Package: ume (via r-universe)

October 13, 2026

Title Ultrahigh-Resolution Mass Spectrometry Data Evaluation for Complex Organic Matter

Version 1.7.1

Description Provides tools for assigning molecular formulas from exact masses obtained by ultrahigh-resolution mass spectrometry. The methodology follows the workflow described in Leefmann et al. (2019) <doi:10.1002/rcm.8315>. The package supports the inspection, filtering and visualization of molecular formula data and includes utilities for calculating common molecular parameters (e.g., double bond equivalents, DBE). A graphical user interface is available via the 'shiny'-based 'ume' application.

URL https://gitlab.awi.de/bkoch/ume, https://github.com/boriskoch/ume, https://doi.org/10.5281/zenodo.20108477, https://www.awi.de/en/ume

Depends R (>= 4.2.0)

Imports data.table, ggplot2 (>= 3.5.0), plotly, vegan, viridis, jsonlite

License MIT + file LICENSE

Encoding UTF-8

LazyData true

LazyDataCompression xz

Roxygen list(markdown = TRUE)

Suggests rmarkdown, knitr, testthat (>= 3.0.0), xml2, pdftools, KEGGREST, zCompositions

VignetteBuilder knitr

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

RoxygenNote 8.0.0

Config/pak/sysreqs cmake make libicu-dev libuv1-dev libssl-dev

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

Date/Publication 2026-07-13 20:38:37 UTC

RemoteUrl https://github.com/boriskoch/ume

RemoteRef HEAD

RemoteSha e59b8eb447e140c8c3c4e4349eace4786fe380b0

Contents

add_known_mf . . . 4
add_missing_element_columns . . . 6
as_mfd . . . 7
as_peaklist . . . 8
as_ume_matrix . . . 9
assign_formulas . . . 10
calc_data_summary . . . 12
calc_dbe . . . 14
calc_eval_params . . . 15
calc_exact_mass . . . 17
calc_ideg . . . 18
calc_isotope_pattern . . . 19
calc_iterr . . . 22
calc_logratio . . . 24
calc_ma . . . 26
calc_ma_abs . . . 27
calc_neutral_mass . . . 27
calc_nm . . . 28
calc_norm_int . . . 29
calc_number_assignment . . . 31
calc_pielou_evenness . . . 32
calc_shannon_index . . . 33
calc_simpson_index . . . 34
calc_time_flattened_mfd . . . 35
check_neutral_mf . . . 36
classify_files . . . 38
color.palette . . . 39
convert_data_table_to_molecular_formulas . . . 40
convert_molecular_formula_to_data_table . . . 42
create_isotope_expanded_table . . . 43
create_ume_formula_library . . . 44
download_library . . . 46
eval_isotopes . . . 47
filter_int . . . 48
filter_mass_accuracy . . . 49
filter_mf_data . . . 50
filter_multi_assignments . . . 52
get_isotope_info . . . 54

inchi_to_mf . . . 55
is_ume_peaklist . . . 56
known_mf . . . 56
lib_demo . . . 57
masses . . . 58
mf_data_demo . . . 59
order_columns . . . 61
peaklist_demo . . . 61
remove_blanks . . . 62
remove_empty_columns . . . 64
remove_id_columns . . . 65
remove_unknown_columns . . . 65
subset_known_mf . . . 66
theme_uplots . . . 67
ume_assign_formulas . . . 67
ume_filter_formulas . . . 69
ume_name_dictionary . . . 71
uplot_cluster . . . 72
uplot_cvm . . . 74
uplot_dbe_minus_o_freq . . . 76
uplot_dbe_vs_c . . . 78
uplot_dbe_vs_ma . . . 80
uplot_dbe_vs_o . . . 81
uplot_freq . . . 83
uplot_freq_ma . . . 84
uplot_freq_vs_ppm . . . 85
uplot_hc_vs_m . . . 87
uplot_heteroatoms . . . 88
uplot_isotope_precision . . . 89
uplot_kmd . . . 91
uplot_lcms . . . 93
uplot_ma_vs_mz . . . 94
uplot_ms . . . 95
uplot_n_mf_per_sample . . . 97
uplot_pca . . . 98
uplot_ppm_avg . . . 100
uplot_ratios . . . 101
uplot_reproducibility . . . 102
uplot_ri_vs_sample . . . 103
uplot_vk . . . 105
ustats_outlier . . . 107
zero_filling . . . 108

Index **110**

add_known_mf *Add metainformation derived from ume::known_mf*

Description

Annotate molecular formulas using metadata from `ume::known_mf`.

The function joins molecular formulas with predefined molecular formula categories such as:

- `CRAM`
- `marine_dom`
- `surfactant`
- `peptide`
- `nucleotide`

Categories are controlled through `ume::ume_name_dictionary` using entries with:

```
semantic_type == "formula_category"
```

This allows centralized management of active categories and labels.

The function works with:

- character vectors of molecular formulas,
- `data.frames`,
- and `data.tables`.

Usage

```
add_known_mf(
  mfd,
  mf_col = "mf",
  known_mf = ume::known_mf,
  category_dictionary = ume::ume_name_dictionary,
  wide = FALSE,
  ...
)
```

Arguments

`mfd`	data.table with molecular formula data as derived from `ume::assign_formulas`. Column names of elements/isotopes must match names in the `isotope` column of `ume::masses`; values are integers representing counts per formula.
`mf_col`	Character. Name of the molecular formula column if `mfd` is a table. Default is `"mf"`.

known_mf — A data.table containing known molecular formulas and their assigned categories. Default is ume::known_mf.

category_dictionary — A data.table containing molecular formula category definitions. Default is ume::ume_name_dictionary.

wide — Logical.

If FALSE (default), return a single comma-separated categories column.

If TRUE, create one column per category (CRAM, surfactant, ...).

... — Reserved for future extensions.

Details

Add metainformation derived from ume::known_mf

Molecular formulas are matched against known_mf.

If wide = FALSE (default), matching categories are returned as a comma-separated character column named categories.

If wide = TRUE, one numeric column per category is returned (CRAM, surfactant, ...).

Category selection is dynamically controlled through ume::ume_name_dictionary, allowing centralized management of known molecular formula categories.

Value

A data.table containing the original input plus additional molecular formula category information.

Author(s)

Boris P. Koch

References

CRAM

Hertkorn N., Benner R., Frommberger M., Schmitt-Kopplin P., Witt M., Kaiser K., Kettrup A., Hedges J.I. (2006). Characterization of a major refractory component of marine dissolved organic matter. *Geochimica et Cosmochimica Acta*, **70**, 2990-3010. doi:10.1016/j.gca.2006.03.021

Surfactants

Lechtenfeld O.J., Koch B.P., Gasparovic B., Frka S., Witt M., Kattner G. (2013). The influence of salinity on the molecular and optical properties of surface microlayers in a karstic estuary. *Marine Chemistry*, **150**, 25-38. doi:10.1016/j.marchem.2013.01.006

Ideg

Flerus R., Lechtenfeld O.J., Koch B.P., McCallister S.L., Schmitt-Kopplin P., Benner R., Kaiser K., Kattner G. (2012). A molecular perspective on the ageing of marine dissolved organic matter. *Biogeosciences*, **9**, 1935-1955. doi:10.5194/bg919352012

iTerr

Medeiros P.M., Seidel M., Niggemann J., Spencer R.G.M., Hernes P.J., Yager P.L., Miller W.L., Dittmar T., Hansell D.A. (2016). A novel molecular approach for tracing terrigenous dissolved organic matter into the deep ocean. *Global Biogeochemical Cycles*, **30**, 689-699. doi:10.1002/2015gb005320

See Also

Other Formula assignment: `calc_eval_params()`, `check_formula_library()`, `eval_isotopes()`, `ume_assign_formulas()`

Examples

```
# Character vector input
add_known_mf(
  c("C10H12O5", "C20H30O10")
)

# Molecular formula table
add_known_mf(
  mfd = mf_data_demo
)

# Wide format
add_known_mf(
  mfd = mf_data_demo,
  wide = TRUE
)
```

`add_missing_element_columns`

Add Missing Isotope Columns to mfd

Description

This function ensures that missing isotope columns are added to the input data table (`mfd`), which is required for further data evaluation that considers isotope information. If any of the specified isotope columns are not already present in the data, they will be added with a default value of `0`.

The function is typically used to standardize the dataset by ensuring that all expected isotopes (e.g., nitrogen-15, carbon-13) are represented, even if they are not initially present in the data. The function works by checking for the existence of each specified isotope column and adding the missing ones.

Usage

```
add_missing_element_columns(mfd, missing_cols = "15n")
```

Arguments

`mfd`	data.table with molecular formula data as derived from `ume::assign_formulas`. Column names of elements/isotopes must match names in the `isotope` column of `ume::masses`; values are integers representing counts per formula.
`missing_cols`	A character vector of isotope column names that should be checked and added if missing. By default, it includes `"15n"`, but additional isotopes can be specified as needed (e.g., `"na"`, `"d"`, `"35cl"`, etc.).

Value

A `data.table` object with the missing isotope columns added, where missing columns are populated with a default value of `0`. The original `mfd` object is modified in place.

See Also

Other tools: `order_columns()`

Examples

```
# Add missing isotope columns to a demo dataset
mfd_with_isotopes <- add_missing_element_columns(mfd = mf_data_demo)

# Add a specific isotope column for Nitrogen-15 (if missing)
mfd_with_15n <- add_missing_element_columns(mfd = mf_data_demo, missing_cols = c("15n", "na"))
```

---

as_mfd *Coerce input to UME molecular formula data*

Description

Flexible entry point for molecular formula data in UME.

