Package ‘metabCombiner’

March 30, 2021

Version 1.0.1

Date 2020-12-05

Title Method for Combining LC-MS Metabolomics Feature Measurements

License GPL-3

Description This package aligns LC-HRMS metabolomics datasets acquired from biologically similar specimens analyzed under similar, but not necessarily identical, conditions. Two peak-picked and aligned metabolomics feature tables (consisting of m/z, rt, and per-sample abundance measurements, plus optional identifiers & adduct annotations) are accepted as input. The package outputs a combined table of feature pair alignments, organized into groups of similar m/z, and ranked by a similarity score. Input tables are assumed to be acquired using similar (but not necessarily identical) analytical methods.

Depends R (>= 4.0), dplyr (>= 1.0)

Imports methods, mgcv, caret, S4Vectors, stats, utils, rlang, graphics, matrixStats

Suggests knitr, rmarkdown, testthat

BugReports `https://www.github.com/hhabra/metabCombiner/issues`

NeedsCompilation yes

RoxygenNote 7.1.1

Encoding UTF-8

Collate 'adjustData.R' 'calcScores.R' 'check_pars.R' 'classes.R' 'combinerCheck.R' 'compare_strings.R' 'data.R' 'detectFields.R' 'evaluateParams.R' 'fit_model.R' 'formCombinedTable.R' 'generics.R' 'labelRows.R' 'metabCombiner.R' 'metabCombiner_package_doc.R' 'metabData.R' 'methods-metabCombiner.R' 'methods-metabData.R' 'mzGroup.R' 'plot_fit.R' 'selectAnchors.R' 'write2file.R' 'zzz.R'

VignetteBuilder knitr

biocViews Software, MassSpectrometry, Metabolomics

LazyData false

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

git_branch RELEASE_3_12

git_last_commit 6fde963

git_last_commit_date 2020-12-07

Date/Publication 2021-03-29

Author Hani Habra [aut, cre],
Alla Karnovsky [ths]

Maintainer Hani Habra <hhabra1@gmail.com>

R topics documented:

adjustData — *Process and Filter Metabolomics Feature Lists*

Description

adjustData contains a set of pre-analysis steps for processing LC-MS metabolomics feature tables individually

Usage

```
adjustData(Data, misspc, measure, rtmin, rtmax, zero, duplicate)
```

Arguments

Data	a metabData object.
misspc	Numeric. Threshold missingness percentage for analysis.
measure	Character. Choice of central abundance measure; either "median" or "mean".
rtmin	Numeric. Minimum retention time for analysis.
rtmax	Numeric. Maximum retention time for analysis.
zero	Logical. Whether to consider zero values as missing.
duplicate	Ordered numeric pair (m/z, rt) tolerance parameters for duplicate feature search.

Details

The pre-analysis adjustment steps include: 1) Restriction to a feature retention time range rtmin $\leq$ rt $\leq$ rtmax 2) Removal of features with a percent missingness exceeding misspc 3) Removal of duplicate metabolomics features.

After processing, abundance quantile (Q) values are calculated between 0 & 1 for the remaining features, as ranked by the measure argument, unless provided by the user.

Value

Updated metabData object. The data field is processed by the listed steps and stats list updated to contain feature statistics.

See Also

metabData: the constructor for metabData objects, filterRT: function for filtering by retention times, findDuplicates: function for finding duplicates calcScores *Compute Feature Similarity Scores*

Description

Calculates a pairwise similarity (between 0 & 1) between all grouped features in metabCombiner object. The similarity score calculation is described in scorePairs.

Usage

```
calcScores(
  object,
  A,
  B,
  C,
  fit = c("gam", "loess"),
  groups = NULL,
  useAdduct = FALSE,
  adduct = 1.25,
  usePPM = FALSE,
  brackets_ignore = c("(", "[", "{")
)
```

Arguments

object	metabCombiner object.
A	Numeric weight for penalizing m/z differences.
B	Numeric weight for penalizing differences between fitted & observed retention times
C	Numeric weight for differences in Q (abundance quantiles).
fit	Character. Choice of fitted rt model, "gam" or "loess."
groups	integer. Vector of feature groups to score. If set to NULL (default), will compute scores for all feature groups.
useAdduct	logical. Option to penalize mismatches in (non-empty, non-bracketed) adduct column annotations.
adduct	numeric. If useAdduct is TRUE, divides score of mismatched, non-empty and non-bracked adduct column labels by this value.
usePPM	logical. Option to use relative (as opposed to absolute) m/z differences in score computations.
brackets_ignore	If useAdduct = TRUE, bracketed adduct character strings of these types will be ignored according to this argument

Details

This function updates the rtProj, score, rankX, and rankY columns in the combinedTable report. First, using the RT mapping model computed in the previous step(s), rtx values are projected onto rty. Then similarity scores are calculated based on m/z, rt (fitted vs observed), and Q differences, with multiplicative weight penalties A, B, and C.

If the datasets contain representative set of shared identities (idx = idy), evaluateParams provides some guidance on appropriate A, B, and C values to use. In testing, the best values for A should lie between 50 and 120, according to mass accuracy; B should lie between 5 and 15 depending on fitting accuracy (higher if datasets processed under roughly identical conditions) ; C should vary between 0 and 1, depending on sample similarity. See examples below.

If using ppm (usePPM = TRUE), do not use the above guidelines for A values. The suggested range is between 0.01 and 0.05, though this hasn't been thoroughly tested yet. Also, if using adduct information (useAdduct = TRUE), the score is divided by the numeric adduct argument if non-empty and non-bracketed adduct values do not match. Be sure that adduct annotations are accurate before using this functionality.

Value

metabCombiner object with updated combinedTable. rtProj column will contain fitted retention times determined from previously computed model; score will contain computed pairwise similarity scores of feature pairs; rankX & rankY are the integer ranks of scores for x & y features in descending order.

See Also

evaluateParams, scorePairs

Examples

```
data(plasma30)
data(plasma20)

p30 <- metabData(plasma30, samples = "CHEAR")
p20 <- metabData(plasma20, samples = "Red", rtmax = 17.25)
p.comb <- metabCombiner(xdata = p30, ydata = p20, binGap = 0.0075)

p.comb <- selectAnchors(p.comb, tolmz = 0.003, tolQ = 0.3, windy = 0.02)
p.comb <- fit_gam(p.comb, k = 20, iterFilter = 1)

#example: moderate m/z deviation, accurate rt fit, high sample similarity
p.comb <- calcScores(p.comb, A = 90, B = 14, C = 0.8, useAdduct = FALSE,
         groups = NULL, fit = "gam", usePPM = FALSE)
cTable = combinedTable(p.comb)  #to view results

#example 2: high m/z deviation, moderate rt fit, low sample similarity
p.comb <- calcScores(p.comb, A = 50, B = 8, C = 0.2)

#example 3: low m/z deviation, poor rt fit, moderate sample similarity
p.comb <- calcScores(p.comb, A = 120, B = 5, C = 0.5)

#example 4: using ppm for mass deviation; note different A value
p.comb <- calcScores(p.comb, A = 0.05, B = 14, C = 0.5, usePPM = TRUE)

#example 5: limiting to specific m/z groups 1-1000
p.comb <- calcScores(p.comb, A = 90, B = 14, C = 0.5, groups = seq(1,1000))

#example 6: using adduct information
p.comb <- calcScores(p.comb, A = 90, B = 14, C = 0.5, useAdduct = TRUE,
                     adduct = 1.25)
```

combinedTable *Obtain metabCombiner Feature Alignment Report*

Description

Obtain constructed table reporting every possible metabolomics feature alignment.

