Package ‘xMSanalyzer’

January 29, 2015

Type Package

Title xMSanalyzer: R package for data analysis, comparison, and annotation of metabolomics data.

Version 2.0.5

Date 2015-1-29

Author Karan Uppal

Maintainer Karan Uppal <kuppal2@emory.edu>

Description xMSanalyzer comprises of utilities that can be classified into four main modules: 1) merging apLCMS or XCMS sample processing results from multiple sets of parameter settings, 2) evaluation of sample quality and feature consistency, 3) feature matching, and 4) characterization of m/z using KEGG REST

License GPL2.0

LazyLoad yes

Depends apLCMS, xcms, XML, R2HTML, limma, RCurl, rgl, doSNOW, mzR, foreach, sva, mixOmics

R topics documented:

`xMSanalyzer-package` *xMSanalyzer*

Description

Set of utilities for improved feature detection, quality assessment, overlap analysis, and batch annotation of metabolomics data.

Details

Package:	xMSanalyzer
Type:	Package
Version:	2.0.5
Date:	2015-1-29
License:	gpl2.0
LazyLoad:	yes

Author(s)

Karan Uppal Maintainer: <kuppal2@emory.edu>

References

Literature or other references for background information Uppal et. al. xMSanalyzer: automated pipeline for improved feature detection and downstream analysis of large-scale, non-targeted metabolomics data, BMC Bioinformatics, 2013 Tianwei Yu. apLCMS - Adaptive Processing of High-Resolution LC/MS Data. Bioinformatics. 2009. C.A. Smith, E.J. Want, G.C. Tong, R. Abagyan, and G. Siuzdak. XCMS: Processing Mass Spectrometry Data for Metabolite Profiling Using Nonlinear Peak Alignment, Matching, and Identification. Analytical Chemistry, 2006. Tautenhahn R, Böttcher C, Neumann S. Highly sensitive feature detection for high resolution LC/MS. BMC Bioinformatics. 2008 Nov 28.

`apLCMS.align` *apLCMS.align*

Description

Call apLCMS function, cdf.to.ftr, at different parameter settings

Usage

```
apLCMS.align(cdfloc, apLCMS.outloc, min.run.list = c(3,3), min.pres.list = c(0.3
minexp = 2, mztol = 1e-5, alignmztol = NA, alignchrtol=NA,
numnodes = 2, run.order.file = NA, subs = NA,filepattern=".cdf",
apLCMSmode = "untargeted", known_table, match_tol_ppm = 5)
```

Arguments

`cdfloc`	The folder where all NetCDF/mzML/mzXML/mzData files to be processed are located. For example "C:/experiment1/cdf/"
`apLCMS.outloc`	The folder where alignment output will be written. For example "C:/experiment1/apLCMSoutput/"
`min.run.list`	List of values for min.run parameter, eg: c(3,6) would run the cdf.to.ftr function at min.run=3 and min.run=6
`min.pres.list`	List of values min.pres, eg: c(0.3,0.8) would run the cdf.to.ftr function at min.run=3 and min.run=6
`minexp`	If a feature is to be included in the final feature table, it must be present in at least this number of spectra, eg:2
`mztol`	The user can provide the m/z tolerance level for peak identification to override the programs selection of the tolerance level. 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.
`alignmztol`	The user can provide the m/z tolerance level for peak alignment to override the programs 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.
`alignchrtol`	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. Default: 10
`numnodes`	Number of nodes to use for processing.
`run.order.file`	Name of a tab-delimited file that includes sample names sorted by the order in which they were run (sample names must match the CDF file names)
`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)
`filepattern`	File format of processed data files. eg: ".cdf", ".mzXML"
`apLCMSmode`	"untargeted" or "hybrid"; Default: "untargeted"
`known_table`	A data frame containing the known metabolite ions and previously found features. Please see documentation of semi.sup() function in apLCMS for more details.
`match_tol_ppm`	The ppm tolerance to match identified features to known metabolites/features. Used by the semi.sup() function in apLCMS. Default: 5

Details

This utility calls the cdf.to.ftr() function in the apLCMS package and performs serial sample processing at multiple combinations of two parameters: min.run (minimum length of elution time for a series of signals grouped by m/z to be considered a feature; default value: 3) and min.pres (minimum proportion of scans in which the signal was present; default values: 0.3, 0.8). The function allows the user to define parameters such as min.exp (minimum number of samples in which a feature is present). This differs from the original apLCMS in that the original only allows one set of parameters, whereas this function allows multiple sets. The resulting tables containing m/z, retention time, and peak intensities in each sample are stored at each parameter combination.

Value

Feature table after weak signal recovery. This is the end product of the function cdf.to.ftr.

Author(s)

Karan Uppal <kuppal2@emory.edu>

References

http://www.sph.emory.edu/apLCMS/

`apLCMS.EIC.plot` *apLCMS.EIC.plot*

Description

Modified version of the EIC plot function in apLCMS

Usage

```
apLCMS.EIC.plot(aligned, rows = NA, colors = NA, transform = "none", subset = NA
minrt = NA, maxrt = NA, min.run = 12, min.pres = 0.5, max.spline.time.points = 1
```

Arguments

`aligned`	Rda object from apLCMS
`rows`	feature row indices eg: c(10,35)
`colors`	color vector eg: c("red","green")
`transform`	Tranformation method eg: "none", "log", "sqrt", or "cuberoot"
`subset`	subset of profiles for which to plot the EICs eg: c(1:6)
`minrt`	minimum retention limit in EIC plot eg:30
`maxrt`	maximum retention time limit in EIC plot eg: 300
`min.run`	same as apLCMS.align
`min.pres`	same as apLCMS.align
`max.spline.time.points`	Maximum number of time points to use for interpolation spline eg: 1000

Details

This function is modified version of the EIC.plot function in apLCMS. Users can define the time range for plotting EICs.

Author(s)

Karan Uppal <kuppal2@emory.edu>; Tianwei Yu

`check.mz.in.replicates` *check.mz.in.replicates*

Description

Function to find metabolic characteristics of individuals.

Usage

```
check.mz.in.replicates(dataA, min.samps=2, min.reps=2, num_replicates=3)
```

Arguments

`dataA`	Sample intensity matrix from apLCMS or XCMS. This should only include columns corresponding to samples. All other information such as m/z, retention time, etc. should be deleted.
`min.samps`	min.samps: minimum number of samples for which a feature signal should be detected in at least min.reps replicates~~
`min.reps`	minimum proportion of replicates in which a signal is present (eg: 0.5 or 1)
`num_replicates`	number of replicats for each sample (eg: 2)

Details

This utility allows identification of rare features that are present in only some biological samples, but are present in majority of the analytical replicates of individual samples as a result of unique environmental exposure. The min.samps and min.reps are user defined values for defining the minimum number of samples and minimum proportion of replicates in which a feature should be detected.

Value

Filtered matrix of features with sample intensities

Author(s)

Karan Uppal <kuppal2@emory.edu>

`countpeaks` *countpeaks*

Description

Count the number of signals in the intensity vector.

