Package 'proFIA'

June 18, 2019

Type Package

Title Preprocessing of FIA-HRMS data

Version 1.10.0

Date 2016-06-06

Author Alexis Delabriere and Etienne Thevenot.

Maintainer Alexis Delabriere <alexis.delabriere@outlook.fr>

biocViews MassSpectrometry, Metabolomics, Lipidomics, Preprocessing, PeakDetection, Proteomics

Depends R (>= 2.5.0), xcms

Imports stats, graphics, utils, grDevices, methods, pracma, Biobase, minpack.lm, BiocParallel, missForest, ropls

Suggests BiocGenerics, plasFIA, knitr,

VignetteBuilder knitr

Description Flow Injection Analysis coupled to High-Resolution Mass Spectrometry is a promising approach for high-throughput metabolomics. FIA- HRMS data, however, cannot be pre-processed with current software tools which rely on liquid chromatography separation, or handle low resolution data only. Here we present the proFIA package, which implements a new methodology to pre-process FIA-HRMS raw data (netCDF, mzData, mzXML, and mzML) including noise modelling and injection peak reconstruction, and generate the peak table. The workflow includes noise modelling, band detection and filtering then signal matching and missing value imputation. The peak table can then be exported as a .tsv file for further analysis. Visualisations to assess the quality of the data and of the signal made are easely produced.

License CeCILL

NeedsCompilation yes

RoxygenNote 6.0.1

Collate 'Denoising.R' 'KNN_T.R' 'cWrapper.R' 'noiseEstimator.R' 'classContainer.R' 'fastMatchPpm.R' 'findPeaksFIA.R' 'methodsContainer.R' 'proFIA-package.R'

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

git_branch RELEASE_3_9

git_last_commit ea10171

git_last_commit_date 2019-05-02

Date/Publication 2019-06-17

R topics documented:

`proFIA-package` *Process FIA-HRMS datasets.*

Description

Process FIA-HRMS datasets passing from raw data (mzMl, CDF,mzXML format) to a peak table suitable for statistical anlysis.

Details

The full workflow is composed of the following chain of function. `proFIAset=>group.FIA=>impute.FIA` and the full process is easy to automate using the analyzeAcquisitionFIA which do all the steps easely. Resulting table may be easely exported using the 3 exports function (`exportDataMatrix`,`exportVariableMetada` Groups may be visualised using `plotEICs` to plot all the EICs of a group. An overview of an acquisition may be otained using the `plot` function.

`acquisitionDirectory` *Ge the class organization of directory.*

Description

Find the classes organization of a directory, and return a table. This function is called by proFIAset, and is useful to check the structure which will be considered by a proFIAset object.

Usage

```
acquisitionDirectory(files = NULL)
```

Arguments

`files`	The path to the experiment directory/

Value

a table containing two columns, the absolute paths of the files and the classes of the acquisition, as given by subdirectories.

Examples

```
if(require(plasFIA)){
    path<-system.file(package="plasFIA","mzML")
    tabClasses<-acquisitionDirectory(path)
    tabClasses
}
```

`analyzeAcquisitionFIA` *Wrapper function for the full FIA analysis workflow.*

Description

Perform the 4 steps pro fia workflow including :

- noise estimation. Noise is estimated.
- bands filtering. Bands are filtered using the `findFIASignal` function.
- peak grouping. Signals from different acquisition are grouped using `group.FIA` function.
- missing values imputations. Missing values are imputated using `impute.KNN_TN` function.

Minimal options to launch the workflow are provided, neithertheless if finer option tuning are necessary, launching the workflow function by function is strongly advised.

Usage

```
analyzeAcquisitionFIA(path, ppm, dmz = 0.001, fracGroup = 0.5,
  ppmGroup = NULL, dmzGroup = 5e-04, parallel = FALSE, bpparam = NULL,
  noiseEstimation = TRUE, SNT = NULL, maxo = FALSE,
  imputation = c("KNN_TN", "randomForest", "None"), ...)
```

Arguments

path	The path to the directory of acquistion.
ppm	The tolerance for deviations in m/z between scan in ppm findFIASignal
dmz	The minimum tolerance for deviations in m/z between scan in Dalton findFIASignal
fracGroup	The fraction of smaple form a class necessary to make a group.
ppmGroup	A ppm tolerance to group signals between samples group.FIA.
dmzGroup	The minimum tolerance in Dalton to group signals between samples group.FIA.
parallel	A boolean indicating if parallelism is supposed to be used.
bpparam	A BiocParallelParam object to be passed i BiocParallel is used.
noiseEstimation	A boolean indicating in noise need to be estimated.
SNT	A value giving the SNT thrshold, used only if noiseEstimation is FALSE.
maxo	Should the maximum intensity be used over the peak area.
imputation	The method to use for imputation 'randomForest' (impute.randomForest) or 'KNN_TN' (impute.KNN_TN). Put 'None' if no imputation should be done.
...	Supplementary arguments to be passed to the imputation method, see impute.FIA for detail.

Value

A filled proFIAset object ready for exportation.

