Package: GCIMS (via r-universe)

June 30, 2024

Type Package

Title Pre-processing for GC-IMS data

Version 0.1.1

Encoding UTF-8

Date 2024-04-29

Description A package for the analysis of ion mobility spectrometry (IMS) measurements, as well as samples from multicapillary columns coupled with IMS (MCC-IMS) and gas chromatography coupled to ion mobility spectrometry (GC-IMS). The package provides a complete workflow for the analysis, importing the data, preprocessing the spectra as well as classification and regression techniques for the modelling of the spectra. The package also includes visualization helpers, to represent topographic plots, extracted and total ion chromatograms and IMS spectra.

Depends R(>= 4.2.0)

License MIT + file LICENSE

Imports Biobase, BiocParallel (>= 1.30.4), BiocGenerics, cli, cluster, digest, dplyr (>= 1.0.2), ggplot2 (>= 3.3.2), glue, MassSpecWavelet (>= 1.63.5), mdendro, methods, ProtGenerics (>= 1.24.0), ptw (>= 1.9-10), purrr (>= 0.3.4), R6, reshape2 (>= 1.1.4), rlang, S4Vectors, sgolay, signal (>= 0.7-6), stats (>= 4.0.2), stringr, tibble, tidyr (>= 1.1.2), utils (>= 4.0.2), yaml

Suggests BiocStyle, cowplot, curl, farver, httr, knitr, labeling, magick, Matrix, plotly, png, pow, pracma, rmarkdown, scales, testthat (>= 3.0.0), viridisLite

biocViews Software, Preprocessing, Visualization, Classification, Cheminformatics, Metabolomics, DataImport

RoxygenNote 7.3.1

VignetteBuilder knitr

Config/testthat/edition 3

Roxygen list(markdown = TRUE)

Remotes sipss/pow

Repository https://zeehio.r-universe.dev

RemoteUrl https://github.com/sipss/GCIMS

RemoteRef HEAD

RemoteSha 6368ffa6787ac7a2d29ad5c89e132ce88b0f3791

Contents

`add_peaklist_rect` *Add peak list rectangles to a raw plot*

Description

Add peak list rectangles to a raw plot

Usage

```
add_peaklist_rect(
  plt,
  peaklist,
  color_by = NULL,
  col_prefix = "",
  pdata = NULL,
  palette = P40
)
```

Arguments

plt	The output of `plot()` when applied to a GCIMSSample
peaklist	A data frame with at least the columns: `dt_min_ms`, `dt_max_ms`, `rt_min_s`, `rt_max_s` and optionally additional columns (e.g. the column given to `color_by`)
color_by	A character with a column name of `peaklist`. Used to color the border of the added rectangles
col_prefix	After clustering, besides `dt_min_ms`, we also have
pdata	A phenotype data data frame, with a SampleID column to be merged into peaklist so color_by can specify a phenotype `freesize_dt_min_ms`. Use `col_prefix = "freesize_"` to plot the `freesize` version
palette	A character vector with color names to use drawing the rectangles. Use `NULL` to let `ggplot2` set the defaults.

Details

If `peaklist` includes `dt_apex_ms` and `rt_apex_s` a cross will be plotted on the peak apex.

Value

The given `plt` with rectangles showing the ROIs and crosses showing the apexes

Examples

```
dt <- 1:10
rt <- 1:10
int <- matrix(0.0, nrow = length(dt), ncol = length(rt))

int[2, 4:8] <- c(.5, .5, 1, .5, 0.5)
int[3, 4:8] <- c(0.5, 2, 2, 2, 0.5)
int[4, 4:8] <- c(1, 2, 5, 2, 1)
int[5, 4:8] <- c(0.5, 2, 2, 2, 0.5)
int[6, 4:8] <- c(.5, .5, 1, .5, 0.5)

dummy_obj <-GCIMSSample(
  drift_time = dt,
  retention_time = rt,
  data = int
)
```

```
plt <- plot(dummy_obj)

# Add a rectangle on top of the plot
rect <- data.frame(
  dt_min_ms = 2.75,
  dt_max_ms = 5.6,
  rt_min_s = 4.6,
  rt_max_s = 7.4
)

add_peaklist_rect(
  plt = plt,
  peaklist = rect
)
```

`align,GCIMSDataset-method`

Align a GCIMS dataset

Description

The alignment uses a multiplicative correction in drift time and a Parametric Time Warping correction in retention time

Usage

```
## S4 method for signature 'GCIMSDataset'
align(
  object,
  method_rt = "ptw",
  align_dt = TRUE,
  align_ip = TRUE,
  reference_sample_idx = NULL,
  ...
)
```

Arguments

`object`	A GCIMSDataset object, modified in-place
`method_rt`	Method for alignment, should be "ptw" or "pow" if pow is selected the package "pow" must be installed, to do so visit: https://github.com/sipss/pow
`align_dt`	if `TRUE` the drift time axis will be aligned using a multiplicative correction
`align_ip`	if `TRUE` a multiplicative correction will be done in retention time before applying the other algorithm
`reference_sample_idx`	One number, the index of the sample to use as reference for the alignment in retention time, if NULL the reference will be calculated automatically depending on the method
`...`	additional parameters for POW alignment

Value

The modified GCIMSDataset

`align,GCIMSSample-method` *Align a GCIMSSample object, in retention time*

Description

Align a GCIMSSample object, in retention time

Usage

```
## S4 method for signature 'GCIMSSample'
align(object, method_rt, ric_ref, ric_ref_rt, ...)
```

Arguments

`object`	A GCIMSSample object
`method_rt`	Method for alignment, should be "ptw" or "pow"
`ric_ref`	The reference Reverse Ion Chromatogram
`ric_ref_rt`	The retention times corresponding to `ric_ref`
`...`	Additional arguments passed on to the alignment method.

Value

The modified GCIMSSample

`alignDt` *Align a GCIMSSample in drift time with a multiplicative correction*

Description

Align a GCIMSSample in drift time with a multiplicative correction

Usage

```
alignDt(object, rip_ref_ms)
```

Arguments

`object`	A GCIMSSample object
`rip_ref_ms`	The position of the RIP in ms

Value

The modified GCIMSSample

alignPlots *Plots to interpret alignment results*

Description

Plots to interpret alignment results

Usage

```
alignPlots(object)
```

Arguments

object	A GCIMSDataset object, modified in-place

Value

A list with plots created with ggplot2.

alignRt_ip *Align a GCIMSSample in retention time with a multiplicative correction*

Description

Align a GCIMSSample in retention time with a multiplicative correction

Usage

```
alignRt_ip(object, min_start, rt_ref)
```

Arguments

object	A GCIMSSample object
min_start	minimun injection point, to calculate where to begin the spectrums and cut as few points as posible
rt_ref	retention time reference

Value

The modified GCIMSSample

`alignRt_pow` *Align a GCIMSSample in retention time with parametric optimized warping*

Description

Align a GCIMSSample in retention time with parametric optimized warping

Usage

```
alignRt_pow(
  object,
  ric_ref,
  ric_ref_rt,
  lambdas = pracma::logspace(-2, 4, 31),
  p = 10,
  max_it = 5000,
  lambda1 = 10^6
)
```

Arguments

`object`	A GCIMSSample object
`ric_ref`	The reference Reverse Ion Chromatogram
`ric_ref_rt`	The retention times corresponding to `ric_ref`
`lambdas`	a vector with the penalties to test the POW
`p`	By default `10`, meaning to use one every `10` points to validate.
`max_it`	Maximum number of iterations
`lambda1`	Regularization parameter for second derivative of warp

Value

The modified GCIMSSample

`alignRt_ptw` *Align a GCIMSSample in retention time using parametric time warping*

Description

Align a GCIMSSample in retention time using parametric time warping

Usage

```
alignRt_ptw(object, ric_ref, ric_ref_rt, ploynomial_order = 5)
```

Arguments

object	A GCIMSSample object
ric_ref	The reference Reverse Ion Chromatogram
ric_ref_rt	The retention times corresponding to ric_ref
ploynomial_order	maximum order of the polynomial to be used by default 5

Value

The modified GCIMSSample

---

as.data.frame.GCIMSSample

Turn the intensity matrix into a data frame

Description

Turn the intensity matrix into a data frame

Usage

```
## S3 method for class 'GCIMSSample'
as.data.frame(
  x,
  row.names = NULL,
  optional = FALSE,
  dt_range = NULL,
  rt_range = NULL,
  dt_idx = NULL,
  rt_idx = NULL,
  ...
)
```

Arguments

x	A GCIMSSample object
row.names	NULL or a character vector giving the row names for the data frame. Missing values are not allowed.
optional	logical. If TRUE, setting row names and converting column names (to syntactic names: see make.names) is optional. Note that all of R's **base** package as.data.frame() methods use optional only for column names treatment, basically with the meaning of data.frame(*, check.names = !optional). See also the make.names argument of the matrix method.

dt_range	The minimum and maximum drift times to extract (length 2 vector)
rt_range	The minimum and maximum retention times to extract (length 2 vector)
dt_idx	A numeric vector with the drift time indices to extract (or a logical vector of the length of drift time)
rt_idx	A numeric vector with the retention time indices to extract (or a logical vector of the length of retention time)
...	unused

Value

A data frame with dt_ms, rt_s and Intensity columns

---

clusterPeaks *Group peaks in clusters*

Description

Peak grouping function, exposing several options useful for benchmarking.

Usage

```
clusterPeaks(
  peaks,
  ...,
  distance_method = "euclidean",
  dt_cluster_spread_ms = 0.1,
  rt_cluster_spread_s = 20,
  distance_between_peaks_from_same_sample = 100,
  clustering = list(method = "hclust"),
  verbose = FALSE
)
```

Arguments

peaks	A data frame with at least the following columns: • "UniqueID" A unique ID for each peak • "SampleID" The sample ID the peak belongs to • "dt_apex_ms", "rt_apex_s" The peak positions • "dt_max_ms", "dt_min_ms", "rt_max_s", "rt_min_s" (for filtering outlier peaks based on their size)
...	Ignored. All other parameters beyond peaks should be named
distance_method	A string. One of the distance methods from stats::dist, "sd_scaled_euclidean" or "mahalanobis"

`dt_cluster_spread_ms`, `rt_cluster_spread_s`
: The typical spread of the clusters. Used for scaling. dimensions when computing distances. When `clustering$method` is `"hclust"`, these spreads are used to cut cluster sizes.