Usage

```
countpeaks(intensity_vec, missing_val=0)
```

Arguments

`intensity_vec`	Vector of intensities
`missing_val`	How are the missing values represented in the intensity vector? eg: NA or 0

Value

Number of peaks with signals.

`evaluate.Features` *evaluate.Features*

Description

Evaluates feature consistency within analytical/technical replicates of samples based on PID or CV. Reports quantile distribution of PID or CV within technical replicates across all samples and computes a quantitative reproducibility score, QRscore, which is defined as the ratio of percentage of biological samples for which more than 50 percent of technical replicates have a signal and median PID or CV

Usage

```
evaluate.Features(curdata, numreplicates, min.samp.percent = 0.6,alignment.tool,
impute.bool,missingvalue=0)
```

Arguments

`curdata`	Feature table from apLCMS or XCMS with sample intensities.
`numreplicates`	Number of replicates per sample. eg: 2
`min.samp.percent`	If signal is detected in x proportion of technical replicates, then the missing values are replaced by mean intensity which is calculated using non-missing replicates. Defaul: 0.6
`alignment.tool`	Name of the feature alignment tool eg: "apLCMS" or "XCMS".

`impute.bool`	Logical (TRUE or FALSE). Used for calculating coefficient of variation (CV). Missing values are replaced by the mean of the non-missing intensities if at least 60 replicates have values. For example, if two out of three replicates have non-missing values (intensity >0) and the third replicate has a missing value (intensity=0), then the intensity of the third replicate is replaced by the mean of the intensities of the other two to calculate CV.
`missingvalue`	How are the missing values represented? eg: 0 or NA

Details

The function calculates Percent Intensity Difference (PID) or Percent Coefficient of Variation (CV) if there are two, or more than two replicates, respectively. PID is defined as percent ratio of the absolute difference of replicate intensities and the mean of replicate intensities. CV is defined as percent ratio of standard deviation of replicate intensities and mean of replicate intensities.

Value

Matrix with summary of feature consistency (min,first quartile (25th percentile), mean,median, third quartile (75th percentile), max,numgoodsamples, and QRscore). QRscore=Percent good samples/median PID or CV

Author(s)

Karan Uppal <kuppal2@emory.edu>

`evaluate.Samples` *evaluate.Samples*

Description

Evaluate sample consistency based on Pearson or Spearman Correlation.

Usage

```
evaluate.Samples(curdata, numreplicates, alignment.tool, cormethod = "pearson",
missingvalue = 0, ignore.missing = TRUE, replace.bad.replicates = TRUE)
```

Arguments

`curdata`	feature alignment output matrix from apLCMS or XCMS with intensities
`numreplicates`	number of technical replicates per sample
`alignment.tool`	name of the feature alignment tool eg: "apLCMS" or "XCMS"
`cormethod`	Pearson or Spearman correlation.
`missingvalue`	How are missing values represented? eg: 0 or NA
`ignore.missing`	Should the missing values be ignored while computing pearson correlation? eg: TRUE or FALSE

`replace.bad.replicates`
: Should the bad replicates be replaced by the average of the good ones? For example, if the number of technical replicates is more than two, and one of the replicates is poorly correlated with the other two but the other replicates have correlation greater than the defined threshold, then the bad replicate is replaced by the average of the good ones.

Details

If at least two analytical replicates are present for each biological sample, this function calculates the mean pairwise Pearson correlation coefficient between sample replicates using the built-in cor() function in R. Only the features with no missing values are used to evaluate correlation. Analytical replicates refer to multiple injections from the same biological sample; whereas, samples refer to different biological samples.

Value

returns a matrix of Pearson or Spearman Correlation Coefficients within technical replicates per sample.

Author(s)

Karan Uppal <kuppal2@emory.edu>

`feat.batch.annotation.KEGG` *feat.batch.annotation.KEGG*

Description

Batch annotation of features using KEGG REST for one or more adducts.

Usage

```
feat.batch.annotation.KEGG(dataA, max.mz.diff=10, queryadductlist=c("M+H"),
xMSanalyzer.outloc,numnodes=1,syssleep=1)
```

Arguments

`dataA`	Feature table from apLCMS or XCMS. The first column should be m/z.
`max.mz.diff`	Metlin matching m/z threshold in ppm. eg: 5
`queryadductlist`	List of adducts to be used for searching. eg: c("M+H","M+Na","M+K"), c("positive") for all positive ion adducts, or c("negative") for all negative ion adducts, or c("all") for all adducts as defined in Metlin.
`xMSanalyzer.outloc`	Output location where the HTML and text reports will be written. eg: "C:/experiment1/xMSanalyzero
`numnodes`	Number of computing nodes to use. Default: 1
`syssleep`	Wait time (seconds) in between queries. Default: 1

Details

This utility uses the readHTMLTable() function in the XML package in R and the KEGG REST interface to get the list of compounds and pathways IDs from KEGG. respectively. The output is generated as an HTML report and a text file that includes pathway and compound annotations with links to external databases such as KEGG Compound, KEGG Pathway, PubChem Substance, HMDB, ChEBI, CAS, and LipidMAPS. METLIN is no longer supported due to their updated terms of use. The function takes as input a data frame with a list of input m/z, a user-defined m/z threshold (ppm) to define the minimum and maximum mass range, list of adducts; eg: c("M+H", "M+H-H2O"), and the output folder location. A sample annotation report is available at the software homepage: SampleAnnotation.html

Value

A list is returned.

`html.res`	Annotation report in HTML format
`text.res`	Text delimited annotation report

Author(s)

Karan Uppal <kuppal2@emory.edu>

`find.Overlapping.mzs` *find.Overlapping.mzs*

Description

This function matches features between two datasets using the following user defined criteria: 1) Maximum m/z difference (+/-) ppm 2) Maximum retention time difference in seconds

Usage

```
find.Overlapping.mzs(dataA, dataB, mz.thresh = 10, time.thresh = NA,
alignment.tool=NA)
```

Arguments

`dataA`	apLCMS or XCMS feature table for dataset A
`dataB`	apLCMS or XCMS feature table for dataset B
`mz.thresh`	Maximum m/z difference (+/-) ppm. eg: 10
`time.thresh`	Maximum retention time difference (+/-) secs. eg: 300
`alignment.tool`	Name of the feature alignment tool eg: "apLCMS" or "XCMS" or "NA" Use "NA" if the input matrix includes only m/z or both m/z and retnetion time values.

Details

The find.Overlapping.mzs function operates on two sets of feature lists with m/z and retention times for each feature, denoted by L1 and L2, and iterates over all m/z values in L1 to find those that are within a user defined m/z (ppm) and retention time (sec) threshold in L2. Optionally, the user can match features based on only the m/z values by setting time.thresh=NA. The find.Unique.mzs function uses a similar algorithm to find unique features that are not within a user defined mass and retention time threshold level.

