Package 'MsQuality'

May 23, 2026

Type Package

Title MsQuality - Quality metric calculation from Spectra, MsExperiment and Chromatograms objects

Version 1.13.0

Date 2026-03-30

VignetteBuilder knitr

Description The MsQuality provides functionality to calculate quality metrics for mass spectrometry-derived, spectral data at the per-sample level. MsQuality relies on the mzQC framework of quality metrics defined by the Human Proteom Organization-Proteomics Standards Initiative (HUPO-PSI). These metrics quantify the quality of spectral raw files using a controlled vocabulary. The package is especially addressed towards users that acquire mass spectrometry data on a large scale (e.g. data sets from clinical settings consisting of several thousands of samples).
The MsQuality package allows to calculate low-level quality metrics that require minimum information on mass spectrometry data: retention time, m/z values, and associated intensities. MsQuality relies on the Spectra package, or alternatively the MsExperiment package, and its infrastructure to store spectral data. Additionally, MsQuality supports Chromatograms objects from the Chromatograms package for chromatographic quality metrics.

Depends R (>= 4.2.0)

Imports BiocParallel (>= 1.32.0), Chromatograms (>= 1.1.5), ggplot2 (>= 3.3.5), htmlwidgets (>= 1.5.3), methods (>= 4.2.0), MsDataHub (>= 1.10.0), MsExperiment (>= 0.99.0), plotly (>= 4.9.4.1), ProtGenerics (>= 1.24.0), rlang (>= 1.1.1), rmzqc (>= 0.7.0), shiny (>= 1.6.0), shinydashboard (>= 0.7.1), Spectra (>= 1.13.2), stats (>= 4.2.0), stringr (>= 1.4.0), tibble (>= 3.1.4), tidyr (>= 1.1.3), utils (>= 4.2.0), MsCoreUtils (>= 1.19.0), MetaboCoreUtils (>= 1.19.2)

Suggests BiocGenerics (>= 0.24.0), BiocStyle (>= 2.6.1), dplyr (>= 1.0.5), knitr (>= 1.11), mzR (>= 2.32.0), rmarkdown (>= 2.7), S4Vectors (>= 0.29.17), testthat (>= 2.2.1)

biocViews Metabolomics, Proteomics, MassSpectrometry, QualityControl

URL https://www.github.com/tnaake/MsQuality/

BugReport https://www.github.com/tnaake/MsQuality/issues/

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.3

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

git_branch devel

git_last_commit 342c9d6

git_last_commit_date 2026-04-28

Repository Bioconductor 3.24

Date/Publication 2026-05-22

Author Thomas Naake [aut, cre] (ORCID:
<https://orcid.org/0000-0001-7917-5580>),
Johannes Rainer [aut] (ORCID: <https://orcid.org/0000-0002-6977-7147>),
Helge Hecht [ctb],
Philippine Louail [aut] (ORCID:
<https://orcid.org/0009-0007-5429-6846>)

Maintainer Thomas Naake <thomasnaake@googlemail.com>

Contents

`MsQuality-package` *MsQuality - Quality metric calculation from Spectra, MsExperiment and Chromatograms objects*

Description

MsQuality enables to calculate quality metrics of mass spectrometry data. It is build upon Spectra and MsExperiment objects.

Details

The DESCRIPTION file: This package was not yet installed at build time.

Index: This package was not yet installed at build time.

Author(s)

Thomas Naake [aut, cre] (ORCID: <https://orcid.org/0000-0001-7917-5580>), Johannes Rainer [aut] (ORCID: <https://orcid.org/0000-0002-6977-7147>), Helge Hecht [ctb], Philippine Louail [aut] (ORCID: <https://orcid.org/0009-0007-5429-6846>) Maintainer: Thomas Naake <thomas-naake@googlemail.com>

Examples

```
## Not run: calculateMetrics(object = spectra)
## Not run: calculateMetrics(object = mse)
```

`.rtOrderSpectra` *Order Spectra according to increasing retention time*

Description

The function `.rtOrderSpectra` orders the features in a `Spectra` object according to the (increasing) retention time values.

Usage

```
.rtOrderSpectra(spectra)
```

Arguments

`spectra`	`Spectra` object

Details

Internal function in quality metric functions.

Value

`Spectra` object with the features ordered according to the (increasing) retention time

Author(s)

Johannes Rainer

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0001847", "HMDB0000001", "HMDB0000001"),
    name = c("Caffeine", "1-Methylhistidine", "1-Methylhistidine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876),
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16))
spd$intensity <- list(
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994),
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643))
spd$rtime <- c(15.84, 9.44, 9.44)
sps <- Spectra(spd)
MsQuality:::.rtOrderSpectra(sps)
```

areaUnderTic *area under TIC (MS:4000155)*

Description

MS:4000155
"The area under the total ion chromatogram." [PSI:MS]

The metric is calculated as follows:
(1) the input object is filtered according to the MS level,
(2) the sum of the ion counts are obtained and returned (NA values are removed by default, controlled by na.rm).

Usage

```
areaUnderTic(object, msLevel = 1L, na.rm = TRUE, ...)
```

Arguments

object	Spectra or Chromatograms object
msLevel	integer
na.rm	logical(1) whether to remove NA values before computing the sum (default TRUE)
...	additional arguments passed to internal helpers

Details

MS:4000155
is_a: MS:4000003 ! single value
relationship: has_metric_category MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000017 ! chromatogram metric

The sum of the TIC is returned as an equivalent to the area.

Value

For Spectra: numeric(1). For Chromatograms: numeric of length equal to length(object), one area per chromatogram.

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
sps <- Spectra(spd)
areaUnderTic(object = sps, msLevel = 2L)
```

`areaUnderTicRtQuantiles` *area under TIC RT quantiles (MS:4000156)*

Description

MS:4000156
"The area under the total ion chromatogram of the retention time quantiles. Number of quantiles are given by the n-tuple." [PSI:MS]

The metric is calculated as follows:
(1) the `Spectra` object is filtered according to the MS level,
(2) the `Spectra` object is ordered according to the retention time,
(3) the 0%, 25%, 50%, 75%, and 100% quantiles of the retention time values are obtained,
(4) the ion count of the intervals between the 0%/25%, 25%/50%, 50%/75%, and 75%/100% are obtained,
(5) the ion counts of the intervals are summed (TIC) and the values returned.

Usage

```
areaUnderTicRtQuantiles(object, msLevel = 1L, ...)
```

Arguments

`object`	`Spectra` or `Chromatograms` object
`msLevel`	`integer`; optional MS level filter (defaults to 1L for `Spectra`; if supplied for `Chromatograms`, data are filtered via `msLevel` before computing the metric)
`...`	not used here

Details

MS:4000156
is_a: MS:4000004 ! n-tuple
is_a: MS:4000009 ! ID free
is_a: MS:4000017 ! chromatogram metric

The sum of the TIC is returned as an equivalent to the area.

Value

For `Spectra`: a named `numeric(4)` with areas per quartile (`"25%"`, `"50%"`, `"75%"`, `"100%"`). For `Chromatograms`: a `matrix` with `nrow` equal to `length(object)` and 4 columns.

Note

This function interprets the *quantiles* from the [PSI:MS] definition as *quartiles*, i.e. the 0, 25, 50, 75 and 100% quantiles are used.

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
spd$rtime <- c(9.44, 9.44, 15.84)
sps <- Spectra(spd)
areaUnderTicRtQuantiles(object = sps, msLevel = 2L)
```

`baselineIntensity` *Baseline Intensity per chromatogram*

Description

The function 'baselineIntensity' estimates the baseline intensity for each chromatogram using a low quantile of the intensity distribution. By default the 5th percentile is used, but this can be changed via the `probs` parameter.

The metric is calculated as follows:
(1) for each chromatogram, the intensity values are extracted,
(2) the quantile at probability `probs` (default 0.05, i.e. the 5th percentile) is calculated and returned.

Usage

```
baselineIntensity(
  chromatograms,
  msLevel = integer(),
  probs = 0.05,
  na.rm = TRUE,
  ...
)
```

Arguments

`chromatograms`	'Chromatograms' object
`msLevel`	'integer' defining the MS level(s) to filter for. If empty (default), no filtering is performed.
`probs`	'numeric(1)' quantile probability (default 0.05)
`na.rm`	'logical(1)' whether to remove 'NA' values (default 'TRUE')
`...`	further arguments passed to 'quantile'

Details

In chromatograms with many zero-intensity points the chosen quantile may fall within the zero-valued portion of the distribution, causing the function to return `0`. This can propagate to downstream metrics that rely on a non-zero baseline (e.g. `peakProminence` divides by the baseline and would return `Inf`). Consider increasing `probs` (e.g. `0.10` or higher) or pre-filtering zero-intensity data points when working with zero-dominated chromatograms.

This function returns a single summary value per chromatogram. For a point-wise baseline estimate (one value per retention-time point), see `estimateBaseline` which offers SNIP, TopHat, Convex-Hull, and median methods.

Value

'numeric' of length equal to 'length(chromatograms)', one baseline intensity per chromatogram

Author(s)

Philippine Louail

Examples

```
library(Chromatograms)
cdata <- data.frame(
    msLevel = c(1L, 1L),
    mz = c(112.2, 123.3),
    dataOrigin = c("mem1", "mem1")
)
pdata <- list(
    data.frame(rtime = c(2.1, 2.5, 3.0, 3.4, 3.9),
               intensity = c(100, 250, 400, 300, 150)),
    data.frame(rtime = c(5.1, 5.8, 6.3, 6.9, 7.5),
               intensity = c(80, 500, 1200, 600, 120))
)
chr <- Chromatograms(ChromBackendMemory(), chromData = cdata, peaksData = pdata)
## Returns 5th percentile of intensity (baseline estimate)
baselineIntensity(chr)
## Use different percentile
baselineIntensity(chr, probs = 0.10)
```

`calculateMetrics` *Calculate QC metrics from a Spectra, MsExperiment, or Chromatograms object*

Description

Calculate QC metrics from a `Spectra`, `MsExperiment`, or `Chromatograms` object. `calculateMetrics` is a generic function that dispatches to the appropriate method based on the class of the input object.

Usage

```
calculateMetrics(
  object,
  metrics = qualityMetrics(object),
  filterEmptyObject = FALSE,
  ...
)
```

Arguments

`object`	`Spectra`, `MsExperiment`, or `Chromatograms` object
`metrics`	`character` specifying the quality metrics to be calculated on `object`
`filterEmptyObject`	`logical(1)` specifying if empty entries and entries with intensity zero of the `Spectra` object will be removed
`...`	arguments passed to the quality metrics functions defined in `metrics`

Details

The metrics are defined by the argument `metrics`. Further arguments passed to the quality metric functions can be specified by the `params` argument. `params` can contain named entries which are matched against the formal arguments of the quality metric functions.

Setting the argument `filterEmptyObject` to `TRUE` will remove zero-length entries, zero-intensity entries, and entries with intensities that are `Inf` from the `Spectra` object.

For `Chromatograms` objects, the argument `filterEmptyObject` is interpreted as `filterEmptyChromatograms`.

Value

`data.frame` containing in the columns the metrics for the different spectra/chromatograms and in rows the samples

Author(s)

Thomas Naake

Examples

```
library(MsDataHub)
library(Spectra)

## define file names containing spectra data for the samples
sciex_file <- c(MsDataHub::X20171016_POOL_POS_1_105.134.mzML(),
   MsDataHub::X20171016_POOL_POS_3_105.134.mzML())

## import the data and assign it to the spectra object
spectra <- Spectra(sciex_file)

## define the quality metrics to be calculated
metrics <- c("areaUnderTic", "chromatographyDuration", "msSignal10xChange")

## calculate the metrics
## additional parameters passed to the quality metrics functions
## (MsLevel is an argument of areaUnderTic and msSignal10xChange,
## relativeTo is an argument of msSignal10xChange) passed to ...
calculateMetrics(object = spectra, metrics = metrics,
    msLevel = 1, change = "jump", relativeTo = "Q1")
calculateMetrics(object = spectra, metrics = metrics,
    msLevel = 1, change = "fall", relativeTo = "previous")
```

---

calculateMetricsFromChromatograms

Calculate QC metrics from a Chromatograms object

Description

The function `calculateMetricsFromChromatograms` calculates quality metrics from a `Chromatograms` object. The function will calculate the metrics per sample according to the grouping parameter `f`, e.g. `dataOrigin` information.

Usage

```
calculateMetricsFromChromatograms(
  chromatograms,
  metrics,
  filterEmptyObject = FALSE,
```

```
  f = dataOrigin(chromatograms),
  format = c("data.frame", "mzQC"),
  ...,
  BPPARAM = bpparam()
)
```

Arguments

chromatograms	Chromatograms object
metrics	character specifying the quality metrics to be calculated on chromatograms
filterEmptyObject	logical(1) specifying if empty entries and entries with intensity zero of the Chromatograms object will be removed
f	character defining which chromatograms in chromatograms belong to one sample. Defaults to f = dataOrigin(chromatograms). Chromatograms from the same original data file are processed together (and in parallel for different files).
format	character(1) specifying if metrics are returned as a data.frame (format = "data.frame") or as a list of MzQCmzQC objects (format = "mzQC")
...	arguments passed to the quality metrics functions defined in metrics
BPPARAM	Parallel processing setup. Defaults to BPPARAM = bpparam(). See [bpparam()] for details on parallel processing with BiocParallel.

Details

The metrics are defined by the argument metrics. Further arguments passed to the quality metric functions can be specified by .... The additional arguments ... are matched against the formal arguments of the quality metric functions. Samples will be processed in parallel using the default parallel processing setup ([bpparam()]) or with the parallel processing setup defined with parameter BPPARAM.

Setting the argument filterEmptyObject to TRUE will remove zero-length entries, zero-intensity entries, and entries with intensities that are Inf from the Chromatograms object.

Value

In case of format = "data.frame", a data.frame containing in the columns the metrics for the different chromatograms of identical dataOrigin{chromatograms} (in rows). In case of format = "mzQC", a list of MzQCmzQC objects containing the metrics for the different chromatograms of identical dataOrigin{chromatograms}

Author(s)

Philippine Louail

Examples

```
library(Chromatograms)
## Create a Chromatograms object with ChromBackendMemory
cdata <- data.frame(
    msLevel = c(1L, 1L, 1L),
    mz = c(112.2, 123.3, 134.4),
    dataOrigin = c("mem1", "mem2", "mem3")
```