`distance_between_peaks_from_same_sample`
: The distance between two peaks from the same sample will be set to `distance_between_peaks_from_sa`

`clustering`
: A named list with "method" and the supported method, as well as further options. For `method = "kmedoids"`, you must provide `Nclusters`, with either the number of clusters to use in the kmedoids algorithm (cluster::pam) or the string `"max_peaks_sample"` to use the maximum number of detected peaks per sample.
For `method = "hclust"`, you can provide `hclust_method`, with the `method` passed to `mdendro::linkage()`.

`verbose`
: logical, to control printing in the function

Value

A list with :

- `peak_list_clustered`: The peak list with a "cluster" column
- `cluster_stats`: Cluster statistics (cluster size...)
- `dist`: peak to peak distance object
- `extra_clustering_info`: Arbitrary clustering extra information, that depends on the clustering method

Examples

```
peak_list_fn <- system.file("extdata", "peak_list.rds", package = "GCIMS")
peak_list <- readRDS(peak_list_fn)

peak_clustering  <- clusterPeaks(peak_list)
```

---

create_annotations_table

Create a table for defining dataset annotations

Description

To process an entire dataset, we need a table that describes the samples, and you may want to add for the analysis.

Usage

```
create_annotations_table(
  samples_dir,
  glob = c("*.mea", "*.mea.gz"),
  recursive = TRUE,
  verbose = TRUE
)
```

Arguments

samples_dir	A directory that contains samples
glob	One or more globs for sample extensions. See the examples.
recursive	Whether to look for samples into samples_dir subdirectories
verbose	If set to TRUE it prints instructions

Details

The table needs to have at least two columns, one with the file name of the sample (FileName) and another one with the sample name (SampleID), that you can set as you like. Besides, you can add additional columns with any metadata/annotations/phenotypes you may consider relevant.

This function will help you list all the samples from a directory to a table. The example below will show you how to save this table as an Excel or CSV file, for you to conveniently modify it and how you can read it back for further analysis.

Value

A data frame with the SampleID and FileName columns

Examples

```
# How to create the annotations table:
#
# First you must tell R where your samples are. Please change "samples_dir"
# below to your samples directory. On Windows you can use:
#     samples_dir <- choose.dir(getwd(), "Choose the folder where the samples are")
# On other systems you can use:
#     library(tcltk)
#     samples_dir <- tclvalue(tkchooseDirectory())
# In this example we use a folder with some demo files:
samples_dir <- system.file("extdata", "sample_formats", package = "GCIMS")
# Then you need to provide the extension to look into. If you use `glob = "*.*"` you
# will catch all files and you can filter the annotations table afterwards:
annotations <- create_annotations_table(samples_dir, glob = "*.mea.gz")
# You can write the annotations table to an Excel or a CSV file:
# For Excel you may need to install the writexl package:
#    install.packages("writexl")
# And then you can use:
#    writexl::write_xlsx(annotations, "annotations.xlsx")
# For csv just use:
#    write.csv(annotations, "annotations.csv")
#
# Modify manually the excel or CSV file
#
# Read it again into R as follows:
#
# For Excel you may need to install the readxl package:
#    install.packages("readxl")
# And then you can use:
#    annotations <- readxl::read_excel("annotations.xlsx")
```

```
# For csv just use:
#    annotations <- read.csv("annotations.csv")
```

`cubic_root_trans` *Cubic root transformation*

Description

A scales transformation to be used with ggplot2.

Usage

```
cubic_root_trans()
```

Details

This function is exported because we are using it in vignettes, but it may become unavailable in future versions

Value

A scale transformation object of name "cubic_root"

Examples

```
library(ggplot2)
x <- 1:10
y <- x^3
df <- data.frame(x = x, y = y)
ggplot(data.frame(x=x, y=y)) +
 geom_point(aes(x = x, y = y)) +
  scale_y_continuous(trans=cubic_root_trans())
```

`decimate,GCIMSDataset-method` *Decimate a GCIMS dataset keeping 1 out of n points*

Description

Decimate a GCIMS dataset keeping 1 out of n points

Usage

```
## S4 method for signature 'GCIMSDataset'
decimate(object, dt_factor = 1L, rt_factor = 1L)
```

Arguments

`object`	A GCIMSDataset object, modified in-place
`dt_factor`	Keep one every `dt_factor` measurement points in drift time
`rt_factor`	Keep one every `rt_factor` measurement points in retention time

Value

The modified GCIMSDataset

`decimate,GCIMSSample-method` *Decimates a GCIMS sample*

Description

This method assumes that the sample has been low-pass filtered to avoid aliasing issues

Usage

```
## S4 method for signature 'GCIMSSample'
decimate(object, dt_factor = 1L, rt_factor = 1L)
```

Arguments

`object`	A GCIMSSample object
`dt_factor`	Keep one every `dt_factor` measurement points in drift time
`rt_factor`	Keep one every `rt_factor` measurement points in retention time

Value

The modified GCIMSSample

`DelayedOperation` *Create a DelayedOperation object*

Description

Delayed operations enables us to process our samples faster on big datasets. See the details section for details on how they work.

Usage

```
DelayedOperation(
  name,
  fun = NULL,
  params = list(),
  params_iter = list(),
  fun_extract = NULL,
  fun_aggregate = NULL
)
```

Arguments

name	A named for de delayed operation, only used for printing.
fun	A function that takes a sample and returns a modified sample
params	A named list with additional arguments to be passed to function
params_iter	A named list with additional arguments to be passed to function. Compared to params, each argument must be a named list of length the number of samples, so each sample will receive its corresponding parameter according to its name
fun_extract	A function that takes a modified sample and returns an extracted object.
fun_aggregate	A function that takes a dataset and a list of extracted objects and returns a modified dataset.

Details

Let's say we have a pipeline with two actions (e.g. smooth() and detectPeaks()). and we want to apply it to a dataset with two samples (e.g s1, s2).

This is a simple pseudocode to execute all actions in all samples. The code is written so you can get an idea of how :

```
dataset = list(s1, s2)
actions = list(smooth, detectPeaks)
for (action in actions) {
  for (i in seq_along(dataset)) {
      dataset[[i]] <- action(dataset[[i]])
  }
}
```

When the dataset is big, samples are stored in disk, and loaded/saved when used:

```
dataset = list(s1, s2)
actions = list(smooth, detectPeaks)
for (action in actions) {
  for (i in seq_along(dataset)) {
      sample <- read_from_disk(i)
      sample <- action(sample)
      save_to_disk(sample)
  }
}
```

So actually, we can avoid "saving and loading" by changing the loop order:

```
dataset = list(s1, s2)
actions = list(smooth, detectPeaks)
for (i in seq_along(dataset)) {
  sample <- read_from_disk(i)
  for (action in actions) {
      sample <- action(sample)
  }
  save_to_disk(sample)
}
```

This requires that when we apply an operation to the dataset, the operation is delayed, so we can stack many delayed operations and run them all at once.

The DelayedOperation class allows us to store all pending actions and run them afterwards when the data is needed.

Besides, samples can be processed in parallel if enough cores and RAM are available.

The DelayedOperation class also considers that sometimes we want to extract some information from each sample (e.g. the Reverse Ion Chromatogram) and build some matrix with the Reverse Ion Chromatograms of all samples. It changes the loops above, so after each action modifies each sample, we can extract something out of the sample and save it. After all actions have executed, we can aggregate the results we have extracted and save them into the dataset. This is used for instance in the `getRIC()` implementation, to extract the RIC from each sample and afterwards aggregate it into a matrix. This is implemented here with the `fun_extract` and `fun_aggregate` functions.

Value

A DelayedOperation object

`DelayedOperation-class` *Delayed Operation class*

Description

DelayedOperation is an S4 class to store a delayed operation

Delayed operations are not applied to the dataset immediately, but rather when some data from the dataset is required. When working on large datasets, keeping all samples in RAM may be impossible, and the DelayedDatasetDisk architecture becomes convenient, where samples are stored in a directory, loaded processed and saved individually.

Under such arquitecture, it is more efficient to load a sample, run as many operations as possible on it and save the sample, instead of loading a sample, running one operation, saving the sample.

See how to create such delayed operations and more details at `vignette("creating-a-workflow-step", package = "GCIMS")`.

Slots

`name` A named for de delayed operation, only used for printing.

`fun` A function that takes a sample object and returns a sample object, usually with some change (filtered,...)

`params` A named list with additional arguments to be passed to `fun`

`params_iter` A named list with additional arguments to be passed to `fun`. Compared to `params`, each argument must be a named list of length the number of samples, so each sample will receive its corresponding parameter according to its name

`fun_extract` A function that takes the modified sample object returned by `fun` and extracts some component out of it. This component will be stored in the dataset for faster access.

`fun_aggregate` A function that takes a dataset object and a list of extracted results (the output of all `fun_extract` calls) and modifies the dataset.

`download_three_ketones_dataset` *Download three samples (6-ketone mixture)*

Description

This function downloads three samples in .mea.gz format. It is useful to run the introductory vignette.

Usage

```
download_three_ketones_dataset(outdir = "2021-mixture-six-ketones-demo")
```

Arguments

`outdir`	Name of the directory where the samples will be saved

Value

Nothing (Files are created in the given folder)

Examples

```
## Not run:
download_three_ketones_dataset(outdir = "sample_dataset")
list.files("sample_dataset")

## End(Not run)
```

`dtime,GCIMSChromatogram-method`

Get the drift time of the chromatogram

Description

Get the drift time of the chromatogram

Usage

```
## S4 method for signature 'GCIMSChromatogram'
dtime(object)
```

Arguments

`object`	A GCIMSChromatogram

Value

The drift time where this chromatogram was extracted from (in ms)

See Also

Other GCIMSChromatogram: `GCIMSChromatogram`, `GCIMSChromatogram-class`, `estimateBaseline,GCIMSChromatogra`
`findPeaks,GCIMSChromatogram-method`, `intensity,GCIMSChromatogram-method`, `rtime,GCIMSChromatogram-metho`
`smooth,GCIMSChromatogram-method`

`dtime,GCIMSDataset-method`

Get A reference drift time vector for the dataset

Description

Get A reference drift time vector for the dataset

Usage

```
## S4 method for signature 'GCIMSDataset'
dtime(object, sample = NULL)
```

Arguments

`object`	A GCIMSDataset
`sample`	A number or a string with the sample index or name. If `NULL`, the reference drift time is returned

Value

a drift time vector

`estimateBaseline,GCIMSChromatogram-method`
Estimate the baseline of a GCIMS Chromatogram using a connect local minima algorithm

---

Description

The baseline is estimated by connecting local minima and interpolating from those. The local minima are identified as "the minima in each region of length x" The length of the regions are given in seconds in the `region_s` parameter.

