Package 'lipidr'

February 2, 2026

Title Data Mining and Analysis of Lipidomics Datasets

Version 2.25.0

Description lipidr an easy-to-use R package implementing a complete workflow for downstream analysis of targeted and untargeted lipidomics data. lipidomics results can be imported into lipidr as a numerical matrix or a Skyline export, allowing integration into current analysis frameworks. Data mining of lipidomics datasets is enabled through integration with Metabolomics Workbench API. lipidr allows data inspection, normalization, univariate and multivariate analysis, displaying informative visualizations. lipidr also implements a novel Lipid Set Enrichment Analysis (LSEA), harnessing molecular information such as lipid class, total chain length and unsaturation.

Depends R (>= 3.6.0), SummarizedExperiment

Imports methods, stats, utils, data.table, S4Vectors, rlang, dplyr, tidyr, forcats, ggplot2, limma, fgsea, ropls, imputeLCMD, magrittr

License MIT + file LICENSE

Suggests knitr, rmarkdown, BiocStyle, ggrepel, plotly, spelling, testthat

VignetteBuilder knitr

biocViews Lipidomics, MassSpectrometry, Normalization, QualityControl, Visualization

Language en-US

RoxygenNote 7.2.3

Roxygen list(markdown = TRUE)

URL https://github.com/ahmohamed/lipidr

BugReports https://github.com/ahmohamed/lipidr/issues/

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

git_branch devel

git_last_commit 364f6ac

git_last_commit_date 2025-10-29

Repository Bioconductor 3.23

Date/Publication 2026-02-01

Author Ahmed Mohamed [cre] (ORCID: <https://orcid.org/0000-0001-6507-5300>),
Ahmed Mohamed [aut],
Jeffrey Molendijk [aut]

Maintainer Ahmed Mohamed <mohamed@kuicr.kyoto-u.ac.jp>

Contents

lipidr-package *Analysis workflow for targeted lipidomics*

Description

lipidr implements a series of functions to facilitate inspection, analysis and visualization of targeted lipidomics datasets. lipidr takes exported Skyline CSV as input, allowing for multiple methods to be analyzed together.

Details

lipidr represents Skyline files as SummarizedExperiment objects, which can easily be integrated with a wide variety of Bioconductor packages. Sample annotations, such as sample group or other clinical information can be loaded. lipidr generates various plots, such as PCA score plots and box plots, for quality control of samples and measured lipids. Normalization methods with and without internal standards are also supported.

Differential analysis can be performed using any of the loaded clinical variables, which can be readily visualized as volcano plots. A novel lipid set enrichment analysis (LSEA) is implemented to detect preferential enrichment of certain lipid classes, total chain lengths or unsaturation patterns. Plots for the visualization of enrichment results are also implemented.

Author(s)

Ahmed Mohamed <ahmed.mohamed@qimrberghofer.edu.au>

add_sample_annotation *Add sample annotation to Skyline data frame*

Description

Add sample annotation to Skyline data frame

Usage

```
add_sample_annotation(data, annot_file)
```

Arguments

data	LipidomicsExperiment object.
annot_file	CSV file or a data.frame with at least 2 columns, sample names & group(s).

Value

LipidomicsExperiment with sample group information.

Examples

```
datadir <- system.file("extdata", package = "lipidr")

# all csv files
filelist <- list.files(datadir, "data.csv", full.names = TRUE)
d <- read_skyline(filelist)

# Add clinical info to existing LipidomicsExperiment object
clinical_file <- system.file("extdata", "clin.csv", package = "lipidr")
d <- add_sample_annotation(d, clinical_file)
colData(d)
d$group

# Subset samples using clinical information
# Note we are subsetting columns
d[, d$group == "QC"]

# Subset lipids using lipid annotation
# Note we are subsetting rows
d[rowData(d)$istd, ]
```

---

annotate_lipids *Parse molecule names to extract lipid class and chain information.*

Description

Parse lipid names to return a data.frame containing lipid class, total chain length and unsaturation. Lipids should follow the pattern 'class xx:x/yy:y', with class referring to the abbreviated lipid class, xx:x as the composition of the first chain and yy:y as the second chain. Alternatively, lipids can be supplied following the pattern 'class zz:z', where zz:z indicates the combined chain length and unsaturation information.

Usage

```
annotate_lipids(molecules, no_match = c("warn", "remove", "ignore"))
```

Arguments

molecules	A character vector containing lipid molecule names.
no_match	How to handle lipids that cannot be parsed? Default is to give warnings.

