejected). If any of the two following ratio are superior to `maxApexResidualRatio`, the fit is rejected: 1) ratio of fit residuals at the apex (predicted apex fit intensity vs measured apex intensity: fit overshoots the apex), 2) ratio of predicted apex fit intensity vs maximum measured peak intensity (fit misses the real apex in the peak).

Usage

```
findTargetFeatures(
    ROIsDataPoints,
    ROI,
    curveModel = "skewedGaussian",
    params = "guess",
    sampling = 250,
    maxApexResidualRatio = 0.2,
    verbose = FALSE,
    ...
)
```

Arguments

`ROIsDataPoints`	(list) A list (one entry per ROI window) of data.frame with signal as row and retention time ('rt'), mass ('mz') and intensity ('int) as columns. Must match each row of ROI.
`ROI`	(data.frame) A data.frame of compounds to target as rows. Columns: `rtMin` (float in seconds), `rtMax` (float in seconds), `mzMin` (float), `mzMax` (float)
`curveModel`	(str) Name of the curve model to fit (currently `skewedGaussian` and `emgGaussian`)
`params`	(list or str) Either 'guess' for automated parametrisation or list (one per ROI windows) of 'guess' or list of curve fit parameters
`sampling`	(int) Number of points to employ when subsampling the fittedCurve (rt, rtMin, rtMax, integral calculation)
`maxApexResidualRatio`	(float) Ratio of maximum allowed fit residual at the peak apex, compared to the fit max intensity. (e.g. 0.2 for a maximum residual of 20% of apex intensity)
`verbose`	(bool) If TRUE message the time taken and number of features found
`...`	Passes arguments to `fitCurve` to alter peak fitting (`params`)

Details

Examples cannot be computed as the function is not exported: ## Load data library(faahKO) library(MSnbase) netcdfFilePath <- system.file('cdf/KO/ko15.CDF', package = 'faahKO') raw_data <- MSnbase::readMSData(netcdfFilePath,centroided=TRUE,mode='onDisk')

targetFeatTable targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(), c('cpdID','cpdName','rtMin','rt','
'mz','mzMax'))), stringsAsFactors=FALSE) targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778, 522.2, 522.205222) targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038, 496.2, 496.204962) targetFeatTable[,3:8] <- vapply(targetFeatTable[,3:8], as.numeric, FUN.VALUE=numeric(2))

ROIsPt <- extractSignalRawData(raw_data, rt=targetFeatTable[,c('rtMin','rtMax')], mz=targetFeatTable[,c('mzMin','mzMa
verbose=TRUE) # Reading data from 2 windows

foundPeaks <- findTargetFeatures(ROIsPt, targetFeatTable, verbose=TRUE) # Warning: rtMin/rtMax outside of ROI; datapoints cannot be used for # mzMin/mzMax calculation, # approximate mz and returning ROI$mzMin and ROI$mzMax for ROI #1 # Found 2/2 features in 0.07 secs

foundPeaks # $peakTable # found rtMin rt rtMax mzMin mz mzMax peakArea # 1 TRUE 3309.759 3346.828 3385.410 522.1948 522.2 522.2052 26133727 # 2 TRUE 3345.377 3386.529 3428.279 496.2000 496.2 496.2000 35472141 # peakAreaRaw maxIntMeasured maxIntPredicted # 1 26071378 889280 901015.8 # 2 36498367 1128960 1113576.7 # # $curveFit # $curveFit[[1]] # $amplitude # [1] 162404.8 # # $center # [1] 3341.888 # # $sigma # [1] 0.07878613 # # $gamma # [1] 0.00183361 # # $fitStatus # [1] 2 # # $curveModel # [1] 'skewedGaussian' # # attr(,'class') # [1] 'peakPantheR_curveFit' # # $curveFit[[2]] # $amplitude # [1] 199249.1 # # $center # [1] 3382.577 # # $sigma # [1] 0.07490442 # # $gamma # [1] 0.00114719 # # $fitStatus # [1] 2 # # $curveModel # [1] 'skewedGaussian' # # attr(,'class') # [1] 'peakPantheR_curveFit'

Value

A list: `list()$peakTable` (*data.frame*) with targeted features as rows and peak measures as columns (see Details), `list()$curveFit` (*list*) a list of `peakPantheR_curveFit` or NA for each ROI.

Details:: The returned `data.frame` is structured as follow:

found	was the peak found
rt	retention time of peak apex (sec)
rtMin	leading edge of peak retention time (sec) determined at 0.5% of apex intensity
rtMax	trailing edge of peak retention time (sec) determined at 0.5% of apex intensity
mz	weighted (by intensity) mean of peak m/z across scans
mzMin	m/z peak minimum (between rtMin, rtMax)
mzMax	m/z peak maximum (between rtMin, rtMax)
peakArea	integrated peak area
peakAreaRaw	integrated peak area from raw data points
maxIntMeasured	maximum peak intensity in raw data
maxIntPredicted	maximum peak intensity based on curve fit (at apex)

`FIR,peakPantheRAnnotation-method`

FIR accessor returns targetFeatTable with cpdID, cpdName added

---

Description

FIR accessor returns targetFeatTable with cpdID, cpdName added

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
FIR(object)
```

Arguments

`object`	peakPantheRAnnotation

Value

(data.frame) target feature table with compounds as row and FIR parameters as columns

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                c('cpdID', 'cpdName', 'rtMin', 'rt', 'rtMax', 'mzMin', 'mz',
                'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

## default values without annotation
FIR(annotation)
#   rtMin rtMax mzMin mzMax cpdID cpdName
```

```
# 1    NA    NA    NA    NA  ID-1   Cpd 1
# 2    NA    NA    NA    NA  ID-2   Cpd 2
}
```

`fitCurve` *Curve fitting using minpack.lm*

Description

Fit different curve models using `minpack`. Fitting parameters can be passed or guessed.

Usage

```
fitCurve(x, y, curveModel = "skewedGaussian", params = "guess")
```

Arguments

`x`	(numeric) x values (e.g. retention time)
`y`	(numeric) y observed values (e.g. spectra intensity)
`curveModel`	(str) name of the curve model to fit (currently `skewedGaussian` and `emgGaussian`)
`params`	(list or str) either 'guess' for automated parametrisation or list of initial parameters (`$init_params`), lower parameter bounds (`$lower_bounds`) and upper parameter bounds (`$upper_bounds`)

Details

Examples cannot be computed as the function is not exported: ## x is retention time, y corresponding intensity input_x <- c(3362.102, 3363.667, 3365.232, 3366.797, 3368.362, 3369.927, 3371.492, 3373.057, 3374.622, 3376.187, 3377.752, 3379.317, 3380.882, 3382.447, 3384.012, 3385.577, 3387.142, 3388.707, 3390.272, 3391.837, 3393.402, 3394.966, 3396.531, 3398.096, 3399.661, 3401.226, 3402.791, 3404.356, 3405.921, 3407.486, 3409.051) input_y <- c(51048, 81568, 138288, 233920, 376448, 557288, 753216, 938048, 1091840, 1196992, 1261056, 1308992, 1362752, 1406592, 1431360, 1432896, 1407808, 1345344, 1268480, 1198592, 1126848, 1036544, 937600, 849792, 771456, 692416, 614528, 546088, 492752, 446464, 400632)

Fit fitted_curve <- fitCurve(input_x, input_y, curveModel='skewedGaussian', params='guess')

Returns the optimal fitting parameters fitted_curve # # $amplitude # [1] 275371.1 # # $center # [1] 3382.577 # # $sigma # [1] 0.07904697 # # $gamma # [1] 0.001147647 # # $fitStatus # [1] 2 # # $curveModel # [1] 'skewedGaussian' # # attr(,'class') # [1] 'peakPantheR_curveFit'

Value

A 'peakPantheR_curveFit': a list of fitted curve parameters, `fitStatus` from `nls.lm$info` and curve shape name `curveModel`. `fitStatus=0` unsuccessful completion: improper input parameters, `fitStatus=1` successful completion: first convergence test is successful, `fitStatus=2` successful completion: second convergence test is successful, `fitStatus=3` successful completion: both convergence test are successful, `fitStatus=4` questionable completion: third convergence test is successful but should be carefully examined (maximizers and saddle points might satisfy), `fitStatus=5` unsuccessful completion: excessive number of function evaluations/iterations

---

`gaussian_cerf` *Gaussian Error function*

Description

Implementation of the gaussian error function

Usage

```
gaussian_cerf(x)
```

Arguments

`x`	(numeric) value at which to evaluate the gaussian error function

Value

Value of the gaussian error function evaluated at x

`gaussian_erf` *Gaussian Error function*

Description

Implementation of the gaussian error function

Usage

```
gaussian_erf(x)
```

Arguments

`x`	(numeric) value at which to evaluate the gaussian error function

Value

Value of the gaussian error function evaluated at x

`generateIonChromatogram`

Generate ion chromatogram from raw data points

Description

On the input raw data, aggregate intensites across the mz range at each retention time to generate an ion chromatogram: `sum` for EIC/TIC, `max`, `min` or `mean`. The number of data points returned correspond to the number of unique scans/retention time measurements in the input data

Usage

```
generateIonChromatogram(ROIDataPoint, aggregationFunction = "sum")
```

Arguments

`ROIDataPoint`	(data.frame) retention time 'rt', mass 'mz' and intensity 'int' (as column) of each raw data points (as row) to use for the ion chromatogram
`aggregationFunction`	(str) Function to use in order to aggregate intensities across mz in each scan. One of `sum`, `max`, `min`, `mean`

Details

Examples cannot be computed as the function is not exported: ## Input data points in_rt <- c(3362.102, 3362.102, 3363.667, 3363.667, 3365.232, 3365.232, 3366.797, 3366.797, 3368.362, 3368.362) in_mz <- c(496.2, 497.2, 496.2, 497.2, 496.2, 497.2, 496.2, 497.2, 496.2, 497.2) in_int <- c(39616, 11432, 63344, 18224, 107352, 30936, 182144, 51776, 295232, 81216) input_ROIDataPoints <- data.frame(rt=in_rt, mz=in_mz, int=in_int)

Aggregate mz to generate EIC EIC <- generateIonChromatogram(input_ROIDataPoints,aggregationFunction='sum') EIC # rt int # 1 3362.102 51048 # 2 3363.667 81568 # 3 3365.232 138288 # 4 3366.797 233920 # 5 3368.362 376448

Value

A data.frame of retention time 'rt' and aggregated intensities 'int'

`getAcquisitionDatemzML` *Parse acquisition date from a mzML file*

Description

Extract acquisition date (‘’startTimeStamp’“) from a mzML file. In case of failure (or the file is not a `mzML`) returns NULL

Usage

```
getAcquisitionDatemzML(mzMLPath, verbose = TRUE)
```

Arguments

`mzMLPath`	(str) path to mzML raw data file
`verbose`	(bool) if TRUE message progress

Value

POSIXct or NA

`getTargetFeatureStatistic` *Calculate chromatographic peak properties*

Description

Calculate the ppm error, retention time deviation, tailing factor and asymmetry factor for each measured feature.