Examples

```
if(require(plasFIA)){
    path<-system.file(package="plasFIA","mzML")

    #Defining parameters for Orbitrap fusion.
    ppm<-2
    dmz <- 0.0005
    ppm_group<-1
    dmz_group <- 0.0003
    paral<-FALSE
    frac_group<-0.2
    k<-2
    maxo<-FALSE
    vimputation <- "randomForest"
   ## Not run: plasSet<-analyzeAcquisitionFIA(path,ppm=ppm,dmz=dmz,imputation=vimputation,fracGroup=frac_gro
}
```

determiningInjectionZone *First guess of the limit of the injection peak.*

Description

Determine a first approximation of the injection peak based on the angle of the injection peak and changing of variation. This peak is not used directly by findFIAsignal, but will be used to initialize the regression giving the injection peak. Shloud be used carefuly.

Usage

```
determiningInjectionZone(xraw, threshold = 0.05, graphical = FALSE,
  scanmin = NULL, scanmax = NULL)
```

Arguments

xraw	An xcmsRaw object as returned by xcmsRaw. It may also be a TIC sequence as a list scan intensity.
threshold	A relative increase born to detect the limit of the injection peak.
graphical	should the resulting limit be plotted.
scanmin	The first scan to consider.
scanmax	The last scan to consider.

Value

A triplet composed of c(left limit,right limit, maximum) of the estimated injection peak.

Examples

```
if(require(plasFIA)){
  #Getting the path of a file.
  path_raw <- list.files(system.file(package="plasFIA","mzML"),full.names=TRUE)[2]

  #Opening the file with xcms
  xraw <- xcmsRaw(path_raw)

  #Getting a first approximation of injection peak;
  sp <- determiningInjectionZone(xraw)
}
```

`determiningSizePeak.Geom` *Determine the limits of the injection peak in a FIA acquisition.*

Description

Determine a first approximation of the injection peak using the Douglas-Peuker Algorithm provided in the `rgeos` package. The object provided must be an xcmsRaw object.

Usage

```
determiningSizePeak.Geom(xraw, scanmin = 1, scanmax = length(xraw@scantime),
  freq = 0.15, graphical = FALSE, smooth = TRUE, extended = FALSE,
  percentSol = NULL)
```

Arguments

`xraw`	An xcmsRaw object as returned by `xcmsRaw`.
`scanmin`	The minimum scan to consider for peak detection.
`scanmax`	the maximum scan to consider. injection peak detection.
`freq`	The degrees of smoothing used on the TIC, corresponding to the cutting frequency of the blackman windowed sync filter.
`graphical`	should the resulting peak be plotted.
`smooth`	Should the TIC be smoothed, recommended.
`extended`	In case of very long tailing, shloud the research be extended.
`percentSol`	If extended is TRUE, the limiting level of solvent for

Value

A triplet composed of c(left limit,right limit, maximum) of the estimated injection peak.

Examples

```
if(require(plasFIA)){
  #Getting the path of a file.
  path_raw <- list.files(system.file(package="plasFIA","mzML"),full.names=TRUE)[2]

  #Opening the file with xcms
  xraw <- xcmsRaw(path_raw)

  #Getting a first approximation of injection peak;
  sp <- determiningSizePeak.Geom(xraw)
}
```

`estimateNoiseMS` *Estimate the noise of a mass spectrometer using multiple MS acquisition.*

Description

Determine the variances of the noise in function of the intensity from multiples FIA acquisitions, using the method from Wentzell and Tarazuk(2014) *Characterization of heteroscedastic measurement noise in the absence of replicates*

Usage

```
estimateNoiseMS(list_files, ppm, nBin = 500, minInt = 500, maxInt = 10^8,
  parallel, includeZero = TRUE, dmz = 5e-04, BPPARAM = NULL)
```

Arguments

`list_files`	A list of files in which the noise should be estimated.
`ppm`	The authorized deviation between scans in ppm, this parameter will also be used to fuse the bands if there are close enough.
`nBin`	The number of intensity bins to be used.
`minInt`	The minimum intensity expected in all the files.
`maxInt`	The maximum intensity expected in all the files.
`parallel`	Shall parrallelism be used.
`includeZero`	Should the left bin start a 0.
`dmz`	The dmz parameters to be passed to findBandsFIA, see findBandsFIA
`BPPARAM`	A BiocParallelParam object to be used for parallelism if parallel is TRUE.

Value

A noise estimator object.

Examples

```
##Listing the files in plasFIA
if(require(plasFIA)){
  list_mzml <- list.files(system.file(package="plasFIA","mzML"),full.names=TRUE)

  ##For speed purpose
  list_mzml <- list_mzml[1:2]
  es <- estimateNoiseMS(list_mzml,2,parallel=FALSE)
}
```

---

exportDataMatrix,proFIAset-method

Export data matrix.

Description

Export the data matrix from a proFIAset object, to be used for statistical analysis.

Usage

```
## S4 method for signature 'proFIAset'
exportDataMatrix(object, filename = NULL)
```

Arguments

object	A proFIAset object.
filename	If not NULL the result will be written in filename as a tabular separated values file.