Value

A data.frame with lipid annotations as columns. Input lipid names are given in a column named "Molecule".

Examples

```
lipid_list <- c(
  "Lyso PE 18:1(d7)",
  "PE(32:0)",
  "Cer(d18:0/C22:0)",
  "TG(16:0/18:1/18:1)"
)
annotate_lipids(lipid_list)
```

`as_lipidomics_experiment` *Convert data.frame/matrix to LipidomicsExperiment*

Description

Convert data.frame/matrix to LipidomicsExperiment

Usage

```
as_lipidomics_experiment(df, logged = FALSE, normalized = FALSE)
```

Arguments

`df`	A data.frame or matrix where rows are lipids and columns are samples. Lipid names should be provided in the first column or in rownames of `df`. Measurements should be numeric. The data is considered `summarized` unless at least one lipid is duplicated (has > 1 row).
`logged`	Whether the data is log-transformed
`normalized`	Whether the data is normalized

Value

LipidomicsExperiment

`data_normalized` *Example dataset (normalized and log2 transformed)*

Description

A dataset containing MRM mass spectrometry-based lipidomics data from murine serum samples. Mice were fed a normal or high-fat diet and had access to normal drinking water or drinking water containing the bile acid deoxycholic acid. Lipid peaks were integrated using Skyline and exported results were imported into R using `lipidr`. The dataset has been normalized and log2 transformed. Please see normalize_pqn for details on how to generate this dataset.

Usage

```
data(data_normalized)
```

Format

An object of class `LipidomicsExperiment` with 278 rows and 56 columns.

See Also

Other lipidr datasets: `lipidDefaults`, `lipidnames_pattern`, `lipidr-data`

Examples

```
data(data_normalized)
```

---

de_analysis *Differential analysis of lipids between sample groups*

Description

`de_analysis` and `de_design` perform differential analysis of measured lipids that are associated with a sample group (annotation). `de_analysis` accepts a list of contrasts, while `de_design` allows users to define a design matrix, useful for complex experimental designs or for adjusting possible confounding variables.

Usage

```
de_analysis(data, ..., measure = "Area", group_col = NULL)

de_design(data, design, ..., coef = NULL, measure = "Area")

significant_molecules(de.results, p.cutoff = 0.05, logFC.cutoff = 1)

plot_results_volcano(de.results, show.labels = TRUE)
```

Arguments

`data`	LipidomicsExperiment object, should be normalized and log2 transformed.
`...`	Expressions, or character strings which can be parsed to expressions, specifying contrasts. These are passed to `limma::makeContrasts`.
`measure`	Which measure to use as intensity, usually Area (default).
`group_col`	Name of the column containing sample groups. If not provided, defaults to first sample annotation column.
`design`	Design matrix generated from `model.matrix()`, or a design formula.
`coef`	Column number or column name specifying which coefficient of the linear model is of interest.

de.results	Output of de_analysis().
p.cutoff	Significance threshold. Default is 0.05.
logFC.cutoff	Cutoff limit for log2 fold change. Default is 1. Ignored in multi-group (ANOVA-style) comparisons.
show.labels	Whether labels should be displayed for significant lipids. Default is TRUE.

Value

TopTable as returned by limma package

significant_molecules returns a character vector with names of significantly differentially changed lipids.

plot_results_volcano returns a ggplot object.

Functions

- significant_molecules(): gets a list of significantly changed lipids for each contrast.
- plot_results_volcano(): plots a volcano chart for differential analysis results.

Examples

```
# type ?normalize_pqn to see how to normalize and log2-transform your data
data(data_normalized)

# Specifying contrasts
de_results <- de_analysis(
  data_normalized,
  HighFat_water - NormalDiet_water,
  measure = "Area"
)
# Using formula
de_results_formula <- de_design(
  data = data_normalized,
  design = ~group,
  coef = "groupHighFat_water",
  measure = "Area"
)

# Using design matrix
design <- model.matrix(~group, data = colData(data_normalized))
de_results_design <- de_design(
  data = data_normalized,
  design = design,
  coef = "groupHighFat_water",
  measure = "Area"
)
significant_molecules(de_results)
plot_results_volcano(de_results, show.labels = FALSE)
```

filter_by_cv *Remove molecules with CV larger that a threshold*

Description

Remove molecules with CV larger that a threshold

Usage

```
filter_by_cv(data, cv.cutoff = 20, measure = "Area")
```

Arguments

data	LipidomicsExperiment object.
cv.cutoff	CV threshold (numeric). Default is 20.
measure	Which measure used to calculate CV, usually Area (default).

