MSnbase

October 25, 2011

`MIAPE-class` *The "MIAPE" Class for Storing Proteomics Experiment Information*

Description

The Minimum Information About a Proteomics Experiment. The current implementation is based on the MIAPE-MS 2.4 document.

Slots

`title`: Object of class `character` containing a single-sentence experiment title.

`abstract`: Object of class `character` containing an abstract describing the experiment.

`url`: Object of class `character` containing a URL for the experiment.

`pubMedIds`: Object of class `character` listing strings of PubMed identifiers of papers relevant to the dataset.

`samples`: Object of class `list` containing information about the samples.

`preprocessing`: Object of class `list` containing information about the pre-processing steps used on the raw data from this experiment.

`other`: Object of class `list` containing other information for which none of the above slots does not applies.

`dateStamp`: Object of class `character`, giving yhe date on which the work described was initiated; given in the standard 'YYYY-MM-DD' format (with hyphens).

`name`: Object of class `character` containing the name of the (stable) primary contact person for this data set; this could be the experimenter, lab head, line manager, ...

`lab`: Object of class `character` containing the laboratory where the experiment was conducted.

`contact`: Object of class `character` containing contact information for lab and/or experimenter.

`instrumentModel`: Object of class `character` indicating the model of the mass spectrometer used to generate the data.

`instrumentManufacturer`: Object of class `character` indicating the manufacturing company of the mass spectrometer.

`instrumentCustomisations`: Object of class `character` describing any significant (i.e. affecting behaviour) deviations from the manufacturer's specification for the mass spectrometer.

`softwareName`: Object of class `character` with the instrument management and data analysis package(s) name(s).

`softwareVersion`: Object of class `character` with the instrument management and data analysis package(s) version(s).

`switchingCriteria`: Object of class `character` describing the list of conditions that cause the switch from survey or zoom mode (MS1) to or tandem mode (MSn where n > 1); e.g. 'parent ion" mass lists, neutral loss criteria and so on [applied for tandem MS only].

`isolationWidth`: Object of class `numeric` describing, for tandem instruments, the total width (i.e. not half for plus-or-minus) of the gate applied around a selected precursor ion m/z, provided for all levels or by MS level.

`parameterFile`: Object of class `character` giving the location and name under which the mass spectrometer's parameter settings file for the run is stored, if available. Ideally this should be a URI+filename, or most preferably an LSID, where feasible.

`ionSource`: Object of class `character` describing the ion source (ESI, MALDI, ...).

`ionSourceDetails`: Object of class `character` describing the relevant details about the ion source. See MIAPE-MI docuement for more details.

`analyser`: Object of class `character` describing the analyser type (Quadrupole, time-of-flight, ion trap, ...).

`analyserDetails`: Object of class `character` describing the relevant details about the analyser. See MIAPE-MI docuement for more details.

`collisionGas`: Object of class `character` describing the composition of the gas used to fragment ions in the collision cell.

`collisionPressure`: Object of class `numeric` providing the pressure (in bars) of the collision gas.

`collisionEnergy`: Object of class `character` specifying for the process of imparting a particular impetus to ions with a given m/z value, as they travel into the collision cell for fragmentation. This could be a global figure (e.g. for tandem TOF's), or a complex function; for example a gradient (stepped or continuous) of m/z values (for quads) or activation frequencies (for traps) with associated collision energies (given in eV). Note that collision energies are also provided for individual "`Spectrum2`" instances, and is the preferred way of accessing this data.

`detectorType`: Object of class `character` describing the type of detector used in the machine (microchannel plate, channeltron, ...).

`detectorSensitivity`: Object of class `character` giving and appropriate measure of the sensitivity of the described detector (e.g. applied voltage).

Methods

The following methods as in "`MIAME`":

`abstract(MIAPE)`: An accessor function for `abstract`.

`expinfo(MIAPE)`: An accessor function for `name`, `lab`, `contact`, `title`, and `url`.

`notes(MIAPE), notes(MIAPE) <- value`: Accessor functions for `other`. `notes(MIAME) <- character` *appends* character to notes; use `notes(MIAPE) <- list` to replace the notes entirely.

`otherInfo(MIAPE)`: An accessor function for `other`.

`preproc(MIAPE)`: An accessor function for `preprocessing`.

`pubMedIds(MIAPE), pubMedIds(MIAME) <- value`: Accessor function for `pubMedIds`.

`samples(MIAPE)`: An accessor function for `samples`.

MIAPE-specific methods, including MIAPE-MS meta-data:

`show(MIAPE)`: Displays the experiment data.

`msInfo(MIAPE)`: Displays 'MIAPE-MS' information.

Extends

Class "MIAxE", directly. Class "Versioned", by class "MIAxE", distance 2.

Author(s)

Laurent Gatto <lg390@cam.ac.uk>

References

About MIAPE: http://www.psidev.info/index.php?q=node/91, and references therein, especially 'Guidelines for reporting the use of mass spectrometry in proteomics', Nature Biotechnology 26, 860-861 (2008).

MSnExp-class *The 'MSnExp' Class for MS Data And Meta-Data*

Description

The `MSnExp` class encapsulates data and meta-data for mass spectrometry experiments, as described in the `slots` section. Several data files (currently in `mzXML`) can be loaded together with the function `readMzXMLData`.

This class extends the virtual "pSet" class.

Objects from the Class

Objects can be created by calls of the form `new("MSnExp",...)`. However, it is preferred to use the `readMzXMLData` function that will read raw mass spectrometry data to generate a valid "MSnExp" instance.

Slots

`assayData`: Object of class "environment" containing the MS spectra (see "Spectrum1" and "Spectrum2"). Slot is inherited from "linkS4class{pSet}".

`phenoData`: Object of class "AnnotatedDataFrame" containing experimenter-supplied variables describing sample (i.e the individual tags for an labelled MS experiment) See `phenoData` for more details. Slot is inherited from "linkS4class{pSet}".

`featureData`: Object of class "AnnotatedDataFrame" containing variables describing features (spectra in our case), e.g. identificaiton data, peptide sequence, identification score,... (inherited from "eSet"). See `featureData` for more details. Slot is inherited from "linkS4class{pSet}".

`experimentData`: Object of class "MIAPE", containing details of experimental methods. See `experimentData` for more details. Slot is inherited from "linkS4class{pSet}".

protocolData: Object of class "AnnotatedDataFrame" containing equipment-generated variables (inherited from "eSet"). See protocolData for more details. Slot is inherited from "linkS4class{pSet}".

processingData: Object of class "MSnProcess" that records all processing. Slot is inherited from "linkS4class{pSet}".

.__classVersion__: Object of class "Versions" describing the versions of R, "linkS4class{Biobase}" "pSet" and MSnExp of the current instance. Slot is inherited from "linkS4class{pSet}". Intended for developer use and debugging (inherited from "eSet").