Value

A matrix with dimension samples x variables.

Examples

```
if(require(plasFIA)){
    data(plasSet)
    dm<-exportDataMatrix(plasSet)
    head(dm)
}
```

---

exportExpressionSet,proFIAset-method

Export proFIAset to ExpressionSet

Description

Eport the data from a proFIAset object as an ExpressionSet object from the Biobase package package.

Usage

```
## S4 method for signature 'proFIAset'
exportExpressionSet(object, colgroup = c("mzMed",
  "scanMin", "scanMax", "nPeaks", "corSampPeakMean",
  "signalOverSolventRatioMean", "timeShifted", "signalOverSolventPvalueMean"))
```

Arguments

object	A proFIAset object.
colgroup	Labels corresponding to the column names of the group table.

Value

An ExpressionSet object from the Biobase package

Examples

```
if(require("plasFIA")&require("Biobase")){
    data(plasSet)
    eset<-exportExpressionSet(plasSet)
    eset
}
```

---

exportPeakTable,proFIAset-method

Export proFIAset as a peak table.

Description

Export the data from a proFIAset object as a peak table which containes the values of measured for each variables for each samples and supplementary information.

Usage

```
## S4 method for signature 'proFIAset'
exportPeakTable(object, colgroup = c("mzMed",
  "corSampPeakMean", "meanSolvent", "signalOverSolventRatioMean", "timeShifted",
  "signalOverSolventPvalueMean"), mval = c("NA", "zero"), filename = NULL)
```

Arguments

object	A proFIAset object.
colgroup	Labels corresponding to the column names of the group table which will be added to the peak table.
mval	How will missing values be treated, in default they will be set to NA, or you can keep 0.
filename	The name of the file for the peak table to be exported.

Value

A dataframe containg the datasets.

Examples

```
if(require("plasFIA")){
    data(plasSet)

    #Creating the peak table
    ptable<-exportPeakTable(plasSet)
    head(ptable)

    #Directly in a file
    ## Not run: ptable<-exportPeakTable(plasSet,filename="peak_table.tsv")
}
```

`exportSampleMetadata,proFIAset-method`
Export samples metadata.

Description

Export the samples metadata of an experiment, to be used for statistical analysis.

Usage

```
## S4 method for signature 'proFIAset'
exportSampleMetadata(object, filename = NULL)
```

Arguments

`object`	A proFIAset object.
`filename`	If not NULL the result will be written in filename

Value

A dataframe with the following columns :

- sampleID an ID similar to the one of the peak table.
- class the group of the sample.

Examples

```
if(require(plasFIA)){
   data(plasSet)
   tsample<-exportSampleMetadata(plasSet)
   head(tsample)
}
```

`exportVariableMetadata,proFIAset-method`
Export variable metadata.

Description

Export the variable metadata of an experiment, to be used for statistical analysis.

Usage

```
## S4 method for signature 'proFIAset'
exportVariableMetadata(object, filename = NULL)
```

Arguments

`object`	A proFIAset object.
`filename`	If not NULL the result will be written in filename

Value

A dataframe with the following columns :

- variableID an ID similar to the one of the peak table.
- mzMed the median value of group in the m/z dimension.
- mzMin the minimum value of the group in the m/z dimension.
- mzMax the maximum value of the group in the m/z dimension.
- scanMin the first scan on which the signal is detected.
- scanMax the last scan on which the signal is detected.
- nPeaks The number of peaks grouped in a group.
- meanSolvent The mean of solvent in the acquisition.
- signalOverSolventPvalue The mean p-value of the group.
- corMean The mean of the matrix effect indicator.
- SigSolMean The mean of ratio of the signal max intensity on the solvent max intensity.

Examples

```
if(require(plasFIA)){
  data(plasSet)
  vtab<-exportVariableMetadata(plasSet)
  head(vtab)
}
```

`findBandsFIA` *Detect band in a FIA acquisition*

Description

Detect bands of points with similar mass in conscutive scans. Points may be moved if a better candidates is found.

Usage

```
findBandsFIA(xraw, firstScan = 1, lastScan = length(xraw@scantime),
  ppm = 2, sizeMin = NULL, dmz = 5e-04, beginning = NULL, end = NULL,
  nIso = 3, fracMin = 0.6)
```

Arguments

`xraw`	An xcmsRaw object as returned by `xcmsRaw`.
`firstScan`	The first scan to be considered, 1 for general use.
`lastScan`	The last scan to be considered.
`ppm`	The mass deviation in ppm for point in consecutives scans.
`sizeMin`	The minimum size of a band.
`dmz`	The minimum mass tolerance,useful for small masses
`beginning`	The scan of the injection. May be determined using `determiningInjectionZone`.
`end`	The end of injection, may be estimated using `determiningInjectionZone`.
`nIso`	The minimum number of consecutive points for a signal to be detected.
`fracMin`	The minimum fraction of points necessary in beginning:end for a signal to be detected. contaminated by solvent.