```
)
pdata <- list(
    data.frame(rtime = c(2.1, 2.5, 3.0, 3.4, 3.9),
               intensity = c(100, 250, 400, 300, 150)),
    data.frame(rtime = c(3.5, 4.0, 4.5),
               intensity = c(80, 120, 90)),
    data.frame(rtime = c(5.1, 5.8, 6.3, 6.9, 7.5),
               intensity = c(80, 500, 1200, 600, 120))
)
chr <- Chromatograms(ChromBackendMemory(), chromData = cdata, peaksData = pdata)
metrics <- c("chromatogramDuration", "maxIntensity", "intensityMean")
```

`calculateMetricsFromMsExperiment` *Calculate QC metrics from a MsExperiment object*

Description

The function `calculateMetricsFromMsExperiment` calculates quality metrics from a `MsExperiment` object. Each spectra in the `msexp` object should refer to one mzML file/to one sample.

Usage

```
calculateMetricsFromMsExperiment(
  msexp,
  metrics = qualityMetrics(msexp),
  filterEmptyObject = FALSE,
  ...,
  BPPARAM = bpparam()
)
```

Arguments

`msexp`	`MsExperiment` object
`metrics`	`character` specifying the quality metrics to be calculated on `msexp`
`filterEmptyObject`	`logical(1)` specifying if empty entries and entries with intensity zero of the `Spectra` object will be removed
`...`	arguments passed to the quality metrics functions defined in `metrics`
`BPPARAM`	Parallel processing setup. Defaults to `BPPARAM = bpparam()`. See [bpparam()] for details on parallel processing with `BiocParallel`.

Details

The metrics are defined by the argument `metrics`. Further arguments passed to the quality metric functions can be specified by the `params` argument. `params` can contain named entries which are matched against the formal arguments of the quality metric functions.

Setting the argument `filterEmptyObject` to `TRUE` will remove zero-length entries, zero-intensity entries, and entries with intensities that are `Inf` from the `Spectra` object.

Value

`data.frame` containing in the columns the metrics for the different spectra (in rows)

Author(s)

Thomas Naake

Examples

```
library(MsDataHub)
library(MsExperiment)
library(S4Vectors)

msexp <- MsExperiment()
sd <- DataFrame(sample_id = c("QC1", "QC2"),
    sample_name = c("QC Pool", "QC Pool"), injection_idx = c(1, 3))
sampleData(msexp) <- sd

## define file names containing spectra data for the samples and
## add them, along with other arbitrary files to the experiment
sciex_file <- unname(c(MsDataHub::X20171016_POOL_POS_1_105.134.mzML(),
    MsDataHub::X20171016_POOL_POS_3_105.134.mzML()))

## import the data and assign it to the msexp object
experimentFiles(msexp) <- MsExperimentFiles(
    mzML_files = sciex_file,
    annotations = "internal_standards.txt")

## link samples to data files: first sample to first file in "mzML_files",
## second sample to second file in "mzML_files"
msexp <- linkSampleData(msexp, with = "experimentFiles.mzML_files",
    sampleIndex = c(1, 2), withIndex = c(1, 2))
msexp <- linkSampleData(msexp, with = "experimentFiles.annotations",
     sampleIndex = c(1, 2), withIndex = c(1, 1))

library(Spectra)
## import the data and add it to the msexp object
sciex_file <- c(MsDataHub::X20171016_POOL_POS_1_105.134.mzML(),
   MsDataHub::X20171016_POOL_POS_3_105.134.mzML())

## import the data and assign it to the spectra object
spectra(msexp) <- Spectra(sciex_file)

## define the quality metrics to be calculated
metrics <- c("areaUnderTic", "chromatographyDuration", "msSignal10xChange")

## additional parameters passed to the quality metrics functions
## (msLevel is an argument of areaUnderTic and msSignal10xChange,
## relativeTo is an argument of msSignal10xChange) passed to ...
calculateMetricsFromMsExperiment(msexp = msexp, metrics = metrics,
    msLevel = 1, change = "jump", relativeTo = "Q1")

calculateMetricsFromMsExperiment(msexp = msexp, metrics = metrics,
    msLevel = 1, change = "fall", relativeTo = "previous")
```

`calculateMetricsFromOneSampleChromatograms` *Calculate QC metrics from a Chromatograms object containing only chromatographic data from one sample*

Description

The function `calculateMetricsFromOneSampleChromatograms` calculates quality metrics from a `Chromatograms` object containing chromatographic data from one sample.

Usage

```
calculateMetricsFromOneSampleChromatograms(
  chromatograms,
  metrics = qualityMetrics(chromatograms),
  filterEmptyObject = FALSE,
  f = chromatograms$dataOrigin,
  ...
)
```

Arguments

`chromatograms`	`Chromatograms` object
`metrics`	`character` specifying the quality metrics to be calculated on `chromatograms`
`filterEmptyObject`	`logical(1)` specifying if empty entries and entries with intensity zero or `Inf` of the `Chromatograms` object will be removed
`f`	`character`, grouping parameter for `chromatograms`. Defaults to `chromatograms$dataOrigin`; if missing, `NULL`, or all `NA`, all chromatograms are treated as one sample.
`...`	arguments passed to the quality metrics functions defined in `metrics`

Details

The metrics are defined by the argument `metrics`. Further arguments passed to the quality metric functions can be specified by the `params` argument. `params` can contain named entries which are matched against the formal arguments of the quality metric functions.

The `Chromatograms` object will only contain chromatographic data from one data origin (e.g. `object$dataOrigin` is of length 1). The grouping is specified by the argument `f`.

Setting the argument `filterEmptyObject` to `TRUE` will remove zero-length entries and entries with intensities that are `Inf` from the `Chromatograms` object.

Each metric returns a per-chromatogram result: scalar metrics return a `numeric` vector of `length(chromatograms)`, while multi-value metrics return a `matrix` with `nrow = length(chromatograms)`.

Value

named `list`. Each element is named by the metric and contains either a `numeric` vector or a `matrix` (for multi-value metrics).

Author(s)

Philippine Louail

Examples

```
library(Chromatograms)
## Create a Chromatograms object with ChromBackendMemory
cdata <- data.frame(
    msLevel = c(1L, 1L, 1L),
    mz = c(112.2, 123.3, 134.4),
    dataOrigin = c("mem1", "mem2", "mem3")
)
pdata <- list(
    data.frame(rtime = c(2.1, 2.5, 3.0, 3.4, 3.9),
               intensity = c(100, 250, 400, 300, 150)),
    data.frame(rtime = c(3.5, 4.0, 4.5),
               intensity = c(80, 120, 90)),
    data.frame(rtime = c(5.1, 5.8, 6.3, 6.9, 7.5),
               intensity = c(80, 500, 1200, 600, 120))
)
chr <- Chromatograms(ChromBackendMemory(), chromData = cdata, peaksData = pdata)
metrics <- c("chromatogramDuration", "maxIntensity", "intensityMean")
```

---

calculateMetricsFromOneSampleSpectra

Calculate QC metrics from a Spectra object containing only spectral data from one sample

---

Description

The function `calculateMetricsFromOneSampleSpectra` calculates quality metrics from a `Spectra` containing spectral data from one sample.

Usage

```
calculateMetricsFromOneSampleSpectra(
  spectra,
  metrics = qualityMetrics(spectra),
  filterEmptyObject = FALSE,
  f = spectra$dataOrigin,
  ...
)
```

Arguments

`spectra`	`Spectra` object
`metrics`	`character` specifying the quality metrics to be calculated on `spectra`
`filterEmptyObject`	`logical(1)` specifying if empty entries and entries with intensity zero or `Inf` of the `Spectra` object will be removed
`f`	`character`, grouping parameter for `spectra`
`...`	arguments passed to the quality metrics functions defined in `metrics`

Details

The metrics are defined by the argument `metrics`. Further arguments passed to the quality metric functions can be specified by the `params` argument. `params` can contain named entries which are matched against the formal arguments of the quality metric functions.

The `Spectra` object will only contain spectral data from one data origin (e.g. `spectra$dataOrigin` is of length 1). The grouping is specified by the argument `f`.

Setting the argument `filterEmptyObject` to `TRUE` will remove zero-length entries, zero-intensity entries, and entries with intensities that are `Inf` from the `Spectra` object.

Value

named `numeric` vector

Author(s)

Thomas Naake

Examples

```
library(MsDataHub)
library(Spectra)

## define file names containing spectra data for the samples
sciex_file <- c(MsDataHub::X20171016_POOL_POS_1_105.134.mzML())

## import the data and assign it to the spectra object
spectra <- Spectra(sciex_file)

## define the quality metrics to be calculated
metrics <- c("areaUnderTic", "chromatographyDuration", "msSignal10xChange")

## calculate the metrics
## additional parameters B to the quality metrics functions
## (MsLevel is an argument of areaUnderTic and msSignal10xChange,
## relativeTo is an argument of msSignal10xChange) passed to ...
MsQuality:::calculateMetricsFromOneSampleSpectra(spectra = spectra,
    metrics = metrics, msLevel = 1, change = "jump", relativeTo = "Q1")
MsQuality:::calculateMetricsFromOneSampleSpectra(spectra = spectra,
    metrics = metrics, msLevel = 1, change = "fall", relativeTo = "previous")
```

`calculateMetricsFromSpectra`

Calculate QC metrics from a Spectra object

Description

The function `calculateMetricsFromSpectra` calculates quality metrics from a `Spectra` object. The function will calculate the metrics per sample according to the grouping parameter `f`, e.g. `dataOrigin` information.

Two format options are available:

- `format = "data.frame"` returns the metrics as a `data.frame`,
- `format = "mzQC"` returns the metrics as a list of `MzQCmzQC` objects.

Usage

```
calculateMetricsFromSpectra(
  spectra,
  metrics,
  filterEmptyObject = FALSE,
  f = dataOrigin(spectra),
  format = c("data.frame", "mzQC"),
  ...,
  BPPARAM = bpparam()
)
```

Arguments

spectra	Spectra object
metrics	character specifying the quality metrics to be calculated on spectra
filterEmptyObject	logical(1) specifying if empty entries and entries with intensity zero of the Spectra object will be removed
f	character defining which spectra in spectra belong to one sample. Defaults to f = dataOrigin(spectra). Spectra from the same original data file are processed together (and in parallel for different files).
format	character(1) specifying if metrics are returned as a data.frame (format = "data.frame") or as a list of MzQCmzQC objects (format = "mzQC")
...	arguments passed to the quality metrics functions defined in metrics
BPPARAM	Parallel processing setup. Defaults to BPPARAM = bpparam(). See [bpparam()] for details on parallel processing with BiocParallel.

Details

The metrics are defined by the argument metrics. Further arguments passed to the quality metric functions can be specified by .... The additional arguments ... are matched against the formal arguments of the quality metric functions.

Samples will be processed in parallel using the default parallel processing setup ([bpparam()]) or with the parallel processing setup defined with parameter BPPARAM.

Setting the argument filterEmptyObject to TRUE will remove zero-length entries, zero-intensity entries, and entries with intensities that are Inf from the Spectra object.

Value

In case of format = "data.frame", a data.frame containing in the columns the metrics for the different spectra of identical dataOrigin{spectra} (in rows). In case of format = "mzQC", a list of MzQCmzQC objects containing the metrics for the different spectra of identical dataOrigin{spectra}

Author(s)

Thomas Naake, Johannes Rainer

Examples

```
library(MsDataHub)
library(Spectra)

## define file names containing spectra data for the samples
sciex_file <- c(MsDataHub::X20171016_POOL_POS_1_105.134.mzML(),
   MsDataHub::X20171016_POOL_POS_3_105.134.mzML())

## import the data and assign it to the spectra object
spectra <- Spectra(sciex_file)

## define the quality metrics to be calculated
metrics <- c("areaUnderTic", "chromatographyDuration", "msSignal10xChange")

## calculate the metrics
## additional parameters passed to the quality metrics functions
## (msLevel is an argument of areaUnderTic and msSignal10xChange,
## relativeTo is an argument of msSignal10xChange) passed to ...

## format = "data.frame"
calculateMetricsFromSpectra(spectra = spectra, metrics = metrics,
    format = "data.frame", msLevel = 1, change = "jump", relativeTo = "Q1")
calculateMetricsFromSpectra(spectra = spectra, metrics = metrics,
    format = "data.frame", msLevel = 1, change = "fall",
    relativeTo = "previous")

## format = "mzQC"
calculateMetricsFromSpectra(spectra = spectra, metrics = metrics,
    format = "mzQC", msLevel = 1, change = "jump", relativeTo = "Q1")
calculateMetricsFromSpectra(spectra = spectra, metrics = metrics,
    format = "mzQC", msLevel = 1, change = "fall", relativeTo = "previous")
```

---

chromatographyDuration

chromatography duration (MS:4000053)

Description

MS:4000053 "The retention time duration of the chromatography in seconds." [PSI:MS]

The metric is calculated as follows:
(1) the retention time associated to the `Spectra` object is obtained,
(2) the maximum and the minimum of the retention time is obtained,
(3) the difference between the maximum and the minimum is calculated and returned.

Usage

```
chromatographyDuration(object, ...)
```

Arguments

object	Spectra or Chromatograms object
...	not used here

Details

MS:4000053 synonym: "RT-Duration" RELATED [PMID:24494671]
is_a: MS:4000003 ! single value
relationship: has_metric_category MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000016 ! retention time metric
relationship: has_value_type xsd:float ! The allowed value-type for this CV term
relationship: has_value_concept NCIT:C25330 ! Duration
relationship: has_units UO:0000010 ! second

Retention time values that are NA are removed.

Value

For Spectra: numeric(1). For Chromatograms: numeric of length equal to length(object), one duration per chromatogram.

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
spd$rtime <- c(9.44, 9.44, 15.84)
sps <- Spectra(spd)
chromatographyDuration(object = sps)
```

`extentIdentifiedPrecursorIntensity` *extent of identified MS2 precursor intensity (MS:4000157)*

Description

MS:4000157
"Ratio of 95th over 5th percentile of MS2 precursor intensity for all quantification data points after user-defined acceptance criteria are applied. Can be used to approximate the dynamic range of signal. The used type of identification should be noted in the metadata or analysis methods section of the recording file for the respective run. In case of multiple acceptance criteria (FDR) available in proteomics, PSM-level FDR should be used for better comparability." [PSI:MS]

The metric is calculated as follows:
(1) the `Spectra` object is filtered according to the MS level,
(2) the intensities of the precursor ions are obtained,
(3) the 5% and 95% quantile of these intensities are obtained (`NA` values are removed),
(4) the ratio between the 95% and the 5% intensity quantile is calculated and returned.