Value

LipidomicsExperiment object with molecules filtered.

Examples

```
data(data_normalized)
filter_by_cv(data_normalized)
```

gen_lipidsets *Generate lipid sets from lipid molecule names*

Description

Generate lipid sets from lipid molecule names

Usage

```
gen_lipidsets(molecules, min_size = 2)
```

Arguments

molecules	A character vector containing lipid molecule names.
min_size	Minimum number of molecules in a set to be included in enrichment.

Value

List of lipid sets

Examples

```
data(data_normalized)
molecules <- rowData(data_normalized)$Molecule
gen_lipidsets(molecules)
```

`impute_na` *Impute missing values in a LipidomicsExperiment*

Description

Impute missing values in a LipidomicsExperiment

Usage

```
impute_na(
  data,
  measure = "Area",
  method = c("knn", "svd", "mle", "QRILC", "minDet", "minProb", "zero"),
  ...
)
```

Arguments

`data`	LipidomicsExperiment object.
`measure`	Which measure to use as intensity, usually Area, Area Normalized or Height. Default is `Area`.
`method`	The imputation method to use. All methods are wrappers for `imputeLCMD` package. These include • knn Wraps `imputeLCMD::impute.wrapper.KNN()`. Default. This requires an additional argument `K` (Number of neighbors used to infer the missing data). • svd Wraps `imputeLCMD::impute.wrapper.SVD()`. This requires an additional argument `K` (Number of principal components to use). • mle Wraps `imputeLCMD::impute.wrapper.MLE()`, • minDet Wraps `imputeLCMD::impute.MinDet()`, • minProb Wraps `imputeLCMD::impute.MinProb()`, • zero Wraps `imputeLCMD::impute.ZERO()`,
`...`	Other arguments passed to the imputation method.

Value

LipidomicsExperiment object with missing values imputed.

Examples

```
data(data_normalized)

# Replace with values calculated using K-nearest neighbors
impute_na(data_normalized, "Area", "knn", 10)

# Replace with values calculated from the first K principal components
impute_na(data_normalized, "Area", "svd", 3)

# Replace with Maximum likelihood estimates
impute_na(data_normalized, "Area", "mle")

# Replace with randomly drawn values from a truncated distribution
impute_na(data_normalized, "Area", "QRILC")

# Replace with a minimal value
impute_na(data_normalized, "Area", "minDet")

# Replace with randomly drawn values from a Gaussian distribution
# cerntered around a minimal value
impute_na(data_normalized, "Area", "minProb")

# Replace with zero (not recommended)
impute_na(data_normalized, "Area", "zero")
```

---

is_logged *Functions to get and set attributes of LipidomicsExperiment objects*

Description

Functions to get and set attributes of LipidomicsExperiment objects

Usage

```
is_logged(data, measure)

set_logged(data, measure, val)

is_normalized(data, measure)

set_normalized(data, measure, val)

is_summarized(data)

set_summarized(data, val)
```

Arguments

data	LipidomicsExperiment object.
measure	Which measure to get / set attributes of.
val	Value to be assigned to the attribute.

Value

Modified LipidomicsExperiment.

Examples

```
data(data_normalized)
is_logged(data_normalized, "Area")
is_summarized(data_normalized)
```

---

lipidDefaults *Default values for lipidr internal functions A set of default mappings and annotation used internally to correctly parse lipid molecule names.*

Description

Default values for lipidr internal functions A set of default mappings and annotation used internally to correctly parse lipid molecule names.

Usage

```
data(lipidDefaults)
```

Format

An object of class `list` of length 2.

See Also

Other lipidr datasets: `data_normalized`, `lipidnames_pattern`, `lipidr-data`

Examples

```
data(lipidDefaults)
```

lipidnames_pattern *Patterns used in parsing lipid names*

Description

A collection of patterns to extract lipid class and chain information from lipid names. Used internally by the package.

Usage

```
data(lipidnames_pattern)
```

Format

An object of class `list` of length 8.