Usage

```
combinedTable(object)

## S4 method for signature 'metabCombiner'
combinedTable(object)
```

Arguments

`object`	metabCombiner object.

Value

Feature Pair Alignment report data.frame. The columns of the report are as follows:

`idx`	Identities of features from dataset X
`idy`	Identities of features from dataset Y
`mzx`	m/z values of features from dataset X
`mzy`	m/z values of features from dataset Y
`rtx`	retention time values of features from dataset X
`rty`	retention time values of features from dataset Y
`rtProj`	model-projected (X->Y) retention times values
`Qx`	abundance quantile values of features from dataset X
`Qy`	abundance quantile values of features from dataset Y
`group`	m/z feature group of feature pairing
`score`	computed similarity scores of feature pairing
`rankX`	ranking of pairing score for X dataset features
`rankY`	ranking of pairing score for Y dataset features
`adductX`	adduct label of features from dataset X
`adductY`	adduct label of features from dataset Y
`...`	Sample and extra columns from both datasets X & Y

Examples

```
data(plasma30)
data(plasma20)

p30 <- metabData(plasma30, samples = "CHEAR")
p20 <- metabData(plasma20, samples = "Red")

p.comb <- metabCombiner(p30, p20)
p.comb.table <- combinedTable(p.comb)
```

combinerCheck *Obtain Errors for metabCombiner Object Checks*

Description

This function stores and returns a customized error message when checking the validity of certain objects.

Usage

```
combinerCheck(errNo, type, error = "stop")
```

Arguments

errNo	integer error code.
type	character object type (either "combinedTable", "metabCombiner" or "metabData")
error	character. If "stop", gives an error message; if "warning", provides a warning message; if "silent", returns silently

Details

In certain functions, an object must be checked for correctness. A `metabData` must have a properly formatted dataset with the correct column names & types.A `metabCombiner` must have properly formatted `combinedTable`, with expected names and columns. If one of these conditions is not met, a non-zero numeric code is returned and this function is used to print a specific error message corresponding to the appropriate object and error code.

Value

A customized error message for specific object check.

crossValFit *Cross Validation for Model Fits*

Description

Helper function for `fit_gam()` & `fit_loess()`. Determines optimal value of `k` basis functions for Generalized Additive Model fits or `span` for loess fits from among user-defined choices, using a 10-fold cross validation minimizing mean squared error.

Usage

```
crossValFit(rts, fit, vals, bs, family, m, method, optimizer, loess.pars, ...)
```

Arguments

`rts` data.frame of ordered pair retention times

`fit` Either "gam" for GAM fits, or "loess" for loess fits

`vals` numeric vector: k values for GAM fits, spans for loess fits. Best value chosen by 10-fold cross validation.

`bs` character. Choice of spline method, either "bs" or "ps"

`family` character. Choice of mgcv family; see: ?mgcv::family.mgcv

`m` integer. Basis and penalty order for GAM; see ?mgcv::s

`method` character. Smoothing parameter estimation method; see: ?mgcv::gam

`optimizer` character. Method to optimize smoothing parameter; see: ?mgcv::gam

`loess.pars` parameters for LOESS fitting; see ?loess.control

`...` Other arguments passed to `mgcv::gam`.

Value

Optimal parameter value as determined by 10-fold cross validation

`detectFields` *Detect metabData Input Columns*

Description

This function ensures that metabolomics datasets used as inputs for the program possess all of the required fields, plus any optional columns that may appear in the final report table.

Usage

```
detectFields(Data, table, mz, rt, id, adduct, samples, extra, Q)
```

Arguments

`Data` a `metabData` object.

`table` data frame containing metabolomics features or path to metabolomics data file.

`mz` Character name(s) / regular expressions associated with data column containing m/z values. The first column whose name contains this expression will be selected for analysis.

`rt` Character name(s) / regular expression associated with data column containing retention time values. The first column whose name contains this expression will be selected for analysis.

`id` Character name(s) or regular expression associated with data column containing metabolomics feature identifiers. The first column whose name contains this expression will be selected for analysis.

`adduct` Character name(s) or regular expression associated with data column containing adduct, formula, or additional annotations. The first column whose name contains this expression will be selected for analysis.

`samples` Character names of columns containing sample values. All numeric columns containing these keywords are selected for analysis. If no keywords given, searches for longest stretch of numeric columns remaining.

`extra` Character names of columns containing additional feature information, e.g. non-analyzed sample values. All columns containing these keywords are selected for analysis.

`Q` Character name(s) or regular expression associated with numeric feature abundance quantiles.

Value

an initialized and formatted `metabData` object.

`evaluateParams` *Evaluate Similarity Score Parameters*

Description

This function provides a method for guiding selection of suitable values for A, B, & C weight arguments in the `calcScores` method, based on the similarity scores of shared identified compounds. Datasets must have at least one identity in common (i.e. idx = idy, case-insensitive), and preferably more than 10.

Usage

```
evaluateParams(
  object,
  A = seq(60, 150, by = 10),
  B = seq(6, 15),
  C = seq(0.1, 0.5, by = 0.1),
  fit = c("gam", "loess"),
  usePPM = FALSE,
  minScore = 0.5,
  penalty = 5,
  groups = NULL,
  brackets_ignore = c("(", "[", "{")
)
```

Arguments

`object` metabCombiner object

`A` Numeric weights for penalizing m/z differences.

`B` Numeric weights for penalizing differences between fitted & observed retention times

`C` Numeric weight for differences in Q (abundance quantiles).

`fit` Character. Choice of fitted rt model, "gam" or "loess."

`usePPM` logical. Option to use relative parts per million (ppm) as opposed to absolute) m/z differences in score computations.

`minScore`	numeric minimum score to count towards objective function calculation for known matching features (idx = idy) and mismatches.
`penalty`	numeric. Subtractive mismatch penalty.
`groups`	integer. Vector of feature groups to score. If set to NULL (default), will compute scores for all feature groups.
`brackets_ignore`	bracketed identity and adduct character strings of these types will be ignored according to this argument

Details

This uses an objective function, based on the accurate and inaccurate alignments of shared pre-identified compounds. For more details, see: `objective`.

Value

A data frame with the following columns:

`A`	m/z weight values
`B`	rt weight values
`C`	Q weight values
`score`	objective function evaluation of (A,B,C) weights

Note

In contrast to `calcScores` function, A, B, & C take numeric vectors as input, as opposed to constants. The total number of rows in the output will be equal to the products of the lengths of these input vectors

See Also

`calcScores`, `objective`

Examples

```
data(plasma30)
data(plasma20)

p30 <- metabData(plasma30, samples = "CHEAR")
p20 <- metabData(plasma20, samples = "Red", rtmax = 17.25)
p.comb = metabCombiner(xdata = p30, ydata = p20, binGap = 0.0075)

p.comb = selectAnchors(p.comb, windx = 0.03, windy = 0.02)
p.comb = fit_gam(p.comb, k = 20, iterFilter = 2)

#example 1
scores = evaluateParams(p.comb, A = seq(60,100,10), B = seq(10,15), C = 0.5,
    minScore = 0.7, penalty = 10)

##example 2: using PPM mass deviation (note change to A argument)
scores = evaluateParams(p.comb, usePPM = TRUE, A = seq(0.01,0.05,0.01))

##example 3: limiting to groups 1-2000
```

```
scores = evaluateParams(p.comb, minScore = 0.5, groups = 1:2000)
```

filterAnchors *Filter Outlier Ordered Pairs*

Description

Helper function for fit_gam & fit_loess. It filters the set of ordered pairs using the residuals calculated from multiple GAM / loess fits.