Usage

```
## S4 method for signature 'GCIMSChromatogram'
estimateBaseline(object, rt_length_s)

## S4 method for signature 'GCIMSChromatogram'
baseline(object, rt_range = NULL, rt_idx = NULL, .error_if_missing = TRUE)

## S4 replacement method for signature 'GCIMSChromatogram'
baseline(object) <- value
```

Arguments

`object`	A GCIMSChromatogram object
`rt_length_s`	The length of the baseline region. It should be comparable or longer than the peak width
`rt_range`	The minimum and maximum retention times to extract (length 2 vector)
`rt_idx`	A numeric vector with the retention time indices to extract (or a logical vector of the length of retention time)
`.error_if_missing`	A logical. If `TRUE` (default) give an error if baseline is not estimated. Returns `NULL` otherwise.
`value`	A vector with a baseline of the same length as `intensity(object)`

Value

The modified GCIMSChromatogram

Functions

- `baseline(GCIMSChromatogram)`: Get the baseline
- `baseline(GCIMSChromatogram) <- value`: Set the baseline

See Also

Other GCIMSChromatogram: GCIMSChromatogram, GCIMSChromatogram-class, dtime,GCIMSChromatogram-method, findPeaks,GCIMSChromatogram-method, intensity,GCIMSChromatogram-method, rtime,GCIMSChromatogram-method, smooth,GCIMSChromatogram-method

---

`estimateBaseline,GCIMSDataset-method`
Estimate the baseline of a GCIMS Sample using a connect local minima algorithm

---

Description

Estimate the baseline of a GCIMS Sample using a connect local minima algorithm

Usage

```
## S4 method for signature 'GCIMSDataset'
estimateBaseline(
  object,
  dt_peak_fwhm_ms,
  dt_region_multiplier,
  rt_length_s,
  remove = TRUE
)
```

Arguments

`object`	A GCIMSDataset object, modified in-place
`dt_peak_fwhm_ms`	Full Width at Half Maximum in milliseconds. Used to determine the length of the regions where local minima are searched.
`dt_region_multiplier`	A multiplier to calculate the region
`rt_length_s`	The length of the baseline region. It should be comparable or longer than the peak width
`remove`	A boolean, if TRUE it removes the baseline from the intensity

Value

The modified GCIMSDataset

`estimateBaseline,GCIMSSample-method`
Estimate the baseline of a GCIMS Sample using a connect local minima algorithm

Description

The baseline is estimated by connecting local minima and interpolating from those. The local minima are identified as "the minima in each region of length x" The length of the regions are estimated as `fwhm * a multiplier / 2.3482`. This assumes it's several times

Usage

```
## S4 method for signature 'GCIMSSample'
estimateBaseline(
  object,
  dt_peak_fwhm_ms,
  dt_region_multiplier,
  rt_length_s,
  remove = TRUE
)

## S4 method for signature 'GCIMSSample'
baseline(
  object,
  dt_range = NULL,
  rt_range = NULL,
  dt_idx = NULL,
  rt_idx = NULL,
  .error_if_missing = TRUE
)

## S4 replacement method for signature 'GCIMSSample'
baseline(object) <- value
```

Arguments

`object`	A GCIMSSample object
`dt_peak_fwhm_ms`	Full Width at Half Maximum in milliseconds. Used to determine the length of the regions where local minima are searched.
`dt_region_multiplier`	A multiplier to calculate the region
`rt_length_s`	The length of the baseline region. It should be comparable or longer than the peak width
`remove`	A boolean, if TRUE it removes the baseline from the intensity

dt_range	The minimum and maximum drift times to extract (length 2 vector)
rt_range	The minimum and maximum retention times to extract (length 2 vector)
dt_idx	A numeric vector with the drift time indices to extract (or a logical vector of the length of drift time)
rt_idx	A numeric vector with the retention time indices to extract (or a logical vector of the length of retention time)
.error_if_missing	A logical. If TRUE, raise error if baseline has not been estimated. If FALSE returns NULL instead.
value	A matrix with the sample baseline of the same dimensions as dim(object)

Value

The modified GCIMSSample

Functions

- baseline(GCIMSSample): Get the baseline
- baseline(GCIMSSample) <- value: Set the baseline

---

estimateBaseline,GCIMSSpectrum-method

Estimate the baseline of a GCIMS Spectrum using a connect local minima algorithm

---

Description

The baseline is estimated by connecting local minima and interpolating from those. The local minima are identified as "the minima in each region of length x" The length of the regions are estimated as fwhm * a multiplier / 2.3482. This assumes it's several times

Usage

```
## S4 method for signature 'GCIMSSpectrum'
estimateBaseline(object, dt_peak_fwhm_ms, dt_region_multiplier = 12)

## S4 method for signature 'GCIMSSpectrum'
baseline(object, dt_range = NULL, dt_idx = NULL, .error_if_missing = TRUE)

## S4 replacement method for signature 'GCIMSSpectrum'
baseline(object) <- value
```

Arguments

`object`	A GCIMSSpectrum object
`dt_peak_fwhm_ms`	Full Width at Half Maximum in milliseconds. Used to determine the length of the regions where local minima are searched.
`dt_region_multiplier`	A multiplier to calculate the region
`dt_range`	The minimum and maximum drift times to extract (length 2 vector)
`dt_idx`	A numeric vector with the drift time indices to extract (or a logical vector of the length of drift time)
`.error_if_missing`	A logical. If `TRUE` (default) give an error if baseline is not estimated. Returns `NULL` otherwise.
`value`	A vector with a baseline of the same length as `intensity(object)`

Value

The modified GCIMSSpectrum

Functions

- `baseline(GCIMSSpectrum)`: Get the baseline
- `baseline(GCIMSSpectrum) <- value`: Set the baseline

`filterDt,GCIMSDataset-method`

Filter GCIMSDataset samples by drift time

Description

Filter GCIMSDataset samples by drift time

Usage

```
## S4 method for signature 'GCIMSDataset'
filterDt(object, dt_range)
```

Arguments

`object`	A GCIMSDataset object
`dt_range`	The minimum and maximum drift times to extract (length 2 vector)

Value

The given object, with a delayed operation to filter retention times

Examples

```
base_dir <- system.file("extdata", "sample_formats", package = "GCIMS")
annot <- data.frame(SampleID = "Sample1", FileName = "small.mea.gz")
dataset <- GCIMSDataset$new(annot, base_dir)
filterDt(dataset, dt_range = c(5, 10))
```

---

filterDt,GCIMSSample-method

Filter GCIMSSample samples by drift time

Description

Filter GCIMSSample samples by drift time

Usage

```
## S4 method for signature 'GCIMSSample'
filterDt(object, dt_range)
```

Arguments

object	A GCIMSSample object
dt_range	The minimum and maximum drift times to extract (length 2 vector)

Value

A subset of the sample, only in the selected dt_range

Examples

```
sample_file <- system.file("extdata", "sample_formats", "small.mea.gz", package = "GCIMS")
s <- read_mea(sample_file)
s <- filterDt(s, dt_range = c(5, 9.5))
```

---

filterRt,GCIMSDataset-method

Filter GCIMSDataset samples by retention time

Description

Filter GCIMSDataset samples by retention time

Usage

```
## S4 method for signature 'GCIMSDataset'
filterRt(object, rt_range)
```

Arguments

object	A GCIMSDataset object
rt_range	The minimum and maximum retention times to extract (length 2 vector)

Value

The given object, with a delayed operation to filter retention times

Examples

```
base_dir <- system.file("extdata", "sample_formats", package = "GCIMS")
annot <- data.frame(SampleID = "Sample1", FileName = "small.mea.gz")
dataset <- GCIMSDataset$new(annot, base_dir)
filterRt(dataset, rt_range = c(5, 50))
```

---

filterRt,GCIMSSample-method

Filter GCIMSSample samples by retention time

Description

Filter GCIMSSample samples by retention time

Usage

```
## S4 method for signature 'GCIMSSample'
filterRt(object, rt_range)
```

Arguments

object	A GCIMSSample object
rt_range	The minimum and maximum retention times to extract (length 2 vector)

Value

A subset of the sample, only in the selected `rt_range`

Examples

```
sample_file <- system.file("extdata", "sample_formats", "small.mea.gz", package = "GCIMS")
s <- read_mea(sample_file)
s <- filterRt(s, rt_range = c(5, 50))
```

`findPeaks` *Find Peaks in an object*

Description

Find Peaks in an object

Usage

```
findPeaks(object, ...)
```

Arguments

`object`	An object to find peaks on
`...`	Additional arguments for downstream methods

Value

The object, with found peaks

`findPeaks,GCIMSChromatogram-method`
Peak detection for a GCIMSChromatogram

Description

Peak detection for a GCIMSChromatogram

Usage

```
## S4 method for signature 'GCIMSChromatogram'
findPeaks(object, ...)
```

Arguments

`object`	A GCIMSChromatogram object
`...`	Arguments passed on to `findPeaksImpl1D` `verbose` If `TRUE` information will be printed on screen `length_in_xunits` Length of the filter used to compute the second derivative. See details. `peakwidth_range_xunits` A vector of length 2 with the minimum and maximum peak width. See details. `peakDetectionCWTParams` Additional parameters to `MassSpecWavelet::peakDetectionCWT()`. See details.

`extension_factor` A number to extend the ROIs beyond their default size

`iou_overlap_threshold` A number, between 0 and 1. Pairs of ROIs with an intersection over union larger than this threshold are merged.

