Package 'apLCMS'

August 19, 2025

Title Adaptive Processing of LC-MS Data

Version 6.8.3

Date 2025-07-30

Description Provides methods for the processing of liquid chromatography-mass spectrometry (LC/MS) based metabolomics data, including adaptive tolerance level searching, non-parametric intensity grouping, the use of run filter to preserve weak signals, model-based estimation of peak intensities, and peak detection based on existing knowledge. Related references include Yu et al. (2009) <doi:10.1093/bioinformatics/btp291>, Liu et al. (2020) <doi:10.1038/s41598-020-70850-0>, Yu et al. (2014) <doi:10.1093/bioinformatics/btu430>, Yu et al. (2013) <doi:10.1021/pr301053d>.

Depends R (>= 2.10), foreach, iterators, ROCR, Rcpp, doParallel

Imports rgl, mzR, e1071, gbm, randomForest, MASS, splines, ROCS

Suggests msdata

biocViews Technology, MassSpectrometry

License GPL (>= 2)

LazyLoad yes

NeedsCompilation yes

LinkingTo Rcpp

Author Tianwei Yu [aut, cre]

Maintainer Tianwei Yu <yutianwei@cuhk.edu.cn>

Repository CRAN

Date/Publication 2025-08-19 14:30:08 UTC

Contents

`apLCMS-package` *Adaptive processing of LC/MS data*

Description

The package generates a feature table from a batch of LC/MS spectra. It finds m/z and retention time tolerance levels from the data. A run-filter is used to detect peaks and remove noise. Non-parametric statistical methods are used to find-tune peak selection and grouping. After retention time correction, a feature table is generated by aligning peaks across spectra.

Author(s)

Tianwei Yu <tyu8@emory.edu> Maintainer: Tianwei Yu <tyu8@emory.edu>

References

Bioinformatics. 25(15):1930-36. BMC Bioinformatics. 11:559. J. Proteome Res. 12(3):1419-27.

---

adaptive.bin *Adaptive binning*

Description

This is an internal function. It creates EICs using adaptive binning procedure

Usage

```
adaptive.bin(x, min.run, min.pres, tol, baseline.correct, weighted=FALSE)
```

Arguments

x	A matrix with columns of m/z, retention time, intensity.
min.pres	Run filter parameter. The minimum proportion of presence in the time period for a series of signals grouped by m/z to be considered a peak.
min.run	Run filter parameter. The minimum length of elution time for a series of signals grouped by m/z to be considered a peak.
tol	m/z tolerance level for the grouping of data points. This value is expressed as the fraction of the m/z value. This value, multiplied by the m/z value, becomes the cutoff level. The recommended value is the machine's nominal accuracy level. Divide the ppm value by 1e6. For FTMS, 1e-5 is recommended.
baseline.correct	After grouping the observations, the highest intensity in each group is found. If the highest is lower than this value, the entire group will be deleted. The default value is NA, in which case the program uses the 75th percentile of the height of the noise groups.
weighted	Whether to weight the local density by signal intensities.

Details

It uses repeated smoothing and splitting to separate EICs. The details are described in the reference and flowchart.

Value

A list is returned.

height.rec	The records of the height of each EIC.
masses	The vector of m/z values after binning.
labels	The vector of retention time after binning.
intensi	The vector of intensity values after binning.
grps	The EIC labels, i.e. which EIC each observed data point belongs to.
times	All the unique retention time values, ordered.
tol	The m/z tolerance level.
min.count.run	The minimum number of elution time points for a series of signals grouped by m/z to be considered a peak.
weighted	Whether to weight the local density by signal intensities.

Author(s)

Tianwei Yu <tyu8@emory.edu>

References

Bioinformatics. 25(15):1930-36. BMC Bioinformatics. 11:559.

---

adaptive.bin.2 *Adaptive binning specifically for the machine learning approach.*

Description

This is an internal function. It creates EICs using adaptive binning procedure

Usage

```
adaptive.bin.2(x, tol, ridge.smoother.window=50, baseline.correct)
```

Arguments

x	A matrix with columns of m/z, retention time, intensity.
tol	m/z tolerance level for the grouping of data points. This value is expressed as the fraction of the m/z value. This value, multiplied by the m/z value, becomes the cutoff level. The recommended value is the machine's nominal accuracy level. Divide the ppm value by 1e6. For FTMS, 1e-5 is recommended.
ridge.smoother.window	The size of the smoother window used by the kernel smoother to remove long ridge noise from the EIC.

`baseline.correct`
: After grouping the observations, the highest intensity in each group is found. If the highest is lower than this value, the entire group will be deleted. The default value is NA, in which case the program uses the 75th percentile of the height of the noise groups.

Details

It uses repeated smoothing and splitting to separate EICs. The details are described in the reference and flowchart.

Value

A list is returned.

`height.rec`	The records of the height of each EIC.
`masses`	The vector of m/z values after binning.
`labels`	The vector of retention time after binning.
`intensi`	The vector of intensity values after binning.
`grps`	The EIC labels, i.e. which EIC each observed data point belongs to.
`times`	All the unique retention time values, ordered.
`tol`	The m/z tolerance level.

Author(s)

Tianwei Yu <tyu8@emory.edu>

References

Bioinformatics. 30(20): 2941-2948. Bioinformatics. 25(15):1930-36. BMC Bioinformatics. 11:559.

`adduct.table` *A table of potential adducts.*

Description

The data is based on the Metabolomics FieHn Lab's Mass Spectrometry Adduct Calculator. It provides the basis for calculating the m/z of the ion forms of known metabolites.

Usage

```
data(adduct.table)
```

Format

A data frame with 47 observations on the following 4 variables.

adduct The ion form.

divider The value to divide the neutral mass by.

addition The value to add after dividing.

charge The charge state of the ion form.

Source

http://fiehnlab.ucdavis.edu/staff/kind/Metabolomics/MS-Adduct-Calculator/

References

Huang N.; Siegel M.M.1; Kruppa G.H.; Laukien F.H. Automation of a Fourier transform ion cyclotron resonance mass spectrometer for acquisition, analysis, and e-mailing of high-resolution exact-mass electrospray ionization mass spectral data. J Am Soc Mass Spectrom 1999, 10, 1166-1173.

Examples

```
data(metabolite.table)
data(adduct.table)
known.table.example<-make.known.table(metabolite.table[1001:1020,], adduct.table[1:4,])
```

---

adjust.time *Adjust retention time across spectra.*

Description

This function adjusts the retention time in each LC/MS profile to achieve better between-profile agreement.

Usage

```
adjust.time(features, mz.tol = NA, chr.tol = NA, colors=NA, find.tol.max.d=1e-4,
max.align.mz.diff=0.01, transform.mz=FALSE, transform.mz.const=0.1)
```

Arguments

features	A list object. Each component is a matrix which is the output from proc.to.feature().
mz.tol	The m/z tolerance level for peak alignment. The default is NA, which allows the program to search for the tolerance level based on the data. This value is expressed as the percentage of the m/z value. This value, multiplied by the m/z value, becomes the cutoff level.
chr.tol	The retention time tolerance level for peak alignment. The default is NA, which allows the program to search for the tolerance level based on the data.

`colors` The vector of colors to be used for the line plots of time adjustments. The default is NA, in which case the program uses a set of default color set.

`find.tol.max.d` Argument passed to find.tol(). Consider only m/z diffs smaller than this value. This is only used when the mz.tol is NA.

`max.align.mz.diff`
As the m/z tolerance is expressed in relative terms (ppm), it may not be suitable when the m/z range is wide. This parameter limits the tolerance in absolute terms. It mostly influences feature matching in higher m/z range.

`transform.mz` Whether to apply a nonlinear transformation to m/z values before alignment.

`transform.mz.const`
A constant used in the m/z transformation function

Details

The function first searches for the m/z tolerance level using a mixture model. After the mz.tol is obtained, the peaks are grouped based on it. The function then searches for the retention time tolerance level. Because the peaks are grouped using m/z, only metabolites that share m/z require this parameter. A rather lenient retention time tolerance level is found using a mixture model.

The profile with the highest number of peaks is selected as the template and every other spetrum is adjusted to it one at a time. At every m/z value, if each of the two spetra has just one peak, and the peaks are within the retention time tolerance range, the pair of retention time values are used in the curve fitting. A kernel smoother is fitted using the difference in retention time against the retention time in the profile to be adjusted.

Value

A list object with the exact same structure as the input object features, i.e. one matrix per profile being processed. The only difference this output object has with the input object is that the retention time column in each of the matrices is changed to new adjusted values.

Author(s)

Tianwei Yu <tyu8@emory.edu>

See Also

feature.align

Examples

```
data(features)
adjusted<-adjust.time(features, colors=c("red","blue","green","cyan"))
```

`cdf.to.ftr` *Convert a number of cdf files in the same directory to a feature table*

Description

This is a wrapper function, which calls four other functions to convert a number of cdf files to a feature table. All cdf files to be processed must be in a single folder.

Usage

```
cdf.to.ftr(folder, output_path, file.pattern=".cdf", n.nodes=4, min.exp=2,
min.pres=0.5, min.run=12, mz.tol=1e-5, baseline.correct.noise.percentile=0.05,
shape.model="bi-Gaussian", BIC.factor=2, baseline.correct=0, peak.estim.method="moment",
min.bw=NA, max.bw=NA, sd.cut=c(0.01,500), sigma.ratio.lim=c(0.01, 100),
component.eliminate=0.01, moment.power=1, subs=NULL, align.mz.tol=NA, align.chr.tol=NA,
max.align.mz.diff=0.01, pre.process=FALSE, recover.mz.range=NA, recover.chr.range=NA,
use.observed.range=TRUE, recover.min.count=3, intensity.weighted=FALSE)
```

Arguments

`folder`	The folder where all CDF files to be processed are located. For example ?C:/CDF/this_experiment?
`output_path`	Path to the output directory
`file.pattern`	The pattern in the names of the files to be processed. The default is ".cdf". Other formats supported by mzR package can also be used, e.g. "mzML" etc.
`n.nodes`	The number of CPU cores to be used through doSNOW.
`min.exp`	If a feature is to be included in the final feature table, it must be present in at least this number of spectra.
`min.pres`	This is a parameter of thr run filter, to be passed to the function proc.cdf(). Please see the help for proc.cdf() for details.
`min.run`	This is a parameter of thr run filter, to be passed to the function proc.cdf(). Please see the help for proc.cdf() for details.
`subs`	If not all the CDF files in the folder are to be processed, the user can define a subset using this parameter. For example, subs=15:30, or subs=c(2,4,6,8)
`mz.tol`	The user can provide the m/z tolerance level for peak identification. This value is expressed as the percentage of the m/z value. This value, multiplied by the m/z value, becomes the cutoff level. Please see the help for proc.cdf() for details.
`baseline.correct.noise.percentile`	The perenctile of signal strength of those EIC that don't pass the run filter, to be used as the baseline threshold of signal strength. This parameter is passed to proc.cdf()
`shape.model`	The mathematical model for the shape of a peak. There are two choices - bi-Gaussian and Gaussian. When the peaks are asymmetric, the bi-Gaussian is better. The default is bi-Gaussian.

`BIC.factor` the factor that is multiplied on the number of parameters to modify the BIC criterion. If larger than 1, models with more peaks are penalized more.

`baseline.correct`
This is a parameter in peak detection. After grouping the observations, the highest observation in each group is found. If the highest is lower than this value, the entire group will be deleted. The default value is NA, which allows the program to search for the cutoff level. Please see the help for proc.cdf() for details.

`peak.estim.method`
the bi-Gaussian peak parameter estimation method, to be passed to subroutine prof.to.features. Two possible values: moment and EM.

`min.bw` The minimum bandwidth in the smoother in prof.to.features(). Please see the help file for prof.to.features() for details.

`max.bw` The maximum bandwidth in the smoother in prof.to.features(). Please see the help file for prof.to.features() for details.

`sd.cut` A parameter for the prof.to.features() function. A vector of two. Features with standard deviation outside the range defined by the two numbers are eliminated.

`sigma.ratio.lim`
A parameter for the prof.to.features() function. A vector of two. It enforces the belief of the range of the ratio between the left-standard deviation and the righ-standard deviation of the bi-Gaussian fuction used to fit the data.

`component.eliminate`
In fitting mixture of bi-Gaussian (or Gaussian) model of an EIC, when a component accounts for a proportion of intensities less than this value, the component will be ignored.

`moment.power` The power parameter for data transformation when fitting the bi-Gaussian or Gaussian mixture model in an EIC.

`align.chr.tol` The user can provide the elution time tolerance level to override the program?s selection. This value is in the same unit as the elution time, normaly seconds. Please see the help for match.time() for details.

