

# Package ‘peakPantheR’

July 10, 2018

**Title** Peak Picking and Annotation of High Resolution Experiments

**Version** 1.2.0

**Date** 2018-07-08

**Description** An automated pipeline for the detection, integration and reporting of predefined features across a large number of mass spectrometry data files.

**Depends** R (>= 3.4.0)

**Imports** foreach (>= 1.4.4), doParallel (>= 1.0.11), ggplot2 (>= 2.2.1), 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)

**biocViews** MSnbase (>= 2.4.0), mzR (>= 2.12.0)

**License** GPL-3

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 6.0.1

**Suggests** testthat, faahKO, knitr, rmarkdown, pander

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Arnaud Wolfer [aut, cre],  
Goncalo Correia [ctb],  
Jake Pearce [ctb]

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

**Repository** CRAN

**Date/Publication** 2018-07-10 17:00:03 UTC

## R topics documented:

acquisitionTime,peakPantheRAnnotation-method . . . . .	3
annotationDiagnosticMultiplot . . . . .	3
annotationDiagnosticPlots,peakPantheRAnnotation-method . . . . .	4

annotationParamsDiagnostic,peakPantheRAnnotation-method . . . . .	5
annotationTable,peakPantheRAnnotation-method . . . . .	5
cpdID,peakPantheRAnnotation-method . . . . .	6
cpdMetadata,peakPantheRAnnotation-method . . . . .	6
cpdName,peakPantheRAnnotation-method . . . . .	7
dataPoints,peakPantheRAnnotation-method . . . . .	7
EICs,peakPantheRAnnotation-method . . . . .	8
extractSignalRawData . . . . .	8
filename,peakPantheRAnnotation-method . . . . .	10
filepath,peakPantheRAnnotation-method . . . . .	11
findTargetFeatures . . . . .	11
FIR,peakPantheRAnnotation-method . . . . .	14
fitCurve . . . . .	14
generateIonChromatogram . . . . .	16
getAcquisitionDateMZML . . . . .	17
getTargetFeatureStatistic . . . . .	17
integrateFIR . . . . .	19
is.peakPantheR_curveFit . . . . .	20
isAnnotated,peakPantheRAnnotation-method . . . . .	20
nbCompounds,peakPantheRAnnotation-method . . . . .	21
nbSamples,peakPantheRAnnotation-method . . . . .	21
outputAnnotationDiagnostic,peakPantheRAnnotation-method . . . . .	22
outputAnnotationParamsCSV,peakPantheRAnnotation-method . . . . .	23
outputAnnotationResult,peakPantheRAnnotation-method . . . . .	23
peakFit,peakPantheRAnnotation-method . . . . .	24
peakPantheR . . . . .	24
peakPantheRAnnotation . . . . .	25
peakPantheR_loadAnnotationParamsCSV . . . . .	30
peakPantheR_parallelAnnotation . . . . .	31
peakPantheR_singleFileSearch . . . . .	33
peakTables,peakPantheRAnnotation-method . . . . .	37
plotEICDetectedPeakwidth . . . . .	38
plotEICFit . . . . .	39
plotHistogram . . . . .	39
plotPeakwidth . . . . .	40
predictCurve . . . . .	41
resetAnnotation,peakPantheRAnnotation-method . . . . .	42
ROI,peakPantheRAnnotation-method . . . . .	43
saveSingleFileMultiEIC . . . . .	43
skewedGaussian_guess . . . . .	44
skewedGaussian_minpack.lm . . . . .	45
skewedGaussian_minpack.lm_objectiveFun . . . . .	45
skew_erf . . . . .	46
spectraMetadata,peakPantheRAnnotation-method . . . . .	46
TIC,peakPantheRAnnotation-method . . . . .	47
uROI,peakPantheRAnnotation-method . . . . .	47
uROIExist,peakPantheRAnnotation-method . . . . .	48
useFIR,peakPantheRAnnotation-method . . . . .	48

<i>acquisitionTime,peakPantheRAnnotation-method</i>	3
useUROI,peakPantheRAnnotation-method . . . . .	49
[,peakPantheRAnnotation,ANY,ANY,ANY-method . . . . .	49
<b>Index</b>	<b>50</b>

---

*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

---

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* 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.