See Also

Other lipidr datasets: `data_normalized`, `lipidDefaults`, `lipidr-data`

Examples

```
data(lipidnames_pattern)
```

LipidomicsExperiment *Constructor for Lipidomics experiment from list of assays*

Description

Constructor for Lipidomics experiment from list of assays

Usage

```
LipidomicsExperiment(assay_list, metadata, colData = NULL, rowData = NULL)
```

Arguments

`assay_list`	A list or SimpleList of matrix-like elements, or a matrix-like object. Passed to `SummarizedExperiment()`.
`metadata`	A list containing arbitrary information about the experiment. It should at least contain 2 elements: • dimnames 2-element character vector with dimension names • summarized Has transitions been summarized?
`colData`	An optional DataFrame describing the samples (contains clinical information). Row names, if present, become the column names of the LipidomicsExperiment.

rowData — A DataFrame object describing the rows (contains generated lipid annotations). Row names, if present, become the row names of the SummarizedExperiment object. The number of rows of the DataFrame must be equal to the number of rows of the matrices in assays.

Value

LipidomicsExperiment object

`LipidomicsExperiment-class` *LipidomicsExperiment object*

Description

LipidomicsExperiment object

`lipidr-data` *Description of lipidr datasets*

Description

lipidr-package has 3 datasets:

- data_normalized Example lipidomics dataset, normalized & log2-transformed.
- lipidDefaults A list of default mappings and annotations for lipids.
- lipidnames_pattern A list of patterns used in parsing lipid names.

See below for detailed description of each dataset.

See Also

Other lipidr datasets: `data_normalized`, `lipidDefaults`, `lipidnames_pattern`

Examples

```
data(data_normalized)
```

`list_mw_studies` *Metabolomics Workbench integration*

Description

These functions use Metabolomics Workbench REST API to support data mining of publicly available lipidomics datasets.

Usage

```
list_mw_studies(keyword = "lipid")

fetch_mw_study(study_id)

read_mwTab(mwTab)

read_mw_datamatrix(file)
```

Arguments

`keyword`	A keyword to search for in Metabolomics Workbench studies.
`study_id`	The Metabolomics Workbench study ID.
`mwTab`	File path or url for a mwTab file.
`file`	File path or url for the file containing the data matrix.

Value

`list_mw_studies` returns a data frame with studies matching the keyword. Study ID, title, author and details are retrieved.

All other functions return a LipidomicsExperiment object containing clinical and lipid intensity data.

Functions

- `list_mw_studies()`: retrieves a list of lipidomics studies from Metabolomics Workbench.
- `fetch_mw_study()`: downloads and parse full data for a study from Metabolomics Workbench using a study ID. The function returns a LipidomicsExperiment where users can directly apply `lipidr` analysis workflow.
- `read_mwTab()`: parses mwTab file into a LipidomicsExperiment.
- `read_mw_datamatrix()`: parses a Metabolomics Workbench data matrix into a LipidomicsExperiment. Data matrix downloaded from Metabolomics Workbench are parsed into a LipidomicsExperiment object to enable `lipidr` workflow analysis.

Examples

```
## Not run:
list_mw_studies()

## End(Not run)
## Not run:
fetch_mw_study("ST001111")

## End(Not run)
```

`lsea` *Lipid set enrichment analysis (LSEA)*

Description

Lipid set enrichment analysis (LSEA)

Usage

```
lsea(
  de.results,
  rank.by = c("logFC", "P.Value", "adj.P.Val"),
  min_size = 2,
  ...
)

significant_lipidsets(enrich.results, p.cutoff = 0.05, size.cutoff = 2)

plot_class_enrichment(de.results, significant.sets, measure = "logFC")

plot_enrichment(
  de.results,
  significant.sets,
  annotation = c("class", "length", "unsat"),
  measure = "logFC"
)
```

Arguments

`de.results`	Output of `de_analysis()`.
`rank.by`	Statistic used to rank the lipid list. Default is `logFC`.
`min_size`	Minimum number of molecules in a set to be included in enrichment.
`...`	Extra parameters passed to `fgsea::fgsea()`.
`enrich.results`	Output of `lsea()`.
`p.cutoff`	Significance threshold. Default is `0.05`.

size.cutoff	Minimum number of lipids in a set tested for enrichment. Default is 2.
significant.sets	List of significantly changed lipid sets (output of significant_lipidsets()).
measure	Which measure to plot the distribution of: logFC, P.Value, Adj.P.Val. Default is logFC.
annotation	Which lipid set collection to plot.