Value

Matrix of overlapping features with columns: index.data.A: index of overlapping m/z in dataset A mz.data.A: m/z in dataset A time.data.A: retention time in dataset A index.data.B: index of overlapping m/z in dataset B mz.data.B: m/z in dataset B time.data.B: retention time in dataset B

Author(s)

Karan Uppal <kuppal2@emory.edu>

See Also

apLCMS.align, XCMS.align, find.Unique.mzs, getVenn

`find.Unique.mzs` *find.Unique.mzs*

Description

This function finds unique m/zs between two datasets.

Usage

```
find.Unique.mzs(dataA, dataB, mz.thresh = 10, time.thresh=NA, alignment.tool)
```

Arguments

`dataA`	apLCMS or XCMS feature table for dataset A
`dataB`	apLCMS or XCMS feature table for dataset B
`mz.thresh`	Maximum m/z difference (+/-) ppm
`time.thresh`	Maximum retention time difference (+/-) seconds
`alignment.tool`	Name of the feature alignment tool eg: "apLCMS" or "XCMS" or "NA" Use "NA" if the input matrix includes only m/z or both m/z and retnetion time values.

Details

The find.Unique.mzs function operates on two sets of feature lists with m/z for each feature, denoted by L1 and L2, and iterates over all m/z values in L1 to find those that are within a user defined m/z (ppm) threshold in L2.

Value

A list is returned:

`uniqueA`	Unique m/zs in dataA
`uniqueB`	Unique m/zs in dataB

Author(s)

Karan Uppal <kuppal2@emory.edu>

See Also

apLCMS.align, XCMS.align, find.Overlapping.mzs

`getCVreplicates` *getCVreplicates*

Description

Evaluate feature consistency based on coefficient of variation, where cv=sd/mean. Only calculate CV for samples for which a signal is detected in at least 60 If signal is detected in 60 which is calculated using non-missing replicates.

Usage

```
getCVreplicates(curdata, alignment.tool, numreplicates,
min.samp.percent=0.6, impute.bool=TRUE, missingvalue)
```

Arguments

`curdata`	Feature alignment output matrix from apLCMS or XCMS with sample intensities
`alignment.tool`	Name of the feature alignment tool eg: "apLCMS" or "XCMS"
`numreplicates`	Number of replicates per sample
`min.samp.percent`	If signal is detected in x proportion of technical replicates, then the missing values are replaced by mean intensity which is calculated using non-missing replicates. eg: 0.7
`impute.bool`	Should the missingvalues be replaced by mean of the other replicates? eg: TRUE or FALSE
`missingvalue`	How are the missing values represented? eg: 0 or NA

Value

Matrix of feature consistency based on CV with columns: mz: m/z of the feature minCV: minimum CV between technical replicates across all samples first_quartileCV: 25th percentile CV medianCV: 50th percentile CV meanCV: average of cofficient of variations between technical replicates per sample across all samples third_quartileCV: 75th percentile CV maxCV: maximum CV between technical replicates across all samples

Author(s)

Karan Uppal <kuppal2@emory.edu>

`getPID` *getPID*

Description

Evaluate feature consistency based on PID. PID is defined as percent ratio of absolute intensity difference to mean intensity deviation to mean intensity. Only samples with no missing values within technical replicates are used to evaluate PID.

Usage

```
getPID(curdata, alignment.tool,missingvalue)
```

Arguments

`curdata`	Feature alignment output matrix from apLCMS or XCMS with sample intensities
`alignment.tool`	Name of the feature alignment tool eg: "apLCMS" or "XCMS"
`missingvalue`	How are the missing values represented? eg: 0 or NA

Value

Matrix of feature consistency based on PID with columns: mz: m/z of the feature min: minimum PID between technical replicates across all samples first_quartile: 25th percentile PID median: 50th percentile PID mean: average of cofficient of variations between technical replicates per sample across all samples third_quartile: 75th percentile PID max: maximum PID between technical replicates across all samples

Author(s)

Karan Uppal <kuppal2@emory.edu>

`getVenn` *getVenn*

Description

This utility calls the find.Overlapping.mzs function and generates a Venn diagram showing the extent of overlap between two datasets.

Usage

```
getVenn(dataA, name_a, dataB, name_b, mz.thresh = 10, time.thresh=30,
alignment.tool, xMSanalyzer.outloc,use.unique.mz=FALSE,plotvenn=TRUE)
```

Arguments

`dataA`	apLCMS or XCMS feature table for dataset A as a data frame.
`name_a`	Name of dataset A (eg: "SetA").
`dataB`	apLCMS or XCMS feature table for dataset B as a data frame.
`name_b`	Name of dataset A (eg: "SetB")
`mz.thresh`	+/- ppm mass tolerance for m/z matching
`time.thresh`	Maximum retention time difference (+/-) secs. eg: 30
`alignment.tool`	"apLCMS" or "XCMS" or "NA". If NA is specified then the first two columns are treated as m/z and retention time, respectively.
`xMSanalyzer.outloc`	Output folder, eg: "C:/experiment1/xMSanalyzeroutput/"
`use.unique.mz`	If "TRUE", the function first finds unique features within each set
`plotvenn`	If "TRUE", the function plots the venn diagram.

Details

This utility calls the find.Overlapping.mzs and find.Unique.mzs functions and generates a Venn diagram showing the extent of overlap between datasets (up to three).

Value

A list is returned.

`common`	Row numbers, m/zs, delta retention times of features that are common between the two datasets.
`commonA`	Overlapping features in dataset A
`uniqueA`	Features that are unique in dataset A
`commonB`	Overlapping features in dataset B
`uniqueB`	Features that are unique in dataset B
`vennCounts`	Output of vennCounts function in limma package

Note

Only the unqiue m/zs within each dataset are used to generate Venn diagram.

Author(s)

Karan Uppal <kuppal2@emory.edu>

References

http://rss.acs.unt.edu/Rdoc/library/limma/html/venn.html

getVennmultiple *getVennmultiple*

Description

This utility calls the find.Overlapping.mzs function and generates a Venn diagram showing the extent of overlap between three datasets.

Usage

```
getVennmultiple(dataA, name_a, dataB, name_b, dataC, name_c, mz.thresh = 10,
time.thresh = 30, alignment.tool, xMSanalyzer.outloc,use.unique.mz=FALSE,
plotvenn=TRUE)
```

Arguments

`dataA`	apLCMS or XCMS feature table for dataset A as a data frame.
`name_a`	Name of dataset A (eg: "SetA").
`dataB`	apLCMS or XCMS feature table for dataset B as a data frame.
`name_b`	Name of dataset B (eg: "SetB")
`dataC`	apLCMS or XCMS feature table for dataset B as a data frame.
`name_c`	Name of dataset C (eg: "SetC")
`mz.thresh`	+/- ppm threshold for m/z matching. eg: 10
`time.thresh`	Maximum retention time difference (+/-) secs. eg: 30
`alignment.tool`	Name of the feature alignment tool eg: "apLCMS" or "XCMS" or "NA" Use "NA" if the input matrix includes only m/z or both m/z and retnetion time values.
`xMSanalyzer.outloc`	xMSanalyzer output location, eg: "C:/experiment1/xMSanalyzeroutput/
`use.unique.mz`	If "TRUE", the function first finds unique features within each set
`plotvenn`	If "TRUE", the function plots the venn diagram.