Value

A vector contaning the inject peak

Examples

```
#Getting the path of a file.
if(require(plasFIA)){
  path_raw<-list.files(system.file(package="plasFIA","mzML"),full.names=TRUE)[2]

  #Opening the file with xcms
  xraw<-xcmsRaw(path_raw)

  #Getting the injection scan
  gp<-determiningInjectionZone(xraw)

  #performing band detection.
  tbands<-findBandsFIA(xraw,ppm = 2,sizeMin = gp[3]-gp[1],beginning=gp[1],end=gp[2])
  head(tbands)
}
```

`findFIASignal` *Detect peaks in an FIA acquisition.*

Description

Detect the peak corresponding to compounds present in the sample in a Flow Injection Analysis (FIA) acquisition. The item provided must be an xcmsRaw object.

Usage

```
findFIASignal(xraw, ppm, es = NULL, solvar = c("throw", "keep"),
  solint = c("poly", "substract", "add"), fullInteg = FALSE, dmz = 0.001,
  graphical = FALSE, SNT = NULL, f = c("regression", "TIC"),
  pvalthresh = NULL, scanmin = 1, scanmax = length(xraw@scantime),
  bandCoverage = 0.3, shiftFactor = 0.3, sizeMin = NULL,
  bandlist = NULL, ...)
```

Arguments

`xraw`	An xcmsRaw object as returned by xcmsRaw.
`ppm`	The authorized deviation between scans in ppm, this parameter will also be used to fuse the bands if there are close enough.
`es`	A noise estimation object as returned by estimateNoiseMS, or NULL if the parameter noise if only an threshold is supposed to be used.
`solvar`	Should the signal corresponding to solvent be kept ? Only their maximum intensiyt will be calculated.
`solint`	How are the intensity of signal with bot solvent and sample be treated in the injection zone region, the area of the rectangle with peak-width and solvent intensity is considered.
`fullInteg`	If no solvent is detected before the injection, should the signals be integrated on the ufll chromatogram.

- poly. Half of this area is kept.
- substract. The area is removed substracted.
- remove. The area is conserved in the final value.

dmz	The minimum absolute value of the deviation between scans, to take into account the higher diviations at low masses.
graphical	A boolean indicating if the detected area shall be plotted.
SNT	NULL by default a relative intensity of signal/intensity of solvent threshold, used only if es is equal to NULL.
f	method to design the filter, "TIC" means that the peak of the TIC is used as a filter. "regression" means that the signal is regressed form the most intense band as an Exponential modified gaussian.
pvalthresh	The threshold used in p-value to discard signal, only used if a noise model is provided.
scanmin	The first scan to consider.
scanmax	The last scan to consider.
bandCoverage	A filter on the number of point found in a band. 0.3 by default to allow matrix effect.
sizeMin	The minimum number of point considered for a band to be considred for solvent filtration.
bandlist	An optional bandlist to be passed.
...	more arguments to be passed to the determiningInjectionZone function.

Value

A numeric matrix with the following column

- mzmin the minimum value of the mass traces in the m/z dimension.
- mzmax the maximum value of the mass traces in the m/z dimension.
- scanMin the first scan on which the signal is detected.
- scanMax the last scan on which the signal is detected.
- areaIntensity the integrated area of the signal.
- maxIntensity the maximum intensity of the signal.
- solventIntensity the intensity of the solvent, 0 means that no significant solvent was detected.
- corPeak An idicator of matrix effect, if it's close to 1, the compound does not suffer from heavy matrix effect, if it is inferior to 0.5, the compound suffer from heavy matrix effect.
- timeShifted An indicator of the shifting of th epeak in the time direction.
- signalOverSolventRatio The ratio of the signal max intensity on the solvent max intensity.

Examples

```
if(require(plasFIA)){
  #Getting the path of a file.
  path_raw<-list.files(system.file(package="plasFIA","mzML"),full.names=TRUE)[2]

  #Opening the file with xcms
  xraw<-xcmsRaw(path_raw)
```

```
  ppm<-2

  #getting the filtered signals without noise model which is not recommended.
  tsignal<-findFIASignal(xraw,ppm=ppm,SNT=3)

  #Getting the noise model un the plasSet object.
  data(plasSet)
  es<-attr(plasSet,"noiseEstimation")

  #Getting the signal with a noise model.
  tsignal<-findFIASignal(xraw,ppm=2,es=es,pvalthresh=0.005)
  head(tsignal)
}
```

`findMzGroup,proFIAset-method`

find a group in a FIA experiment.

Description

Find a group corresponding to the given mass in a proFIAset object, given a tolerance in ppm.The mz considered for a group is the median fo the grouped signals between the various acquisition.

Usage

```
## S4 method for signature 'proFIAset'
findMzGroup(object, mz, tol, dmz = 0.005,
  closest = TRUE)
```

Arguments

`object`	A proFIAset object.
`mz`	A numeric vector of masses to be looked for.
`tol`	The tolerance in ppm.
`dmz`	The minimum tolerance in absolute mz compared to the tolerance in ppm.
`closest`	Shall only the closest group be returned.