`as_mfd()` normalizes column names using `ume::ume_name_dictionary`, adds missing structural columns where possible, optionally calculates Kendrick mass defect (`kmd`), validates required columns, and returns a standardized `data.table`.

Usage

```
as_mfd(
  mfd,
  required_cols = c("file_id", "mz", "mf", "m", "ppm"),
  target = "mfd",
  dictionary = ume::ume_name_dictionary,
  add_peak_id = TRUE,
  add_kmd = TRUE,
  kmd_digits = 3,
  kmd_base_mass = 14.01565,
  track_original_names = TRUE,
```

```
  ...
)
```

Arguments

mfd	A data.frame or data.table containing molecular formula data.
required_cols	Character vector of required UME columns.
target	Character. Target dictionary context. Default is "mfd".
dictionary	Dictionary table used for column-name mapping. Default is ume::ume_name_dictionary.
add_peak_id	Logical. If TRUE, add peak_id if missing.
add_kmd	Logical. If TRUE, add kmd if missing and column m is available.
kmd_digits	Integer. Number of digits used for KMD rounding.
kmd_base_mass	Numeric. Kendrick base mass. Default is CH2.
track_original_names	Logical. If TRUE, store original column-name mapping as attribute "original_colnames".
...	Reserved for future extensions.

Details

Coerce input to UME molecular formula data

Value

A standardized molecular formula data.table with attribute ume_type = "mfd".

See Also

Other check ume objects: as_peaklist(), check_formula_library(), check_mfd()

---

as_peaklist *Check format of peaklist*

Description

Flexible entry point for UME. Accepts:

- data.frame / data.table peaklists
- numeric m/z vectors
- file paths (csv, txt, tsv, rds)

Normalizes column names, adds missing structural columns (file_id, peak_id), removes invalid rows, validates schema, and assigns the UME peaklist class. Creates a standardized data.table ready for formula assignment.

Usage

```
as_peaklist(pl, verbose = FALSE, track_original_names = TRUE, ...)
```

Arguments

pl	Input object representing a peaklist. Can be: • data.frame or data.table • file path to a supported tabular format • numeric vector of m/z values
verbose	logical; if TRUE, show progress messages.
track_original_names	Logical (default: TRUE). If TRUE, as_peaklist() stores a "original_colnames" attribute mapping canonical UME names (e.g. "mz") to the user's original column names (e.g. "m/z"). Internal functions that perform many := operations (e.g. assign_formulas()) may set this to FALSE to avoid attribute- related shallow-copy warnings.
...	Reserved for future extensions.

Value

A validated and normalized peaklist as a data.table with class "ume_peaklist".

See Also

Other check ume objects: as_mfd(), check_formula_library(), check_mfd()

---

as_ume_matrix *Convert molecular formula data to a feature matrix*

Description

Converts long molecular formula data into a wide sample-by-feature matrix for multivariate statistics.

Rows are defined by grp, columns by feature_col, and values by value_col. Duplicate group-feature combinations are aggregated.

Usage

```
as_ume_matrix(
  mfd,
  grp = "file_id",
  feature_col = "mf",
  value_col = "norm_int",
  fun.aggregate = mean,
  fill = 0,
  as_matrix = TRUE
)
```

Arguments

mfd	data.table with molecular formula data as derived from ume::assign_formulas. Column names of elements/isotopes must match names in the isotope column of ume::masses; values are integers representing counts per formula.
grp	Character vector. Names of columns (e.g., sample or file identifiers) used to aggregate results.
feature_col	Character. Column defining features. Default is "mf".
value_col	Character. Column defining values. Default is "norm_int".
fun.aggregate	Function used for duplicate combinations. Default is mean.
fill	Value used for missing combinations. Default is 0.
as_matrix	Logical. If TRUE, return a numeric matrix. If FALSE, return a data.frame with row names.

Value

A numeric matrix or data.frame with groups as rows and features as columns.

Examples

```
mat <- as_ume_matrix(
  mfd = mf_data_demo,
  grp = "file_id",
  value_col = "norm_int"
)
```

---

assign_formulas *Molecular Formula Assignment*

Description

Assigns molecular formulas to molecular masses using a predefined library. Input of the peaklist (pl) is internally checked as_peaklist(), converted to neutral masses calc_neutral_mass(), and assigned with molecular formulas based on the mass accuracy (ma_dev) provided calc_ma_abs(). The input can be either:

- A peaklist (data.table) containing m/z values or neutral masses and additional metadata .
- A numeric vector of m/z values or neutral masses without additional metadata (internally checked and standardized by as_peaklist()).

Usage

```
assign_formulas(pl, formula_library, verbose = FALSE, ...)
```

Arguments

pl	Either a peaklist (data.table) with at least columns mz, i_magnitude, and file_id, or a numeric vector of masses. For numeric input, a minimal peaklist is constructed internally.
formula_library	Molecular formula library: a predefined data.table used for assigning molecular formulas to a peak list and for mass calibration. The library requires a fixed format, including mass values for matching. Predefined libraries are available in the R package *ume.formulas* and further described in Leefmann et al. (2019). A standard library for marine dissolved organic matter is ume.formulas::lib_02. New libraries can be built using ume::create_ume_formula_library().
verbose	logical; if TRUE, show progress messages.
...	Arguments passed on to calc_ma_abs, calc_neutral_mass m Measured mass ma_dev Mass accuracy in +/- parts per million (ppm) pol Character: "neg", "pos", or "neutral". mz String. Name of the column that contains mass-to-charge information (default: "mz").

Details

This function calculates the neutral mass of peaks in pl and compares it to mass values in formula_library, assigning molecular formulas based on mass accuracy thresholds. If 13C, 15N, or 34S isotope information is missing, additional columns are added to the output table.

Value

A data.table where each row represents a molecular formula assigned to a mass peak. The table contains:

- All columns of the input peaklist pl (e.g. mz, i_magnitude, file_id).
- All columns of the input formula_library (e.g. mf, element counts).
- Calculated columns:
  - m — neutral mass.
  - m_cal — exact mass of the assigned formula.
  - del — absolute mass error (Da).
  - ppm — mass error in parts per million.
  - mf_id — unique ID for each (file_id, mf) combination.
- Added isotope columns (13C, 15N, 34S) if missing in the library.

One peak may receive zero, one, or multiple assigned formulas depending on the mass accuracy threshold.

Author(s)

Boris P. Koch

Examples

```
# Example using demo data and demo peak list:



assign_formulas(pl = peaklist_demo,
                formula_library = ume::lib_demo,
                pol = "neg",
                ma_dev = 0.2,
                verbose = FALSE)

 # Example using a given mass and UME demo library:
 mfd <- assign_formulas(pl = 254.0426527, formula_library = ume::lib_demo,
 pol = "neutral", ma_dev = 0.5, verbose = TRUE)
```

`calc_data_summary` *Create a Data Summary Table for Element Ratios and Parameters*

Description

Generates a data summary table that provides intensity-weighted averages for element ratios, mass accuracy, and additional parameters. Results can be grouped based on the specified grouping columns.

Usage

```
calc_data_summary(mfd, grp = "file_id", ...)
```

Arguments

`mfd`	data.table with molecular formula data as derived from `ume::assign_formulas`. Column names of elements/isotopes must match names in the `isotope` column of `ume::masses`; values are integers representing counts per formula.
`grp`	Character vector. Names of columns (e.g., sample or file identifiers) used to aggregate results.
`...`	Additional arguments passed to methods.

Details

This function computes a variety of weighted averages and summary statistics for mass spectrometry data using the provided peak list (`mfd`). Calculated values include weighted averages for elemental counts (e.g., Carbon, Hydrogen), elemental ratios (e.g., O/C, H/C), and additional parameters such as the base peak intensity and summed intensities. It also calculates the aromaticity index (`wa(AI)`) based on the elemental composition. If grouping columns are provided, the summary statistics are calculated for each group.

The function also joins additional indices (`ideg`, `iterr`) from related functions `calc_ideg()` and `calc_iterr()` to the final summary table.

Value

A `data.table` containing the summarized results, with columns including:

n(mf) Number of molecular formulas per group.

accuracy (median) Median accuracy in parts-per-million (ppm) for the identified peaks.

accuracy (3 sigma cut-off) Maximum ppm accuracy within a three-sigma range.

wa(mz) Weighted average m/z value.

wa(DBE) Weighted average Double Bond Equivalent (DBE).

wa(element) Weighted averages for elements (C, H, N, O, P, S) and ratios (O/C, H/C, N/C, S/C).

wa(NOSC) Weighted average nominal oxidation state of carbon.

wa(delG0_Cox) Weighted average Gibbs free energy (Cox) in kJ/mol.

wa(AI) Weighted average aromaticity index.

wa(C/N) and wa(C/S) Ratios derived from N/C and S/C.

ideg, ideg_n Indices for degree of degradation, as calculated by `calc_ideg()`.

iterr, iterr_n, iterr2, iterr2_n Terrestrial indices derived from `calc_iterr()`.

median(i_magnitude) Median intensity value.

int(basepeak) Intensity of the base peak.

int(summed) Summed intensity of all peaks.