Usage

```
filterAnchors(
  rts,
  fit,
  vals,
  iterFilter,
  ratio,
  frac,
  bs,
  m,
  family,
  method,
  optimizer,
  loess.pars,
  ...
)
```

Arguments

rts	Data frame of ordered retention time pairs.
fit	Either "gam" for GAM fits, or "loess" for loess fits
vals	numeric values: k values for GAM fits, spans for loess fits
iterFilter	integer number of residual filtering iterations
ratio	numeric. A point is an outlier if the ratio of residual to mean residual of a fit exceeds this value. Must be greater than 1.
frac	numeric. A point is excluded if deemed a residual in more than this fraction value times the number of fits. Must be between 0 & 1.
bs	character. Choice of spline method from mgcv; either "bs" or "ps"
m	integer. Basis and penalty order for GAM; see ?mgcv::s
family	character. Choice of mgcv family; see: ?mgcv::family.mgcv
method	character. Smoothing parameter estimation method; see: ?mgcv::gam
optimizer	character. Method to optimize smoothing parameter; see: ?mgcv::gam
loess.pars	parameters for LOESS fitting; see ?loess.control
...	other arguments passed to mgcv::gam.

Value

anchor rts data frame with updated weights.

filterRT *Filter Features by Retention Time*

Description

Restricts input metabolomics feature table in `metabData` object to a range of retention times defined by `rtmin` & `rtmax`.

Usage

```
filterRT(data, rtmin, rtmax)
```

Arguments

`data`	formatted metabolomics data frame.
`rtmin`	lower range of retention times for analysis. If "min", defaults to minimum observed retention time. .
`rtmax`	upper range of retention times for analysis. If "max", defaults to maximum observed retention time.

Details

Retention time restriction is often recommended to aid the analysis of comparable metabolomics datasets. The beginning and end of a chromatogram typically contain features that do not correspond with true biological compounds derived from the sample. `rtmin` and `rtmax` should be set slightly before and slightly after the first and last commonly observed metabolites, respectively.

Value

A data frame of metabolomics features, limited to time window `rtmin` $\leq$ rt $\leq$ `rtmax`)

findDuplicates *Find and Remove Duplicate Features*

Description

Pairs of features with nearly identical m/z and retention time values are removed in this step.

Usage

```
findDuplicates(data, missing, counts, duplicate)
```

Arguments

`data`	Constructed metabolomics data frame.
`missing`	Numeric vector. Percent missingness for each feature.
`counts`	Numeric vector. Central measure for each feature.
`duplicate`	Ordered numeric pair (m/z, rt) tolerance parameters for duplicate feature search.

Details

Pairs of features are deemed duplicates if pairwise differences in m/z & rt fall within tolerances defined by the `duplicate` argument. If a pair of duplicate features is found, one member is removed. The determination of which feature to remove is first by percent missingness, followed by central abundance measure (median or mean). If the features have equal missingness and abundance, then row order determines the feature to be removed.

Value

integer indices of removable duplicate features

`fit_gam` *Fit RT Projection Model With GAMs*

Description

Fits a (penalized) basis splines curve through a set of ordered pair retention times, modeling one set of retention times (rty) as a function on the other set (rtx).Filtering iterations of high residual points are performed first. Multiple acceptable values of `k` can be supplied used, with one value selected through 10-fold cross validation.

Usage

```
fit_gam(
  object,
  useID = FALSE,
  k = seq(10, 20, by = 2),
  iterFilter = 2,
  ratio = 2,
  frac = 0.5,
  bs = c("bs", "ps"),
  family = c("scat", "gaussian"),
  weights = 1,
  m = c(3, 2),
  method = "REML",
  optimizer = "newton",
  ...
)
```

Arguments

`object`	a metabCombiner object.
`useID`	logical. Option to use matched IDs to inform fit
`k`	integer vector values controling the number of basis functions for GAM construction. Best value chosen by 10-fold cross validation.
`iterFilter`	integer number of residual filtering iterations to perform
`ratio`	numeric. A point is an outlier if the ratio of residual to mean residual of a fit exceeds this value. Must be greater than 1.
`frac`	numeric. A point is excluded if deemed a residual in more than this fraction value times the number of fits. Must be between 0 & 1.

`bs` character. Choice of spline method from mgcv, either "bs" (basis splines) or "ps" (penalized basis splines)

`family` character. Choice of mgcv family; see: ?mgcv::family.mgcv

`weights` Optional user supplied weights for each ordered pair. Must be of length equal to number of anchors (n) or a divisor of (n + 2).

`m` integer. Basis and penalty order for GAM; see ?mgcv::s

`method` character. Smoothing parameter estimation method; see: ?mgcv::gam

`optimizer` character. Method to optimize smoothing parameter; see: ?mgcv::gam

`...` Other arguments passed to `mgcv::gam`.

Details

A set of ordered pair retention times must be previously computed using `selectAnchors()`. The minimum and maximum retention times from both input datasets are included in the set as ordered pairs (min_rtx, min_rty) & (max_rtx, max_rty).

The `weights` argument initially determines the contribution of each point to the model fits; they are equally weighed by default, but can be changed using an `n+2` length vector, where `n` is the number of ordered pairs and the first and last of the weights determines the contribution of the min and max ordered pairs.

The model complexity is determined by `k`. Multiple values of `k` are allowed, with the best value chosen by 10 fold cross validation. Before this happens, certain ordered pairs are removed based on the model errors. In each iteration, a GAM is fit using each selected value of `k`. A point is "removed" (its corresponding `weights` value set to 0) if its residual is `ratio` times average residual for a fraction of fitted models, as determined by `frac`. If an ordered pair is an "identity" (discovered in the `selectAnchors` by setting the `useID` to TRUE), then setting `useID` here will prevent its removal.

Other arguments, e.g. `family`, `m`, `optimizer`, `bs`, and `method` are GAM specific parameters. The `family` option is currently limited to the "scat" (scaled t) and "gaussian" families; scat family model fits are more robust to outliers than gaussian fits, but compute much slower. Type of splines are currently limited to basis splines ($bs = "bs"$) or penalized basis splines ($bs = "ps"$).