Extends

Class "pSet", directly. Class "VersionedBiobase", by class "pSet", distance 2. Class "Versioned", by class "pSet", distance 3.

Methods

See the "linkS4class{pSet}" class for documentation on accessors inherited from pSet, subsetting and general attribute accession.

clean signature(object = "MSnExp"): Removes unused 0 intensity data points. See clean documentation for more details and examples.

extractPrecSpectra signature(object = "MSnExp", prec = "numeric"): extracts spectra with precursor MZ value equal to prec and returns an object of class 'MSnExp'. See extractPrecSpectra documentation for more details and examples.

extractSpectra signature(object = "MSnExp", selected = "logical"): extracts spectra corresponding to 'TRUE' in selected. See extractSpectra documentation for more details and examples.

plot signature(x = "MSnExp", y = "missing"): Plots all the spectra of the MSnExp instance. See plot.MSnExp documentation for more details.

plot2d signature(object = "MSnExp", z = "character"): Plots retention time against precursor MZ for MSnExp instance. See plot2d documentation for more details.

quantify signature(object = "MSnExp"): Performs quantification for all the MS2 spectra of the MSnExp instance. See quantify documentation for more details.

removePeaks signature(object = "MSnExp"): Removes peaks lower that a threshold t. See removePeaks documentation for more details and examples.

show signature(object = "MSnExp"): Displays object content as text.

trimMz signature(object = "MSnExp"): Trims the MZ range of all the spectra of the MSnExp instance. See trimMz documentation for more details and examples.

Author(s)

Laurent Gatto <lg390@cam.ac.uk>

References

Information about the mzXML format as well converters from vendor specific formats to mzXML: http://tools.proteomecenter.org/wiki/index.php?title=Formats:mzXML.

See Also

"pSet" and readMzXMLData for loading mzXML data to generate an instance of MSnExp.

Examples

```
mzxmlfile <- dir(system.file("extdata",package="MSnbase"),
                 pattern="mzXML",full.names=TRUE)
msnexp <- readMzXMLData(mzxmlfile)
msnexp
```

MSnProcess-class *The "MSnProcess" Class*

Description

MSnProcess is a container for MSnExp and MSnSet processing information. It records data files, processing steps, thresholds, analysis methods and times that have been applied to MSnExp or MSnSet instances.

Slots

files: Object of class "character" storing the raw data files used in experiment described by the "MSnProcess" instance.

processing: Object of class "character" storing all the processing steps and times.

merged: Object of class "logical" indicating whether spectra have been merged.

cleaned: Object of class "logical" indicating whether spectra have been cleaned. See clean for more details and examples.

removedPeaks: Object of class "character" describing whether peaks have been removed and which threshold was used. See removePeaks for more details and examples.

smoothed: Object of class "logical" indicating whether spectra have been smoothed.

centroided: Object of class "logical" indicating whether spectra have been centroided.

trimmed: Object of class "numeric" documenting if/how the data has been trimmed.

normalised: Object of class "logical" describing whether and how data have been normalised.

MSnbaseVersion: Object of class "character" indicating the version of MSnbase.

.__classVersion__: Object of class "Versions" indicating the version of the MSnProcess instance. Intended for developer use and debugging.

Extends

Class "Versioned", directly.

Methods

fileNames signature(object = "MSnExp"): Returns the file names used in experiment described by the "MSnProcess" instance.

show signature(object = "MSnExp"): Displays object content as text.

Note

This class is likely to be updated using an AnnotatedDataFrame.

Author(s)

Laurent Gatto <lg390@cam.ac.uk>

See Also

See the "MSnExp" and "MSnSet" classes that actually use MSnProcess as a slot.

Examples

```
showClass("MSnProcess")
```

MSnSet-class *The "MSnSet" Class for MS Proteomics Expression Data and Meta-Data*

Description

The MSnSet holds quantified expression data for MS proteomics data and the experimental meta-data. The MSnSet class is derived from the "eSet" class and mimics the "ExpressionSet" class classically used for microarray data.

Objects from the Class

Objects can be created by calls of the form new("MSnSet", exprs, ...). See also "ExpressionSet" for helpful information. Expression data produced from other softwares can thus make use of this standardized data container to benefit R and Bioconductor packages. Importer functions will be developed to stream-line the generation of "MSnSet" instances from third-party software.

In the frame of the MSnbase package, MSnSet instances are generated from "MSnExp" experiments when the "ReporterIons" using the "quantify" method).

Slots

qual: Object of class "data.frame" that records peaks data for each of the reporter ions to be used as quality metrics.

processingData: Object of class "MSnProcess" that records all processing.

assayData: Object of class "assayData" containing a matrix with equal with column number equal to nrow(phenoData). assayData must contain a matrix exprs with rows represening features (e.g., reporters ions) and columns representing samples. See the "AssayData" class, exprs and assayData accessor for more details. This slot in indirectly inherited from "eSet".

phenoData: Object of class "AnnotatedDataFrame" containing experimenter-supplied variables describing sample (i.e the individual tags for an labelled MS experiment) (indireclty inherited from "eSet"). See phenoData and the "eSet" class for more details.

featureData: Object of class "AnnotatedDataFrame" containing variables describing features (spectra in our case), e.g. identificaiton data, peptide sequence, identification score,... (inherited indirectly from "eSet"). See featureData and the "eSet" class for more details.

experimentData: Object of class "MIAPE", containing details of experimental methods (inherited from "eSet"). See experimentData and the "eSet" class for more details.

`annotation`**:** not used here.

`protocolData`**:** Object of class "`AnnotatedDataFrame`" containing equipment-generated variables (inherited indirectly from "`eSet`"). See `protocolData` and the "`eSet`" class for more details.

`.__classVersion__`**:** Object of class "`Versions`" describing the versions of R, "`linkS4class{Biobase}`" "`eSet`", "`eSet`" and `MSnSet` of the current instance. Intended for developer use and debugging (inherited from indirectly "`eSet`").

Extends

Class "`eSet`", directly. Class "`VersionedBiobase`", by class "eSet", distance 2. Class "`Versioned`", by class "eSet", distance 3.

Methods

MSnSet specific methods or over-ridding it's super-class are described below. See also more "`eSet`" for inherited methods.

dim `signature(x = "MSnSet")`: Returns the dimensions of object's assay data, i.e the number of samples and the number of features.

fileNames `signature(object = "MSnSet")`: Access file names in the `processingData` slot.

msInfo `signature(object = "MSnSet")`: Prints the MIAPE-MS meta-data stored in the `experimentData` slot.

processingData `signature(object = "MSnSet")`: Access the `processingData` slot.

show `signature(object = "MSnSet")`: Displays object content as text.

qual `signature(object = "MSnSet")`: Access the reporter ion peaks description.

Author(s)

Laurent Gatto <lg390@cam.ac.uk>

See Also

"`eSet`" and "`ExpressionSet`".

