Package: mpactr (via r-universe)

September 11, 2024

Title Correction of Preprocessed MS Data

Version 0.1.0

Description An 'R' implementation of the 'python' program Metabolomics Peak Analysis Computational Tool ('MPACT') (Robert M. Samples, Sara P. Puckett, and Marcy J. Balunas (2023) <doi:10.1021/acs.analchem.2c04632>). Filters in the package serve to address common errors in tandem mass spectrometry preprocessing, including: (1) isotopic patterns that are incorrectly split during preprocessing, (2) features present in solvent blanks due to carryover between samples, (3) features whose abundance is greater than user-defined abundance threshold in a specific group of samples, for example media blanks, (4) ions that are inconsistent between technical replicates, and (5) in-source fragment ions created during ionization before fragmentation in the tandem mass spectrometry workflow.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2.9000

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Depends data.table, R (>= 4.1.0)

Imports cli, ggplot2, R6, readr, stats, treemapify, viridis

LazyData true

Config/Needs/website rmarkdown, tidyverse, viridis, plotly, Hmisc, corrplot, ggdendro, ggtext

URL https://mums2.github.io/mpactr/, https://github.com/mums2/mpactr

BugReports https://github.com/mums2/mpactr/issues

Repository https://mums2.r-universe.dev

RemoteUrl https://github.com/mums2/mpactr

RemoteRef HEAD

RemoteSha a9e30123c568f58edd13d2e5997b81db7e89ad81

Contents

`cultures_data` *LC-MS/MS sample data*

Description

A `mpactr` R6 class object of contining a feature table and associated sample metadata.

Usage

```
cultures_data
```

Format

`culture_data`**:**

A `mpactr` with 2 attributes:

peak_table A feature table of class `data.table`

meta_data A `data.table` with associated sample metadata

Value

An `mpactr` R6 class object.

example *Get file paths for examples*

Description

mpactr contains a number of example files in the `inst/extdata` directory. This function makes them accessible in documentation that shows how file paths are used in function examples.

Usage

```
example(file = NULL)
```

Arguments

`file`	Name of a file. If `NULL`, all examples files will be listed.

Value

A file path to example data stored in the `inst/extdata` directory of the package.

Examples

```
example()

example("metadata.csv")
```

filter_cv *Filter Non-reproducible ions*

Description

`filter_cv()` removes feature ions that are found to be non-reproducible between technical injection replicates. Reproducibility is assessed via mean or median coefficient of variation (CV) between technical replicates. As such, this filter is expecting an input dataset with at least two replicate injections per sample.

`copy_object`: mpactr is built on an R6 class-system, meaning it operates on reference semantics in which data is updated *in-place*. Compared to a shallow copy, where only data pointers are copied, or a deep copy, where the entire data object is copied in memory, any changes to the original data object, regardless if they are assigned to a new object, result in changes to the original data object. We recommend using the default `copy_object = FALSE` as this makes for an extremely fast and memory-efficient way to chain mpactr filters together; however, if you would like to run the filters individually with traditional R style objects, you can set `copy_object` to `TRUE` as shown in the filter examples.

Usage

```
filter_cv(mpactr_object, cv_threshold = NULL, cv_param, copy_object = FALSE)
```

Arguments

mpactr_object	An mpactr_object. See import_data().
cv_threshold	Coefficient of variation threshold. A lower cv_threshold will result in more stringent filtering and higher reproducibility. Recommended values between 0.2 - 0.5.
cv_param	Coefficient of variation (CV) statistic to use for filtering Options are "mean" or "median", corresponding to mean and median CV, respectively.
copy_object	A boolean parameter that allows users to return a copied object instead of modifying the object.

Value

an mpactr_object.