Value

metabCombiner with a fitted GAM model object

See Also

`selectAnchors`,`fit_loess`,

Examples

```
data(plasma30)
data(plasma20)

p30 <- metabData(plasma30, samples = "CHEAR")
p20 <- metabData(plasma20, samples = "Red", rtmax = 17.25)
p.comb = metabCombiner(xdata = p30, ydata = p20, binGap = 0.0075)

p.comb = selectAnchors(p.comb, tolmz = 0.003, tolQ = 0.3, windy = 0.02)
anchors = getAnchors(p.comb)
```

```
#version 1: using faster, but less robust, gaussian family
p.comb = fit_gam(p.comb, k = c(10,12,15,17,20), frac = 0.5,
    family = "gaussian")


#version 2: using slower, but more robust, scat family
p.comb = fit_gam(p.comb, k = seq(12,20,2), family = "scat",
                     iterFilter = 1, ratio = 3, method = "GCV.Cp")

#version 3 (with identities)
p.comb = selectAnchors(p.comb, useID = TRUE)
anchors = getAnchors(p.comb)
p.comb = fit_gam(p.comb, useID = TRUE, k = seq(12,20,2), iterFilter = 1)

#version 4 (using identities and weights)
weights = ifelse(anchors$labels == "I", 2, 1)
p.comb = fit_gam(p.comb, useID = TRUE, k = seq(12,20,2),
                     iterFilter = 1, weights = weights)

#version 5 (assigning weights to the boundary points
weights = c(2, rep(1, nrow(anchors)), 2)
p.comb = fit_gam(p.comb, k = seq(12,20,2), weights = weights)

#to preview result of fit_gam
plot(p.comb, xlab = "CHEAR Plasma (30 min)",
     ylab = "Red-Cross Plasma (20 min)", pch = 19,
     main = "Example fit_gam Result Fit")
```

fit_loess *Fit RT Projection Model With LOESS*

Description

Fits a local regression smoothing spline through a set of ordered pair retention times. modeling one set of retention times (rty) as a function on the other set (rtx). Filtering iterations of high residual points are performed first. Multiple acceptable values of span can be used, with one value selected through 10-fold cross validation.

Usage

```
fit_loess(
  object,
  useID = FALSE,
  spans = seq(0.2, 0.3, by = 0.02),
  iterFilter = 2,
  ratio = 2,
  frac = 0.5,
  iterLoess = 10,
  weights = 1
)
```

Arguments

`object`	metabCombiner object.
`useID`	logical. Option to use matched IDs to inform fit
`spans`	numeric span values (between 0 & 1) used for loess fits
`iterFilter`	integer number of residual filtering iterations to perform
`ratio`	numeric. A point is an outlier if the ratio of residual to mean residual of a fit exceeds this value. Must be greater than 1.
`frac`	numeric. A point is excluded if deemed a residual in more than this fraction value times the number of fits. Must be between 0 & 1.
`iterLoess`	integer. Number of robustness iterations to perform in `loess()`.See ?loess.control for more details.
`weights`	Optional user supplied weights for each ordered pair. Must be of length equal to number of anchors (n) or a divisor of (n + 2).

Value

`metabCombiner` object with `model` slot updated to contain the fitted loess model

See Also

`selectAnchors`,`fit_gam`

Examples

```
data(plasma30)
data(plasma20)

p30 <- metabData(plasma30, samples = "CHEAR")
p20 <- metabData(plasma20, samples = "Red", rtmax = 17.25)
p.comb = metabCombiner(xdata = p30, ydata = p20, binGap = 0.0075)
p.comb = selectAnchors(p.comb, tolmz = 0.003, tolQ = 0.3, windy = 0.02)

#version 1
p.comb = fit_loess(p.comb, spans = seq(0.2,0.3,0.02), iterFilter = 1)

#version 2 (using weights)
anchors = getAnchors(p.comb)
weights = c(2, rep(1, nrow(anchors)), 2)  #weight = 2 to boundary points
p.comb = fit_loess(p.comb, spans = seq(0.2,0.3,0.02), weights = weights)

#version 3 (using identities)
p.comb = selectAnchors(p.comb, useID = TRUE, tolmz = 0.003)
p.comb = fit_loess(p.comb, spans = seq(0.2,0.3,0.02), useID = TRUE)

#to preview result of fit_loess
plot(p.comb, fit = "loess", xlab = "CHEAR Plasma (30 min)",
     ylab = "Red-Cross Plasma (20 min)", pch = 19,
     main = "Example fit_loess Result Fit")
```

`formCombinedTable` *Form Combiner Report Table*

Description

Takes previously computed m/z groups using `mzGroup()` and creates merged `combinedTable` consisting of all possible feature alignments, all initialized with equivalent score and ranking.

Usage

```
formCombinedTable(object, xset, yset, nGroups)
```

Arguments

`object`	metabCombiner object
`xset`	data frame. A processed metabolomics feature table.
`yset`	data frame. A processed metabolomics feature table.
`nGroups`	integer. Total number of computed feature groups

Value

metabCombiner object with initialized combinedTable data frame.

`getAnchors` *Get Ordered Retention Time Pairs*

Description

This returns the data frame of feature alignments used to anchor a retention time projection model, constructed by `selectAnchors`.

Usage

```
getAnchors(object)

## S4 method for signature 'metabCombiner'
getAnchors(object)
```

Arguments

`object`	metabCombiner object

Value

Data frame of anchor features

See Also

`selectAnchors`

Examples

```
data(plasma30)
data(plasma20)

p30 <- metabData(plasma30, samples = "CHEAR")
p20 <- metabData(plasma20, samples = "Red")

p.comb <- metabCombiner(p30, p20)
p.comb <- selectAnchors(p.comb, windx = 0.05, windy = 0.03)

anchors <- getAnchors(p.comb)
```

getCoefficients *Obtain Last-Used Score Coefficients*

Description

Provides the last used weight arguments from `calcScores()` function. Returns empty list if `calcScores()` has not yet been called.

Usage

```
getCoefficients(object)

## S4 method for signature 'metabCombiner'
getCoefficients(object)
```

Arguments

`object`	metabCombiner object

Value

A list of the last used weight parameters:

A	Specific weight penalizing feature m/z differences
B	Specific weight penalizing retention time projection error
C	Specific weight penalizing differences in abundance quantiles

Examples

```
data(plasma30)
data(plasma20)

p30 <- metabData(plasma30, samples = "CHEAR")
p20 <- metabData(plasma20, samples = "Red")

p.comb <- metabCombiner(p30, p20)
p.comb <- selectAnchors(p.comb, windx = 0.05, windy = 0.03)
p.comb <- fit_gam(p.comb, k = 20, iterFilter = 1)
p.comb <- calcScores(p.comb, A = 90, B = 14, C = 0.5)
```

```
getCoefficients(p.comb)
```

getData *Get Processed Dataset*

Description

Get Processed Dataset

Usage

```
getData(object)

## S4 method for signature 'metabData'
getData(object)
```

Arguments

object	metabData object

Value

Single Metabolomics Data Frame

Examples

```
data(plasma30)

p30 <- metabData(plasma30, samples = "CHEAR")
data <- getData(p30)
```

getExtra *Get Extra Data Column Names*

Description

Get Extra Data Column Names

Usage

```
getExtra(object, data = c("x", "y"))

## S4 method for signature 'metabData'
getExtra(object)
```

Arguments

object	metabCombiner or metabData object
data	Choice of input dataset, ’x’ or ’y’

Value

character vector of extra column names

Examples

```
data(plasma30)
p30 <- metabData(plasma30, samples = "CHEAR", extra = "Red")
getExtra(p30)
```

---

getModel *Get Fitted RT Model*

Description

Returns the last fitted RT projection model from a metabCombiner object of type "gam" or "loess".