Value

lsea returns enrichment results (data.frame) as returned from fgsea::fgsea(). The results also contain the following attributes:

- de.results Original de.results input.
- rank.by Measure used to rank lipid molecules.
- sets Lipid sets tested, with their member molecules.

significant_lipidsets returns a list of character vectors of significantly enriched sets for each contrast.

plot_enrichment returns a ggplot object.

Functions

- significant_lipidsets(): gets a list of significantly changed lipid sets
- plot_enrichment(): is usually used to look at log2 fold change distribution of lipids in each class, chain length or unsaturation, marking significantly enriched sets. It can also be used to plot P.Value or Adj.P.Val.

Examples

```
data(data_normalized)
de_results <- de_analysis(
  data_normalized,
  HighFat_water - NormalDiet_water,
  measure = "Area"
)
enrich_results <- lsea(
  de_results,
  rank.by = "logFC", min_size = 4, nperm = 1000
)
sig_lipidsets <- significant_lipidsets(enrich_results)
plot_enrichment(de_results, sig_lipidsets, annotation="class")
plot_enrichment(de_results, sig_lipidsets, annotation="length")
```

mva *Perform multivariate analyses to investigate sample clustering*

Description

mva performs multivariate analysis using several possible methods. The available methods are PCA, PCoA, OPLS and OPLS-DA. The OPLS method requires a numeric y-variable, whilst OPLS-DA requires two groups for comparison. By default, for OPLS and OPLS-DA the number of predictive and orthogonal components are set to 1. Blank samples are automatically detected (using TIC) and excluded. Missing data are imputed using average lipid intensity across all samples.

Usage

```
mva(
  data,
  measure = "Area",
  method = c("PCA", "PCoA", "OPLS", "OPLS-DA"),
  group_col = NULL,
  groups = NULL,
  ...
)

plot_mva(
  mvaresults,
  components = c(1, 2),
  color_by = NULL,
  ellipse = TRUE,
  hotelling = TRUE
)

plot_mva_loadings(
  mvaresults,
  components = c(1, 2),
  color_by = NULL,
  top.n = nrow(mvaresults$loadings)
)

top_lipids(mvaresults, top.n = 10)
```

Arguments

data	LipidomicsExperiment object.
measure	Which measure to use as intensity, usually Area (default). The measure should be already summarized and normalized.
method	Either PCA, PCoA, OPLS or OPLS-DA. Default is PCA.

group_col	Sample annotation to use as grouping column. If not provided, samples are treated independently.
groups	A numeric grouping (OPLS) or two groups to be used for supervised analysis (OPLS-DA), ignored in other methods.
...	Extra arguments to be passed to opls() for OPLS-DA, ignored in other methods.
mvaresults	Results obtained from mva().
components	Which components to plot. Ignored for PCoA, OPLS and OPLS-DA results. Default is first 2 components.
color_by	Sample annotation (or lipid annotation in case of plot_mva_loadings) to use as color. Defaults to individual samples / lipids
ellipse	Whether to plot ellipses around groups
hotelling	Whether to plot Hotelling T2.
top.n	Number of top ranked features to highlight in the plot. If omitted, returns top 10 lipids.

Value

Multivariate analysis results in mvaresults object. The object contains the following:

- scores Sample scores
- loadings Feature or component loadings (not for PCoA)
- method Multivariate method that was used
- row_data Lipid molecule annotations
- col_data Sample annotations
- original_object Original output object as returned by corresponding analysis methods

plot_mva returns a ggplot of the sample scores.

plot_mva_loadings returns a ggplot of the loadings.

top_lipids returns s dataframe of top.n lipids with their annotations.

Functions

- plot_mva(): plots a multivariate scatterplot of sample scores to investigate sample clustering.
- plot_mva_loadings(): Plot a multivariate scatterplot of feature loadings to investigate feature importance.
- top_lipids(): extracts top lipids from OPLS-DA results

Examples

```
data(data_normalized)

# PCA
mvaresults <- mva(data_normalized, measure = "Area", method = "PCA")
plot_mva(mvaresults, color_by = "group")
```

```
# NOT RUN
# plot_mva(mvaresults, color_by = "Diet", components = c(2, 3))

# PCoA
mvaresults <- mva(data_normalized, measure = "Area", method = "PCoA")
# NOT RUN
# plot_mva(mvaresults, color_by = "group")

# OPLS-DA
mvaresults <- mva(
  data_normalized,
  method = "OPLS-DA", group_col = "Diet", groups = c("HighFat", "Normal")
)
plot_mva(mvaresults, color_by = "group")
plot_mva_loadings(mvaresults, color_by = "Class", top.n = 10)
top_lipids(mvaresults, top.n = 10)
```

`non_parsed_molecules` *Get a list of molecules that couldn't be parsed by* `lipidr`

Description

Get a list of molecules that couldn't be parsed by `lipidr`

Usage

```
non_parsed_molecules(data)
```

Arguments

`data`	LipidomicsExperiment object.