`align.mz.tol` The user can provide the m/z tolerance level for peak alignment to override the program?s selection. This value is expressed as the percentage of the m/z value. This value, multiplied by the m/z value, becomes the cutoff level.Please see the help for feature.align() for details.

`max.align.mz.diff`
As the m/z tolerance in alignment is expressed in relative terms (ppm), it may not be suitable when the m/z range is wide. This parameter limits the tolerance in absolute terms. It mostly influences feature matching in higher m/z range.

`pre.process` Logical. If true, the program will not perform time correction and alignment. It will only generate peak tables for each spectra and save the files. It allows manually dividing the task to multiple machines.

`recover.mz.range`
A parameter of the recover.weaker() function. The m/z around the feature m/z to search for observations. The default value is NA, in which case 1.5 times the m/z tolerance in the aligned object will be used.

`recover.chr.range`
: A parameter of the recover.weaker() function. The retention time around the feature retention time to search for observations. The default value is NA, in which case 0.5 times the retention time tolerance in the aligned object will be used.

`use.observed.range`
: A parameter of the recover.weaker() function. If the value is TRUE, the actual range of the observed locations of the feature in all the spectra will be used.

`recover.min.count`
: The minimum time point count for a series of point in the EIC for it to be considered a true feature.

`intensity.weighted`
: Whether to weight the local density by signal intensities in the initial peak detection.

Details

The wrapper function calls five other functions to perform the feature table generation. Every spectrum (cdf file) first goes through proc.cdf() and prof.to.feature() to generate a spectrum-level peak table. The eluction time correction is done by match.time(). Then the peaks are aligned across spectra by feature.align(). For features deteced in a portion of the spectra, weaker signals in other spectra are recovered by recover.weaker(). From version 4, the parameter mz.tol can no longer be NA. This is to allow the program better process data other than FTLCMS. It is recommended that the user use the machine's claimed accuracy. For FTMS, 1e-5 is recommended.

Value

A list is returned.

`features`	A list object, each component of which being the peak table from a single spectrum.
`features2`	A list object, each component of which being the peak table from a single spectrum, after elution time correction.
`aligned.ftrs`	Feature table BEFORE weak signal recovery.
`final.ftrs`	Feature table after weak signal recovery. This is the end product of the function.
`pk.times`	Table of feature elution time BEFORE weak signal recovery.
`final.times`	Table of feature elution time after weak signal recovery.
`mz.tol`	The input mz.tol value by the user.
`align.mz.tol`	The m/z tolerance level in the alignment across spectra, either input from the user or automatically selected when the user input is NA.
`align.chr.tol`	The retention time tolerance level in the alignment across spectra, either input from the user or automatically selected when the user input is NA.

Author(s)

Tianwei Yu <tyu8@sph.emory.edu>

See Also

proc.cdf, prof.to.feature, adjust.time, feature.align, recover.weaker

---

`cont.index` *Continuity index*

Description

This is an internal function. It uses continuity index (or "run filter") to select putative peaks from EIC.

Usage

```
cont.index(newprof, min.pres = 0.6, min.run = 5)
```

Arguments

`newprof`	The matrix containing m/z, retention time, intensity, and EIC label as columns.
`min.pres`	Run filter parameter. The minimum proportion of presence in the time period for a series of signals grouped by m/z to be considered a peak.
`min.run`	Run filter parameter. The minimum length of elution time for a series of signals grouped by m/z to be considered a peak.

Details

This is the run filter described in Yu et al Bioinformatics 2009.

Value

A list is returned.

`new.rec`	The matrix containing m/z, retention time, intensity, and EIC label as columns after applying the run filter.
`height.rec`	The vector of peak heights.
`time.range.rec`	The vector of peak retention time span.
`mz/pres.rec`	The vector of proportion of non-missing m/z.

Author(s)

Tianwei Yu <tyu8@emory.edu>

References

Bioinformatics. 25(15):1930-36. BMC Bioinformatics. 11:559.

`eic.disect` *Internal function: Extract data feature from EIC.*

Description

The function extracts data features after applying different smoother settings.

Usage

```
eic.disect(raw.prof, smoother.window = c(1, 5, 10))
```

Arguments

`raw.prof`	The data after adaptive binning, i.e. the output from adaptive.bin.2().
`smoother.window`	The smoother window sizes to use for data feature extraction.

Details

We take a number of data characteristic measurements from each EIC, including m/z span, m/z standard deviation, retention time (RT) span, RT peak location, and summary statistics on the raw intensity values of the EIC. We also centroid the data in each EIC such that it becomes two-dimensional data (intensity v.s. RT). We then apply different smoothers (shape/window size) in combination of different weighting schemes (unweighted, weighted with intensity, weighted with log intensity) to each EIC. At each smoothing setting, we record summary statistics of smoothed data.

Value

A matrix. Every row corresponds to an EIC. Every column corresponds to a data feature.

Author(s)

Tianwei Yu <tyu8@emory.edu>

References

Bioinformatics. 30(20): 2941-2948.

EIC.plot *Plot extracted ion chromatograms*

Description

Given an output object from the function cdf.to.ftr(), this function plots the EICs selected by the user.

Usage

```
EIC.plot(aligned, rows = NA, colors = NA, transform = "none",
                 subset = NA, min.run, min.pres, max.spline.time.points
                 = 1000)
```

Arguments

aligned	An output object from cdf.to.ftr().
rows	A numeric vector selecting which rows of the aligned feature table to be plotted.
colors	The colors (one per profile) the user wishes to use for the plots. The default is NA, in which case a default color set is used.
transform	There are four possible values. "none": the original intensity data is plotted; "log": the intensity data is transformed by log(x+1); "sqrt": the intensity data is square root transformed; "cuberoot": the intensity data is cube root transformed.
subset	The user can choose a subset of the profiles for which the EICs are plotted. It is given as a vector of profile indecies. The default is NA, in which case the EICs from all the profiles are plotted.
min.run	The min.run parameter used in the proc.cdf() step.
min.pres	The min.pres parameter used in the proc.cdf() step.
max.spline.time.points	The maximum time points to use in spline fit.

Details

The EICs are plotted as overlaid line plots. The graphic device is divided into four parts, each of which is used to plot one EIC. When all four parts are occupied, the function calls x11() to open another graphic device. The colors used (one per profile) is printed in the command window.

Value

There is no return value.

Author(s)

Tianwei Yu <tyu8@emory.edu>

References

Bioinformatics. 25(15):1930-36. BMC Bioinformatics. 11:559.

`EIC.plot.learn` *Plot extracted ion chromatograms based on the machine learning method output*

Description

Given an output object from the function semi.sup.learn(), this function plots the EICs selected by the user.

Usage

```
EIC.plot.learn(aligned, rows = NA, colors = NA, transform = "none",
                 subset = NA, tol = 2.5e-05, ridge.smoother.window =
                 50, baseline.correct = 0, max.spline.time.points =
                 1000)
```

Arguments

`aligned`	An output object from cdf.to.ftr().
`rows`	A numeric vector selecting which rows of the aligned feature table to be plotted.
`colors`	The colors (one per profile) the user wishes to use for the plots. The default is NA, in which case a default color set is used.
`transform`	There are four possible values. "none": the original intensity data is plotted; "log": the intensity data is transformed by log(x+1); "sqrt": the intensity data is square root transformed; "cuberoot": the intensity data is cube root transformed.
`subset`	The user can choose a subset of the profiles for which the EICs are plotted. It is given as a vector of profile indecies. The default is NA, in which case the EICs from all the profiles are plotted.
`tol`	The mz tolerance level used in learn.cdf().
`ridge.smoother.window`	The ridge.smoother.window parameter value used in learn.cdf().
`baseline.correct`	The baseline.correct parameter value used in learn.cdf().
`max.spline.time.points`	The maximum number of points to use in the spline fit along the retention time axis.

Details

The function plots a single EIC. It plots intensity against retention time. It uses different color for different profiles.

Value

There is no return value.

Author(s)

Tianwei Yu <tyu8@emory.edu>

References

Bioinformatics. 25(15):1930-36. BMC Bioinformatics. 11:559.

`eic.pred` *Internal function: calculate the score for each EIC based on prediction of match status.*

Description

This function uses predictive models to evaluate the data features, and give scores to every EIC, which serves as the basis for EIC selection.

Usage

```
eic.pred(eic.rec, known.mz, mass.matched = NA, to.use = 10, do.plot = FALSE,
match.tol.ppm = 5, do.grp.reduce = TRUE, remove.bottom = 5, max.fpr = 0.3, min.tpr = 0.8)
```

Arguments

`eic.rec`	The matrix of data features from every EIC. Each row is an EIC. Each column is a data feature value.
`known.mz`	The m/z values of the known metabolic features.
`mass.matched`	An indicator vector. "1" means the corresponding EIC has an m/z matched to known features. The default is NA, in which case the matching is done inside this function.
`to.use`	The maximum number of data features to use in the predictive models.
`do.plot`	Whether diagnostic plots would be generated.
`match.tol.ppm`	The tolerance level in the m/z match, at ppm scale.
`do.grp.reduce`	Whether to reduce the data features first by reducing each group of similar features into one.
`remove.bottom`	The number of worst performing data features to remove before model building. If true, the removal is done based on single predictor ROC analysis.
`max.fpr`	The threshold for selecting unmatched EICs. Each EIC is assigned an FPR value based on the final prediction model. Those with FPR smaller than this threshold will be selected. If a vector is provided, the first one will be used. But all FPR values will also be returned. So other functions will be able to make selections based on other threshold values.

`min.tpr` The threshold for selecting matched EICs. Each EIC is assigned an TPR value based on the final prediction model. Those with TPR larger than this threshold will be selected. If a vector is provided, the first one will be used. But all TPR values will also be returned. So other functions will be able to make selections based on other threshold values.

Details

The function first subsample the EICs to balance the unmatched/matched. Then it randomly split the data into training and testing set. Combinations of feature ranking and predictive models are used, and their performance guaged using the testing set. The overall best model is selected, and the EICs each receive a score based on this model.

Although there is a single scoring system for all EICs, those matched are treated differently than unmatched, because we have higher confidence in them being real metabolites. The matched are selected using the "min.tpr" threshold, to ensure the majority of them enter next step. Those unmatched are selected using the "max.fpr" threshold.

Value

A list item is returned.

`chosen` An indicator vector. "1" means the EIC is selected; "0" means unselected. When multiple min.tpr and/or max.fpr are provided, this vector corresponds to the combination of the first min.tpr and max.fpr.

`fpr` The vector of FPR values, each value corresponds to the FPR at the cutoff of the specific EIC.

`tpr` The vector of TPR values, each value corresponds to the TPR at the cutoff of the specific EIC.

`matched` An indicator vector. "1" means matched to known features. "0" means unmatched.

`pred.performance` Prediction performance of all models tested.

`feature.rank.method` Which method is used for ranking features.

`model` Which prediction model is used.

`feature importance` The importance score of all data features generated by the feature ranking method.

`used.features` The names of the features used in the final model.

`final.auc` The AUC of the selected model.

Author(s)

Tianwei Yu <tianwei.yu@emory.edu>

References

Bioinformatics. 30(20): 2941-2948.

See Also

semi.sup.learn, eic.qual, eic.disect

`eic.qual` *Internal function: Calculate the single predictor quality.*

Description

For each column of an EIC data feature matrix, find its predictive power on the m/z match to known metabolites.

Usage

```
eic.qual(eic.rec, known.mz, mass.matched = NA, match.tol.ppm = 5, do.plot = FALSE,
pos.confidence = 0.99, neg.confidence = 0.99)
```

Arguments

`eic.rec`	The EIC data feature matrix. Each row is an EIC. Each column is a data feature.
`known.mz`	The known m/z values to be matched to.
`mass.matched`	A vector of indicators of whether the m/z of each EIC is matched to the known m/z values. The default is NA, in which case it is calculated within the function.
`match.tol.ppm`	The tolerance level of m/z match.
`do.plot`	Whether to produce plots of the ROCS.
`pos.confidence`	The confidence level for the features matched to the known feature list.
`neg.confidence`	The confidence level for the features not matching to the known feature list.

Value

A matrix of four columns. The first two columns are the VUS and AUC without uncertainty. The next two columns are the VUS and AUC with uncertainty.

Author(s)

Tianwei Yu <tyu8@emory.edu>

References

Bioinformatics. 30(20): 2941-2948.

`feature.align` *Align peaks from spectra into a feature table.*

Description

Identifies which of the peaks from the profiles correspond to the same feature.