`debug` If TRUE, return as well the debug information

Value

The modified GCIMSChromatogram, with a peak list

See Also

Other GCIMSChromatogram: `GCIMSChromatogram`, `GCIMSChromatogram-class`, `dtime,GCIMSChromatogram-method`, `estimateBaseline,GCIMSChromatogram-method`, `intensity,GCIMSChromatogram-method`, `rtime,GCIMSChromatogram-method`, `smooth,GCIMSChromatogram-method`

---

findPeaks,GCIMSDataset-method

Peak detection on the GCIMS dataset

Description

Peak detection on the GCIMS dataset

Usage

```
## S4 method for signature 'GCIMSDataset'
findPeaks(object, ...)
```

Arguments

`object`	A GCIMSDataset object, modified in-place
`...`	Arguments passed on to `findPeaksImpl`
	`verbose` If `TRUE` information will be printed on screen
	`dt_length_ms,rt_length_s` Length of the filters used to compute the second derivative. See details.
	`dt_peakwidth_range_ms,rt_peakwidth_range_s` A vector of length 2 with the minimum and maximum peak width. See details
	`exclude_rip` Whether to exclude ROIs with a drift time apex smaller than the RIP drift time end.
	`iou_overlap_threshold` A number, between 0 and 1. Pairs of ROIs with an intersection over union larger than this threshold are merged.
	`debug_idx` A list with two numeric vectors named `dt` and `rt` each of them having a the indices to where debug info is kept

Value

The modified GCIMSDataset, with a peak list

`findPeaks,GCIMSSample-method`

Peak detection for a GCIMSSample

Description

Peak detection for a GCIMSSample

Usage

```
## S4 method for signature 'GCIMSSample'
findPeaks(object, ...)
```

Arguments

`object`	A GCIMSSample object
`...`	Arguments passed on to `findPeaksImpl`
	`verbose` If `TRUE` information will be printed on screen
	`dt_length_ms,rt_length_s` Length of the filters used to compute the second derivative. See details.
	`dt_peakwidth_range_ms,rt_peakwidth_range_s` A vector of length 2 with the minimum and maximum peak width. See details
	`exclude_rip` Whether to exclude ROIs with a drift time apex smaller than the RIP drift time end.
	`iou_overlap_threshold` A number, between 0 and 1. Pairs of ROIs with an intersection over union larger than this threshold are merged.
	`debug_idx` A list with two numeric vectors named `dt` and `rt` each of them having a the indices to where debug info is kept

Value

The modified GCIMSSample, with a peak list

`findPeaks,GCIMSSpectrum-method`

Peak detection for a GCIMSSpectrum

Description

Peak detection for a GCIMSSpectrum

Usage

```
## S4 method for signature 'GCIMSSpectrum'
findPeaks(object, ...)
```

Arguments

object	A GCIMSSpectrum object
...	Arguments passed on to findPeaksImpl1D
	verbose If TRUE information will be printed on screen
	length_in_xunits Length of the filter used to compute the second derivative. See details.
	peakwidth_range_xunits A vector of length 2 with the minimum and maximum peak width. See details.
	peakDetectionCWTParams Additional parameters to MassSpecWavelet::peakDetectionCWT(). See details.
	extension_factor A number to extend the ROIs beyond their default size
	iou_overlap_threshold A number, between 0 and 1. Pairs of ROIs with an intersection over union larger than this threshold are merged.
	debug If TRUE, return as well the debug information

Value

The modified GCIMSSpectrum, with a peak list

GCIMS-generics *GCIMS Generics*

Description

Generics defined at the GCIMS package. We are open to moving them to an existing generics-only package if you need so.

Usage

```
dtime(object, ...)

getTIS(object, ...)

getRIC(object, ...)

plotTIS(object, ...)

plotRIC(object, ...)

filterDt(object, ...)

decimate(object, ...)

align(object, ...)
```

```
prealign(object, ...)

estimateBaseline(object, ...)

baseline(object, ...)

baseline(object) <- value

integratePeaks(object, ...)
```

Arguments

object	An object to get the baseline
...	Additional arguments for downstream methods
value	baseline to set

Value

A numeric vector with the drift time

The Total Ion Spectrum as a numeric vector or a matrix (depending if the object is one sample or several)

The Reverse Ion Chromatogram, as a numeric vector or a matrix (depending if the object is one sample or several)

A plot

A plot

The object, modified

The object, modified

The object, modified

The object, modified

The object, with a baseline estimated

The baseline of the object

The object

The object, with integrated peaks

Functions

- `dtime()`: Get drift time vector
- `getTIS()`: Get the Total Ion Spectrum
- `getRIC()`: Get the Reverse Ion Chromatogram
- `plotTIS()`: Plot total ion spectrum
- `plotRIC()`: Plot Reverse Ion Chromatogram
- `filterDt()`: Filter in Drift time
- `decimate()`: Decimate an object

- `align()`: Align an object
- `prealign()`: Align an object in drift time
- `estimateBaseline()`: Estimate the baseline in an object
- `baseline()`: Get the baseline of an object
- `baseline(object) <- value`: Set the baseline of an object
- `integratePeaks()`: Integrate peaks of an object

Examples

```
x <- GCIMSSample(
  drift_time = 1:2,
  retention_time = 1:3,
  data = matrix(1:6, nrow = 2, ncol = 3)
)
dtime(x) # c(1,2)
```

`GCIMSChromatogram` *Create a GCIMSChromatogram object*

Description

Create a GCIMSChromatogram object

Usage

```
GCIMSChromatogram(
  retention_time,
  intensity,
  drift_time_idx = NA_integer_,
  drift_time_ms = NA_real_,
  description = "",
  baseline = NULL,
  peaks = NULL,
  peaks_debug_info = NULL
)
```

Arguments

`retention_time`	A numeric vector with retention times
`intensity`	A numeric vector with the corresponding intensities
`drift_time_idx`	The index or indices used to get the intensity
`drift_time_ms`	The drift times corresponding to `drift_time_idx`.
`description`	A string with a description (used as plot title, useful e.g. to know the sample it came from)

baseline — A numeric vector of the same length as intensity with the corresponding baseline, or NULL if not set. Use estimateBaseline() to estimate it, baseline() to directly access it.

peaks — A data frame with peaks, use findPeaks() to fill it, or peaks() to set/get it

peaks_debug_info — A list with arbitrary debug information from findPeaks()

Value

A GCIMSChromatogram object

See Also

Other GCIMSChromatogram: GCIMSChromatogram-class, dtime,GCIMSChromatogram-method, estimateBaseline,GCIMSChromatogram-method, findPeaks,GCIMSChromatogram-method, intensity,GCIMSChromato
rtime,GCIMSChromatogram-method, smooth,GCIMSChromatogram-method

Examples

```
GCIMSChromatogram(
  retention_time = seq(from = 0, to = 10, length.out = 200),
  intensity = 1:200
)
```

GCIMSChromatogram-class

GCIMSChromatogram class

Description

GCIMSChromatogram is an S4 class to store a GCIMS Chromatogram It can be a single chromatogram or the aggregation of several chromatograms.

Usage

```
## S3 method for class 'GCIMSChromatogram'
as.data.frame(x, row.names = NULL, optional = FALSE, ...)

## S4 method for signature 'GCIMSChromatogram'
description(object)

## S4 replacement method for signature 'GCIMSChromatogram,ANY'
description(object) <- value

## S4 method for signature 'GCIMSChromatogram'
peaks(object)
```

```
## S4 replacement method for signature 'GCIMSChromatogram'
peaks(object) <- value

## S4 method for signature 'GCIMSChromatogram,ANY'
plot(x, y, ...)
```

Arguments

x	A GCIMSChromatogram object to plot
row.names	NULL or a character vector giving the row names for the data frame. Missing values are not allowed.
optional	logical. If TRUE, setting row names and converting column names (to syntactic names: see make.names) is optional. Note that all of R's **base** package as.data.frame() methods use optional only for column names treatment, basically with the meaning of data.frame(*, check.names = !optional). See also the make.names argument of the matrix method.
...	additional arguments to be passed to or from methods.
object	A GCIMSChromatogram object
value	A data frame with the peak list
y	From the generic plot function, ignored for GCIMSChromatogram class objects

Value

The description of the chromatogram

The chromatogram object

A data frame with the peaks in the chromatogram

The GCIMSChromatogram object

A ggplot2 plot object

Functions

- as.data.frame(GCIMSChromatogram): Coerce to data frame
- description(GCIMSChromatogram): Get the description
- description(object = GCIMSChromatogram) <- value: Set the description
- peaks(GCIMSChromatogram): Get the peak list
- peaks(GCIMSChromatogram) <- value: Set the peak list
- plot(x = GCIMSChromatogram, y = ANY): plot method

Slots

retention_time A numeric vector with retention times

intensity A numeric vector with the corresponding intensities

baseline A numeric vector of the same length as intensity with the corresponding baseline, or NULL if not set. Use estimateBaseline() to estimate it, baseline() to directly access it.

`drift_time_idx` The index or indices used to get the intensity

`drift_time_ms` The drift times corresponding to `drift_time_idx`.

`description` A string with a description (used as plot title, useful e.g. to know the sample it came from)

`peaks` A data frame with peaks, use `findPeaks()` to fill it, or `peaks()` to set/get it

`peaks_debug_info` A list with arbitrary debug information from `findPeaks()`

See Also

Other GCIMSChromatogram: `GCIMSChromatogram`, `dtime,GCIMSChromatogram-method`, `estimateBaseline,GCIMSChro`
`findPeaks,GCIMSChromatogram-method`, `intensity,GCIMSChromatogram-method`, `rtime,GCIMSChromatogram-metho`
`smooth,GCIMSChromatogram-method`

`GCIMSDataset` *GCIMSDataset*

Description

GCIMSDataset is an R6 class to store a dataset.

When the dataset is created, the `on_ram` option controls whether the actual data is stored not in memory or it is read/saved from/to files as needed, so the dataset object scales with large number of samples.

Constructors:

- `GCIMSDataset$new()`
- `GCIMSDataset$new_from_list()`: Create a new GCIMSDataset from a list of samples
- `GCIMSDataset$new_from_saved_dir()`: Create a new on disk GCIMSDataset from a directory

Constructor `new_from_list()`:

Create a new GCIMSDataset object from a list of samples. Note that with this constructor `on_ram` is `TRUE` by default

Usage:

```
GCIMSDataset$new_from_list(
  samples,
  pData=NULL,
  scratch_dir = NULL,
  keep_intermediate = FALSE,
  on_ram = TRUE
)
```

Arguments:

See `GCIMSDataset$new()`

Constructor `new_from_saved_dir()`**:**

Creates a new GCIMSDataset object from a directory where a GCIMSDataset with `on_ram=FALSE` was saved.

Usage:

```
GCIMSDataset$new_from_saved_dir(
  input_dir,
  scratch_dir = dirname(input_dir)
)
```

Arguments:

- `input_dir`: The path to the directory where the `dataset.rds` is saved and all the corresponding `sample_*.rds` files are. Typically a subdirectory of `scratch_dir`.
- `scratch_dir`: The new scratch directory where further processing samples will be saved. By default it is the parent of `input_dir`.

Public fields

`pData` A data frame with at least the SampleID and filename columns.