Examples

```
mzxmlfile <- dir(system.file("extdata",package="MSnbase"),
                 pattern="mzXML",full.names=TRUE)
msnexp <- readMzXMLData(mzxmlfile,verbose=FALSE)
data(iTRAQ4)
msnset <- quantify(msnexp,method="trap",reporters=iTRAQ4,verbose=FALSE)
msnset
```

MSnbase-package *MSnbase: Base Functions and Classes for MS-based Proteomics*

Description

MSnbase provides classes, methods and functions for visualisation, manipulation and processing of mass spectrometry data.

Important class are "MSnExp" (raw data file), "MSnSet" (quantitation data) and "ReporterIons" (reporter ions for labelled proteomics).

Other classes are "Spectrum" and the subclasses "Spectrum1" (for MS spectra) and "Spectrum2" (for MSMS spectra), "MIAPE" (Minimum Information about Proteomics Experiments) and "MSnProcess" (for processing information). These should however not be of direct utility to users.

Author(s)

Laurent Gatto

Maintainer: Laurent Gatto <lg390@cam.ac.uk>

References

Gatto L. and Lilley K.S., Towards reproducible MSMS data preprocessing, quality control and quantification. BSPR/EBI Proteomics Meeting, Hinxton, United Kingdom, 13-15 July 2010, http://dx.doi.org/10.1038/npre.2010.5010.1.

See Also

Introductory information, use cases and details are available from the vignettes:

- The demo vignette describe an use-case using a dummy data set provided with the package. It can be accessed with `vignette("MSnbase-demo",package="MSnbase")`.
- The development vignette describes the classes implemented in `MSnbase` and can be accessed with `vignette("MSnbase-development",package="MSnbase")`.

Complete listing of available documentation with `library(help="MSnbase")`.

NAnnotatedDataFrame-class *Class Containing Measured Variables and Their Meta-Data Description for*

Description

An `NAnnotatedDataFrame` is an "AnnotatedDataFrame", as defined in the 'Biobase' package that includes additional labels for multiplexing annotation.

Objects from the Class

See "AnnotatedDataFrame" for object creation with `new`. Multiplexing data is defined by setting the `multiplex` and `multiLables` paramters.

Slots

`multiplex`: Object of class `"numeric"` indicating the number of multiplexed samples described.

`multiLabels`: Object of class `"character"` describing the multiplexing.

`varMetadata`: Object of class `"data.frame"` with number of rows equal number of columns in `data`, and at least one column, named `labelDescription`, containing a textual description of each variable. Inherited from `"AnnotatedDataFrame"`.

`data`: Object of class `"data.frame"` containing samples (rows) and measured variables (columns). Inherited from `"AnnotatedDataFrame"`.

`dimLabels`: Object of class `"character"` of length 2 that provides labels for the rows and columns in the `show` method. Inherited from `"AnnotatedDataFrame"`.

`.__classVersion__`: Object of class `"Versions"` describing the instance version. Intended for developer use. Inherited from `"AnnotatedDataFrame"`.

Extends

Class `"AnnotatedDataFrame"`, directly. Class `"Versioned"`, by class "AnnotatedDataFrame", distance 2.

Methods

dim `signature(object = "NAnnotatedDataFrame")`: Returns the number of samples, variables and multiplex cardinality in the object.

multiplex `signature(object = "NAnnotatedDataFrame")`: Returns the number of multipexed samples described by the object.

multiLabels `signature(object = "NAnnotatedDataFrame")`: Returns the multiplex labels.

show `signature(object = "NAnnotatedDataFrame")`: Textual description of the object.

Author(s)

Laurent Gatto <lg390@cam.ac.uk>

See Also

`"AnnotatedDataFrame"`.

Examples

```
df <- data.frame(x=1:3,
                 y=LETTERS[1:3],
                 row.names=paste("Sample",1:3,sep=""))
metaData <-
  data.frame(labelDescription=c(
                "Numbers",
                "Factor levels"))
mplx <- c("M1","M2")
new("NAnnotatedDataFrame",
    data=df,
    varMetadata=metaData,
    multiplex=length(mplx),
    multiLabels=mplx)
```

ReporterIons-class *The "ReporterIons" Class*

Description

The ReporterIons class allows to define a set of isobaric reporter ions that are used for quantification in MSMS mode, e.g. iTRAQ (isobaric tag for relative and absolute quantitation) or TMT (tandem mass tags). ReporterIons instances can them be used when quantifying "MSnExp" data of plotting the reporters peaks based on in "Spectrum2" ojects.

Some reporter ions are provided with MSnbase an can be loaded with the data function. These reporter ions data sets are:

iTRAQ4: ReporterIon object for the iTRAQ 4-plex set. Load with data(iTRAQ4).

iTRAQ5: ReporterIon object for the iTRAQ 4-plex set plus the isobaric tag. Load with data(iTRAQ5).

TMT6: ReporterIon object for the TMT 6-plex set. Load with data(TMT6).

TMT7: ReporterIon object for the TMT 6-plex set plus the isobaric tag. Load with data(TMT6).

Objects from the Class

Objects can be created by calls of the form new("ReporterIons", ...).

Slots

name: Object of class "character" to identify the ReporterIons instance.

reporterNames: Object of class "character" naming each individual reporter of the ReporterIons instance. If not provided explicitely, they are names by concatenating the ReporterIons name and the respective MZ values.

description: Object of class "character" to describe the ReporterIons instance.

mz: Object of class "numeric" providing the MZ values of the reporter ions.

col: Object of class "character" providing colours to highlight the reporters on plots.

width: Object of class "numeric" indicating the width around the individual reporter ions MZ values were to search for peaks. This is dependent on the mass spectrometer's resolution and is used for peak picking when quantifying the reporters. See quantify for more details about quantification.

.__classVersion__: Object of class "Versions" indicating the version of the ReporterIons instance. Intended for developer use and debugging.

Extends

Class "Versioned", directly.

Methods

show(object) Displays object content as text.

object[] Subsets one or several reporter ions of the ReporterIons object and returns a new instance of the same class.

length(object) Returns the number of reporter ions in the instance.

`mz(object)` Returns the expected mz values of reporter ions.

`reporterColours(object)` **or reporterColors(object)** Returns the colours used to highlight the reporter ions.

`reporterNames(object)` Returns the name of the individual reporter ions. If not specified or is an incorrect number of names is provided at initialisation, the names are generated automatically by concatenating the instance name and the reporter's MZ values.

`reporterNames(object) <- value` Sets the reporter names to `value`, which must be a character of the same length as the number of reporter ions.

`width(object)` Returns the widhts in which the reporter ion peaks are expected.