Usage

```
extentIdentifiedPrecursorIntensity(
  spectra,
  msLevel = 1L,
  identificationLevel = c("all", "identified", "unidentified"),
  ...
)
```

Arguments

`spectra`	`Spectra` object
`msLevel`	`integer`
`identificationLevel`	`character(1)`, one of `"all"`, `"identified"`, or `"unidentified"`
`...`	not used here

Details

MS:4000157
is_a: MS:4000001 ! QC metric
is_a: MS:4000003 ! single value
is_a: MS:4000008 ! ID based
relationship: has_metric_category MS:4000022 ! MS2 metric
synonym: "MS1-3A" RELATED [PMID:19837981]

Precursor intensity values that are `NA` are removed.

An attribute containing the PSI:MS term will only be returned if `identificationLevel` is `"identified"`.

Value

numeric(1)

Note

The Spectra object might contain features that were not identified. If the calculation needs to be done according to *MS:4000157*, the Spectra object should be prepared accordingly, i.e. being subsetted to spectra with identification data.

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
spd$precursorIntensity <- c(100, 100, 100)
sps <- Spectra(spd)
extentIdentifiedPrecursorIntensity(spectra = sps, msLevel = 2L)
```

---

gaussianSimilarity *Gaussian Similarity (Peak Shape Quality) per Chromatogram*

Description

The function ‘gaussianSimilarity‘ assesses the shape quality of each chromatographic peak by fitting a beta-distribution curve via MetaboCoreUtils::betaValues() and returning two diagnostics.

The metric is calculated as follows (per chromatogram):
(1) the retention time and intensity values are extracted,
(2) if not provided, peak boundaries are determined using peakBoundary(),
(3) the data is subset to the peak region defined by the boundaries,
(4) MetaboCoreUtils::betaValues() is called on the peak region with a set of skew parameters (shape1 = 3, 3.5, 4, 4.5, 5, shape2 = 5),

(5) two values are returned: `gaussian_similarity` (maximum correlation between the observed peak and the best-fitting beta curve; values close to 1 indicate a symmetric, bell-shaped peak) and `gaussian_residuals` (standard deviation of the residuals after normalising and subtracting the best-fit curve; lower values indicate a cleaner peak shape).

Usage

```
gaussianSimilarity(
  chromatograms,
  msLevel = integer(),
  peakBoundary = NULL,
  ...
)
```

Arguments

`chromatograms`	'Chromatograms' object
`msLevel`	'integer' defining the MS level(s) to filter for. If empty (default), no filtering is performed.
`peakBoundary`	optional peak boundary: either a 'matrix' with columns 'left_boundary' and 'right_boundary' (one row per chromatogram), or a 'numeric(2)' named vector applied to all chromatograms.
`...`	further arguments (currently unused)

Details

The underlying `betaValues()` function (Kumler et al. 2023) compares the observed chromatographic peak to a family of beta-distribution curves of varying skew. The first returned value is the Pearson correlation between the observed and best-fit curve, and the second is the standard deviation of the normalised residuals.

Requires at least 5 data points within the peak region.

Value

'matrix' with 'nrow' equal to 'length(chromatograms)' and 2 columns ('gaussian_similarity', 'gaussian_residuals'). Returns 'NA' for chromatograms where values cannot be calculated.

Author(s)

William Kumler, Philippine Louail

References

Kumler W, Hazelton B J and Ingalls A E (2023) "Picky with peakpicking: assessing chromatographic peak quality with simple metrics in metabolomics" BMC Bioinformatics 24(1):404. doi: 10.1186/s12859-023-05533-4

Examples

```
library(Chromatograms)
cdata <- data.frame(
    msLevel = c(1L, 1L),
    mz = c(100.0, 200.0),
```

```
    dataOrigin = c("mem1", "mem1")
)
pdata <- list(
    data.frame(rtime = seq(1, 20, by = 0.5),
               intensity = c(10, 15, 25, 50, 100, 200, 350, 500, 650,
                             700, 650, 500, 350, 200, 100, 50, 25, 15,
                             10, 8, 6, 5, 4, 3, 2, 2, 1, 1, 1, 1, 1,
                             1, 1, 1, 1, 1, 1, 1, 1)),
    data.frame(rtime = c(1, 2, 3, 4, 5, 6, 7),
               intensity = c(0, 10, 50, 100, 50, 10, 0))
)
chr <- Chromatograms(ChromBackendMemory(), chromData = cdata, peaksData = pdata)
gaussianSimilarity(chr)
```

`intensityMean` *Intensity statistics (Mean) per chromatogram*

Description

The function ‘intensityMean‘ calculates the mean of intensity values within each chromatogram.

The metric is calculated as follows:
(1) for each chromatogram in the ‘Chromatograms‘ object, the intensity values are extracted,
(2) the arithmetic mean per chromatogram is calculated and returned.

Usage

```
intensityMean(chromatograms, msLevel = integer(), na.rm = TRUE, ...)
```

Arguments

`chromatograms`	‘Chromatograms‘ object
`msLevel`	‘integer‘ defining the MS level(s) to filter for. If empty (default), no filtering is performed.
`na.rm`	‘logical(1)‘ whether to remove ‘NA‘ values (default ‘TRUE‘)
`...`	further arguments passed to ‘mean‘

Value

‘numeric‘ of length equal to ‘length(chromatograms)‘

Author(s)

Philippine Louail

Examples

```
library(Chromatograms)
cdata <- data.frame(
    msLevel = c(1L, 1L),
    mz = c(112.2, 123.3),
    dataOrigin = c("mem1", "mem1")
)
pdata <- list(
    data.frame(rtime = c(2.1, 2.5, 3.0, 3.4, 3.9),
               intensity = c(100, 250, 400, 300, 150)),
    data.frame(rtime = c(5.1, 5.8, 6.3, 6.9, 7.5),
               intensity = c(80, 500, 1200, 600, 120))
)
chr <- Chromatograms(ChromBackendMemory(), chromData = cdata, peaksData = pdata)
## Returns mean intensity per chromatogram
intensityMean(chr)
```

---

intensityQuartiles *Intensity Quartiles per chromatogram*

Description

The function ‘intensityQuartiles‘ calculates the first, second, and third quartile (Q1, Q2, Q3) of intensity values within each chromatogram.

The metric is calculated as follows:
(1) for each chromatogram, the intensity values are extracted,
(2) the quartiles Q1 (25th), Q2 (50th, median), and Q3 (75th) are calculated and returned as a row of the result matrix.

Usage

```
intensityQuartiles(chromatograms, msLevel = integer(), ...)
```

Arguments

chromatograms	‘Chromatograms‘ object
msLevel	‘integer‘ defining the MS level(s) to filter for. If empty (default), no filtering is performed.
...	further arguments passed to ‘quantile‘

Value

‘matrix‘ with ‘nrow‘ equal to ‘length(chromatograms)‘ and 3 columns (Q1, Q2, Q3)

Author(s)

Philippine Louail

Examples

```
library(Chromatograms)
cdata <- data.frame(
    msLevel = c(1L, 1L),
    mz = c(112.2, 123.3),
    dataOrigin = c("mem1", "mem1")
)
pdata <- list(
    data.frame(rtime = c(2.1, 2.5, 3.0, 3.4, 3.9),
               intensity = c(100, 250, 400, 300, 150)),
    data.frame(rtime = c(5.1, 5.8, 6.3, 6.9, 7.5),
               intensity = c(80, 500, 1200, 600, 120))
)
chr <- Chromatograms(ChromBackendMemory(), chromData = cdata, peaksData = pdata)
## Returns a 2x3 matrix with Q1, Q2, Q3 per chromatogram
intensityQuartiles(chr)
```

`intensityRange` *Intensity Range per chromatogram*

Description

The function ‘intensityRange‘ calculates the range (min, max) of intensity values within each chromatogram.

The metric is calculated as follows:
(1) for each chromatogram, the intensity values are extracted,
(2) the minimum and maximum values per chromatogram are obtained and returned as a row of the result matrix.

Usage

```
intensityRange(chromatograms, msLevel = integer(), na.rm = TRUE, ...)
```

Arguments

`chromatograms`	‘Chromatograms‘ object
`msLevel`	‘integer‘ defining the MS level(s) to filter for. If empty (default), no filtering is performed.
`na.rm`	‘logical(1)‘ whether to remove ‘NA‘ values (default ‘TRUE‘)
`...`	further arguments passed to ‘range‘

Value

‘matrix‘ with ‘nrow‘ equal to ‘length(chromatograms)‘ and 2 columns (min, max)

Author(s)

Philippine Louail

Examples

```
library(Chromatograms)
cdata <- data.frame(
    msLevel = c(1L, 1L),
    mz = c(112.2, 123.3),
    dataOrigin = c("mem1", "mem1")
)
pdata <- list(
    data.frame(rtime = c(2.1, 2.5, 3.0, 3.4, 3.9),
               intensity = c(100, 250, 400, 300, 150)),
    data.frame(rtime = c(5.1, 5.8, 6.3, 6.9, 7.5),
               intensity = c(80, 500, 1200, 600, 120))
)
chr <- Chromatograms(ChromBackendMemory(), chromData = cdata, peaksData = pdata)
## Returns a 2x2 matrix with (min, max) per chromatogram
intensityRange(chr)
```

---

intensitySd *Intensity statistics (Standard Deviation) per chromatogram*

Description

The function ‘intensitySd‘ calculates the standard deviation of intensity values within each chromatogram.

The metric is calculated as follows:
(1) for each chromatogram in the ‘Chromatograms‘ object, the intensity values are extracted,
(2) the standard deviation per chromatogram is calculated and returned.

Usage

```
intensitySd(chromatograms, msLevel = integer(), na.rm = TRUE, ...)
```

Arguments

chromatograms	‘Chromatograms‘ object
msLevel	‘integer‘ defining the MS level(s) to filter for. If empty (default), no filtering is performed.
na.rm	‘logical(1)‘ whether to remove ‘NA‘ values (default ‘TRUE‘)
...	further arguments passed to ‘sd‘

Value

‘numeric‘ of length equal to ‘length(chromatograms)‘

Author(s)

Philippine Louail

Examples

```
library(Chromatograms)
cdata <- data.frame(
    msLevel = c(1L, 1L),
    mz = c(112.2, 123.3),
    dataOrigin = c("mem1", "mem1")
)
pdata <- list(
    data.frame(rtime = c(2.1, 2.5, 3.0, 3.4, 3.9),
               intensity = c(100, 250, 400, 300, 150)),
    data.frame(rtime = c(5.1, 5.8, 6.3, 6.9, 7.5),
               intensity = c(80, 500, 1200, 600, 120))
)
chr <- Chromatograms(ChromBackendMemory(), chromData = cdata, peaksData = pdata)
## Returns standard deviation of intensities per chromatogram
intensitySd(chr)
```

Lee_2019 *Example data for* MsQuality*: data set of Lee et al. (2019)*

Description

The data set of Lee et al. (2019) contains metabolite information measured by reverse phase liquid chromatography (RPLC) coupled to mass spectrometry and hydrophilic interaction liquid chromatography (HILIC) coupled to mass spectrometry (file 'STables - rev1.xlsx' in the Supplementary Information).

It will be used as an example data set in the vignette to show the functionality of the packages. The file contains Spectra and MsExperiment objects that store the mass spectrometry data.

Format

Spectra and MsExperiment

Value

Spectra and MsExperiment objects

Author(s)

Thomas Naake, <thomasnaake@googlemail.com>

Source

See the file Lee2019-data-source.R in scripts for the source code how sps_hilic and sps_rplc were created.

References

Lee et al. (2019). A large-scale analysis of targeted metabolomics data from heterogeneous biological samples provides insights into metabolite dynamics. Metabolomics, 103, doi: 10.1007/s11306-019-1564-8.

Lee_2019_meta_vals *Example data for* MsQuality*: data set of Lee et al. (2019)*

Description

The data set of Lee et al. (2019) contains metabolite information measured by reverse phase liquid chromatography (RPLC) coupled to mass spectrometry and hydrophilic interaction liquid chromatography (HILIC) coupled to mass spectrometry (file 'STables - rev1.xlsx' in the Supplementary Information). The xlsx sheets 'Methods' and 'Raw data' were stored as txt files.

Lee_2019_meta_vals contains two data frame objects: one containing information on metabolite meta-data and one containing intensity values on metabolites. The object will be used as an example data set in the vignette to show the functionality of the packages.

Format

data.frame

Value

data.frame

Author(s)

Thomas Naake, <thomasnaake@googlemail.com>

Source

path_to_meta <- "Lee_et_al_2019_Stables_rev1_Methods.txt" meta <- read.delim(path_to_meta, dec = ".", header = TRUE)

print number of metabolites per measurement (meta data) table(meta$Method)

path_to_vals <- "Lee_et_al_2019_Stables_rev1_Raw_data.txt" vals <- read.delim(path_to_vals, dec = ".", header = TRUE)

print number of metabolites per measurement (intensity data) table(grepl(vals$Metabolite, pattern = "_rp$")) table(grepl(vals$Metabolite, pattern = "_hn$"))

save the two objects as an RData object save(meta, vals, file = "Lee_2019_meta_vals.RData", compress = "xz")

References

Lee et al. (2019). A large-scale analysis of targeted metabolomics data from heterogeneous biological samples provides insights into metabolite dynamics. Metabolomics, 103, doi: 10.1007/s11306-019-1564-8.

`maxIntensity` *Maximum intensity per chromatogram*

Description

The function ‘maxIntensity‘ returns the maximum intensity value observed within each chromatogram.

The metric is calculated as follows:
(1) for each chromatogram in the ‘Chromatograms‘ object, the intensity values are extracted,
(2) the maximum value per chromatogram is obtained and returned.

Usage

```
maxIntensity(chromatograms, msLevel = integer(), na.rm = TRUE, ...)
```

Arguments

`chromatograms`	‘Chromatograms‘ object
`msLevel`	‘integer‘ defining the MS level(s) to filter for. If empty (default), no filtering is performed.
`na.rm`	‘logical(1)‘ whether to remove ‘NA‘ values (default ‘TRUE‘)
`...`	further arguments passed to ‘max‘

Value

‘numeric‘ of length equal to ‘length(chromatograms)‘, one maximum intensity value per chromatogram

Author(s)

Philippine Louail

Examples