`align` To store alignment results

`peaks` To store the peak list

`TIS` A matrix of n_samples vs drift time, with the Total Ion Spectrum of each sample

`RIC` A matrix of n_samples vs retention time, with the Reverse Ion Chromatogram of each sample

`dt_ref` A numeric drift time of reference

`rt_ref` A numeric retention time of reference

`userData` A list to store arbitrary data in the dataset

Active bindings

`sampleNames` The sample names of the GCIMSDataset samples

Methods

Public methods:

- `GCIMSDataset$new()`
- `GCIMSDataset$print()`
- `GCIMSDataset$subset()`
- `GCIMSDataset$.impl__subset__()`
- `GCIMSDataset$appendDelayedOp()`
- `GCIMSDataset$hasDelayedOps()`
- `GCIMSDataset$realize()`
- `GCIMSDataset$getSample()`
- `GCIMSDataset$extract_dtime_rtime()`
- `GCIMSDataset$getRIC()`
- `GCIMSDataset$extract_RIC_and_TIS()`
- `GCIMSDataset$is_on_disk()`

- GCIMSDataset$copy()
- GCIMSDataset$updateScratchDir()
- GCIMSDataset$getCurrentDir()
- GCIMSDataset$clone()

Method `new()`**:** Create a new GCIMSDataset object

Usage:

```
GCIMSDataset$new(
  pData = NULL,
  base_dir = NULL,
  ...,
  samples = NULL,
  parser = "default",
  scratch_dir = NULL,
  keep_intermediate = FALSE,
  on_ram = FALSE
)
```

Arguments:

`pData` A data frame holding phenotype data for the samples (or `NULL`). The data frame should at least have a `SampleID` column, and a `filename` column if samples are stored in files.

`base_dir` The base directory. Sample `i` is found on `file.path(base_dir, pData$filename[i])`.

`...` Unused

`samples` A named list of `GCIMSSample` objects to be included in the dataset (or `NULL`). Names should correspond to the `SampleID` column in the `pData` data frame.

`parser` Function that takes a file path and returns a GCIMSSample object. Use `"default"` to use the default parser in the GCIMS package, that supports `.mea` files (from GAS). Check out `vignette("importing-custom-data-formats", package = "GCIMS")` for more information

`scratch_dir` A directory where intermediate and processed samples will be stored

`keep_intermediate` If `TRUE`, intermediate results will not be deleted (ignored if `on_ram` is `TRUE`).

`on_ram` If `TRUE`, samples are not stored on disk, but rather kept on RAM. Set it to `TRUE` only with small datasets.

Examples:

```
dummy_dataset <- GCIMSDataset$new(
  pData = data.frame(SampleID = character(), filename = character(0)),
  base_dir = tempdir()
)
```

Method `print()`**:** prints the dataset to the screen

Usage:

```
GCIMSDataset$print()
```

Method `subset()`**:** Create a new dataset containing a subset of the samples

Usage:

```
GCIMSDataset$subset(samples, inplace = FALSE, new_scratch_dir = NA)
```

Arguments:

`samples` A numeric vector (sample indices), a character vector (sample names) or a logical vector of the length equal to the number of samples in the dataset (`TRUE` elements will be subset)

`inplace` if `TRUE` subset happens in-place, otherwise subset will return a copy.

`new_scratch_dir` A new scratch directory, only used if `inplace=FALSE` and the dataset is on-disk.

Returns: A GCIMSDataset (new or the current one depending on `inplace`), with the requested sample subset

Method `.impl__subset__()`**:** Do not call this method. It does an inplace subset. Use `obj$subset(samples, inplace = TRUE)` instead

Usage:

```
GCIMSDataset$.impl__subset__(samples)
```

Arguments:

`samples` A numeric vector (sample indices), a character vector (sample names) or a logical vector of the length equal to the number of samples in the dataset (`TRUE` elements will be subset)

Returns: The given GCIMSDataset object, with a subset of the samples

Method `appendDelayedOp()`**:** Appends a delayed operation to the dataset so it will run afterwards

Usage:

```
GCIMSDataset$appendDelayedOp(operation)
```

Arguments:

`operation` A DelayedOperation object

Returns: The modified GCIMSDataset object

Method `hasDelayedOps()`**:** Find out if the dataset has pending operations

Usage:

```
GCIMSDataset$hasDelayedOps()
```

Returns: Returns `TRUE` if the dataset has pending operations, `FALSE` otherwise

Method `realize()`**:** Execute all pending operations on the dataset

Usage:

```
GCIMSDataset$realize(keep_intermediate = NA)
```

Arguments:

`keep_intermediate` logical or `NA`. Only when the analysis is on disk, keep intermediate result files. If `NA`, the `keep_intermediate` option given at the dataset initialization takes precedence.

Returns: The dataset object, invisibly

Method `getSample()`**:** Get a sample from a GCIMSDataset

Usage:

```
GCIMSDataset$getSample(sample)
```

Arguments:

`sample` Either an integer (sample index) or a string (sample name)

Returns: The GCIMSSample object

Method `extract_dtime_rtime()`**:** Sets an action to extract the reference retention and drift times

Usage:

```
GCIMSDataset$extract_dtime_rtime()
```

Method `getRIC()`**:** Get the Reverse Ion Chromatogram

Usage:

```
GCIMSDataset$getRIC()
```

Returns: A matrix with the reverse ion chromatograms for all samples

Method `extract_RIC_and_TIS()`**:** Extracts the RIC and the TIS

Usage:

```
GCIMSDataset$extract_RIC_and_TIS()
```

Returns: The GCIMSDataset

Method `is_on_disk()`**:** Whether the dataset is saved on disk or stored in RAM

Usage:

```
GCIMSDataset$is_on_disk()
```

Returns: `TRUE` if on disk, `FALSE` otherwise

Method `copy()`**:** Creates a copy of the dataset. If the dataset is stored on disk, then a new `scratch_dir` must be used.

Usage:

```
GCIMSDataset$copy(scratch_dir = NA)
```

Arguments:

`scratch_dir` The scratch directory where samples being processed will be stored, if the copy is on disk.

Returns: A new GCIMSDataset object

Method `updateScratchDir()`**:** For on-disk datasets, copy all samples to a new scratch dir. This is useful when creating copies of the dataset, using the `dataset$copy()` method.

Usage:

```
GCIMSDataset$updateScratchDir(scratch_dir, override_current_dir = NULL)
```

Arguments:

`scratch_dir` The new `scratch_dir`, must be different from the current one

`override_current_dir` Typically used only internally, overrides the location of the samples. Useful when we are loading a dataset from a directory and the directory was moved since it was saved.

Method `getCurrentDir()`**:** Get the directory where processed samples are being saved, on on-disk datasets.

Usage:

```
GCIMSDataset$getCurrentDir()
```

Returns: Either a path or `NULL`. `NULL` is returned if samples have not been saved (either because have not been loaded or because the dataset is stored on RAM)

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
GCIMSDataset$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
## ------------------------------------------------
## Method `GCIMSDataset$new`
## ------------------------------------------------

dummy_dataset <- GCIMSDataset$new(
  pData = data.frame(SampleID = character(), filename = character(0)),
  base_dir = tempdir()
)
```

`GCIMSDataset_fromList` *GCIMSDataset_fromList*

Description

GCIMSDataset_fromList

Usage

```
GCIMSDataset_fromList(
  samples,
  pData = NULL,
  scratch_dir = NULL,
  keep_intermediate = FALSE,
  on_ram = TRUE
)
```

Arguments

samples	A named list of GCIMSSample objects. names should match pData$SampleID
pData	A data frame with at least the SampleID and filename columns.
scratch_dir	A directory to save intermediate results.
keep_intermediate	Whether to keep sample files for intermediate results. Only used if on_ram=FALSE
on_ram	logical. Whether the dataset should be kept stored on RAM or on disk.

Value

A GCIMSDataset object

Examples

```
# Create a new GCIMSDataset with the convenient constructor function:
sample1 <- GCIMSSample(
  drift_time = 1:2,
  retention_time = 1:3,
  data = matrix(1:6, nrow = 2, ncol = 3)
)
dummy_obj <- GCIMSDataset_fromList(
  pData = data.frame(SampleID = "Sample1", Sex = "female"),
  samples = list(Sample1 = sample1)
)
```

GCIMSSample *Create a GCIMSSample object*

Description

Create a GCIMSSample object

Usage

```
GCIMSSample(drift_time, retention_time, data, ...)
```

Arguments

drift_time, retention_time, data, ...	See the Slots section in GCIMSSample page

Value

A GCIMSSample object

Examples

```
# Create a new GCIMSSample with the convenient constructor function:
dummy_obj <-GCIMSSample(
  drift_time = 1:2,
  retention_time = 1:3,
  data = matrix(1:6, nrow = 2, ncol = 3),
  gc_column = "Optional column name",
  drift_gas = "nitrogen",
  drift_tube_length = 98.0 # in mm
)
```

GCIMSSample-class *GCIMSSample class*

Description

GCIMS Sample is an S4 class to store one sample with the drift and retention time ranges and other relevant attributes (GC column, drift tube length...) if available

The actual spectra is stored in the `data` slot, in a matrix, where the first index (rows) corresponds to drift times and the second to retention times (columns).

Slots

`drift_time` numeric. (required)

`retention_time` numeric. (required)

`data` matrix A matrix with drift time in the rows and retention time in columns. (required)

`gc_column` character. (optional) The type of chromatographic column used

`drift_tube_length` numeric (optional) The length of the drift tube, in mm

`drift_gas` character. (optional) The drift gas used (e.g "nitrogen")

`params` list (optional) Arbitrary list of parameters and annotations

`history` character. A character vector with a summary of information of the processing details the sample has gone through already

`filepath` character. A string with the path to the raw data

`description` A string (optional). A sample name or ID or description used in plots

`proc_params` list (internal). Data processing parameters computed and used internally.

`peaks` A data frame (internal). The peak list, typically set using `findPeaks()`. Use `peaks()` to get/set this.

`peaks_debug_info` A list with arbitrary debug information from `findPeaks()`.

`baseline` A matrix of the same dimensions as `data` with the baseline. Use `estimateBaseline()` to estimate it and `baseline()` to get or set it.

`class_version` "numeric_version" (internal) The GCIMSSample object defines internally a class version, so if a GCIMSSample object is saved, the GCIMS package is updated and the GCIMSSample class has changed during the upgrade it may be possible to upgrade the previously saved object when it's loaded.

See Also

GCIMSSample-methods

Examples

```
# Create a new GCIMSSample with methods::new()
dummy_obj <-methods::new(
  "GCIMSSample",
  drift_time = 1:2,
  retention_time = 1:3,
  data = matrix(1:6, nrow = 2, ncol = 3),
  gc_column = "Optional column name",
  drift_gas = "nitrogen",
  drift_tube_length = 98.0 # in mm
)
```

GCIMSSample-methods *Methods for the GCIMSSample class*

Description

Methods for the GCIMSSample class

Usage

```
## S3 method for class 'GCIMSSample'
x[i, j, ...]