Author(s)

Laurent Gatto <lg390@cam.ac.uk>

References

Ross PL, Huang YN, Marchese JN, Williamson B, Parker K, Hattan S, Khainovski N, Pillai S, Dey S, Daniels S, Purkayastha S, Juhasz P, Martin S, Bartlet-Jones M, He F, Jacobson A, Pappin DJ. "Multiplexed protein quantitation in Saccharomyces cerevisiae using amine-reactive isobaric tagging reagents." *Mol Cell Proteomics*, 2004 Dec;3(12):1154-69. Epub 2004 Sep 22. PubMed PMID: 15385600.

Thompson A, Sch\"afer J, Kuhn K, Kienle S, Schwarz J, Schmidt G, Neumann T, Johnstone R, Mohammed AK, Hamon C. "Tandem mass tags: a novel quantification strategy for comparative analysis of complex protein mixtures by MS/MS." *Anal Chem.* 2003 Apr 15;75(8):1895-904. *Erratum* in: *Anal Chem.* 2006 Jun 15;78(12):4235. Mohammed, A Karim A [added] and *Anal Chem.* 2003 Sep 15;75(18):4942. Johnstone, R [added]. PubMed PMID: 12713048.

See Also

`TMT6` or `iTRAQ4` for readily available examples.

Examples

```
## Code used for the iTRAQ4 set
rep <- new("ReporterIons",
       description="4-plex iTRAQ",
       name="iTRAQ4",
       reporterNames=c("iTRAQ4.114","iTRAQ4.115",
                       "iTRAQ4.116","iTRAQ4.117"),
       mz=c(114.1,115.1,116.1,117.1),
       col=c("red","green","blue","yellow"),
       width=0.05)
rep
reporterNames(rep)
rep[1:2]
```

Spectrum-class *The "Spectrum" Class*

Description

Virtual container for spectrum data common to all different types of spectra. A Spectrum object can not be directly instanciated. Use "Spectrum1" and "Spectrum2" instead.

Slots

msLevel: Object of class "integer" indicating the MS level: 1 for MS1 level Spectrum1 objects and 2 for MSMSM Spectrum2 objects. Levels > 2 have not been tested and will be handled as MS2 spectra.

peaksCount: Object of class "integer" indicating the number of MZ peaks.

rt: Object of class "numeric" indicating the retention time (in seconds) for the current ions.

acquisitionNum: Object of class "integer" corresponding to the acquisition number of the current spectrum.

scanIndex: Object of class "integer" indicating the scan index of the current spectrum.

mz: Object of class "numeric" of length equal to the peaks count (see peaksCount slot) indicating the MZ values that have been measured for the current ion.

intensity: Object of class "numeric" of same length as mz indicating the intensity at which each mz datum has been measured.

fromFile: Object of class "integer" referencing the file the spectrum originates. The file names are stored in the processingData slot of the "MSnExp" or "MSnSet" instance that contains the current "Spectrum" instance.

.__classVersion__: Object of class "Versions" indicating the version of the Spectrum class. Intended for developer use and debugging.

Extends

Class "Versioned", directly.

Methods

acquisitionNum(object) Returns the acquisition number of the spectrum as an integer.

fromFile(object) Returns the index of the raw data file from which the current instances originates as an integer.

intensity(object) Returns an object of class "numeric" containing the intensities of the spectrum.

msLevel(object) Returns an MS level of the spectrum as an integer.

mz(object) Returns an object of class "numeric" containing the MZ value of the spectrum peaks.

peaksCount(object) Returns the number of peaks (possibly of 0 intensity) as an integer.

rtime(object) Returns the retention time for the spectrum as an integer.

tic(object) Returns the total ion count for the spectrum as a numeric.

clean `signature(object = "Spectrum")`: Removes unused 0 intensity data points. See `clean` documentation for more details and examples.

plot `signature(x = "Spectrum", y = "missing")`: Plots intensity against mz. See `plot.Spectrum` documentation for more details.

quantify `signature(object = "Spectrum")`: Quatifies defined peaks in the spectrum. See `quantify` documentation for more details.

removePeaks `signature(object = "Spectrum")`: Remove peaks lower that a threshold `t`. See `removePeaks` documentation for more details and examples.

show `signature(object = "Spectrum")`: Displays object content as text.

trimMz `signature(object = "Spectrum")`: Trims the MZ range of all the spectra of the `MSnExp` instance. See `trimMz` documentation for more details and examples.

Note

This is a virtual class and can not be instanciated directly.

Author(s)

Laurent Gatto <lg390@cam.ac.uk>

See Also

Instaciable sub-classes `"Spectrum1"` and `"Spectrum2"` for MS1 and MS2 spectra.

Spectrum1-class *The "Spectrum1" Class for MS1 Spectra*

Description

`Spectrum1` extends the `"Spectrum"` class and introduces an MS1 specific attribute in addition to the slots in `"Spectrum"`. `Spectrum1` instances are not created directly but are contained in the `assayData` slot of an `"MSnExp"`.

Slots

`polarity`: Object of class `"integer"` indicating the polarity if the ion.

`msLevel`: Object of class `"integer"` indicating the MS level: always 1 in this case (inherited from `"Spectrum"`).

`peaksCount`: Object of class `"integer"` indicating the number of MZ peaks (inherited from `"Spectrum"`).

`rt`: Object of class `"numeric"` indicating the retention time (in seconds) for the current ions (inherited from `"Spectrum"`).

`acquisitionNum`: Object of class `"integer"` corresponding to the acquisition number of the current spectrum (inherited from `"Spectrum"`).

`scanIndex`: Object of class `"integer"` indicating the scan index of the current spectrum (inherited from `"Spectrum"`).

`mz`: Object of class `"numeric"` of length equal to the peaks count (see `peaksCount` slot) indicating the MZ values that have been measured for the current ion (inherited from `"Spectrum"`).

`intensity`: Object of class "`numeric`" of same length as `mz` indicating the intensity at which each `mz` datum has been measured "`Spectrum`").

`.__classVersion__`: Object of class "`Versions`" indicating the versions of the `Spectrum` and `Spectrum1` classes of the instance. Intended for developer use and debugging.

Extends

Class "`Spectrum`", directly. Class "`Versioned`", by class "Spectrum", distance 2.

Methods

See "`Spectrum`" for additional accessors and methods to process `Spectrum1` objects.

`polarity(object)` Returns the polarity of the spectrum as an integer.

Author(s)

Laurent Gatto <lg390@cam.ac.uk>

See Also

Virtual super-class "`Spectrum`", "`Spectrum2`" for MS2 spectra and "`MSnExp`" for a full experiment container.

Spectrum2-class *The "Spectrum2" Class for MSMS Spectra*

Description

`Spectrum2` extends the "`Spectrum`" class and introduces several MS2 specific attributes in addition to the slots in "`Spectrum`". `Spectrum2` are not created directly but are contained in the `assayData` slot of an "`MSnExp`".

Slots

`merged`: Object of class "`numeric`" indicating of how many combination the current spectrum is the result of.

`ms1scan`: Object of class "`integer`" indicating the relative MS1 scan index of the parent ion.