Usage

```
feature.align(features, min.exp = 2, mz.tol = NA, chr.tol = NA,find.tol.max.d=1e-4,
max.align.mz.diff=0.01)
```

Arguments

`features`	A list object. Each component is a matrix which is the output from proc.to.feature().
`min.exp`	A feature has to show up in at least this number of profiles to be included in the final result.
`mz.tol`	The m/z tolerance level for peak alignment. The default is NA, which allows the program to search for the tolerance level based on the data. This value is expressed as the percentage of the m/z value. This value, multiplied by the m/z value, becomes the cutoff level.
`chr.tol`	The retention time tolerance level for peak alignment. The default is NA, which allows the program to search for the tolerance level based on the data.
`find.tol.max.d`	Argument passed to find.tol(). Consider only m/z diffs smaller than this value.This is only used when the mz.tol is NA.
`max.align.mz.diff`	As the m/z tolerance is expressed in relative terms (ppm), it may not be suitable when the m/z range is wide. This parameter limits the tolerance in absolute terms. It mostly influences feature matching in higher m/z range.

Details

The function first searches for the m/z tolerance level using a mixture model. After the mz.tol is obtained, the peaks are grouped based on it. Consecutive peaks with m/z value difference smaller than the tolerance level are considered to belong to the same peak group. Non-parametric density estimation within each peak group is used to further split peak groups. The function then searches for the retention time tolerance level. Because the peaks are grouped using m/z, only metabolites that share m/z require this parameter. A rather lenient retention time tolerance level is found using a mixture model. After splitting the peak groups by this value, non-parametric density estimation is used to further split peak groups. Peaks belonging to one group are considered to correspond to the same feature.

Value

Returns a list object with the following objects in it:

`aligned.ftrs`	A matrix, with columns of m/z values, elution times, signal strengths in each spectrum.
`pk.times`	A matrix, with columns of m/z, median elution time, and elution times in each spectrum.
`mz.tol`	The m/z tolerance used in the alignment.
`chr.tol`	The elution time tolerance in the alignment.

Author(s)

Tianwei Yu <tyu8@emory.edu>

See Also

proc.to.feature

Examples

```
data(features)
features.2<-adjust.time(features)
this.aligned<-feature.align(features,min.exp=2)
summary(this.aligned)
this.aligned$aligned.ftrs[1:5,]
this.aligned$pk.times[1:5,]
```

---

features *Sample feature tables from 4 profiles*

Description

A list object containing 4 matrices, each of which is the feature table from a profile.

Usage

```
data(features)
```

Format

List object containing multiple matrices. One matrix from each spectrum.

Source

Data from Dean Jones lab, Emory University School of Medicine.

Examples

```
data(features)
```

`find.match` *Internal function: finding the best match between a set of detected features and a set of known features.*

Description

Given a small matrix of distances, find the best column-row pairing that minimize the sum of distances of the matched pairs.

Usage

```
find.match(a, unacceptable = 4)
```

Arguments

`a`	A matrix of distances.
`unacceptable`	A distance larger than which cannot be accepted as pairs.

Value

A matrix the same dimension as the input matrix, with matched position taking value 1, and all other positions taking value 0.

Author(s)

Tianwei Yu <tyu8@emory.edu>

`find.tol` *An internal function that is not supposed to be directly accessed by the user. Find m/z tolerance level.*

Description

The function finds the tolerance level in m/z from a given vector of observed m/z values.

Usage

```
find.tol(a, uppermost=1e-4, aver.bin.size=4000, min.bins=50, max.bins=200)
```

Arguments

a	The vector of observed m/z values.
uppermost	Consider only m/z diffs smaller than this value.
aver.bin.size	The average bin size to determine the number of equally spaced points in the kernel density estimation.
min.bins	the minimum number of bins to use in the kernel density estimation. It overrides aver.bin.size when too few observations are present.
max.bins	the maximum number of bins to use in the kernel density estimation. It overrides aver.bin.size when too many observations are present.

Details

The method assumes a mixture model: an unknown distribution of m/z variations in the same peak, and an exponential distribution of between-peak diffs. The parameter of the exponential distribution is estimated by the upper 75

Value

The tolerance level is returned.

Author(s)

Tianwei Yu <tyu8@emory.edu>

Examples

```
data(prof)
find.tol(prof[[1]][,1])
```

---

find.tol.time *An internal function that is not supposed to be directly accessed by the user. Find elution time tolerance level.*

Description

This function finds the time tolerance level. Also, it returns the grouping information given the time tolerance.

Usage

```
find.tol.time(mz, chr, lab, num.exp, mz.tol = 2e-05, chr.tol = NA,
                aver.bin.size = 200, min.bins = 50, max.bins = 100,
                max.mz.diff = 0.01, max.num.segments = 10000)
```

Arguments

mz	mz value of all peaks in all profiles in the study.
chr	retention time of all peaks in all profiles in the study.
lab	label of all peaks in all profiles in the study.
num.exp	The number of spectra in this analysis.
mz.tol	m/z tolerance level for the grouping of signals into peaks. This value is expressed as the percentage of the m/z value. This value, multiplied by the m/z value, becomes the cutoff level.
chr.tol	the elution time tolerance. If NA, the function finds the tolerance level first. If a numerical value is given, the function directly goes to the second step - grouping peaks based on the tolerance.
aver.bin.size	The average bin size to determine the number of equally spaced points in the kernel density estimation.
min.bins	the minimum number of bins to use in the kernel density estimation. It overrides aver.bin.size when too few observations are present.
max.mz.diff	As the m/z tolerance in alignment is expressed in relative terms (ppm), it may not be suitable when the m/z range is wide. This parameter limits the tolerance in absolute terms. It mostly influences feature matching in higher m/z range.
max.bins	the maximum number of bins to use in the kernel density estimation. It overrides aver.bin.size when too many observations are present.
max.num.segments	the maximum number of segments.

Details

The peaks are first ordered by m/z, and split into groups by the m/z tolerance. Then within every peak group, the pairwise elution time difference is calculated. All the pairwise elution time differences within groups are merged into a single vector. A mixture model (unknown distribution for distance between peaks from the same feature, and a triangle-shaped distribution for distance between peaks from different features) is fit to find the elution time tolerance level. The elution times within each peak group are then ordered. If a gap between consecutive retention times is larger than the elution time tolerance level, the group is further split at the gap. Grouping information is returned, as well as the elution time tolerance level.

Value

A list object is returned:

chr.tol	The elution time tolerance level.
comp2	A matrix with six columns. Every row corrsponds to a peak in one of the spectrum. The columns are: m/z, elution time, spread, signal strength, spectrum label, and peak group label. The rows are ordered by the median m/z of each peak group, and with each peak group the rows are ordered by the elution time.

Author(s)

Tianwei Yu <tyu8@emory.edu>

`find.turn.point` *Find peaks and valleys of a curve.*

Description

This is an internal function which is not supposed to be directly accessed by the user. Finds the peaks and valleys of a smooth curve.

Usage

```
find.turn.point(y)
```

Arguments

`y`	The y values of a curve in x-y plane.

Value

A list object:

`pks`	The peak positions.
`vlys`	The valley positions

Author(s)

Tianwei Yu <tyu8@emory.edu>

References

Bioinformatics. 25(15):1930-36. BMC Bioinformatics. 11:559.

`interpol.area` *Interpolate missing intensities and calculate the area for a single EIC.*

Description

This is an internal function that's not supposed to be called directly by the user.

Usage

```
interpol.area(x, y, all.x, all.w)
```

Arguments

x	the positions of x(retention time) where non-NA y is observed.
y	the observed intensities.
all.x	all possible x(retention time) in the LCMS profile.
all.w	the "footprint" of each measured retention time, used as weight for the corresponding y.

Details

This is an internal function. It interpolates missing y using linear interpolation, and then calculates the area under the curve.

Value

The area is returned.

Author(s)

Tianwei Yu <tyu8@emory.edu>

---

learn.cdf *Peak detection using the machine learning approach.*

Description

The procedure uses information of known metabolites, and constructs prediction models to differentiate EICs.

Usage

```
learn.cdf(filename, output_path, tol = 2e-05, min.run = 4, min.pres = 0.3,
baseline.correct = 0, ridge.smoother.window = 50, smoother.window = c(1, 5, 10),
known.mz, match.tol.ppm = 5, do.plot = FALSE, pos.confidence = 0.99,
neg.confidence = 0.99, max.ftrs.to.use = 10, do.grp.reduce = TRUE,
remove.bottom.ftrs = 0, max.fpr = seq(0, 0.6, by = 0.1), min.tpr = seq(0.8, 1, by = 0.1),
intensity.weighted=FALSE)
```

Arguments

filename	The cdf file name. If the file is not in the working directory, the path needs to be given.
output_path	Path to the output directory
min.pres	Run filter parameter. The minimum proportion of presence in the time period for a series of signals grouped by m/z to be considered a peak.
min.run	Run filter parameter. The minimum length of elution time for a series of signals grouped by m/z to be considered a peak.

`tol` m/z tolerance level for the grouping of data points. This value is expressed as the fraction of the m/z value. This value, multiplied by the m/z value, becomes the cutoff level. The recommended value is the machine's nominal accuracy level. Divide the ppm value by 1e6. For FTMS, 1e-5 is recommended.

`baseline.correct` After grouping the observations, the highest intensity in each group is found. If the highest is lower than this value, the entire group will be deleted. The default value is NA, in which case the program uses the 75th percentile of the height of the noise groups.

`ridge.smoother.window` The size of the smoother window used by the kernel smoother to remove long ridge noise from each EIC.

`smoother.window` The smoother windows to use in data feature generation.

`known.mz` The m/z values of the known metabolites.

`match.tol.ppm` The ppm tolerance to match identified features to known metabolites/features.

`do.plot` Whether to produce diagnostic plots.

`pos.confidence` The confidence level for the features matched to the known feature list.

`neg.confidence` The confidence level for the features not matching to the known feature list.

`max.ftrs.to.use` The maximum number of data features to use in a predictive model.

`do.grp.reduce` Whether to reduce data features that are similar. It is based on data feature predictability.

`remove.bottom.ftrs` The number of worst performing data features to remove before model building.

`max.fpr` The proportion of unmatched features to be selected in the feature detection step.

`min.tpr` The proportion of matched features to be selected in the feature detection step.

`intensity.weighted` Whether to weight the local density by signal intensities.

Details

The subroutine takes CDF, mxml etc LC/MS profile. First the profile is sliced into EICs using adaptive binning. Then data features are extracted from each EIC. The EICs are classified into two groups: those that have m/z values that match to known m/z values, and those that don't. Classification models are built to separate the two classes, and each EIC is given a score by the classification model. Those with better scores are selected to enter the feature quantification step.

Value

A matrix with four columns: m/z value, retention time, intensity, and group number.

Author(s)

Tianwei Yu <tyu8@emory.edu>

`load.lcms` *Loading LC/MS data.*

Description

This is an internal function. It loads LC/MS data into memory.

Usage

```
load.lcms(filename)
```

Arguments

`filename`	The CDF file name.

Details

The function uses functionality provided by the mzR package from Bioconductor.

Value

A list is returned.

`masses`	The vector of m/z values.
`labels`	The vector of retention times.
`intensi`	The vector of intensity values.
`times`	The vector of unique time points.

Author(s)

Tianwei Yu <tyu8@emory.edu>

References

Bioinformatics. 25(15):1930-36. BMC Bioinformatics. 11:559.

`make.known.table` *Producing a table of known features based on a table of metabolites and a table of allowable adducts.*

Description

Given a table of known metabolites with original mass and charge information, and a table of allowable adducts, this function outputs a new table of potential features.

Usage

```
make.known.table(metabolite.table, adduct.table, ion.mode = "+")
```

Arguments

`metabolite.table`	A table of known metabolites. See the description of the object "metabolite.table" for details.
`adduct.table`	A table of allowable adducts. See the description of the object "adduct.table" for details.
`ion.mode`	Character. Either "+" or "-".

Details

For each allowable ion form, the function produces the m/z of every metabolite given to it. The output table follows the format that is required by the function semi.sup(), so that the user can directly use the table for semi supervised feature detection.