Value

A list is returned.

`uniqueA`	Features that are unique in dataset A
`uniqueB`	Features that are unique in dataset B
`uniqueC`	Features that are unique in dataset C
`common`	Features that are common
`vennCounts`	Output of vennCounts function in limma package

Note

Only the unqiue m/zs within each dataset are used to generate Venn diagram.

Author(s)

Karan Uppal <kuppal2@emory.edu>

References

http://rss.acs.unt.edu/Rdoc/library/limma/html/venn.html

`merge.Results` *merge.Results*

Description

Function that merges results from different parameter settings.

Usage

```
merge.Results(dataA, dataB, feat.eval.A, feat.eval.B, max.mz.diff = 15,
max.rt.diff = 300,merge.eval.pvalue = 0.05, alignment.tool="apLCMS", numnodes =
mult.test.cor=FALSE,mergecorthresh=0.7, missingvalue=0)
```

Arguments

`dataA`	feature alignment output matrix from apLCMS or XCMS with intensities at parameter settings P1
`dataB`	feature alignment output matrix from apLCMS or XCMS with intensities at parameter settings P2
`feat.eval.A`	feature evaluations results from evaluate.Features for results at parameter settings P1
`feat.eval.B`	feature evaluations results from evaluate.Features for results at parameter settings P2
`max.mz.diff`	+/- mz tolerance in ppm for feature matching
`max.rt.diff`	retention time tolerance for feature matching
`merge.eval.pvalue`	Threshold for defining signifcance level of the paired t-test or the Pearson correlation during the merge stage in xMSanalyzer. The p-value is used to determine whether two features with same m/z and retention time have identical intensity profiles.
`mergecorthresh`	Correlation threshold to be used during the merge stage in xMSanalyzer to determine whether two features with same m/z and retention time have identical intensity profiles.
`alignment.tool`	Name of the feature alignment tool eg: "apLCMS" or "XCMS"
`numnodes`	Number of computing nodes to use. Default: 1
`mult.test.cor`	Should Bonferroni multiple testing correction method be applied for comparing intensities of overlapping m/z? Default: FALSE
`missingvalue`	How are missing values represented. eg: 0 or NA

Details

We use a four-step process to merge features from different parameter settings. In step one, features detected at settings P1 and P2 are combined into one list. In step two, features are grouped by a user-defined m/z tolerance (5 ppm is appropriate for high resolution MS but may not be suitable for lower resolution instruments; for the LTQ-FT/MS, examination of m/z tolerance shows little difference between 5 and 10 ppm). In step three, features are further sub-grouped based on a user-defined retention time tolerance. Users are recommended to use the find.Overlapping.mzs function below to optimize the retention time tolerance threshold. In step four, a paired t-test & a Spearman correlation test is used to compare the intensity levels of the metabolites only for the redundant features that have m/z and retention time within defined tolerance levels as described above. Features with minimum median PID (or median CV; for more than two technical replicates) are chosen as representatives of each sub-group, and added to the final list. This scheme allows identification of unique features, and selection of the most consistent feature as a representative for features that overlap.

Value

Returns a matrix of columns of unique m/z values, elution times, signal strengths in each spectrum after merging results

Author(s)

Karan Uppal <kuppal2@emory.edu>

`XCMS.align.centWave` *XCMS.align.centWave*

Description

Wrapper function for XCMS using the centwave alignment algorithm.

Usage

```
XCMS.align.centWave(cdfloc, XCMS.outloc, ppm.list = c(10), mz.diff.list = c(-0.0
sn.thresh.list = c(10), prefilter.list = c(3, 100), bw.val = c(10),
groupval.method = "medret", step.list = c(0.1), max = 50, minfrac.val = 0.5,
minsamp.val = 1, mzwid.val = 0.25, sleep.val = 0, run.order.file = NA, subs = NA
retcor.method = "obiwarp", retcor.family = "symmetric", retcor.plottype = "devia
peakwidth = c(20, 50), nSlaves=2)
```

Arguments

`cdfloc`	The folder where all CDF/mzXML files to be processed are located. For example "C:/CDF/"
`XCMS.outloc`	The folder where alignment output will be written. For example "C:/CDFoutput/"
`ppm.list`	list containing values for maximal tolerated m/z deviation in consecutive scans, in ppm
`mz.diff.list`	list containing values for the minimum difference for features with retention time overlap. eg: c(0.001,0.1)

`sn.thresh.list`	list containing values for signal to noise ratio cutoff variable. eg: c(3,10)
`prefilter.list`	prefiltering values c(k,l) where mass traces that do not contain at least k peaks with intensity>=l are filtered
`bw.val`	bandwidth value
`groupval.method`	Conflict resolution method while calculating peak values for each group. eg: "medret" or "maxint"
`step.list`	list containing values for the step size. eg: c(0.1,1)
`max`	Value for maxnimum number of peaks per EIC variable. eg: 50
`minfrac.val`	minimum fraction of samples necessary in at least one of the sample groups for it to be a valid group
`minsamp.val`	minimum number of samples necessary in at least one of the sample groups for it to be a valid group
`mzwid.val`	width of overlapping m/z slices to use for creating peak density chromatograms and grouping peaks across samples
`sleep.val`	seconds to pause between plotting successive steps of the peak grouping algorithm. peaks are plotted as points showing relative intensity. identified groups are flanked by dotted vertical lines.
`run.order.file`	Name of a tab-delimited file that includes sample names sorted by the order in which they were run(sample names must match the CDF file names)
`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)
`retcor.method`	Method for aligning retention times across samples. eg: "loess" or "obiwarp"
`retcor.family`	Used by matchedFilter alignment method. Use "gaussian" to perform fitting by least squares without outlier removal. Or "symmetric" to use a redescending M estimator with Tukey's biweight function that allows outlier removal.
`retcor.plottype`	Used by both matchedFilter and centWave alignment methods. eg: "deviation" or "mdevden"
`peakwidth`	Chromtagrophic peak width in seconds. eg: c(20,50)
`nSlaves`	Number of computing cores to be used. eg: 2

Details

This is a wrapper function based on the xcms Bioconductor package for preprocessing/analysis of mass spectral data. The resulting tables containing m/z, retention time, and mean peak intensities in each sample are stored at each parameter combination.

Value

A matrix, with columns of m/z values, elution times, mean signal strengths in each spectrum

Note

Please refer to the xcms manual in Bioconductor for more details.

Author(s)

Karan Uppal

References

Tautenhahn R, Bottcher C, Neumann S. Highly sensitive feature detection for high resolution LC/MS. BMC Bioinformatics. 2008 Nov 28.

`XCMS.align.matchedFilter` *XCMS.align.matchedFilter*

Description

Runs XCMS using the matchedFilter alignment algorithm at different parameter settings.