Value

A character vector of the molecule names that could not be parsed.

Examples

```
data(data_normalized)
non_parsed_molecules(data_normalized)
```

normalize_istd *Normalize each class by its corresponding internal standard(s).*

Description

Normalize each class by its corresponding internal standard(s). Lipid classes are normalized using corresponding internal standard(s) of the same lipid class. If no corresponding internal standard is found the average of all measured internal standards is used instead.

Usage

```
normalize_istd(data, measure = "Area", exclude = "blank", log = TRUE)
```

Arguments

data	LipidomicsExperiment object.
measure	Which measure to use as intensity, usually Area, Area Normalized or Height. Default is Area.
exclude	Samples to exclude, can be either: "blank" - automatically detected blank samples and exclude them logical vector with the same length as samples. Default.
log	whether the normalized values should be log2 transformed. Default is TRUE.

Value

A LipidomicsExperiment object with normalized values. Each molecule is normalized against the internal standard from the same class.

Examples

```
datadir <- system.file("extdata", package = "lipidr")
filelist <- list.files(datadir, "data.csv", full.names = TRUE)
d <- read_skyline(filelist)
clinical_file <- system.file("extdata", "clin.csv", package = "lipidr")
d <- add_sample_annotation(d, clinical_file)
d_summarized <- summarize_transitions(d, method = "average")

# Normalize data that have been summarized (single value per molecule).
data_norm_istd <- normalize_istd(
  d_summarized,
  measure = "Area", exclude = "blank", log = TRUE
)
```

normalize_pqn *Perform Probabilistic Quotient Normalization for intensities.*

Description

Perform Probabilistic Quotient Normalization (PQN) for sample intensities. The PQN method determines a dilution factor for each sample by comparing the distribution of quotients between samples and a reference spectrum, followed by sample normalization using this dilution factor. The reference spectrum in this method is the average lipid abundance of all samples (excluding blanks).

Usage

```
normalize_pqn(data, measure = "Area", exclude = "blank", log = TRUE)
```

Arguments

data	LipidomicsExperiment object.
measure	Which measure to use as intensity, usually Area, Area Normalized or Height. Default is Area.
exclude	Samples to exclude, can be either: "blank" - automatically detected blank samples and exclude them logical vector with the same length as samples. Default.
log	Whether the normalized values should be log2 transformed. Default is TRUE.

Value

A LipidomicsExperiment object with normalized values

References

Dieterle, F., Ross, A., Schlotterbeck, G., & Senn, H. (2006). Probabilistic quotient normalization as robust method to account for dilution of complex biological mixtures. Application in 1H NMR metabonomics. Analytical chemistry, 78(13), 4281-4290.

Examples

```
datadir <- system.file("extdata", package = "lipidr")
filelist <- list.files(datadir, "data.csv", full.names = TRUE)
d <- read_skyline(filelist)
clinical_file <- system.file("extdata", "clin.csv", package = "lipidr")
d <- add_sample_annotation(d, clinical_file)
d_summarized <- summarize_transitions(d, method = "average")

# Normalize data that have been summarized (single value per molecule).
data_normalized <- normalize_pqn(
  d_summarized,
  measure = "Area", exclude = "blank", log = TRUE
)
```

plot_chain_distribution *Plot logFC of lipids per class showing chain information*

Description

Plot a chart of (log2) fold changes of lipids per class showing chain lengths and unsaturations. If multiple molecules with the same total chain length and unsaturation are present in the dataset, the `measure` is averaged, and the number of molecules is indicated on the plot.

Usage

```
plot_chain_distribution(de_results, contrast = NULL, measure = "logFC")
```

Arguments

`de_results`	Output of `de_analysis()`.
`contrast`	Which comparison to plot. if not provided, defaults to the the first comparison.
`measure`	Which measure to plot the distribution of: logFC, P.Value, Adj.P.Val. Default is `logFC`

Value

A ggplot object.