Usage

```
getModel(object, fit = c("gam", "loess"))

## S4 method for signature 'metabCombiner'
getModel(object, fit = c("gam", "loess"))
```

Arguments

object	metabCombiner object
fit	Choice of model, "gam" or "loess"

Value

nonlinear retention time fit object

See Also

fit_gam, fit_loess

Examples

```
data(plasma30)
data(plasma20)
p30 <- metabData(plasma30, samples = "CHEAR")
p20 <- metabData(plasma20, samples = "Red", rtmax = 17.25)
p.comb <- metabCombiner(xdata = p30, ydata = p20, binGap = 0.005)
p.comb <- selectAnchors(p.comb, tolmz = 0.003, tolQ = 0.3, windy = 0.02)
p.comb <- fit_gam(p.comb, iterFilter = 1, k = 20)
p.comb <- fit_loess(p.comb, iterFilter = 1, spans = 0.2)
model.gam <- getModel(p.comb, fit = "gam")
model.loess <- getModel(p.comb, fit = "loess")
```

getSamples *Get Sample Names From metabCombiner or metabData Object*

Description

Returns the sample names from one of the two datasets used in metabCombiner analysis, denoted as 'x' or 'y.'

Usage

```
getSamples(object, data = c("x", "y"))

## S4 method for signature 'metabCombiner'
getSamples(object, data = c("x", "y"))

## S4 method for signature 'metabData'
getSamples(object)
```

Arguments

`object`	A metabData object
`data`	Character. One of either 'x' or 'y'.

Value

character vector of sample names. For `metabCombiner` objects these may come from the 'x' dataset (if `data` = "x") or the 'y' dataset (if `data` = "y").

names of samples of formatted dataset.

Examples

```
data(plasma30)
data(plasma20)

p30 <- metabData(plasma30, samples = "CHEAR")
p20 <- metabData(plasma20, samples = "Red", rtmax = 17.25)

p.comb <- metabCombiner(xdata = p30, ydata = p20)

getSamples(p30)
getSamples(p.comb, data = "x")  #equivalent to previous
getSamples(p20)
getSamples(p.comb, data = "y")  #equivalent to previous
```

getStats *Get Object Statistics*

Description

Prints out a list of object-specific statistics for both `metabCombiner` and `metabData` objects

Usage

```
getStats(object)

## S4 method for signature 'metabCombiner'
getStats(object)

## S4 method for signature 'metabData'
getStats(object)
```

Arguments

`object`	metabCombiner or metabData object

Value

list of object-specific statistics

Methods (by class)

- `metabCombiner`: Method for ’metabCombiner’ object

Examples

```
data(plasma30)
data(plasma20)
p30 <- metabData(plasma30, samples = "CHEAR")
p20 <- metabData(plasma20, samples = "Red", rtmax = 17.25)

getStats(p30) #metabData stats

p.comb <- metabCombiner(xdata = p30, ydata = p20, binGap = 0.005)
p.comb <- selectAnchors(p.comb, tolmz = 0.003, tolQ = 0.3, windy = 0.02)
p.comb <- fit_gam(p.comb, iterFilter = 1, k = 20)

getStats(p.comb) #metabCombiner stats
```

`identityAnchorSelection` *Select Matching Ids as Anchors*

Description

This is an optional helper function for `selectAnchors`. Uses identities to guide selection of ordered retention time pairs. If useID option is set to TRUE, it will select pairs of features with matching ID character strings before proceeding with iterative anchor selection.

Usage

```
identityAnchorSelection(cTable, windx, windy, useID, brackets)
```

Arguments

`cTable`	data frame, contains only feature ids, mzs, rts, Qs, & labels
`windx`	numeric positive retention time exclusion window in X dataset
`windy`	numeric positive retention time exclusion window in Y dataset
`useID`	logical. Operation proceeds if TRUE, terminates otherwise.
`brackets`	If useID = TRUE, bracketed identity strings of the types included in this argument will be ignored

Details

Identity anchors are allowed to violate constraints of m/z, Q, and rtq difference tolerances, and will not be removed if they fall within a rt exclusion window of other features. If a name appears more than once, only the pair with the highest relative abundance is selected.

Value

combinedTable with updated anchor labels

See Also

`selectAnchors`

`isCombinedTable` *Determine combinedTable Validity*

Description

Checks whether input object is a valid metabData.Returns an integer code if invalid. Function is used alongside `combinerCheck`.

Usage

```
isCombinedTable(object)
```

Arguments

`object`	Any R object.

Value

0 if object is a valid Combiner Table; an integer code otherwise

`isMetabCombiner` *Determine if object is a valid metabCombiner object*

Description

Checks whether input object is a valid metabCombiner.Returns an integer code if invalid. Function is used alongside `combinerCheck`.

Usage

```
isMetabCombiner(object)
```

Arguments

`object`	Any R object.

Value

0 if object is a valid metabData object; an integer code otherwise

`isMetabData` *Determine validity of input metabData object*

Description

Checks whether input object is a valid metabData.Returns an integer code if invalid. Function is used alongside `combinerCheck`.

Usage

```
isMetabData(object)
```

Arguments

`object`	Any R object

Value

0 if object is a valid metabData object; an integer code otherwise.

`iterativeAnchorSelection` *Iterative Selection of Ordered Pairs*

Description

This is a helper function for `selectAnchors`. Anchors are iteratively selected from highly abundant feature pairs, subject to feature m/z, rt, & Q constraints set by the user.

Usage

```
iterativeAnchorSelection(cTable, windx, windy, swap = FALSE)
```

Arguments

`cTable`	data frame, contains only feature ids, mzs, rts, Qs, & labels
`windx`	numeric positive retention time exclusion window in X dataset.
`windy`	numeric positive retention time exclusion windown in Y dataset.
`swap`	logical. When FALSE, searches for abundant features in dataset X, complemented by dataset Y features; when TRUE, searches for abundant features in dataset Y, complemented by dataset X features.

Value

data frame of anchor feature alignments.

See Also

`selectAnchors`

`labelRows` *Annotate and Remove Report Rows*

Description

Method for annotation of identity-matched, removable, & conflicting feature pair alignments (FPAs) in `combinedTable`. FPAs that fall within some small measure (in score or mz/rt) of the top-ranked FPA may require further inspection are organized into subgroups. .

Usage

```
labelRows(
  object,
  maxRankX = 3,
  maxRankY = 3,
  minScore = 0.3,
  conflict,
  method = c("score", "mzrt"),
  balanced = TRUE,
  remove = FALSE,
  brackets_ignore = c("(", "[", "{")
)
```

Arguments

`object`	Either a `metabCombiner` object or `combinedTable`.
`maxRankX`	Integer. Maximum allowable rank for X dataset features.
`maxRankY`	Integer. Maximum allowable rank for Y dataset features.
`minScore`	Numeric. Minimum allowable score (between 0 & 1) for metabolomics FPAs.
`conflict`	numeric used to determine subgroups. If method = "score", a constant (between 0 & 1) score difference between a pair of conflicting FPAs. If method = "mzrt", a length 4 numeric: (m/z, rt, m/z, rt) tolerances, the first pair for X dataset features and the second pair for Y dataset features.
`method`	Conflict detection method. If equal to "score" (default), assigns a conflict subgroup if score of lower-ranking FPA is within some tolerance of higher-ranking FPA. If set to "mzrt", assigns a conflicting subgroup if within a small m/z & rt distance of the top-ranked FPA.
`balanced`	Logical. Optional processing of "balanced" groups, defined as groups with an equal number of features from input datasets where all features have a 1-1 match.
`remove`	Logical. Option to keep or discard rows deemed removable.
`brackets_ignore`	character. Bracketed identity strings of the types in this argument will be ignored

Details

`metabCombiner` initially reports all possible FPAs in the rows of the `combinedTable` report. Most of these are misalignments that require removal. This function is used to automate most of the reduction process by labeling rows as removable or conflicting, based on certain conditions, and is performed after computing similarity scores.