## S3 method for class 'GCIMSSample'
dim(x)

## S3 method for class 'GCIMSSample'
subset(x, dt_idx = NULL, rt_idx = NULL, dt_range = NULL, rt_range = NULL, ...)

## S4 method for signature 'GCIMSSample'
description(object)

## S4 replacement method for signature 'GCIMSSample,ANY'
description(object) <- value

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

## S4 replacement method for signature 'GCIMSSample'
peaks(object) <- value
```

Arguments

x	A GCIMSSample object
i	index for drift time to subset
j	index for retention time to subset
...	ignored
dt_idx	A numeric vector with the drift time indices to extract (or a logical vector of the length of drift time)
rt_idx	A numeric vector with the retention time indices to extract (or a logical vector of the length of retention time)
dt_range	The minimum and maximum drift times to extract (length 2 vector)
rt_range	The minimum and maximum retention times to extract (length 2 vector)
object	A GCIMSSample object
value	A data frame with the peak list

Value

`[`: object `x` with features `i` and cells `j`

An integer vector with the number of rows and columns of the matrix

`subset`: A subsetted `GCIMSSample` object

Functions

- `[`: Simple subsetter for GCIMSSample objects
- `dim(GCIMSSample)`: Dimension of the data matrix
- `subset(GCIMSSample)`: Subset a GCIMSSample object
- `description(GCIMSSample)`: Get the description
- `description(object = GCIMSSample) <- value`: Set the description
- `peaks(GCIMSSample)`: Get the peak list
- `peaks(GCIMSSample) <- value`: Set the peak list

See Also

`base::subset()`

Examples

```
# `[' examples

obj <- GCIMSSample(drift_time=1:2, retention_time=1:3, data = matrix(1:6, nrow=2, ncol=3))
dim(obj)
```

GCIMSSample-rtime-dtime-intensity

Drift time, Retention time, Intensity of GCIMSSamples

Description

Functions to extract the drift time, the retention time and the intensity.

Usage

```
## S4 method for signature 'GCIMSSample'
dtime(object)

## S4 method for signature 'GCIMSSample'
rtime(object)

## S4 method for signature 'GCIMSSample'
intensity(
  object,
  dt_range = NULL,
  rt_range = NULL,
  dt_idx = NULL,
  rt_idx = NULL
)

## S4 replacement method for signature 'GCIMSSample'
intensity(object) <- value
```

Arguments

`object`	A GCIMSSample object.
`dt_range`	The minimum and maximum drift times to extract (length 2 vector)
`rt_range`	The minimum and maximum retention times to extract (length 2 vector)
`dt_idx`	A numeric vector with the drift time indices to extract (or a logical vector of the length of drift time)
`rt_idx`	A numeric vector with the retention time indices to extract (or a logical vector of the length of retention time)
`value`	A matrix of dimensions `dim(object)`

Value

The drift time of the sample

The retention time of the sample

Functions

- `dtime(GCIMSSample)`: Get the drift time vector
- `rtime(GCIMSSample)`: Get the retention time vector
- `intensity(GCIMSSample)`: Get the intensity matrix
- `intensity(GCIMSSample) <- value`: Set the intensity matrix

Examples

```
mea_file <- system.file("extdata",  "sample_formats", "small.mea.gz", package = "GCIMS")
gcims_sample <- read_mea(mea_file)
my_matrix <- intensity(gcims_sample, dt_range = c(7, 8), rt_range = c(1,30))
mea_file <- system.file("extdata",  "sample_formats", "small.mea.gz", package = "GCIMS")
gcims_sample <- read_mea(mea_file)
my_matrix <- intensity(gcims_sample)
intensity(gcims_sample) <- my_matrix/100
```

GCIMSSpectrum-class *GCIMSSpectrum class*

Description

GCIMSSpectrum is an S4 class to store a GCIMS Spectrum. It can be a single spectrum or the aggregation of several spectra.

Usage

```
GCIMSSpectrum(...)

## S4 method for signature 'GCIMSSpectrum'
description(object)

## S4 replacement method for signature 'GCIMSSpectrum,ANY'
description(object) <- value

## S4 method for signature 'GCIMSSpectrum'
dtime(object)

## S4 method for signature 'GCIMSSpectrum'
rtime(object)

## S4 method for signature 'GCIMSSpectrum'
intensity(object, dt_range = NULL, dt_idx = NULL)

## S4 method for signature 'GCIMSSpectrum'
peaks(object)
```

```
## S4 replacement method for signature 'GCIMSSpectrum'
peaks(object) <- value

## S4 method for signature 'GCIMSSpectrum,ANY'
plot(x, y, ...)
```

Arguments

...	See the slots section
object	A GCIMSSpectrum object
value	A data frame with the peak list
dt_range	The minimum and maximum drift times to extract (length 2 vector)
dt_idx	A numeric vector with the drift time indices to extract (or a logical vector of the length of drift time)
x	A GCIMSSpectrum object to plot
y	unused

Value

A GCIMSSpectrum object

A data frame with the peaks in the spectrum

The GCIMSSpectrum object

Functions

- `GCIMSSpectrum()`: Friendly constructor
- `description(GCIMSSpectrum)`: Get the description
- `description(object = GCIMSSpectrum) <- value`: Set the description
- `dtime(GCIMSSpectrum)`: Get the drift time vector
- `rtime(GCIMSSpectrum)`: Get the retention time where this spectrum was extracted
- `intensity(GCIMSSpectrum)`: Get the intensity matrix
- `peaks(GCIMSSpectrum)`: Get the peak list
- `peaks(GCIMSSpectrum) <- value`: Set the peak list
- `plot(x = GCIMSSpectrum, y = ANY)`: plot method

Slots

`drift_time` A numeric vector with drift times

`intensity` A numeric vector with the corresponding intensities

`baseline` A numeric vector of the same length as `intensity` with the corresponding baseline, or `NULL` if not set. Use `estimateBaseline()` to estimate it, `baseline()` to directly access it.

`retention_time_idx` The index or indices used to get the intensity

`retention_time_s` The retention times corresponding to the retention time indices.

description A string with a description (used as plot title, useful e.g. to know the sample it came from)

peaks A data frame with peaks, use `findPeaks()` to fill it, or `peaks()` to set/get it

peaks_debug_info A list with arbitrary debug information from `findPeaks()`

Examples

```
spec <- GCIMSSpectrum(drift_time = 1:10, intensity = c(1:5, 6:2))
```

`getChromatogram` *Get the extracted ion chromatogram*

Description

Get the extracted ion chromatogram

Usage

```
getChromatogram(
  object,
  dt_range = NULL,
  rt_range = NULL,
  dt_idx = NULL,
  rt_idx = NULL,
  aggregate = colSums
)
```

Arguments

object	A GCIMSSample object
dt_range	The minimum and maximum drift times to extract (length 2 vector)
rt_range	The minimum and maximum retention times to extract (length 2 vector)
dt_idx	A numeric vector with the drift time indices to extract (or a logical vector of the length of drift time)
rt_idx	A numeric vector with the retention time indices to extract (or a logical vector of the length of retention time)
aggregate	Function that takes the subsetted intensity matrix according to the region of interest and aggregates the drift times, returning a vector representing the chromatogram intensity. `colSums` by default.

Value

A GCIMSChromatogram object

Examples

```
x <- GCIMSSample(
  drift_time = 1:2,
  retention_time = 1:3,
  data = matrix(1:6, nrow = 2, ncol = 3)
)
getChromatogram(x)
# Take the maximum intensity in the region for each retention time:
sp1 <- getChromatogram(x, aggregate = function(x) apply(x, 2, max))
```

`getRIC,GCIMSDataset-method`

Get Reverse Ion Chromatogram

Description

Get Reverse Ion Chromatogram

Usage

```
## S4 method for signature 'GCIMSDataset'
getRIC(object)
```

Arguments

`object`	A GCIMSDataset object

Value

The RIC matrix

`getRIC,GCIMSSample-method`

Get the reverse ion chromatogram

Description

Get the reverse ion chromatogram

Usage

```
## S4 method for signature 'GCIMSSample'
getRIC(object)
```

Arguments

`object`	A GCIMSSample object

Value

A numeric vector with the reverse ion chromatogram

Examples

```
sample_file <- system.file("extdata", "sample_formats", "small.mea.gz", package = "GCIMS")
s <- read_mea(sample_file)
ric <- getRIC(s)
```

`getSpectrum` *Get IMS spectrum from a sample*

Description

Get IMS spectrum from a sample

Usage

```
getSpectrum(
  object,
  dt_range = NULL,
  rt_range = NULL,
  dt_idx = NULL,
  rt_idx = NULL,
  aggregate = rowSums
)
```

Arguments

`object`	A GCIMSSample object
`dt_range`	The minimum and maximum drift times to extract (length 2 vector)
`rt_range`	The minimum and maximum retention times to extract (length 2 vector)
`dt_idx`	A numeric vector with the drift time indices to extract (or a logical vector of the length of drift time)
`rt_idx`	A numeric vector with the retention time indices to extract (or a logical vector of the length of retention time)
`aggregate`	Function that takes the subsetted intensity matrix according to the region of interest and aggregates the retention times, returning a vector representing the spectrum intensity. `rowSums` by default.

Value

A GCIMSSpectrum object

Examples

```
x <- GCIMSSample(
  drift_time = 1:2,
  retention_time = 1:3,
  data = matrix(1:6, nrow = 2, ncol = 3)
)
getSpectrum(x, rt_idx = 2)

# Take the maximum intensity in the region for each drift time:
sp1 <- getSpectrum(x, aggregate = function(x) apply(x, 1, max))
```

`getTIS,GCIMSDataset-method`

Get Total Ion Spectra matrix

Description

Get Total Ion Spectra matrix

Usage

```
## S4 method for signature 'GCIMSDataset'
getTIS(object)
```

Arguments

`object`	A GCIMSDataset object

Value

A matrix with samples in rows and the drift time in columns

`getTIS,GCIMSSample-method`

Get the total ion spectrum

Description

Get the total ion spectrum

Usage

```
## S4 method for signature 'GCIMSSample'
getTIS(object)
```

Arguments

object	A GCIMSSample object

Value

A numeric vector with the total ion spectrum

Examples

```
sample_file <- system.file("extdata", "sample_formats", "small.mea.gz", package = "GCIMS")
s <- read_mea(sample_file)
tis <- getTIS(s)
```

---

imputePeakTable *Impute a Peak table*

Description

Impute a Peak table

Usage

```
imputePeakTable(peak_table, dataset, cluster_stats)
```

Arguments

peak_table	A matrix, with samples in rows and clusters in columns. It must have row names and column names.
dataset	The dataset object to extract samples from
cluster_stats	A data frame with the [dt|rt]_[min|max]_[ms|s] columns

Value

The imputed peak_table

Examples

```
# We are going to create a peak table matrix, typically resulting from [peakTable()]
# The peak table may have some missing values
# Since the missing values correspond to peaks that have not been detected
# in those particular samples, we can integrate the region where they should appear
# to get a value different than zero that reflects the noise level.
#
# Ingredients:
# - The GCIMSSample objects, so we can integrate the regions of interest (given as a GCIMSDataset)
# - The peak table matrix we want to impute
# - The definition of the regions corresponding to each cluster (cluster_stats)
#
```