References

Koch B.P., Dittmar T. (2006). From mass to structure: an aromaticity index for high-resolution mass data of natural organic matter. *Rapid Communications in Mass Spectrometry*, **20**, 926-932. doi:10.1002/rcm.2386

LaRowe D.E., Van Cappellen P. (2011). Degradation of natural organic matter: A thermodynamic analysis. *Geochimica et Cosmochimica Acta*, **75**, 2030-2042. doi:10.1016/j.gca.2011.01.020

Medeiros P.M., Seidel M., Niggemann J., Spencer R.G.M., Hernes P.J., Yager P.L., Miller W.L., Dittmar T., Hansell D.A. (2016). A novel molecular approach for tracing terrigenous dissolved organic matter into the deep ocean. *Global Biogeochemical Cycles*, **30**, 689-699. doi:10.1002/2015gb005320

See Also

Other calculations: `calc_dbe()`, `calc_eval_params()`, `calc_exact_mass()`, `calc_ideg()`, `calc_ma()`, `calc_neutral_mass()`, `calc_nm()`, `calc_norm_int()`, `calc_number_assignment()`, `calc_number_occurrence()`, `calc_recalibrate_ms()`

Examples

```
# Example using demo data, grouping by file ID
calc_data_summary(mfd = mf_data_demo, grp = c("file_id"))
```

`calc_dbe` *Calculate Double Bond Equivalent (DBE)*

Description

Calculates the Double Bond Equivalent (DBE) for a given neutral molecular formula. DBE is a measure of unsaturation, representing the total number of rings and pi bonds in a molecule. The function uses the `ume::masses` data table to determine valence information for each element in the input molecular formula. #' It can be calculated from the molecular formula using atomic valences:

$$\mathrm{DBE} = 1 + \frac{1}{2}\sum_i n_i(v_i - 2)$$

where:

- n_i: number of atoms of element i
- v_i: valence of element i (e.g., C = 4, H = 1, N = 3, O = 2, S = 2/4/6 depending on bonding state)

This formula works for any set of elements as long as their valence is known. Be aware that some elements can have more than one valence at normal conditions (e.g. Sulfur can have valences of 2, 4 and 6). The function uses the valence that is represented in `ume:masses$valence`.

For a reasonable neutral molecule DBE has an integer value >=0. A higher DBE indicates a more unsaturated structure; a lower DBE indicates a more saturated structure.

Usage

```
calc_dbe(mfd, masses = ume::masses, verbose = FALSE, ...)
```

Arguments

`mfd`	data.table with molecular formula data as derived from `ume::assign_formulas`. Column names of elements/isotopes must match names in the `isotope` column of `ume::masses`; values are integers representing counts per formula.
`masses`	A data.table. Defaults to `ume::masses` (based on NIST data) containing isotope information for elements, including nominal and exact mass, relative abundance, and Hill system order.
`verbose`	logical; if `TRUE`, show progress messages.
`...`	Additional arguments passed to methods.

Details

This function computes DBE based on the molecular formula specified in `mfd`. `mfd` can be a data.table or a character string or character vector of molecular formula strings.

For each isotope in the formula, DBE is calculated as the sum of (valence - 2) multiplied by the count of that isotope, divided by 2, and then adding 1. Elements with a valence of 2 are excluded from the DBE calculation.

The function stops with an informative error if valence information is missing for any element or isotope present in `mfd`.

Value

A numeric vector of the same length as the number of rows in `mfd`, where each entry represents the calculated DBE for the corresponding molecular formula. The result vector is named 'dbe'.

See Also

Other calculations: `calc_data_summary()`, `calc_eval_params()`, `calc_exact_mass()`, `calc_ideg()`, `calc_ma()`, `calc_neutral_mass()`, `calc_nm()`, `calc_norm_int()`, `calc_number_assignment()`, `calc_number_occurrence()`, `calc_recalibrate_ms()`

Examples

```
# Example with user-defined data
calc_dbe("C6H10O6")
calc_dbe("C6H10Br2")
calc_dbe(c("C3[13C1]H10O4", "C6H10O6"))

# Example with demo data from UME package
calc_dbe(mfd = mf_data_demo)
```

---

calc_eval_params *Calculate UME Evaluation Parameters*

Description

This function calculates and adds several evaluation parameters as additional columns to the `mfd` data table. These parameters are essential for evaluating the molecular structure and isotopic distribution, enabling further analysis. For a detailed description of the output table, see `help(mf_data_demo)`.

Usage

```
calc_eval_params(mfd, verbose = FALSE, ...)
```

Arguments

`mfd`	data.table with molecular formula data as derived from `ume::assign_formulas`. Column names of elements/isotopes must match names in the `isotope` column of `ume::masses`; values are integers representing counts per formula.
`verbose`	logical; if `TRUE`, show progress messages.
`...`	Additional arguments passed to methods.

Value

The original `data.table` `mfd` with additional evaluation columns:

`nm` Nominal molecular mass: Calculated if not already present.

`dbe` Double Bond Equivalent (measure of unsaturation).

`kmd` Kendrick mass defect for CH4 versus O exchange.

`O/C`, `H/C`, `N/C`, `S/C` Element ratios for a molecular formula.

`nsp_type`, `snp_check` Types of combinations of N, S, and P atoms in a formula.

`nosc` Weighted average nominal oxidation state of carbon.

`delG0_Cox` Weighted average Gibbs free energy (Cox) in kJ/mol.

`ai` Aromaticity index.

`ppm_filt` A mass accuracy threshold calculated for each spectrum.

Author(s)

Boris P. Koch

References

Hughey C.A., Hendrickson C.L., Rodgers R.P., Marshall A.G., Qian K.N. (2001). Kendrick mass defect spectrum: A compact visual analysis for ultrahigh-resolution broadband mass spectra. *Analytical Chemistry*, **73**, 4676-4681. doi:10.1021/ac010560w

Koch B.P., Dittmar T. (2006). From mass to structure: an aromaticity index for high-resolution mass data of natural organic matter. *Rapid Communications in Mass Spectrometry*, **20**, 926-932. doi:10.1002/rcm.2386

LaRowe D.E., Van Cappellen P. (2011). Degradation of natural organic matter: A thermodynamic analysis. *Geochimica et Cosmochimica Acta*, **75**, 2030-2042. doi:10.1016/j.gca.2011.01.020

See Also

Other Formula assignment: `add_known_mf()`, `check_formula_library()`, `eval_isotopes()`, `ume_assign_formulas()`

Other calculations: `calc_data_summary()`, `calc_dbe()`, `calc_exact_mass()`, `calc_ideg()`, `calc_ma()`, `calc_neutral_mass()`, `calc_nm()`, `calc_norm_int()`, `calc_number_assignment()`, `calc_number_occurrence()`, `calc_recalibrate_ms()`

Examples

```
# Example usage with a demo molecular formula dataset
mfd_with_params <- calc_eval_params(mfd = mf_data_demo, verbose = TRUE)
```

`calc_exact_mass` *Calculate Exact Monoisotopic Mass of a Molecule*

Description

This function calculates the exact monoisotopic mass for each molecule in a given data table based on the specified isotope composition. Exact masses of elements and isotopes used in the calculation are retrieved from the `ume::masses` data, based on data from NIST (https://www.nist.gov/pml/atomic-weights-and-isotopic-compositions-relative-atomic-masses).

Usage

```
calc_exact_mass(mfd, ...)
```

Arguments

`mfd`	data.table with molecular formula data as derived from `ume::assign_formulas`. Column names of elements/isotopes must match names in the `isotope` column of `ume::masses`; values are integers representing counts per formula.
`...`	Additional arguments passed to methods.

Value

A numeric vector of the calculated exact monoisotopic mass.

Author(s)

Boris P. Koch

See Also

Other calculations: `calc_data_summary()`, `calc_dbe()`, `calc_eval_params()`, `calc_ideg()`, `calc_ma()`, `calc_neutral_mass()`, `calc_nm()`, `calc_norm_int()`, `calc_number_assignment()`, `calc_number_occurrence()`, `calc_recalibrate_ms()`

Examples

```
# Example with demo data
calc_exact_mass(mfd = mf_data_demo)
# Custom example
calc_exact_mass(data.table::data.table(c = 3, h = 8, o = 1))
```

`calc_ideg` *Calculate Degradation Index (Ideg)*

Description

This function calculates the degradation index ('Ideg') following Flerus et al. (2012). High Ideg values indicate 'older' marine DOM (i.e., a higher contribution of peaks that correlate negatively with delta14C), while low values indicate 'younger' DOM (i.e., a higher contribution of peaks that correlate positively with delta14C)./

Ideg is computed as the ratio of summed magnitudes for five negative (NEG) molecular formulas to the total summed magnitudes of five positive (POS) and five negative (NEG) molecular formulas:

$$Ideg = \frac{\sum NEG}{\sum NEG + \sum POS}$$

The index ranges from 0 to 1 and is valid only if all required formulas (n = 10) are present. Ideg depends strongly on the type of sample preparation, ionization method, and instrument settings, and should only be interpreted for relative changes within the same dataset.

Usage

```
calc_ideg(
  mfd,
  mf_col = "mf",
  magnitude_col = "i_magnitude",
  grp = "file_id",
  ...
)
```

Arguments

`mfd`	data.table with molecular formula data as derived from `ume::assign_formulas`. Column names of elements/isotopes must match names in the `isotope` column of `ume::masses`; values are integers representing counts per formula.
`mf_col`	Character. The name of the column containing molecular formulas. Default is "mf".
`magnitude_col`	Character. The name of the column containing magnitude values (absolute or relative). Default is "i_magnitude".
`grp`	Character vector. Names of columns (e.g., sample or file identifiers) used to aggregate results.
`...`	Additional arguments passed to methods.