Examples

```
data <- import_data(example("coculture_peak_table.csv"),
  example("metadata.csv"),
  format = "Progenesis"
)

data_filter <- filter_cv(data,
  cv_threshold = 0.01,
  cv_param = "mean",
  copy_object = TRUE
)

data_filter <- filter_cv(data,
  cv_threshold = 0.01,
  cv_param = "median",
  copy_object = TRUE
)
```

---

filter_group *Filter Ions by Group*

Description

Filter Ions by Group

Usage

```
filter_group(
  mpactr_object,
  group_threshold = 0.01,
  group_to_remove,
  remove_ions = TRUE,
  copy_object = FALSE
)
```

Arguments

mpactr_object	An mpactr_object. See import_data().
group_threshold	Relative abundance threshold at which to remove ions. Default = 0.01.
group_to_remove	Biological group name to remove ions from.
remove_ions	A boolean parameter. If TRUE failing ions will be removed from the peak table. Default = TRUE.
copy_object	A boolean parameter that allows users to return a copied object instead of modifying the object.

Details

filter_group() removes feature ions that are present in a user-defined group based on a relative abundance threshold. This could be particularly useful to filter out features found present in solvent blank samples. Further, this filter can be ultilized to remove features in media blank sample for experiments on microbial cultures. The presence or absence of features in a group of samples is determined by first averaging injection replicates and then averaging biological replicates within each biological treatment group. A feature is present in a group if its abundance is greater than the user-defined group_threshold. The default is 0.01, meaning a feature is removed if its abundance is 1% of that in the sample group in which it is most abundant. For example, blank filtering can remove features whose mean abundance in solvent blank injections is greater than 1% of their maximum mean abundance in experimental samples.

If you would like to remove features found in media blank samples, we recommend testing the group_threshold parameter.

copy_object: mpactr is built on an R6 class-system, meaning it operates on reference semantics in which data is updated *in-place*. Compared to a shallow copy, where only data pointers are copied, or a deep copy, where the entire data object is copied in memory, any changes to the original data object, regardless if they are assigned to a new object, result in changes to the original data object. We recommend using the default copy_object = FALSE as this makes for an extremely fast and memory-efficient way to chain mpactr filters together; however, if you would like to run the filters individually with traditional R style objects, you can set copy_object to TRUE as shown in the filter examples.

Value

an mpactr_object.

Examples

```
data <- import_data(example("coculture_peak_table.csv"),
  example("metadata.csv"),
  format = "Progenesis"
)

data_filter <- filter_group(data,
  group_threshold = 0.01,
  group_to_remove = "Blanks",
  remove_ions = TRUE
)
```

---

filter_insource_ions *Filter Insource ions*

Description

`filter_insource_ions()` identifies and removes in-source ion clusters based on a Pearson correlation threshold. Groups of co-eluting features with identical retention time are identified and used to generate Pearson correlation matrices. Clusters with self-similarity greater than the user-defined `cluster_threshold` within these matrices are identified as likely belonging to a single precursor ion and is associated insource ion. Highly correlated ions are identified and removed.

`copy_object`: mpactr is built on an R6 class-system, meaning it operates on reference semantics in which data is updated *in-place*. Compared to a shallow copy, where only data pointers are copied, or a deep copy, where the entire data object is copied in memory, any changes to the original data object, regardless if they are assigned to a new object, result in changes to the original data object. We recommend using the default `copy_object = FALSE` as this makes for an extremely fast and memory-efficient way to chain mpactr filters together; however, if you would like to run the filters individually with traditional R style objects, you can set `copy_object` to `TRUE` as shown in the filter examples.

Usage

```
filter_insource_ions(
  mpactr_object,
  cluster_threshold = 0.95,
  copy_object = FALSE
)
```

Arguments

`mpactr_object`	An `mpactr_object`. See `import_data()`.
`cluster_threshold`	Cluster threshold for ion deconvolution. Default = 0.95.
`copy_object`	A `boolean` parameter that allows users to return a copied object instead of modifying the object.