Value

A data frame containing the known metabolite ions. It contains 18 columns: "chemical_formula": the chemical formula if knonw; "HMDB_ID": HMDB ID if known; "KEGG_compound_ID": KEGG compound ID if known; "neutral.mass": the neutral mass if known: "ion.type": the ion form, such as H+, Na+, ..., if known; "m.z": m/z value, either theoretical for known metabolites, or mean observed value for unknown but previously found features; "Number_profiles_processed": the total number of LC/MS profiles that were used to build this database; "Percent_found": in what percentage was this feature found historically amount all data processed in building this database; "mz_min": the minimum m/z value observed for this feature; "mz_max": the maximum m/z value observed for this feature; "RT_mean": the mean retention time observed for this feature; "RT_sd": the standard deviation of retention time observed for this feature; "RT_min": the minimum retention time observed for this feature; "RT_max": the maximum retention time observed for this feature; "int_mean.log.": the mean log intensity observed for this feature; "int_sd.log.": the standard deviation of log intensity observed for this feature; "int_min.log.": the minimum log intensity observed for this feature; "int_max.log.": the maximum log intensity observed for this feature;

Author(s)

Tianwei Yu <tyu8@emory.edu>

References

Yu T, Park Y, Li S, Jones DP (2013) Hybrid feature detection and information accumulation using high-resolution LC-MS metabolomics data. J. Proteome Res. 12(3):1419-27.

See Also

metabolite.table, adduct.table, semi.sup

Examples

```
data(metabolite.table)
data(adduct.table)
known.table.example<-make.known.table(metabolite.table[1001:1020,], adduct.table[1:4,])
```

`mass.match` *An internal function: finding matches between two vectors of m/z values.*

Description

Given two vectors of m/z values and the tolerance ppm level, find the potential matches between the two vectors.

Usage

```
mass.match(x, known.mz, match.tol.ppm = 5)
```

Arguments

`x`	m/z values from the data.
`known.mz`	m/z values from the known feature table.
`match.tol.ppm`	tolerance level in ppm.

Value

A vector the same length as x. 1 indicates matched, and 0 indicates unmatched.

Author(s)

Tianwei Yu <tyu8@emory.edu>

merge_seq_3 *An internal function.*

Description

This is a internal function. It shouldn't be called by the end user.

Usage

```
merge_seq_3(a, mz, inte)
```

Arguments

a	vector of retention time.
mz	vector of m/z ratio.
inte	vector of signal strength.

Author(s)

Tianwei Yu <tyu8@emory.edu>

References

Bioinformatics. 25(15):1930-36. BMC Bioinformatics. 11:559.

metabolite.table *A known metabolite table based on HMDB.*

Description

This table was compiled from HMDB metabolites. It contains only the basic information of known metabolites. It can be used to produce feature tables with ion forms of the users' choice.

Usage

```
data(metabolite.table)
```

Format

A data frame containing the known metabolites. It contains 4 columns: "chemical_formula": the chemical formula of the known table; "HMDB_ID": HMDB ID; "KEGG_compound_ID": KEGG compound ID if known; "mass": the neutral mass;

Details

It is to be used in combination with the object "adduct.table", to create feature table with ion forms of the user's choice. Which ion form to choose should be based on the LC/MS system.

Source

Wishart, D. S., et al. (2009). HMDB: a knowledgebase for the human metabolome. Nucleic Acids Res 37, D603-10.

Examples

```
data(metabolite.table)
data(adduct.table)
known.table.example<-make.known.table(metabolite.table[1001:1020,], adduct.table[1:4,])
```

---

peak.characterize *Internal function: Updates the information of a feature for the known feature table.*

Description

The function takes the information about the feature in the known feature table (if available), and updates it using the information found in the current dataset.

Usage

```
peak.characterize(existing.row = NA, ftrs.row, chr.row)
```

Arguments

existing.row	The existing row in the known feature table (detailed in the semi.sup() document).
ftrs.row	The row of the matched feature in the new aligned feature table.
chr.row	The row of the matched feature in the new retention time table of aligned features.

Details

The function calculates and updates the mean intensity, variation of intensity, mean retention time etc.

Value

A vector, the updated row for the known feature table.

Author(s)

Tianwei Yu <tyu8@emory.edu>

plot_cdf_2d *Plot the data in the m/z and retention time plane.*

Description

This is a diagnostic function. It takes the original CDF file, as well as the detected feature table, and plots the data in the m/z - retention time plane, using a user-defined range. The entire data is too big to plot, thus the main purpose is to focus on small subregions of the data and check the peak detection results.

Usage

```
plot_cdf_2d(rawname, f, mzlim, timelim, lwd = 1)
```

Arguments

rawname	The CDF file name.
f	The output object of prof.to.feature().
mzlim	The m/z range to plot.
timelim	The retention time range to plot.
lwd	Line width parameter, to be passed on to the function line().

Value

There is no return value.

Author(s)

Tianwei Yu <tyu8@emory.edu>

References

Bioinformatics. 25(15):1930-36. BMC Bioinformatics. 11:559.

plot_txt_2d *Plot the data in the m/z and retention time plane.*

Description

This is a diagnostic function. It takes the original text file, as well as the detected feature table, and plots the data in the m/z - retention time plane, using a user-defined range. The entire data is too big to plot, thus the main purpose is to focus on small subregions of the data and check the peak detection results.

Usage

```
plot_txt_2d(rawname, f, mzlim, timelim, lwd = 1)
```

Arguments

rawname	The text file name.
f	The output object of prof.to.feature().
mzlim	The m/z range to plot.
timelim	The retention time range to plot.
lwd	Line width parameter, to be passed on to the function line().

Details

The columns in the text file need to be separated by tab. The first column needs to be the retention time, the second column the m/z values, and the third column the intensity values. The first row needs to be the column labels, rather than values.

Value

There is no return value.

Author(s)

Tianwei Yu <tyu8@emory.edu>

References

Bioinformatics. 25(15):1930-36. BMC Bioinformatics. 11:559.

---

present.cdf.3d *Generates 3 dimensional plots for LCMS data.*

Description

This function takes the matrix output from proc.cdf() and generates a 3D plot of the data. It relies on the rgl library.

Usage

```
present.cdf.3d(prof, fill.holes = TRUE, transform = "none", time.lim = NA,
mz.lim = NA, box = TRUE, axes = TRUE)
```

Arguments

prof	The matrix output from the proc.cdf() function.
fill.holes	A lot of peaks have missing values at some time points. If fill.holes is TRUE, the function will fill in the missing values by interpolation.
transform	If the value is "sqrt", the values are transformed by taking square root. If "cube-root", the values are transformed by taking cubic root.
time.lim	The limit in retention time for the area of spectrum to be plotted. It should be either NA or a vector of two values: the lower limit and the upper limit.
mz.lim	The limit in m/z value for the area of spectrum to be plotted. It should be either NA or a vector of two values: the lower limit and the upper limit.
box	If a box should be drawn.
axes	If the axes should be drawn.

Details

The function calls the rgl library. Spectrum values within the time.lim and mz.lim range is plotted in 3D.

Value

There is no return value from this function.

Author(s)

Tianwei Yu <tyu8@emory.edu>

References

http://rgl.neoscientists.org/about.shtml

Examples

```
data(prof)
present.cdf.3d(prof[[2]],time.lim=c(250,400), mz.lim=c(400,500))
```

---

proc.cdf *Filter noise and detect peaks from LC/MS data in CDF format*

Description

This function applies the run filter to remove noise. Data points are grouped into EICs in this step.

Usage

```
proc.cdf(filename, output_path, min.pres=0.5, min.run=12, tol=1e-5, baseline.correct=0,
baseline.correct.noise.percentile=0, do.plot=TRUE, intensity.weighted=FALSE)
```

Arguments

filename	The cdf file name. If the file is not in the working directory, the path needs to be given.
output_path	Path to the output directory
min.pres	Run filter parameter. The minimum proportion of presence in the time period for a series of signals grouped by m/z to be considered a peak.
min.run	Run filter parameter. The minimum length of elution time for a series of signals grouped by m/z to be considered a peak.
tol	m/z tolerance level for the grouping of data points. This value is expressed as the fraction of the m/z value. This value, multiplied by the m/z value, becomes the cutoff level. The recommended value is the machine's nominal accuracy level. Divide the ppm value by 1e6. For FTMS, 1e-5 is recommended.
baseline.correct	After grouping the observations, the highest intensity in each group is found. If the highest is lower than this value, the entire group will be deleted. The default value is NA, in which case the program uses a percentile of the height of the noise groups. If given a value, the value will be used as the threshold, and baseline.correct.noise.percentile will be ignored.
baseline.correct.noise.percentile	The perenctile of signal strength of those EIC that don't pass the run filter, to be used as the baseline threshold of signal strength.
do.plot	Whether to generate diagnostic plots.
intensity.weighted	Whether to weight the local density by signal intensities.

Details

The subroutine takes CDF, mxml etc LC/MS profile. The m/z are grouped based on the tolerance level using multi-stage smoothing and peak finding. Non-parametric density estimation is used in both m/z dimension and elution time dimension to fine-tune the signal grouping. A run filter is applied, which requires a "true peak" to have a minimum length in the retention time dimension (parameter: min.run), as well as being detected at or higher than a proportion of the time points within the time period (parameter: min.pres).

Value

A matrix with four columns: m/z value, retention time, intensity, and group number.

Author(s)

Tianwei Yu <tyu8@emory.edu>

proc.cdf.2d *Compute a 2D Binned Kernel Density Estimate from LC/MS data in CDF format.*

Description

This function provided a method to compute the density estimate of a LC/MS data matrix based on each point's density. It will return a set of peak's centre information including the point's coordinate in each coordinate axis and all the distances between the peak point and grid boundaries.

Usage

```
proc.cdf.2d(filename, mz.cut = 5e-4, rt.cut = 50, mz.search.range = 2e-3,
rt.search.range = 200, mz.search.step = 5e-4, rt.search.step = 50,
intensity.limit.quantile = 0.1, bPlot = FALSE, transform.mz=FALSE, transform.mz.const=0.1)
```

Arguments

filename	The cdf file name. If the file is not in the working directory, the path needs to be given.
mz.cut	The divided gird width in m/z when calculate the density of each point.
rt.cut	The divided gird width in RT when calculate the density of each point.
mz.search.range	maximum peak width in m/z
rt.search.range	(maximum peak width in RT
mz.search.step	maximum search step in m/z
rt.search.step	maximum search step in RT
intensity.limit.quantile	intensity threshold
bPlot	Whether to plot
transform.mz	Whether to apply a nonlinear transformation to m/z values before alignment.
transform.mz.const	A constant used in the m/z transformation function

Value

finalMatrix	A matrix contains the information of peaks. Each row contains one peak's information and each colunm represent one aspect of the peak's information. Column 1 's value represent the X Position of each peak's centre. Column 2 's value represent the Y Position of each peak's centre. Column 3 's value represent the distance between centre of the peak and the top boudary of divided grid. Column 4 's value represent the distance between centre of the peak and the bottom boudary of divided grid. Column 5 's vlaue represent the peak's value . Column 6 's value represent the distance between centre of the peak and the left boudary of divided grid. Column 7 's value represent the distance between centre of the peak and the right boudary of divided grid.

Examples

```
library(msdata)
filepath <- system.file("microtofq", package = "msdata")
file <- list.files(filepath, pattern="MM14.mzML",
                    full.names=TRUE, recursive = TRUE)
peakInfo <- proc.cdf.2d(file)
```

---

proc.txt *Filter noise and detect peaks from LC/MS data in text format*

Description

This function applies the run filter to remove noise. Data points are grouped into EICs in this step.

Usage

```
proc.txt(filename, output_path, min.pres=0.5, min.run=12,tol=NA, find.tol.maxd=1e-4,
baseline.correct.noise.percentile=0.25, baseline.correct=0)
```

Arguments

filename	The text file name. If the file is not in the working directory, the path needs to be given.
output_path	Path to the output directory
min.pres	Run filter parameter. The minimum proportion of presence in the time period for a series of signals grouped by m/z to be considered a peak.
min.run	Run filter parameter. The minimum length of elution time for a series of signals grouped by m/z to be considered a peak.
tol	m/z tolerance level for the grouping of data points. This value is expressed as the fraction of the m/z value. This value, multiplied by the m/z value, becomes the cutoff level. The recommended value is the machine's nominal accuracy level. Divide the ppm value by 1e6. For FTMS, 1e-5 is recommended.
find.tol.maxd	maximum distance between datapoints that are allowed in the procedure to find tolerance.
baseline.correct	After grouping the observations, the highest intensity in each group is found. If the highest is lower than this value, the entire group will be deleted. The default value is NA, in which case the program uses the 75th percentile of the height of the noise groups.
baseline.correct.noise.percentile	The perenctile of signal strength of those EIC that don't pass the run filter, to be used as the baseline threshold of signal strength.

Details

The columns in the text file need to be separated by tab. The first column needs to be the retention time, the second column the m/z values, and the third column the intensity values. The first row needs to be the column labels, rather than values. The m/z are grouped based on the tolerance level using multi-stage smoothing and peak finding. Non-parametric density estimation is used in both m/z dimension and elution time dimension to fine-tune the signal grouping. A run filter is applied, which requires a "true peak" to have a minimum length in the retention time dimension (parameter: min.run), as well as being detected at or higher than a proportion of the time points within the time period (parameter: min.pres).