Value

A vector of integer of the same length than mz giving the row of the found group in the object group slot, or NA if the group is not found.

If closest is FALSE a list giving the index of the found group, if closest is TRUE a vector giving the position. NA indicates that the mass signal have not bene found.

See Also

You can visualize the group using `plotFlowgrams` function.

Examples

```
if(require("plasFIA")){
#proFIAset object is loaded
data(plasSet)

#The table of spiked molecule is loaded
data(plasMols)

#Mass to search and toolerance are defined
mass<-plasMols[22,"mass_M+H"]
tolppm <- 1

plasSet<- makeDataMatrix(plasSet)

index <- findMzGroup(plasSet,mass,tol=tolppm)

plasMols[22,]
#We extract the corresponding group.
groupMatrix(plasSet)[index,]
}
```

`getInjectionPeak` *Fit an injection peak to an FIA acquisition.*

Description

Determine an injection peak as an exponential modified gaussian function and a second order exponential corresponding to matrix effect to the most intense signals in an acquisition.

Usage

```
getInjectionPeak(xraw, bandlist, sec = 2, iquant = 0.95, gpeak = NULL,
  selIndex = NULL, scanmin = 1, scanmax = length(xraw@scantime),
  refinement = TRUE, graphical = FALSE)
```

Arguments

`xraw`	An xcmsRaw object as returned by `xcmsRaw`.
`bandlist`	A list of bands which can be used. Shall stay NULL in general use. bands will be determined automatically.
`sec`	A tolerance in sec to group the signals.
`iquant`	The maximum intensity intensity threshold under which the peaks would not be used for peak determination.
`gpeak`	An approximation of the injection peak, if NULL determining sizepeak.Geom will be used.
`selIndex`	A seleciton of index to make the regression. We recommend to let it to NULL.
`scanmin`	The first scan to consider for regression.
`scanmax`	The last scan to consider for regression.
`refinement`	Should the starting point of the regression be refined using the selected EICs.
`graphical`	shald the individually fitted components be plotted.

Value

A vector contaning the injection peak

Examples

```
if(require(plasFIA)){
  #Getting the path of a file.
  path_raw <- list.files(system.file(package="plasFIA","mzML"),full.names=TRUE)[2]

  #Opening the file with xcms
  xraw <- xcmsRaw(path_raw)

  #Getting the injection scan
  gp <- determiningInjectionZone(xraw)

  #performing band detection.
  tbands <- findBandsFIA(xraw,ppm = 2,sizeMin = gp[3]-gp[1],beginning=gp[1],end=gp[2])

  #Getting the injection peak
  vpeak <- getInjectionPeak(xraw,bandlist=tbands,gpeak=gp)
  plot(vpeak,type="l")
}
```

---

group.FIA,proFIAset-method

Group the peaks of an FIA acquisition.

Description

Group the peaks in a FIA experiemnts by clustering under an estimated density based on the accuracy in ppm of the mass spectrometer.

Usage

```
## S4 method for signature 'proFIAset'
group.FIA(object, ppmGroup, solvar = FALSE, sleep = 0,
  dmzGroup = 5e-04, fracGroup = 0.5)
```

Arguments

object	A proFIAset object.
ppmGroup	A ppm parameter giving the size of the windows considered, we recommend to use 0.5*ppm where ppm is the pp parameter of band detection in the proFIAset function.
solvar	Shall the group corresponding to solvent signal be kept. This in only useful if solvent signal have been conserved in the findFIASignal function.
sleep	If not 0 densities are plotted every sleep ms.
dmzGroup	A minimum mz deviation value which can be considered over the deviation in ppm if it is higher. This account for low mass deviation.
fracGroup	The minimum fraction of samples of a class required to make a group.

Value

A proFIAset object with the group slot filled. See proFIAset-class.

Examples

```
if(require("plasFIA")){
#proFIAset object is loaded
data(plasSet)

#Parameters are defined.
ppm_group <- 1
dmz_group <- 0.0005
frac_group<-0.2

plasSet<-group.FIA(plasSet,ppmGroup=ppm_group,fracGroup=frac_group,dmzGroup=dmz_group)
plasSet
}
```

---

impute.FIA,proFIAset-method

Fill missing values using the provided method.

Description

Impute the missing values in an FIA experiment using a Weighted K-Nearest Neighbours on Truncated Distribution implemented in impute.KNN_TN or by random forest using the impute.randomForest function.

Usage

```
## S4 method for signature 'proFIAset'
impute.FIA(object, method = c("KNN_TN", "randomForest"),
  ...)
```

Arguments

object	A proFIAset object.
method	The method to be used for missing value imputation.
...	Arguments furnished to the imputation method. No argument is required for impute.randomForest and the number of neighbours should be furnished using the impute.KNN_TN function.