A label may take on one of four values:

a) "": No determination made b) "IDENTITY": an alignment with matching identity "idx & idy" strings c) "REMOVE": a row determined to be a misalignment d) "CONFLICT": competing alignments for one or multiple shared features

The labeling rules are as follows: 1) Rows with matching idx & idy strings are labeled "IDENTITY". These rows are not labeled "REMOVE", irrespective of subsequent criteria. 2) Groups determined to be 'balanced': label rows with rankX > 1 & rankY > 1 "REMOVE" irrespective of `conflict` criteria 3) Rows with a score < `minScore`: label "REMOVE" 4) Rows with rankX > `maxRankX` and/or rankY > `maxRankY`: label "REMOVE" 5) Conflicting subgroup assignment as determined by `method` & `conflict` arguments. Conflicting alignments following outside `conflict` thresholds: labeled "REMOVE". Otherwise,

Value

updated `combinedTable` or `metabCombiner` object. The table will have three new columns:

`labels`	characterization of feature alignments as described
`subgroup`	conflicting subgroup number of feature alignments
`alt`	alternate subgroup for rows in multiple feature pair conflicts

Examples

```
data(plasma30)
data(plasma20)

p30 <- metabData(plasma30, samples = "CHEAR")
p20 <- metabData(plasma20, samples = "Red", rtmax = 17.25)
p.comb = metabCombiner(xdata = p30, ydata = p20, binGap = 0.0075)
p.comb = selectAnchors(p.comb, tolmz = 0.003, tolQ = 0.3, windy = 0.02)
p.comb = fit_gam(p.comb, k = 20, iterFilter = 1)
p.comb = calcScores(p.comb, A = 90, B = 14, C = 0.5)
cTable = combinedTable(p.comb)

##example using score-based conflict detection method
lTable = labelRows(cTable, maxRankX = 3, maxRankY = 2, minScore = 0.5,
    method = "score", conflict = 0.2)

##example using mzrt-based conflict detection method
lTable = labelRows(cTable, method = "mzrt", maxRankX = 3, maxRankY = 2,
                     conflict = c(0.005, 1, 0.005,0.5))
```

metabCombiner *Form a metabCombiner object.*

Description

metabCombiner() takes two metabolomics featurelists, contained in metabData objects and constructs a merged dataset containing groups of features detected in both featurelists with similar m/z values.

Takes two untargeted metabolomics feature lists (consisting of m/z, rt, and sample intensity measurements, plus optional identifiers & adduct labels) and outputs a merged feature list consisting of potential compound matches, ranked by a similarity score for groups of features. Inputs are assumed to be derived from biologically similar samples analyzed with a similar analytical method.

Usage

```
metabCombiner(xdata, ydata, binGap = 0.005)
```

Arguments

xdata	metabData object. One of two datasets to be combined.
ydata	metabData object. One of two datasets to be combined.
binGap	numeric. Parameter used for grouping features by m/z. See ?mzGroup for more details.

Details

The binGap argument defines consecutive differences between grouped m/z features. metabCombiner objects are used for all subsequent processing steps.

Value

a metabCombiner object

Examples

```
data(plasma30)
data(plasma20)

p30 <- metabData(plasma30, samples = "CHEAR")
p20 <- metabData(plasma20, samples = "Red", rtmax = 17.25)

p.comb = metabCombiner(xdata = p30, ydata = p20, binGap = 0.0075)
```

metabCombiner-class *'metabCombiner' Combined Metabolomics Dataset Class*

Description

This is the main object for the metabCombiner package workflow. This object holds a combined feature table, along with a retention time projection model, the ordered pair anchors used to generate this model, and key object statistics.

Slots

- `combinedTable` data.frame displaying all feature pair alignments, combining measurements of all possible shared compounds
- `nonmatched` list of data frames consisting of nonmatched features
- `anchors` data.frame of of feature alignments used for rt modeling
- `model` list containing the last fitted nonlinear model(s)
- `coefficients` list of last used A,B,C similarity weight values
- `samples` list of sample name vectors from input datasets
- `extra` list of extra column name vectors from input datasets
- `stats` set of useful metabCombiner statistics

metabData *Constructor for the metabData object.*

Description

This is a constructor for objects of type `metabData`.

Usage

```
metabData(
  table,
  mz = "mz",
  rt = "rt",
  id = "id",
  adduct = "adduct",
  samples = NULL,
  Q = NULL,
  extra = NULL,
  rtmin = "min",
  rtmax = "max",
  misspc = 50,
  measure = c("median", "mean"),
  zero = FALSE,
  duplicate = c(0.0025, 0.05)
)
```

Arguments

`table` Path to file containing feature table or data.frame object containing features

`mz` Character name(s) or regular expression associated with data column containing m/z values. The first column whose name contains this expression will be selected for analysis.

`rt` Character name(s) or regular expression associated with data column containing retention time values. The first column whose name contains this expression will be selected for analysis.

`id` Character name(s) or regular expression associated with data column containing metabolomics feature identifiers. The first column whose name contains this expression will be selected for analysis.

`adduct` Character name(s) or regular expression associated with data column containing adduct or chemical formula annotations. The first column whose name contains this expression will be selected for analysis.

`samples` Character name(s) or regular expression associated with data columns. All numeric columns whose names contain these keywords are selected for analysis. If no keywords given, program searches longest stretch of remaining numeric columns.

`Q` Character name(s) or regular expression associated with numeric feature abundance quantiles. If NULL, abundance quantiles are calculated from sample intensities.

`extra` Character names of columns containing additional feature information, e.g. non-analyzed sample values. All columns containing these keywords selected and will be displayed in the final output.

`rtmin` Numeric. Minimum retention time for analysis.

`rtmax` Numeric. Maximum retention time for analysis.

`misspc` Numeric. Threshold missingness percentage for analysis.

`measure` Central quantitation measure, either "median" or "mean".

`zero` Logical. Whether to consider zero values as missing.

`duplicate`	Numeric ordered pair (m/z, rt) duplicate feature tolerances. Pairs of features within these tolerances are deemed duplicates and one of the pair is removed (see: `findDuplicates`)

Details

Processed metabolomics feature table must contain columns for m/z, rt, and numeric sample intensities. Some optional fields such as identity `id` and `adduct` label columns may also be supplied. Non-analyzed columns can be included into the final output by specifying the names of these columns in the `extra` argument. All required arguments are checked for validity (e.g. no negative m/z or rt values, each column is used at most once, column types are valid, etc...).

Following this is a pre-analysis filtering of rows that are either: 1) Outside of a specified retention time range (`rtmin`,`rtmax`), 2) Missing in excess of `misspc` percent of analyzed samples, or 3) deemed duplicates by small pairwise <m/z, rt> differences as specified by the `duplicate` argument.