Value

A matrix with four columns: m/z value, retention time, intensity, and group number.

Author(s)

Tianwei Yu <tyu8@emory.edu>

---

prof *Sample profile data after noise filtration by the run filter*

Description

A list object containing 4 matrices. Each matrix is from an LC/MS profile.

Usage

```
data(prof)
```

Format

Each matrix contains 4 columns: m/z, retention time, intensity, and group number.

Source

Data from Dean Jones lab, Emory University School of Medicine.

Examples

```
data(prof)
present.cdf.3d(prof[[2]],time.lim=c(250,400), mz.lim=c(400,500))
this.feature<-prof.to.features(prof[[1]])
```

`prof.to.features` *Generate feature table from noise-removed LC/MS profile*

Description

Each LC/MS profile is first processed by the function proc.cdf() to remove noise and reduce data size. A matrix containing m/z value, retention time, intensity, and group number is output from proc.cdf(). This matrix is then fed to the function prof.to.features() to generate a feature list. Every detected feature is summarized into a single row in the output matrix from this function.

Usage

```
prof.to.features(a, bandwidth=0.5, min.bw=NA, max.bw=NA, sd.cut=c(0.1, 100),
sigma.ratio.lim=c(0.1, 10), shape.model="bi-Gaussian", estim.method="moment",
do.plot=TRUE, power=1, component.eliminate=0.01, BIC.factor=2)
```

Arguments

`a`	The matrix output from proc.cdf(). It contains columns of m/z value, retention time, intensity and group number.
`bandwidth`	A value between zero and one. Multiplying this value to the length of the signal along the time axis helps determine the bandwidth in the kernel smoother used for peak identification. See the details section.
`min.bw`	The minimum bandwidth to use in the kernel smoother. See the details section.
`max.bw`	The maximum bandwidth to use in the kernel smoother. See the details section.
`sd.cut`	A vector of two. Features with standard deviation outside the range defined by the two numbers are eliminated.
`sigma.ratio.lim`	A vector of two. It enforces the belief of the range of the ratio between the left-standard deviation and the righ-standard deviation of the bi-Gaussian fuction used to fit the data.
`shape.model`	The mathematical model for the shape of a peak. There are two choices - "bi-Gaussian" and "Gaussian". When the peaks are asymmetric, the bi-Gaussian is better. The default is "bi-Gaussian".
`estim.method`	The estimation method for the bi-Gaussian peak model. Two possible values: moment and EM.
`do.plot`	Whether to generate diagnostic plots.
`component.eliminate`	In fitting mixture of bi-Gaussian (or Gaussian) model of an EIC, when a component accounts for a proportion of intensities less than this value, the component will be ignored.
`power`	The power parameter for data transformation when fitting the bi-Gaussian or Gaussian mixture model in an EIC.
`BIC.factor`	the factor that is multiplied on the number of parameters to modify the BIC criterion. If larger than 1, models with more peaks are penalized more.

Details

This function generates the feature table from the noise-removed profile. The m/z values are already grouped by the function proc.cdf() to generate EICs. The task of this function is to look at every EIC and determine (1) how many peaks there are, and (2) the location, spread and area of each peak. For the first task, when a series of signals is found at an m/z group, kernel smoother is fit along the time axis to determine whether there is one single peak or multiple peaks. The bandwidth of the kernel smoother is determined as follows: multiply the length of the signals by the bandwidth parameter. If the resulting value is between the parameters min.bw and max.bw, use that value; else if the value is below min.bw, use min.bw; else if the value is above max.bw, use max.bw. The default values of min.bw and max.bw are NA, in which case min.bw is set to be 1/30 of the retention time range, and max.bw is set to be 1/15 of the retention time range. A modified EM algorithm which allows missing completely at random at certain time points is used for the evaluation of the peak location and area. If a single peak is detected by the kernel smoother, the maximum likelihood normal curve is fitted. If multiple peaks are detected, the EM algorithm finds the normal mixture that best explain the data.

Value

A matrix is returned. The columns are: m/z value, retention time, spread (standard deviation of the estimated normal curve), and estimated total signal strength (total area of the estimated normal curve).

Author(s)

Tianwei Yu <tyu8@sph.emory.edu>

See Also

proc.cdf

Examples

```
data(prof)
this.feature<-prof.to.features(prof[[1]])
this.feature[1:5,]
```

---

recover.weaker *Recover weak signals in some profiles that is not identified as a peak, but corresponds to identified peaks in other spectra.*

Description

Given the aligned feature table, some features are identified in a subgroup of spectra. This doesn't mean they don't exist in the other spectra. The signal could be too low to pass the run filter. Thus after obtaining the aligned feature table, this function re-analyzes each spectrum to try and fill in the holes in the aligned feature table.

Usage

```
recover.weaker(filename, loc, aligned.ftrs, pk.times, align.mz.tol,
                 align.chr.tol, this.f1, this.f2, mz.range = NA,
                 chr.range = NA, use.observed.range = TRUE, orig.tol =
                 1e-05, min.bw = NA, max.bw = NA, bandwidth = 0.5,
                 recover.min.count = 3, intensity.weighted=FALSE)
```

Arguments

filename	the cdf file name from which weaker signal is to be recovered.
loc	the location of the filename in the vector of filenames.
aligned.ftrs	matrix, with columns of m/z values, elution times, signal strengths in each spectrum.
pk.times	matrix, with columns of m/z, median elution time, and elution times in each spectrum.
align.mz.tol	the m/z tolerance used in the alignment.
align.chr.tol	the elution time tolerance in the alignment.
this.f1	The matrix which is the output from proc.to.feature().
this.f2	The matrix which is the output from proc.to.feature(). The retention time in this object have been adjusted by the function adjust.time().
orig.tol	The mz.tol parameter provided to the proc.cdf() function. This helps retrieve the intermediate file.
mz.range	The m/z around the feature m/z to search for observations. The default value is NA, in which case 1.5 times the m/z tolerance in the aligned object will be used.
chr.range	The retention time around the feature retention time to search for observations. The default value is NA, in which case 0.5 times the retention time tolerance in the aligned object will be used.
use.observed.range	If the value is TRUE, the actual range of the observed locations of the feature in all the spectra will be used.
min.bw	The minimum bandwidth to use in the kernel smoother.See the details section.
max.bw	The maximum bandwidth to use in the kernel smoother.See the details section.
bandwidth	A value between zero and one. Multiplying this value to the length of the signal along the time axis helps determine the bandwidth in the kernel smoother used for peak identification. See the details section.
recover.min.count	minimum number of raw data points to support a recovery.
intensity.weighted	Whether to use intensity to weight mass density estimation.

Details

For every feature, if it is not present in a spectrum, open the spectrum, and look around the m/z and elution time location of the feature. The observed intensities with m/z and elution time most consistent with the feature are collected. The peak location and intensity is evaluated. For each spectrum, the partially processed file: .rawprof is loaded. This file is the product of the function proc.cdf(). The m/z values are already grouped and the median taken. The function searches around the feature m/z and retention time. When a series of signals is found at an m/z group, kernel smoother is fit along the time axis to determine whether there is one single peak or multiple peaks. The bandwidth of the kernel smoother is determined as follows: multiply the length of the signals by the bandwidth parameter. If the resulting value is between min.bw and max.bw, use that value; else if the value is below min.bw, use min.bw; else if the value is above max.bw, use max.bw. The default values of min.bw and max.bw are NA, in which case min.bw is set to be 1/30 of the retention time range, and max.bw is set to be 1/15 of the retention time range. A modified EM algorithm which allows missing completely at random at certain time points is used for the evaluation of the peak location and area. If a single peak is detected by the kernel smoother, the maximum likelihood normal curve is fitted. If multiple peaks are detected, the EM algorithm finds the normal mixture that best explain the data. After finding the peaks around the target feature, find the closest one to the target feature and record its information in the $aligned.ftrs and $pk.times matrices.

Value

Returns a list object with the following objects in it:

`aligned.ftrs`	A matrix, with columns of m/z values, elution times, and signal strengths in each spectrum.
`pk.times`	A matrix, with columns of m/z, median elution time, and elution times in each spectrum.
`mz.tol`	The m/z tolerance in the aligned object.
`chr.tol`	The elution time tolerance in the aligned object.

Author(s)

Tianwei Yu <tyu8@sph.emory.edu>

`rm.ridge` *Removing long ridges at the same m/z.*

Description

This is an internal function. It substracts a background estimated through kernel smoothing when an EIC continuously span more than half the retention time range.

Usage

```
rm.ridge(x, y2, bw)
```

Arguments

x	Retetion time vector.
y2	Intensity vector.
bw	Bandwidth for the kernel smoother. A very wide one is used here.

Value

A vector of intensity value is returned.

Author(s)

Tianwei Yu <tyu8@emory.edu>

References

Bioinformatics. 25(15):1930-36. BMC Bioinformatics. 11:559.

---

semi.sup *Semi-supervised feature detection*

Description

The semi-supervised procedure utilizes a database of known metabolites and previously detected features to identify features in a new dataset. It is recommended ONLY for experienced users. The user may need to construct the known feature database that strictly follows the format described below.

Usage

```
semi.sup(folder, output_path, file.pattern = ".cdf", known.table = NA, n.nodes = 4,
min.exp = 2, min.pres = 0.5, min.run = 12, mz.tol = 1e-5,
baseline.correct.noise.percentile = 0.05, shape.model = "bi-Gaussian", BIC.factor = 2,
baseline.correct = 0, peak.estim.method = "moment", min.bw = NA, max.bw = NA,
sd.cut = c(0.01, 500), sigma.ratio.lim = c(0.01, 100), component.eliminate = 0.01,
moment.power = 1, subs = NULL, align.mz.tol = NA, align.chr.tol = NA,
max.align.mz.diff = 0.01, pre.process = FALSE, recover.mz.range = NA,
recover.chr.range = NA, use.observed.range = TRUE, match.tol.ppm = NA,
new.feature.min.count = 2, recover.min.count = 3, intensity.weighted = FALSE)
```

Arguments

folder	The folder where all CDF files to be processed are located. For example "C:/CDF/this_experiment"
output_path	Path to the output directory
file.pattern	The pattern in the names of the files to be processed. The default is ".cdf". Other formats supported by mzR package can also be used, e.g. "mzML" etc.

`known.table` A data frame containing the known metabolite ions and previously found features. It contains 18 columns: "chemical_formula": the chemical formula if knonw; "HMDB_ID": HMDB ID if known; "KEGG_compound_ID": KEGG compound ID if known; "neutral.mass": the neutral mass if known: "ion.type": the ion form, such as H+, Na+, ..., if known; "m.z": m/z value, either theoretical for known metabolites, or mean observed value for unknown but previously found features; "Number_profiles_processed": the total number of LC/MS profiles that were used to build this database; "Percent_found": in what percentage was this feature found historically amount all data processed in building this database; "mz_min": the minimum m/z value observed for this feature; "mz_max": the maximum m/z value observed for this feature; "RT_mean": the mean retention time observed for this feature; "RT_sd": the standard deviation of retention time observed for this feature; "RT_min": the minimum retention time observed for this feature; "RT_max": the maximum retention time observed for this feature; "int_mean.log.": the mean log intensity observed for this feature; "int_sd.log.": the standard deviation of log intensity observed for this feature; "int_min.log.": the minimum log intensity observed for this feature; "int_max.log.": the maximum log intensity observed for this feature;

`n.nodes` The number of CPU cores to be used through doSNOW.

`min.exp` If a feature is to be included in the final feature table, it must be present in at least this number of spectra.

`min.pres` This is a parameter of thr run filter, to be passed to the function proc.cdf(). Please see the help for proc.cdf() for details.

`min.run` This is a parameter of thr run filter, to be passed to the function proc.cdf(). Please see the help for proc.cdf() for details.

`subs` If not all the CDF files in the folder are to be processed, the user can define a subset using this parameter. For example, subs=15:30, or subs=c(2,4,6,8)

`mz.tol` The user can provide the m/z tolerance level for peak identification. This value is expressed as the percentage of the m/z value. This value, multiplied by the m/z value, becomes the cutoff level. Please see the help for proc.cdf() for details.

`baseline.correct.noise.percentile`
The perenctile of signal strength of those EIC that don't pass the run filter, to be used as the baseline threshold of signal strength. This parameter is passed to proc.cdf()

`shape.model` The mathematical model for the shape of a peak. There are two choices - bi-Gaussian and Gaussian. When the peaks are asymmetric, the bi-Gaussian is better. The default is bi-Gaussian.