```
library(Chromatograms)
cdata <- data.frame(
    msLevel = c(1L, 1L),
    mz = c(112.2, 123.3),
    dataOrigin = c("mem1", "mem1")
)
pdata <- list(
    data.frame(rtime = c(2.1, 2.5, 3.0, 3.4, 3.9),
               intensity = c(100, 250, 400, 300, 150)),
    data.frame(rtime = c(5.1, 5.8, 6.3, 6.9, 7.5),
               intensity = c(80, 500, 1200, 600, 120))
)
chr <- Chromatograms(ChromBackendMemory(), chromData = cdata, peaksData = pdata)
## Returns max intensity per chromatogram: c(400, 1200)
maxIntensity(chr)
```

`meanCharge` *mean MS2 precursor charge in all spectra (MS:4000173) or mean MS2 precursor charge in identified spectra (MS:4000174)*

Description

MS:4000173
"Mean MS2 precursor charge in all spectra" [PSI:MS]

MS:4000174
"Mean MS2 precursor charge in identified spectra. The used type of identification should be noted in the metadata or analysis methods section of the recording file for the respective run. In case of multiple acceptance criteria (FDR) available in proteomics, PSM-level FDR should be used for better comparability." [PSI:MS]

The metric is calculated as follows:
(1) the `Spectra` object is filtered according to the MS level,
(2) the precursor charge is obtained,
(3) the mean of the precursor charge values is calculated and returned.

Usage

```
meanCharge(
  spectra,
  msLevel = 1L,
  identificationLevel = c("all", "identified", "unidentified"),
  ...
)
```

Arguments

`spectra`	`Spectra` object
`msLevel`	`integer`
`identificationLevel`	`character(1)`, one of `"all"`, `"identified"`, or `"unidentified"`
`...`	not used here

Details

MS:4000173
is_a: MS:4000001 ! QC metric is_a: MS:4000003 ! single value is_a: MS:4000009 ! ID free metric relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000020 ! ion source metric
relationship: has_metric_category MS:4000022 ! MS2 metric
synonym: "MS2 known precursor charges fractions" RELATED []
synonym: "MS2-PrecZ-1" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-2" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-3" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-4" RELATED [PMID:24494671]

synonym: "MS2-PrecZ-5" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-more" RELATED [PMID:24494671]

MS:4000174
is_a: MS:4000001 ! QC metric
is_a: MS:4000003 ! single value
is_a: MS:4000008 ! ID based
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000020 ! ion source metric
relationship: has_metric_category MS:4000022 ! MS2 metric
synonym: "MS2 known precursor charges fractions" RELATED []
synonym: "MS2-PrecZ-1" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-2" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-3" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-4" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-5" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-more" RELATED [PMID:24494671]

An attribute containing the PSI:MS term will only be returned if `identificationLevel` is either `"all"` or `"identified"`.

Value

`numeric(1)`

Note

The `Spectra` object might contain features that were not identified. If the calculation needs to be done according to *MS:4000174*, the `Spectra` object should be prepared accordingly.

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
```

```
spd$precursorCharge <- c(1L, 1L, 1L)
sps <- Spectra(spd)
meanCharge(spectra = sps, msLevel = 2L)
```

`medianCharge` *median MS2 precursor charge in all spectra (MS:4000175) or median MS2 precursor charge in identified spectra (MS:4000176)*

Description

MS:4000175
"Median MS2 precursor charge in all spectra" [PSI:MS]

MS:4000176
"Median MS2 precursor charge in identified spectra. The used type of identification should be noted in the metadata or analysis methods section of the recording file for the respective run. In case of multiple acceptance criteria (FDR) available in proteomics, PSM-level FDR should be used for better comparability." [PSI:MS]

The metric is calculated as follows:
(1) the `Spectra` object is filtered according to the MS level,
(2) the precursor charge is obtained,
(3) the median of the precursor charge values is calculated and returned.

Usage

```
medianCharge(
  spectra,
  msLevel = 1L,
  identificationLevel = c("all", "identified", "unidentified"),
  ...
)
```

Arguments

`spectra`	`Spectra` object
`msLevel`	`integer`
`identificationLevel`	`character(1)`, one of `"all"`, `"identified"`, or `"unidentified"`
`...`	not used here

Details

MS:4000175
is_a: MS:4000001 ! QC metric
is_a: MS:4000003 ! single value
is_a: MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000020 ! ion source metric
relationship: has_metric_category MS:4000022 ! MS2 metric

synonym: "MS2 known precursor charges fractions" RELATED []
synonym: "MS2-PrecZ-1" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-2" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-3" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-4" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-5" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-more" RELATED [PMID:24494671]

MS:4000176
is_a: MS:4000001 ! QC metric
is_a: MS:4000003 ! single value
is_a: MS:4000008 ! ID based
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000020 ! ion source metric
relationship: has_metric_category MS:4000022 ! MS2 metric
synonym: "MS2 known precursor charges fractions" RELATED []
synonym: "MS2-PrecZ-1" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-2" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-3" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-4" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-5" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-more" RELATED [PMID:24494671]

An attribute containing the PSI:MS term will only be returned if `identificationLevel` is either `"all"` or `"identified"`.

Value

`numeric(1)`

Note

The `Spectra` object might contain features that were not identified. If the calculation needs to be done according to *MS:4000176*, the `Spectra` object should be prepared accordingly.

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
```

```
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
sps <- Spectra(spd)
spd$precursorCharge <- c(1L, 1L, 1L)
medianCharge(spectra = sps, msLevel = 2L)
```

`medianPrecursorMz` *MS2 precursor median m/z of identified quantification data points (MS:4000152)*

Description

MS:4000152
"Median m/z value for MS2 precursors of all quantification data points after user-defined acceptance criteria are applied. These data points may be for example XIC profiles, isotopic pattern areas, or reporter ions (see MS:1001805). The used type should be noted in the metadata or analysis methods section of the recording file for the respective run. In case of multiple acceptance criteria (FDR) available in proteomics, PSM-level FDR should be used for better comparability." [PSI:MS]

The metric is calculated as follows:
(1) the `Spectra` object is filtered according to the MS level,
(2) the precursor m/z values are obtained,
(3) the median value is returned (`NA`s are removed).

Usage

```
medianPrecursorMz(
  spectra,
  msLevel = 1L,
  identificationLevel = c("all", "identified", "unidentified"),
  ...
)
```

Arguments

`spectra`	`Spectra` object
`msLevel`	`integer`
`identificationLevel`	`character(1)`, one of `"all"`, `"identified"`, or `"unidentified"`
`...`	not used here

Details

MS:4000152
is_a: MS:4000003 ! single value
is_a: MS:4000008 ! ID based
is_a: MS:4000020 ! ion source metric

relationship: has_metric_category MS:4000022 ! MS2 metric
relationship: has_units MS:1000040 ! m/z

An attribute containing the PSI:MS term will only be returned if `identificationLevel` is `"identified"` and `msLevel` is 1.

Value

`numeric(1)`

Note

`medianPrecursorMz` will calculate the *precursor* median m/z of all Spectra within `spectra`. If the calculation needs be done according to *MS:4000152*, the `Spectra` object should be prepared accordingly, i.e. filtered with e.g. [filterPrecursorMz()] or subsetted to spectra with identification data.

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
spd$precursorMz <- c(170.16, 170.16, 195.0876)
sps <- Spectra(spd)
medianPrecursorMz(spectra = sps, msLevel = 2L)
```

`medianTicOfRtRange` *median of TIC values in the shortest RT range in which half of the quantification data points are identified (MS:4000159)*

Description

MS:4000159
"Median of TIC values in the shortest RT range in which half of the quantification data points are identified. These data points may be for example XIC profiles, isotopic pattern areas, or reporter ions (see MS:1001805). The used type should be noted in the metadata or analysis methods section of the recording file for the respective run. In case of multiple acceptance criteria (FDR) available in proteomics, PSM-level FDR should be used for better comparability." [PSI:MS]

The metric is calculated as follows:
(1) the `Spectra` object is filtered according to the MS level,
(2) the `Spectra` object is ordered according to the retention time,
(3) the number of features in the `Spectra` object is obtained and the number for half of the features is calculated,
(4) iterate through the features (always by taking the neighbouring half of features) and calculate the retention time range of the set of features,
(5) retrieve the set of features with the minimum retention time range,
(6) calculate from the set of (5) the median TIC (`NA` values are removed) and return it.

Usage

```
medianTicOfRtRange(
  spectra,
  msLevel = 1L,
  identificationLevel = c("all", "identified", "unidentified"),
  ...
)
```

Arguments

`spectra`	`Spectra` object
`msLevel`	`integer`
`identificationLevel`	`character(1)`, one of `"all"`, `"identified"`, or `"unidentified"`
`...`	not used here

Details

MS:4000159
is_a: MS:4000001 ! QC metric
is_a: MS:4000003 ! single value
is_a: MS:4000008 ! ID based
synonym: "MS1-2B" RELATED [PMID:19837981]

The function `medianTicOfRtRange` uses the function `ionCount` as an equivalent to the TIC.

An attribute containing the PSI:MS term will only be returned if `identificationLevel` is `"identified"`.

Value

`numeric(1)`

Note

The Spectra object might contain features that were not identified. If the calculation needs to be done according to *MS:4000159*, the Spectra object should be prepared accordingly, i.e. being subsetted to spectra with identification data.

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
spd$rtime <- c(9.44, 9.44, 15.84)
sps <- Spectra(spd)
medianTicOfRtRange(spectra = sps, msLevel = 2L)
```

---

medianTicRtIqr *median of TIC values in the RT range in which the middle half of quantification data points are identified (MS:4000158)*

Description

MS:4000158
"Median of TIC values in the RT range in which half of quantification data points are identified (RT values of Q1 to Q3 of identifications). These data points may be for example XIC profiles, isotopic pattern areas, or reporter ions (see MS:1001805). The used type should be noted in the metadata or analysis methods section of the recording file for the respective run. In case of multiple acceptance criteria (FDR) available in proteomics, PSM-level FDR should be used for better comparability." [PSI:MS]

The metric is calculated as follows:
(1) the Spectra object is filtered according to the MS level,
(2) the Spectra object is ordered according to the retention time,
(3) the features between the 1st and 3rd quartile are obtained (half of the features that are present in the Spectra object),

(4) the ion count of the features within the 1st and 3rd quartile is obtained,
(5) the median value of the ion count is calculated (NA values are removed) and the median value is returned.

Usage

```
medianTicRtIqr(
  object,
  msLevel = 1L,
  identificationLevel = c("all", "identified", "unidentified"),
  ...
)
```

Arguments

object	Spectra or Chromatograms object
msLevel	integer
identificationLevel	character(1), one of "all", "identified", or "unidentified"
...	not used here

Details

MS:4000158
is_a: MS:4000001 ! QC metric
is_a: MS:4000003 ! single value
is_a: MS:4000008 ! ID based

The function medianTicRtIqr uses the function [ionCount()] as an equivalent to the TIC.

An attribute containing the PSI:MS term will only be returned if identificationLevel is "identified".

Value

numeric(1)

Note

The Spectra object might contain features that were not identified. If the calculation needs to be done according to *MS:4000158*, the Spectra object should be prepared accordingly, i.e. being subsetted to spectra with identification data.

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
```

```
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
spd$rtime <- c(9.44, 9.44, 15.84)
sps <- Spectra(spd)
medianTicRtIqr(object = sps, msLevel = 2L)
```

`msSignal10xChange` *MS1 signal jump (10x) count (MS:4000097) or MS1 signal fall (10x) count (MS:4000098)*

Description

MS:4000097
"The number of times where MS1 TIC increased more than 10-fold between adjacent MS1 scans. An unusual high count of signal jumps or falls can indicate ESI stability issues." [PSI:MS]

MS:4000098
"The number of times where MS1 TIC decreased more than 10-fold between adjacent MS1 scans. An unusual high count of signal jumps or falls can indicate ESI stability issues." [PSI:MS]

The metric is calculated as follows:
(1) the `Spectra` object is filtered according to the MS level,
(2) the intensity of the precursor ions within the `Spectra` object are obtained,
(3) the intensity values of the features are obtained via the ion count,
(4) the signal jumps/declines of the intensity values with the two subsequent intensity values is calculated,
(5) in the case of *MS:4000097*, the signal jumps by a factor of ten or more are counted and returned;
in the case of *MS:4000098*, the signal declines by a factor of ten or more are counted and returned.

Usage

```
msSignal10xChange(object, change = "jump", msLevel = 1L, minIntensity = 0, ...)
```

Arguments

`object`	`Spectra` or `Chromatograms` object
`change`	`character(1)`, one of `"jump"` or `"fall"`
`msLevel`	`integer`

`minIntensity`	`numeric(1)` minimum intensity threshold for `Chromatograms` input. Data points with intensity below this value are removed before computing ratios (default `0`, i.e. only zero and negative values are removed).
`...`	not used here

Details

MS:4000097
is_a: MS:4000003 ! single value
relationship: has_metric_category MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000021 ! MS1 metric
relationship: has_units UO:0000189 ! count unit
relationship: has_value_type xsd:integer ! The allowed value-type for this CV term
synonym: "IS-1A" RELATED []

MS:4000098
is_a: MS:4000003 ! single value
relationship: has_metric_category MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000021 ! MS1 metric
relationship: has_units UO:0000189 ! count unit
relationship: has_value_type xsd:integer ! The allowed value-type for this CV term
synonym: "IS-1B" RELATED []

The function `msSignal10xChange` uses the function `ionCount` as an equivalent to the TIC.

For `Chromatograms` objects the metric operates on raw point-by-point intensities rather than aggregated TIC values. This makes it sensitive to noise: small oscillations near zero can produce many spurious 10-fold jumps or falls. The `minIntensity` parameter (default `0`) allows filtering out low-intensity points before computing ratios. Increasing `minIntensity` to the expected noise floor (e.g. the baseline intensity) is recommended for noisy chromatograms. Pre-smoothing the chromatogram before calling this function is another option.

An attribute containing the PSI:MS term will only be returned if `msLevel` is 1.

Value

`numeric(1)`

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
```

```
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
spd$rtime <- c(9.44, 9.44, 15.84)
sps <- Spectra(spd)
msSignal10xChange(object = sps, change = "jump", msLevel = 2L)
msSignal10xChange(object = sps, change = "fall", msLevel = 2L)
```

`mzAcquisitionRange` *m/z acquisition range (MS:4000069)*

Description

MS:4000069
"Upper and lower limit of m/z precursor values at which MSn spectra are recorded." [PSI:MS]

The metric is calculated as follows:
(1) the `Spectra` object is filtered according to the MS level,
(2) the precursor m/z values of the peaks within the `Spectra` object are obtained,
(3) the minimum and maximum precursor m/z values are obtained and returned.

Usage

```
mzAcquisitionRange(spectra, msLevel = 2L, ...)
```

Arguments

`spectra`	`Spectra` object
`msLevel`	`integer`
`...`	not used here

Details

MS:4000069
is_a: MS:4000004 ! n-tuple
relationship: has_metric_category MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000019 ! MS metric
relationship: has_units MS:1000040 ! m/z
relationship: has_value_concept STATO:0000035 ! range

Value

`numeric(2)`

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"),
    precursorMz = c(170.16, 170.16, 195.08))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
sps <- Spectra(spd)
mzAcquisitionRange(spectra = sps, msLevel = 2L)
```

`numberEmptyScans` *number of empty MS1 scans (MS:4000099), number of empty MS2 scans (MS:4000100), or number of empty MS3 scans (MS:4000101)*

Description

MS:4000099
"Number of MS1 scans where the scans' peaks intensity sums to 0 (i.e. no peaks or only 0-intensity peaks)." [PSI:MS]

MS:4000100
"Number of MS2 scans where the scans' peaks intensity sums to 0 (i.e. no peaks or only 0-intensity peaks)." [PSI:MS]

MS:4000101
"Number of MS3 scans where the scans' peaks intensity sums to 0 (i.e. no peaks or only 0-intensity peaks)." [PSI:MS]

The metric is calculated as follows:
(1) the `Spectra` object is filtered according to the MS level,
(2) the intensities per entry are obtained,
(3) the number of intensity entries that are `NULL`, `NA`, or that have a sum of `0` are obtained and returned.