Value

A data.table with columns:

- grp: Grouping variable.
- ideg: Calculated degradation index (rounded to 3 decimals).
- ideg_n: Number of assigned formulas used in the calculation.

See Also

Other calculations: calc_data_summary(), calc_dbe(), calc_eval_params(), calc_exact_mass(), calc_ma(), calc_neutral_mass(), calc_nm(), calc_norm_int(), calc_number_assignment(), calc_number_occurrence(), calc_recalibrate_ms()

Examples

```
# Create a minimal dataset containing all required POS and NEG formulas
library(data.table)

demo_ideg <- data.table(
  file_id = 1,
  mf = c(
    "C17H20O9", "C19H22O10", "C20H22O10", "C20H24O11", "C21H26O11",   # NEG
    "C13H18O7", "C14H20O7", "C15H22O7", "C15H22O8", "C16H24O8"        # POS
  ),
  i_magnitude = c(
    1200, 900, 1500, 700, 800,     # NEG intensities
    2000, 1800, 2200, 1600, 1900   # POS intensities
  )
)

calc_ideg(
  mfd = demo_ideg,
  mf_col = "mf",
  magnitude_col = "i_magnitude",
  grp = "file_id"
)
```

---

calc_isotope_pattern *Calculate Isotope Pattern*

Description

Calculates the theoretical isotope pattern of a molecular formula based on natural isotope abundances using multinomial/binomial isotope combinations.

Usage

```
calc_isotope_pattern(
  mf,
  masses = ume::masses,
  threshold = 1e-12,
  rel_threshold = 1e-06,
  max_peaks = 5000L,
  mass_digits = 6L
)
```

Arguments

`mf`	A character vector of molecular formulas or a `data.table` containing isotope count columns.
`masses`	A data.table. Defaults to `ume::masses` (based on NIST data) containing isotope information for elements, including nominal and exact mass, relative abundance, and Hill system order.
`threshold`	Numeric. Minimum absolute isotope probability retained during intermediate calculations.
`rel_threshold`	Numeric. Minimum relative abundance retained in the final isotope pattern.
`max_peaks`	Integer. Maximum number of isotope peaks retained during intermediate calculations.
`mass_digits`	Integer. Number of decimal places used to merge nearly identical masses during intermediate calculations.

Details

Calculate Theoretical Isotope Pattern

The function calculates all relevant isotope combinations for each element in a molecular formula and combines them into a theoretical isotope pattern.

For each isotope peak, the function returns the exact mass, nominal mass, absolute probability, relative abundance, elemental molecular formula (`mf`), and isotope-specific molecular formula (`mf_iso`).

The isotope-specific molecular formula uses bracket notation, for example `[12C2][13C][1H6][16O]`.

Very small isotope peaks can be removed using `threshold` and `rel_threshold` to keep the output compact.

Value

A `data.table` with one row per isotope peak and the following columns:

mf Elemental molecular formula.

mf_iso Isotope-specific molecular formula.

mass Exact mass of the isotope composition.

nominal_mass Nominal mass of the isotope composition.

prob Absolute probability of the isotope composition.

relative_abundance Relative abundance normalized to the most abundant isotope peak.

isotope_peak Peak number ordered by increasing mass.

Mathematical background

The theoretical isotope pattern is calculated from the natural isotope abundances of all elements present in a molecular formula.

For an element with n atoms and k naturally occurring isotopes, all possible isotope compositions are generated and their probabilities are calculated using the multinomial distribution:

$$P(x_1, \ldots, x_k) = \frac{n!}{x_1! x_2! \cdots x_k!} \prod_{i=1}^{k} p_i^{x_i}$$

where x_i is the number of atoms of isotope i, p_i is its natural abundance, and $\sum x_i = n$.

For elements with only two relevant isotopes (e.g. carbon, ^{12}C and ^{13}C), the multinomial distribution reduces to the binomial distribution:

$$P(m) = \binom{n}{m} p^m (1-p)^{n-m}$$

where m is the number of heavy isotopes.

In this special case, the relative isotope abundances follow the coefficients of Pascal's triangle. For example, the carbon isotope pattern of butane (C_4H_{10}) can be represented as:

$$(^{12}C + ^{13}C)^4 = 1 : 4 : 6 : 4 : 1$$

corresponding to the isotope peaks M, M+1, M+2, M+3 and M+4.

Exact masses are calculated from the isotope composition and isotope patterns of individual elements are subsequently combined by convolution, yielding the theoretical isotope pattern of the complete molecular formula.

The implementation therefore provides exact multinomial isotope distributions for all supported elements and molecular formulas.

References

Yergey JA, Heller DN, Hansen G, Cotter RJ, Fenselau C (1983). "Isotopic distributions in mass spectra of large molecules". Analytical Chemistry, 55(2), 353–356. doi:10.1021/ac00252a730

Valkenborg D, Mertens I, Lemiere F, Witters E, Burzykowski T (2012). "The isotopic distribution conundrum". Mass Spectrometry Reviews, 31(1), 96–109. doi:10.1002/mas.20339

See Also

Other isotopes: `create_isotope_expanded_table()`, `eval_isotopes()`, `uplot_isotope_precision()`

Examples

```
calc_isotope_pattern("C4H10")
calc_isotope_pattern("C2H6O")
calc_isotope_pattern("FeC10H10", rel_threshold = 1e-4)
```

`calc_iterr` *Calculate terrestrial indeces Iterr and Iterr2 (after Medeiros et al. 2016)*

Description

Calculate a degradation index 'Iterr' and modified index 'iterr2' after Medeiros et al. (2016). High Iterr values represent higher contribution of terrestrial material (i.e. higher contribution of peaks that correlate positively with delta13C) while low values represent less terrestrial material (i.e. higher contribution of peaks that correlate negatively with delta13C). Iterr / Iterr2 are calculated from a peak magnitude ratio of 50 or 5 POS and NEG formulas, respectively: sum(terr) / (sum(terr) + sum(marine)) Therefore Iterr / Iterr2 range between 1 and 0. It should be noted that absolute values strongly depend on factors such as type of solid phase extraction, ionization method, instrument settings etc. Therefore values can only be interpreted as relative changes. It should also be noted that for an appropriate evaluation ALL index formulas must be present.

Usage

```
calc_iterr(
  mfd,
  mf_col = "mf",
  magnitude_col = "i_magnitude",
  grp = "file_id",
  ...
)
```

Arguments

`mfd`	data.table with molecular formula data as derived from `ume::assign_formulas`. Column names of elements/isotopes must match names in the `isotope` column of `ume::masses`; values are integers representing counts per formula.
`mf_col`	Name of the column containing molecular formulas (string)
`magnitude_col`	Name of the column containing absolute or relative mass peak magnitudes (string).
`grp`	Character vector. Names of columns (e.g., sample or file identifiers) used to aggregate results.
`...`	Additional arguments passed to methods.

Value

Iterr and iterr2 values

References

Medeiros P.M., Seidel M., Niggemann J., Spencer R.G.M., Hernes P.J., Yager P.L., Miller W.L., Dittmar T., Hansell D.A. (2016). A novel molecular approach for tracing terrigenous dissolved organic matter into the deep ocean. *Global Biogeochemical Cycles*, **30**, 689-699. doi:10.1002/2015gb005320

Examples

```
library(data.table)

# Create a minimal dataset containing all required
# POS, NEG, POS2, and NEG2 formulas for demonstration

demo_iterr <- data.table(
  file_id = 1,
  mf = c(
    # NEG (Iterr)
    'C13H12O5','C15H14O4','C14H12O5','C14H14O5','C13H12O6',
    'C16H16O4','C15H14O5','C14H12O6','C15H16O5','C14H14O6',
    'C16H14O5','C16H16O5','C15H14O6','C15H16O6','C14H14O7',
    'C17H16O5','C16H14O6','C17H18O5','C16H16O6','C15H14O7',
    'C17H16O6','C16H14O7','C18H18O6','C17H16O7','C17H18O7',
    'C18H16O7','C18H18O7','C17H16O8','C19H18O7','C20H20O7',
    'C19H18O8','C20H18O9','C19H16O10','C21H20O9','C20H18O10',
    'C22H22O9','C21H20O10','C23H22O10','C24H24O10','C25H26O10',

    # POS (Iterr)
    'C15H19NO6','C15H21NO6','C17H21NO7','C17H23NO7','C17H22O8',
    'C16H21NO8','C17H20N2O7','C17H19NO8','C18H23NO7','C17H21NO8',
    'C18H24O8','C16H19NO9','C17H23NO8','C17H22O9','C17H24O9',
    'C18H21NO8','C17H19NO9','C18H23NO8','C18H22O9','C17H21NO9',
    'C18H24O9','C18H20N2O8','C18H21NO9','C19H24O9','C18H23NO9',
    'C18H22O10','C18H24O10','C20H24O9','C19H22O10','C20H26O9',
    'C19H24O10','C19H26O10','C20H24O10','C20H26O10','C19H24O11',
    'C20H24O11','C20H26O11','C20H26O12','C22H28O11','C21H28O12',

    # NEG2 (Iterr2)
    'C17H18O7','C18H18O7','C17H16O7','C17H16O8','C15H16O6',

    # POS2 (Iterr2)
    'C20H24O9','C20H24O10','C19H22O10','C17H21NO8','C20H26O9'
  ),

  # Assign magnitude values (arbitrary but valid)
  i_magnitude = c(
    rep(1000, 40),  # NEG
    rep(2000, 40),  # POS
    rep(1500, 5),   # NEG2
    rep(1800, 5)    # POS2
  )
)
```

```
calc_iterr(
  mfd = demo_iterr,
  mf_col = "mf",
  magnitude_col = "i_magnitude",
  grp = "file_id"
)
```

`calc_logratio` *calc_logratio*

Description

Calculates log-ratio transformed values for compositional molecular formula data and returns either the original long table with added log-ratio columns or a complete group-feature table.