`BIC.factor` the factor that is multiplied on the number of parameters to modify the BIC criterion. If larger than 1, models with more peaks are penalized more.

`baseline.correct`
This is a parameter in peak detection. After grouping the observations, the highest observation in each group is found. If the highest is lower than this value, the entire group will be deleted. The default value is NA, which allows the program to search for the cutoff level. Please see the help for proc.cdf() for details.

`peak.estim.method`
the bi-Gaussian peak parameter estimation method, to be passed to subroutine prof.to.features. Two possible values: moment and EM.

`min.bw` The minimum bandwidth in the smoother in prof.to.features(). Please see the help file for prof.to.features() for details.

`max.bw` The maximum bandwidth in the smoother in prof.to.features(). Please see the help file for prof.to.features() for details.

`sd.cut` A parameter for the prof.to.features() function. A vector of two. Features with standard deviation outside the range defined by the two numbers are eliminated.

`sigma.ratio.lim`
A parameter for the prof.to.features() function. A vector of two. It enforces the belief of the range of the ratio between the left-standard deviation and the righ-standard deviation of the bi-Gaussian fuction used to fit the data.

`component.eliminate`
In fitting mixture of bi-Gaussian (or Gaussian) model of an EIC, when a component accounts for a proportion of intensities less than this value, the component will be ignored.

`moment.power` The power parameter for data transformation when fitting the bi-Gaussian or Gaussian mixture model in an EIC.

`align.chr.tol` The user can provide the elution time tolerance level to override the program's selection. This value is in the same unit as the elution time, normaly seconds. Please see the help for match.time() for details.

`align.mz.tol` The user can provide the m/z tolerance level for peak alignment to override the program's selection. This value is expressed as the percentage of the m/z value. This value, multiplied by the m/z value, becomes the cutoff level.Please see the help for feature.align() for details.

`max.align.mz.diff`
As the m/z tolerance in alignment is expressed in relative terms (ppm), it may not be suitable when the m/z range is wide. This parameter limits the tolerance in absolute terms. It mostly influences feature matching in higher m/z range.

`pre.process` Logical. If true, the program will not perform time correction and alignment. It will only generate peak tables for each spectra and save the files. It allows manually dividing the task to multiple machines.

`recover.mz.range`
A parameter of the recover.weaker() function. The m/z around the feature m/z to search for observations. The default value is NA, in which case 1.5 times the m/z tolerance in the aligned object will be used.

`recover.chr.range`
A parameter of the recover.weaker() function. The retention time around the feature retention time to search for observations. The default value is NA, in which case 0.5 times the retention time tolerance in the aligned object will be used.

`use.observed.range`
A parameter of the recover.weaker() function. If the value is TRUE, the actual range of the observed locations of the feature in all the spectra will be used.

`match.tol.ppm` The ppm tolerance to match identified features to known metabolites/features.

`new.feature.min.count`
The number of profiles a new feature must be present for it to be added to the database.

`recover.min.count`
: The minimum time point count for a series of point in the EIC for it to be considered a true feature.

`intensity.weighted`
: Whether to weight the local density by signal intensities.

Details

The function first conducts a unsupervised feature detection in the new dataset. It then matches the newly identified features to the database. Then merging unfound features in the database and the newly found features, a weak signal recovery is performed. The final feature table is used to update the database.

Value

A list is returned.

`features`
: A list object, each component of which being the peak table from a single spectrum.

`features2`
: A list object, each component of which being the peak table from a single spectrum, after elution time correction.

`aligned.ftrs`
: Feature table BEFORE weak signal recovery.

`final.ftrs`
: Feature table after weak signal recovery. This is the end product of the function.

`pk.times`
: Table of feature elution time BEFORE weak signal recovery.

`final.times`
: Table of feature elution time after weak signal recovery.

`mz.tol`
: The input mz.tol value by the user.

`align.mz.tol`
: The m/z tolerance level in the alignment across spectra, either input from the user or automatically selected when the user input is NA.

`align.chr.tol`
: The retention time tolerance level in the alignment across spectra, either input from the user or automatically selected when the user input is NA.

`updated.known.table`
: The known table updated using the newly processed data. It should be used for future datasets generated using the same machine and LC column.

`ftrs.known.table.pairing`
: The paring information between the feature table of the current dataset and the known feature tabel.

`intensity.weighted`
: Whether to weight the local density by signal intensities in the initial peak detection stage.

Author(s)

Tianwei Yu < tianwei.yu@emory.edu>

See Also

cdf.to.ftrs, proc.cdf, prof.to.feature, adjust.time, feature.align, recover.weaker

`semi.sup.2d` *Semi-supervised feature detection using 2D peak detection*

Description

The semi-supervised procedure utilizes a database of known metabolites and previously detected features to identify features in a new dataset. It is recommended ONLY for experienced users. The user may need to construct the known feature database that strictly follows the format described below.

Usage

```
semi.sup.2d(folder, output_path, file.pattern=".cdf", known.table=NA, n.nodes=4,
min.exp=2, mz.cut = 5e-5, rt.cut = 50,mz.search.range =  2e-4, rt.search.range = 200,
intensity.limit.quantile = 0.05, mPower=4, mz.tol=1e-5, subs=NULL, align.mz.tol=NA,
align.chr.tol=NA, max.align.mz.diff=0.01, pre.process=FALSE, recover.mz.range=NA,
recover.chr.range=NA, use.observed.range=TRUE, match.tol.ppm=NA, new.feature.min.count=2,
recover.min.count=3, intensity.weighted=FALSE)
```

Arguments

`folder`	The folder where all CDF files to be processed are located. For example "C:/CDF/this_experiment"
`output_path`	Path to the output directory
`file.pattern`	The pattern in the names of the files to be processed. The default is ".cdf". Other formats supported by mzR package can also be used, e.g. "mzML" etc.
`known.table`	A data frame containing the known metabolite ions and previously found features. It contains 18 columns: "chemical_formula": the chemical formula if knonw; "HMDB_ID": HMDB ID if known; "KEGG_compound_ID": KEGG compound ID if known; "neutral.mass": the neutral mass if known: "ion.type": the ion form, such as H+, Na+, ..., if known; "m.z": m/z value, either theoretical for known metabolites, or mean observed value for unknown but previously found features; "Number_profiles_processed": the total number of LC/MS profiles that were used to build this database; "Percent_found": in what percentage was this feature found historically amount all data processed in building this database; "mz_min": the minimum m/z value observed for this feature; "mz_max": the maximum m/z value observed for this feature; "RT_mean": the mean retention time observed for this feature; "RT_sd": the standard deviation of retention time observed for this feature; "RT_min": the minimum retention time observed for this feature; "RT_max": the maximum retention time observed for this feature; "int_mean.log.": the mean log intensity observed for this feature; "int_sd.log.": the standard deviation of log intensity observed for this feature; "int_min.log.": the minimum log intensity observed for this feature; "int_max.log.": the maximum log intensity observed for this feature;
`n.nodes`	The number of CPU cores to be used through doSNOW.
`min.exp`	If a feature is to be included in the final feature table, it must be present in at least this number of spectra.

`mz.cut` The divided gird width in m/z when calculate the density of each point.

`rt.cut` The divided gird width in RT when calculate the density of each point.

`mz.search.range`
maximum peak width in m/z

`rt.search.range`
(maximum peak width in RT

`intensity.limit.quantile`
intensity threshold

`mPower` The power parameter for data transformation when fitting the bi-Gaussian or Gaussian mixture model in an EIC.

`subs` If not all the CDF files in the folder are to be processed, the user can define a subset using this parameter. For example, subs=15:30, or subs=c(2,4,6,8)

`mz.tol` The user can provide the m/z tolerance level for peak identification. This value is expressed as the percentage of the m/z value. This value, multiplied by the m/z value, becomes the cutoff level. Please see the help for proc.cdf() for details.

`align.mz.tol` The user can provide the m/z tolerance level for peak alignment to override the program's selection. This value is expressed as the percentage of the m/z value. This value, multiplied by the m/z value, becomes the cutoff level.Please see the help for feature.align() for details.

`align.chr.tol` The user can provide the elution time tolerance level to override the program's selection. This value is in the same unit as the elution time, normaly seconds. Please see the help for match.time() for details.

`max.align.mz.diff`
As the m/z tolerance in alignment is expressed in relative terms (ppm), it may not be suitable when the m/z range is wide. This parameter limits the tolerance in absolute terms. It mostly influences feature matching in higher m/z range.

`pre.process` Logical. If true, the program will not perform time correction and alignment. It will only generate peak tables for each spectra and save the files. It allows manually dividing the task to multiple machines.

`recover.mz.range`
A parameter of the recover.weaker() function. The m/z around the feature m/z to search for observations. The default value is NA, in which case 1.5 times the m/z tolerance in the aligned object will be used.

`recover.chr.range`
A parameter of the recover.weaker() function. The retention time around the feature retention time to search for observations. The default value is NA, in which case 0.5 times the retention time tolerance in the aligned object will be used.

`use.observed.range`
A parameter of the recover.weaker() function. If the value is TRUE, the actual range of the observed locations of the feature in all the spectra will be used.

`match.tol.ppm` The ppm tolerance to match identified features to known metabolites/features.

`new.feature.min.count`
The number of profiles a new feature must be present for it to be added to the database.

`recover.min.count`
: The minimum time point count for a series of point in the EIC for it to be considered a true feature.

`intensity.weighted`
: Whether to weight the local density by signal intensities.

Details

The function first conducts a unsupervised feature detection in the new dataset. It then matches the newly identified features to the database. Then merging unfound features in the database and the newly found features, a weak signal recovery is performed. The final feature table is used to update the database.

Value

A list is returned.

`features`
: A list object, each component of which being the peak table from a single spectrum.

`features2`
: A list object, each component of which being the peak table from a single spectrum, after elution time correction.

`aligned.ftrs`
: Feature table BEFORE weak signal recovery.

`final.ftrs`
: Feature table after weak signal recovery. This is the end product of the function.

`pk.times`
: Table of feature elution time BEFORE weak signal recovery.

`final.times`
: Table of feature elution time after weak signal recovery.

`mz.tol`
: The input mz.tol value by the user.

`align.mz.tol`
: The m/z tolerance level in the alignment across spectra, either input from the user or automatically selected when the user input is NA.

`align.chr.tol`
: The retention time tolerance level in the alignment across spectra, either input from the user or automatically selected when the user input is NA.

`updated.known.table`
: The known table updated using the newly processed data. It should be used for future datasets generated using the same machine and LC column.

`ftrs.known.table.pairing`
: The paring information between the feature table of the current dataset and the known feature tabel.

`intensity.weighted`
: Whether to weight the local density by signal intensities in the initial peak detection stage.

Author(s)

Tianwei Yu < tianwei.yu@emory.edu>

See Also

cdf.to.ftrs, proc.cdf, prof.to.feature, adjust.time, feature.align, recover.weaker

semi.sup.learn *Semi-supervised feature detection using machine learning approach.*

Description

The semi-supervised procedure utilizes a database of known metabolites and previously detected features to identify features in a new dataset. It is recommended ONLY for experienced users. The user may need to construct the known feature database that strictly follows the format described below.

Usage

```
semi.sup.learn(folder, output_path, file.pattern=".cdf", known.table=NA, n.nodes=4,
min.exp=2, min.pres=0.3, min.run=4, mz.tol=1e-5, shape.model="bi-Gaussian",
baseline.correct=0, peak.estim.method="moment", min.bw=NA, max.bw=NA, sd.cut=c(0.01,500),
component.eliminate=0.01, moment.power=1, sigma.ratio.lim=c(0.01, 100), subs=NULL,
align.mz.tol=NA, align.chr.tol=NA, max.align.mz.diff=0.01, pre.process=FALSE,
recover.mz.range=NA, recover.chr.range=NA, use.observed.range=TRUE, match.tol.ppm=5,
new.feature.min.count=2, recover.min.count=3, use.learn=TRUE, ridge.smoother.window=50,
smoother.window=c(1, 5, 10),pos.confidence=0.99, neg.confidence=0.99, max.ftrs.to.use=10,
do.grp.reduce=TRUE, remove.bottom.ftrs=0, max.fpr=0.5, min.tpr=0.9,
intensity.weighted=FALSE)
```

Arguments

folder	The folder where all CDF files to be processed are located. For example "C:/CDF/this_experiment"
output_path	Path to the output directory
file.pattern	The pattern in the names of the files to be processed. The default is ".cdf". Other formats supported by mzR package can also be used, e.g. "mzML" etc.
known.table	A data frame containing the known metabolite ions and previously found features. It contains 18 columns: "chemical_formula": the chemical formula if knonw; "HMDB_ID": HMDB ID if known; "KEGG_compound_ID": KEGG compound ID if known; "neutral.mass": the neutral mass if known: "ion.type": the ion form, such as H+, Na+, ..., if known; "m.z": m/z value, either theoretical for known metabolites, or mean observed value for unknown but previously found features; "Number_profiles_processed": the total number of LC/MS profiles that were used to build this database; "Percent_found": in what percentage was this feature found historically amount all data processed in building this database; "mz_min": the minimum m/z value observed for this feature; "mz_max": the maximum m/z value observed for this feature; "RT_mean": the mean retention time observed for this feature; "RT_sd": the standard deviation of retention time observed for this feature; "RT_min": the minimum retention time observed for this feature; "RT_max": the maximum retention time observed for this feature; "int_mean.log.": the mean log intensity observed for this feature; "int_sd.log.": the standard deviation of log intensity observed for this feature; "int_min.log.": the minimum log intensity observed for this feature; "int_max.log.": the maximum log intensity observed for this feature;

`n.nodes` The number of CPU cores to be used through doSNOW.