Usage

```
numberEmptyScans(object, msLevel = 1L, ...)
```

Arguments

object	Spectra or Chromatograms object
msLevel	integer
...	not used here

Details

MS:4000099
is_a: MS:4000003 ! single value
relationship: has_metric_category MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000021 ! MS1 metric
relationship: has_units UO:0000189 ! count unit
relationship: has_value_type xsd:integer ! The allowed value-type for this CV term

MS:4000100
is_a: MS:4000003 ! single value
relationship: has_metric_category MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000022 ! MS2 metric
relationship: has_units UO:0000189 ! count unit
relationship: has_value_type xsd:integer ! The allowed value-type for this CV term

MS:4000101
is_a: MS:4000003 ! single value
relationship: has_metric_category MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_units UO:0000189 ! count unit
relationship: has_value_type xsd:integer ! The allowed value-type for this CV term

#' For *MS:4000099*, msLevel is set to 1. For *MS:4000100*, msLevel is set to 2. For *MS:4000101*, msLevel is set to 3.

An attribute containing the PSI:MS term will only be returned if msLevel is either 1, 2, or 3.

Value

numeric(1)

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)
```

```
spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
sps <- Spectra(spd)
numberEmptyScans(object = sps, msLevel = 1L)
numberEmptyScans(object = sps, msLevel = 2L)
```

`numberSpectra` *number of MS1 spectra (MS:4000059) or number of MS2 spectra (MS:4000060)*

Description

MS:4000059
"The number of MS1 events in the run." [PSI:MS]

MS:4000060
"The number of MS2 events in the run." [PSI:MS]

For *MS:4000059*, `msLevel` is set to 1. For *MS:4000060*, `msLevel` is set to 2.

The metric is calculated as follows:
(1) the `Spectra` object is filtered according to the MS level,
(2) the number of the spectra are obtained (`length` of `Spectra`) and returned.

Usage

```
numberSpectra(spectra, msLevel = 1L, ...)
```

Arguments

`spectra`	`Spectra` object
`msLevel`	`integer`
`...`	not used here

Details

MS:4000059
synonym: "MS1-Count" EXACT [PMID:24494671]
is_a: MS:4000003 ! single value
relationship: has_metric_category MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000021 ! MS1 metric
relationship: has_value_type xsd:int ! The allowed value-type for this CV term
relationship: has_units UO:0000189 ! count unit

MS:4000060
synonym: "MS2-Count" EXACT [PMID:24494671]
is_a: MS:4000003 ! single value
relationship: has_metric_category MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000022 ! MS2 metric
relationship: has_value_type xsd:int ! The allowed value-type for this CV term
relationship: has_units UO:0000189 ! count unit

An attribute containing the PSI:MS term will only be returned if `msLevel` is either 1 or 2.

Value

`numeric(1)`

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
sps <- Spectra(spd)
numberSpectra(spectra = sps, msLevel = 1L)
numberSpectra(spectra = sps, msLevel = 2L)
```

peakCount *Peak Count (Number of data points) per chromatogram*

Description

The function ‘peakCount‘ returns the number of data points (retention time - intensity pairs) for each chromatogram.

Usage

```
peakCount(chromatograms, msLevel = integer(), na.rm = FALSE, ...)
```

Arguments

chromatograms	‘Chromatograms‘ object
msLevel	‘integer‘ defining the MS level(s) to filter for. If empty (default), no filtering is performed.
na.rm	‘logical(1)‘ indicating whether ‘NA‘ values should be removed before counting (default ‘FALSE‘)
...	further arguments (currently ignored)

Details

The function returns the count of data points per chromatogram.

No specific PSI:MS term exists for chromatogram peak count.

Value

‘integer‘ vector of length equal to number of chromatograms

Author(s)

Philippine Louail

Examples

```
library(Chromatograms)
cdata <- data.frame(
    msLevel = c(1L, 1L, 1L),
    mz = c(112.2, 123.3, 134.4),
    dataOrigin = c("mem1", "mem1", "mem1")
)
pdata <- list(
    data.frame(rtime = c(2.1, 2.5, 3.0, 3.4, 3.9),
               intensity = c(100, 250, 400, 300, 150)),
    data.frame(rtime = numeric(), intensity = numeric()),
    data.frame(rtime = c(5.1, 5.8, 6.3, 6.9, 7.5),
               intensity = c(80, 500, 1200, 600, 120))
)
chr <- Chromatograms(ChromBackendMemory(), chromData = cdata, peaksData = pdata)
peakCount(chr)
```

`peakProminence` *Peak Prominence (Peak-to-Baseline Ratio) per Chromatogram*

Description

The function ‘peakProminence‘ calculates the prominence of the main chromatographic peak relative to its baseline in each chromatogram.

The metric is calculated as follows (per chromatogram):
(1) the retention time and intensity values are extracted,
(2) if `peakBoundary` is provided, the data is subset to the peak region,
(3) the maximum intensity is determined,
(4) the baseline intensity is estimated from the `baselineQuantile` of intensities,
(5) the prominence is calculated as `(max - baseline) / baseline` and returned.

Usage

```
peakProminence(
  chromatograms,
  msLevel = integer(),
  peakBoundary = NULL,
  baselineQuantile = 0.1,
  ...
)
```

Arguments

`chromatograms`	‘Chromatograms‘ object
`msLevel`	‘integer‘ defining the MS level(s) to filter for. If empty (default), no filtering is performed.
`peakBoundary`	optional peak boundary: either a ‘matrix‘ with columns ‘left_boundary‘ and ‘right_boundary‘ (one row per chromatogram), or a ‘numeric(2)‘ named vector applied to all chromatograms.
`baselineQuantile`	‘numeric(1)‘ quantile used to estimate the baseline intensity (default 0.1 = 10th percentile of intensities).
`...`	further arguments (currently unused)

Details

Higher values indicate more prominent peaks that stand out clearly from the baseline.

Value

‘numeric‘ of length equal to ‘length(chromatograms)‘. Returns ‘NA‘ for chromatograms where prominence cannot be calculated.

Author(s)

Philippine Louail

Examples

```
library(Chromatograms)

cdata <- data.frame(
    msLevel = c(1L, 1L),
    mz = c(100.0, 200.0),
    dataOrigin = c("mem1", "mem1")
)
## Good peak with high prominence and noisy plateau with low prominence
pdata <- list(
    data.frame(rtime = 1:20,
               intensity = c(100, 100, 100, 200, 500, 1000, 2000, 5000,
                             10000, 15000, 10000, 5000, 2000, 1000, 500,
                             200, 100, 100, 100, 100)),
    data.frame(rtime = 1:20,
               intensity = c(3000, 3500, 4000, 5000, 8000, 10000, 12000,
                             11000, 10000, 11000, 12000, 10000, 9000,
                             8000, 7000, 6000, 5000, 4000, 3500, 3000))
)
chr <- Chromatograms(ChromBackendMemory(), chromData = cdata, peaksData = pdata)
peakProminence(chr)
```

`peakWidth` *Peak Width per Chromatogram*

Description

The function ‘peakWidth‘ calculates the width of the main peak in each chromatogram.

The metric is calculated as follows (per chromatogram):
(1) if not provided, peak boundaries are determined using `peakBoundary()`,
(2) the peak width is calculated as the difference between the right and left boundary retention times and returned.

Usage

```
peakWidth(chromatograms, msLevel = integer(), peakBoundary = NULL, ...)
```

Arguments

`chromatograms`	‘Chromatograms‘ object
`msLevel`	‘integer‘ defining the MS level(s) to filter for. If empty (default), no filtering is performed.
`peakBoundary`	optional peak boundary: either a ‘matrix‘ with columns ‘left_boundary‘ and ‘right_boundary‘ (one row per chromatogram, as returned by ‘peakBoundary()‘), or a ‘numeric(2)‘ named vector applied to all chromatograms. If not provided, boundaries are calculated automatically.
`...`	further arguments passed to ‘peakBoundary()‘

Details

This is different from FWHM which measures width at 50% of max intensity.

Value

'numeric' of length equal to 'length(chromatograms)'. Returns 'NA' for chromatograms where width cannot be calculated.

Author(s)

Philippine Louail

Examples

```
library(Chromatograms)
cdata <- data.frame(
    msLevel = c(1L, 1L),
    mz = c(100.0, 200.0),
    dataOrigin = c("mem1", "mem1")
)
pdata <- list(
    data.frame(rtime = c(2.1, 2.5, 3.0, 3.4, 3.9),
               intensity = c(100, 250, 400, 300, 150)),
    data.frame(rtime = c(1, 2, 3, 4, 5, 6, 7),
               intensity = c(0, 10, 50, 100, 50, 10, 0))
)
chr <- Chromatograms(ChromBackendMemory(), chromData = cdata, peaksData = pdata)
peakWidth(chr)

## Use pre-computed peak boundaries for efficiency
pb <- peakBoundary(chr)
peakWidth(chr, peakBoundary = pb)
```

`plotMetric` *Visualize a quality metric*

Description

The function `plotMetric` visualizes the metric values per sample. The function accepts the output of `calculateMetrics` or, `calculateMetricsFromSpectra`, or `calculateMetricsFromMsExperiment` and a vector specifying the metric to display.

Usage

```
plotMetric(qc, metric = "areaUnderTic", plotly = TRUE)
```

Arguments

qc	matrix/data.frame
metric	character
plotly	logical(1)

Details

plotMetric will select all columns that start with metric. The different levels in the name column in the returned tibble correspond to the columns that were selected and do not contain the metric prefix. In case there is no additional specification (e.g. for the metric chromatographyDuration only the column chromatographyDuration will be selected), the name column will include the metric (chromatographyDuration).

Value

gg plotly

Author(s)

Thomas Naake, <thomasnaake@googlemail.com>

Examples

```
library(MsDataHub)
library(MsExperiment)
library(S4Vectors)
msexp <- MsExperiment()
sd <- DataFrame(sample_id = c("QC1", "QC2"),
    sample_name = c("QC Pool", "QC Pool"), injection_idx = c(1, 3))
sampleData(msexp) <- sd

## define file names containing spectra data for the samples and
## add them, along with other arbitrary files to the experiment
sciex_file <- c(MsDataHub::X20171016_POOL_POS_1_105.134.mzML(),
    MsDataHub::X20171016_POOL_POS_3_105.134.mzML())

library(Spectra)
## import the data and add it to the msexp object
spectra(msexp) <- Spectra(sciex_file)

## define the quality metrics to be calculated
metrics <- c("areaUnderTic", "chromatographyDuration", "msSignal10xChange")

## calculate the metrics
## additional parameters passed to the quality metrics functions
## (msLevel is an argument of areaUnderTic and msSignal10xChange,
## relativeTo is an argument of msSignal10xChange)
qc <- calculateMetricsFromMsExperiment(msexp = msexp, metrics = metrics,
    msLevel = 1, relativeTo = "Q1", change = "jump")
rownames(qc) <- c("Sample 1", "Sample 2")

## do the actual plotting
plotMetric(qc, metric = "areaUnderTic", plotly = TRUE)
```

---

plotMetricTibble *Helper function for plotMetric*

Description

The function plotMetricTibble is a helper function for the function plotMetric. It returns a tibble in long format that is interpretable by ggplot2.

Usage

```
plotMetricTibble(qc, metric)
```

Arguments

qc	data.frame
metric	character

Details

plotMetricRibble will select all columns that start with metric. The different levels in the name column in the returned tibble correspond to the columns that were selected and do not contain the metric prefix. In case there is no additional specification (e.g. for the metric chromatographyDuration only the column chromatographyDuration will be selected), the name column will include the metric (chromatographyDuration).

Value

tibble

Author(s)

Thomas Naake, <thomasnaake@googlemail.com>

Examples

```
library(MsDataHub)
library(MsExperiment)
library(S4Vectors)
msexp <- MsExperiment()
sd <- DataFrame(sample_id = c("QC1", "QC2"),
    sample_name = c("QC Pool", "QC Pool"), injection_idx = c(1, 3))
sampleData(msexp) <- sd

## define file names containing spectra data for the samples and
## add them, along with other arbitrary files to the experiment
## define file names containing spectra data for the samples
sciex_file <- c(MsDataHub::X20171016_POOL_POS_1_105.134.mzML(),
    MsDataHub::X20171016_POOL_POS_3_105.134.mzML())

experimentFiles(msexp) <- MsExperimentFiles(
    mzML_files = sciex_file,
    annotations = "internal_standards.txt")

## link samples to data files: first sample to first file in "mzML_files",
## second sample to second file in "mzML_files"
msexp <- linkSampleData(msexp, with = "experimentFiles.mzML_files",
    sampleIndex = c(1, 2), withIndex = c(1, 2))
msexp <- linkSampleData(msexp, with = "experimentFiles.annotations",
    sampleIndex = c(1, 2), withIndex = c(1, 1))
```

```
library(Spectra)
## import the data and add it to the mse object
spectra(msexp) <- Spectra(sciex_file)

## define the quality metrics to be calculated
metrics <- c("areaUnderTic", "chromatographyDuration", "msSignal10xChange")

## calculate the metrics
## additional parameters passed to the quality metrics functions
## (msLevel is an argument of areaUnderTic and msSignal10xChange,
## relativeTo is an argument of msSignal10xChange)
qc <- calculateMetricsFromMsExperiment(msexp = msexp, metrics = metrics,
    msLevel = 1, relativeTo = "Q1", change = "jump")
rownames(qc) <- c("Sample 1", "Sample 2")
plotMetricTibble(qc, metric = "areaUnderTic")
```

`precursorIntensityMean`

MS2 precursor intensity distribution mean (MS:4000117), identified MS2 precursor intensity distribution mean (MS:4000163), or unidentified MS2 precursor intensity distribution mean (MS:4000164)

Description

MS:4000117
"From the distribution of MS2 precursor intensities, the mean. The intensity distribution of the precursors informs about the dynamic range of the acquisition." [PSI:MS]

MS:4000163
"From the distribution of identified MS2 precursor intensities, the mean. The intensity distribution of the identified precursors informs about the dynamic range of the acquisition in relation to identifiability." [PSI:MS]

MS:4000164
"From the distribution of unidentified MS2 precursor intensities, the mean. The intensity distribution of the unidentified precursors informs about the dynamic range of the acquisition in relation to identifiability." [PSI:MS]

The metric is calculated as follows:
(1) the `Spectra` object is filtered according to the MS level,
(2) the intensity of the precursor ions within the `Spectra` object are obtained,
(3) the mean of the precursor intensity values is obtained (`NA` values are removed) and returned.