`precursorMz`: Object of class "`numeric`" providing the precursor ion MZ value.

`precursorIntensity`: Object of class "`numeric`" providing the precursor ion intensity.

`precursorCharge`: Object of class "`integer`" indicating the precursor ion charge.

`collisionEnergy`: Object of class "`numeric`" indicating the collision energy used to fragment the parent ion.

`msLevel`: Object of class "`integer`" indicating the MS level: 2 in this case (inherited from "`Spectrum`").

`peaksCount`: Object of class "`integer`" indicating the number of MZ peaks (inherited from "`Spectrum`").

`rt`: Object of class "`numeric`" indicating the retention time (in seconds) for the current ions (inherited from "`Spectrum`").

`acquisitionNum`: Object of class "`integer`" corresponding to the acquisition number of the current spectrum (inherited from "`Spectrum`").

`scanIndex`: Object of class "`integer`" indicating the scan index of the current spectrum (inherited from "`Spectrum`").

`mz`: Object of class "`numeric`" of length equal to the peaks count (see `peaksCount` slot) indicating the MZ values that have been measured for the current ion (inherited from "`Spectrum`").

`intensity`: Object of class "`numeric`" of same length as `mz` indicating the intensity at which each `mz` datum has been measured "`Spectrum`").

`.__classVersion__`: Object of class "`Versions`" indicating the versions of the `Spectrum` and `Spectrum2` classes of the instance. Intended for developer use and debugging.

Extends

Class "`Spectrum`", directly. Class "`Versioned`", by class "Spectrum", distance 2.

Methods

See "`Spectrum`" for additional accessors and methods for `Spectrum2` objects.

`ms1scan(object)` Returns the relative MS1 scan index of the parent ion as an integer.

`precursorMz(object)` Returns the precursor MZ value as a numeric.

`precursorIntensity(object)` Returns the precursor intensity as a numeric.

`precursorCharge(object)` Returns the precursor intensity as a integer.

`collisionEnergy(object)` Returns the collision energy as an numeric.

Author(s)

Laurent Gatto <lg390@cam.ac.uk>

See Also

Virtual super-class "`Spectrum`", "`Spectrum1`" for MS1 spectra and "`MSnExp`" for a full experiment container.

TMT6 *TMT 6-plex set*

Description

This instance of class "`ReporterIons`" corresponds to the TMT 6-plex set, i.e the 126, 127, 128, 129, 130 and 131 isobaric tags. In the TMT7 data set, an unfragmented tag, i.e reporter and attached isobaric tag, is also included at MZ 229. These objects are used to plot the reporter ions of interest in an MSMS spectra (see "`Spectrum2`") as well as for quantification (see `quantify`).

Usage

```
TMT6
TMT7
```

References

Thompson A, Sch\"afer J, Kuhn K, Kienle S, Schwarz J, Schmidt G, Neumann T, Johnstone R, Mohammed AK, Hamon C. "Tandem mass tags: a novel quantification strategy for comparative analysis of complex protein mixtures by MS/MS." *Anal Chem.* 2003 Apr 15;75(8):1895-904. *Erratum* in: *Anal Chem.* 2006 Jun 15;78(12):4235. Mohammed, A Karim A [added] and *Anal Chem.* 2003 Sep 15;75(18):4942. Johnstone, R [added]. PubMed PMID: 12713048.

See Also

iTRAQ4.

Examples

```
TMT6
TMT6[1:2]

newReporter <- new("ReporterIons",
                   description="an example",
                   name="my reporter ions",
                   reporterNames=c("myrep1","myrep2"),
                   mz=c(121,122),
                   col=c("red","blue"),
                   width=0.05)
newReporter
```

---

clean-methods *Cleans 'MSnExp' or 'Spectrum' instances*

Description

This method cleans out individual spectra (`Spectrum` instances) or whole experiments (`MSnExp` instances) of 0-intensity peaks. Origninal 0-intensity values are retained only around peaks. If more than two 0's were separating two peaks, only the first and last ones, those directly adjacent to the peak ranges are kept. If two peaks are separated by only one 0-intensity value, it is retained. An illustrative example is shown below.

Methods

`signature(object = "MSnExp", verbose = "logical")` Cleans all spectra in `MSnExp` object. Displays a control bar if verbose set to `TRUE` (default). Returns a cleaned `MSnExp` instance.

`signature(object = "Spectrum")` Cleans the `Spectrum` object. Returns a cleaned `Spectrum` instance.

Author(s)

Laurent Gatto <lg390@cam.ac.uk>

See Also

`removePeaks` and `trimMz` for other spectra processing methods.

Examples

```
int <- c(1,0,0,0,1,1,1,0,0,1,1,0,0,0,1,0,0,0)
sp1 <- new("Spectrum2",
           intensity=int,
           mz=1:length(int))
sp2 <- clean(sp1)
intensity(sp1)
intensity(sp2)
file <- dir(system.file(package="MSnbase",dir="extdata"),
            full.name=TRUE,pattern="mzXML$")
aa <- readMzXMLData(file,verbose=FALSE)
bb <- clean(aa,verbose=FALSE)
sum(peaksCount(aa))
sum(peaksCount(bb))
processingData(bb)
```

---

extractPrecSpectra-methods

Extracts precursor-specific spectra from an 'MSnExp' object

Description

Extracts the MSMS spectra that originate from the precursor(s) having the same MZ value as defined in the`prec` argument.

A warning will be issued of one or several of the precursor MZ values in `prec` are absent in the experiment precursor MZ values (i.e in `precursorMz(object)`).

Methods

`signature(object = "MSnExp", prec = "numeric")` Returns an "`MSnExp`" containing MSMS spectra whose precursor MZ values are in `prec`.

Author(s)

Laurent Gatto <lg390@cam.ac.uk>

See Also

`extractSpectra`

Examples

```
file <- dir(system.file(package="MSnbase",dir="extdata"),
            full.name=TRUE,pattern="mzXML$")
aa <- readMzXMLData(file,verbose=FALSE)
my.prec <- precursorMz(aa)[1:2]
bb <- extractPrecSpectra(aa,my.prec)
stopifnot(all(precursorMz(bb) %in% my.prec))
processingData(bb)
```

extractSpectra-methods

Extracts a set of spectra from an 'MSnExp' object

Description

Extracts the spectra defined by the `selected` parameter and returns a subsetted "`MSnExp`" instance. The `featureData` slot is also subsetted accordingly.

Note that values in `selected` are recycled shorter than `length(MSnExp)`.

Methods

`signature(object = "MSnExp", selected = "logical")` Returns an "`MSnExp`" object containing spectra defined by the logical `selected` argument.