Usage

```
XCMS.align.matchedFilter(cdfloc, XCMS.outloc, step.list = c(0.001), mz.diff.list
sn.thresh.list = c(3), max =50, bw.val = c(10), minfrac.val = 0.5, minsamp.val =
mzwid.val = 0.25, sleep.val = 0, run.order.file = NA, subs = NA, retcor.family =
retcor.plottype = "mdevden", groupval.method = "medret")
```

Arguments

`cdfloc`	The folder where all CDF/mzXML files to be processed are located. For example "C:/CDF/"
`XCMS.outloc`	The folder where alignment output will be written. For example "C:/CDFoutput/"
`step.list`	list containing values for the step size
`mz.diff.list`	list containing values for the minimum difference for features with retention time overlap
`sn.thresh.list`	list containing values for signal to noise ratio cutoff variable
`max`	Value for maxnimum number of peaks per EIC variable: eg: 5
`bw.val`	bandwidth value
`minfrac.val`	minimum fraction of samples necessary in at least one of the sample groups for it to be a valid group
`minsamp.val`	minimum number of samples necessary in at least one of the sample groups for it to be a valid group
`mzwid.val`	width of overlapping m/z slices to use for creating peak density chromatograms and grouping peaks across samples
`sleep.val`	seconds to pause between plotting successive steps of the peak grouping algorithm. peaks are plotted as points showing relative intensity. identified groups are flanked by dotted vertical lines.
`run.order.file`	Name of a tab-delimited file that includes sample names sorted by the order in which they were run(sample names must match the CDF file names)

`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)
`retcor.family`	Used by matchedFilter alignment method. Use "gaussian" to perform fitting by least squares without outlier removal. Or "symmetric" to use a redescending M estimator with Tukey's biweight function that allows outlier removal.
`retcor.plottype`	Used by both matchedFilter and centWave alignment methods. eg: "deviation" or "mdevden"
`groupval.method`	Conflict resolution method while calculating peak values for each group. eg: "medret" or "maxint"

Details

This is a wrapper function based on the xcms Bioconductor package for preprocessing/analysis of mass spectral data. The XCMS.align utility performs serial sample processing at multiple combinations of four parameters: step (the step size; default values: 0.001, 0.01, 0.1), mzdiff (minimum difference for features with retention time overlap; default values: 0.001, 0.01, 0.1), snthresh (signal-to-noise ratio cutoff; default values: 3, 6, 10), and max (maximum number of peaks per EIC; default values: 5, 10). The resulting tables containing m/z, retention time, and mean peak intensities in each sample are stored at each parameter combination.

Value

A matrix, with columns of m/z values, elution times, mean signal strengths in each spectrum

Note

Please refer to the xcms manual in Bioconductor for more details.

Author(s)

Karan Uppal <kuppal2@emory.edu>

`xMSwrapper` *xMSwrapper*

Description

Tells user about the usage of wrapper functions for apLCMS and XCMS.

Usage

```
xMSwrapper()
```

Note

Shows the usage for using apLCMS and XCMS wrapper functions

Author(s)

Karan Uppal <kuppal2@emory.edu>

References

put references to the literature/web site here

See Also

xMSwrapper.apLCMS, xMSwrapper.XCMS.centWave, xMSwrapper.XCMS.matchedFilter

---

`xMSwrapper.apLCMS` *xMSwrapper.apLCMS*

Description

Wrapper function based on apLCMS.align,evaluate.Features, evaluate.Samples,merge.Results, and feat.batch.annotation.

Usage

```
xMSwrapper.apLCMS(cdfloc, apLCMS.outloc, xMSanalyzer.outloc,
                 min.run.list = c(4,3), min.pres.list = c(0.5, 0.8),
                 minexp.pct = 0.1, mztol = NA, alignmztol = NA,
                 alignchrtol = NA, numnodes = NA, run.order.file = NA,
                 apLCMSmode = "untargeted", known_table = NA,
                 match_tol_ppm = 5, max.mz.diff = 15, max.rt.diff =
                 300, merge.eval.pvalue = 0.2, mergecorthresh = 0.7,
                 deltamzminmax.tol = 10, subs = NA, num_replicates = 3,
                 mz.tolerance.dbmatch = 10, adduct.list = c("M+H"),
                 samp.filt.thresh = 0.7, feat.filt.thresh = 50,
                 cormethod = "pearson", mult.test.cor = TRUE,
                 missingvalue = 0, ignore.missing = TRUE, filepattern =
                 ".cdf", sample_info_file = NA, refMZ = NA,
                 refMZ.mz.diff = 10, refMZ.time.diff = NA,
                 void.vol.timethresh = 30, replacezeroswithNA = TRUE,
                 scoreweight = 30,charge_type="pos", syssleep=0.5)
```

Arguments

`cdfloc`	The folder where all NetCDF/mzML/mzXML/mzData) files to be processed are located. For example "C:/CDF/"
`apLCMS.outloc`	The folder where apLCMS feature alignment output will be written. For example "C:/apLCMSoutput/"
`xMSanalyzer.outloc`	The folder where xMSanalyzer output will be written. For example "C:/xMSanalyzeroutput/"
`min.run.list`	List of values for min.run parameter, eg: c(3,6) would run the cdf.to.ftr function at min.run=3 and min.run=6

`min.pres.list`	List of values min.pres, eg: c(0.3,0.8) would run the cdf.to.ftr function at min.run=3 and min.run=6
`minexp.pct`	If a feature is to be included in the final feature table, it must be present in at least this fraction of spectra. eg: 0.2
`mztol`	The user can provide the m/z tolerance level for peak identification to override the programs selection of the tolerance level. 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.
`alignmztol`	The user can provide the m/z tolerance level for peak alignment to override the programs 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.
`numnodes`	Number of computational nodes to use for sample processing. eg: 2
`run.order.file`	Name of a tab-delimited file that includes sample names sorted by the order in which they were run (sample names must match the CDF file names)
`max.mz.diff`	+/- mz tolerance in ppm for feature matching
`max.rt.diff`	retention time tolerance for feature matching
`merge.eval.pvalue`	Threshold for defining signifcance level of the paired t-test or the Pearson correlation during the merge stage in xMSanalyzer. The p-value is used to determine whether two features with same m/z and retention time have identical intensity profiles.
`mergecorthresh`	Correlation threshold to be used during the merge stage in xMSanalyzer to determine whether two features with same m/z and retention time have identical intensity profiles.
`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)
`num_replicates`	Number of replicates per sample
`mz.tolerance.dbmatch`	m/z threshold for database matching
`adduct.list`	List of adducts for matching m/zs in KEGG. eg: c("M+H", "M+Na")
`samp.filt.thresh`	Threshold for filtering samples based on Pearson correlation between technical replicates. eg: 0.7
`feat.filt.thresh`	Threshold for filtering samples based on percent intensity difference or coefficient of variation. eg: 50
`cormethod`	Method for determing correlation between technical replicates. eg: "pearson" or "spearman
`mult.test.cor`	Should Bonferroni multiple testing correction method be applied for comparing intensities of overlapping m/z? Default: FALSE
`missingvalue`	How are missing values represented? eg: 0 or NA