Examples

```
if(require(plasFIA)){
    data(plasSet)

    ###Reinitializing the data matrix an using KNN
    plasSet<-makeDataMatrix(plasSet,maxo=FALSE)
    plasSet<-impute.FIA(plasSet,method="KNN_TN",k=2)

    ###Reinitializing the data matrix and using randomForest
```

```
    plasSet<-makeDataMatrix(plasSet,maxo=FALSE)
    plasSet<-impute.FIA(plasSet,method="randomForest")
}
```

`impute.KNN_TN,proFIAset-method`

Fill missing values in the peak table using K-nearest Neighbour for tuncated distributions.

Description

Impute the missing values in an FIA experiment using a Weighted K-Nearest Neighbours on Truncated Distribution described by Jasmit S. Shah et al.

Usage

```
## S4 method for signature 'proFIAset'
impute.KNN_TN(object, k = 0.6, classes = c("split",
  "unique"))
```

Arguments

`object`	A proFIAset object.
`k`	The number of neighbors considered, can be a fraction then in this case the k will be taken for each class as the frac of the effective of the class. 3 at minima because comparison is based on correlation.
`classes`	how to handle imputation for different classes, if 'split', the classes are taken separately, if 'unique', the imputation is done on the full data matrix.

Value

A proFIAset object with the missing values imputated.

References

Distribution based nearest neighbor imputation for truncated high dimensional data with applications to pre-clinical and clinical metabolomics studies, J.S Shah 2017, BMC Bioinformatics.

Examples

```
if(require(plasFIA)){
    data(plasSet)

    ###Reinitializing the data matrix
    plasSet<-makeDataMatrix(plasSet,maxo=FALSE)
    plasSet<-impute.KNN_TN(plasSet,2)
}
```

impute.randomForest,proFIAset-method

Fill missing values in the peak table using random forest.

Description

Impute the missing values in an FIA experiment using a random forest implemented in the missForest package.

Usage

```
## S4 method for signature 'proFIAset'
impute.randomForest(object, parallel = FALSE, ...)
```

Arguments

object	A proFIAset object.
parallel	Shall parallelism be used.
...	supplementary arguements to be passed to missForest function.

Value

A proFIAset object with the missing values imputated.

References

Stekhoven, D.J. and Buehlmann, P. (2012), 'MissForest - nonparametric missing value imputation for mixed-type data', Bioinformatics, 28(1) 2012, 112-118, doi: 10.1093/bioinformatics/btr597

Examples

```
if(require(plasFIA)){
    data(plasSet)
    ###Reinitializing the data matrix
    plasSet<-makeDataMatrix(plasSet,maxo=FALSE)
    plasSet<-impute.randomForest(plasSet)
}
```

makeDataMatrix,proFIAset-method

Construct the data matrix of a proFIAset object.

Description

Construct the data matrix of a proFIA set object, using the selected measure for the intensity, area or maximum intensity. The choice of area or maximum intensity depends of your acquisition, and your preferences.

Usage

```
## S4 method for signature 'proFIAset'
makeDataMatrix(object, maxo = FALSE)
```

Arguments

object	A proFIAset object.
maxo	Shall the intensity used to the area or the maximum intensity

Value

A proFIAset object with the dataMatrix slot filled.

See Also

To obtain this data matrix see proFIAset.

Examples

```
if(require("plasFIA")){
#proFIAset object is loaded
data(plasSet)

plasSet<-makeDataMatrix(plasSet)
plasSet
}
```

---

noiseEstimation-class *An S4 class to represent heteroscedasctic noise of MS.*

Description

An S4 class to represent heteroscedasctic noise of MS.

Slots

bins The limit of the bins on which the noise have been estimated.

varmean The estimate of the mean of the variance in each bin.

size The size of each bins in number of elements.

estimation The estimation function, if a model have been fitted.

filelist The list of the fitted file.

estimated A boolean indicating if a model have been fitted.

frac A numeric giving the fraction of the point ot be conserved when estimating the noise model.

intlim The maximum and minmum intensity on which the estimationis done.

reglim The limits of the regression of the model on these data. detected for each experiment.

peaksGroup,proFIAset-method

Return the peaks corresponding to a group.

Description

Return the peaks corresponding ot a group given by his index.

Usage

```
## S4 method for signature 'proFIAset'
peaksGroup(object, index = NULL)
```

Arguments

object	A proFIAset object.
index	A numeric vector r giving the group to be returned. NA are ignored.

Value

The peaks in the given group, see proFIAset-class.

Examples

```
if(require(plasFIA)){
    data(plasSet)
    data(plasMols)

    #finding the molecules of plasMols
    vmatch<-findMzGroup(plasSet,mz=plasMols[,"mass_M+H"],tol=5)

    mol_peaks<-peaksGroup(plasSet,index=vmatch)
    head(mol_peaks)
}
```

plot,proFIAset,ANY-method

Plot a summary of an FIA experiment.

Description

Plot a summary of an FIA acquisition. This summary aims to provides an overview of the FIA acquisition and the processinf of FIA acquisition. It includes the following graphs :

- Barplot A barplot giving the number of peak detected in each sample. The number of shifted peak is indicated, which can indicate wrong FIA-acquisition, as well as the number of peak badly with a low correlation with the injection peak, which can indicate a strong matrix effect.
- Injection_Peaks A representation of all the injection peaks of the acquisition, a greatly different peak in th einjection may indicate an issue with the injection, or a problem of regression. If the peaks seems all different, try to use proFIAset with the f paramter on TIC, to avoid regression.