Usage

```
precursorIntensityMean(
  spectra,
  msLevel = 1L,
  identificationLevel = c("all", "identified", "unidentified"),
  ...
)
```

Arguments

spectra	Spectra object
msLevel	integer
identificationLevel	character(1), one of "all", "identified", or "unidentified"
...	not used here

Details

MS:4000117
is_a: MS:4000003 ! single value
relationship: has_metric_category MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000022 ! MS2 metric
relationship: has_value_concept STATO:0000401 ! sample mean
relationship: has_value_type xsd:float ! The allowed value-type for this CV term
relationship: has_units MS:1000043 ! intensity unit

MS:4000163
is_a: MS:4000003 ! single value
is_a: MS:4000008 ! ID based
relationship: has_metric_category MS:4000022 ! MS2 metric
relationship: has_value_concept STATO:0000401 ! sample mean
relationship: has_value_type xsd:float ! The allowed value-type for this CV term
relationship: has_units MS:1000043 ! intensity unit

MS:4000164
is_a: MS:4000003 ! single value
is_a: MS:4000008 ! ID based
relationship: has_metric_category MS:4000022 ! MS2 metric
relationship: has_value_concept STATO:0000401 ! sample mean
relationship: has_value_type xsd:float ! The allowed value-type for this CV term
relationship: has_units MS:1000043 ! intensity unit

Value

numeric(1)

Note

The Spectra object might contain features that were (not) identified. If the calculation needs to be done according to *MS:4000163*/*MS:4000164*, the Spectra object should be prepared accordingly.

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
spd$precursorIntensity <- c(100.0, 100.0, 100.0)
sps <- Spectra(spd)
precursorIntensityMean(spectra = sps, msLevel = 2L)
```

`precursorIntensityQuartiles`

MS2 precursor intensity distribution (MS:4000116), identified MS2 precursor intensity distribution (MS:4000161), or unidentified MS2 precursor intensity distribution (MS:4000162)

Description

MS:4000116
"From the distribution of MS2 precursor intensities, the quantiles. E.g. a value triplet represents the quartiles Q1, Q2, Q3. The intensity distribution of the precursors informs about the dynamic range of the acquisition." [PSI:MS]

MS:40000161
From the distribution of identified MS2 precursor intensities, the quantiles. E.g. a value triplet represents the quartiles Q1, Q2, Q3. The intensity distribution of the precursors informs about the dynamic range of the acquisition in relation to identifiability. The used type of identification should be noted in the metadata or analysis methods section of the recording file for the respective run. In case of multiple acceptance criteria (FDR) available in proteomics, PSM-level FDR should be used for better comparability." [PSI:MS]"

id: MS:4000162
"From the distribution of unidentified MS2 precursor intensities, the quantiles. E.g. a value triplet represents the quartiles Q1, Q2, Q3. The intensity distribution of the precursors informs about the dynamic range of the acquisition in relation to identifiability. The used type of identification should be noted in the metadata or analysis methods section of the recording file for the respective run. In case of multiple acceptance criteria (FDR) available in proteomics, PSM-level FDR should be used for better comparability." [PSI:MS]"

The metric is calculated as follows:
(1) the `Spectra` object is filtered according to the MS level,
(2) the intensity of the precursor ions within the `Spectra` object are obtained,
(3) the 25%, 50%, and 75% quantile of the precursor intensity values are obtained (`NA` values are removed) and returned.

Usage

```
precursorIntensityQuartiles(
  spectra,
  msLevel = 1L,
  identificationLevel = c("all", "identified", "unidentified"),
  ...
)
```

Arguments

`spectra`	`Spectra` object
`msLevel`	`integer`
`identificationLevel`	`character(1)`, one of `"all"`, `"identified"`, or `"unidentified"`
`...`	not used here

Details

id: MS:4000116
is_a: MS:4000004 ! n-tuple
relationship: has_metric_category MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000022 ! MS2 metric
relationship: has_value_concept STATO:0000291 ! quantile
relationship: has_value_type xsd:float ! The allowed value-type for this CV term
relationship: has_units MS:1000043 ! intensity unit

MS:4000161
is_a: MS:4000004 ! n-tuple
is_a: MS:4000008 ! ID based
relationship: has_metric_category MS:4000022 ! MS2 metric
relationship: has_value_concept STATO:0000291 ! quantile
relationship: has_value_type xsd:float ! The allowed value-type for this CV term
relationship: has_units MS:1000043 ! intensity unit

id: MS:4000162
is_a: MS:4000004 ! n-tuple
is_a: MS:4000008 ! ID based
relationship: has_metric_category MS:4000022 ! MS2 metric
relationship: has_value_concept STATO:0000291 ! quantile
relationship: has_value_type xsd:float ! The allowed value-type for this CV term
relationship: has_units MS:1000043 ! intensity unit

Value

numeric(3)

Note

The Spectra object might contain features that were (not) identified. If the calculation needs to be done according to *MS:4000161*/*MS:4000162*, the Spectra object should be prepared accordingly.

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
spd$precursorIntensity <- c(100.0, 100.0, 100.0)
sps <- Spectra(spd)

precursorIntensityQuartiles(spectra = sps, msLevel = 2L)
```

---

precursorIntensityRange

MS2 precursor intensity range (MS:4000160)

Description

MS:4000160
"Minimum and maximum MS2 precursor intensity recorded. The intensity range of the precursors informs about the dynamic range of the acquisition." [PSI:MS]

The metric is calculated as follows:
(1) the Spectra object is filtered according to the MS level,
(2) the intensity of the precursor ions within the Spectra object are obtained,
(3) the minimum and maximum precursor intensity values are obtained and returned.

Usage

```
precursorIntensityRange(spectra, msLevel = 1, ...)
```

Arguments

spectra	Spectra object
msLevel	integer
...	not used here

Details

MS:4000160
is_a: MS:4000001 ! QC metric
is_a: MS:4000004 ! n-tuple
is_a: MS:4000009 ! ID free
relationship: has_metric_category MS:4000022 ! MS2 metric

Value

```
numeric(2)
```

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
spd$precursorIntensity <- c(100.0, 100.0, 100.0)
sps <- Spectra(spd)
precursorIntensityRange(spectra = sps, msLevel = 2L)
```

`precursorIntensitySd` *MS2 precursor intensity distribution sigma (MS:4000118), identified MS2 precursor intensity distribution sigma (MS:4000165), or unidentified MS2 precursor intensity distribution sigma (MS:4000166)*

Description

MS:4000118
"From the distribution of MS2 precursor intensities, the sigma value. The intensity distribution of the precursors informs about the dynamic range of the acquisition." [PSI:MS]

MS:4000165
"From the distribution of identified MS2 precursor intensities, the sigma value. The intensity distribution of the precursors informs about the dynamic range of the acquisition in relation to identifiability. The used type of identification should be noted in the metadata or analysis methods section of the recording file for the respective run. In case of multiple acceptance criteria (FDR) available in proteomics, PSM-level FDR should be used for better comparability." [PSI:MS]

MS:4000166
"From the distribution of unidentified MS2 precursor intensities, the sigma value. The intensity distribution of the precursors informs about the dynamic range of the acquisition in relation to identifiability. The used type of identification should be noted in the metadata or analysis methods section of the recording file for the respective run. In case of multiple acceptance criteria (FDR) available in proteomics, PSM-level FDR should be used for better comparability." [PSI:MS]

The metric is calculated as follows:
(1) the `Spectra` object is filtered according to the MS level,
(2) the intensity of the precursor ions within the `Spectra` object are obtained,
(3) the standard deviation of precursor intensity values is obtained (`NA` values are removed) and returned.

Usage

```
precursorIntensitySd(
  spectra,
  msLevel = 1L,
  identificationLevel = c("all", "identified", "unidentified"),
  ...
)
```

Arguments

`spectra`	`Spectra` object
`msLevel`	`integer`
`identificationLevel`	`character(1)`, one of `"all"`, `"identified"`, or `"unidentified"`
`...`	not used here

Details

MS:4000118
is_a: MS:4000003 ! single value
relationship: has_metric_category MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000022 ! MS2 metric
relationship: has_value_concept STATO:0000237 ! standard deviation
relationship: has_value_type xsd:float ! The allowed value-type for this CV term
relationship: has_units MS:1000043 ! intensity unit

MS:4000165
is_a: MS:4000003 ! single value
relationship: has_metric_category MS:4000008 ! ID based
relationship: has_metric_category MS:4000022 ! MS2 metric
relationship: has_value_concept STATO:0000237 ! standard deviation
relationship: has_value_type xsd:float ! The allowed value-type for this CV term
relationship: has_units MS:1000043 ! intensity unit

MS:4000166
is_a: MS:4000003 ! single value
relationship: has_metric_category MS:4000008 ! ID based
relationship: has_metric_category MS:4000022 ! MS2 metric
relationship: has_value_concept STATO:0000237 ! standard deviation
relationship: has_value_type xsd:float ! The allowed value-type for this CV term
relationship: has_units MS:1000043 ! intensity unit

Value

`numeric(1)`

Note

The `Spectra` object might contain features that were (not) identified. If the calculation needs to be done according to *MS:4000165*/*MS:4000166*, the `Spectra` object should be prepared accordingly.

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
```

```
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
spd$precursorIntensity <- c(100.0, 100.0, 100.0)
sps <- Spectra(spd)
precursorIntensitySd(spectra = sps, msLevel = 2L)
```

`qualityMetrics` *Get a vector of quality metrics than can be applied to* `object`

Description

The function `qualityMetrics` returns a character vector with available quality metrics depending on `object`.

Usage

```
qualityMetrics(object)
```

Arguments

`object`	object of type `Spectra`, `MsExperiment`, or `Chromatograms`

Details

`object` is a `Spectra`, `MsExperiment`, or `Chromatograms`.

Value

`character`

Author(s)

Thomas Naake, Philippine Louail

Examples

```
library(Spectra)
spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
```

```
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
spd$dataOrigin <- rep("sample_1", 3)
sps <- Spectra(spd)

qualityMetrics(object = sps)
```

`ratioCharge1over2` *ratio of 1+ over 2+ of all MS2 known precursor charges (MS:4000167) or ratio of 1+ over 2+ of identified MS2 known precursor charges (MS:4000168)*

Description

MS:4000167
"The ratio of 1+ over 2+ MS2 precursor charge count of all spectra. High ratios of 1+/2+ MS2 precursor charge count may indicate inefficient ionization." [PSI:MS]

MS:4000168
"The ratio of 1+ over 2+ MS2 precursor charge count of identified spectra. High ratios of 1+/2+ MS2 precursor charge count may indicate inefficient ionization. The used type of identification should be noted in the metadata or analysis methods section of the recording file for the respective run. In case of multiple acceptance criteria (FDR) available in proteomics, PSM-level FDR should be used for better comparability." [PSI:MS]

The metric is calculated as follows:
(1) the `Spectra` object is filtered according to the MS level,
(2) the precursor charge is obtained,
(3) the number of precursors with charge 1+ is divided by the number of precursors with charge 2+ and the ratio is returned.

Usage

```
ratioCharge1over2(
  spectra,
  msLevel = 1L,
  identificationLevel = c("all", "identified", "unidentified"),
  ...
)
```

Arguments

`spectra`	`Spectra` object
`msLevel`	`integer`
`identificationLevel`	`character(1)`, one of `"all"`, `"identified"`, or `"unidentified"`
`...`	not used here

Details

MS:4000167
is_a: MS:4000001 ! QC metric
is_a: MS:4000003 ! single value
is_a: MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000020 ! ion source metric
relationship: has_metric_category MS:4000022 ! MS2 metric
synonym: "IS-3A" RELATED [PMID:19837981]
synonym: "MS2 known precursor charges fractions" RELATED []
synonym: "MS2-PrecZ-1" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-2" RELATED [PMID:24494671]

MS:4000168
is_a: MS:4000001 ! QC metric
is_a: MS:4000003 ! single value
is_a: MS:4000008 ! ID based
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000020 ! ion source metric
relationship: has_metric_category MS:4000022 ! MS2 metric
synonym: "IS-3A" RELATED [PMID:19837981]
synonym: "MS2 known precursor charges fractions" RELATED []
synonym: "MS2-PrecZ-1" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-2" RELATED [PMID:24494671]

`NA` is returned if there are no features with precursor charge of 1+ or 2+.

An attribute containing the PSI:MS term will only be returned if `identificationLevel` is either `"all"` or `"identified"`.

Value

`numeric(1)`

Note

The `Spectra` object might contain features that were not identified. If the calculation needs to be done according to *MS:4000168*, the `Spectra` object should be prepared accordingly.