Examples

```
data(data_normalized)
de_results <- de_analysis(
  data_normalized,
  HighFat_water - NormalDiet_water,
  measure = "Area"
)
plot_chain_distribution(de_results)
```

plot_lipidclass *Informative plots to investigate lipid classes*

Description

lipidr supports two types of plots for to visualize at lipid classes.

sd plots a bar chart for standard deviation of a certain measure in each class. This plot type is usually used to look at standard deviations of intensity in each class, but can also be used to look at different measures such as Retention Time, to ensure all lipids are eluted within the expected range. To assess instrumental variation apply the function to technical quality control samples.

boxplot Plots a boxplot chart to examine the distribution of values per class. This plot type is usually used to look at the intensity distribution in each class, but can also be used to look at different measures, such as Retention Time or Background.

Usage

```
plot_lipidclass(data, type = c("boxplot", "sd"), measure = "Area", log = TRUE)
```

Arguments

data	LipidomicsExperiment object.
type	plot type, either boxplot or sd. Default is boxplot.
measure	Which measure to plot the distribution of: usually Area, Area Normalized, Height or Retention Time. Default is Area
log	Whether values should be log2 transformed. Default is TRUE (Set FALSE for retention time).

Value

A ggplot object.

Examples

```
data(data_normalized)

d_qc <- data_normalized[, data_normalized$group == "QC"]
plot_lipidclass(d_qc, "sd", "Area", log = TRUE)
plot_lipidclass(d_qc, "sd", "Retention Time", log = FALSE)
plot_lipidclass(d_qc, "boxplot", "Area", log = TRUE)
plot_lipidclass(d_qc, "boxplot", "Retention Time", log = FALSE)
```

---

plot_molecules *Informative plots to investigate individual lipid molecules*

Description

lipidr supports three types of plots for to visualize at lipid molecules.

cv plots a bar chart for coefficient of variation of lipid molecules. This plot type is usually used to investigate the CV in lipid intensity or retention time, in QC samples.

sd plots a bar chart for standard deviations of a certain measure in each lipid. This plot type is usually used to look at standard deviation of intensity for each lipid, but can also be used to look at different measures such as Retention Time, to ensure all lipids elute within expected range.

boxplot plots a boxplot chart to examine the distribution of values per lipid. This plot type is usually used to look at intensity distribution for each lipid, but can also be used to look at different measures, such as Retention Time or Background.

Usage

```
plot_molecules(
  data,
  type = c("cv", "sd", "boxplot"),
  measure = "Area",
  log = TRUE,
  color = "Class"
)
```

Arguments

data	LipidomicsExperiment object.
type	plot type, either cv, sd or boxplot. Default is cv.
measure	Which measure to plot the distribution of: usually Area, Area Normalized or Height. Default is Area
log	Whether values should be log2 transformed (Set FALSE for retention time). Default is TRUE
color	The column name of a row annotation to be used as color

Value

A ggplot object.

Examples

```
data(data_normalized)
d_qc <- data_normalized[, data_normalized$group == "QC"]

# plot the variation in intensity and retention time of all measured
#   lipids in QC samples
plot_molecules(d_qc, "cv", "Area")
plot_molecules(d_qc, "cv", "Retention Time", log = FALSE)
```

```
# plot the variation in intensity, RT of ISTD (internal standards)
#   in QC samples
d_istd_qc <- data_normalized[
  rowData(data_normalized)$istd,
  data_normalized$group == "QC"
]
plot_molecules(d_istd_qc, "sd", "Area")
plot_molecules(d_istd_qc, "sd", "Retention Time", log = FALSE)

plot_molecules(d_istd_qc, "boxplot")
plot_molecules(d_istd_qc, "boxplot", "Retention Time", log = FALSE)
```

`plot_samples` *Informative plots to investigate samples*

Description

`lipidr` supports two types of plots for sample quality checking.

`tic` plots a bar chart for total sample intensity.

`boxplot` plots a boxplot chart to examine the distribution of values per sample.

Usage

```
plot_samples(
  data,
  type = c("tic", "boxplot"),
  measure = "Area",
  log = TRUE,
  color = NULL
)
```

Arguments

`data`	LipidomicsExperiment object.
`type`	plot type, either `tic` or `boxplot`. Default is `tic`.
`measure`	Which measure to use as intensity, usually Area, Area Normalized or Height. Default is `Area`
`log`	Whether values should be log2 transformed. Default is `TRUE`
`color`	The column name of a sample annotation to be used as color

Value

A ggplot object.