Value

an mpactr_object

Examples

```
data <- import_data(example("coculture_peak_table.csv"),
  example("metadata.csv"),
  format = "Progenesis"
)

data_filter <- filter_insource_ions(data,
  cluster_threshold = 0.95
)
```

---

filter_mispicked_ions *Mispicked ions filter*

Description

filter_mispicked_ions() identifies ions that were incorrectly split into separate features during preprocessing. This filter checks the feature table for similar ions in terms of mass and retention time. Peaks found to be similar are merged into a single feature given merge_peaks is TRUE.

The parameter ringwin is the detector saturation mass window, specific for some instruments, such as Waters Synapse G2-Si-Q-ToF, to account for high concentration samples.

Parameter isowin is the isotopic mass window, which accounts for isotopic peaks of the same precussor mass that were incorrectly assigned during preprocessing.

copy_object: mpactr is built on an R6 class-system, meaning it operates on reference semantics in which data is updated *in-place*. Compared to a shallow copy, where only data pointers are copied, or a deep copy, where the entire data object is copied in memory, any changes to the original data object, regardless if they are assigned to a new object, result in changes to the original data object. We recommend using the default copy_object = FALSE as this makes for an extremely fast and memory-efficient way to chain mpactr filters together; however, if you would like to run the filters individually with traditional R style objects, you can set copy_object to TRUE as shown in the filter examples.

Usage

```
filter_mispicked_ions(
  mpactr_object,
  ringwin = 0.5,
  isowin = 0.01,
  trwin = 0.005,
  max_iso_shift = 3,
  merge_peaks = TRUE,
  merge_method = "sum",
  copy_object = FALSE
)
```

Arguments

mpactr_object	An mpactr_object. See import_data().
ringwin	Ringing mass window or detector saturation mass window. Default = 0.5 atomic mass units (AMU).
isowin	Isotopic mass window. Default = 0.01 AMU.
trwin	A numeric denoting the retention time threshold for assessing if ions should be merged. Default = 0.005.
max_iso_shift	A numeric. Default = 3.
merge_peaks	A boolean parameter to determine if peaks found to belong to the same ion should be merged in the feature table.
merge_method	If merge_peaks is TRUE, a method for how similar peaks should be merged. Can be one of "sum".
copy_object	A boolean parameter that allows users to return a copied object instead of modifying the object.

Value

an mpactr_object.

Examples

```
data <- import_data(example("coculture_peak_table.csv"),
  example("metadata.csv"),
  format = "Progenesis"
)

data_filter <- filter_mispicked_ions(data,
  ringwin = 0.5,
  isowin = 0.01,
  trwin = 0.005,
  max_iso_shift = 3,
  merge_peaks = TRUE,
  merge_method = "sum"
)
```

---

filter_summary *Return the summary for a single mpactr filter.*

Description

filter_summary() is a wrapper function to return the summary from a single filter within the given mpactr object.

Usage

```
filter_summary(mpactr_object, filter, group = NULL)
```

Arguments

mpactr_object	The mpactr object that is created by calling the import_data() function.
filter	The name of a filter whose summary is to be extracted. Must be one of: "mispicked", "group", "replicability", or "insource".
group	If filter = "group", the name of the Biological_Group used to filter.

Value

a list reporting 1) compound ids for compounds which failed the filter and 2) compound ids for compounds which passed the filter.

Examples

```
data <- import_data(example("coculture_peak_table.csv"),
  example("metadata.csv"),
  format = "Progenesis"
)

data_filter <- filter_mispicked_ions(data)

mispicked_summary <- filter_summary(data_filter, filter = "mispicked")
mispicked_summary
```

---

get_cv_data *Get CV values.*

Description

get_cv_data() is a wrapper function to return cv (coefficient of variation) calculated with filter_cv().

Usage

```
get_cv_data(mpactr_object)
```

Arguments

mpactr_object	The mpactr object that is created by calling the import_data() function.