Usage

```
getTargetFeatureStatistic(
    fittedCurves,
    targetFeatTable,
    foundPeakTable,
    verbose = FALSE
)
```

Arguments

`fittedCurves`	(list) A list (one entry per ROI window) of `peakPantheR_curveFit` or NA
`targetFeatTable`	a `data.frame` of compounds to target as rows. Columns: `cpdID` (str), `cpdName` (str), `rtMin` (float in seconds), `rt` (float in seconds, or *NA*), `rtMax` (float in seconds), `mzMin` (float), `mz` (float or *NA*), `mzMax` (float).
`foundPeakTable`	a `data.frame` as generated by `findTargetFeatures`, with features as rows and peak properties as columns. The following columns are mandatory: `mzMin`, `mz`, `mzMax`, `rtMin`, `rt`, `rtMax`
`verbose`	(bool) if TRUE message when NA scans are removed

Details

Examples cannot be computed as the function is not exported: # fittedCurve cFit1 <- list(amplitude=162404.8057918259, center=3341.888, sigma=0.078786133031045896, gamma=0.0018336101984172684, fitStatus=2, curveModel='skewedGaussian') class(cFit1) <- 'peakPantheR_curveFit' cFit2 <- list(amplitude=199249.10572753669, center=3382.577, sigma=0.074904415304607966, gamma=0.0011471899372353885, fitStatus=2, curveModel='skewedGaussian') class(cFit2) <- 'peakPantheR_curveFit' input_fitCurves <- list(cFit1, cFit2)

ROI input_ROI <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(), c('cpdID', 'cpdName', 'rtMin', 'rt', 'rtMax', 'mzMin', 'mz', 'mzMax'))), stringsAsFactors=FALSE) input_ROI[1,] <- c('ID-1', 'testCpd 1', 3310., 3344.88, 3390., 522.19, 522.2, 522.21) input_ROI[2,] <- c('ID-2', 'testCpd 2', 3280., 3385.58, 3440., 496.19, 496.2, 496.21) input_ROI[,3:8] <- vapply(input_ROI[,3:8], as.numeric, FUN.VALUE=numeric(2))

foundPeakTable input_foundPeakTable <- data.frame(matrix(vector(), 2, 10, dimnames=list(c(), c('found', 'rtMin', 'rt', 'rtMax', 'mzMin', 'mz', 'mzMax','peakArea','maxIntMeasured','maxIntPredicted'))), stringsAsFactors=FALSE) input_foundPeakTable[1,] <- c(TRUE, 3309.758, 3346.827, 3385.410, 522.19, 522.2, 522.21, 26133726, 889280, 901015) input_foundPeakTable[2,] <- c(TRUE, 3345.376, 3386.529, 3428.279, 496.19, 496.2, 496.21, 35472141, 1128960, 1113576) input_foundPeakTable[,1] <- vapply(input_foundPeakTable[,c(1)], as.logical, FUN.VALUE=logical(1))

Run peak statistics peakStatistics <- getTargetFeatureStatistic(input_fitCurves, input_ROI, input_foundPeakTable) peakStatistics # found rtMin rt rtMax mzMin mz mzMax peakArea # 1 TRUE 3309.758 3346.827 3385.410 522.19 522.2 522.21 26133726 # 2 TRUE 3345.376 3386.529 3428.279 496.19 496.2 496.21 35472141 # maxIntMeasured maxIntPredicted ppm_error rt_dev_sec tailingFactor # 1 889280 901015 0 1.947 1.015385 # 2 1128960 1113576 0 0.949 1.005372 # asymmetryFactor # 1 1.026886 # 2 1.009304

Value

A `data.frame` with measured compounds as rows and measurements and properties as columns (see Details).

Details:: The returned `data.frame` is structured as follow:

found	was the peak found
rt	retention time of peak apex (sec)
rtMin	leading edge of peak retention time (sec) determined at 0.5% of apex intensity

rtMax	trailing edge of peak retention time (sec) determined at 0.5% of apex intensity
mz	weighted (by intensity) mean of peak m/z across scans
mzMin	m/z peak minimum (between rtMin, rtMax)
mzMax	m/z peak maximum (between rtMin, rtMax)
peakArea	integrated peak area
peakAreaRaw	integrated peak area from raw data points
maxIntMeasured	maximum peak intensity in raw data
maxIntPredicted	maximum peak intensity based on curve fit
ppm_error	difference in ppm between the expected and measured m/z
rt_dev_sec	difference in seconds between the expected and measured rt
tailingFactor	the tailing factor is a measure of peak tailing. It is defined as the distance from the front slope of the peak
asymmetryFactor	the asymmetry factor is a measure of peak tailing. It is defined as the distance from the center line of the

---

`initialise_annotation_from_files_UI_helper`

UI data import helper - initialise new annotation from files

Description

Fully initialise a `peakPantheRAnnotation` using the target files path, CSV parameter path and metadata.

Usage

```
initialise_annotation_from_files_UI_helper(
    CSVParamPath,
    spectraPaths,
    cpdMetadataPath = NULL,
    spectraMetadata = NULL,
    verbose = TRUE
)
```

Arguments

`CSVParamPath`	(str) Path to a CSV file of fit parameters
`spectraPaths`	(str) character vector of spectra file paths, to set samples to process
`cpdMetadataPath`	NULL or path to a csv of compound metadata, with compounds as row and metadata as columns
`spectraMetadata`	NULL or DataFrame of sample metadata, with samples as row and metadata as columns
`verbose`	(bool) If TRUE message progress

Value

(peakPantheRAnnotation) Object initialised with ROI, uROI and FIR read from the CSV file

Examples

```
## Input data
input_CSV <- data.frame(matrix(nrow=2,ncol=21,dimnames=list(c(),
    c('cpdID', 'cpdName',
    'X','ROI_rt', 'ROI_mz', 'ROI_rtMin', 'ROI_rtMax','ROI_mzMin','ROI_mzMax',
    'X','uROI_rtMin', 'uROI_rtMax', 'uROI_mzMin', 'uROI_mzMax', 'uROI_rt',
    'uROI_mz', 'X', 'FIR_rtMin', 'FIR_rtMax', 'FIR_mzMin', 'FIR_mzMax'))))
input_CSV[1,]  <- c('ID-1', 'Cpd 1', '|', 1.,  2.,  3.,  4.,  5.,  6.,  '|',
                    7.,  8.,  9.,  10., 11., 12., '|', 13., 14., 15., 16.)
input_CSV[2,]  <- c('ID-2', 'Cpd 2', '|', 17., 18., 19., 20., 21., 22., '|',
                    23., 24., 25., 26., 27., 28., '|', 29., 30., 31., 32.)
input_CSV[,-c(1,2,3,10,17)]  <- vapply(input_CSV[,-c(1,2,3,10,17)],
                                        as.numeric, FUN.VALUE=numeric(2))

input_spectraPaths    <- c('./path/file1', './path/file2', './path/file3')

# temporary file location
savePath1      <- tempfile(pattern='file', tmpdir=tempdir(), fileext='.csv')
# save csv
utils::write.csv(input_CSV, file=savePath1, row.names=FALSE)

# Load parameters from CSV
loadedAnnotation <- initialise_annotation_from_files_UI_helper(savePath1,
                                                    input_spectraPaths,
                                                    verbose=TRUE)
# An object of class peakPantheRAnnotation
#   2 compounds in 3 samples.
#   updated ROI exist (uROI)
#   does not use updated ROI (uROI)
#   does not use fallback integration regions (FIR)
#   is not annotated
```

`integrateFIR` *Integrate fallback integration regions*

Description

Integrate region defined in FIR if a feature is not found

Usage

```
integrateFIR(rawSpec, FIR, foundPeakTable, verbose = TRUE)
```

Arguments

`rawSpec`	an `OnDiskMSnExp-class`
`FIR`	(data.frame) Fallback Integration Regions (FIR) to integrate when a feature is not found. Compounds as row are identical to the targeted features, columns are `rtMin` (float in seconds), `rtMax` (float in seconds), `mzMin` (float), `mzMax` (float)
`foundPeakTable`	a `data.frame` as generated by findTargetFeatures, with features as rows and peak properties as columns. The following columns are mandatory: `found`, `is_filled`, `mz`, `mzMin`, `mzMax`, `rt`, `rtMin`, `rtMax`, `peakArea`, `peakAreaRaw`, `maxIntMeasured`, `maxIntPredicted`.
`verbose`	(bool) if TRUE message progress

Value

an updated foundPeakTable with FIR integration values

`is.peakPantheR_curveFit`

Check if object is of class peakPantheR_curveFit

Description

Check if object is of class peakPantheR_curveFit

Usage

```
is.peakPantheR_curveFit(x)
```

Arguments

`x`	object to test

Value

(bool) TRUE or FALSE

isAnnotated,peakPantheRAnnotation-method

isAnnotated accessor

Description

isAnnotated accessor

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
isAnnotated(object)
```

Arguments

`object`	peakPantheRAnnotation

Value

(bool) flag if the annotation has taken place

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

isAnnotated(annotation)
# [1] FALSE
}
```

`load_annotation_from_file_UI_helper` *UI data import helper - check loaded annotation*

Description

Load a .RData file (check it exists) and that a peakPantheRAnnotation named "annotationObject" is present. Returns the annotation if everything is valid

Usage

```
load_annotation_from_file_UI_helper(annotationPath)
```

Arguments

`annotationPath`	(str) Path to a RData file containing a peakPantheRAnnotation names ‘annotationObject‘

Value

(peakPantheRAnnotation) Object loaded from file

Examples

```
## Initialise a peakPantheRAnnotation object with 3 samples and 2 compounds

## Inputs
# spectraPaths
spectraPaths <- c('./path/file1', './path/file2', './path/file3')

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotationObject <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                        targetFeatTable=targetFeatTable)

# save annotation to disk
annotPath <- tempfile(pattern="file", tmpdir=tempdir(), fileext='.RData')
save(annotationObject, file=annotPath, compress=TRUE)

# Load annotation
```

```
load_annotation_from_file_UI_helper(annotationPath = annotPath)
# An object of class peakPantheRAnnotation
#   2 compounds in 3 samples.
#   updated ROI do not exist (uROI)
#   does not use updated ROI (uROI)
#   does not use fallback integration regions (FIR)
#   is not annotated
```

`nbCompounds,peakPantheRAnnotation-method`

nbCompounds accessor established on cpdID

Description

nbCompounds accessor established on cpdID

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
nbCompounds(object)
```

Arguments

`object`	peakPantheRAnnotation

Value

(int) number of samples

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
##  compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
```

```
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

nbCompounds(annotation)
# [1] 2
}
```

`nbSamples,peakPantheRAnnotation-method`

nbSamples accessor established on filepath

Description

nbSamples accessor established on filepath

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
nbSamples(object)
```

Arguments

`object`	peakPantheRAnnotation

Value

(int) number of samples

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
```

```
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

nbSamples(annotation)
# [1] 3
}
```

`outputAnnotationDiagnostic,peakPantheRAnnotation-method` *Save to disk the annotation parameters as CSV and a diagnostic plot per fitted compound*

Description

Save to disk the annotation parameters as CSV (as generated by `outputAnnotationParamsCSV()`) and a diagnostic plot per fitted compound (as generated by `annotationDiagnosticMultiplot()`) if `savePlots` is TRUE

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
outputAnnotationDiagnostic(
    object,
    saveFolder,
    savePlots = TRUE,
    sampleColour = NULL,
    verbose = TRUE,
    ncores = 0,
    svgPlot = FALSE,
    ...
)
```

Arguments

`object`	(peakPantheRAnnotation) Annotated peakPantheRAnnotation object
`saveFolder`	(str) Path of folder where annotationParameters_summary.csv and plots will be saved
`savePlots`	(bool) If TRUE save a diagnostic plot for each compound
`sampleColour`	(str) NULL or vector colour for each sample
`verbose`	(bool) If TRUE message progress
`ncores`	(int) Number of cores to use to save plots in parallel
`svgPlot`	(bool) If TRUE save plots as 'svg', otherwise as 'png'
`...`	Additional parameters for plotting i.e. `sampling` for the number of points to employ when plotting fittedCurve

Value

None

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                FUN.VALUE=numeric(2))

emptyAnnotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

# Calculate annotation
annotation <- peakPantheR_parallelAnnotation(emptyAnnotation, ncores=0,
                                getAcquTime=FALSE, verbose=FALSE)$annotation

# temporary location
savePath1       <- tempdir()
outputAnnotationDiagnostic(annotation, saveFolder=savePath1, savePlots=FALSE,
                            verbose=TRUE)
}
```

`outputAnnotationFeatureMetadata_UI_helper`

UI export helper - feature metadata

Description

Return a table with features as rows and all feature metadata as columns

Usage

```
outputAnnotationFeatureMetadata_UI_helper(annot)
```

Arguments

`annot`	(peakPantheRAnnotation) Annotation object

Value

(data.frame) Features metadata

Examples

```
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
spectraPaths <- c('./path/file1', './path/file2', './path/file3')

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

emptyAnnotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                        targetFeatTable=targetFeatTable)

# featureMetadata of an empty annotation
outputAnnotationFeatureMetadata_UI_helper(emptyAnnotation)
# data frame with 0 columns and 2 rows
```