---

## Description

Generate fit diagnostic plots 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

---

annotationParamsDiagnostic,peakPantheRAnnotation-method

*Set uROI and FIR based on annotation results 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).*

---

### Description

Set uROI and FIR based on annotation results 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

---

annotationTable,peakPantheRAnnotation-method

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

---

### Description

annotationTable accessor 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

---

cpdID,peakPantheRAnnotation-method  
*cpdID accessor*

---

**Description**

cpdID accessor

**Usage**

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

**Arguments**

object	peakPantheRAnnotation
--------	-----------------------

---

cpdMetadata,peakPantheRAnnotation-method  
*cpdMetadata accessor*

---

**Description**

cpdMetadata accessor

**Usage**

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

**Arguments**

object	peakPantheRAnnotation
--------	-----------------------

---

cpdName,peakPantheRAnnotation-method  
*cpdName accessor*

---

### **Description**

cpdName accessor

### **Usage**

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

### **Arguments**

object            peakPantheRAnnotation

---

dataPoints,peakPantheRAnnotation-method  
*dataPoints accessor*

---

### **Description**

dataPoints accessor

### **Usage**

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

### **Arguments**

object            peakPantheRAnnotation

---

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

---

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



**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.

**Examples**

```
## Not run:
## Use a file from 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=T)

# 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
```

```
## End(Not run)
```

---

filename,peakPantheRAnnotation-method

*filename accessor by splitting filepath*

---

### **Description**

filename accessor by splitting filepath

**Usage**

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

**Arguments**

```
object          peakPantheRAnnotation
```

---

```
filepath,peakPantheRAnnotation-method
      filepath accessor
```

---

**Description**

filepath accessor

**Usage**

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

**Arguments**

```
object          peakPantheRAnnotation
```

---

```
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

**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)
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)

**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
maxIntMeasured	maximum peak intensity in raw data
maxIntPredicted	maximum peak intensity based on curve fit (at apex)

**Examples**

```
## Not run:
## Load data
library(faahK0)
library(MSnbase)
netcdfFilePath <- system.file('cdf/KO/ko15.CDF', package = "faahK0")
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", "rtMax", "mzMin", "mz", "mzMax"))),
      stringsAsFactors=F)
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] <- sapply(targetFeatTable[,3:8], as.numeric)

ROIsPt              <- extractSignalRawData(raw_data, rt=targetFeatTable[,c('rtMin','rtMax')],
      mz=targetFeatTable[,c('mzMin','mzMax')], verbose=TRUE)

# Reading data from 2 windows

foundPeaks <- findTargetFeatures(ROIsPt, targetFeatTable, verbose=T)
# 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 maxIntMeasured maxIntPredicted
# 1  TRUE 3309.759 3346.828 3385.410 522.1948 522.2 522.2052 26133727      889280      901015.8
# 2  TRUE 3345.377 3386.529 3428.279 496.2000 496.2 496.2000 35472141     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"

## End(Not run)

```

---

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

---

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)
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)

**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

**Examples**

```
## Not run:
## 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"

## End(Not run)

```

---

```
generateIonChromatogram
```

*Generate ion chromatogram from raw data points*

---

### Description

On the input raw data, aggregate intensities across the m/z 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 m/z in each scan. One of sum, max, min, mean

### Value

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

### Examples

```

## Not run:
## 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

## End(Not run)
```

---

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 <code>data.frame</code> 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 <code>data.frame</code> as generated by <code>findTargetFeatures</code> , 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

**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
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 peak

**Examples**

```
## Not run:
# 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)

input_ROI <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(), c("cpdID",
```

```

      "cpdName", "rtMin", "rt", "rtMax", "mzMin", "mz", "mzMax"))),
      stringsAsFactors=F)
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] <- sapply(input_ROI[,3:8], as.numeric)

# foundPeakTable
input_foundPeakTable <- data.frame(matrix(vector(), 2, 10, dimnames=list(c(),
  c("found", "rtMin", "rt", "rtMax", "mzMin", "mz", "mzMax",
  "peakArea", "maxIntMeasured", "maxIntPredicted"))),
  stringsAsFactors=F)
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] <- sapply(input_foundPeakTable[,c(1)], as.logical)

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

## End(Not run)

```

---

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, 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`

---

*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*

---

*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*

---

outputAnnotationDiagnostic,peakPantheRAnnotation-method

*Save to disk the annotation parameters as CSV and a diagnostic plot per fitted compound 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*

---

### Description

Save to disk the annotation parameters as CSV and a diagnostic plot per fitted compound 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, ...)
```

### 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
...	Additional parameters for plotting i.e. sampling for the number of points to employ when plotting fittedCurve