Author(s)

Laurent Gatto <lg390@cam.ac.uk>

See Also

`extractPrecSpectra`

Examples

```
file <- dir(system.file(package="MSnbase",dir="extdata"),
            full.name=TRUE,pattern="mzXML$")
aa <- readMzXMLData(file,verbose=FALSE)
sel <- rep(FALSE,length(aa))
sel[1:3] <- TRUE
bb <- extractSpectra(aa,sel)
stopifnot(length(bb) == sum(sel))
processingData(bb)
```

formatRt

Format Retention Time

Description

Converts seconds to min:sec format

Usage

```
formatRt(rt)
```

Arguments

`rt`	retention in in seconds

Details

This function is used to convert retention times, expressed in seconds, in the more human friendly format mm:sec.

Value

a character string in mm:ss format

Author(s)

Laurent Gatto <lg390@cam.ac.uk>

Examples

```
formatRt(1524)
```

---

iTRAQ4 *iTRAQ 4-plex set*

Description

This instance of class "ReporterIons" corresponds to the iTRAQ 4-plex set, i.e the 114, 115, 116 and 117 isobaric tags. In the iTRAQ5 data set, an unfragmented tag, i.e reporter and attached isobaric tag, is also included at MZ 145. These objects are used to plot the reporter ions of interest in an MSMS spectra (see "Spectrum2") as well as for quantification (see quantify).

Usage

```
iTRAQ4
iTRAQ5
iTRAQ8
iTRAQ9
```

References

Ross PL, Huang YN, Marchese JN, Williamson B, Parker K, Hattan S, Khainovski N, Pillai S, Dey S, Daniels S, Purkayastha S, Juhasz P, Martin S, Bartlet-Jones M, He F, Jacobson A, Pappin DJ. "Multiplexed protein quantitation in Saccharomyces cerevisiae using amine-reactive isobaric tagging reagents." *Mol Cell Proteomics*, 2004 Dec;3(12):1154-69. Epub 2004 Sep 22. PubMed PMID: 15385600.

See Also

TMT6.

Examples

```
iTRAQ4
iTRAQ4[1:2]

newReporter <- new("ReporterIons",
                   description="an example",
                   name="my reporter ions",
                   reporterNames=c("myrep1","myrep2"),
                   mz=c(121,122),
                   col=c("red","blue"),
                   width=0.05)
newReporter
```

`pSet-class` *Class to Contain Raw Mass-Spectrometry Assays and Experimental Metadata*

Description

Container for high-throughput mass-spectrometry assays and experimental metadata. This class is based on Biobase's "eSet" virtual class, with the notable exception that 'assayData' slot is an environment contain objects of class "Spectrum".

Objects from the Class

A virtual Class: No objects may be created from it. See "MSnExp" for instantiatable sub-classes.

Slots

assayData: Object of class "environment" containing the MS spectra (see "Spectrum1" and "Spectrum2"). Slot is inherited from "linkS4class{pSet}".

phenoData: Object of class "AnnotatedDataFrame" containing experimenter-supplied variables describing sample (i.e the individual tags for an labelled MS experiment) See phenoData for more details. Slot is inherited from "linkS4class{pSet}".

featureData: Object of class "AnnotatedDataFrame" containing variables describing features (spectra in our case), e.g. identificaiton data, peptide sequence, identification score,... (inherited from "eSet"). See featureData for more details. Slot is inherited from "linkS4class{pSet}".

experimentData: Object of class "MIAPE", containing details of experimental methods. See experimentData for more details. Slot is inherited from "linkS4class{pSet}".

protocolData: Object of class "AnnotatedDataFrame" containing equipment-generated variables (inherited from "eSet"). See protocolData for more details. Slot is inherited from "linkS4class{pSet}".

processingData: Object of class "MSnProcess" that records all processing. Slot is inherited from "linkS4class{pSet}".

.__classVersion__: Object of class "Versions" describing the versions of R, "linkS4class{Biobase}" "pSet" and MSnExp of the current instance. Slot is inherited from "linkS4class{pSet}". Intended for developer use and debugging (inherited from "eSet").

Extends

Class "VersionedBiobase", directly. Class "Versioned", by class "VersionedBiobase", distance 2.

Methods

Methods defined in derived classes may override the methods described here.

[`signature(x = "pSet")`: Subset current object and return object of same class.

[[`signature(x = "pSet")`: Direct access to individual spectra.

abstract Access abstract in `experimentData`.

assayData `signature(object = "pSet")`: Access the `assayData` slot. Returns an `environment`.

desciption `signature(x = "pSet")`: Synonymous with experimentData.

dim `signature(x = "pSet")`: Returns the dimensions of the `phenoData` slot.

experimentData `signature(x = "pSet")`: Access details of experimental methods.

featureData `signature(x = "pSet")`: Access the `featureData` slot.

fData `signature(x = "pSet")`: Access feature data information.

featureNames `signature(x = "pSet")`: Coordinate access of feature names (e.g spectra, peptides or proteins) in `assayData` slot.

fileNames `signature(object = "pSet")`: Access file names in the `processingData` slot.

fromFile `signature(object = "pSet")`: Access raw data file indexes (to be found in the 'codeprocessingData' slot) from which the individual object's spectra where read from.

fvarMetadata `signature(x = "pSet")`: Access metadata describing features reported in `fData`.

fvarLabels `signature(x = "pSet")`: Access variable labels in `featureData`.

length `signature(x = "pSet")`: Returns the number of features in the `assayData` slot.

notes `signature(x = "pSet")`: Retrieve and unstructured notes associated with `pSet` in the `experimentData` slot.

pData `signature(x = "pSet")`: Access sample data information.

phenoData `signature(x = "pSet")`: Access the `phenoData` slot.

processingData `signature(object = "pSet")`: Access the `processingData` slot.

protocolData `signature(x = "pSet")`: Access the `protocolData` slot.

pubMedIds `signature(x = "pSet")`: Access PMIDs in `experimentData`.

sampleNames `signature(x = "pSet")`: Access sample names in `phenoData`.

spectra `signature(x = "pSet")`: Access the `assayData` slot, returning the features as a `list`.

varMetadata `signature(x = "pSet")`: Access metadata describing variables reported in `pData`.

varLabels `signature(x = "pSet")`: Access variable labels in `phenoData`.

acquisitionNum `signature(object = "pSet")`: Accessor for spectra acquisition numbers.

collisionEnergy `signature(object = "pSet")`: Accessor for MS2 spectra collision energies.

intensity `signature(object = "pSet")`: Accessor for spectra instenities, returned as named list.

msInfo `signature(object = "pSet")`: Prints the MIAPE-MS meta-data stored in the `experimentData` slot.

msLevel `signature(object = "pSet")`: Accessor for spectra MS levels.

mz `signature(object = "pSet")`: Accessor for spectra M/Z values, returned as a named list.

peaksCount `signature(object = "pSet")`: Accessor for spectra preak counts.

polarity `signature(object = "pSet")`: Accessor for MS1 spectra polarities.

precursorCharge `signature(object = "pSet")`: Accessor for MS2 precursor charges.

precursorIntensity `signature(object = "pSet")`: Accessor for MS2 precursor intensity.

precursorMz `signature(object = "pSet")`: Accessor for MS2 precursor M/Z values.

rtime `signature(object = "pSet")`: Accessor for spectra retention times.

tic `signature(object = "pSet")`: Accessor for spectra total ion counts.

header `signature(object = "pSet")`: Returns a data frame containing all available spectra parameters (MSn only).