`outputAnnotationParamsCSV,peakPantheRAnnotation-method` *Save annotation parameters as CSV*

Description

Save annotation parameters (ROI, uROI and FIR) to disk as a CSV file for editing

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
outputAnnotationParamsCSV(
    object,
    saveFolder,
    verbose,
    noSave
)
```

Arguments

`object`	(peakPantheRAnnotation) Annotated peakPantheRAnnotation object
`saveFolder`	(str) Path of folder where annotationParameters_summary.csv will be saved
`verbose`	(bool) If TRUE message progress
`noSave`	(bool) If TRUE the resulting table will be returned without saving to disk

Value

None

Examples

```
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
spectraPaths <- c('./path/file1', './path/file2', './path/file3')

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

emptyAnnotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                        targetFeatTable=targetFeatTable)

# temporary file
savePath        <- tempdir()

# statistics of an empty annotation
outputAnnotationParamsCSV(emptyAnnotation, saveFolder=savePath, verbose=TRUE)
```

`outputAnnotationResult,peakPantheRAnnotation-method`

Save to disk all annotation results as csv files

Description

Save to disk all annotation results as `annotationName_ ... .csv` files: compound metadata (`cpdMetadata`, `cpdID`, `cpdName`) and spectra metadata (`spectraMetadata`, `acquisitionTime`, `TIC`), summary of fit (ratio of peaks found: `ratio_peaks_found`, ratio of peaks filled: `ratio_peaks_filled`, mean ppm_error: `ppm_error`, mean rt_dev_sec: `rt_dev_sec`), and a file for each column of `peakTables` (with samples as rows and compounds as columns)

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
outputAnnotationResult(
    object,
    saveFolder,
    annotationName = "annotationResult",
    verbose = TRUE
)
```

Arguments

`object`	(peakPantheRAnnotation) Annotated peakPantheRAnnotation object
`saveFolder`	(str) Path of folder where the annotation result csv will be saved
`annotationName`	(str) name of annotation to use in the saved csv
`verbose`	(bool) If TRUE message progress

Value

None

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

emptyAnnotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                        targetFeatTable=targetFeatTable)

# Calculate annotation
annotation      <- peakPantheR_parallelAnnotation(emptyAnnotation, ncores=0,
                                getAcquTime=FALSE, verbose=FALSE)$annotation
```

```
# temporary location
savePath1       <- tempdir()
outputAnnotationResult(annotation, saveFolder=savePath1,
                        annotationName='testProject', verbose=TRUE)
}
```

---

outputAnnotationSpectraMetadata_UI_helper

UI export helper - spectra path and metadata

Description

Return a table with spectra as rows and filepath and all spectra metadata columns

Usage

```
outputAnnotationSpectraMetadata_UI_helper(annot)
```

Arguments

`annot`	(peakPantheRAnnotation) Annotation object

Value

(data.frame) Spectra paths and metadata

Examples

```
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
spectraPaths <- c('./path/file1', './path/file2', './path/file3')

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

emptyAnnotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                        targetFeatTable=targetFeatTable)

# spectraMetada of an empty annotation
outputAnnotationSpectraMetadata_UI_helper(emptyAnnotation)
```

```
#       filepath
# 1 ./path/file1
# 2 ./path/file2
# 3 ./path/file3
```

`peakFit,peakPantheRAnnotation-method` *peakFit accessor*

Description

peakFit accessor

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
peakFit(object)
```

Arguments

`object`	peakPantheRAnnotation

Value

A list of length number of spectra files. Each list element is a *curveFit* list of `peakPantheR_curveFit` or NA for each ROI

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))
```

```
annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

## default values without annotation
peakFit(annotation)
# [[1]]
# NULL
# [[2]]
# NULL
# [[3]]
# NULL
}
```

`peakPantheR` *peakPantheR: A package for Peak Picking and ANnoTation of High resolution Experiments*

Description

peakPantheR detects, integrates and reports pre-defined features in mass spectrometry data files. It enables the real time annotation of multiple compounds in a single file, or the parallel annotation of multiple compounds in multiple files.

Details

The main functions of **peakPantheR** are `peakPantheR_singleFileSearch` for realtime annotation, and `peakPantheR_parallelAnnotation` for parallel annotation. The `peakPantheRAnnotation` object stores parallel annotation results, while reporting functions help assess the quality of annotation and update fitting parameters. Refer to the vignettes for graphical user interface and command line tutorials.

Author(s)

Maintainer: Arnaud Wolfer <`adwolfer@gmail.com`> (ORCID)

Authors:

- Goncalo Correia <`gscorreia89@gmail.com`> (ORCID)

Other contributors:

- Jake Pearce <`jake.pearce@gmail.com`> [contributor]
- Caroline Sands <`caz119@hotmail.co.uk`> [contributor]

See Also

Useful links:

- `https://github.com/phenomecentre/peakPantheR`
- Report bugs at `https://github.com/phenomecentre/peakPantheR/issues/new`

`peakPantheRAnnotation` *An S4 class to represent peakPantheR annotation results*

Description

The `peakPantheRAnnotation` class is designed to run and store peakPantheR parallel annotation results. Instances of the class are created with the `peakPantheRAnnotation` constructor function, which initialises an object of proper dimension with `spectraPaths` (set samples to process) and `targetFeatTable` (set compounds to target). `spectraPaths` is a character vector of spectra file paths. `targetFeatTable` is a `data.frame` of compounds to target as rows and parameters as columns: `cpdID` (int), `cpdName` (str), `rtMin` (float in seconds), `rt` (float in seconds, or *NA*), `rtMax` (float in seconds), `mzMin` (float), `mz` (float or *NA*), `mzMax` (float).

`peakPantheRAnnotation()`: create an instance of the `peakPantherAnnotation` class.

Usage

```
peakPantheRAnnotation(spectraPaths = NULL, targetFeatTable = NULL,
    cpdID = character(), cpdName = character(), ROI = data.frame(rtMin
    = numeric(), rt = numeric(), rtMax = numeric(), mzMin = numeric(), mz =
    numeric(), mzMax = numeric(), stringsAsFactors = FALSE),
    FIR = data.frame(rtMin = numeric(), rtMax = numeric(), mzMin =
    numeric(), mzMax = numeric(), stringsAsFactors = FALSE),
    uROI = data.frame(rtMin = numeric(), rt = numeric(), rtMax = numeric(),
    mzMin = numeric(), mz = numeric(), mzMax = numeric(), stringsAsFactors =
    FALSE), filepath = character(), cpdMetadata = data.frame(),
    spectraMetadata = data.frame(), acquisitionTime = character(),
    uROIExist = FALSE, useUROI = FALSE, useFIR = FALSE,
    TIC = numeric(), peakTables = list(), dataPoints = list(),
    peakFit = list(), isAnnotated = FALSE)