### Value

None

---

outputAnnotationParamsCSV,peakPantheRAnnotation-method  
*Save annotation parameters as CSV Save annotation parameters (ROI, uROI and FIR) to disk as a CSV file for editing*

---

### Description

Save annotation parameters as CSV 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)
```

### Arguments

object	(peakPantheRAnnotation) Annotated peakPantheRAnnotation object
saveFolder	(str) Path of folder where annotationParameters_summary.csv will be saved
verbose	(bool) If TRUE message progress

### Value

None

---

outputAnnotationResult,peakPantheRAnnotation-method  
*Save to disk all annotation results as csv files 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)*

---

### Description

Save to disk all annotation results as csv files 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

---

peakFit, peakPantheRAnnotation-method  
*peakFit accessor*

---

**Description**

peakFit accessor

**Usage**

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

**Arguments**

object peakPantheRAnnotation

---

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 and 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>

Other contributors:

- Goncalo Correia <gscorreia89@gmail.com> [contributor]
- Jake Pearce <jake.pearce@imperial.ac.uk> [contributor]

---

`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 = F), FIR = data.frame(rtMin
  = numeric(), rtMax = numeric(), mzMin = numeric(), mzMax = numeric(),
  stringsAsFactors = F), uROI = data.frame(rtMin = numeric(), rt = numeric(),
  rtMax = numeric(), mzMin = numeric(), mz = numeric(), mzMax = numeric(),
  stringsAsFactors = F), 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 = F), FIR = data.frame(rtMin
= numeric(), rtMax = numeric(), mzMin = numeric(), mzMax = numeric(),
stringsAsFactors = F), uROI = data.frame(rtMin = numeric(), rt = numeric(),
rtMax = numeric(), mzMin = numeric(), mz = numeric(), mzMax = numeric(),
stringsAsFactors = F), 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 <code>data.frame</code> of compounds to target as rows and parameters as columns: cpdID (str), cpdName (str), rtMin (float in seconds), rt (float in seconds, or <i>NA</i> ), rtMax (float in seconds), mzMin (float), mz (float or <i>NA</i> ), 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 <code>data.frame</code> of Regions Of Interest (ROI) with compounds as row and ROI parameters as columns: rtMin (float in seconds), rt (float in seconds, or <i>NA</i> ), rtMax (float in seconds), mzMin (float), mz (float or <i>NA</i> ), mzMax (float).
FIR	A <code>data.frame</code> 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 <code>data.frame</code> of updated Regions Of Interest (uROI) with compounds as row and uROI parameters as columns: rtMin (float in seconds), rt (float in seconds, or <i>NA</i> ), rtMax (float in seconds), mzMin (float), mz (float or <i>NA</i> ), mzMax (float).
filepath	A character vector of file paths, of length number of spectra files
cpdMetadata	A <code>data.frame</code> of compound metadata, with compounds as row and metadata as columns
spectraMetadata	A <code>data.frame</code> 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 <code>data.frame</code> , of length number of spectra files. Each peak-Table <code>data.frame</code> has compounds as rows and peak annotation results as columns.

dataPoints	A list of length number of spectra files. Each list element is a <i>ROIsDataPoint</i> 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 <i>curveFit</i> 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.

## 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: <code>rtMin</code> (float in seconds), <code>rt</code> (float in seconds, or <i>NA</i> ), <code>rtMax</code> (float in seconds), <code>mzMin</code> (float), <code>mz</code> (float or <i>NA</i> ), <code>mzMax</code> (float).
FIR	A data.frame of Fallback Integration Regions (FIR) with compounds as row and FIR parameters as columns: <code>rtMin</code> (float in seconds), <code>rtMax</code> (float in seconds), <code>mzMin</code> (float), <code>mzMax</code> (float).
uROI	A data.frame of updated Regions Of Interest (uROI) with compounds as row and uROI parameters as columns: <code>rtMin</code> (float in seconds), <code>rt</code> (float in seconds, or <i>NA</i> ), <code>rtMax</code> (float in seconds), <code>mzMin</code> (float), <code>mz</code> (float or <i>NA</i> ), <code>mzMax</code> (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 <i>ROIsDataPoint</i> 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 <i>curveFit</i> 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
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 to the peak apex
asymmetryFactor	the asymmetry factor is a measure of peak tailing. It is defined as the distance from the center line of the peak to the peak apex

### See Also

Other peakPantheR: [peakPantheR\\_parallelAnnotation](#), [peakPantheR\\_singleFileSearch](#)

Other parallelAnnotation: [peakPantheR\\_parallelAnnotation](#), [peakPantheR\\_singleFileSearch](#)

### Examples