- Density A density plot of the m/z of all the features found. A missing interval at the end of the mz range may indicate a too low ppm parameter, while a missing interval at the beginning of the mz range may indicate a too low dmz parameter.
- PCA A PCA plot to obtain a quick diagnostic of the data. The PCA plot is supposed to be different before and after imputation.

Usage

```
## S4 method for signature 'proFIAset,ANY'
plot(x, type = c("sample", "class"), ...)
```

Arguments

x	A proFIAset object.
type	Shall the plotting be done by sample or by class for the barplot ?
...	Not used at the moment.

Value

Nothing

Examples

```
if(require("plasFIA")){
   data(plasSet)


   ####Diagnostic plot after imputation
   plot(plasSet)

   ####The same plot by classes.
   plot(plasSet,type="class")

   ####Diagnostic plot before imputation
   plasSet <- makeDataMatrix(plasSet)
  plot(plasSet)

}
```

---

plotFlowgrams,proFIAset-method

Plot raw temporal profiles of the selected group.

Description

Plot raw temporal profiles from proFIAset object corresponding to one or more molecules. The function will priorize index, only using mz if index is set to NULL. As this require to come back to the raw data, this can take some times, an we don't recommend using it on more than 30 files. It is better to choose 30 files in the proFIAset object.

Usage

```
## S4 method for signature 'proFIAset'
plotFlowgrams(object, index = NULL, mz = NULL,
  subsample = NULL, ppm = 5, margin = 2e-04, posleg = c("topright",
  "bottomright", "bottom", "bottomleft", "left", "topleft", "top", "right",
  "center"), title = NULL, scaled = FALSE, area = FALSE, ...)
```

Arguments

object	A proFIAset object.
index	The index of the group to be plotted.
mz	An mz value to be looked for only used if index is null. the research use the findMzGroup function.
subsample	A subset of sample to be plotted.
ppm	The tolerance for the research if mz is provided.
margin	An area outer the EICs mz range on which the EIC may be extended.
posleg	The position of the legend on the figure. See legend.
title	An optional vector of title for the plot. Need to be of the same
scaled	Shall all the EIC be put on the same scale with maximum to 1.
area	Shall the detectged area be plotted using transparency.
...	Supplementary graphical parameters to be passed to lines. length than index.

Value

No returned value

Examples

```
if(require(plasFIA)){
    data(plasMols)
    data(plasSet)
    plotFlowgrams(plasSet,mz=plasMols[7,"mass_M+H"])
}
```

---

plotNoise *Plot the estimated noise from a proFIAset object.*

Description

Plot an intensity vs variances plot for the noise estimated from a MS acquisition. If a model is fitted to the data it will be plotted. Only the interval used for the regression and on which the estimation will be used is shown.

Usage

```
plotNoise(object, xlim = NULL, ylim = NULL, ...)
```

Arguments

object	A noise estimation object or a proFIAset-class object.
xlim	The xlim paramter to be passed to plot.
ylim	The ylim paramter to be passed to plot.
...	SUpplementary arguments to be passed to plot
plotNoise	plotNoise,proFIAset-method

Value

The plotted value as an x y list.

Examples

```
if(require(plasFIA)){
  data(plasSet)
  plotNoise(plasSet)
}
```

---

plotRaw,proFIAset-method

Plotting of raw data

Description

Plot the raw data from a proFIAset object,the the type of plot determines if the full raw data needs to be plotted or only the data conressponding to the detected peaks needs to be plottes. The path in the classes table of the proFIAset object needs to be correct.

Usage

```
## S4 method for signature 'proFIAset'
plotRaw(object, type = c("raw", "peaks"),
  sample = NULL, bandl = TRUE, legend = TRUE, ...)
```

Arguments

object	A proFIAset object.
type	"raw" indicate that raw data needs to be plotted and "peak" indicate that only the filtered signals will be plotted.
sample	The number of the sample in the object classes table to be plotted. The classes table can be taken using the phenoClasses function.
bandl	A boolean shall a line be added on the detected bands. Only used in type is set to "peak"
legend	Shall the legend be plotted along the graph
...	xlim,ylim and size to be passed to plot functions.

Value

No value is returned.

Examples

```
if(require("plasFIA")){
    data(plasSet)

    #Visualising the raw data
    plotRaw(plasSet,type="raw",ylim=c(215.9,216.2),sample=4)

    #Plotting the filtered signals only.
    plotRaw(plasSet,type="peaks",ylim=c(215.9,216.2),sample=4)
}
```

---

plotSamplePeaks,proFIAset-method

Plot the sample peak of a proFIAset object.

Description

Plot the sample peaks determined by regression for each sample. If you want to check the various regression performed by *proFIA* in an individual sample we recommend you to use the `getInjectionPeak`. A sample peak highly different form the other may indicate a problem of the regression process, or an injection issue in the acquisition.