peakPantheRAnnotation(spectraPaths = NULL, targetFeatTable = NULL,
    cpdID = character(), cpdName = character(), ROI = data.frame(rtMin
    = numeric(), rt = numeric(), rtMax = numeric(), mzMin = numeric(), mz =
    numeric(), mzMax = numeric(), stringsAsFactors = FALSE),
    FIR = data.frame(rtMin = numeric(), rtMax = numeric(), mzMin =
    numeric(), mzMax = numeric(), stringsAsFactors = FALSE),
    uROI = data.frame(rtMin = numeric(), rt = numeric(), rtMax = numeric(),
    mzMin = numeric(), mz = numeric(), mzMax = numeric(), stringsAsFactors =
    FALSE), filepath = character(), cpdMetadata = data.frame(),
    spectraMetadata = data.frame(), acquisitionTime = character(),
    uROIExist = FALSE, useUROI = FALSE, useFIR = FALSE,
    TIC = numeric(), peakTables = list(), dataPoints = list(),
    peakFit = list(), isAnnotated = FALSE)
```

Arguments

`spectraPaths`	NULL or a character vector of spectra file paths, to set samples to process

`targetFeatTable`	NULL or a `data.frame` of compounds to target as rows and parameters as columns: `cpdID` (str), `cpdName` (str), `rtMin` (float in seconds), `rt` (float in seconds, or *NA*), `rtMax` (float in seconds), `mzMin` (float), `mz` (float or *NA*), `mzMax` (float). Set compounds to target.
`cpdID`	A character vector of compound IDs, of length number of compounds
`cpdName`	A character vector of compound names, of length number of compounds
`ROI`	A data.frame of Regions Of Interest (ROI) with compounds as row and ROI parameters as columns: `rtMin` (float in seconds), `rt` (float in seconds, or *NA*), `rtMax` (float in seconds), `mzMin` (float), `mz` (float or *NA*), `mzMax` (float).
`FIR`	A data.frame of Fallback Integration Regions (FIR) with compounds as row and FIR parameters as columns: `rtMin` (float in seconds), `rtMax` (float in seconds), `mzMin` (float), `mzMax` (float).
`uROI`	A data.frame of updated Regions Of Interest (uROI) with compounds as row and uROI parameters as columns: `rtMin` (float in seconds), `rt` (float in seconds, or *NA*), `rtMax` (float in seconds), `mzMin` (float), `mz` (float or *NA*), `mzMax` (float).
`filepath`	A character vector of file paths, of length number of spectra files
`cpdMetadata`	A data.frame of compound metadata, with compounds as row and metadata as columns
`spectraMetadata`	A data.frame of sample metadata, with samples as row and metadata as columns
`acquisitionTime`	A character vector of acquisition date-time (converted from POSIXct) or NA
`uROIExist`	A logical stating if uROI have been set
`useUROI`	A logical stating if uROI are to be used
`useFIR`	A logical stating if FIR are to be used
`TIC`	A numeric vector of TIC or NA, of length number of spectra files
`peakTables`	A list of peakTable data.frame, of length number of spectra files. Each peakTable data.frame has compounds as rows and peak annotation results as columns.
`dataPoints`	A list of length number of spectra files. Each list element is a *ROIsDataPoint* list of `data.frame` of raw data points for each ROI/uROI (retention time 'rt', mass 'mz' and intensity 'int' (as column) of each raw data points (as row))
`peakFit`	A list of length number of spectra files. Each list element is a *curveFit* list of `peakPantheR_curveFit` or NA for each ROI
`isAnnotated`	A logical stating in the annotation took place

Details

The `validObject` method ensures the conformity of an object to the `peakPantheRAnnotation-class`. The number of compounds is based on `cpdID()` length, and the number of samples is based on `filepath()` length. Slot type is not checked as `setClass` enforces it. peakTables and EICs type are checked on the first list element. `annotationTable(object, column)` where *column* is a column from *peakTable*, returns a data.frame of values with the samples as rows, ROI as columns.

Value

(peakPantheRAnnotation)

Slots

`cpdID` A character vector of compound IDs, of length number of compounds

`cpdName` A character vector of compound names, of length number of compounds

`ROI` A data.frame of Regions Of Interest (ROI) with compounds as row and ROI parameters as columns: `rtMin` (float in seconds), `rt` (float in seconds, or *NA*), `rtMax` (float in seconds), `mzMin` (float), `mz` (float or *NA*), `mzMax` (float).

`FIR` A data.frame of Fallback Integration Regions (FIR) with compounds as row and FIR parameters as columns: `rtMin` (float in seconds), `rtMax` (float in seconds), `mzMin` (float), `mzMax` (float).

`uROI` A data.frame of updated Regions Of Interest (uROI) with compounds as row and uROI parameters as columns: `rtMin` (float in seconds), `rt` (float in seconds, or *NA*), `rtMax` (float in seconds), `mzMin` (float), `mz` (float or *NA*), `mzMax` (float).

`filepath` A character vector of file paths, of length number of spectra files

`cpdMetadata` A data.frame of compound metadata, with compounds as row and metadata as columns

`spectraMetadata` A data.frame of sample metadata, with samples as row and metadata as columns

`acquisitionTime` A character vector of acquisition date-time (converted from POSIXct) or NA

`uROIExist` A logical stating if uROI have been set

`useUROI` A logical stating if uROI are to be used

`useFIR` A logical stating if FIR are to be used

`TIC` A numeric vector of TIC or NA, of length number of spectra files

`peakTables` A list of peakTable data.frame, of length number of spectra files. Each peakTable data.frame has compounds as rows and peak annotation results as columns.

`dataPoints` A list of length number of spectra files. Each list element is a *ROIsDataPoint* list of `data.frame` of raw data points for each ROI/uROI (retention time 'rt', mass 'mz' and intensity 'int' (as column) of each raw data points (as row))

`peakFit` A list of length number of spectra files. Each list element is a *curveFit* list of `peakPantheR_curveFit` or NA for each ROI

`isAnnotated` A logical stating if the annotation has taken place

Details:: The *peakTables* `data.frame` are structured as follow:

cpdID	database compound ID
cpdName	compound name
found	was the peak found
rt	retention time of peak apex (sec)
rtMin	leading edge of peak retention time (sec) determined at 0.5% of apex intensity
rtMax	trailing edge of peak retention time (sec) determined at 0.5% of apex intensity
mz	weighted (by intensity) mean of peak m/z across scans
mzMin	m/z peak minimum (between rtMin, rtMax)

mzMax	m/z peak maximum (between rtMin, rtMax)
peakArea	integrated peak area
peakAreaRaw	integrated peak area from raw data points
maxIntMeasured	maximum peak intensity in raw data
maxIntPredicted	maximum peak intensity based on curve fit
is_filled	Logical indicate if the feature was integrated using FIR (Fallback Integration Region)
ppm_error	difference in ppm between the expected and measured m/z
rt_dev_sec	difference in seconds between the expected and measured rt
tailingFactor	the tailing factor is a measure of peak tailing.It is defined as the distance from the front slope of the peak t
asymmetryFactor	the asymmetry factor is a measure of peak tailing. It is defined as the distance from the center line of the p

See Also

Other peakPantheR: `peakPantheR_parallelAnnotation()`, `peakPantheR_singleFileSearch()`

Other parallelAnnotation: `peakPantheR_parallelAnnotation()`, `peakPantheR_singleFileSearch()`

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(),2,8,dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

annotation
# An object of class peakPantheRAnnotation
#  2 compounds in 3 samples.
#   updated ROI do not exist (uROI)
#   does not use updated ROI (uROI)
#   does not use fallback integration regions (FIR)
#   is not annotated
```

```
slotNames(annotation)
# [1] 'cpdID'       'cpdName'      'ROI'             'FIR'       'uROI'
# [6] 'filepath'    'cpdMetadata'  'spectraMetadata' 'acquisitionTime'
# [10] 'uROIExist'  'useUROI'      'useFIR'          'TIC'       'peakTables'
# [15] 'dataPoints' 'peakFit'      'isAnnotated'

## Slots shouldn't be accessed directly, accessors are available:
cpdID(annotation)
# [1] 'ID-1' 'ID-2'
cpdName(annotation)
# [1] 'Cpd 1' 'Cpd 2'
ROI(annotation)
#   rtMin       rt rtMax    mzMin    mz    mzMax cpdID cpdName
# 1  3310 3344.888  3390 522.1948 522.2 522.2052  ID-1   Cpd 1
# 2  3280 3385.577  3440 496.1950 496.2 496.2050  ID-2   Cpd 2
FIR(annotation)
#   rtMin rtMax mzMin mzMax cpdID cpdName
# 1    NA    NA    NA    NA  ID-1   Cpd 1
# 2    NA    NA    NA    NA  ID-2   Cpd 2
uROI(annotation)
#   rtMin rt rtMax mzMin mz mzMax cpdID cpdName
# 1    NA NA    NA    NA NA    NA  ID-1   Cpd 1
# 2    NA NA    NA    NA NA    NA  ID-2   Cpd 2
filepath(annotation)
# [1] 'C:/R/R-3.6.0/library/faahKO/cdf/KO/ko15.CDF'
# [2]'C:/R/R-3.6.0/library/faahKO/cdf/KO/ko16.CDF'
# [3] 'C:/R/R-3.6.0/library/faahKO/cdf/KO/ko18.CDF'
cpdMetadata(annotation)
# data frame with 0 columns and 2 rows
spectraMetadata(annotation)
# data frame with 0 columns and 3 rows
acquisitionTime(annotation)
# [1] NA NA NA
uROIExist(annotation)
# [1] FALSE
useUROI(annotation)
# [1] FALSE
useFIR(annotation)
# [1] FALSE
TIC(annotation)
# [1] NA NA NA
peakTables(annotation)
# [[1]]
# NULL
# [[2]]
# NULL
# [[3]]
# NULL
dataPoints(annotation)
# [[1]]
# NULL
# [[2]]
# NULL
```

```
# [[3]]
# NULL
peakFit(annotation)
# [[1]]
# NULL
# [[2]]
# NULL
# [[3]]
# NULL
isAnnotated(annotation)
# [1] FALSE
}
```

---

peakPantheR_applyRTCorrection

Correct targeted retention time based on reference compounds

Description

Correct targeted features retention time using the RT and RT deviation of previously fitted compounds. The ‘method‘ and ‘params‘ are used to select and parametrise the retention time correction method employed. When ‘robust‘ is set to TRUE, the RANSAC algorithm is used to automatically flag outliers and robustify the correction function fitting.

Usage

```
peakPantheR_applyRTCorrection(
    targetFeatTable,
    referenceTable,
    method = "polynomial",
    params = list(polynomialOrder = 3),
    robust = TRUE
)
```

Arguments

`targetFeatTable`	a `data.frame` of compounds to target as rows and parameters as columns: `cpdID` (str), `cpdName` (str), `rtMin` (float in seconds), `rt` (float in seconds, or *NA*), `rtMax` (float in seconds), `mzMin` (float), `mz` (float or *NA*), `mzMax` (float).
`referenceTable`	a `data.frame` of reference compound information as rows and properties as columns: `cpdID` (str), `cpdName` (str), `rt` (float), `rt_dev_sec` (float)
`method`	(str) name of RT correction method to use (currently `polynomial`)
`params`	(list) list of parameters to pass to the each correction method. Currently allowed inputs are `polynomialOrder` for `method='polynomial'`
`robust`	(bool) whether to use the RANSAC algorithm to flag and ignore outliers during retention time correction

Value

a targetFeatTable with corrected RT

`peakPantheR_loadAnnotationParamsCSV`

Load fit parameters from CSV

Description

Initialise a new `peakPantheRAnnotation` object after loading ROI, uROI and FIR parameters from CSV. `spectraPaths`, `spectraMetadata` or `cpdMetadata` are not initialised and will need to be filled before annotation. `useUROI` and `useFIR` are set to `FALSE` and will need to be set accordingly. `uROIExist` is established depending on the uROI columns present in the CSV and will be set to `TRUE` only if no `NA` are present

Usage

```
peakPantheR_loadAnnotationParamsCSV(CSVParamPath, verbose = TRUE)
```

Arguments

`CSVParamPath`	(str) Path to a CSV file of fit parameters (e.g. as saved by `outputAnnotationDiagnostic`)
`verbose`	(bool) If TRUE message progress

Value

(peakPantheRAnnotation) Object initialised with ROI, uROI and FIR read from the CSV file

Examples

```
## Input data
input_CSV <- data.frame(matrix(nrow=2,ncol=21,dimnames=list(c(),
    c('cpdID', 'cpdName',
    'X','ROI_rt', 'ROI_mz', 'ROI_rtMin', 'ROI_rtMax','ROI_mzMin','ROI_mzMax',
    'X','uROI_rtMin', 'uROI_rtMax', 'uROI_mzMin', 'uROI_mzMax', 'uROI_rt',
    'uROI_mz', 'X', 'FIR_rtMin', 'FIR_rtMax', 'FIR_mzMin', 'FIR_mzMax'))))
input_CSV[1,]  <- c('ID-1', 'Cpd 1', '|', 1.,  2.,  3.,  4.,  5.,  6.,  '|',
                    7.,  8.,  9.,  10., 11., 12., '|', 13., 14., 15., 16.)
input_CSV[2,]  <- c('ID-2', 'Cpd 2', '|', 17., 18., 19., 20., 21., 22., '|',
                    23., 24., 25., 26., 27., 28., '|', 29., 30., 31., 32.)
input_CSV[,-c(1,2,3,10,17)]  <- vapply(input_CSV[,-c(1,2,3,10,17)],
                                        as.numeric, FUN.VALUE=numeric(2))

# temporary file location
savePath1      <- tempfile(pattern='file', tmpdir=tempdir(), fileext='.csv')
# save csv
utils::write.csv(input_CSV, file=savePath1, row.names=FALSE)
```

```
# Load parameters from CSV
loadedAnnotation <- peakPantheR_loadAnnotationParamsCSV(savePath1,
                                                        verbose=TRUE)
# uROIExist set to TRUE
# New peakPantheRAnnotation object initialised for 2 compounds
# An object of class peakPantheRAnnotation
#  2 compounds in 0 samples.
#   updated ROI exist (uROI)
#   does not use updated ROI (uROI)
#   does not use fallback integration regions (FIR)
#   is not annotated
```

---

peakPantheR_parallelAnnotation

Search, integrate and report targeted features in a multiple spectra

Description

Integrate all target features in all files defined in the initialised input object and store results. The use of updated ROI and the integration of FIR are controled by the input object slots `useUROI` and `useFIR`. Files are processed in parallel using peakPantheR_singleFileSearch; `ncores` controls the number of cores used for parallelisation, with `ncores=0` corresponding to serial processing. If the processing of a file fails (file does not exist or error during execution) the sample is removed from the outputed object.

Usage

```
peakPantheR_parallelAnnotation(
    object,
    ncores = 0,
    getAcquTime = TRUE,
    resetWorkers = 1,
    centroided = TRUE,
    curveModel = "skewedGaussian",
    verbose = TRUE,
    ...
)
```

Arguments

`object`	(peakPantheRAnnotation) Initialised peakPantheRAnnotation object defining the samples to process and compounds to target. The slots `useUROI` and `useFIR` controls if uROI must be used and FIR integrated if a feature is not found
`ncores`	(int) Number of cores to use for parallelisation. Default 0 for no parallelisation.
`getAcquTime`	(bool) If TRUE will extract sample acquisition date-time from the mzML metadata (the additional file access will impact run time)

`resetWorkers`	(int) If 0, the parallel cluster is only initiated once. If >0 the cluster will be reset (and the memory of each worker freed) once `ncores * resetWorkers` files have been processed. Default value is 1, the cluster is reset once `ncores` files have been processed. While potentially impacting performance (need to wait until all `ncores * resetWorkers` files are processed before restarting the cluster), shutting down the workers processes regularly will ensure the OS can reallocate memory more efficiently. For values >1, ensure sufficient system memory is available
`centroided`	(bool) use TRUE if the data is centroided, used by `readMSData` when reading the raw data files
`curveModel`	(str) specify the peak-shape model to fit, by default `skewedGaussian`. Accepted values are `skewedGaussian` and `emgGaussian`
`verbose`	(bool) If TRUE message calculation progress, time taken, number of features found (total and matched to targets) and failures
`...`	Passes arguments to `findTargetFeatures` to alter peak-picking parameters

Value

a list: `list()$result` *(peakPantheRAnnotation)* fully annotated object, `list()$failures` *(list)* list of failed samples and error message

See Also

Other peakPantheR: `peakPantheRAnnotation`, `peakPantheR_singleFileSearch()`

Other parallelAnnotation: `peakPantheRAnnotation`, `peakPantheR_singleFileSearch()`