```

if(requireNamespace("faahKO") & getRversion()<"3.6"){
## 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=F)
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)] <- sapply(targetFeatTable[,c(3:8)], as.numeric)

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"       "filepath"
# [7] "cpdMetadata" "spectraMetadata" "acquisitionTime" "uROIExist" "useUROI"    "useFIR"
# [13] "TIC"       "peakTables" "dataPoints" "peakFit"   "isAnnotated"

annotation@cpdID
# [1] "ID-1" "ID-2"
annotation@cpdName
# [1] "Cpd 1" "Cpd 2"
annotation@ROI
#   rtMin   rt   rtMax   mzMin   mz   mzMax
# 1 3310 3344.888 3390 522.1948 522.2 522.2052
# 2 3280 3385.577 3440 496.1950 496.2 496.2050
annotation@FIR
#   rtMin rtMax mzMin mzMax
# 1   NA   NA   NA   NA
# 2   NA   NA   NA   NA
annotation@uROI
#   rtMin rt   rtMax mzMin mz   mzMax
# 1   NA NA   NA   NA NA   NA
# 2   NA NA   NA   NA NA   NA
annotation@filepath
# [1] "C:/R/R-3.4.3/library/faahK0/cdf/K0/ko15.CDF" "C:/R/R-3.4.3/library/faahK0/cdf/K0/ko16.CDF"
# [2] "C:/R/R-3.4.3/library/faahK0/cdf/K0/ko18.CDF"
annotation@cpdMetadata
# data frame with 0 columns and 2 rows
annotation@spectraMetadata
# data frame with 0 columns and 3 rows
annotation@acquisitionTime
# [1] NA NA NA
annotation@uROIExist
# [1] FALSE
annotation@useUROI
# [1] FALSE
annotation@useFIR
# [1] FALSE
annotation@TIC
# [1] NA NA NA
annotation@peakTables
# [[1]]
# NULL
# [[2]]
# NULL
# [[3]]
# NULL
annotation@dataPoints
# [[1]]
# NULL
# [[2]]
# NULL
# [[3]]

```

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

---

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 as saved by outputAnnotationDiagnostic  
verbose (bool) If TRUE message progress

## Value

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

---

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 controlled by the input object slots useUROI and useFIR. Files are processed in parallel using `link{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 outputted object.

## Usage

```
peakPanther_parallelAnnotation(object, ncores = 0, getAcquTime = TRUE,  
  resetWorkers = 1, 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 meta-data (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 <code>ncores * resetWorkers</code> files have been processed. Default value is 1, the cluster is reset once <code>ncores</code> files have been processed. While potentially impacting performance (need to wait until all <code>ncores * resetWorkers</code> 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
verbose	(bool) If TRUE message calculation progress, time taken, number of features found (total and matched to targets) and failures
...	Passes arguments to <code>findTargetFeatures</code> 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("faahK0") & getRversion()<"3.6"){
## Load data
library(faahK0)

# 3 files
input_spectraPaths <- c(system.file('cdf/K0/ko15.CDF', package = "faahK0"),
                        system.file('cdf/K0/ko16.CDF', package = "faahK0"),
                        system.file('cdf/K0/ko18.CDF', package = "faahK0"))

# 4 features
input_ROI      <- data.frame(matrix(vector(), 4, 8,
                                   dimnames=list(c("cpdID", "cpdName", "rtMin", "rt",
                                                  "rtMax", "mzMin", "mz", "mzMax"))),
                              stringsAsFactors=F)
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)] <- sapply(input_ROI[,c(3:8)], as.numeric)

# 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: FALSE
# FIR: FALSE
# ----- 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\_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, 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 meta-data (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).