Usage

```
calc_logratio(
  mfd,
  grp = "file_id",
  feature_col = "mf",
  value_col = "i_magnitude",
  method = c("clr", "alr", "rclr"),
  zero = c("min", "median", "CZM", "GBM", "none"),
  zero_factor = 0.65,
  reference = NULL,
  reclose = TRUE,
  remove_empty = TRUE,
  z_warning = 1,
  fun.aggregate = mean,
  complete = FALSE
)
```

Arguments

`mfd`	data.table with molecular formula data as derived from `ume::assign_formulas`. Column names of elements/isotopes must match names in the `isotope` column of `ume::masses`; values are integers representing counts per formula.
`grp`	Character. Column defining samples or groups. Default is `"file_id"`.
`feature_col`	Character. Column defining features. Default is `"mf"`.
`value_col`	Character. Column defining values. Default is `"i_magnitude"`.
`method`	Character. Log-ratio method. One of `"clr"`, `"alr"`, or `"rclr"`.
`zero`	Character. Zero-replacement method passed to `zero_filling()`. One of `"min"`, `"median"`, `"CZM"`, `"GBM"`, or `"none"`.
`zero_factor`	Numeric. Multiplication factor for `zero = "min"`. Default is `0.65`.

reference	Character or integer. Reference feature for ALR. If NULL, the feature with the lowest log-variance after zero replacement is used.
reclose	Logical. If TRUE, rows are normalized to unit sum after zero replacement. Passed to zero_filling(). Default is TRUE.
remove_empty	Logical. If TRUE, columns containing only zeros are removed before zero replacement. Passed to zero_filling(). Default is TRUE.
z_warning	Numeric. Passed to zero_filling() and then to zCompositions::cmultRepl() if zero is "CZM" or "GBM".
fun.aggregate	Function used for duplicate group-feature combinations. Default is mean.
complete	Logical. If TRUE, return the complete group-feature table generated from the log-ratio matrix. If FALSE, return the original mfd with added log-ratio columns. Default is FALSE.

Details

Calculate log-ratio values for molecular formula data

The input must be a molecular formula table in long format. Internally, the data are converted to a sample-by-feature matrix using as_ume_matrix(), zero-filled using zero_filling(), log-ratio transformed using vegan::decostand(), and then converted back to long format.

If complete = FALSE, the original mfd is returned with two additional columns: logratio_type and logratio_value.

If complete = TRUE, the returned table contains all group-feature combinations present in the transformed matrix. This is useful for PCA and clustering based on logratio_value.

Value

A data.table with the columns logratio_type and logratio_value. If complete = FALSE, all original mfd columns are retained. If complete = TRUE, the output contains grp, feature_col, logratio_type, and logratio_value.

Examples

```
mfd_lr <- calc_logratio(
  mfd = mf_data_demo,
  grp = "file_id",
  feature_col = "mf",
  value_col = "i_magnitude",
  method = "clr",
  zero = "min"
)

## Not run:
mfd_lr_complete <- calc_logratio(
  mfd = mf_data_demo,
  grp = "file_id",
  feature_col = "mf",
  value_col = "i_magnitude",
  method = "clr",
```

```
  zero = "min",
  complete = TRUE
)

res_pca <- uplot_pca(
  mfd = mfd_lr_complete,
  grp = "file_id",
  int_col = "logratio_value"
)

## End(Not run)
```

`calc_ma` *Calculate mass accuracy*

Description

Calculates relative mass accuracy (ma, in parts per million) as:

$(m_{meas} - m_{calc})/m_{calc} \times 10^6$ where:

- m_{meas} = measured mass
- m_{calc} = calculated / theoretical (exact) mass

Returned value is rounded to 4 digits. In this context the theoretical mass is represented by the mass of the assigned molecular formula. A small absolute **ppm value** indicates a very precise measurement and increases confidence in correct molecular formula assignment.

Usage

```
calc_ma(m, m_cal, ...)
```

Arguments

`m`	Measured mass
`m_cal`	Calculated (theoretical) mass.
`...`	Additional arguments passed to methods.

Value

A numeric vector of mass accuracy (rounded to 4 decimals).

See Also

Other calculations: `calc_data_summary()`, `calc_dbe()`, `calc_eval_params()`, `calc_exact_mass()`, `calc_ideg()`, `calc_neutral_mass()`, `calc_nm()`, `calc_norm_int()`, `calc_number_assignment()`, `calc_number_occurrence()`, `calc_recalibrate_ms()`

Examples

```
# Use of single values
calc_ma(m = 264.08641, m_cal = 264.08653)
# Use in a molecular formula table
calc_ma(m = mf_data_demo$m, m_cal = mf_data_demo$m_cal)
mf_data_demo[, .(m, m_cal, accuracy_in_ppm = calc_ma(m, m_cal))]
```

---

calc_ma_abs *Calculate absolute mass accuracy range (ma)*

Description

This function calculates the absolute mass accuracy range for a neutral mass (m) at a given a mass accuracy (ma_dev).

Usage

```
calc_ma_abs(m, ma_dev, ...)
```

Arguments

m	Measured mass
ma_dev	Mass accuracy in +/- parts per million (ppm)
...	Additional arguments passed to methods.

Value

Returns a list with two values: m_min, m_max

Examples

```
calc_ma_abs(m = 327.0134, ma_dev = 0.5)
```

---

calc_neutral_mass *Calculate neutral molecular mass*

Description

Calculates neutral molecular masses for singly charged ions with full numerical precision. No user options are modified.

The conversion used is:

- negative mode: m = mz + 1.0072763
- positive mode: m = mz - 1.0072763
- neutral: m = mz

Usage

```
calc_neutral_mass(mz, pol = c("neg", "pos", "neutral"), ...)
```

Arguments

mz	String. Name of the column that contains mass-to-charge information (default: "mz").
pol	Character: "neg", "pos", or "neutral".
...	Additional arguments passed to methods.

Value

Numeric vector of neutral masses.

See Also

Other calculations: calc_data_summary(), calc_dbe(), calc_eval_params(), calc_exact_mass(), calc_ideg(), calc_ma(), calc_nm(), calc_norm_int(), calc_number_assignment(), calc_number_occurrence(), calc_recalibrate_ms()

Examples

```
calc_neutral_mass(199.32, pol = "neg")
```

---

calc_nm *Calculate Nominal Mass of a Molecule*

Description

Computes the nominal mass (integer mass) for each molecular formula in the provided data. This function uses isotope masses stored in the dataset ume::masses, based on values from NIST, for accurate calculation of each element's nominal mass contribution.

Usage

```
calc_nm(mfd, ...)
```

Arguments

mfd	data.table with molecular formula data as derived from ume::assign_formulas. Column names of elements/isotopes must match names in the isotope column of ume::masses; values are integers representing counts per formula.
...	Additional arguments passed to methods.

Details

The function calculates the nominal mass of each molecular formula by retrieving the relevant integer mass values of isotopes from `ume::masses`. This information is processed to create a calculation string which is then evaluated to obtain the nominal mass for each molecule.

The nominal mass is derived by summing the integer masses of each constituent element in the formula, where the integer mass for each element is multiplied by the number of atoms of that element in the molecule.

Note: This function depends on `ume::get_isotope_info()` for isotope data retrieval.

Value

A numeric vector of the calculated nominal mass.

See Also

Other calculations: `calc_data_summary()`, `calc_dbe()`, `calc_eval_params()`, `calc_exact_mass()`, `calc_ideg()`, `calc_ma()`, `calc_neutral_mass()`, `calc_norm_int()`, `calc_number_assignment()`, `calc_number_occurrence()`, `calc_recalibrate_ms()`

Examples

```
# Example using a demo dataset to calculate nominal mass
calc_nm(mfd = mf_data_demo)
```

---

calc_norm_int *Calculate Normalized Peak Intensities*

Description

Computes normalized peak intensities for a molecular formula dataset and adds the results as additional columns to the input `data.table` (`mfd`). It also calculates:

- the number of molecular formula assignments per peak (`n_assignments`)
- the total occurrences of each formula across the dataset (`n_occurrence`)

Normalized intensities are stored in a new column `norm_int`, and the reference intensity used for normalization is stored in `int_ref`.