Remaining features are ranked by abundance quantiles, Q, using a central `measure`, either "median" or "mean." Alternatively, the abundance quantiles column can be specified in the argument `Q`.

Value

An object of class metabData containing the specific information specified by `mz,rt,samples,id,adduct,Q,and` `extra` arguments, and adjusted by pre-processing steps.

Examples

```
data(plasma30)

#samples: CHEAR; RedCross samples non-analyzed "extra" columns
p30 <- metabData(plasma30, mz = "mz", rt = "rt", id = "identity",
                 adduct = "adduct", samples = "CHEAR", extra = "RedCross")

getSamples(p30)  #should print names of 5 CHEAR Sample column names
getExtra(p30)    #should print names of 5 Red Cross Sample column names

#equivalent to above
p30 <- metabData(plasma30, id = "id", samples = "CHEAR", extra = "Red")

#analyzing Red Cross samples with retention time limitations (0.5-17.5min)
p30 <- metabData(plasma30, samples = "Red", rtmin = 0.5, rtmax = 17.5)
data = getData(p30)
range(data$rt)

#using regular expressions for field searches
p30.2 <- metabData(plasma30, id = "identity|id|ID", samples = ".[3-5]$")
getSamples(p30.2)    #should print all column names ending in .3, .4, .5
```

`metabData-class` *'metabData' Single Metabolomics Dataset Class*

Description

This class is designed to process and format input metabolomics feature tables. It stores the information from individual metabolomics datasets, including the formatted feature table, sample names, and feature statistics.

Slots

- `data` formatted metabolomics data frame.
- `samples` character vector of analyzed sample names
- `extra` character vector of non-analyzed columns names
- `stats` A list of dataset statistics

`mzGroup` *Binning of mass spectral features in m/z dimension*

Description

Features in two input feature lists are grouped by their m/z values.

Usage

```
mzGroup(xset, yset, binGap)
```

Arguments

`xset`	data frame containing metabolomics features
`yset`	data frame containing metabolomics features
`binGap`	numeric gap value between consecutive sorted & pooled feature m/z values.

Details

The m/z values from both datasets are pooled, sorted, and binned by the `binGap` argument. Feature groups form when there is at least one pair of features from both datasets whose consecutive difference is less than `binGap`. Grouped features are joined together in `combinedTable` data report.

Value

list object containing updated xset & yset with group information

`nonmatched` *Get Nonmatched Features*

Description

Features that lack a any counterparts in the complementary dataset may be obtained from this method.

Usage

```
nonmatched(object, data = c("x", "y"))

## S4 method for signature 'metabCombiner'
nonmatched(object, data = c("x", "y"))
```

Arguments

`object`	metabCombiner object
`data`	Either one of 'x' or 'y', specifying which dataset's nonmatched features to return.

Value

If data is "x", returns non-matched X features ; if "y", returns non-matched Y features

Examples

```
data(plasma30)
data(plasma20)

p30 <- metabData(plasma30, samples = "CHEAR")
p20 <- metabData(plasma20, samples = "Red", rtmax = 17.25)
p.comb <- metabCombiner(xdata = p30, ydata = p20, binGap = 0.005)

nnmx <- nonmatched(p.comb, data = "x")
nnmy <- nonmatched(p.comb, data = "y")
```

`objective` *Weight Parameter Objective Function*

Description

This function evaluates the A, B, C weight parameters in terms of score separability of matching versus mismatching compound alignments. Higher objective function value imply a superior weight parameter selection.

Usage

```
objective(
  cTable,
  idtable,
  A,
  B,
  C,
  minScore,
  mzdiff,
  rtdiff,
  qdiff,
  rtrange,
  adductdiff,
  penalty,
  matches,
  mismatches
)
```

Arguments

`cTable` data frame. Abridged `metabCombiner` report table.

`idtable` data frame containing all evaluated identities

`A` Numeric weight for penalizing m/z differences.

`B` Numeric weight for penalizing differences between fitted & observed retention times

`C` Numeric weight for differences in Q (abundance quantiles).

`minScore` numeric. Minimum score to count towards objective value.

`mzdiff` numeric differences between feature m/z values

`rtdiff` Differences between model-projected retention time value & observed retention time

`qdiff` Difference between feature quantile Q values.

`rtrange` range of dataset Y retention times

`adductdiff` Numeric divisors of computed score when non-empty adduct labels do not match

`penalty` positive numeric penalty wherever S(i,j) > S(i,i), i =/= j

`matches` integer row indices of identity matches

`mismatches` list of integer identity row mismatches for each identity

Details

First, the similarity scores between all grouped features are calculated as described in `scorePairs`

Then, the objective value for a similarity S is evaluated as:

$$OBJ(S) = \sum h(S(i,i)) - h(S(i,j)) - p(S(i,i) > S(i,j))$$

-S(i,i) represents the similarity between correct identity alignments
-S(i,j), represents the maximum similarity of i to grouped feature j, i =/= j (the highest-scoring misalignment)
-h(x) = x if x > `minScore`, 0 otherwise
-p(COND) = 0 if the condition is true, and a `penalty` value otherwise

This is summed over all labeled compound identities (e.g. idx = idy) shared between input datasets.

Value

A numeric value quantifying total separability of compound match similarity scores from mismatch scores, given A,B,C values

`plasma20` *20 minute Metabolomics Analysis of Human Plasma*

Description

An example metabolomics dataset consisting of human plasma from Red Cross and CHEAR cohorts, plus pooled aliquots and blanks, acquired with a 20 minute total Reversed-Phase Liquid Chromatography & QTOF-MS instrument in the positive ionization mode.

Usage

```
data(plasma20)
```

Format

A data frame with 8910 rows and 22 columns:

`plasma30` *30 minute Metabolomics Analysis of Human Plasma*

Description

An example metabolomics dataset consisting of human plasma from Red Cross and CHEAR cohorts, plus pooled aliquots and blanks, acquired with a 30 minute total Reversed-Phase Liquid Chromatography and a QTOF-MS instrument in the positive ionization mode.

Usage

```
data(plasma30)
```

Format

A data frame with 8910 rows and 22 columns

`plot,metabCombiner,ANY-method` *Plot metabCombiner Fits*

Description

This is a plotting method for metabCombiner objects. It displays ordered pairs and a curve fit computed using `fit_gam` or `fit_loess`, using base R graphics.

Usage

```
## S4 method for signature 'metabCombiner,ANY'
plot(x, y, ...)

plot_fit(
  object,
  fit = c("gam", "loess"),
  pcol,
  lcol,
  lwd,
  remove.outliers = FALSE,
  ...
)
```

Arguments

`x` metabCombiner object

`y` ...

`...` Other variables passed into graphics::plot

`object` metabCombiner object

`fit` choice of model (either "gam" or "loess").

`pcol` color of the points (ordered pairs) in the plot.