Examples

```
if(requireNamespace('faahKO')){
## Load data
library(faahKO)

# 3 files
input_spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                        system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                        system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# 4 features
input_ROI     <- data.frame(matrix(vector(), 4, 8,
                    dimnames=list(c(), c('cpdID', 'cpdName', 'rtMin', 'rt',
                                        'rtMax', 'mzMin', 'mz', 'mzMax'))),
                    stringsAsFactors=FALSE)
input_ROI[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                    522.2, 522.205222)
input_ROI[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                    496.2, 496.204962)
input_ROI[3,] <- c('ID-3', 'Cpd 3', 3420., 3454.435, 3495., 464.195358,
                    464.2, 464.204642)
input_ROI[4,] <- c('ID-4', 'Cpd 4', 3670., 3701.697, 3745., 536.194638,
```

```
                    536.2, 536.205362)
input_ROI[,c(3:8)] <- vapply(input_ROI[,c(3:8)], as.numeric,
                            FUN.VALUE=numeric(4))

# Initialise object
initAnnotation <- peakPantheRAnnotation(spectraPaths=input_spectraPaths,
                                        targetFeatTable=input_ROI)
# to use updated ROI:
# uROIExist=TRUE, useUROI=TRUE, uROI=input_uROI
# to use FallBack Integration Regions:
# useFIR=TRUE, FIR=input_FIR

# Run serially
result_parallelAnnotation <- peakPantheR_parallelAnnotation(initAnnotation,
                                                            ncores=0,
                                                            getAcquTime=FALSE,
                                                            verbose=TRUE)
# Processing 4 compounds in 3 samples:
#  uROI:\tFALSE
#  FIR:\tFALSE
# ----- ko15 -----
# Polarity can not be extracted from netCDF files, please set manually the
#  polarity with the 'polarity' method.
# Reading data from 4 windows
# Data read in: 0.24 secs
# Warning: rtMin/rtMax outside of ROI; datapoints cannot be used for
#  mzMin/mzMax calculation, approximate mz and returning ROI$mzMin and
#  ROI$mzMax for ROI #1
# Warning: rtMin/rtMax outside of ROI; datapoints cannot be used for
#  mzMin/mzMax calculation, approximate mz and returning ROI$mzMin and
#  ROI$mzMax for ROI #3
# Found 4/4 features in 0.06 secs
# Peak statistics done in: 0.02 secs
# Feature search done in: 0.76 secs
# ----- ko16 -----
# Polarity can not be extracted from netCDF files, please set manually the
#  polarity with the 'polarity' method.
# Reading data from 4 windows
# Data read in: 0.24 secs
# Warning: rtMin/rtMax outside of ROI; datapoints cannot be used for
#  mzMin/mzMax calculation, approximate mz and returning ROI$mzMin and
#  ROI$mzMax for ROI #1
# Warning: rtMin/rtMax outside of ROI; datapoints cannot be used for
#  mzMin/mzMax calculation, approximate mz and returning ROI$mzMin and
#  ROI$mzMax for ROI #2
# Warning: rtMin/rtMax outside of ROI; datapoints cannot be used for
#  mzMin/mzMax calculation, approximate mz and returning ROI$mzMin and
#  ROI$mzMax for ROI #3
# Warning: rtMin/rtMax outside of ROI; datapoints cannot be used for
#  mzMin/mzMax calculation, approximate mz and returning ROI$mzMin and
#  ROI$mzMax for ROI #4
# Found 4/4 features in 0.08 secs
# Peak statistics done in: 0 secs
```

```
# Feature search done in: 0.71 secs
# ----- ko18 -----
# Polarity can not be extracted from netCDF files, please set manually the
#  polarity with the 'polarity' method.
# Reading data from 4 windows
# Data read in: 0.25 secs
# Warning: rtMin/rtMax outside of ROI; datapoints cannot be used for
#  mzMin/mzMax calculation, approximate mz and returning ROI$mzMin and
#  ROI$mzMax for ROI #1
# Warning: rtMin/rtMax outside of ROI; datapoints cannot be used for
#  mzMin/mzMax calculation, approximate mz and returning ROI$mzMin and
#  ROI$mzMax for ROI #2
# Warning: rtMin/rtMax outside of ROI; datapoints cannot be used for
#  mzMin/mzMax calculation, approximate mz and returning ROI$mzMin and
#  ROI$mzMax for ROI #4
# Found 4/4 features in 0.06 secs
# Peak statistics done in: 0 secs
# Feature search done in: 0.71 secs
# ----------------
# Parallel annotation done in: 2.18 secs

# No failures
result_parallelAnnotation$failures

result_parallelAnnotation$annotation
# An object of class peakPantheRAnnotation
#  4 compounds in 3 samples.
#    updated ROI do not exist (uROI)
#    does not use updated ROI (uROI)
#    does not use fallback integration regions (FIR)
#    is annotated
}
```

`peakPantheR_plotEICFit` *Plot samples raw data and detected feature for a single ROI*

Description

plot a ROI across multiple samples (x axis is RT, y axis is intensity). If curveFit is provided, the fitted curve for each sample is added.

Usage

```
peakPantheR_plotEICFit(
    ROIDataPointSampleList,
    curveFitSampleList = NULL,
    rtMin = NULL,
    rtMax = NULL,
```

```
    sampling = 250,
    sampleColour = NULL,
    verbose = TRUE
)
```

Arguments

`ROIDataPointSampleList`	(list) list of `data.frame` of raw data points for each sample (retention time 'rt', mass 'mz' and intensity 'int' (as column) of each raw data points (as row)).
`curveFitSampleList`	(list) NULL or a list of `peakPantheR_curveFit` (or NA) for each sample
`rtMin`	(float) NULL or vector of detected peak minimum retention time (in sec)
`rtMax`	(float) NULL or vector of detected peak maximum retention time (in sec)
`sampling`	(int) Number of points to employ when plotting fittedCurve
`sampleColour`	(str) NULL or vector colour for each sample (same length as `ROIDataPointSampleList`, `rtMin`, `rtMax`)
`verbose`	(bool) if TRUE message when NA scans are removed

Value

Grob (ggplot object)

Examples

```
## Input data
# fake sample 1
# ROI data points
rt1             <- seq(990, 1010, by=20/250)
mz1             <- rep(522., length(rt1))
int1            <- (dnorm(rt1, mean=1000, sd=1.5) * 100) + 1
tmp_DataPoints1 <- data.frame(rt=rt1, mz=mz1, int=int1)
# fittedCurve
fit1        <- list(amplitude=37.068916502809756, center=999.3734222573454,
                    sigma=0.58493182568124724, gamma=0.090582029276037035,
                    fitStatus=2, curveModel='skewedGaussian')
class(fit1)     <- 'peakPantheR_curveFit'

# fake sample 2
# ROI data points
rt2             <- seq(990, 1010, by=20/250)
mz2             <- rep(522., length(rt2))
int2            <- (dnorm(rt2, mean=1002, sd=1.5) * 100) + 1
tmp_DataPoints2 <- data.frame(rt=rt2, mz=mz2, int=int2)
# fittedCurve
fit2        <- list(amplitude=37.073067416755556, center=1001.3736564832565,
                    sigma=0.58496485738212201, gamma=0.090553713725151905,
                    fitStatus=2, curveModel='skewedGaussian')
class(fit2)     <- 'peakPantheR_curveFit'
```

```
## Plot features in 1 sample without colours
peakPantheR_plotEICFit(ROIDataPointSampleList=list(tmp_DataPoints1),
                        curveFitSampleList=list(fit1),
                        rtMin=995., rtMax=1005.,
                        sampling=250, sampleColour=NULL, verbose=FALSE)

## Plot features in 2 samples with colours
peakPantheR_plotEICFit(
                ROIDataPointSampleList=list(tmp_DataPoints1,tmp_DataPoints2),
                curveFitSampleList=list(fit1, fit2),
                rtMin=c(995., 997.), rtMax=c(1005.,1007.),
                sampling=250, sampleColour=c('blue', 'red'), verbose=FALSE)
```

`peakPantheR_plotPeakwidth` *Plot peak value and peakwidth by acquisition time or in input order*

Description

For a single ROI, plot the peak value and peakwidth (RT, m/z, ...) of detected peaks across multiple samples, by acquisition time or in input order. If `rotateAxis=FALSE` x is run order / plot order, y is the `apexValue` / `widthMin` / `widthMax`, if `rotateAxis=TRUE` x is the measurement values and y the run order.

Usage

```
peakPantheR_plotPeakwidth(
    apexValue,
    widthMin = NULL,
    widthMax = NULL,
    acquTime = NULL,
    varName = "variable",
    sampleColour = NULL,
    rotateAxis = FALSE,
    verbose = TRUE
)
```

Arguments

`apexValue`	(float) vector of apex value
`widthMin`	(float) vector of detected peak minimum peakwidth value or NULL (if NULL no peakwidth)
`widthMax`	(float) vector of detected peak maximum peakwidth value or NULL (uf NULL no peakwidth)
`acquTime`	(POSIXct) vector of sample acquisition time as POSIXct or NULL (if NULL points are plotted in the order values are passed as input with the first on top or left)

`varName`	(str) Name of the variable to plot
`sampleColour`	(str) NULL or vector colour for each sample (same length as `apexValue`, `widthMin`, `widthMax`, `acquTime`)
`rotateAxis`	(bool) if TRUE x and y axis are reversed
`verbose`	(bool) if TRUE message when NA scans are removed

Value

Grob (ggplot object)

Examples

```
## Input data
apexVal <- c(1, 2, 3, 4)
minVal  <- c(0, 0, 2, 2)
maxVal  <- c(2, 4, 4, 5)
acqTime <- as.POSIXct(c('2017-07-13 21:06:14', '2017-07-14 21:06:14',
                        '2017-07-15 21:06:14', '2017-07-16 21:06:14'))

## Plot 4 sampels with colour
peakPantheR_plotPeakwidth(apexValue=apexVal, widthMin=minVal,widthMax=maxVal,
                        acquTime=NULL, varName='Test variable 1',
                        sampleColour=c('blue','red','green','orange'),
                        rotateAxis=FALSE, verbose=FALSE)

## Plot 4 samples with colour by acquisition time
peakPantheR_plotPeakwidth(apexValue=apexVal, widthMin=minVal,widthMax=maxVal,
                        acquTime=acqTime, varName='Test variable 2',
                        sampleColour=c('blue','red','green','orange'),
                        rotateAxis=FALSE, verbose=FALSE)

## Plot 4 samples with colour, rotate axis
peakPantheR_plotPeakwidth(apexValue=apexVal, widthMin=minVal,widthMax=maxVal,
                        acquTime=NULL, varName='Test variable 3',
                        sampleColour=c('blue','red','green','orange'),
                        rotateAxis=TRUE, verbose=FALSE)

## Plot 4 samples with colour by acquisition time, rotate axis
peakPantheR_plotPeakwidth(apexValue=apexVal, widthMin=minVal,widthMax=maxVal,
                        acquTime=acqTime, varName='Test variable 4',
                        sampleColour=c('blue','red','green','orange'),
                        rotateAxis=FALSE, verbose=FALSE)
```

`peakPantheR_quickEIC` *Extract and plot a EIC from a raw data file*

Description

Simple plot of an Extracted Ion Chromatogram (EIC) from a raw data file and a provided mz and rt window. Return ggplot plot object.

Usage

```
peakPantheR_quickEIC(
    spectraPath,
    rt,
    mz,
    valuesOnly = "Plot",
    centroided = TRUE,
    msLevel = 1L,
    verbose = TRUE
)
```

Arguments

`spectraPath`	(str) Path to the raw data file to read (uses ‘MSnbase::readMSData()‘)
`rt`	(numeric(2) or two-column matrix) the lower and upper retention time range from which the data should be extracted. If a matrix is passed, each row corresponds to a different window. If not provided, the full retention time range will be extracted.
`mz`	(numeric(2) or two-column matrix) the lower and upper mass range from which the data should be extracted. If a matrix is passed, each row corresponds to a different window. If not provided, the full mass range will be extracted.
`valuesOnly`	(str) If ’Raw’ only load the file and return a table of raw extracted values (exported version of ‘extractSignalRawData()‘). If ‘EIC‘ return a table of EIC data point. Else return the EIC plot (default).
`centroided`	(bool) Indicate to ‘MSnbase::readMSData()‘ whether the spectra file is centroided or not (default to ’TRUE’).
`msLevel`	(int) The MS level at which the data should be extracted (default to MS level 1).
`verbose`	(bool) Output proress information or not.