Supported normalization methods:

- `"none"` – no normalization; raw peak intensities are copied to `norm_int`
- `"bp"` – normalized to the base peak intensity per spectrum
- `"sum"` – normalized by the total sum of intensities per spectrum
- `"sum_ubiq"` – normalized by the sum of intensities of ubiquitous peaks across the dataset
- `"sum_rank"` – normalized by the sum of the top `n_rank` most intense peaks per spectrum
- `"euc"` – Euclidean normalization (optional, not implemented in current version)

Usage

```
calc_norm_int(
  mfd,
  ms_id = "file_id",
  peak_id = "peak_id",
  peak_magnitude = "i_magnitude",
  normalization = c("bp", "sum", "sum_ubiq", "sum_rank", "none"),
  n_rank = 200,
  verbose = FALSE,
  ...
)
```

Arguments

mfd	data.table with molecular formula data as derived from ume::assign_formulas. Column names of elements/isotopes must match names in the isotope column of ume::masses; values are integers representing counts per formula.
ms_id	Character; name of the column identifying individual spectra (default: "file_id").
peak_id	Character; name of the column identifying unique peaks (default: "peak_id").
peak_magnitude	Character; name of the column containing peak intensity values (default: "i_magnitude").
normalization	Character; normalization method to apply. One of "bp", "sum", "sum_ubiq", "sum_rank", "none". Default is "bp".
n_rank	Integer; number of top-ranked peaks to use for "sum_rank" normalization (default: 200).
verbose	logical; if TRUE, show progress messages.
...	Additional arguments (currently unused).

Value

A data.table identical to mfd but with additional columns:

norm_int Normalized peak intensity based on selected method.

int_ref Reference intensity used for normalization (e.g., sum, base peak).

n_assignments Number of formula assignments per peak (calculated internally).

n_occurrence Number of occurrences of each formula across all spectra (calculated internally).

See Also

Other calculations: calc_data_summary(), calc_dbe(), calc_eval_params(), calc_exact_mass(), calc_ideg(), calc_ma(), calc_neutral_mass(), calc_nm(), calc_number_assignment(), calc_number_occurrence(calc_recalibrate_ms()

Examples

```
mfd_norm <- calc_norm_int(
  mfd = mf_data_demo,
  normalization = "sum_ubiq"
)
```

`calc_number_assignment` *Calculate Number of Molecular Formula Assignments per Peak*

Description

This function calculates the number of molecular formula (mf) assignments for each individual peak (peak_id) within a specified mass spectrum (ms_id). It counts the occurrences of molecular formulas assigned to each peak and returns a vector of counts corresponding to the number of assignments for each unique combination of mass spectrum ID, peak ID, and molecular formula.

Usage

```
calc_number_assignment(ms_id, peak_id, mf, ...)
```

Arguments

`ms_id`	A vector containing the mass spectrum ID for each peak.
`peak_id`	A vector containing the peak ID for each peak.
`mf`	Character vector of molecular formula(s) (e.g., `c("C10H23NO4", "C10H24N4O2S")`).
`...`	Additional arguments passed to methods.

Value

A vector of integer counts representing the number of molecular formula assignments for each unique combination of mass spectrum ID, peak ID, and molecular formula.

See Also

Other calculations: `calc_data_summary()`, `calc_dbe()`, `calc_eval_params()`, `calc_exact_mass()`, `calc_ideg()`, `calc_ma()`, `calc_neutral_mass()`, `calc_nm()`, `calc_norm_int()`, `calc_number_occurrence()`, `calc_recalibrate_ms()`

Examples

```
ms_ids <- c("file1", "file1", "file2", "file2", "file3")
peak_ids <- c(1, 2, 2, 3, 4)
mfs <- c("C10H10N2O8", "C10H12N2O8", "C10H10N2O8", "C10H11NOS4", "C10H24N4O2S")
n_assignments <- calc_number_assignment(ms_id = ms_ids, peak_id = peak_ids, mf = mfs)
print(n_assignments)

mf_data_demo[, calc_number_assignment(file_id, peak_id, mf)]
```

`calc_pielou_evenness` *Calculate Pielou's Evenness*

Description

This function calculates Pielou's evenness index, a measure of the distribution of abundances across molecular formulas. Evenness ranges from 0 (one molecular formula dominates) to 1 (all formulas are equally abundant).

Evenness is derived using the Shannon index:

$$E = \frac{H}{\log(S)}$$

where:

- H is the Shannon diversity index.
- S is the number of unique molecular formulas.

If there is only one molecular formula, evenness is defined as 1.

Usage

```
calc_pielou_evenness(mf, magnitude)
```

Arguments

`mf`	Character vector. A list of unique molecular formulas.
`magnitude`	Numeric vector. A list of respective intensities (abundances) for each molecular formula. Must be non-negative and have the same length as `mf`.

Value

A single numeric value representing Pielou's evenness.

Examples

```
calc_pielou_evenness(
  mf = c("C10H20O5", "C12H18O3", "C18H30O6"),
  magnitude = c(1982375, 2424, 312410)
)
```

`calc_shannon_index` *Calculate the Shannon Diversity Index*

Description

The Shannon diversity index is calculated to quantify the diversity of molecular formulas based on their relative abundances. This index considers both the richness (number of unique formulas) and the evenness (distribution of abundances). Higher values indicate greater diversity.

The Shannon index is defined as:

$$H = -\sum(p_i \cdot \ln(p_i))$$

where:

- p_i is the relative abundance of the i-th molecular formula.

Zero-abundance formulas are excluded from the calculation.

Usage

```
calc_shannon_index(mf, magnitude)
```

Arguments

`mf`	Character vector. A list of unique molecular formulas.
`magnitude`	Numeric vector. A list of respective abundances (intensities) for each molecular formula. Must be non-negative and have the same length as `mf`.

Value

A single numeric value representing the Shannon diversity index. Returns 0 if `magnitude` is all zeros.

Examples

```
calc_shannon_index(
  mf = c("C10H20O5", "C12H18O3", "C18H30O6"),
  magnitude = c(1982375, 2424, 312410)
)
```

`calc_simpson_index` *Calculate the Simpson Diversity Index*

Description

The Simpson diversity index is calculated to measure the probability that two randomly selected individuals (e.g., molecular formulas) belong to the same category. It quantifies the dominance or evenness within a dataset.

The Simpson index is defined as:

$$D = \sum(p_i^2)$$

where:

- p_i is the relative abundance of the i-th molecular formula.

The index ranges between 0 and 1:

- A value near 0 indicates high diversity (even distribution of abundances).
- A value of 1 indicates no diversity (one molecular formula dominates).

Usage

```
calc_simpson_index(mf, magnitude)
```

Arguments

`mf`	Character vector. A list of unique molecular formulas.
`magnitude`	Numeric vector. A list of respective abundances (intensities) for each molecular formula. Must be non-negative and have the same length as `mf`.

Value

A single numeric value representing the Simpson diversity index. Returns 0 if `magnitude` is all zeros.

Examples

```
calc_simpson_index(
  mf = c("C10H20O5", "C12H18O3", "C18H30O6"),
  magnitude = c(1982375, 2424, 312410)
)
```

`calc_time_flattened_mfd` *Create a time-flattened molecular formula data table*

Description

Create a time-flattened molecular formula data table

Usage

```
create_time_flattened_mfd(
  mfd,
  ret_time_col = "ret_time_min",
  check_chromatographic = TRUE,
 group_cols = c("mf", "sample_id", "link_rawdata", "bl", "CRAM", "IOS", "amino_acids",
   "ideg_neg", "ideg_pos", "iterr2_neg", "iterr2_pos", "iterr_neg", "iterr_pos",
  "marine_dom", "nucleotide", "peptide", "surfactant", "vkey", "c", "h", "n", "o", "p",
  "s", "m_cal", "nm", "dbe", "oc", "hc", "nc", "sc", "dbe_o", "ai", "wf", "z", "nosc",
    "delg0_cox", "snp_check", "nsp_type", "co_tot", "nsp_tot"),
  sum_cols = c("i_magnitude"),
  mean_cols = c("mz", "res", "m", "ppm", "dev_n_c", "s_n", "kmd", "ppm_filt"),
  check_discrete = TRUE,
  max_unique_frac = 0.5
)
```

Arguments

`mfd`	A `data.table` containing molecular formula assignment results.
`ret_time_col`	Character string. Name of the retention time column. Defaults to `"ret_time_min"`.
`check_chromatographic`	Logical. If `TRUE`, check whether `mfd` contains a chromatographic retention time dimension. Defaults to `TRUE`.
`group_cols`	Character vector with columns used for grouping. Only columns present in `mfd` are used.
`sum_cols`	Character vector with columns to sum.
`mean_cols`	Character vector with columns to average.
`check_discrete`	Logical. If `TRUE`, grouping columns are checked for high numbers of unique values. Defaults to `TRUE`.
`max_unique_frac`	Numeric between `0` and `1`. If a grouping column contains more unique values than this fraction of all rows, a warning is issued. Defaults to `0.5`.

Details

calc_time_flattened_mfd() collapses chromatographic molecular formula data across the retention time dimension. This creates a pseudo-direct-infusion-like molecular formula data table from LC-MS or other chromatographic MS data.

The function checks whether the input contains chromatographic data using ret_time_col, which defaults to "ret_time_min". If this column is missing or contains only one distinct non-missing value, the function stops unless check_chromatographic = FALSE.

Grouping columns should contain discrete values. Default grouping columns are only used if they are present in mfd. Numeric measurement columns are then aggregated within each group. By default, i_magnitude is summed, while mass accuracy and signal-related columns are averaged.

Value

A data.table with retention-time-flattened molecular formula data.

---

check_neutral_mf *Check neutral molecular formulas*

Description

Checks whether character strings are valid neutral molecular formulas that can be parsed by convert_molecular_formula_t

The function is intended as a lightweight pre-check before converting molecular formulas into element-count tables. It identifies common non-formula entries such as InChIKeys, charged formulas, empty values, unsupported isotope notation, and formulas containing unknown element or isotope labels.

Usage

```
check_neutral_mf(mf, masses = ume::masses)
```

Arguments

mf	A character vector of molecular formulas.
masses	A data.table containing valid element and isotope definitions. By default, ume::masses is used. The table must contain at least the columns symbol and label.