Value

a data.table reporting the mean and median coefficient of variation for each input ion.

Examples

```
data <- import_data(example("coculture_peak_table.csv"),
  example("metadata.csv"),
  format = "Progenesis"
)

data_filter <- filter_cv(data,
  cv_threshold = 0.01,
  cv_param = "median"
)

cv <- get_cv_data(data_filter)
head(cv)
```

`get_group_averages` *Get groups averages.*

Description

`get_group_averages()` is a wrapper function to return group averages for the filtered peak table.

Usage

```
get_group_averages(mpactr_object)
```

Arguments

`mpactr_object`	The mpactr object that is created by calling the import_data() function.

Value

a `data.table` reporting the average and relative standard deviation across biological groups and technical replicates within each group.

Examples

```
data <- import_data(example("coculture_peak_table.csv"),
  example("metadata.csv"),
  format = "Progenesis"
)

data_filter <- filter_group(data, group_to_remove = "Blanks")

group_averages <- get_group_averages(data_filter)
head(group_averages)
```

get_meta_data *Return the meta_data data.table from the mpactr object.*

Description

`get_meta_data()` a wrapper function to return the meta data object of the given mpactr object.

Usage

```
get_meta_data(mpactr_object)
```

Arguments

`mpactr_object`	The mpactr object that is created by calling the import_data() function.

Value

a `data.table`.

Examples

```
data <- import_data(example("coculture_peak_table.csv"),
  example("metadata.csv"),
  format = "Progenesis"
)

meta_data <- get_meta_data(data)
```

get_peak_table *Return the peak table data.table from the mpactr object.*

Description

`get_peak_table()` a wrapper function to return the peak table object of the given mpactr object.

Usage

```
get_peak_table(mpactr_object)
```

Arguments

`mpactr_object`	The mpactr object that is created by calling the import_data() function.

Value

a `data.table`.

Examples

```
data <- import_data(example("coculture_peak_table.csv"),
  example("metadata.csv"),
  format = "Progenesis"
)

peak_table <- get_peak_table(data)
```

`get_raw_data` *Return the input peak table from mpactr object.*

Description

`get_raw_data` a wrapper function to return the meta data object of the given mpactr object.

Usage

```
get_raw_data(mpactr_object)
```

Arguments

`mpactr_object`	The mpactr object that is created by calling the import_data() function.

Value

a `data.table`.

Examples

```
data <- import_data(example("coculture_peak_table.csv"),
  example("metadata.csv"),
  format = "Progenesis"
)

raw_data <- get_raw_data(data)
```

get_similar_ions *Get similar ion groups.*

Description

`get_similar_ions()` is a wrapper function to return similar ion groups determined with the `filter_mispicked_ions()`.

Usage

```
get_similar_ions(mpactr_object)
```

Arguments

`mpactr_object`	The mpactr object that is created by calling the import_data() function.

Value

a `data.table` reporting the main ion and those found to be similar with `filter_mispicked_ions()`.

Examples

```
data <- import_data(example("coculture_peak_table.csv"),
  example("metadata.csv"),
  format = "Progenesis"
)

data_filter <- filter_mispicked_ions(data)

mispicked_ion_groups <- get_similar_ions(data_filter)
mispicked_ion_groups
```

import_data *Import data into an mpactr object.*

Description

`import_data()` takes two file paths, one for the pre-processed feature table and one for sample metadata. Both files should be .csv.

Usage

```
import_data(peak_table, meta_data, format = "none")
```

Arguments

peak_table	The file path or valid https url to your feature table file.
meta_data	The file path to your meta_data file or data.frame.
format	The expected exported type of your peak table, can be one of "Progenesis", "Metaboscape", "None".

Details

mpactr requires a peak table and meta data as input. Files are expected to be comma separated files (*.csv*).