`ignore.missing`	Should the missing values be ignored while computing pearson correlation. eg: TRUE or FALSE
`filepattern`	File format of spectral data files. eg: ".cdf", ".mzXML"
`alignchrtol`	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. Default: 10
`apLCMSmode`	"untargeted" or "hybrid"; Default: "untargeted"
`known_table`	A data frame containing the known metabolite ions and previously found features. Please see documentation of semi.sup() function in apLCMS for more details.
`match_tol_ppm`	The ppm tolerance to match identified features to known metabolites/features. Used by the semi.sup() function in apLCMS. Default: 5
`deltamzminmax.tol`	Maximum allowed delta ppm between mz.min and mz.max. Eg: 10
`refMZ`	Full path of the file with m/z of the targeted chemicals to search for. If the value is "NA", then the list of metabolites in the data(example_target_list) is used. The input file should be formatted as data(example_target_list): Column A: m/z of the targeted metabolite (required) Column B: retention time (Optional) Column C: Name of the metabolite (required)
`refMZ.mz.diff`	The ppm tolerance to search for targeted metabolites/features in xMSanalyzer. Default: 10
`refMZ.time.diff`	The time tolerance to search for targeted metabolites/features in xMSanalyzer. Default: NA
`void.vol.timethresh`	Threshold for void volume. The program searches for the void volume cutoff within the defined time limit. Default: 30
`replacezeroswithNA`	Should 0s be treated as missing values during ComBat (TRUE or FALSE). Default: TRUE
`scoreweight`	The w parameter in the scoring function defined in the xMSanalyzer manuscript. Uppal 2013, BMC Bioinformatiocs. Default: 30
`sample_info_file`	File listing the order in which the samples were run. The format of the file should be as follows: Column A:File names matching the CDF/mzXML files Column B: Sample ID Column C: Batch number (This column should be labeled "Batch") Column D: Additional covariates to adjust for (Optional)
`charge_type`	Ionization mode. "pos" or "neg" Default: "pos"
`syssleep`	Sleep time in between KEGG REST queries. Default: "0.5"

Details

The wrapper function includes five stages to utilize information from the technical replicates to optimize the data extraction process, enhance data quality, search for targeted features/metabollites, perform QA & QC evaluations including batch-effect evaluation & correction. : 1) features are extracted using different parameters 2) results from each parameter setting are evaluated for sample quality & feature consistency, 3) filtered results from individual settings are merged to obtain a combined feature table; the optimization score that takes into account the number of features and average CV (see Uppal 2013) is used to determine the most optimal set of parameters. 4) A targeted feature table using the list of m/z in the "refMZ" file is generated 5) Quality measures of each featue includes: number of samples including replicates with non-missing values (NumPres.All.Samples), number of biological samples for which more than 60 median coefficient of variation (CV) within technical replicates summarized across all samples; Qscore (Quality score which is calculated using NumPres.All.Samples, NumPres.Biol.Samples, median CV, and delta ppm between mz.min and mz.max; Higher is better) Users have the option to filter poor quality samples and features based on correlation between technical replicates and feature reproducibility measures such as PID or CV, respectively.

Value

A list is returned.

`apLCMS.merged.res`	Merged feature table, P1 U P2 where P1 and P2 are two sets of parameter settings. Please note that the four columns include the characteristics of the feature from apLCMS. The "npeaks" column includes the number of samples with a non-missing intensity. The "QRscore" is defined as the ratio of percentage of biological samples for which more than 50 percent of technical replicates have a signal and median PID or CV. QRscore=Percent good samples/median PID or CV
`apLCMS.ind.res`	List with results from individual parameter settings.
`apLCMS.ind.res.filtered`	List with results from individual parameter settings after filtering.
`annot.res`	List with annotation results after merging results.
`feat.eval.ind`	List with feature evaluation results based on CV or PID at each parameter setting.
`sample.eval.ind`	List with sample evaluation results at each parameter setting based on correlation between technical replicates.

Outputs the following to apLCMS.outloc: -apLCMS results at each parameter settings Outputs the following to xMSanalyzer.outloc: -feature consistency results -sample evaluation results -feature list after merging results from parameter settings A and B -merge summary

Author(s)

Karan Uppal <kuppal2@emory.edu>

`xMSwrapper.XCMS.centWave` *xMSwrapper.XCMS.centWave*

Description

Wrapper function for XCMS using the centwave alignment algorithm, evaluate.Features,evaluate.Samples,merge.Results and feat.batch.annotation

Usage

```
xMSwrapper.XCMS.centWave(cdfloc, XCMS.outloc, xMSanalyzer.outloc,
ppm.list =c(10, 25, 30), mz.diff.list = c(-0.001, 0.1), sn.thresh.list = c(3, 5,
prefilter.list = c(3,100), bw.val = c(10, 30), groupval.method = "medret",
step.list = c(0.1, 1), max = 50, minfrac.val = 0.5,minsamp.val = 2, mzwid.val =
retcor.method = "obiwarp", retcor.family ="symmetric", retcor.plottype = "deviat
peakwidth = c(20, 50), numnodes = 2, run.order.file = NA,max.mz.diff = 15,
max.rt.diff = 300, merge.eval.pvalue = 0.2, mergecorthresh = 0.7,
deltamzminmax.tol = 10,num_replicates = 2, subs = NA, mz.tolerance.dbmatch =10,
adduct.list = c("M+H"), samp.filt.thresh = 0.7,feat.filt.thresh = 50,
cormethod = "pearson",mult.test.cor = TRUE, missingvalue = 0, ignore.missing= TR
sample_info_file = NA, refMZ = NA,refMZ.mz.diff = 10, refMZ.time.diff = NA,
void.vol.timethresh = 30, replacezeroswithNA = TRUE, scoreweight = 30,
filepattern = ".cdf", charge_type="pos",minexp.pct=0.1,syssleep=0.5)
```

Arguments

`cdfloc`	The folder where all CDF/mzXML files to be processed are located. For example "C:/CDF/"
`XCMS.outloc`	The folder where alignment output will be written. For example "C:/CDFoutput/"
`xMSanalyzer.outloc`	The folder where xMSanalyzer output will be written. For example "C:/xMSanalyzeroutput/"
`ppm.list`	list containing values for maximal tolerated m/z deviation in consecutive scans, in ppm
`mz.diff.list`	list containing values for the minimum difference for features with retention time overlap
`sn.thresh.list`	list containing values for signal to noise ratio cutoff variable
`prefilter.list`	prefiltering values c(k,l) where mass traces that do not contain at least k peaks with intensity>=l are filtered
`bw.val`	bandwidth value
`groupval.method`	Conflict resolution method while calculating peak values for each group. eg: "medret" or "maxint"
`step.list`	list containing values for the step size
`max`	list containing values for maxnimum number of peaks per EIC variable
`minfrac.val`	minimum fraction of samples necessary in at least one of the sample groups for it to be a valid group
`minsamp.val`	minimum number of samples necessary in at least one of the sample groups for it to be a valid group
`mzwid.val`	width of overlapping m/z slices to use for creating peak density chromatograms and grouping peaks across samples
`sleep.val`	seconds to pause between plotting successive steps of the peak grouping algorithm. peaks are plotted as points showing relative intensity. identified groups are flanked by dotted vertical lines.
`retcor.method`	Method for aligning retention times across samples. eg: "loess" or "obiwarp"