Details

Check molecular formulas for neutral formula validity

This function validates syntax only. It does not check chemical plausibility, valence rules, isotope natural abundance, charge balance, or whether the molecular formula corresponds to a real compound.

The parser uses the valid element symbols and isotope labels provided in masses. This avoids hard-coding element symbols and ensures that the validation is consistent with convert_molecular_formula_to_data_table().

Supported isotope notation follows the convention used in ume, for example:

- [13C] for one carbon-13 atom
- [13C2] for two carbon-13 atoms
- [18O2] for two oxygen-18 atoms

The alternative notation [13C]2 is currently classified as unsupported because the isotope count is placed outside the brackets.

Charged formulas such as "C10H13N2+", "C11H18N2+2", or "C18H35CaO2Zn+3" are classified as charged and therefore not neutral.

InChIKeys such as "IOVCWXUNBOPUCH-UHFFFAOYSA-M" are detected separately and classified as non-formula identifiers.

Value

A data.table with one row per input entry and the following columns:

mf Original input string.

is_empty Logical; TRUE if the input is NA or empty.

is_inchikey Logical; TRUE if the input resembles an InChIKey.

has_charge Logical; TRUE if the formula ends with charge notation such as +, -, +2, -3, 2+, or 3-.

is_parseable Logical; TRUE if the string can be fully tokenized using valid element and isotope labels from masses.

is_neutral_mf Logical; TRUE if the input is non-empty, does not resemble an InChIKey, has no terminal charge notation, and can be fully parsed as a molecular formula.

issue Character label describing the detected issue. Valid neutral formulas are labelled "valid_neutral_mf".

See Also

Other molecular formula functions: convert_data_table_to_molecular_formulas(), convert_molecular_formula_to

Examples

```
mf <- c(
  "C6H6",
  "C6[13C2]HF15O2",
  "C6[13C]2HF15O2",
  "C4H5FeO4+",
  "C11H18N2+2",
  "IOVCWXUNBOPUCH-UHFFFAOYSA-M",
  NA_character_
)

check_neutral_mf(mf)

valid_mf <- check_neutral_mf(mf)[is_neutral_mf == TRUE, mf]
```

`classify_files` *Classify FTMS files into categories based on filename patterns*

Description

Classifies entries into categories (blank, standard, pool, sample, ...) based on pattern rules applied to a specific search column. The identifiers returned in each category are also configurable.

Usage

```
classify_files(
  fi,
  search_col = "link_rawdata",
  id_col = "file_id",
 patterns = list(blank = c("blk", "blank", "mq"), standard = c("srfa", "standard"), pool
    = c("pool")),
  include_blank_check = TRUE,
  return = c("list", "table")
)
```

Arguments

`fi`	`data.table`. Must contain the columns specified in `search_col` and `id_col`.
`search_col`	Character. Name of the column used for pattern matching. Defaults to `"link_rawdata"`.
`id_col`	Character. Name of the column whose values are returned for each category. Defaults to `"file_id"`.
`patterns`	Named list of character vectors. Each list entry is a category name, and its value is a vector of patterns.
`include_blank_check`	Logical; if TRUE and `blank_check` exists, it is used to assign `"blank"`.
`return`	Either `"list"` (default) or `"table"`.

- `"list"` → named list of ID vectors
- `"table"` → `fi` with added column `category_analysis`

Details

Default behavior:

- `"blank"`: `blank_check == "blank"` or pattern `"blk"`
- `"standard"`: pattern `"srfa"`
- `"pool"`: pattern `"pool"`
- `"sample"`: everything unmatched

Pattern matching is case-insensitive.

Value

Named list or a classified `data.table`.

Examples

```
# Minimal demo data
fi <- data.table::data.table(
  file_id       = 1:6,
  filename      = c("NS_blk_01.raw", "SRFA_20.raw", "Pool_A.raw",
                    "Sample_01.raw", "Sample_02.raw", "MQ_blank.raw"),
  blank_check   = c("blank", NA, NA, NA, NA, "blank"),  # optional column
  link_rawdata  = c("NS_blk_01.raw", "SRFA_20.raw", "Pool_A.raw",
                    "Sample_01.raw", "Sample_02.raw", "MQ_blank.raw")
)

# 1) Default behavior: return named list of file_ids by category
classify_files(fi)

# 2) Use a different column for pattern matching
classify_files(fi, search_col = "filename")

# 3) Return another ID field (here: file_id → stays the same for demo)
classify_files(fi, id_col = "file_id")

# 4) Return the full table with new category column
classify_files(fi, return = "table")
```

---

color.palette *Create a Custom Interpolated Color Palette*

Description

Constructs a continuous color palette from a sequence of base colors. Intermediate colors are interpolated between each pair of adjacent colors, optionally using a custom number of interpolation steps.

Usage

```
color.palette(steps, n.steps.between = NULL, ...)
```

Arguments

`steps`	A character vector of base colors (e.g., hex codes or color names). These colors define the breakpoints in the palette.
`n.steps.between`	An optional integer vector specifying how many interpolated colors should be added **between** each pair of entries in `steps`. Must have length `length(steps) - 1`. If `NULL` (default), no intermediate colors are added beyond the endpoints.
`...`	Additional arguments passed to methods.

Details

This helper is primarily used for UME visualizations (e.g., color bars in density plots), but it can be used independently for any plotting task.

Value

A function of class `"colorRampPalette"` that generates interpolated color vectors when called with a single integer argument `n`.

For example, `pal <- color.palette(c("blue", "white", "red")); pal(100)` returns a vector of 100 smoothly interpolated colors.

Examples

```
# Generate a simple blue-white-red palette
pal <- color.palette(c("blue", "white", "red"))
pal(10)

# Add additional steps between colors
pal2 <- color.palette(c("blue", "white", "red"), n.steps.between = c(5, 10))
pal2(20)
```

---

convert_data_table_to_molecular_formulas

Convert Data Table with Element Counts to Molecular Formulas

Description

Creates standardized molecular formula strings from isotope or element count columns and adds them to the input `data.table`.

Usage

```
convert_data_table_to_molecular_formulas(
  mfd,
  isotope_formulas = FALSE,
  keep_element_sums = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

`mfd`	data.table with molecular formula data as derived from `ume::assign_formulas`. Column names of elements/isotopes must match names in the `isotope` column of `ume::masses`; values are integers representing counts per formula.

`isotope_formulas`
: Logical. If `TRUE`, an additional isotope-specific molecular formula string `mf_iso` is created.

`keep_element_sums`
: Logical. If `TRUE`, additional columns with total atom counts per element are returned, for example `C_tot`.

`verbose`
: Logical. If `TRUE`, progress messages are printed.

`...`
: Additional arguments passed to `get_isotope_info()`.

Details

The function extracts element or isotope counts from a table with one column per isotope or element. Valid isotope columns are detected using `get_isotope_info()` and the reference table `ume::masses`.

The standard molecular formula `mf` is created by summing isotopes belonging to the same element and arranging elements according to Hill order.

If `isotope_formulas = TRUE`, an additional `mf_iso` column is created that keeps isotope-specific information, for example `[12C5][13C][1H12][16O6]`.

The function preserves the original row order and keeps duplicate rows.

Value

The original table `mfd` as a `data.table` with additional columns:

mf Standardized molecular formula following Hill order.

mf_iso If `isotope_formulas = TRUE`, isotope-specific molecular formula.

C_tot If `keep_element_sums = TRUE`, total count of all carbon isotopes. Equivalent `*_tot` columns are created for other elements.

Notes

- Isotopic columns such as `13C` are formatted as `[13C]` in `mf_iso`.
- The output follows Hill order: `C`, `H`, then all other elements alphabetically.
- Single-element counts, e.g. `C1H4`, are formatted without explicit `1`.

References

Hill E.A. (1900). On a system of indexing chemical literature; adopted by the classification division of the U. S. patent office. *Journal of the American Chemical Society*, **22**, 478-494. doi:10.1021/ja02046a005

See Also

Other molecular formula functions: `check_neutral_mf()`, `convert_molecular_formula_to_data_table()`

Examples

```
convert_data_table_to_molecular_formulas(
  mf_data_demo[, .(`12C`, `1H`, `14N`, `16O`, `31P`, `32S`)]
)
```

`convert_molecular_formula_to_data_table`
Convert Molecular Formulas to a Data Table of Element Counts

Description

Parses molecular formulas and returns a `data.table` where each row represents one molecular formula and each element or isotope is represented by a separate count column.

Usage