1. peak_table: a peak table where rows are expected to be compounds. mpactr supports import of feature table files from multiple tools through the format argument. Currently supported value for format are "Progenesis", "Metaboscape", or "None".

format = "Progensis" allows users to provide a feature table exported by Progenesis. To export a compatable peak table in Progenesis, navigate to the *Review Compounds* tab then File -> Export Compound Measurements. Select the following properties: Compound, m/z, Retention time (min), and Raw abundance and click ok.

format = "Metaboscape" allows users to provide a feature table exported by Metaboscape with default settings. The import function will save the raw peak table in the mpactr_object and store a formatted peak table for filtering. Reformatting includes selecting "FEATURE_ID", "RT", "PEPMASS", and sample columns. Sample columns are determined from the "Injection" column in meta_data (see below). "PEPMASS" is converted to m/z using the "ADDUCT" column and compound metadata columns are renamed for mpactr.

format = "None" allows users to provide a feature table file in the expected format. This can be useful if you have a file from another tool and want to manually format it in R. The table rows are expected to be individual features, while columns are compound metadata and samples. The feature table must have the compound metadata columns "Compound", "mz", and "rt". Where "Compound" is the compound id, and can be numeric or character. "mz" is the compound m/z, and should be numeric. "rt" is the retention time, in mintues, and should be numeric. The remaining columns should be samples, and match the names in the "Injection" column of the meta_data file. 2. meta_data: a table with sample information. Either a file path or data.frame can be supplied. At minimum the following columns are expected: "Injection", "Sample_Code", and "Biological_Group". "Injection" is the sample name and is expected to match sample column names in the peak_table. "Sample_Code" is the id for technical replicate groups. "Biological_Group" is the id for biological replicate groups. Other sample metadata can be added, and is encouraged for downstream analysis following filtering with mpactr.

Value

an mpactr_object.

Examples

```
data <- import_data(example("coculture_peak_table.csv"),
  example("metadata.csv"),
  format = "Progenesis"
```

```
)

meta_data <- read.csv(example("metadata.csv"))
data <- import_data(example("coculture_peak_table.csv"),
  meta_data,
  format = "Progenesis"
)
```

`plot_qc_tree` *Visualize Filtering Summary as Tree Map*

Description

`plot_qc_tree()` visualizes the filtering summary as a treemap. Ion status (see `qc_summary()`) is reported here as percentage of all pre-filtered ions.

Usage

```
plot_qc_tree(mpactr_object)
```

Arguments

`mpactr_object`	an `mpactr_object`.

Value

a tree map plot of class `ggplot`.

Examples

```
data <- import_data(example("coculture_peak_table.csv"),
  example("metadata.csv"),
  format = "Progenesis"
)

data_filter <- filter_mispicked_ions(data,
  ringwin = 0.5,
  isowin = 0.01,
  trwin = 0.005,
  max_iso_shift = 3,
  merge_peaks = TRUE
)

plot_qc_tree(data_filter)
```

qc_summary *Summary of Fitering*

Description

Parses an mpactr object and exracts a summary of all applied filters. Specifically, the fate of each input ion is reported as ion status. Status options are: Passed, mispicked, group, replicability, and insouce. A status of Passed ions is returned for ions that passed all applied filters and therefore are expected to be high quaility ions. Ions tagged as group, mispicked, replicability, or ionsource were removed during the correspoding filter.

Usage

```
qc_summary(mpactr_object)
```

Arguments

mpactr_object	an mpactr_object.

Value

a data.table reporting the number of high quality ions ("Passed") or the filter in which they were removed.

Examples

```
data <- import_data(example("coculture_peak_table.csv"),
  example("metadata.csv"),
  format = "Progenesis"
)

data_filter <- filter_mispicked_ions(data,
  ringwin = 0.5,
  isowin = 0.01,
  trwin = 0.005,
  max_iso_shift = 3,
  merge_peaks = TRUE
)

summary <- qc_summary(data_filter)
summary
```

Index