Author(s)

Laurent Gatto <lg390@cam.ac.uk>

References

The "eSet" class, on which `pSet` is based.

See Also

"MSnExp" for an instantiatable application of `pSet`.

Examples

```
showClass("pSet")
```

---

plot-methods *Plotting 'Spectrum' object(s)*

Description

These method plot mass spectra MZ values against the intensities as line plots. Full spectra (using the `full` parameter) or specific peaks of interest can be plotted using the `reporters` parameter. If `reporters` are specified and `full` is set to 'TRUE', a sub-figure of the reporter ions is inlaid inside the full spectrum.

If an "MSnExp" is provided as argument, all the spectra are aligned vertically. Experiments can be subsetted to extract spectra of interest using the `extractSpectra` or `extractPrecSpectra` methods.

The methods make use the `ggplot2` system. An object of class 'ggplot' is returned invisibly.

Arguments

x	Objects of class "Spectrum" or "MSnExp" to be plotted.
y	Not used in these methods.
reporters	An object of class "ReporterIons" that defines the peaks to be plotted. If not specified, full must be set to 'TRUE'.
full	Logical indicating whether full spectrum (respectively spectra) of only reporter ions of interest should be plotted. Default is 'FALSE', in which case reporters must be defined.

Methods

signature(x = "MSnExp", y = "missing", reporters = "ReporterIons", full = "logic
: Plots *all* the spectra in the MSnExp object vertically. One of reporters must be defined or full set to 'TRUE'. In case of MSnExp objects, repoter ions are not inlaid when full is 'TRUE'.

signature(x = "Spectrum", y = "missing", reporters = "ReporterIons", full = "log
: Displays the MZs against intensities of the Spectrum object as a line plot. At least one of reporters must be defined or full set to 'TRUE'.

reporters and full are used only for "Spectrum2" objects. Full "Spectrum1" spectra are plotted by default.

Author(s)

Laurent Gatto <lg390@cam.ac.uk>

Examples

```
file <- dir(system.file(package="MSnbase",dir="extdata"),full.name=TRUE,pattern="mzXML$")
aa <- readMzXMLData(file,verbose=FALSE)
data(iTRAQ4)
plot(extractPrecSpectra(aa,precursorMz(aa)[1]),reporters=iTRAQ4)
plot(extractPrecSpectra(aa,precursorMz(aa)[1]),full=TRUE)
plot(aa[["X43"]],reporters=iTRAQ4,full=TRUE)
```

---

plot2d-methods *Graphical overview of the 'MSnExp' object*

Description

These methods plot the retention time vs. precursor MZ for the whole "MSnExp" experiment. Individual dots will be colour-coded to describe individual spectra's peaks count, total ion count, precursor charge (MS2 only) or file of origin.

The methods make use the ggplot2 system. An object of class 'ggplot' is returned invisibly.

Arguments

object	An object of class "MSnExp" or and 'data.frame'. In the latter case, the data frame must have numerical columns named 'retention.time' and 'precursor.mz' and one of 'tic', 'file', 'peaks.count' or 'charge', depending on the z parameter. Such a data.frame is typically generated using the header method on "MSnExp" object.
z	A character indicating according to what variable to colour the dots. One of, possibly abreviated, "tic" (total ion count), "file" (raw data file), "peaks.count" (peaks count) or "charge" (precursor charge).

Methods

signature(object = "MSnExp", z = "character") Plots a 'MSnExp' summary.

signature(object = "data.frame", z = "character") Plots a summary of the 'MSnExp' experiment described by the data frame.

Author(s)

Laurent Gatto <lg390@cam.ac.uk>

Examples

```
file <- dir(system.file(package="MSnbase",dir="extdata"),full.name=TRUE,pattern="mzXML$")
aa <- readMzXMLData(file,verbose=FALSE)
plot2d(aa,z="tic")
plot2d(aa,z="file")
```

---

quantify-methods *Quantifies 'MSnExp' and 'Spectrum' objects*

Description

This method quantifies individual "Spectrum" objects or full "MSnExp" experiments. Current implementation quantify specific peaks in individual spectra. The peaks of interest are defined by the reporters parameter using one of the possible method methods. This essentially qualifies for MSMS quantitation using isobaric tags, altough any peak can be defined in reporters.

Arguments

object	Objects of class "Spectrum" or "MSnExp".
method	Peak quantitation method. One of, possibly abreviated "trapezoidation", "max", or "sum". These methods return respectively the area under the peak, the maximum of the peak or the sum of all intensities of the peaks.
reporters	An object of class "ReporterIons" that defines the peaks to be quantified.
verbose	Verbose of the output (only for MSnExp objects).

Details

"ReporterIons" define specific MZ at which peaks are expected and a window around that MZ value. A peak of interest is searched for in that window and warnings are thrown in case no peak is found in that region or the peak extends outside the window. Once the range of the curve is found, quantification is performed. If no data points are found in the expected region, 'NA' is returned for the reporter peak MZ.

Methods

`signature(object = "MSnExp", method = "character", reporters = "ReporterIons", v`
Quantifies peaks defined in `reporters` using `method` in all spectra of the `MSnExp` object. If verbose is set to `TRUE`, a progress bar will be displayed.

An object of class "MSnSet" is returned containing the quantified feature expression and all meta data inherited from the `MSnExp` argument.

`signature(object = "Spectrum", method = "character", reporters = "ReporterIons")`
Quantifies peaks defined in `reporters` using `method` in the `Spectrum` object.

A list of length 2 will be returned. The first element, named `peakQuant`, is a 'numeric' of length equal to `length(reporters)` with quantitation of the reporter peaks using `method`.

The second element, names `curveStats`, is a 'data.frame' of dimension `length(reporters)` times 7 giving, for each reporter curve parameters: maximum intensity ('maxInt'), number of maxima ('nMaxInt'), number of data points defined the curve ('baseLength'), lower and upper MZ values for the curve ('lowerMz' and 'upperMz'), reporter ('reporter') and precursor MZ value ('precursor') when available.

Author(s)

Laurent Gatto <lg390@cam.ac.uk>

Examples

```
file <- dir(system.file(package="MSnbase",dir="extdata"),full.name=TRUE,pattern="mzXML$")
aa <- readMzXMLData(file,verbose=FALSE)
data(iTRAQ4)
msnset <- quantify(aa,method="trap",reporters=iTRAQ4)
msnset
qty64 <- quantify(aa[["X43"]],method="trap",iTRAQ4[1])
qty64$peakQuant
qty64$curveStats
```

---

readIspyData *Reads an ispy2 result spread sheet and creates a fully featured*

Description

Reads an ispy2 tab-delimited spreadsheet and generates the corresponding MSnSet object.