`retcor.family`	Used by matchedFilter alignment method. Use "gaussian" to perform fitting by least squares without outlier removal. Or "symmetric" to use a redescending M estimator with Tukey's biweight function that allows outlier removal.
`retcor.plottype`	Used by both matchedFilter and centWave alignment methods. eg: "deviation" or "mdevden"
`peakwidth`	Chromtagrophic peak width in seconds. eg: c(20,50)
`numnodes`	Number of computing nodes to use. eg: 1
`run.order.file`	Name of a tab-delimited file that includes sample names sorted by the order in which they were run (sample names must match the CDF file names)
`max.mz.diff`	+/- mz tolerance in ppm for feature matching
`max.rt.diff`	retention time tolerance for feature matching
`merge.eval.pvalue`	Threshold for defining signifcance level of the paired t-test or the Pearson correlation during the merge stage in xMSanalyzer. The p-value is used to determine whether two features with same m/z and retention time have identical intensity profiles.
`mergecorthresh`	Correlation threshold to be used during the merge stage in xMSanalyzer to determine whether two features with same m/z and retention time have identical intensity profiles.
`num_replicates`	Number of replicates per sample
`subs`	If not all the CDF files in the folder are to be processed, the user can define a subset using this parameter.
`mz.tolerance.dbmatch`	m/z threshold for database matching
`adduct.list`	List of adducts for matching m/zs in KEGG. eg: c("M+H","M+H-H2O")
`samp.filt.thresh`	Threshold for filtering samples based on Pearson correlation between technical replicates. eg: 0.7
`feat.filt.thresh`	Threshold for filtering samples based on percent intensity difference or coefficient of variation. eg: 50
`cormethod`	Method for determing correlation between technical replicates. eg: "pearson" or "spearman
`mult.test.cor`	Should Bonferroni multiple testing correction method be applied for comparing intensities of overlapping m/z? Default: FALSE
`missingvalue`	How are missing values represented? eg: 0 or NA
`ignore.missing`	Should the missing values be ignored while computing pearson correlation. eg: TRUE or FALSE
`deltamzminmax.tol`	Maximum allowed delta ppm between mz.min and mz.max. Eg: 10

`refMZ`	Full path of the file with m/z of the targeted chemicals to search for. If the value is "NA", then the list of metabolites in the data(example_target_list) is used. The input file should be formatted as data(example_target_list): Column A: m/z of the targeted metabolite (required) Column B: retention time (Optional) Column C: Name of the metabolite (required)
`refMZ.mz.diff`	The ppm tolerance to search for targeted metabolites/features in xMSanalyzer. Default: 10
`refMZ.time.diff`	The time tolerance to search for targeted metabolites/features in xMSanalyzer. Default: NA
`void.vol.timethresh`	Time threshold for void volume. The program searches for the void volume cutoff within the defined time limit. Default: 30
`replacezeroswithNA`	Should 0s be treated as missing values during ComBat (TRUE or FALSE). Default: TRUE
`scoreweight`	The w parameter in the scoring function defined in the xMSanalyzer manuscript. Uppal 2013, BMC Bioinformatiocs. Default: 30
`sample_info_file`	File listing the order in which the samples were run. The format of the file should be as follows: Column A:File names matching the CDF/mzXML files Column B: Sample ID Column C: Batch number (This column should be labeled "Batch") Column D: Additional covariates to adjust for (Optional)
`filepattern`	File format of spectral data files. eg: ".cdf", ".mzXML"
`charge_type`	Ionization mode. "pos" or "neg" Default: "pos"
`minexp.pct`	Minimum fraction of samples in which the signal should be present. Default: "0.1"
`syssleep`	Sleep time in between KEGG REST queries. Default: "0.5"

Details

The wrapper function includes five stages to utilize information from the technical replicates to optimize the data extraction process, enhance data quality, search for targeted features/metabollites, perform QA & QC evaluations including batch-effect evaluation & correction. : 1) features are extracted using different parameters 2) results from each parameter setting are evaluated for sample quality & feature consistency, 3) filtered results from individual settings are merged to obtain a combined feature table; the optimization score that takes into account the number of features and average CV (see Uppal 2013) is used to determine the most optimal set of parameters. 4) A targeted feature table using the list of m/z in the "refMZ" file is generated 5) Quality measures of each featue includes: number of samples including replicates with non-missing values (NumPres.All.Samples), number of biological samples for which more than 60 median coefficient of variation (CV) within technical replicates summarized across all samples; Qscore (Quality score which is calculated using NumPres.All.Samples, NumPres.Biol.Samples, median CV, and delta ppm between mz.min and mz.max; Higher is better) Users have the option to filter poor quality samples and features based on correlation between technical replicates and feature reproducibility measures such as PID or CV, respectively.

Value

A list is returned.

`XCMS.merged.res`	Merged feature table, P1 U P2 where P1 and P2 are two sets of parameter settings. The "QRscore" is defined as the ratio of percentage of biological samples for which more than 50 percent of technical replicates have a signal and median PID or CV. QRscore=Percent good samples/median PID or CV
`XCMS.ind.res`	List with results from individual parameter settings.
`XCMS.ind.res.filtered`	List with results from individual parameter settings after filtering.
`annot.res`	List with annotation results after merging results.
`feat.eval.ind`	List with feature evaluation results based on CV or PID at each parameter setting.
`sample.eval.ind`	List with sample evaluation results at each parameter setting based on correlation between technical replicates.

Outputs XCMS results at each parameter settings to XCMS.outloc and the following to xMSanalyzer.outloc: -feature consistency results -sample evaluation results -feature list after merging results from parameter settings A and B -merge summary

Author(s)

Karan Uppal <kuppal2@emory.edu>

`xMSwrapper.XCMS.matchedFilter`
xMSwrapper.XCMS.matchedFilter

Description

Wrapper function for XCMS based on matched filter alignment algorithm, evaluate.Features,evaluate.Samples,merge.Res
and feat.batch.annotation

Usage

```
xMSwrapper.XCMS.matchedFilter(cdfloc, XCMS.outloc, xMSanalyzer.outloc, step.list
        c(0.001, 0.01, 0.1), mz.diff.list = c(0.001, 0.01,0.1), sn.thresh.list
 max = 50, bw = c(10), minfrac.val = 0.5, minsamp = 2, mzwid =0.25, sleep = 0,
 retcor.family = "symmetric",retcor.plottype = "mdevden", groupval.method ="medr
 numnodes = 2, run.order.file = NA,max.mz.diff = 15, max.rt.diff = 300,
 merge.eval.pvalue= 0.2, mergecorthresh = 0.7, deltamzminmax.tol = 10,
 num_replicates = 2, subs = NA, mz.tolerance.dbmatch =10, adduct.list = c("M+H")
 samp.filt.thresh = 0.7, feat.filt.thresh = 50, cormethod = "pearson",
        mult.test.cor = TRUE, missingvalue = 0, ignore.missing= TRUE, sample_in
 refMZ = NA, refMZ.mz.diff = 10, refMZ.time.diff = NA,
        void.vol.timethresh = 30, replacezeroswithNA = TRUE,
        scoreweight = 30, filepattern = ".cdf", charge_type="pos",minexp.pct=0.
```