Usage

```
## S4 method for signature 'proFIAset'
plotSamplePeaks(object, subsample = NULL,
  diagPlotL = FALSE, ...)
```

Arguments

`object`	A proFIAset object.
`subsample`	The subset of sample on which the sample may be plotted. If it is numeric it will be viewed as sample row in the classes table, if it is a character it will be viewed as a factor.
`diagPlotL`	Boolean used by the usmmary plot function. Keep to FALSE in the general case.
`...`	SUpplementary arguments which will be passed to the lines function.

Value

No value returned.

Examples

```
if(require(plasFIA)){
    data(plasSet)
    plotSamplePeaks(plasSet)
}
```

proFIAset *Process FIA experiment.*

Description

Processes an experiment ordered as a tree of files, and return a proFIAset object. TO check the funirshed structure you may use the acquisitionDirectory function.

Usage

```
proFIAset(path, ppm, parallel = TRUE, BPPARAM = NULL,
  noiseEstimation = TRUE, graphical = FALSE, ...)
```

Arguments

path	The path of the files to be processed.
ppm	The tolerance of the algorithms in deviation between scans.
parallel	Shall parallelism using **BiocParallel** be used.
BPPARAM	A BiocParallelParam object to be used for parallelism if parallel is TRUE.
noiseEstimation	Shall noise be estimated (recommended)
graphical	Shall the plot of the regressed injection peak be shown.
...	Supplementary arguments to be passed to findFIAsignal, see findFIASignal.

Value

A proFIAset object.

See Also

To obtain more detail about the output see proFIAset-class.

Examples

```
if(require("plasFIA")){

pathplas<-system.file(package="plasFIA","mzML")

#Parameters are defined.
ppm<-2

plasSet<-proFIAset(pathplas,ppm=ppm,parallel=FALSE)
plasSet

###An example using parallelism with a snow cluster using BiocParallel package.
## Not run: plasSet<-proFIAset(pathplas,ppm=ppm,parallel=TRUE,BPPARAM=bpparam("SnowParam"))
}
```

proFIAset-class *An S4 class to represent an FIA experiments.*

Description

The S4 class also includes all the informations about processing, and the detected signals are stored.

Usage

```
## S4 method for signature 'proFIAset'
phenoClasses(object)

## S4 method for signature 'proFIAset'
dataMatrix(object)

## S4 method for signature 'proFIAset'
groupMatrix(object)

## S4 method for signature 'proFIAset'
peaks(object)

## S4 method for signature 'proFIAset'
injectionPeaks(object)
```

Arguments

object	A proFIAset object.

Methods (by generic)

- phenoClasses: Extract the classes and the paths of the samples.
- dataMatrix: Extract the dataMatrix containg variables as rows and samples as columns
- groupMatrix: Extract the matrix of group, see exportPeakTable for better output.
- peaks: Extract all the signals detected in individual samples.
- injectionPeaks: Extract all the regressed injection peaks.

Slots

peaks A matrix containg all the peaks which have been detected in each individual files.

- mzmin the minimum value of the mass traces in the m/z dimension.
- mzmax the maximum value of the mass traces in the m/z dimension.
- scanMin the first scan on which the signal is detected.
- scanMax the last scan on which the signal is detected.
- areaIntensity the integrated area of the signal.
- maxIntensity the maximum intensity of the signal.
- solventIntensity the intensity of the solvent, 0 means that no significant solvent was detected.

- corSampPeak An idicator of matrix effect, if it's close to 1, the compound does not suffer from heavy matrix effect, if it is inferior to 0.5, the compound suffer from heavy matrix effect.
- signalOverSolventRatio The ratio of the signal max intensity on the oslvent max intensity.

`group` A matrix containing the information on the groups done between all the acquisitions.

- mzMed the median value of group in the m/z dimension.
- mzMin the minimum value of the group in the m/z dimension.
- mzMax the maximum value of the group in the m/z dimension.
- scanMin the first scan on which the signal is detected.
- scanMax the last scan on which the signal is detected.
- nPeaks The number of peaks grouped in a group.
- meanSolvent The mean of solvent in the acquisition.
- pvalueMedian The median p-value of the group.
- corMean The mean of the matrix effect indicator.
- signalOverSolventMean The mean of ratio of the signal max intensity on the oslvent max intensity.
- corSd The standard deviation of the matrix effect indicator.
- timeShifted Is the peak shifted compared to the injection peak.

`groupidx` The row of the peaks corresponding to each group in `peaks`.

`step` The step of processing of the experiment.

`path` The path of the experiment.

`classes` A table with two columns, "rname" the absolute path of a file, and "group" the class to which the file belong.

`dataMatrix` A matrix variables x experiments suitable for statistical analysis.

`noiseEstimation` A model of noise as estimated by estimateNoiseMS

`injectionPeaks` A list of the injection peak which have been detected for each experiment.

`injectionScan` A numeric vector giving the scan of injection of sample.

Index