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
```

```
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
spd$precursorCharge <- c(1L, 1L, 1L)
sps <- Spectra(spd)
ratioCharge1over2(spectra = sps, msLevel = 2L)
```

`ratioCharge3over2` *ratio of 3+ over 2+ of all MS2 known precursor charges (MS:4000169) or ratio of 3+ over 2+ of identified MS2 known precursor charges (MS:4000170)*

Description

MS:4000169
"The ratio of 3+ over 2+ MS2 precursor charge count of all spectra. Higher ratios of 3+/2+ MS2 precursor charge count may preferentially favor longer e.g. peptides." [PSI:MS]

MS:4000170
"The ratio of 3+ over 2+ MS2 precursor charge count of identified spectra. Higher ratios of 3+/2+ MS2 precursor charge count may preferentially favor longer e.g. peptides. The used type of identification should be noted in the metadata or analysis methods section of the recording file for the respective run. In case of multiple acceptance criteria (FDR) available in proteomics, PSM-level FDR should be used for better comparability." [PSI:MS]

The metric is calculated as follows:
(1) the `Spectra` object is filtered according to the MS level,
(2) the precursor charge is obtained,
(3) the number of precursors with charge 3+ is divided by the number of precursors with charge 2+ and the ratio is returned.

Usage

```
ratioCharge3over2(
  spectra,
  msLevel = 1L,
  identificationLevel = c("all", "identified", "unidentified"),
  ...
)
```

Arguments

spectra	Spectra object
msLevel	integer
identificationLevel	character(1), one of "all", "identified", or "unidentified"
...	not used here

Details

MS:4000169
is_a: MS:4000001 ! QC metric
is_a: MS:4000003 ! single value
is_a: MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000020 ! ion source metric
relationship: has_metric_category MS:4000022 ! MS2 metric
synonym: "IS-3B" RELATED [PMID:19837981]
synonym: "MS2 known precursor charges fractions" RELATED []
synonym: "MS2-PrecZ-2" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-3" RELATED [PMID:24494671]

MS:4000170
is_a: MS:4000001 ! QC metric
is_a: MS:4000003 ! single value
is_a: MS:4000008 ! ID based
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000020 ! ion source metric
relationship: has_metric_category MS:4000022 ! MS2 metric
synonym: "IS-3B" RELATED [PMID:19837981]
synonym: "MS2 known precursor charges fractions" RELATED []
synonym: "MS2-PrecZ-2" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-3" RELATED [PMID:24494671]

NA is returned if there are no features with precursor charge of 2+ or 3+.

An attribute containing the PSI:MS term will only be returned if identificationLevel is either "all" or "identified".

Value

numeric(1)

Note

The Spectra object might contain features that were not identified. If the calculation needs to be done according to *MS:4000170*, the Spectra object should be prepared accordingly.

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
spd$precursorCharge <- c(1L, 1L, 1L)
sps <- Spectra(spd)
ratioCharge3over2(spectra = sps, msLevel = 2L)
```

`ratioCharge4over2` *ratio of 4+ over 2+ of all MS2 known precursor charges (MS:4000171) or ratio of 4+ over 2+ of identified MS2 known precursor charges (MS:4000172)*

Description

MS:4000171
"The ratio of 4+ over 2+ MS2 precursor charge count of all spectra. Higher ratios of 4+/2+ MS2 precursor charge count may preferentially favor longer e.g. peptides." [PSI:MS]

MS:4000172
"The ratio of 4+ over 2+ MS2 precursor charge count of identified spectra. Higher ratios of 4+/2+ MS2 precursor charge count may preferentially favor longer e.g. peptides. The used type of identification should be noted in the metadata or analysis methods section of the recording file for the respective run. In case of multiple acceptance criteria (FDR) available in proteomics, PSM-level FDR should be used for better comparability." [PSI:MS]

The metric is calculated as follows:
(1) the `Spectra` object is filtered according to the MS level,
(2) the precursor charge is obtained,
(3) the number of precursors with charge 4+ is divided by the number of precursors with charge 2+ and the ratio is returned.

Usage

```
ratioCharge4over2(
  spectra,
  msLevel = 1L,
```

```
  identificationLevel = c("all", "identified", "unidentified"),
  ...
)
```

Arguments

spectra	Spectra object
msLevel	integer
identificationLevel	character(1), one of "all", "identified", or "unidentified"
...	not used here

Details

MS:4000171
is_a: MS:4000001 ! QC metric
is_a: MS:4000003 ! single value
is_a: MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000020 ! ion source metric
relationship: has_metric_category MS:4000022 ! MS2 metric
synonym: "IS-3C" RELATED [PMID:19837981]
synonym: "MS2 known precursor charges fractions" RELATED []
synonym: "MS2-PrecZ-2" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-4" RELATED [PMID:24494671]

MS:4000172
is_a: MS:4000001 ! QC metric
is_a: MS:4000003 ! single value
is_a: MS:4000008 ! ID based
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000020 ! ion source metric
relationship: has_metric_category MS:4000022 ! MS2 metric
synonym: "IS-3C" RELATED [PMID:19837981]
synonym: "MS2 known precursor charges fractions" RELATED []
synonym: "MS2-PrecZ-2" RELATED [PMID:24494671]
synonym: "MS2-PrecZ-4" RELATED [PMID:24494671]

An attribute containing the PSI:MS term will only be returned if `identificationLevel` is either `"all"` or `"identified"`.

Value

`numeric(1)`

Note

The `Spectra` object might contain features that were not identified. If the calculation needs to be done according to *MS:4000172*, the `Spectra` object should be prepared accordingly.

`NA` is returned if there are no features with precursor charge of 2+ or 3+.

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
spd$precursorCharge <- c(1L, 1L, 1L)
sps <- Spectra(spd)
ratioCharge4over2(spectra = sps, msLevel = 2L)
```

---

rtAcquisitionRange *retention time acquisition range (MS:4000070)*

Description

MS:4000070
"Upper and lower limit of retention time at which spectra are recorded." [PSI:MS]

The metric is calculated as follows:
(1) the `Spectra` object is filtered according to the MS level,
(2) the retention time values of the features within the `Spectra` object are obtained,
(3) the minimum and maximum retention time values are obtained and returned.

Usage

```
rtAcquisitionRange(object, msLevel = 1L, ...)
```

Arguments

`object`	`Spectra` or `Chromatograms` object
`msLevel`	`integer`
`...`	not used here

Details

MS:4000070
is_a: MS:4000004 ! n-tuple
relationship: has_metric_category MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000016 ! retention time metric
relationship: has_units UO:0000010 ! second
relationship: has_value_concept STATO:0000035 ! range

Value

`numeric(2)`

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
spd$rtime <- c(9.44, 9.44, 15.84)
sps <- Spectra(spd)
rtAcquisitionRange(object = sps, msLevel = 2L)
```

---

rtIqr *interquartile RT period for identified quantification data points (MS:4000153)*

Description

MS:4000153
"The interquartile retention time period, in seconds, for all quantification data points after user-defined acceptance criteria are applied over the complete run. Longer times indicate better chromatographic separation. These data points may be for example XIC profiles, isotopic pattern areas, or reporter ions (see MS:1001805). The used type should be noted in the metadata or analysis

methods section of the recording file for the respective run. In case of multiple acceptance criteria (FDR) available in proteomics, PSM-level FDR should be used for better comparability." [PSI:MS]

The metric is calculated as follows:
(1) the `Spectra` object is filtered according to the MS level,
(2) the retention time values are obtained,
(3) the interquartile range is obtained from the values and returned (`NA` values are removed).

Usage

```
rtIqr(
  object,
  msLevel = 1L,
  identificationLevel = c("all", "identified", "unidentified"),
  ...
)
```

Arguments

`object`	`Spectra` or `Chromatograms` object
`msLevel`	`integer` (only used for `Spectra`)
`identificationLevel`	`character(1)`, one of `"all"`, `"identified"`, or `"unidentified"` (only used for `Spectra`)
`...`	not used here

Details

MS:4000153
is_a: MS:4000003 ! single value
is_a: MS:4000008 ! ID based
is_a: MS:4000017 ! chromatogram metric
relationship: has_units UO:0000010 ! second
synonym: "C-2A" RELATED [PMID:19837981]

Retention time values that are `NA` are removed.

An attribute containing the PSI:MS term will only be returned if `identificationLevel` is `"identified"`.

Value

For `Spectra`: `numeric(1)`. For `Chromatograms`: `numeric` of length equal to `length(object)`, one RT IQR per chromatogram.

Note

The `Spectra` object might contain features that were not identified. If the calculation needs to be done according to *MS:4000153*, the `Spectra` object should be prepared accordingly, i.e. subsetted to `spectra` with identification data.

The stored retention time information in `spectra` might have a different unit than seconds. `rtIqr` will return the IQR based on the values stored in `spectra` and will not convert these values to seconds.

For Chromatograms objects, the function calculates the IQR of retention times across all chromatograms.

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
spd$rtime <- c(9.44, 9.44, 15.84)
sps <- Spectra(spd)
rtIqr(object = sps, msLevel = 2L)
```

---

rtIqrRate *rate of the interquartile RT period for identified quantification data points (MS:4000154)*

Description

MS:4000154
"The rate of identified quantification data points for the interquartile retention time period, in identified quantification data points per second. Higher rates indicate efficient sampling and identification. These data points may be for example XIC profiles, isotopic pattern areas, or reporter ions (see MS:1001805). The used type should be noted in the metadata or analysis methods section of the recording file for the respective run. In case of multiple acceptance criteria (FDR) available in proteomics, PSM-level FDR should be used for better comparability." [PSI:MS]

The metric is calculated as follows:
(1) the Spectra object is filtered according to the MS level,
(2) the retention time values are obtained,
(3) the 25% and 75% quantiles are obtained from the retention time values (NA values are removed),
(4) the number of eluted features between this 25% and 75% quantile is calculated,
(5) the number of features is divided by the interquartile range of the retention time and returned.

Usage

```
rtIqrRate(
  spectra,
  msLevel = 1L,
  identificationLevel = c("all", "identified", "unidentified"),
  ...
)
```

Arguments

spectra	Spectra object
msLevel	integer
identificationLevel	character(1), one of "all", "identified", or "unidentified"
...	not used here

Details

MS:4000154
is_a: MS:4000003 ! single value
is_a: MS:4000008 ! ID based
is_a: MS:4000017 ! chromatogram metric
relationship: has_units UO:0000106 ! hertz synonym: "C-2B" RELATED [PMID:19837981]

An attribute containing the PSI:MS term will only be returned if identificationLevel is "identified".

Value

numeric(2)

Note

The Spectra object might contain features that were not identified. If the calculation needs to be done according to *MS:4000154*, the Spectra object should be prepared accordingly, i.e. being subsetted to spectra with identification data.

The stored retention time information in spectra might have a different unit than seconds. .rtIqr will return the IQR based on the values stored in spectra and will not convert these values to seconds.

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
```

```
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
spd$rtime <- c(9.44, 9.44, 15.84)
sps <- Spectra(spd)
rtIqrRate(spectra = sps, msLevel = 2L)
```

`rtOverMsQuarters` *MS1 quarter RT fraction (MS:4000055) or MS2 quarter RT fraction (MS:4000056)*

Description

MS:4000055
"The interval used for acquisition of the first, second, third, and fourth quarter of all MS1 events divided by retention time duration." [PSI:MS]

MS:4000056
"The interval used for acquisition of the first, second, third, and fourth quarter of all MS2 events divided by retention time duration." [PSI:MS]

The metric is calculated as follows:
(1) the retention time duration of the whole `Spectra` object is determined (taking into account all the MS levels),
(2) the `Spectra` object is filtered according to the MS level and subsequently ordered according to the retention time
(3) the MS events are split into four (approximately) equal parts,
(4) the relative retention time is calculated (using the retention time duration from (1) and taking into account the minimum retention time),
(5) the relative retention time values associated to the MS event parts are returned.

Usage

```
rtOverMsQuarters(spectra, msLevel = 1L, ...)
```

Arguments

`spectra`	`Spectra` object
`msLevel`	`integer`
`...`	not used here

Details

MS:4000055
synonym: "RT-MS-Q1" RELATED [PMID:24494671]
synonym: "RT-MS-Q2" RELATED [PMID:24494671]
synonym: "RT-MS-Q3" RELATED [PMID:24494671]
synonym: "RT-MS-Q4" RELATED [PMID:24494671]
is_a: MS:4000004 ! n-tuple
relationship: has_metric_category MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000016 ! retention time metric
relationship: has_metric_category MS:4000021 ! MS1 metric
relationship: has_value_type xsd:float ! The allowed value-type for this CV term
relationship: has_units UO:0000191 ! fraction

MS:4000056
synonym: "RT-MSMS-Q1" RELATED [PMID:24494671]
synonym: "RT-MSMS-Q2" RELATED [PMID:24494671]
synonym: "RT-MSMS-Q3" RELATED [PMID:24494671]
synonym: "RT-MSMS-Q4" RELATED [PMID:24494671]
is_a: MS:4000004 ! n-tuple
relationship: has_metric_category MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000016 ! retention time metric
relationship: has_metric_category MS:4000022 ! MS2 metric
relationship: has_value_type xsd:float ! The allowed value-type for this CV term
relationship: has_units UO:0000191 ! fraction

The function returns `c(NaN, NaN, NaN, NaN)` if the filtered `spectra` object has less than 4 scan events.

An attribute containing the PSI:MS term will only be returned if `msLevel` is 1 or 2.

Value

`numeric(4)`

Note

`chromatographyDuration` considers the total runtime (including MS1 and MS2 scans).

Author(s)

Thomas Naake, Johannes Rainer

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L, 2L),
    polarity = c(1L, 1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847", "unknown"),
```

```
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine", "unknown"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876),
    c(83.0603, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994),
    c(3.146, 61.611))
spd$rtime <- c(9.44, 9.44, 15.84, 15.81)
sps <- Spectra(spd)
rtOverMsQuarters(spectra = sps, msLevel = 2L)
```

`shinyMsQuality` *Shiny application to visualize quality metrics*

Description

The function `shinyMsQuality` function starts a shiny application to visualize the quality metrics interactively. It allows to display all metrics contained in `qc`.

The function accepts the output of `calculateMetrics`, `calculateMetricsFromSpectra`, or `calculateMetricsFromMs`

Usage

```
shinyMsQuality(qc)
```

Arguments

qc	`matrix`, contains the calculated quality metrics, the columns contain the metrics and the rows the samples

Details

The plots within the shiny application can be saved by clicking on the download button.

Value

`shiny`

Author(s)

Thomas Naake, <thomasnaake@googlemail.com>

Examples

```
library(MsDataHub)
library(MsExperiment)
library(S4Vectors)
msexp <- MsExperiment()
sd <- DataFrame(sample_id = c("QC1", "QC2"),
    sample_name = c("QC Pool", "QC Pool"), injection_idx = c(1, 3))
sampleData(msexp) <- sd

## define file names containing spectra data for the samples and
## add them, along with other arbitrary files to the experiment
## define file names containing spectra data for the samples
sciex_file <- c(MsDataHub::X20171016_POOL_POS_1_105.134.mzML(),
    MsDataHub::X20171016_POOL_POS_3_105.134.mzML())

experimentFiles(msexp) <- MsExperimentFiles(
    mzML_files = sciex_file,
    annotations = "internal_standards.txt")
## link samples to data files: first sample to first file in "mzML_files",
## second sample to second file in "mzML_files"
msexp <- linkSampleData(msexp, with = "experimentFiles.mzML_files",
    sampleIndex = c(1, 2), withIndex = c(1, 2))
msexp <- linkSampleData(msexp, with = "experimentFiles.annotations",
    sampleIndex = c(1, 2), withIndex = c(1, 1))

library(Spectra)
## import the data and add it to the mse object
spectra(msexp) <- Spectra(sciex_file)

## define the quality metrics to be calculated
metrics <- c("areaUnderTic", "chromatographyDuration", "msSignal10xChange")