Value

Grob (ggplot object) of the EIC plot, if ‘valuesOnly=’Raw’‘ returns a data.frame of raw datapoints with as columns ’rt’, ’mz’ and ’int’. If ‘valuesOnly=’EIC’‘ returns a data.frame of EIC datapoints with as columns ’rt’ and ’int’.

Examples

```
## Use a file form the faahKO package and plot an EIC of interest
library(faahKO)
spectraPath <- system.file('cdf/KO/ko15.CDF',package='faahKO')
peakPantheR_quickEIC(spectraPath,
                     rt = c(3290., 3410.),
                     mz = c(522.194778, 522.205222))
```

`peakPantheR_ROIStatistics` *Save to disk each ROI EIC and mean IS RT*

Description

Using reference samples (`referenceSpectraFiles`), save (to `saveFolder`) each ROI EIC (`ROI`) and reports the mean apex RT for all IS (`IS_ROI`) across samples

Usage

```
peakPantheR_ROIStatistics(
    referenceSpectraFiles,
    saveFolder,
    ROI = NULL,
    IS_ROI = NULL,
    sampleColour = NULL,
    ncores = 0,
    saveISPlots = TRUE,
    verbose = TRUE
)
```

Arguments

`referenceSpectraFiles`	(str) A character vector of paths to the reference spectra files
`saveFolder`	(str) Path to the folder where EICs and IS mean RT (`IS_mean_RT.csv`) will be saved
`ROI`	(data.frame) NULL or a data.frame of Regions Of Interest (ROI) with compounds as row and ROI parameters as columns: `rtMin` (float in seconds), `rt` (float in seconds, or *NA*), `rtMax` (float in seconds), `mzMin` (float), `mz` (float or *NA*), `mzMax` (float) (if NULL, ROI EICs are not saved)
`IS_ROI`	(data.frame) NULL or a data.frame of IS ROI with IS as row and ROI parameters as columns: `rtMin` (float in seconds), `rt` (float in seconds, or *NA*), `rtMax` (float in seconds), `mzMin` (float), `mz` (float or *NA*), `mzMax` (float) (if NULL IS mean RT is not calculated and saved in `IS_mean_RT.csv`)
`sampleColour`	(str) NULL or vector colour for each sample
`ncores`	(int) Number of cores to use to integrate IS in parallel
`saveISPlots`	(bool) If TRUE save a diagnostic plot for each IS to `saveFolder/IS_search` compound
`verbose`	(bool) If TRUE message progress

Value

None

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 2 samples and 1 targeted
## compound

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 1, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(1))

# input
refSpecFiles  <- spectraPaths
input_ROI     <- targetFeatTable
input_IS_ROI  <- targetFeatTable
sampleColour  <- c('blue', 'red')

# temporary saveFolder
saveFolder1   <- tempdir()

# Calculate ROI statiscs
peakPantheR_ROIStatistics(refSpecFiles, saveFolder1, ROI=input_ROI,
                            IS_ROI=input_IS_ROI, sampleColour=sampleColour,
                            ncores=0, saveISPlots=TRUE, verbose=TRUE)
}
```

`peakPantheR_singleFileSearch`

Search, integrate and report targeted features in a raw spectra

---

Description

Report for a raw spectra the TIC, acquisition time, integrated targeted features, fitted curves and datapoints for each region of interest. Optimised to reduce the number of file access. Features not detected can be integrated using fallback integration regions (FIR).

Usage

```
peakPantheR_singleFileSearch(
    singleSpectraDataPath,
    targetFeatTable,
```

```
    peakStatistic = FALSE,
    plotEICsPath = NA,
    getAcquTime = FALSE,
    FIR = NULL,
    centroided = TRUE,
    curveModel = "skewedGaussian",
    verbose = TRUE,
    ...
)
```

Arguments

`singleSpectraDataPath`	(str) path to netCDF or mzML raw data file (centroided, **only with the channel of interest**).
`targetFeatTable`	a `data.frame` of compounds to target as rows. Columns: `cpdID` (str), `cpdName` (str), `rtMin` (float in seconds), `rt` (float in seconds, or *NA*), `rtMax` (float in seconds), `mzMin` (float), `mz` (float or *NA*), `mzMax` (float).
`peakStatistic`	(bool) If TRUE calculates additional peak statistics: 'ppm_error', 'rt_dev_sec', 'tailing factor' and 'asymmetry factor'
`plotEICsPath`	(str or NA) If not NA, will save a *.png* of all ROI EICs at the path provided (`'filepath/filename.png'` expected). If NA no plot saved
`getAcquTime`	(bool) If TRUE will extract sample acquisition date-time from the mzML metadata (the additional file access will impact run time)
`FIR`	(data.frame or NULL) If not NULL, integrate Fallback Integration Regions (FIR) when a feature is not found. Compounds as row are identical to `targetFeatTable`, columns are `rtMin` (float in seconds), `rtMax` (float in seconds), `mzMin` (float), `mzMax` (float).
`centroided`	(bool) use TRUE if the data is centroided, used by `readMSData` when reading the raw data file
`curveModel`	(str) specify the peak-shape model to fit, by default `skewedGaussian`. Accepted values are `skewedGaussian` and `emgGaussian`
`verbose`	(bool) If TRUE message calculation progress, time taken and number of features found
`...`	Passes arguments to `findTargetFeatures` to alter peak-picking parameters (e.g. `curveModel`, `sampling`, `params` as a list of parameters for each ROI or 'guess',...)

Value

a list: `list()$TIC` *(int)* TIC value, `list()$peakTable` *(data.frame)* targeted features results (see Details), `list()$curveFit` *(list)* list of `peakPantheR_curveFit` or NA for each ROI, `list()$acquTime` *(POSIXct or NA)* date-time of sample acquisition from mzML metadata, `list()$ROIsDataPoint` *(list)* a list of `data.frame` of raw data points for each ROI (retention time 'rt', mass 'mz' and intensity 'int' (as column) of each raw data points (as row)).

Details:: The returned *peakTable* `data.frame` is structured as follow:

cpdID	database compound ID
cpdName	compound name
found	was the peak found
rt	retention time of peak apex (sec)
rtMin	leading edge of peak retention time (sec) determined at 0.5% of apex intensity
rtMax	trailing edge of peak retention time (sec) determined at 0.5% of apex intensity
mz	weighted (by intensity) mean of peak m/z across scans
mzMin	m/z peak minimum (between rtMin, rtMax)
mzMax	m/z peak maximum (between rtMin, rtMax)
peakArea	integrated peak area
peakAreaRaw	integrated peak area from raw data points
maxIntMeasured	maximum peak intensity in raw data
maxIntPredicted	maximum peak intensity based on curve fit
is_filled	Logical indicate if the feature was integrated using FIR (Fallback Integration Region)
ppm_error	difference in ppm between the expected and measured m/z
rt_dev_sec	difference in seconds between the expected and measured rt
tailingFactor	the tailing factor is a measure of peak tailing.It is defined as the distance from the front slope of the peak t
asymmetryFactor	the asymmetry factor is a measure of peak tailing. It is defined as the distance from the center line of the p

See Also

Other peakPantheR: `peakPantheRAnnotation`, `peakPantheR_parallelAnnotation()`

Other parallelAnnotation: `peakPantheRAnnotation`, `peakPantheR_parallelAnnotation()`

Examples

```
if(requireNamespace('faahKO')){
## Load data
library(faahKO)
netcdfFilePath <- system.file('cdf/KO/ko15.CDF', package = 'faahKO')

## targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

res <- peakPantheR_singleFileSearch(netcdfFilePath,targetFeatTable,
                                    peakStatistic=TRUE)
# Polarity can not be extracted from netCDF files, please set manually the
#    polarity with the 'polarity' method.
# Reading data from 2 windows
# Data read in: 0.16 secs
```

```
# Warning: rtMin/rtMax outside of ROI; datapoints cannot be used for
#   mzMin/mzMax calculation, approximate mz and returning ROI$mzMin and
#   ROI$mzMax for ROI #1
# Found 2/2 features in 0.05 secs
# Peak statistics done in: 0 secs
# Feature search done in: 0.75 secs

res
# $TIC
# [1] 2410533091
#
# $peakTable
#   found    rtMin       rt    rtMax    mzMin    mz    mzMax peakArea
# 1  TRUE 3309.759 3346.828 3385.410 522.1948 522.2 522.2052 26133727
# 2  TRUE 3345.377 3386.529 3428.279 496.2000 496.2 496.2000 35472141
#   peakAreaRaw maxIntMeasured maxIntPredicted cpdID cpdName is_filled
# 1    26071378         889280        901015.8  ID-1   Cpd 1     FALSE
# 2    36498367        1128960       1113576.7  ID-2   Cpd 2     FALSE
#    ppm_error   rt_dev_sec  tailingFactor  asymmetryFactor
# 1 0.02337616    1.9397590       1.015357         1.026824
# 2 0.02460103    0.9518072       1.005378         1.009318
#
# $acquTime
# [1] NA
#
#
# $curveFit
# $curveFit[[1]]
# $amplitude
# [1] 162404.8
#
# $center
# [1] 3341.888
#
# $sigma
# [1] 0.07878613
#
# $gamma
# [1] 0.00183361
#
# $fitStatus
# [1] 2
#
# $curveModel
# [1] 'skewedGaussian'
#
# attr(,'class')
# [1] 'peakPantheR_curveFit'
#
# $curveFit[[2]]
# $amplitude
# [1] 199249.1
#
```

```
# $center
# [1] 3382.577
#
# $sigma
# [1] 0.07490442
#
# $gamma
# [1] 0.00114719
#
# $fitStatus
# [1] 2
#
# $curveModel
# [1] 'skewedGaussian'
#
# attr(,'class')
# [1] 'peakPantheR_curveFit'
#
#
# $ROIsDataPoint
# $ROIsDataPoint[[1]]
#          rt    mz    int
# 1  3315.154 522.2   2187
# 2  3316.719 522.2   3534
# 3  3318.284 522.2   6338
# 4  3319.849 522.2  11718
# 5  3321.414 522.2  21744
# 6  3322.979 522.2  37872
# 7  3324.544 522.2  62424
# 8  3326.109 522.2  98408
# 9  3327.673 522.2 152896
# 10 3329.238 522.2 225984
# ...
#
# $ROIsDataPoint[[2]]
#          rt    mz     int
# 1  3280.725 496.2    1349
# 2  3290.115 496.2    2069
# 3  3291.680 496.2    3103
# 4  3293.245 496.2    5570
# 5  3294.809 496.2   10730
# 6  3296.374 496.2   20904
# 7  3297.939 496.2   38712
# 8  3299.504 496.2   64368
# 9  3301.069 496.2   97096
# 10 3302.634 496.2  136320
# ...
}
```

`peakPantheR_start_GUI` *peakPantheR Graphical User Interface*

Description

peakPantheR Graphical User Interface (GUI) implements all the functions for the parallel detection, integration and reporting of pre-defined features in multiple mass spectrometry data files. To exit press `ESC` in the command line.

Usage

```
peakPantheR_start_GUI(browser = TRUE)
```

Arguments

`browser`	If TRUE open the graphical user interface in a web browser instead of a R window. Default is TRUE

Value

None, start GUI. To exit press `ESC` in the command line.

Examples

```
print("Start graphical interface, press 'ESC' in the command line to stop")
# peakPantheR_start_GUI()
```

`peakTables,peakPantheRAnnotation-method`

peakTables accessor with cpdID and cpdName added back

Description

peakTables accessor with cpdID and cpdName added back

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
peakTables(object)
```

Arguments

`object`	peakPantheRAnnotation

Value

(data.frame) A list of peakTable data.frame, of length number of spectra files. Each peakTable data.frame has compounds as rows and peak annotation results as columns, with added compound ID and name.