Usage

```
readIspyData(file = "ispy_results.tsv", uniquePeps = TRUE, pep = 0.05,
  na.rm = TRUE, min.int = 0, reporters = 19:23, skipFillUp = 24,
 fillUp = TRUE, verbose = TRUE)
```

Arguments

file	A character, indicating the file name to be read in. Default is "ispy_results.tsv".
uniquePeps	A logical, indicating whether only unique peptides should be included. Default is TRUE.
pep	A numeric indicating the posterior error probability thersholdfor peptides to be considered correctly identified. Default is 0.05.
na.rm	A logical indicating whether reporter ions containing one or more NA values should be excluded. Default is TRUE.
min.int	A numeric indicating the minimal summed intensity threshold for reporter data to be imported. Default is 0.
reporters	A numeric indicating column indices of reporter ions quantitation data. Default is 19:23 for iTRAQ 4-plex.
skipFillUp	A numeric indicating which column not to fill up. Generally the column following the reporter ions quantitation data. Default is 24.
fillUp	A logical specifying if empty cells should be filled up using their upper neighbour value. Default is TRUE, but should be set to FALSE of result sheet is already filled up (recent ispy result sheets are) to speed up import.
verbose	A logical indicating whether verbose output is to be printed out.

Value

An object of class "MSnSet".

Author(s)

Laurent Gatto

References

Ispy is a set of perl script to analyse SILAC, 15N and MSMS data developed by Phil D. Charles <pdc35@cam.ac.uk> at CCP http://www.bio.cam.ac.uk/proteomics/. No ispy references published yet.

See Also

readMzXMLData to import raw data.

Examples

```
## Not run: ispy <- readIspyData("ispy_results.tsv")
```

readMzXMLData *Import mzXML files as 'MSnExp' instances.*

Description

Reads mzXML files and generates an "MSnExp" object.

Usage

```
readMzXMLData(files, pdata = NULL, msLevel = 2, verbose = TRUE, centroided = FAL
```

Arguments

files	file names to be read in a character vector.
pdata	an object of class "NAnnotatedDataFrame".
msLevel	MS level spectra to be read. Use '1' for MS1 spectra of any larger numeric for MSn spectra. Default is '2'.
verbose	verbosity flag.
centroided	Logical indicating whether spectra are centroided or not. Default is 'FALSE'. Used to initialise "MSnProcess" object in processingData slot.
smoothed	Logical indicating whether spectra already smoothed or not. Default is 'FALSE'. Used to initialise "MSnProcess" object in processingData slot.
removePeaks	If > 0 (default), all peaks less or equal then value will set to 0. See removePeaks for more details and examples.
clean	Logical indicating whether 0 intensity peaks should be discarded from spectra. Useful is removePeaks is set. Default is 'FALSE'. See clean for more details and examples.

Value

An "MSnExp" object.

Author(s)

Laurent Gatto <lg390@cam.ac.uk>

See Also

"MSnExp"

Examples

```
file <- dir(system.file(package="MSnbase",dir="extdata"),
            full.name=TRUE,
            pattern="mzXML$")
aa <- readMzXMLData(file)
aa
```

`removePeaks-methods` *Removes low intensity peaks*

Description

This method sets low intensity peaks from individual spectra (`Spectrum` instances) or whole experiments (`MSnExp` instances) to 0. The intensity threshold is set with the `t` parameter. Default is the `"min"` character. The threshold is then set as the non-0 minimum intensity found in the spectrum. Any other numeric values is valid. All peaks with maximum intensity smaller or equal to `t` are set to 0.

Note that the number of peaks is not changed; the peaks below the threshold are set to 0 and the object is not cleanded out (see `clean`). An illustrative example is shown below.

Methods

`signature(object = "MSnExp", t, verbose = "logical" )` Removes low intensity peaks of all spectra in `MSnExp` object. `t` sets the minimum peak intensity. Default is "min", i.e the smallest intensity in each spectrum. Other `numeric` values are valid. Displays a control bar if verbose set to `TRUE` (default). Returns a new `MSnExp` instance.

`signature(object = "Spectrum", t)` Removes low intensity peaks of `Spectrum` object. `t` sets the minimum peak intensity. Default is "min", i.e the smallest intensity in each spectrum. Other `numeric` values are valid. Returns a new `Spectrum` instance.

Author(s)

Laurent Gatto <lg390@cam.ac.uk>

See Also

`clean` and `trimMz` for other spectra processing methods.

Examples

```
int <- c(2,0,0,0,1,5,1,0,0,1,3,1,0,0,1,4,2,1)
sp1 <- new("Spectrum2",
           intensity=int,
           mz=1:length(int))
sp2 <- removePeaks(sp1) ## no peaks are removed here
                        ## as min intensity is 1 and
                        ## no peak has a max int <= 1
sp3 <- removePeaks(sp1,3)
intensity(sp1)
intensity(sp2)
intensity(sp3)

peaksCount(sp1) == peaksCount(sp2)
peaksCount(sp1) < peaksCount(sp3)

file <- dir(system.file(package="MSnbase",dir="extdata"),
            full.name=TRUE,pattern="mzXML$")
aa <- readMzXMLData(file)
```

```
aa <- removePeaks(aa)
processingData(aa)
```

`trimMz-methods` *Trims 'MSnExp' or 'Spectrum' instances*

Description

This method selects a range of MZ values in a single spectrum (`Spectrum` instances) or all the spectra of an experiment (`MSnExp` instances). The regions to trim are defined by the range of `mzlim` argument, such as MZ values < `min(mzlim)` and MZ values > `max(mzlim)` are trimmed away.

Methods

`signature(object = "MSnExp", mzlim = "numeric")` Trims all spectra in `MSnExp` object according to `mzlim`. Returns a cleaned `MSnExp` instance.

`signature(object = "Spectrum", mzlim = "numeric")` Trims the `Spectrum` object and retruns a new trimmed object.

Author(s)

Laurent Gatto <lg390@cam.ac.uk>

See Also

`removePeaks` and `clean` for other spectra processing methods.

Examples

```
mz <- 1:100
sp1 <- new("Spectrum2",
           mz=mz,
           intensity=abs(rnorm(length(mz))))
sp2 <- trimMz(sp1,c(25,75))
range(mz(sp1))
range(mz(sp2))
file <- dir(system.file(package="MSnbase",dir="extdata"),
            full.name=TRUE,pattern="mzXML$")
aa <- readMzXMLData(file,verbose=FALSE)
bb <- trimMz(aa,c(113,117))
range(mz(bb))
range(mz(bb))
processingData(bb)
```

Index