`min.exp` If a feature is to be included in the final feature table, it must be present in at least this number of spectra.

`min.pres` This is a parameter of thr run filter, to be passed to the function proc.cdf(). Please see the help for proc.cdf() for details.

`min.run` This is a parameter of thr run filter, to be passed to the function proc.cdf(). Please see the help for proc.cdf() for details.

`mz.tol` The user can provide the m/z tolerance level for peak identification. This value is expressed as the percentage of the m/z value. This value, multiplied by the m/z value, becomes the cutoff level. Please see the help for proc.cdf() for details.

`shape.model` The mathematical model for the shape of a peak. There are two choices - "bi-Gaussian" and "Gaussian". When the peaks are asymmetric, the bi-Gaussian is better. The default is "bi-Gaussian".

`baseline.correct`
This is a parameter in peak detection. After grouping the observations, the highest observation in each group is found. If the highest is lower than this value, the entire group will be deleted. The default value is NA, which allows the program to search for the cutoff level. Please see the help for proc.cdf() for details.

`peak.estim.method`
the bi-Gaussian peak parameter estimation method, to be passed to subroutine prof.to.features. Two possible values: moment and EM.

`min.bw` The minimum bandwidth in the smoother in prof.to.features(). Please see the help file for prof.to.features() for details.

`max.bw` The maximum bandwidth in the smoother in prof.to.features(). Please see the help file for prof.to.features() for details.

`sd.cut` A parameter for the prof.to.features() function. A vector of two. Features with standard deviation outside the range defined by the two numbers are eliminated.

`sigma.ratio.lim`
A parameter for the prof.to.features() function. A vector of two. It enforces the belief of the range of the ratio between the left-standard deviation and the righ-standard deviation of the bi-Gaussian fuction used to fit the data.

`subs` If not all the CDF files in the folder are to be processed, the user can define a subset using this parameter. For example, subs=15:30, or subs=c(2,4,6,8)

`component.eliminate`
In fitting mixture of bi-Gaussian (or Gaussian) model of an EIC, when a component accounts for a proportion of intensities less than this value, the component will be ignored.

`moment.power` The power parameter for data transformation when fitting the bi-Gaussian or Gaussian mixture model in an EIC.

`align.chr.tol` The user can provide the elution time tolerance level to override the program's selection. This value is in the same unit as the elution time, normaly seconds. Please see the help for match.time() for details.

`align.mz.tol` The user can provide the m/z tolerance level for peak alignment to override the program's selection. This value is expressed as the percentage of the m/z value. This value, multiplied by the m/z value, becomes the cutoff level.Please see the help for feature.align() for details.

`max.align.mz.diff`
: As the m/z tolerance in alignment is expressed in relative terms (ppm), it may not be suitable when the m/z range is wide. This parameter limits the tolerance in absolute terms. It mostly influences feature matching in higher m/z range.

`pre.process`
: Logical. If true, the program will not perform time correction and alignment. It will only generate peak tables for each spectra and save the files. It allows manually dividing the task to multiple machines.

`recover.mz.range`
: A parameter of the recover.weaker() function. The m/z around the feature m/z to search for observations. The default value is NA, in which case 1.5 times the m/z tolerance in the aligned object will be used.

`recover.chr.range`
: A parameter of the recover.weaker() function. The retention time around the feature retention time to search for observations. The default value is NA, in which case 0.5 times the retention time tolerance in the aligned object will be used.

`use.observed.range`
: A parameter of the recover.weaker() function. If the value is TRUE, the actual range of the observed locations of the feature in all the spectra will be used.

`match.tol.ppm`
: The ppm tolerance to match identified features to known metabolites/features.

`new.feature.min.count`
: The number of profiles a new feature must be present for it to be added to the database.

`recover.min.count`
: The minimum time point count for a series of point in the EIC for it to be considered a true feature.

`use.learn`
: whether to use machine learning approach. The default is TRUE.

`ridge.smoother.window`
: The size of the smoother window used by the kernel smoother to remove long ridge noise from each EIC.

`smoother.window`
: The smoother windows to use in data feature generation.

`pos.confidence`
: The confidence level for the features matched to the known feature list.

`neg.confidence`
: The confidence level for the features not matching to the known feature list.

`max.ftrs.to.use`
: The maximum number of data features to use in a predictive model.

`do.grp.reduce`
: Whether to reduce data features that are similar. It is based on data feature predictability.

`remove.bottom.ftrs`
: The number of worst performing data features to remove before model building.

`max.fpr`
: The proportion of unmatched features to be selected in the feature detection step.

`min.tpr`
: The proportion of matched features to be selected in the feature detection step.

`intensity.weighted`
: Whether to weight the local density by signal intensities in the initial peak detection stage.

Details

The function first conducts a machine-learning feature detection in the new dataset. And the conducts the regular feature alignment, retention time adjustment and weak signal recovery.

Value

A list is returned.

`features` A list object, each component of which being the peak table from a single spectrum.

`features2` A list object, each component of which being the peak table from a single spectrum, after elution time correction.

`aligned.ftrs` Feature table BEFORE weak signal recovery.

`final.ftrs` Feature table after weak signal recovery. This is the end product of the function.

`pk.times` Table of feature elution time BEFORE weak signal recovery.

`final.times` Table of feature elution time after weak signal recovery.

`mz.tol` The input mz.tol value by the user.

`align.mz.tol` The m/z tolerance level in the alignment across spectra, either input from the user or automatically selected when the user input is NA.

`align.chr.tol` The retention time tolerance level in the alignment across spectra, either input from the user or automatically selected when the user input is NA.

`updated.known.table`
The known table updated using the newly processed data. It should be used for future datasets generated using the same machine and LC column.

`ftrs.known.table.pairing`
The paring information between the feature table of the current dataset and the known feature tabel.

Author(s)

Tianwei Yu < tianwei.yu@emory.edu>

See Also

cdf.to.ftrs, semi.sup, learn.cdf, prof.to.feature, adjust.time, feature.align, recover.weaker

`target.search` *Targeted search of metabolites with given m/z and (optional) retention time*

Description

The function conducts targeted search only. The search is based on m/z and (optionally) retention time. If there are sufficient number of peaks (>=100) in each profile, the function will conduct retention time correction and peak alignment, in order to reduce potential redundancies.

Usage

```
target.search(folder, output_path, file.pattern = ".cdf", known.table = NA, n.nodes = 4,
  min.exp = 2, min.bw = NA, max.bw = NA, subs = NULL, align.mz.tol = 2e-05,
  align.chr.tol = 150, max.align.mz.diff = 0.01, recover.mz.range = NA,
  recover.chr.range = NA, use.observed.range = TRUE, match.tol.ppm = 5,
  new.feature.min.count = 2, recover.min.count = 3)
```

Arguments

`folder`	The folder where all CDF files to be processed are located. For example "C:/CDF/this_experiment"
`output_path`	Path to the output directory
`file.pattern`	The pattern in the names of the files to be processed. The default is ".cdf". Other formats supported by mzR package can also be used, e.g. "mzML" etc.
`known.table`	A data frame containing the known metabolite ions and previously found features. It contains 18 columns: "chemical_formula": the chemical formula if knonw; "HMDB_ID": HMDB ID if known; "KEGG_compound_ID": KEGG compound ID if known; "neutral.mass": the neutral mass if known: "ion.type": the ion form, such as H+, Na+, ..., if known; "m.z": m/z value, either theoretical for known metabolites, or mean observed value for unknown but previously found features; "Number_profiles_processed": the total number of LC/MS profiles that were used to build this database; "Percent_found": in what percentage was this feature found historically amount all data processed in building this database; "mz_min": the minimum m/z value observed for this feature; "mz_max": the maximum m/z value observed for this feature; "RT_mean": the mean retention time observed for this feature; "RT_sd": the standard deviation of retention time observed for this feature; "RT_min": the minimum retention time observed for this feature; "RT_max": the maximum retention time observed for this feature; "int_mean.log.": the mean log intensity observed for this feature; "int_sd.log.": the standard deviation of log intensity observed for this feature; "int_min.log.": the minimum log intensity observed for this feature; "int_max.log.": the maximum log intensity observed for this feature;
`n.nodes`	The number of CPU cores to be used through doSNOW.
`min.exp`	If a feature is to be included in the final feature table, it must be present in at least this number of spectra.

`min.bw` The minimum bandwidth in the smoother in prof.to.features(). Please see the help file for prof.to.features() for details.

`max.bw` The maximum bandwidth in the smoother in prof.to.features(). Please see the help file for prof.to.features() for details.

`subs` If not all the CDF files in the folder are to be processed, the user can define a subset using this parameter. For example, subs=15:30, or subs=c(2,4,6,8)

`align.chr.tol` The user can provide the elution time tolerance level to override the program's selection. This value is in the same unit as the elution time, normaly seconds. Please see the help for match.time() for details.

`align.mz.tol` The user can provide the m/z tolerance level for peak alignment to override the program's selection. This value is expressed as the percentage of the m/z value. This value, multiplied by the m/z value, becomes the cutoff level.Please see the help for feature.align() for details.

`max.align.mz.diff`
As the m/z tolerance in alignment is expressed in relative terms (ppm), it may not be suitable when the m/z range is wide. This parameter limits the tolerance in absolute terms. It mostly influences feature matching in higher m/z range.

`recover.mz.range`
A parameter of the recover.weaker() function. The m/z around the feature m/z to search for observations. The default value is NA, in which case 1.5 times the m/z tolerance in the aligned object will be used.

`recover.chr.range`
A parameter of the recover.weaker() function. The retention time around the feature retention time to search for observations. The default value is NA, in which case 0.5 times the retention time tolerance in the aligned object will be used.

`use.observed.range`
A parameter of the recover.weaker() function. If the value is TRUE, the actual range of the observed locations of the feature in all the spectra will be used.

`match.tol.ppm` The ppm tolerance to match identified features to known metabolites/features.

`new.feature.min.count`
The number of profiles a new feature must be present for it to be added to the database.

`recover.min.count`
The minimum time point count for a series of point in the EIC for it to be considered a true feature.

Value

`features` A list object, each component of which being the peak table from a single spectrum.

`filled.ftrs` The target features are filled one by one. Notice this table may contain duplicates if some target features are too close.

`reduced.ftrs` If the number of target features are big enough (>=100 detected in each profile), retention time correction and peak alignments are conducted to generate this feature table without redundancy.

`filled.times` The target features are filled one by one. This is the retention time table. Notice this table may contain duplicates if some target features are too close.

`reduced.times` If the number of target features are big enough (>=100 detected in each profile), retention time correction and peak alignments are conducted to generate this feature table without redundancy. This is the retention time table of the aligned features.

Author(s)

Tianwei Yu < tianwei.yu@emory.edu>

See Also

cdf.to.ftrs, proc.cdf, prof.to.feature, adjust.time, feature.align, recover.weaker

---

`two.step.hybrid` *Two step hybrid feature detection.*

Description

A two-stage hybrid feature detection and alignment procedure, for data generated in multiple batches.

Usage

```
two.step.hybrid(folder, info, min.within.batch.prop.detect=0.4,
min.within.batch.prop.report=0.5, min.batch.prop=0.5, batch.align.mz.tol=1e-5,
batch.align.chr.tol=50, file.pattern=".cdf", known.table=NA, n.nodes=4,
min.pres=0.5, min.run=12, mz.tol=1e-5, baseline.correct.noise.percentile=0.05,
shape.model="bi-Gaussian",baseline.correct=0, peak.estim.method="moment", min.bw=NA,
max.bw=NA, sd.cut=c(0.1, 100), sigma.ratio.lim=c(0.05, 20), component.eliminate=0.01,
moment.power=2, align.mz.tol=NA, align.chr.tol=NA, max.align.mz.diff=0.01,
pre.process=FALSE, recover.mz.range=NA, recover.chr.range=NA, use.observed.range=TRUE,
match.tol.ppm=NA, new.feature.min.count=2, recover.min.count=3)
```

Arguments

`folder` The folder where all CDF files to be processed are located. For example "C:/CDF/this_experiment"

`info` A table with two columns. The first column is the file names, and the second column is the batch label of each file.