```
# We will prepare here a synthetic example, check the vignette for a real use case
#
# Imagine we have information on the location of Cluster1 and Cluster2
cluster_stats <- data.frame(
  cluster = c("Cluster1", "Cluster2"),
  dt_min_ms = c(8, 10),
  dt_max_ms = c(9, 12),
  rt_min_s = c(120, 300),
  rt_max_s = c(128, 320)
)

# We have a peak table for two samples and two peaks
peak_table <- matrix(NA_real_, nrow = 2, ncol = 2)
rownames(peak_table) <- c("Sample1", "Sample2")
colnames(peak_table) <- c("Cluster1", "Cluster2")

# where we previously integrated Cluster2 in Sample 1 and Cluster1 in Sample2:
peak_table["Sample1", "Cluster2"] <- 9.5
peak_table["Sample2", "Cluster1"] <- 3.6
# The table has missing values, because some peaks were not detected.
# Maybe they are close to the noise level, or maybe they do not exist
peak_table

# We will fill the missing values by integrating whatever we find
# (typically noise or small peaks) in the cluster regions of each sample. So we
# need the sample matrices.
#
# Let's build dummy Sample1 and Sample2:
## Create drift time and retention time vectors:
dt <- seq(from = 0, to = 13, by = 0.1)  # ms
rt <- seq(from = 0, to = 350, by = 1)   # s

## Create matrices with random gaussian noise.
set.seed(42)
s1_intensity <- matrix(
  rnorm(length(dt)*length(rt), sd = 0.1),
  nrow = length(dt),
  ncol = length(rt)
)
s2_intensity <- matrix(
  rnorm(length(dt)*length(rt), sd = 0.1),
  nrow = length(dt),
  ncol = length(rt)
)

# The matrix will have a pleateau in a region where the peak is supposed
# to be, so when we impute the region corresponding to Sample1-Cluster1 we see a
# higher value:
s1_intensity[dt > 8.25 & dt < 8.75, rt > 122 & rt < 126] <- 1

## Create GCIMSSample objects
s1 <- GCIMSSample(
  drift_time = dt,
```

```
  retention_time = rt,
  data = s1_intensity
)
s2 <- GCIMSSample(
  drift_time = dt,
  retention_time = rt,
  data = s2_intensity
)
## And a dataset with the samples:
dataset <- GCIMSDataset_fromList(list(Sample1 = s1, Sample2 = s2))

# Now we can impute the table
peak_table_imp <- imputePeakTable(
  peak_table = peak_table,
  dataset = dataset,
  cluster_stats = cluster_stats
)
peak_table_imp
```

`integratePeaks,GCIMSDataset-method`

Integrate peaks in a GCIMSDataset

Description

Integrate peaks in a GCIMSDataset

Usage

```
## S4 method for signature 'GCIMSDataset'
integratePeaks(
  object,
  peak_list,
  integration_size_method = c("fixed_size", "free_size"),
  rip_saturation_threshold = 0.1
)
```

Arguments

`object`	The GCIMSDataset object, modified inline
`peak_list`	A data frame with peak lists
`integration_size_method`	Either fixed_size or free_size
`rip_saturation_threshold`	The threshold

Value

A modified GCIMSDataset object

`integratePeaks,GCIMSSample-method`
Peak integration for a GCIMSSample

Description

Peak integration for a GCIMSSample

Usage

```
## S4 method for signature 'GCIMSSample'
integratePeaks(
  object,
  peak_list,
  integration_size_method = c("fixed_size", "free_size"),
  rip_saturation_threshold = 0.1,
  verbose = FALSE
)
```

Arguments

`object`	A GCIMSSample object
`peak_list`	A data frame with the peak list
`integration_size_method`	If "fixed_size", the ROI integration limits are the same for all the peaks that belong to the same cluster. If "free_size", each ROI has its own integration limits, regardless of the cluster it is assigned to.
`rip_saturation_threshold`	Used to compute the "Saturation" column. If the ratio of the RIP intensity at the ROI apex with respect to the maximum RIP is below this threshold, the RIP is considered almost depleted, and it's more likely that the ROI suffers from non-linearities.
`verbose`	If `TRUE`, debug information will be printed

Value

The modified GCIMSSample, with an updated peak list

`intensity,GCIMSChromatogram-method` *Get the intensity vector*

Description

Get the intensity vector

Usage

```
## S4 method for signature 'GCIMSChromatogram'
intensity(object, rt_range = NULL, rt_idx = NULL)
```

Arguments

`object`	A GCIMSChromatogram object
`rt_range`	The minimum and maximum retention times to extract (length 2 vector)
`rt_idx`	A numeric vector with the retention time indices to extract (or a logical vector of the length of retention time)

Value

The retention intensity vector

See Also

Other GCIMSChromatogram: `GCIMSChromatogram`, `GCIMSChromatogram-class`, `dtime,GCIMSChromatogram-method`, `estimateBaseline,GCIMSChromatogram-method`, `findPeaks,GCIMSChromatogram-method`, `rtime,GCIMSChromatogram`
`smooth,GCIMSChromatogram-method`

`omit_times` *Omit ROIs present in certain retention and drift times*

Description

Extract the volume of each ROI across samples to create a peak table.

Usage

```
omit_times(peak_list, rt_time_2_omit = NULL, dt_time_2_omit = NULL)
```

Arguments

peak_list	The output of peaks(). Also, you can create your own peak table and use it as input value for peak_list
rt_time_2_omit	A vector including a set of retention times where ROIs detected should not be considered. As default is is set as NULL
dt_time_2_omit	A vector including a set of drift times where ROIs detected should not be considered. As default is is set as NULL

Value

A peak_list without the ROIs present in the retention and drift times not desired.

Examples

```
peak_list <- data.frame(
  rt_apex_s =  c(1, 2, 3, 3, 4, 4,  5, 5, 6, 6),
  dt_apex_ms = c(2, 4, 6, 4, 8, 4, 10, 4, 4, 12)
)
peak_list_filt <- omit_times(peak_list, dt_time_2_omit = 4)
```

---

overlay_peaklist *Overlay a peak list to a plot*

Description

Overlay a peak list to a plot

Usage

```
overlay_peaklist(
  peaklist = NULL,
  ...,
  dt_range = NULL,
  rt_range = NULL,
  pdata = NULL,
  color_by = NULL,
  mapping_roi = c(dt_min_ms = "dt_min_ms", dt_max_ms = "dt_max_ms", rt_min_s =
    "rt_min_s", rt_max_s = "rt_max_s"),
  palette = P40
)
```

Arguments

peaklist	A data frame with at least the columns: dt_min_ms, dt_max_ms, rt_min_s, rt_max_s and optionally additional columns (e.g. the column given to color_by)
...	Ignored.
dt_range	The minimum and maximum drift times to extract (length 2 vector)
rt_range	The minimum and maximum retention times to extract (length 2 vector)
pdata	A phenotype data data frame, with a SampleID column. This column will be used to merge pdata with peaklist, so color_by can specify a phenotype.
color_by	A character with a column name of peaklist or pdata. Used to color the border of the added rectangles and apices. A string with a color name is also acceptable.
mapping_roi	A 4-elements named character vector with the names of the columns from peaklist that will be used as the rectangle coordinates.
palette	A character vector with color names to use drawing the rectangles. Use NULL to let ggplot2 set the defaults.

Details

If peaklist includes dt_apex_ms and rt_apex_s a cross will be plotted on the peak apex.

Value

A list with the ggplot layers to overlay, including geom_rect and possibly geom_point and scale_fill_manual.

Examples

```
dt <- 1:10
rt <- 1:10
int <- matrix(0.0, nrow = length(dt), ncol = length(rt))

int[2, 4:8] <- c(.5, .5, 1, .5, 0.5)
int[3, 4:8] <- c(0.5, 2, 2, 2, 0.5)
int[4, 4:8] <- c(1, 2, 5, 2, 1)
int[5, 4:8] <- c(0.5, 2, 2, 2, 0.5)
int[6, 4:8] <- c(.5, .5, 1, .5, 0.5)

dummy_obj <-GCIMSSample(
  drift_time = dt,
  retention_time = rt,
  data = int
)
plt <- plot(dummy_obj)

# Add a rectangle on top of the plot
rect <- data.frame(
  dt_min_ms = 2.75,
  dt_max_ms = 5.6,
  rt_min_s = 4.6,
  rt_max_s = 7.4
```

```
)

plt + overlay_peaklist(
  peaklist = rect
)
```

---

pData,GCIMSDataset-method

Get/Set the phenotype data

Description

Get/Set the phenotype data

Usage

```
## S4 method for signature 'GCIMSDataset'
pData(object)

## S4 replacement method for signature 'GCIMSDataset,ANY'
pData(object) <- value
```

Arguments

object	A GCIMSDataset object
value	The data frame with annotations, it should have a FileName column and a SampleID column.

Value

A data frame with the phenotype data

Functions

- pData(object = GCIMSDataset) <- value: Set pData

peaks,GCIMSDataset-method *Get the peak list*

Description

Get the peak list

Usage

```
## S4 method for signature 'GCIMSDataset'
peaks(object)

## S4 replacement method for signature 'GCIMSDataset'
peaks(object) <- value
```

Arguments

object	A GCIMSDataset object
value	The data frame with a peak list

Value

A data frame with the detected peaks

Functions

- peaks(GCIMSDataset) <- value: Set the peak list

peakTable *Build a peak table*

Description

Extract the volume of each ROI across samples to create a peak table.

Usage

```
peakTable(peak_list_clustered, aggregate_conflicting_peaks = NULL)
```

Arguments

peak_list_clustered
: A peak list with clusters assigned. Also, you can create your own peak table and use it as input value for peak_list_clustered (see first example below)

aggregate_conflicting_peaks
: NULL or a function. What to do, in case two peaks from the same sample have been assigned to the same cluster. If NULL, throw an error. If mean, max or any other function, we will summarize all the conflicting volumes into that number (e.g. "take the maximum of the peaks")

Value

A list with three fields: peak_table, peak_table_matrix, and peak_table_duplicity. peak_table, and peak_table_matrix, provide information of the peak table. peak_table is a dataframe containing cluster volumes, whose columns represent samples and rows clusters. peak_table_matrix presents the same information content as peak_table but in matrix form. Note that in peak_table columns represent clusters and rows samples. Finally, peak_table_duplicity is a dataframe that shows ROI duplicity information among clusters. Ideally, only one peak per sample should belong to a cluster.

Examples