**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:

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

mzMax	m/z peak maximum (between rtMin, rtMax)
peakArea	integrated peak area
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 to the center line 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 peak to the back slope of the peak

## See Also

Other peakPantheR: [peakPantheRAnnotation](#), [peakPantheR\\_parallelAnnotation](#)

Other parallelAnnotation: [peakPantheRAnnotation](#), [peakPantheR\\_parallelAnnotation](#)

## Examples

```
if(requireNamespace("faahK0") & getRversion()<"3.6"){
## Load data
library(faahK0)
netcdfFilePath <- system.file('cdf/K0/ko15.CDF', package = "faahK0")

## targetFeatTable
targetFeatTable <- data.frame(matrix(vector(), 2, 8, dimnames=list(c(), c("cpdID",
"cpdName", "rtMin", "rt", "rtMax", "mzMin", "mz", "mzMax"))),
stringsAsFactors=F)
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)] <- sapply(targetFeatTable[,c(3:8)], as.numeric)

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 maxIntMeasured
# 1 TRUE 3309.759 3346.828 3385.410 522.1948 522.2 522.2052 26133727 889280
# 2 TRUE 3345.377 3386.529 3428.279 496.2000 496.2 496.2000 35472141 1128960
# maxIntPredicted cpdID cpdName is_filled ppm_error rt_dev_sec tailingFactor
```

```
# 1      901015.8 ID-1  Cpd 1    FALSE 0.02337616  1.9397590    1.015357
# 2      1113576.7 ID-2  Cpd 2    FALSE 0.02460103  0.9518072    1.005378
#   asymmetryFactor
# 1      1.026824
# 2      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
# ...
}
```

---

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

---

 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 of detected 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)

---

plotEICFit	<i>Plot samples raw data and detected feature for a single ROI</i>
------------	--

---

**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**

```
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)

---

plotHistogram	<i>Plot variable histogram and density</i>
---------------	--

---

**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)

---

plotPeakwidth	<i>Plot peak value and peakwidth by acquisition time or in input order</i>
---------------	--

---

**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**

```
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 (if 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)



---

predictCurve	<i>Predict curve values</i>
--------------	-----------------------------

---

## 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

## Value

fitted curve values at x

## Examples

```
## Not run:
## 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 1.043341e+03 1.736915e+04
# [8] 1.447754e+05 6.061808e+05 1.280037e+06 1.369651e+06 7.467333e+05 2.087477e+05

## End(Not run)
```

---

resetAnnotation,peakPantheRAnnotation-method

*Reset a peakPantheRAnnotation and alter samples and compounds information 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.*

---

## Description

Reset a peakPantheRAnnotation and alter samples and compounds information 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 <a href="#">data.frame</a> 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 <a href="#">data.frame</a> 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 <a href="#">data.frame</a> 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

---

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

---

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 evaluate 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 evaluate the skewed gaussian

### Value

difference between observed and expected skewed gaussian value evaluated at xx

---

skew_erf	<i>Gaussian Error function</i>
----------	--------------------------------

---

**Description**

Implementation of the gaussian error function

**Usage**

```
skew_erf(x)
```

**Arguments**

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

**Value**

Value of the gaussian error function evaluated at x

---

spectraMetadata,peakPantheRAnnotation-method	<i>spectraMetadata accessor</i>
--	---------------------------------