`min.within.batch.prop.detect`
A feature needs to be present in at least this proportion of the files, for it to be initially detected as a feature for a batch. This parameter replaces the "min.exp" parameter in semi.sup().

`min.within.batch.prop.report`
A feature needs to be present in at least this proportion of the files, in a proportion of batches controlled by "min.batch.prop", to be included in the final feature table. This parameter replaces the "min.exp" parameter in semi.sup().

`min.batch.prop` A feature needs to be present in at least this proportion of the batches, for it to be considered in the entire data.

`batch.align.mz.tol`
The m/z tolerance in ppm for between-batch alignment.

`batch.align.chr.tol`
The RT tolerance for between-batch alignment.

`file.pattern` The pattern in the names of the files to be processed. The default is ".cdf". Other formats supported by mzR package can also be used, e.g. "mzML" etc.

`known.table` A data frame containing the known metabolite ions and previously found features. It contains 18 columns: "chemical_formula": the chemical formula if knonw; "HMDB_ID": HMDB ID if known; "KEGG_compound_ID": KEGG compound ID if known; "neutral.mass": the neutral mass if known: "ion.type": the ion form, such as H+, Na+, ..., if known; "m.z": m/z value, either theoretical for known metabolites, or mean observed value for unknown but previously found features; "Number_profiles_processed": the total number of LC/MS profiles that were used to build this database; "Percent_found": in what percentage was this feature found historically amount all data processed in building this database; "mz_min": the minimum m/z value observed for this feature; "mz_max": the maximum m/z value observed for this feature; "RT_mean": the mean retention time observed for this feature; "RT_sd": the standard deviation of retention time observed for this feature; "RT_min": the minimum retention time observed for this feature; "RT_max": the maximum retention time observed for this feature; "int_mean.log.": the mean log intensity observed for this feature; "int_sd.log.": the standard deviation of log intensity observed for this feature; "int_min.log.": the minimum log intensity observed for this feature; "int_max.log.": the maximum log intensity observed for this feature;

`n.nodes` The number of CPU cores to be used through doSNOW.

`min.pres` This is a parameter of thr run filter, to be passed to the function proc.cdf(). Please see the help for proc.cdf() for details.

`min.run` This is a parameter of thr run filter, to be passed to the function proc.cdf(). Please see the help for proc.cdf() for details.

`mz.tol` The user can provide the m/z tolerance level for peak identification. This value is expressed as the percentage of the m/z value. This value, multiplied by the m/z value, becomes the cutoff level. Please see the help for proc.cdf() for details.

`baseline.correct.noise.percentile`
The perenctile of signal strength of those EIC that don't pass the run filter, to be used as the baseline threshold of signal strength. This parameter is passed to proc.cdf()

`shape.model` The mathematical model for the shape of a peak. There are two choices - "bi-Gaussian" and "Gaussian". When the peaks are asymmetric, the bi-Gaussian is better. The default is "bi-Gaussian".

`baseline.correct`
This is a parameter in peak detection. After grouping the observations, the highest observation in each group is found. If the highest is lower than this value, the entire group will be deleted. The default value is NA, which allows the program to search for the cutoff level. Please see the help for proc.cdf() for details.

peak.estim.method
: the bi-Gaussian peak parameter estimation method, to be passed to subroutine prof.to.features. Two possible values: moment and EM.

min.bw
: The minimum bandwidth in the smoother in prof.to.features(). Please see the help file for prof.to.features() for details.

max.bw
: The maximum bandwidth in the smoother in prof.to.features(). Please see the help file for prof.to.features() for details.

sd.cut
: A parameter for the prof.to.features() function. A vector of two. Features with standard deviation outside the range defined by the two numbers are eliminated.

sigma.ratio.lim
: A parameter for the prof.to.features() function. A vector of two. It enforces the belief of the range of the ratio between the left-standard deviation and the righ-standard deviation of the bi-Gaussian fuction used to fit the data.

component.eliminate
: In fitting mixture of bi-Gaussian (or Gaussian) model of an EIC, when a component accounts for a proportion of intensities less than this value, the component will be ignored.

moment.power
: The power parameter for data transformation when fitting the bi-Gaussian or Gaussian mixture model in an EIC.

align.chr.tol
: The user can provide the elution time tolerance level to override the program's selection. This value is in the same unit as the elution time, normaly seconds. Please see the help for match.time() for details.

align.mz.tol
: The user can provide the m/z tolerance level for peak alignment to override the program's selection. This value is expressed as the percentage of the m/z value. This value, multiplied by the m/z value, becomes the cutoff level.Please see the help for feature.align() for details.

max.align.mz.diff
: As the m/z tolerance in alignment is expressed in relative terms (ppm), it may not be suitable when the m/z range is wide. This parameter limits the tolerance in absolute terms. It mostly influences feature matching in higher m/z range.

pre.process
: Logical. If true, the program will not perform time correction and alignment. It will only generate peak tables for each spectra and save the files. It allows manually dividing the task to multiple machines.

recover.mz.range
: A parameter of the recover.weaker() function. The m/z around the feature m/z to search for observations. The default value is NA, in which case 1.5 times the m/z tolerance in the aligned object will be used.

recover.chr.range
: A parameter of the recover.weaker() function. The retention time around the feature retention time to search for observations. The default value is NA, in which case 0.5 times the retention time tolerance in the aligned object will be used.

use.observed.range
: A parameter of the recover.weaker() function. If the value is TRUE, the actual range of the observed locations of the feature in all the spectra will be used.

match.tol.ppm
: The ppm tolerance to match identified features to known metabolites/features.

`new.feature.min.count`
: The number of profiles a new feature must be present for it to be added to the database.

`recover.min.count`
: The minimum time point count for a series of point in the EIC for it to be considered a true feature.

Details

The function first conducts hybrid feature detection and alignment in each batch separately. Then a between-batch RT correction and feature alignment is conducted. Weak signal recovery is conducted at the single feature table level.

Value

A list is returned.

`batchwise.results`
: A list. Each item in the list is the product of semi.sup() from a single batch.

`final.ftrs`
: Feature table. This is the end product of the function.

Author(s)

Tianwei Yu < tianwei.yu@emory.edu>

See Also

semi.sup, cdf.to.ftrs, proc.cdf, prof.to.feature, adjust.time, feature.align, recover.weaker

---

two.step.hybrid.2d *Two step hybrid feature detection using 2D peak detection.*

Description

A two-stage hybrid feature detection and alignment procedure, for data generated in multiple batches.

Usage

```
two.step.hybrid.2d(folder, info, min.within.batch.prop.detect=0.4,
min.within.batch.prop.report=0.5, min.batch.prop=0.5, batch.align.mz.tol=1e-5,
batch.align.chr.tol=50, file.pattern=".cdf", known.table=NA, n.nodes=4, mz.cut = 1e-4,
rt.cut = 50, mz.search.range =  5e-4, rt.search.range = 200,
intensity.limit.quantile = 0.05, mPower=4, mz.tol=1e-5, align.mz.tol=NA, align.chr.tol=NA,
max.align.mz.diff=0.01, pre.process=FALSE, recover.mz.range=NA, recover.chr.range=NA,
use.observed.range=TRUE, match.tol.ppm=NA, new.feature.min.count=2, recover.min.count=3)
```

Arguments

folder	The folder where all CDF files to be processed are located. For example "C:/CDF/this_experiment"
info	A table with two columns. The first column is the file names, and the second column is the batch label of each file.
min.within.batch.prop.detect	A feature needs to be present in at least this proportion of the files, for it to be initially detected as a feature for a batch. This parameter replaces the "min.exp" parameter in semi.sup().
min.within.batch.prop.report	A feature needs to be present in at least this proportion of the files, in a proportion of batches controlled by "min.batch.prop", to be included in the final feature table. This parameter replaces the "min.exp" parameter in semi.sup().
min.batch.prop	A feature needs to be present in at least this proportion of the batches, for it to be considered in the entire data.
batch.align.mz.tol	The m/z tolerance in ppm for between-batch alignment.
batch.align.chr.tol	The RT tolerance for between-batch alignment.
file.pattern	The pattern in the names of the files to be processed. The default is ".cdf". Other formats supported by mzR package can also be used, e.g. "mzML" etc.
known.table	A data frame containing the known metabolite ions and previously found features. It contains 18 columns: "chemical_formula": the chemical formula if knonw; "HMDB_ID": HMDB ID if known; "KEGG_compound_ID": KEGG compound ID if known; "neutral.mass": the neutral mass if known: "ion.type": the ion form, such as H+, Na+, ..., if known; "m.z": m/z value, either theoretical for known metabolites, or mean observed value for unknown but previously found features; "Number_profiles_processed": the total number of LC/MS profiles that were used to build this database; "Percent_found": in what percentage was this feature found historically amount all data processed in building this database; "mz_min": the minimum m/z value observed for this feature; "mz_max": the maximum m/z value observed for this feature; "RT_mean": the mean retention time observed for this feature; "RT_sd": the standard deviation of retention time observed for this feature; "RT_min": the minimum retention time observed for this feature; "RT_max": the maximum retention time observed for this feature; "int_mean.log.": the mean log intensity observed for this feature; "int_sd.log.": the standard deviation of log intensity observed for this feature; "int_min.log.": the minimum log intensity observed for this feature; "int_max.log.": the maximum log intensity observed for this feature;
n.nodes	The number of CPU cores to be used through doSNOW.
mz.cut	The divided gird width in m/z when calculate the density of each point.
rt.cut	The divided gird width in RT when calculate the density of each point.
mz.search.range	maximum peak width in m/z
rt.search.range	(maximum peak width in RT

`intensity.limit.quantile`
: intensity threshold

`mPower`
: The power parameter for data transformation when fitting the bi-Gaussian or Gaussian mixture model in an EIC.

`mz.tol`
: The user can provide the m/z tolerance level for peak identification. This value is expressed as the percentage of the m/z value. This value, multiplied by the m/z value, becomes the cutoff level. Please see the help for proc.cdf() for details.

`align.mz.tol`
: The user can provide the m/z tolerance level for peak alignment to override the program's selection. This value is expressed as the percentage of the m/z value. This value, multiplied by the m/z value, becomes the cutoff level.Please see the help for feature.align() for details.

`align.chr.tol`
: The user can provide the elution time tolerance level to override the program's selection. This value is in the same unit as the elution time, normaly seconds. Please see the help for match.time() for details.

`max.align.mz.diff`
: As the m/z tolerance in alignment is expressed in relative terms (ppm), it may not be suitable when the m/z range is wide. This parameter limits the tolerance in absolute terms. It mostly influences feature matching in higher m/z range.

`pre.process`
: Logical. If true, the program will not perform time correction and alignment. It will only generate peak tables for each spectra and save the files. It allows manually dividing the task to multiple machines.

`recover.mz.range`
: A parameter of the recover.weaker() function. The m/z around the feature m/z to search for observations. The default value is NA, in which case 1.5 times the m/z tolerance in the aligned object will be used.

`recover.chr.range`
: A parameter of the recover.weaker() function. The retention time around the feature retention time to search for observations. The default value is NA, in which case 0.5 times the retention time tolerance in the aligned object will be used.

`use.observed.range`
: A parameter of the recover.weaker() function. If the value is TRUE, the actual range of the observed locations of the feature in all the spectra will be used.

`match.tol.ppm`
: The ppm tolerance to match identified features to known metabolites/features.

`new.feature.min.count`
: The number of profiles a new feature must be present for it to be added to the database.

`recover.min.count`
: The minimum time point count for a series of point in the EIC for it to be considered a true feature.

Details

The function first conducts hybrid feature detection and alignment in each batch separately. Then a between-batch RT correction and feature alignment is conducted. Weak signal recovery is conducted at the single feature table level.

Value

A list is returned.

`batchwise.results`
: A list. Each item in the list is the product of semi.sup() from a single batch.

`final.ftrs`
: Feature table. This is the end product of the function.

Author(s)

Tianwei Yu < tianwei.yu@emory.edu>

See Also

semi.sup, cdf.to.ftrs, proc.cdf, prof.to.feature, adjust.time, feature.align, recover.weaker

Index