```
# Create your peak table from scratch:
pl <- data.frame(
  SampleID = c("S1", "S1", "S2", "S2"),
  cluster = c("Cluster1", "Cluster2", "Cluster1", "Cluster2"),
  Volume = c(10, 20, 8, 18)
)
peak_table <- peakTable(pl)

peak_table$peak_table_matrix

# You can use imputePeakTable() to fill in the missing values

# If the clustering doesn't work great, you may end up with two peaks
# from the same sample on the same cluster. This does not make sense
# empirically, because it's either one or the other. In case of such
# ambiguity, peakTable() will give an error.
#
# If you want, you can override the error by taking the average volume
# of those ambiguous peaks, or the maximum, using,
# e.g. `aggregate_conflicting_peaks = max`.
#
# In any case, you will get information on how many peaks were aggregated
# in the `peak_table_duplicity` field (ideally should be full of `1`):
peak_table$peak_table_duplicity
```

plot,GCIMSSample,ANY-method

Topographical plot of a GCIMSSample object

Description

Topographical plot of a GCIMSSample object

Usage

```
## S4 method for signature 'GCIMSSample,ANY'
plot(
  x,
  dt_range = NULL,
  rt_range = NULL,
  ...,
  remove_baseline = FALSE,
  trans = "cubic_root"
)
```

Arguments

x	A GCIMSSample object
dt_range	A numeric vector of length 2 with the drift time range to plot (in milliseconds)
rt_range	A numeric vector of length 2 with the retention time range to plot (in seconds)
...	Ignored
remove_baseline	Set to TRUE to subtract the estimated baseline first
trans	The transformation to the intensity values. "cubic_root" is the default. "intensity" is also valid. See the trans argument in ggplot2::continuous_scale() for other possibilities.

Value

A plot of the GCIMSSample

Examples

```
dummy_obj <-GCIMSSample(
  drift_time = 1:2,
  retention_time = 1:3,
  data = matrix(1:6, nrow = 2, ncol = 3),
  gc_column = "Optional column name",
  drift_gas = "nitrogen",
  drift_tube_length = 98.0 # in mm
)
plot(dummy_obj)
```

---

plotRIC,GCIMSDataset-method

Plot Reverse Ion Chromatograms

Description

Plot Reverse Ion Chromatograms

Usage

```
## S4 method for signature 'GCIMSDataset'
plotRIC(object, rt_range = NULL, sample = NULL)
```

Arguments

object	A GCIMSDataset object
rt_range	The minimum and maximum retention times to extract (length 2 vector)
sample	A number or a string with the sample index or name. If NULL, all samples are returned

Value

A plot

---

plotTIS,GCIMSDataset-method

Plot Total Ion Spectra

Description

Plot Total Ion Spectra

Usage

```
## S4 method for signature 'GCIMSDataset'
plotTIS(object, dt_range = NULL, sample = NULL)
```

Arguments

object	A GCIMSDataset object
dt_range	The minimum and maximum drift times to extract (length 2 vector)
sample	A number or a string with the sample index or name. If NULL, all samples are returned

Value

The plot of the TIS

plot_interactive *Make a plot interactive*

Description

Wraps the `plt` with `plotly::ggplotly()` and sets the `xaxis` and `yaxis` ticks to `"auto"`, so the axis labels are updated when zooming.

Usage

```
plot_interactive(plt)
```

Arguments

`plt`	A ggplot plot

Value

A plotly plot

Examples

```
d <- data.frame(x = c(1,2), y=c(1,2))
plt <- ggplot2::ggplot(d) +
  ggplot2::geom_point(ggplot2::aes(x = x, y = y))
plot_interactive(plt)
```

prealign,GCIMSSample-method *Align a GCIMSSample object, in drift time and to the injection point in retention time*

Description

Align a GCIMSSample object, in drift time and to the injection point in retention time

Usage

```
## S4 method for signature 'GCIMSSample'
prealign(object, align_dt, align_ip, rip_ref_ms, min_start, rt_ref)
```

Arguments

object	A GCIMSSample object
align_dt	if TRUE, align the drift time axis using a multiplicative correction
align_ip	if TRUE a multiplicative correction will be done in retention time before applying the other algorithm
rip_ref_ms	The reference position of the Reactant Ion Peak in the dataset (in ms)
min_start	minimun injection point, to calculate where to begin the spectrums and cut as few points as posible, to be used in injection point alignment
rt_ref	retention time reference for alignment to injection point

Value

The modified GCIMSSample

---

read_mea *Read .mea files (from GAS Dortmund)*

Description

This function reads a .mea file (supporting gzip compressed .mea.gz files as well) and returns a GCIMS object

Usage

```
read_mea(filename)
```

Arguments

filename	A .mea or a .mea.gz path to a file

Details

Thanks to Daniel Sanders and Thomas Wortelmann from GAS Dortmund for providing the necessary support to implement this function.

Value

The GC-IMS sample in a GCIMSSample object

Examples

```
mea_file <- system.file("extdata/sample_formats/small.mea.gz", package = "GCIMS")
sample <- read_mea(mea_file)
```

realize *Runs all delayed operations on the object*

Description

Runs all delayed operations on the object

Usage

```
realize(object, keep_intermediate = NA)
```

Arguments

object	A GCIMSDataset object, modified in-place
keep_intermediate	A logical, whether to keep the intermediate files of the previous realization once this one finishes. If NA, keeping will depend on the object.

Value

The same GCIMSDataset object, without pending operations

Examples

```
base_dir <- system.file("extdata", "sample_formats", package = "GCIMS")
annot <- data.frame(SampleID = "Sample1", FileName = "small.mea.gz")
dataset <- GCIMSDataset$new(annot, base_dir)
print(dataset)
realize(dataset)
print(dataset)
```

rtime,GCIMSChromatogram-method *Get the retention time vector*

Description

Get the retention time vector

Usage

```
## S4 method for signature 'GCIMSChromatogram'
rtime(object)
```

Arguments

object	A GCIMSChromatogram

Value

The retention time vector (in s)

See Also

Other GCIMSChromatogram: GCIMSChromatogram, GCIMSChromatogram-class, dtime,GCIMSChromatogram-method, estimateBaseline,GCIMSChromatogram-method, findPeaks,GCIMSChromatogram-method, intensity,GCIMSChromato smooth,GCIMSChromatogram-method

---

rtime,GCIMSDataset-method

Get a reference retention time vector for the dataset

Description

Get a reference retention time vector for the dataset

Usage

```
## S4 method for signature 'GCIMSDataset'
rtime(object, sample = NULL)
```

Arguments

object	A GCIMSDataset
sample	A number or a string with the sample index or name. If NULL, the reference drift time is returned

Value

a retention time vector

sampleNames,GCIMSDataset-method *Sample names*

Description

Sample names

Usage

```
## S4 method for signature 'GCIMSDataset'
sampleNames(object)

## S4 replacement method for signature 'GCIMSDataset,ANY'
sampleNames(object) <- value
```

Arguments

object	GCIMSDataset object
value	A character vector of length the number of samples with the sample names

Value

The GCIMSDataset object

Functions

- `sampleNames(object = GCIMSDataset) <- value`: Sample names

show_progress_bar *Show progress bar*

Description

Show progress bar

Usage

```
show_progress_bar()
```

Value

logic to show the progress bar

---

smooth,GCIMSChromatogram-method

Smoothing a GCIMS chromatogram using a Savitzky-Golay filter

Description

Smoothing a GCIMS chromatogram using a Savitzky-Golay filter

Usage

```
## S4 method for signature 'GCIMSChromatogram'
smooth(x, rt_length_s = 3, rt_order = 2L)
```

Arguments

x	A GCIMSChromatogram object
rt_length_s	The length of the filter in retention time (in s)
rt_order	The order of the filter in retention time

Value

The modified GCIMSChromatogram

See Also

Other GCIMSChromatogram: GCIMSChromatogram, GCIMSChromatogram-class, dtime,GCIMSChromatogram-method, estimateBaseline,GCIMSChromatogram-method, findPeaks,GCIMSChromatogram-method, intensity,GCIMSChromato rtime,GCIMSChromatogram-method

---

smooth,GCIMSDataset-method

Smoothing a GCIMS dataset using a Savitzky-Golay filter

Description

Smoothing a GCIMS dataset using a Savitzky-Golay filter

Usage

```
## S4 method for signature 'GCIMSDataset'
smooth(x, dt_length_ms = 0.14, rt_length_s = 3, dt_order = 2, rt_order = 2)
```

Arguments

x	A GCIMSDataset object, modified in-place
dt_length_ms	the length of the filter in drift time (in ms)
rt_length_s	The length of the filter in retention time (in s)
dt_order	The order of the filter in drift time
rt_order	The order of the filter in retention time

Value

The modified GCIMSDataset

---

smooth,GCIMSSample-method

Smoothing a GCIMS sample using a Savitzky-Golay filter

Description

Smoothing a GCIMS sample using a Savitzky-Golay filter

Usage

```
## S4 method for signature 'GCIMSSample'
smooth(x, dt_length_ms, rt_length_s, dt_order = 2, rt_order = 2)
```

Arguments

x	A GCIMSSample object
dt_length_ms	the length of the filter in drift time (in ms)
rt_length_s	The length of the filter in retention time (in s)
dt_order	The order of the filter in drift time
rt_order	The order of the filter in retention time

Value

The modified GCIMSSample

`smooth,GCIMSSpectrum-method`

Smoothing a GCIMS Spectrum using a Savitzky-Golay filter

Description

Smoothing a GCIMS Spectrum using a Savitzky-Golay filter

Usage

```
## S4 method for signature 'GCIMSSpectrum'
smooth(x, dt_length_ms, dt_order = 2)
```

Arguments

`x`	A GCIMSSpectrum object
`dt_length_ms`	the length of the filter in drift time (in ms)
`dt_order`	The order of the filter in drift time

Value

The modified GCIMSSpectrum

`updateObject,GCIMSSample-method`

Updates old saved GCIMSSample object to the latest version

Description

This function is useful when you have saved a GCIMSSample object with a previous version of the GCIMS package and you want to load it using a new version of the package.

Usage

```
## S4 method for signature 'GCIMSSample'
updateObject(object, ..., verbose = FALSE)
```

Arguments

`object`	A GCIMSSample object, typically that has been serialized and loaded from disk
`...`	Unused
`verbose`	Unused

Details

The function allows you to update the old object, adding missing slots, etc so it is fully compatible with the new class definition.

Value

The updated GCIMSSample object

Examples

```
obj <- GCIMSSample(drift_time=1:2, retention_time=1:3, data = matrix(1:6, nrow=2, ncol=3))
# Update the object:
newobj <- updateObject(obj)
```

Index