## calculate the metrics
## additional parameters passed to the quality metrics functions
## (msLevel is an argument of areaUnderTic and msSignal10xChange,
## relativeTo is an argument of msSignal10xChange)
qc <- calculateMetricsFromMsExperiment(msexp = msexp, metrics = metrics,
    msLevel = 1, relativeTo = "Q1", change = "jump")
qc <- as.matrix(qc)
rownames(qc) <- c("Sample 1", "Sample 2")

if (interactive())
    shinyMsQuality(qc = qc)
```

`signalToNoiseRatio` *Signal-to-Noise Ratio per chromatogram*

Description

The function ‘signalToNoiseRatio‘ estimates a classical signal-to-noise ratio for each chromatogram using the MAD-based noise estimate from `MsCoreUtils::noise()`.

The metric is calculated as follows (per chromatogram):
(1) the retention time and intensity values are extracted,
(2) a noise level is estimated at every data point using the Median Absolute Deviation (MAD) via `MsCoreUtils::noise()`,
(3) the signal is taken as the maximum intensity,
(4) the noise is summarised as the median of the MAD-estimated noise vector,
(5) the ratio `signal / noise` is returned.

Usage

```
signalToNoiseRatio(chromatograms, msLevel = integer(), ...)
```

Arguments

`chromatograms`	'Chromatograms' object
`msLevel`	'integer' defining the MS level(s) to filter for. If empty (default), no filtering is performed.
`...`	currently not used but included for consistency with other metric functions

Details

The MAD method (`MsCoreUtils::noise(x, y, method = "MAD")`) computes a single global Median Absolute Deviation across all intensity values in the chromatogram and replicates it to every data point. Because the estimate is global, large peaks inflate the MAD, leading to a **conservative** (under-estimated) S/N ratio. This is acceptable for quality-control purposes where consistency across samples matters more than an absolute noise floor.

The MAD estimator is statistically unstable with very few data points (roughly < 5); results from very short chromatograms should be interpreted with caution.

Returns `NA_real_` when the noise estimate is zero (e.g. constant signal) or when the chromatogram is empty.

Value

'numeric' of length equal to 'length(chromatograms)', one signal-to-noise ratio per chromatogram

Author(s)

Philippine Louail

`ticQuantileRtFraction` *TIC quantile RT fraction (MS:4000183)*

Description

MS:4000183 "The interval when the respective quantile of the TIC accumulates divided by retention time duration. The number of values in the tuple implies the quantile mode." [PSI:MS]

The metric informs about the dynamic range of the acquisition along the chromatographic separation. The metric provides information on the sample (compound) flow along the chromatographic

run, potentially revealing poor chromatographic performance, such as the absence of a signal for a significant portion of the run.

The metric is calculated as follows:
(1) the `Spectra` object is ordered according to the retention time,
(2) the cumulative sum of the ion count is calculated (TIC),
(3) the quantiles are calculated according to the `probs` argument, e.g. when `probs` is set to `c(0, 0.25, 0.5, 0.75, 1)` the 0%, 25%, 50%, 75%, and 100% quantile is calculated,
(4) the retention time/relative retention time (retention time divided by the total run time taking into account the minimum retention time) is calculated,
(5) the (relative) duration of the LC run after which the cumulative TIC exceeds (for the first time) the respective quantile of the cumulative TIC is calculated and returned.

Usage

```
ticQuantileRtFraction(
  object,
  probs = seq(0, 1, 0.25),
  msLevel = 1L,
  relative = TRUE,
  ...
)
```

Arguments

`object`	`Spectra` or `Chromatograms` object
`probs`	`numeric` defining the quantiles. See `probs = seq(0, 1, 0.25)`.
`msLevel`	`integer`
`relative`	`logical`, if set to `TRUE` the relative retention time will be returned instead of the abolute retention time
`...`	additional arguments passed to internal helpers

Details

MS:4000183 synonym: "RT-TIC-Q1" RELATED [PMID:24494671]
synonym: "RT-TIC-Q2" RELATED [PMID:24494671]
synonym: "RT-TIC-Q3" RELATED [PMID:24494671]
synonym: "RT-TIC-Q4" RELATED [PMID:24494671]
is_a: MS:4000004 ! n-tuple
relationship: has_metric_category MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000016 ! retention time metric
relationship: has_metric_category MS:4000017 ! chromatogram metric
relationship: has_units UO:0000191 ! fraction
relationship: has_value_concept STATO:0000291
relationship: has_value_type xsd:float
#'

Value

For `Spectra`: `numeric` of length equal to `length(probs)` with the relative duration (duration divided by the total run time) after which the TIC exceeds the respective quantile of the TIC. For `Chromatograms`: a `matrix` with `nrow` equal to `length(object)` and `ncol` equal to `length(probs)`.

Author(s)

Thomas Naake, Johannes Rainer

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
spd$rtime <- c(9.44, 9.44, 15.84)
sps <- Spectra(spd)
ticQuantileRtFraction(object = sps, msLevel = 2L)
```

`ticQuartileToQuartileLogRatio` *MS1 TIC-change quartile ratios (MS:4000057) or MS1 TIC quartile ratios (MS:4000058)*

Description

MS:4000057
"The log ratios of successive TIC-change quartiles. The TIC changes are the list of MS1 total ion current (TIC) value changes from one to the next scan, produced when each MS1 TIC is subtracted from the preceding MS1 TIC. The metric's value triplet represents the log ratio of the TIC-change Q2 to Q1, Q3 to Q2, TIC-change-max to Q3" [PSI:MS]
For calculation of MS:400057 set `mode = "TIC_change"`.

MS:4000058
"The log ratios of successive TIC quartiles. The metric's value triplet represents the log ratios of TIC-Q2 to TIC-Q1, TIC-Q3 to TIC-Q2, TIC-max to TIC-Q3." [PSI:MS]
For calculation of MS:400058 set `mode = "TIC"`.

The metric is calculated as follows:
(1) the TIC (`ionCount`) of the `Spectra` object is calculated per scan event (with spectra ordered by retention time),
(2) for *MS:4000057*, the differences between TIC values are calculated between subsequent scan events, for *MS:4000058*, the TIC values between subsequent scan events are taken as they are,

(3) for *MS:4000057* and *MS:4000058* the ratios between the 25%, 50%, 75%, and 100% quantile to the 25% quantile of the values of (2) are calculated. Alternatively, if `relativeTo = "Q1"`, the ratios are calculated between the 50%/25%, 75%/25%, and 100%/25% quantiles,
(4) The `log` values of the ratios are returned.

Usage

```
ticQuartileToQuartileLogRatio(
  spectra,
  relativeTo = c("previous", "Q1"),
  mode = c("TIC_change", "TIC"),
  msLevel = 1L,
  ...
)
```

Arguments

`spectra`	`Spectra` object
`relativeTo`	`character(1)`, one of `"Q1"` or `"previous"`
`mode`	`character(1)`, one of `"TIC_change"` or `"TIC"`
`msLevel`	`integer`
`...`	not used here

Details

MS:4000057
synonym: "MS1-TIC-Change-Q2" RELATED [PMID:24494671]
synonym: "MS1-TIC-Change-Q3" RELATED [PMID:24494671]
synonym: "MS1-TIC-Change-Q4" RELATED [PMID:24494671]
is_a: MS:4000004 ! n-tuple
relationship: has_metric_category MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000017 ! chromatogram metric
relationship: has_metric_category MS:4000021 ! MS1 metric
relationship: has_value_type xsd:float ! The allowed value-type for this CV term
relationship: has_value_concept STATO:0000105 ! log signal intensity ratio

MS:4000058
synonym: "MS1-TIC-Q2" RELATED [PMID:24494671]
synonym: "MS1-TIC-Q3" RELATED [PMID:24494671]
synonym: "MS1-TIC-Q4" RELATED [PMID:24494671]
is_a: MS:4000004 ! n-tuple
relationship: has_metric_category MS:4000009 ! ID free metric
relationship: has_metric_category MS:4000012 ! single run based metric
relationship: has_metric_category MS:4000017 ! chromatogram metric
relationship: has_metric_category MS:4000021 ! MS1 metric
relationship: has_value_type xsd:float ! The allowed value-type for this CV term
relationship: has_value_concept STATO:0000105 ! log signal intensity ratio

An attribute containing the PSI:MS term will only be returned if `relativeTo` is `"previous"` and `msLevel` is 1.

Value

numeric(1)

Note

This function interprets the *quantiles* from the [PSI:MS] definition as *quartiles*, i.e. the 0, 25, 50, 75 and 100% quantiles are used.

Author(s)

Thomas Naake

Examples

```
library(S4Vectors)
library(Spectra)

spd <- DataFrame(
    msLevel = c(2L, 2L, 2L),
    polarity = c(1L, 1L, 1L),
    id = c("HMDB0000001", "HMDB0000001", "HMDB0001847"),
    name = c("1-Methylhistidine", "1-Methylhistidine", "Caffeine"))
## Assign m/z and intensity values
spd$mz <- list(
    c(109.2, 124.2, 124.5, 170.16, 170.52),
    c(83.1, 96.12, 97.14, 109.14, 124.08, 125.1, 170.16),
    c(56.0494, 69.0447, 83.0603, 109.0395, 110.0712,
        111.0551, 123.0429, 138.0662, 195.0876))
spd$intensity <- list(
    c(3.407, 47.494, 3.094, 100.0, 13.240),
    c(6.685, 4.381, 3.022, 16.708, 100.0, 4.565, 40.643),
    c(0.459, 2.585, 2.446, 0.508, 8.968, 0.524, 0.974, 100.0, 40.994))
sps <- Spectra(spd)

## MS:4000057
ticQuartileToQuartileLogRatio(spectra = sps, relativeTo = "previous",
    msLevel = 2L, mode = "TIC_change")
ticQuartileToQuartileLogRatio(spectra = sps, relativeTo = "Q1",
    msLevel = 2L, mode = "TIC_change")

## MS:4000058
ticQuartileToQuartileLogRatio(spectra = sps, relativeTo = "previous",
    msLevel = 2L, mode = "TIC")
ticQuartileToQuartileLogRatio(spectra = sps, relativeTo = "Q1",
    msLevel = 2L, mode = "TIC")
```

---

transformIntoMzQC *Transform the metrics into a list of* MzQCmzQC *objects*

Description

The function transformIntoMzQC transfers the metrics stored in spectra_metrics into a list of MzQCmzQC objects. Each list entry will refer to the corresponding entry in spectra_metrics. As such, each entry contains information from a single dataOrigin of a Spectra or Chromatograms object.

The function transformIntoMzQC is a helper function within calculateMetricsFromSpectra and calculateMetricsFromChromatograms.

Usage

```
transformIntoMzQC(spectra_metrics)
```

Arguments

spectra_metrics
: list of named vector

Details

The MzQCmzQC object will only contain those quality metrics that have a corresponding attribute with a [PSI:MS] identifier. The matching is done via the names of each vector in spectra_metrics.

The Field "version" is set to the current version of the rmzqc package.

The entry of "MzQCanalysisSoftware" is filled with the [PSI:MS] id of MsQuality ("MS:") and the version is taken from packageDescription("MsQuality")[["Version"]].

Value

list containing as entries MzQCmzQC objects for each Spectra with same dataOrigin

Author(s)

Thomas Naake, Johannes Rainer

Examples

```
library(MsDataHub)
library(Spectra)
library(MsQuality)
register(SerialParam())


## define file names containing spectra data for the samples
sciex_file <- c(MsDataHub::X20171016_POOL_POS_1_105.134.mzML(),
   MsDataHub::X20171016_POOL_POS_3_105.134.mzML())

## import the data and assign it to the spectra object
spectra <- Spectra(sciex_file)

## define the quality metrics to be calculated
metrics <- c("areaUnderTic", "chromatographyDuration", "msSignal10xChange")

## obtain the spectra_metrics object
f <- dataOrigin(spectra)
f_unique <- unique(f)
```

```
spectra_metrics <- bplapply(f_unique, function(f_unique_i) {
     calculateMetricsFromOneSampleSpectra(
         spectra = spectra[f == f_unique_i], metrics = metrics)
     }, BPPARAM = bpparam())
names(spectra_metrics) <- f_unique

## transform to mzQC format
transformIntoMzQC(spectra_metrics)
```

`xicFwhm` *Full Width at Half Maximum (FWHM) per Chromatogram*

Description

The function ‘xicFwhm‘ calculates the Full Width at Half Maximum (FWHM) for each chromatogram in the ‘Chromatograms‘ object.

The metric is calculated as follows (per chromatogram):
(1) the retention time and intensity values are extracted,
(2) if `peakBoundary` is provided, the data is subset to the peak region,
(3) the maximum intensity and its index are determined,
(4) the half-maximum intensity (50% of maximum) is calculated,
(5) the left and right crossing points where intensity equals the half-maximum are found by linear interpolation,
(6) the FWHM is returned as the difference between the right and left crossing retention times.

Usage

```
xicFwhm(chromatograms, msLevel = integer(), peakBoundary = NULL, ...)
```

Arguments

`chromatograms`	‘Chromatograms‘ object
`msLevel`	‘integer‘ defining the MS level(s) to filter for. If empty (default), no filtering is performed.
`peakBoundary`	optional peak boundary: either a ‘matrix‘ with columns ‘left_boundary‘ and ‘right_boundary‘ (one row per chromatogram), or a ‘numeric(2)‘ named vector applied to all chromatograms.
`...`	further arguments

Details

This metric is analogous to MS:4000051 (XIC-FWHM quantiles).

If `peakBoundary` is provided, the calculation is restricted to the peak region defined by those boundaries. `peakBoundary` can be a ‘matrix‘ (one row per chromatogram, as returned by ‘peakBoundary()‘) or a ‘numeric(2)‘ vector applied to all chromatograms.

Value

‘numeric‘ of length equal to ‘length(chromatograms)‘. Returns ‘NA‘ for chromatograms where FWHM cannot be calculated.

Author(s)

Philippine Louail

Examples

```
library(Chromatograms)
cdata <- data.frame(
    msLevel = c(1L, 1L),
    mz = c(112.2, 123.3),
    dataOrigin = c("mem1", "mem1")
)
pdata <- list(
    data.frame(rtime = c(2.1, 2.5, 3.0, 3.4, 3.9),
               intensity = c(100, 250, 400, 300, 150)),
    data.frame(rtime = c(5.1, 5.8, 6.3, 6.9, 7.5),
               intensity = c(80, 500, 1200, 600, 120))
)
chr <- Chromatograms(ChromBackendMemory(), chromData = cdata, peaksData = pdata)
xicFwhm(chr)

## Use pre-computed peak boundaries for efficiency
pb <- peakBoundary(chr)
xicFwhm(chr, peakBoundary = pb)
```

Index