---

**Description**

spectraMetadata accessor

**Usage**

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

**Arguments**

object peakPantheRAnnotation

---

TIC,peakPantheRAnnotation-method  
*TIC accessor*

---

**Description**

TIC accessor

**Usage**

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

**Arguments**

object            peakPantheRAnnotation

---

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

---

uROIExist,peakPantheRAnnotation-method  
*uROIExist accessor*

---

**Description**

uROIExist accessor

**Usage**

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

**Arguments**

object            peakPantheRAnnotation

---

useFIR,peakPantheRAnnotation-method  
*useFIR accessor*

---

**Description**

useFIR accessor

**Usage**

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

**Arguments**

object            peakPantheRAnnotation



---

```
useUROI,peakPantheRAnnotation-method
      useUROI accessor
```

---

**Description**

useUROI accessor

**Usage**

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

**Arguments**

object            peakPantheRAnnotation

---

```
[,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

# Index

[,peakPantherAnnotation, ANY, ANY, ANY-method, 7  
49  
[,peakPantherAnnotation-method 7  
([,peakPantherAnnotation, ANY, ANY, ANY-method),  
49  
\_PACKAGE (peakPanther), 24  
dataPoints  
(dataPoints,peakPantherAnnotation-method),  
7  
acquisitionTime  
(acquisitionTime,peakPantherAnnotation-method),  
3  
dataPoints,peakPantherAnnotation-method,  
7  
acquisitionTime,peakPantherAnnotation-method,  
3  
annotationDiagnosticMultiplot, 3  
annotationDiagnosticPlots  
(annotationDiagnosticPlots,peakPantherAnnotation-method),  
8  
4  
EICs,peakPantherAnnotation-method, 8  
extractSignalRawData, 8  
annotationDiagnosticPlots,peakPantherAnnotation-method,  
4  
filename  
(filename,peakPantherAnnotation-method),  
10  
annotationParamsDiagnostic,peakPantherAnnotation-method,  
5  
filename,peakPantherAnnotation-method,  
10  
annotationTable  
(filepath,peakPantherAnnotation-method),  
11  
(annotationTable,peakPantherAnnotation-method),  
11  
annotationTable,peakPantherAnnotation-method,  
5  
filepath,peakPantherAnnotation-method,  
11  
findTargetFeatures, 11, 18, 19, 44  
FIR (FIR,peakPantherAnnotation-method),  
14  
cpdID  
(cpdID,peakPantherAnnotation-method),  
6  
FIR,peakPantherAnnotation-method, 14  
fitCurve, 14  
cpdID,peakPantherAnnotation-method, 6  
cpdMetadata  
(cpdMetadata,peakPantherAnnotation-method),  
6  
generateIonChromatogram, 16  
getAcquisitionDateMZML, 17  
cpdMetadata,peakPantherAnnotation-method,  
6  
getTargetFeatureStatistic, 17  
cpdName  
integrateFIR, 19  
(cpdName,peakPantherAnnotation-method),  
19  
is.peakPanther\_curveFit, 20

- isAnnotated
  - (isAnnotated, peakPantherAnnotation-method), 20
- isAnnotated, peakPantherAnnotation-method, 20
- nbCompounds
  - (nbCompounds, peakPantherAnnotation-method), 21
- nbCompounds, peakPantherAnnotation-method, 21
- nbSamples
  - (nbSamples, peakPantherAnnotation-method), 21
- nbSamples, peakPantherAnnotation-method, 21
- outputAnnotationDiagnostic
  - (outputAnnotationDiagnostic, peakPantherAnnotation-method), 22
- outputAnnotationDiagnostic, peakPantherAnnotation-method, 22
- outputAnnotationParamsCSV
  - (outputAnnotationParamsCSV, peakPantherAnnotation-method), 23
- outputAnnotationParamsCSV, peakPantherAnnotation-method, 23
- outputAnnotationResult
  - (outputAnnotationResult, peakPantherAnnotation-method), 23
- outputAnnotationResult, peakPantherAnnotation-method, 23
- peakFit
  - (peakFit, peakPantherAnnotation-method), 24
- peakFit, peakPantherAnnotation-method, 24
- peakPanther, 24
- peakpanther (peakPanther), 24
- peakPanther-package (peakPanther), 24
- peakPanther\_loadAnnotationParamsCSV, 30
- peakPanther\_parallelAnnotation, 25, 28, 31, 35
- peakPanther\_singleFileSearch, 25, 28, 31, 33
- peakPantherAnnotation, 25, 31, 35
- peakPantherAnnotation-class
  - (peakPantherAnnotation), 25
- peakTables
  - (peakTables, peakPantherAnnotation-method), 37
- peakTables, peakPantherAnnotation-method, 37
- plotEICDetectedPeakwidth, 38
- plotEICFit, 39
- plotHistogram, 39
- plotPeakwidth, 40
- predictCurve, 41
- resetAnnotation
  - (resetAnnotation, peakPantherAnnotation-method), 42
- resetAnnotation, peakPantherAnnotation-method, 42
- ROI (ROI, peakPantherAnnotation-method), 43
- ROI, peakPantherAnnotation-method, 43
- saveSingleFileMultiEIC, 43
- skew\_erf, 46
- skewedGauss in methods, 44
- skewedGaussian\_minpack.lm, 45
- skewedGaussian\_minpack.lm\_objectiveFun, 45
- spectraMetadata
  - (spectraMetadata, peakPantherAnnotation-method), 46
- spectraMetadata, peakPantherAnnotation-method, 46
- TIC (TIC, peakPantherAnnotation-method), 47
- TIC, peakPantherAnnotation-method, 47
- uROI
  - (uROI, peakPantherAnnotation-method), 47
- uROI, peakPantherAnnotation-method, 47
- uROIExist
  - (uROIExist, peakPantherAnnotation-method), 48
- uROIExist, peakPantherAnnotation-method, 48
- useFIR
  - (useFIR, peakPantherAnnotation-method), 48
- useFIR, peakPantherAnnotation-method, 48

useFIR, peakPantherAnnotation-method,  
48

useUR0I  
(useUR0I, peakPantherAnnotation-method),  
49

useUR0I, peakPantherAnnotation-method,  
49