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

## default values without annotation
peakTables(annotation)
# [[1]]
# NULL
# [[2]]
# NULL
# [[3]]
# NULL
}
```

`plotEICDetectedPeakwidth`

Plot samples raw data and detected feature for a single ROI

Description

Plot a ROI across multiple samples (x axis is RT, y axis is intensity) with the matching detected peak rt and peakwidth under it. If curveFit is provided, the fitted curve for each compound is added. RT and peakwidth are plotted in the order spectra are passed, with the first spectra on top.

Usage

```
plotEICDetectedPeakwidth(
    ROIDataPointSampleList,
```

```
    cpdID,
    cpdName,
    rt,
    rtMin,
    rtMax,
    mzMin,
    mzMax,
    ratio = 0.85,
    sampling = 250,
    curveFitSampleList = NULL,
    sampleColour = NULL,
    verbose = TRUE
)
```

Arguments

`ROIDataPointSampleList`	(list) list of `data.frame` of raw data points for each sample (retention time 'rt', mass 'mz' and intensity 'int' (as column) of each raw data points (as row)).
`cpdID`	(str) Compound ID
`cpdName`	(str) Compound Name
`rt`	(float) vector of detected peak apex retention time (in sec)
`rtMin`	(float) vector of detected peak minimum retention time (in sec)
`rtMax`	(float) vector ofdetected peak maximum retention time (in sec)
`mzMin`	(float) ROI minimum m/z (matching EIC)
`mzMax`	(float) ROI maximum m/z (matching EIC)
`ratio`	(float) value between 0 and 1 defining the vertical percentage taken by the EICs subplot
`sampling`	(int) Number of points to employ when plotting fittedCurve
`curveFitSampleList`	(list) NULL or a list of `peakPantheR_curveFit` (or NA) for each sample
`sampleColour`	(str) NULL or vector colour for each sample (same length as `EICs`, `rt`, `rtMin`, `rtMax`)
`verbose`	(bool) if TRUE message when NA scans are removed

Value

Grob (ggplot object)

`plotHistogram` *Plot variable histogram and density*

Description

Plot the histogram and density of the variable

Usage

```
plotHistogram(var, varName = "Variable", density = TRUE, ...)
```

Arguments

`var`	(float) vector of values to plot
`varName`	(str) Name of the variable to plot
`density`	(bool) If TRUE plot overlay the density on the variable
`...`	Passes arguments to ggplot2::geom_histogram, e.g. `bins=20`, `binwidth=1`

Value

Grob (ggplot object)

`predictCurve` *Predict curve values*

Description

Evaluate fitted curve values at `x` data points

Usage

```
predictCurve(fittedCurve, x)
```

Arguments

`fittedCurve`	(peakPantheR_curveFit) A 'peakPantheR_curveFit': a list of curve fitting parameters, curve shape model `curveModel` and nls.lm fit status `fitStatus`.
`x`	(numeric) values at which to evaluate the fitted curve

Details

Examples cannot be computed as the function is not exported: ## Input a fitted curve fittedCurve <- list(amplitude=275371.1, center=3382.577, sigma=0.07904697, gamma=0.001147647, fitStatus=2, curveModel='skewedGaussian') class(fittedCurve) <- 'peakPantheR_curveFit' input_x <- c(3290, 3300, 3310, 3320, 3330, 3340, 3350, 3360, 3370, 3380, 3390, 3400, 3410)

Predict y at each input_x pred_y <- predictCurve(fittedCurve, input_x) pred_y # [1] 2.347729e-08 1.282668e-05 3.475590e-03 4.676579e-01 3.129420e+01 # [6] 1.043341e+03 1.736915e+04 1.447754e+05 6.061808e+05 1.280037e+06 # [11] 1.369651e+06 7.467333e+05 2.087477e+05

Value

fitted curve values at x

`prepare_advanced_target_parameters` *Process target region parameters (with uROI, FIR) for object initialisation*

Description

Process target region parameters with uROI and FIR (`cpdID`, `cpdName`,`ROI_rt`, `ROI_mz`, `ROI_rtMin`, `ROI_rtMax`, `ROI_mzMin`, `ROI_mzMax`, `uROI_rtMin`,`uROI_rtMax`, `uROI_mzMin`, `uROI_mzMax`, `uROI_rt`, `uROI_mz`, `FIR_rtMin`, `FIR_rtMax`,`FIR_mzMin`, `FIR_mzMax`) and return input variables for `peakPantheRAnnotation()`

Usage

```
prepare_advanced_target_parameters(paramTable, verbose)
```

Arguments

`paramTable`	(data.frame) Target region parameters
`verbose`	(bool) If TRUE message progress

Value

(list) List of `targetFeatTable`, `uROI`, `FIR`, `uROIExist`

`prepare_basic_target_parameters` *Process target region parameters (simple format) for object initialisation*

Description

Process the simple target region parameters (`cpdID`, `cpdName`, `mzMin`,`mzMax`, `mz`, `rtMin`, `rtMax`, `rt`) and return input variables for `peakPantheRAnnotation()`

Usage

```
prepare_basic_target_parameters(paramTable)
```

Arguments

`paramTable`	(data.frame) Target region parameters

Value

(list) List of `targetFeatTable`, `uROI`, `FIR`, `uROIExist`

`resetAnnotation,peakPantheRAnnotation-method` *Reset a peakPantheRAnnotation and alter samples and compounds information*

Description

Reset a peakPantheRAnnotation (remove results and set `isAnnotated=FALSE`). If a different number of samples (`spectraPaths`) or compounds (`targetFeatTable`) are passed, the object will be initialised to the new size. For input values left as NULL, the slots (`filepath` (from `spectraPaths`), `ROI`, `cpdID`, `cpdName` (from `targetFeatTable`), `uROI`, `FIR`, `cpdMetadata`, `spectraMetadata`, `uROIExist`, `useUROI` and `useFIR`) will be filled with values from `previousAnnotation`.

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
resetAnnotation(
    previousAnnotation,
    spectraPaths = NULL,
    targetFeatTable = NULL,
    uROI = NULL,
    FIR = NULL,
    cpdMetadata = NULL,
    spectraMetadata = NULL,
```

```
    uROIExist = NULL,
    useUROI = NULL,
    useFIR = NULL,
    verbose = TRUE,
    ...
)
```

Arguments

`previousAnnotation`	(peakPantheRAnnotation) object to reset
`spectraPaths`	NULL or a character vector of spectra file paths, to set samples to process
`targetFeatTable`	NULL or a `data.frame` of compounds to target as rows and parameters as columns: `cpdID` (str), `cpdName` (str), `rtMin` (float in seconds), `rt` (float in seconds, or *NA*), `rtMax` (float in seconds), `mzMin` (float), `mz` (float or *NA*), `mzMax` (float). Set compounds to target.
`uROI`	NULL or a data.frame of updated Regions Of Interest (uROI) with compounds as row and uROI parameters as columns: `rtMin` (float in seconds), `rt` (float in seconds, or *NA*), `rtMax` (float in seconds), `mzMin` (float), `mz` (float or *NA*), `mzMax` (float).
`FIR`	NULL or a data.frame of Fallback Integration Regions (FIR) with compounds as row and FIR parameters as columns: `rtMin` (float in seconds), `rtMax` (float in seconds), `mzMin` (float), `mzMax` (float).
`cpdMetadata`	NULL or a data.frame of compound metadata, with compounds as row and metadata as columns
`spectraMetadata`	NULL or a data.frame of sample metadata, with samples as row and metadata as columns
`uROIExist`	NULL or a logical stating if uROI have been set
`useUROI`	NULL or a logical stating if uROI are to be used
`useFIR`	NULL or a logical stating if FIR are to be used
`verbose`	(bool) If TRUE message progress
`...`	Additional slots and values to set when resetting the object (`cpdID`, `cpdName`, `ROI`, `filepath`, `TIC`, `acquisitionTime`, `peakTables`, `dataPoints`, `peakFit`)

Value

(peakPantheRAnnotation) object reset with previous results removed and slots updated

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
```

```
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

smallAnnotation  <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                        targetFeatTable=targetFeatTable)
smallAnnotation
# An object of class peakPantheRAnnotation
#  2 compounds in 2 samples.
#  updated ROI do not exist (uROI)
#  does not use updated ROI (uROI)
#  does not use fallback integration regions (FIR)
#  is not annotated

# Reset and change number of spectra
newSpectraPaths  <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))
largerAnnotation <- resetAnnotation(smallAnnotation,
                                    spectraPaths=newSpectraPaths,
                                    verbose=TRUE)
largerAnnotation
# An object of class peakPantheRAnnotation
#  2 compounds in 3 samples.
#  updated ROI do not exist (uROI)
#  does not use updated ROI (uROI)
#  does not use fallback integration regions (FIR)
#  is not annotated
}
```

`resetFIR,peakPantheRAnnotation-method`
Reset FIR windows to uROI or ROI values Reset FIR windows to uROI (or ROI if `uROIExist=FALSE`*)*

Description

Reset FIR windows to uROI or ROI values Reset FIR windows to uROI (or ROI if `uROIExist=FALSE`)

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
resetFIR(object, verbose)
```

Arguments

`object`	(peakPantheRAnnotation) object for which FIR are to be reset
`verbose`	(bool) If TRUE message progress

Value

(peakPantheRAnnotation) object with FIR values reset

Examples

```
## Initialise a peakPantheRAnnotation object with 2 targeted compounds

## targetFeatTable
input_targetFeatTable <- data.frame(matrix(vector(), 2, 8,
                            dimnames=list(c(), c('cpdID', 'cpdName', 'rtMin',
                            'rt', 'rtMax', 'mzMin', 'mz', 'mzMax'))),
                            stringsAsFactors=FALSE)
input_targetFeatTable[1,] <- c('ID-1', 'Cpd 1',  3.,  1.,  4.,  5.,  2.,  6.)
input_targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 19., 17., 20., 21., 18., 22.)
input_targetFeatTable[,c(3:8)] <- sapply(input_targetFeatTable[,c(3:8)],
                                        as.numeric)
## FIR
input_FIR       <- data.frame(matrix(vector(), 2, 4, dimnames=list(c(),
                                c('rtMin', 'rtMax', 'mzMin', 'mzMax'))),
                                stringsAsFactors=FALSE)
input_FIR[1,]   <- c(13., 14., 15., 16.)
input_FIR[2,]   <- c(29., 30., 31., 32.)

annotation <- peakPantheRAnnotation(targetFeatTable = input_targetFeatTable,
                                    FIR = input_FIR, uROIExist = FALSE)

## Reset FIR with ROI values as uROI are not set
updatedAnnotation <- resetFIR(annotation, verbose=TRUE)
# FIR will be reset with ROI values as uROI values are not set
```

`retentionTimeCorrection,peakPantheRAnnotation-method`

Apply retention time correction methods to adjust the retention time information in the uROI of peakPantheRAnnotation object

Description

Performs retention time correction to re-adjust the expected retention time position of compounds. Requires an annotated peakPantheRAnnotation object (`isAnnotated=TRUE`). The original `rt` value is used as expected and the observed deviation measured in the `rt_dev_sec` field is taken as the deviation to be corrected.