Examples

```
data(data_normalized)

plot_samples(data_normalized, type = "tic", "Area", log = TRUE)
plot_samples(data_normalized, type = "tic", "Background", log = FALSE)
plot_samples(
  data_normalized[, data_normalized$group == "QC"],
  type = "boxplot",
  measure = "Retention Time", log = FALSE
)
```

`plot_trend` *Plot a regulation trend line between logFC and chain annotation*

Description

Fit and plot a regression line of (log2) fold changes and total chain lengths or unsaturations. If multiple comparisons are included, one regression is plotted for each.

Usage

```
plot_trend(de_results, annotation = c("length", "unsat"))
```

Arguments

`de_results`	Output of `de_analysis`.
`annotation`	Whether to fit trend line against chain `length` or `unsat`.

Value

A ggplot object.

Examples

```
data(data_normalized)
de_results <- de_analysis(
  data_normalized,
  HighFat_water - NormalDiet_water,
  NormalDiet_DCA - NormalDiet_water,
  measure = "Area"
)
plot_trend(de_results, "length")
```

read_skyline *Read Skyline exported files*

Description

Read Skyline exported files

Usage

```
read_skyline(files)
```

Arguments

files	Character vector with filepaths to Skyline exported files in CSV format.

Value

LipidomicsExperiment object.

Examples

```
datadir <- system.file("extdata", package = "lipidr")

# all csv files
filelist <- list.files(datadir, "data.csv", full.names = TRUE)
d <- read_skyline(filelist)

# View automatically generated lipid annotations
rowData(d)
```

remove_non_parsed_molecules *Remove molecules that couldn't be parsed by* lipidr *from the dataset*

Description

Remove molecules that couldn't be parsed by lipidr from the dataset

Usage

```
remove_non_parsed_molecules(data)
```

Arguments

data	LipidomicsExperiment object.

Value

A filtered LipidomicsExperiment object.

Examples

```
data(data_normalized)
remove_non_parsed_molecules(data_normalized)
```

`summarize_transitions` *Summarize transitions*

Description

Calculate a single intensity for molecules with multiple transitions, by determining the average or maximum intensity.

Usage

```
summarize_transitions(data, method = c("max", "average"))
```

Arguments

`data`	LipidomicsExperiment object.
`method`	Choose to summarize multiple transitions by taking the average or maximum intensity. Default is `max`

Value

A LipidomicsExperiment object with single intensities per lipid molecule

Examples

```
datadir <- system.file("extdata", package = "lipidr")
filelist <- list.files(datadir, "data.csv", full.names = TRUE)
d <- read_skyline(filelist)
clinical_file <- system.file("extdata", "clin.csv", package = "lipidr")
d <- add_sample_annotation(d, clinical_file)
d_summarized <- summarize_transitions(d, method = "average")
```

update_molecule_names *Rename molecules in a dataset.*

Description

This function enables users to rename selected molecules in the dataset, so that they can be parsed correctly by `lipidr` or modify the lipid class. `lipidr` automatically updates the annotation for the renamed molecules.

Usage

```
update_molecule_names(data, old, new)
```

Arguments

`data`	LipidomicsExperiment object.
`old`	A character vector of the molecule names to be renamed.
`new`	A character vector of the new molecule names.

Value

A LipidomicsExperiment object with molecules name and annotation updated.

Examples

```
data(data_normalized)
old_names <- rowData(data_normalized)$Molecule
# replace PC0 with plasmenylPC
new_names <- sub("^LPE", "LysoPE", old_names)
update_molecule_names(data_normalized, old_names, new_names)
```

use_interactive_graphics *Activate interactive graphics*

Description

Use this function to turn on/off interactive graphics plotting. Interactive plots require `plotly` to be installed. Interactive graphics are disabled by default.

Usage

```
use_interactive_graphics(interactive = TRUE)
```

Arguments

interactive	Should interactive plots be displayed? Default is TRUE.

Value

None

Examples

```
data(data_normalized)
use_interactive_graphics()

# plot the variation in intensity and retention time of all measured
#  lipids in QC samples
d_qc <- data_normalized[, data_normalized$group == "QC"]
# plot_molecules(d_qc, "cv", "Area")

# turn off interactivity
use_interactive_graphics(interactive = FALSE)
```

%>% *Pipe operator*

Description

See `magrittr::%>%` for details.

Usage

```
lhs %>% rhs
```

Value

Result of `rhs(lhs, ...)`.

Examples

```
data(data_normalized)
data_normalized %>% filter_by_cv()
```

Index