`lcol` color of the fitted line in the plot

`lwd` line width of the curve fit between anchor points

`remove.outliers` logical, option to exclude outlier anchors in plot

Value

no values returned

Examples

```
data(plasma30)
data(plasma20)

p30 <- metabData(plasma30, samples = "CHEAR")
p20 <- metabData(plasma20, samples = "Red", rtmax = 17.25)
p.comb = metabCombiner(xdata = p30, ydata = p20, binGap = 0.0075)
p.comb = selectAnchors(p.comb, tolmz = 0.003, tolQ = 0.3, windy = 0.02)
p.comb = fit_gam(p.comb, k = 20, iterFilter = 1)

##plot of GAM fit
plot(p.comb, main = "Example GAM Fit Plot", xlab = "X Dataset RTs",
     ylab = "Y Dataset RTs", pcol = "red", lcol = "blue", lwd = 5,
     fit = "gam", pch = 19, remove.outliers = TRUE)

grid(lwd =  2, lty = 3 ) #adding gridlines
```

scorePairs *Calculate Pairwise Alignment Scores*

Description

Helper function for calcScores & evaluateParams. Calculates a pairwise similarity score between grouped features using differences in m/z, rt, and Q.

Usage

```
scorePairs(A, B, C, mzdiff, rtdiff, qdiff, rtrange, adductdiff)
```

Arguments

A	Numeric weight for penalizing m/z differences.
B	Numeric weight for penalizing differences between fitted & observed retention times.
C	Numeric weight for differences in Q (abundance quantiles).
mzdiff	Numeric differences between feature m/z values
rtdiff	Differences between model-projected retention time value & observed retention time
qdiff	Difference between feature quantile Q values
rtrange	Range of dataset Y retention times
adductdiff	Numeric divisors of computed score when non-empty adduct labels do not match

Details

The score between two grouped features x & y is calculated as:

$$S = -exp(-A|mzx - mzy| - B|rty - rtproj|/rtrange - C|Qx - Qy|)$$

where `mzx` & `Qx` correspond to the m/z and abundance quantile values of feature x; `mzy`, `rty`, and `Qy` correspond to the m/z, retention time, and quantile values of feature y; `rtproj` is the model-projected retention time of feature x onto the Y dataset chromatogram and `rtrange` is the retention time range of the Y dataset chromatogram. `A`, `B`, `C` are non-negative constant weight parameters for penalizing m/z, rt, and Q differences. Values between 0 (no confidence alignment) and 1 (high confidence alignment).

Value

Numeric similarity score between 0 & 1

selectAnchors *Select Anchors for Nonlinear RT Model*

Description

A subset of possible alignments in the `combinedTable` are used as ordered pairs to anchor a retention time projection model. Alignments of abundant features are prominent targets for anchor selection, but shared identified features (i.e. feature pairs where idx = idy) may be used.

Usage

```
selectAnchors(
  object,
  useID = FALSE,
  tolmz = 0.003,
  tolQ = 0.3,
  tolrtq = 0.5,
  windx = 0.03,
  windy = 0.03,
  brackets_ignore = c("(", "[", "{")
)
```

Arguments

`object`	metabCombiner object.
`useID`	logical. Option to first search for IDs as anchors.
`tolmz`	numeric. m/z tolerance for prospective anchors
`tolQ`	numeric. Quantile Q tolerance for prospective anchors
`tolrtq`	numeric. Linear RT quantile tolerance for prosepctive anchors.
`windx`	numeric. Retention time exclusion window around each anchor in X dataset. Optimal values are between 0.01 and 0.05 min (1-3s)
`windy`	numeric. Retention time exclusion window around each anchor in dataset Y. Optimal values are between 0.01 and 0.05 min (1-3s)
`brackets_ignore`	If useID = TRUE, bracketed identity strings of the types included in this argument will be ignored.

Details

In order to map between two sets of retention times, a set of ordered pairs need to be selected for the spline fit. This function relies on mutually abundant features to select these ordered pairs. In iterative steps, the most abundant (as indicated by Q value) in one dataset is selected along with its counterpart, and all features within some retention time window specified by `windx` & `windy` arguments are excluded. This process is repeated until all features have been considered.

`tolQ` & `tolmz` arguments restrict to feature pairs that have differences in Q & m/z within these tolerances. `tolrtq` further limits to feature pairs those with relative differences in linear retention time quantiles, calculated as $rtqx = (rtx - min(rtx))/(max(rtx) - min(rtx))$ & $rtqy = (rty - min(rty))/(max(rty) - min(rty))$

Shared identities (in which idx & idy columns have matching, non-empty & non-bracketed strings) may be used if `useID` is set to TRUE. In this case, shared identities will be searched first and will not be subject to any of the restrictions in m/z, Q, or rt. The iterative process proceeds after processing of shared identities.

Value

`metabCombiner` object with updated `anchors` slot. This is a data.frame of feature pairs that shall be used to map between retention times using a GAM or LOESS model.

`idx`	identities of features from dataset X
`idy`	identities of features from dataset Y
`mzx`	m/z values of features from dataset X
`mzy`	m/z values of features from dataset Y
`rtx`	retention time values of features from dataset X
`rty`	retention time values of features from dataset Y
`rtProj`	model-projected retention time values from X to Y
`Qx`	abundance quantile values of features from dataset X
`Qy`	abundance quantile values of features from dataset Y
`adductX`	adduct label of features from dataset X
`adductY`	adduct label of features from dataset Y
`group`	m/z feature group of feature pairing
`labels`	anchor labels; "I" for identity, "A" for normal anchors

See Also

`getAnchors`, `fit_gam`, `fit_loess`

Examples

```
data(plasma30)
data(plasma20)

p30 <- metabData(plasma30, samples = "CHEAR")
p20 <- metabData(plasma20, samples = "Red", rtmax = 17.25)
p.comb <- metabCombiner(xdata = p30, ydata = p20, binGap = 0.005)

##example 1 (no known IDs used)
p.comb <- selectAnchors(p.comb, tolmz = 0.003, tolQ = 0.3, windx = 0.03,
    windy = 0.02, tolrtq = 0.3)

##example 2 (known IDs used)
p.comb <- selectAnchors(p.comb, useID = TRUE, tolmz = 0.003, tolQ = 0.3)

##To View Plot of Ordered Pairs
anchors = getAnchors(p.comb)
plot(anchors$rtx, anchors$rty, main = "Selected Anchor Ordered Pairs",
    xlab = "rtx", ylab = "rty")
```

write2file *Print metabCombiner Report to File.*

Description

Prints a combinedTable report to a file, specified by file argument. Output file has an empty line between each separate m/z group for ease of viewing.

Usage

```
write2file(object, file, sep = ",")
```

Arguments

object	metabCombiner object or combinedTable
file	character string naming the output file path
sep	Character field separator. Values within each row are separated by this character.

Value

no values returned

Examples

```
data(plasma30)
data(plasma20)

p30 <- metabData(plasma30, samples = "CHEAR")
p20 <- metabData(plasma20, samples = "Red", rtmax = 17.25)
p.comb <- metabCombiner(xdata = p30, ydata = p20, binGap = 0.0075)

p.comb <- selectAnchors(p.comb, tolmz = 0.003, tolrtq = 0.3, windy = 0.02)
p.comb <- fit_gam(p.comb, k = 20, iterFilter = 1)
p.comb <- calcScores(p.comb, A = 90, B = 14, C = 0.5)


###using metabCombiner object as input
write2file(p.comb, file = "plasma-combined.csv", sep = ",")

###using combinedTable report as input
cTable <- combinedTable(p.comb)
write2file(cTable, file = "plasma-combined.txt", sep = "\t")
```

Index