Arguments

`cdfloc`	The folder where all CDF/mzXML files to be processed are located. eg: "C:/CDF/". Use "cdfloc=NA" if the XCMS results already exist in XCMS.outloc.
`XCMS.outloc`	The folder where XCMS output will be written. eg: "C:/XCMSoutput/"
`xMSanalyzer.outloc`	The folder where xMSanalyzer output will be written. eg: "C:/xMSanalyzeroutput/"
`step.list`	list containing values for the step size
`mz.diff.list`	list containing values for the minimum difference for features with retention time overlap
`sn.thresh.list`	list containing values for signal to noise ratio cutoff variable
`max`	Value for maxnimum number of peaks per EIC variable
`bw`	bandwidth value list. eg: c(10,30)
`minfrac.val`	minimum fraction of samples necessary in at least the sample groups for it to be a valid group
`minsamp`	minimum number of samples necessary in at least one of the sample groups for it to be a valid group
`mzwid`	width of overlapping m/z slices to use for creating peak density chromatograms and grouping peaks across samples
`sleep`	seconds to pause between plotting successive steps of the peak grouping algorithm. Peaks are plotted as points showing relative intensity. identified groups are flanked by dotted vertical lines.
`retcor.family`	Used by matchedFilter alignment method. Use "gaussian" to perform fitting by least squares without outlier removal. Or "symmetric" to use a redescending M estimator with Tukey's biweight function that allows outlier removal.
`retcor.plottype`	Used by both matchedFilter and centWave alignment methods. eg: "deviation" or "mdevden"
`groupval.method`	Conflict resolution method while calculating peak values for each group. eg: "medret" or "maxint"
`numnodes`	Number of computing nodes to use. eg: 1
`run.order.file`	Name of a tab-delimited file that includes sample names sorted by the order in which they were run (sample names must match the CDF file names)
`max.mz.diff`	+/- mz tolerance in ppm for feature matching
`max.rt.diff`	retention time tolerance for feature matching
`merge.eval.pvalue`	Threshold for defining signifcance level of the paired t-test or the Pearson correlation during the merge stage in xMSanalyzer. The p-value is used to determine whether two features with same m/z and retention time have identical intensity profiles.
`mergecorthresh`	Correlation threshold to be used during the merge stage in xMSanalyzer to determine whether two features with same m/z and retention time have identical intensity profiles.

`num_replicates`	Number of replicates per sample
`subs`	If not all the CDF files in the folder are to be processed, the user can define a subset using this parameter.
`mz.tolerance.dbmatch`	m/z threshold for database matching
`adduct.list`	List of adducts for matching m/zs in KEGG. eg: c("M+H","M+H-H2O")
`samp.filt.thresh`	Threshold for filtering samples based on Pearson correlation between technical replicates. eg: 0.7
`feat.filt.thresh`	Threshold for filtering samples based on percent intensity difference or coefficient of variation. eg: 50
`cormethod`	Method for determing correlation between technical replicates. eg: "pearson" or "spearman
`mult.test.cor`	Should Bonferroni multiple testing correction method be applied for comparing intensities of overlapping m/z? Default: FALSE
`missingvalue`	How are missing values represented? eg: 0 or NA
`ignore.missing`	Should the missing values be ignored while computing pearson correlation. eg: TRUE or FALSE
`deltamzminmax.tol`	Maximum allowed delta ppm between mz.min and mz.max. Eg: 10
`refMZ`	Full path of the file with m/z of the targeted chemicals to search for. If the value is "NA", then the list of metabolites in the data(example_target_list) is used. The input file should be formatted as data(example_target_list): Column A: m/z of the targeted metabolite (required) Column B: retention time (Optional) Column C: Name of the metabolite (required)
`refMZ.mz.diff`	The ppm tolerance to search for targeted metabolites/features in xMSanalyzer. Default: 10
`refMZ.time.diff`	The time tolerance to search for targeted metabolites/features in xMSanalyzer. Default: NA
`void.vol.timethresh`	Time threshold for void volume. The program searches for the void volume cutoff within the defined time limit. Default: 30
`replacezeroswithNA`	Should 0s be treated as missing values during ComBat (TRUE or FALSE). Default: TRUE
`scoreweight`	The w parameter in the scoring function defined in the xMSanalyzer manuscript. Uppal 2013, BMC Bioinformatiocs. Default: 30
`sample_info_file`	File listing the order in which the samples were run. The format of the file should be as follows: Column A:File names matching the CDF/mzXML files Column B: Sample ID Column C: Batch number (This column should be labeled "Batch") Column D: Additional covariates to adjust for (Optional)
`filepattern`	File format of spectral data files. eg: ".cdf", ".mzXML"

`charge_type`	Ionization mode. "pos" or "neg" Default: "pos"
`minexp.pct`	Minimum fraction of samples in which the signal should be present. Default: "0.1"
`syssleep`	Sleep time in between KEGG REST queries. Default: "0.5"

Details

The wrapper function includes five stages to utilize information from the technical replicates to optimize the data extraction process, enhance data quality, search for targeted features/metabollites, perform QA & QC evaluations including batch-effect evaluation & correction. : 1) features are extracted using different parameters 2) results from each parameter setting are evaluated for sample quality & feature consistency, 3) filtered results from individual settings are merged to obtain a combined feature table; the optimization score that takes into account the number of features and average CV (see Uppal 2013) is used to determine the most optimal set of parameters. 4) A targeted feature table using the list of m/z in the "refMZ" file is generated 5) Quality measures of each featue includes: number of samples including replicates with non-missing values (NumPres.All.Samples), number of biological samples for which more than 60 median coefficient of variation (CV) within technical replicates summarized across all samples; Qscore (Quality score which is calculated using NumPres.All.Samples, NumPres.Biol.Samples, median CV, and delta ppm between mz.min and mz.max; Higher is better) Users have the option to filter poor quality samples and features based on correlation between technical replicates and feature reproducibility measures such as PID or CV, respectively.

Value

A list is returned.

`XCMS.merged.res`	Merged feature table, P1 U P2 where P1 and P2 are two sets of parameter settings. The "QRscore" is defined as the ratio of percentage of biological samples for which more than 50 percent of technical replicates have a signal and median PID or CV. QRscore=Percent good samples/median PID or CV
`XCMS.ind.res`	List with results from individual parameter settings.
`XCMS.ind.res.filtered`	List with results from individual parameter settings after filtering.
`annot.res`	List with annotation results after merging results.
`feat.eval.ind`	List with feature evaluation results based on CV or PID at each parameter setting.
`sample.eval.ind`	List with sample evaluation results at each parameter setting based on correlation between technical replicates.

Outputs XCMS results at each parameter settings to XCMS.outloc and the following to xMSanalyzer.outloc: -feature consistency results -sample evaluation results -feature list after merging results from parameter settings A and B -merge summary

Author(s)

Karan Uppal <kuppal2@emory.edu>