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
retentionTimeCorrection(
    annotationObject,
    rtCorrectionReferences = NULL,
    method = "polynomial",
    params = list(polynomialOrder = 2),
    robust = FALSE,
    rtWindowWidth = 15,
    diagnostic = TRUE
)
```

Arguments

`annotationObject`	(peakPantheRAnnotation) object with previous fit results to adjust retention time values in uROI and FIR annotationObject, rtCorrectionReferences=NULL,
`rtCorrectionReferences`	(list) of compounds IDs (`cpdID`) to be used as retention time references. All `cpdID` entries must be present in the object and previously annotated. If NULL, use all compounds.
`method`	(str) name of RT correction method to use (currently `polynomial` or `constant`
`params`	(list) list of parameters to pass to each correction method. Currently allowed inputs are `polynomialOrder` for `method='polynomial'`
`robust`	(bool) whether to use the RANSAC algorithm to flag and ignore outliers during retention time correction
`rtWindowWidth`	(numeric) full width in seconds of the retention time window defined around the corrected retention time value for each compound
`diagnostic`	(bool) If TRUE returns diagnostic plots (specific to each correction method)

Value

(list) containing entries ‘annotation‘, with the new and retention time corrected peakPantheRAnnotation, and ‘plot‘ (if `diagnostic=TRUE`).

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds
```

```
# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

smallAnnotation  <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                        targetFeatTable=targetFeatTable)
# annotate files serially
annotation_result <- peakPantheR_parallelAnnotation(smallAnnotation,
                                            ncores=0, verbose=TRUE)
data_annotation   <- annotation_result$annotation

# Example with constant correction
rtCorrectionOutput <- retentionTimeCorrection(
                        annotationObject = data_annotation,
                        rtCorrectionReferences=c('ID-1'),
                        method='constant', params=list(),
                        robust=FALSE,
                        rtWindowWidth=15,
                        diagnostic=TRUE)

rtCorrectedAnnotation <- rtCorrectionOutput$annotation

# rtCorrectedAnnotation
# An object of class peakPantheRAnnotation
#  2 compounds in 2 samples.
#  updated ROI exists, with a modified rt (uROI)
#  uses updated ROI (uROI)
#  uses fallback integration regions (FIR)
#  is annotated

rtCorrectionPlot <- rtCorrectionOutput$plot
# rtCorrectedPlot
# A ggplot2 object
#  Scatterplot where x=`r` in the and y=`rt_dev_sec` from data_annotation
#  Points colored depending on whether the reference was used to fit
# the correction model
}
```

`ROI,peakPantheRAnnotation-method` *ROI accessor returns targetFeatTable with cpdID, cpdName added*

Description

ROI accessor returns targetFeatTable with cpdID, cpdName added

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
ROI(object)
```

Arguments

`object`	peakPantheRAnnotation

Value

(data.frame) target feature table with compounds as row and ROI parameters as columns

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                c('cpdID', 'cpdName', 'rtMin', 'rt', 'rtMax', 'mzMin', 'mz',
                'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

ROI(annotation)
#   rtMin       rt rtMax    mzMin    mz    mzMax cpdID cpdName
# 1  3310 3344.888  3390 522.1948 522.2 522.2052  ID-1   Cpd 1
```

```
# 2  3280 3385.577  3440 496.1950 496.2 496.2050  ID-2   Cpd 2
}
```

`saveSingleFileMultiEIC` *Save to disk a plot of all ROI EIC and detected feature range*

Description

Plot and save a `.png` of all ROI (x is RT, y is intensity), with the matching detected peak rt and peakwidth under it.

Usage

```
saveSingleFileMultiEIC(
    ROIsDataPoint,
    curveFit,
    foundPeakTable,
    savePath,
    width = 15,
    height = 15,
    verbose = TRUE
)
```

Arguments

`ROIsDataPoint`	(list) a list of `data.frame` of raw data points for each ROI (retention time 'rt', mass 'mz' and intensity 'int' (as column) of each raw data points (as row)).
`curveFit`	(list) a list of `peakPantheR_curveFit` or NA for each ROI
`foundPeakTable`	(data.frame) `data.frame` as generated by findTargetFeatures, with features as rows and peak properties as columns. The following columns are mandatory: `cpdID`, `cpdName`, `rt`, `rtmin`, `rtmax`, `mzmin`, `mzmax`.
`savePath`	(str) Full path to save a *.png* of all ROI EICs, expect `'filepath/filename.png'`.
`width`	(float) Width in cm for a single ROI plot (if more than one plot in total, 2 columns will be used). dpi set to a 100.
`height`	(float) height in a cm for a single ROI plot. dpi set to 100
`verbose`	(bool) if TRUE message progress

Value

None

`skewedGaussian_guess` *Guess function for initial skewed gaussian parameters and bounds*

Description

Guess function for initial skewed gaussian parameters and bounds, at the moment only checks the x position

Usage

```
skewedGaussian_guess(x, y)
```

Arguments

`x`	(numeric) x values (e.g. retention time)
`y`	(numeric) y observed values (e.g. spectra intensity)

Value

A list of guessed starting parameters `list()$init_params`, lower `list()$lower_bounds` and upper bounds `list()$upper_bounds` (`$gamma`, `$center`, `$sigma`, `$amplitude`)

`skewedGaussian_minpack.lm` *Implementation of the Skewed Gaussian peak shape for use with minpack.lm*

Description

Implementation of the Skewed Gaussian peak shape for use with minpack.lm

Usage

```
skewedGaussian_minpack.lm(params, xx)
```

Arguments

`params`	(list) skewed gaussian parameters (`params$gamma`, `params$center`, `params$sigma`, `params$amplitude`)
`xx`	(numeric) values at which to evalute the skewed gaussian

Value

value of the skewed gaussian evaluated at xx

`skewedGaussian_minpack.lm_objectiveFun`
Skewed Gaussian minpack.lm objective function

Description

Skewed Gaussian minpack.lm objective function, calculates residuals using the skewed gaussian Peak Shape

Usage

```
skewedGaussian_minpack.lm_objectiveFun(params, observed, xx)
```

Arguments

`params`	(list) skewed gaussian parameters (`params$gamma`, `params$center`, `params$sigma`, `params$amplitude`)
`observed`	(numeric) observed y value at xx
`xx`	(numeric) value at which to evalute the skewed gaussian

Value

difference between observed and expected skewed gaussian value evaluated at xx

---

`spectraMetadata,peakPantheRAnnotation-method`
spectraMetadata accessor

Description

spectraMetadata accessor

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
spectraMetadata(object)
```

Arguments

`object`	peakPantheRAnnotation

Value

(data.frame) A data.frame of sample metadata, with samples as row and metadata as columns

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

## default values not initialised
spectraMetadata(annotation)
# data frame with 0 columns and 3 rows
}
```

---

spectraPaths_and_metadata_UI_helper

UI data import helper - prepare file paths and metadata

Description

Return spectraPaths and spectraMetadata from a .csv file (if available). If reading from the spectraMetadata file, the spectraPaths are taken from the ‘filepath‘ column

Usage

```
spectraPaths_and_metadata_UI_helper(
    spectraPaths = NULL,
    spectraMetadataPath = NULL
)
```

Arguments

`spectraPaths`	NULL or character vector of spectra file paths, to set samples to process
`spectraMetadataPath`	NULL or path to a csv of spectra metadata, with spectra as row and metadata as columns. (spectraPaths in column ‘filepath‘)

Value

spectraPaths (str) and spectraMetadata (DataFrame or NULL) read from the CSV file

Examples

```
## Input data
# spectraPath
input_spectraPaths    <- c('./path/file1', './path/file2', './path/file3')

# spectraMetadata
input_spectraMetadata <- data.frame(matrix(data=c(input_spectraPaths,
                            c('a','b','c')), nrow=3, ncol=2,
                            dimnames=list(c(),c('filepath', 'testcol')),
                            byrow=FALSE), stringsAsFactors=FALSE)

# temporary file location
spectraMetaPath <- tempfile(pattern="file", tmpdir=tempdir(), fileext='.csv')
# save csv
utils::write.csv(input_spectraMetadata,
                    file=spectraMetaPath,
                    row.names=FALSE)

# load data from CSV
spectraPaths_and_metadata_UI_helper(spectraPaths = NULL,
                                    spectraMetadataPath = spectraMetaPath)
```

`spectra_metadata_colourScheme_UI_helper`

UI export plot helper - sample colour

Description

Return a vector of spectra colours based on a metadata column

Usage

```
spectra_metadata_colourScheme_UI_helper(
    annot,
    splColrColumn = NULL
)
```

Arguments

`annot`	(peakPantheRAnnotation) Annotation object
`splColrColumn`	(str) NULL, None or a spectraMetadata column for colouring each sample

Value

(character) Vector of colours

Examples

```
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
spectraPaths <- c('./path/file1', './path/file2', './path/file3')

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

emptyAnnotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                        targetFeatTable=targetFeatTable)

# colour scheme with no spectraMetadata
outputAnnotationFeatureMetadata_UI_helper(emptyAnnotation)
# NULL
```

`TIC,peakPantheRAnnotation-method` *TIC accessor*

Description

TIC accessor

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
TIC(object)
```

Arguments

`object`	peakPantheRAnnotation

Value

(float) A numeric vector of Total Ion Chromatogram or NA, of length number of spectra files

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

## default values without annotation
TIC(annotation)
# [1] NA NA NA
}
```

`uROI,peakPantheRAnnotation-method`

uROI accessor returns targetFeatTable with cpdID, cpdName added

---

Description

uROI accessor returns targetFeatTable with cpdID, cpdName added

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
uROI(object)
```

Arguments

`object`	peakPantheRAnnotation

Value

(data.frame) target feature table with compounds as row and uROI parameters as columns

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable     <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                c('cpdID', 'cpdName', 'rtMin', 'rt', 'rtMax', 'mzMin', 'mz',
                'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

## default values without annotation
uROI(annotation)
#   rtMin rt rtMax mzMin mz mzMax cpdID cpdName
# 1    NA NA    NA    NA NA    NA  ID-1   Cpd 1
# 2    NA NA    NA    NA NA    NA  ID-2   Cpd 2
}
```

`uROIExist,peakPantheRAnnotation-method`

uROIExist accessor

Description

uROIExist accessor

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
uROIExist(object)
```

Arguments

`object`	peakPantheRAnnotation

Value

(bool) flag if uROI have been set

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

uROIExist(annotation)
# [1] FALSE
}
```

`useFIR,peakPantheRAnnotation-method`
useFIR accessor

Description

useFIR accessor

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
useFIR(object)
```

Arguments

`object`	peakPantheRAnnotation

Value

(bool) flag if FIR are to be used

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

useFIR(annotation)
# [1] FALSE
}
```

`useUROI,peakPantheRAnnotation-method`
useUROI accessor

Description

useUROI accessor

Usage

```
## S4 method for signature 'peakPantheRAnnotation'
useUROI(object)
```

Arguments

`object`	peakPantheRAnnotation

Value

(bool) flag if uROI are to be used

Examples

```
if(requireNamespace('faahKO')){
## Initialise a peakPantheRAnnotation object with 3 samples and 2 targeted
## compounds

# Paths to spectra files
library(faahKO)
spectraPaths <- c(system.file('cdf/KO/ko15.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko16.CDF', package = 'faahKO'),
                    system.file('cdf/KO/ko18.CDF', package = 'faahKO'))

# targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(),
                    c('cpdID','cpdName','rtMin','rt','rtMax','mzMin','mz',
                    'mzMax'))), stringsAsFactors=FALSE)
targetFeatTable[1,] <- c('ID-1', 'Cpd 1', 3310., 3344.888, 3390., 522.194778,
                        522.2, 522.205222)
targetFeatTable[2,] <- c('ID-2', 'Cpd 2', 3280., 3385.577, 3440., 496.195038,
                        496.2, 496.204962)
targetFeatTable[,c(3:8)] <- vapply(targetFeatTable[,c(3:8)], as.numeric,
                                    FUN.VALUE=numeric(2))

annotation <- peakPantheRAnnotation(spectraPaths=spectraPaths,
                                    targetFeatTable=targetFeatTable)

useUROI(annotation)
# [1] FALSE
}
```

`[,peakPantheRAnnotation,ANY,ANY,ANY-method`
extract parts of peakPantheRAnnotation class

Description

extract parts of peakPantheRAnnotation class

Usage

```
## S4 method for signature 'peakPantheRAnnotation,ANY,ANY,ANY'
x[i, j, drop = "missing"]
```

Arguments

`x`	object from which to extract element(s) or in which to replace element(s).
`i`	(sample) indices specifying elements to extract or replace
`j`	(compound) indices specifying elements to extract or replace
`drop`	not applicable

Value

(peakPantheRAnnotation) object subset

Index