```
convert_molecular_formula_to_data_table(
  mf,
  masses = ume::masses,
  table_format = c("wide", "long"),
  keep_mf_old = TRUE,
  isotope_default = c("most_abundant", "lightest"),
  check_neutral = FALSE
)
```

Arguments

`mf`	Character vector of molecular formula(s) (e.g., `c("C10H23NO4", "C10H24N4O2S")`).
`masses`	A data.table. Defaults to `ume::masses` (based on NIST data) containing isotope information for elements, including nominal and exact mass, relative abundance, and Hill system order.
`table_format`	A string controlling the output format. Either `"wide"` (default) or `"long"`.
`keep_mf_old`	Logical. If `TRUE` (default), the original input formula is returned in a column named `mf_old`.
`isotope_default`	A string defining which isotope should be used when an element is given without explicit isotope notation. Either `"most_abundant"` (default) or `"lightest"`.
`check_neutral`	Logical. If `TRUE` (default = `FALSE`), input formulas are checked with `check_neutral_mf()` before parsing.

Details

The function supports normal element notation such as `C6H12O6` and bracketed isotope notation such as `[13C]`, `[13C2]`, and `[18O2]`.

Input formulas are parsed using the element symbols and isotope labels provided in `masses`. This avoids hard-coded element lists and allows rare elements to be parsed as long as they are present in `masses`.

By default, input formulas are checked with `check_neutral_mf()` before parsing.

The standardized molecular formula `mf` is generated using dynamic Hill ordering:

- if carbon is present: `C`, then `H`, then all other elements alphabetically
- if carbon is absent: all elements alphabetically, including `H`

Value

A `data.table` in wide or long format.

See Also

Other molecular formula functions: `check_neutral_mf()`, `convert_data_table_to_molecular_formulas()`

---

create_isotope_expanded_table

Create an Expanded Table of Parent and Isotope Daughter Formulas

---

Description

Creates a new molecular formula table containing the original parent formulas and their corresponding single-isotope daughter formulas.

Usage

```
create_isotope_expanded_table(
  mfd,
  id_col = "peak_id",
  allow_duplicates = TRUE,
  elements = NULL
)
```

Arguments

`mfd`	A `data.table` containing molecular formula information in wide format, including isotope count columns, or a character vector of molecular formulas. Character input is first converted with `convert_molecular_formula_to_data_table()`.
`id_col`	Name of the column in `mfd` used to define isotope groups. Default is `"peak_id"`.

`allow_duplicates`
: Logical. If `TRUE` (default), isotope daughter formulas are created for each input row using `id_col` as group identifier. If `FALSE`, the result is based on unique isotope compositions only.

`elements`
: Optional character vector of element symbols (matching `masses$symbol`) to restrict isotope expansion. If `NULL` (default), all eligible elements detected in `mfd` are used.

Details

The output includes annotation columns that facilitate isotope validation in downstream workflows:

- `iso_role` indicates whether a row represents a `"parent"` or `"daughter"` isotopologue.
- `iso_element` stores the element symbol for which the isotope substitution was generated (e.g. `"C"`, `"N"`, `"S"`).
- `iso_from` and `iso_to` store the parent and daughter isotope labels (e.g. `"12C"` and `"13C"`).

Value

A `data.table` containing parent and daughter formulas, including isotope annotation columns for downstream validation.

See Also

Other isotopes: `calc_isotope_pattern()`, `eval_isotopes()`, `uplot_isotope_precision()`

---

create_ume_formula_library

Create a molecular formula library for UME

Description

Generates all combinations of element / isotope counts between `min_formula` and `max_formula`, filtered by mass, DBE, element ratios, and heuristic rules (Kind & Fiehn 2007).

Usage

```
create_ume_formula_library(
  max_formula,
  min_formula = "C1H1",
  lib_version = 99,
  masses = ume::masses,
  max_mass = 152,
  ratio_filter = TRUE,
  heu_filter = TRUE,
  max_oc = 1.2,
  max_hc = 3.1,
```

```
  max_nc = 1.3,
  max_pc = 0.3,
  max_sc = 0.8,
  verbose = FALSE
)
```

Arguments

max_formula	Character. Maximum element/isotope counts, e.g. "C20H40O10" or "C1000\[13C1\]H2000".
min_formula	Character. Minimum element/isotope counts (default "C1H1").
lib_version	Integer. Library version identifier (default 99).
masses	A data.table. Defaults to ume::masses (based on NIST data) containing isotope information for elements, including nominal and exact mass, relative abundance, and Hill system order.
max_mass	Numeric. Maximum allowed exact mass.
ratio_filter	Logical. Apply O/C, H/C, N/C, P/C, S/C filters.
heu_filter	Logical. Apply Kind - Fiehn heuristic rules.
max_oc	Maximum oxygen / carbon ratio in a molecule; (UM_orig: 1.5; 7 rules: 1.2)
max_hc	Maximum hydrogen / carbon ratio in a molecule; (UM_orig: ; 7 rules: 1.2)
max_nc	Maximum nitrogen / carbon ratio in a molecule; (UM_orig: 0.5; 7 rules: 1.3)
max_pc	Maximum phosphorus / carbon ratio in a molecule; (UM_orig: 3; 7 rules: 0.3)
max_sc	Maximum sulfur / carbon ratio in a molecule; (UM_orig: 4; 7 rules: 0.8)
verbose	Logical. Print progress messages.

Value

A data.table containing the generated molecular formula library. The returned object has class "ume_library" and includes one row per molecular formula, with columns for:

- elemental and isotopic counts (e.g., 12C, 13C, 1H, 16O, ...)
- double bond equivalent (dbe)
- exact mass (mass)
- molecular formula string (mf)
- a unique versioned key (vkey)

Additional metadata is stored as attributes:

- "lib_version": numeric version identifier
- "min_formula": user-supplied minimum formula
- "max_formula": user-supplied maximum formula
- "max_mass": maximum allowed exact mass
- "filters": list describing applied ratio and heuristic filters
- "call": the matched function call

The object inherits from both "ume_library" and "data.table".

References

Kind T., Fiehn O. (2007). Seven Golden Rules for heuristic filtering of molecular formulas obtained by accurate mass spectrometry. *BMC Bioinformatics*, **8**, 105. doi:10.1186/147121058105

Hill E.A. (1900). On a system of indexing chemical literature; adopted by the classification division of the U. S. patent office. *Journal of the American Chemical Society*, **22**, 478-494. doi:10.1021/ja02046a005

`download_library` *Download and Load a UME Formula Library*

Description

Downloads and loads a UME formula library from Zenodo.

The function is designed to be CRAN-compliant:

- It does not write to the user's file system unless `dest` is explicitly provided.
- If `dest` is provided and the file already exists, it is loaded directly and not downloaded again.
- If `dest = NULL`, the library is downloaded to a temporary file and loaded into memory only.
- Downloads are only performed in interactive sessions.
- Existing local files can also be loaded in non-interactive sessions.

Usage

```
download_library(
  library = "lib_05.rds",
  doi = "10.5281/zenodo.17606457",
  dest = NULL,
  overwrite = FALSE,
  check_hash = TRUE,
  stop_on_hash_mismatch = FALSE
)
```

Arguments

`library`	Character. Name of the library file. Currently `"lib_02.rds"` or `"lib_05.rds"`.
`doi`	Character. Zenodo DOI.
`dest`	Character or `NULL`. Optional local file path. If provided, the library is saved there after explicit confirmation. If the file already exists, it is loaded directly.
`overwrite`	Logical. If `TRUE`, an existing file at `dest` can be replaced after confirmation.
`check_hash`	Logical. Check SHA-256 hash for known libraries?
`stop_on_hash_mismatch`	Logical. If `TRUE`, a hash mismatch causes an error. If `FALSE`, only a warning is issued.

Value

A `data.table` containing the formula library, or invisible `NULL` if no download is performed.

`eval_isotopes` *Evaluate isotope information*

Description

Add isotope information to the parent mass and optionally remove isotopoloques from mfd table. Required for further data evaluation that considers isotope information.

Usage

```
eval_isotopes(mfd, remove_isotopes = TRUE, verbose = FALSE, ...)
```

Arguments

`mfd`	data.table with molecular formula data as derived from `ume::assign_formulas`. Column names of elements/isotopes must match names in the `isotope` column of `ume::masses`; values are integers representing counts per formula.
`remove_isotopes`	If set to TRUE (default), all entries for isotopologues are removed from mfd. The main isotope information for each parent ion is still maintained in the "intxy"-columns.
`verbose`	logical; if `TRUE`, show progress messages.
`...`	Additional arguments passed to methods.

Value

A data.table with additional columns such as "int_13c" containing stable isotope abundance information.

Author(s)

Boris P. Koch

See Also

Other Formula assignment: `add_known_mf()`, `calc_eval_params()`, `check_formula_library()`, `ume_assign_formulas()`

Other isotopes: `calc_isotope_pattern()`, `create_isotope_expanded_table()`, `uplot_isotope_precision()`

Examples

```
eval_isotopes(mfd = mf_data_demo)
```

`filter_int` *Filter by (relative) peak magnitude*

Description

This function filters molecular formulas by (relative) peak abundances.

Usage

```
filter_int(mfd, norm_int_min = NULL, norm_int_max = NULL, verbose = FALSE, ...)
```

Arguments

`mfd`	data.table with molecular formula data as derived from `ume::assign_formulas`. Column names of elements/isotopes must match names in the `isotope` column of `ume::masses`; values are integers representing counts per formula.
`norm_int_min`	Lower threshold (>=) of (normalized) peak magnitude
`norm_int_max`	Upper threshold (<=) of (normalized) peak magnitude
`verbose`	logical; if `TRUE`, show progress messages.
`...`	Arguments passed on to `calc_norm_int` `ms_id` Character; name of the column identifying individual spectra (default: `"file_id"`). `peak_id` Character; name of the column identifying unique peaks (default: `"peak_id"`). `peak_magnitude` Character; name of the column containing peak intensity values (default: `"i_magnitude"`). `normalization` Character; normalization method to apply. One of `"bp"`, `"sum"`, `"sum_ubiq"`, `"sum_rank"`, `"none"`. Default is `"bp"`. `n_rank` Integer; number of top-ranked peaks to use for `"sum_rank"` normalization (default: 200).

Value

data.table; subset of original molecular formula table

See Also

Other Formula subsetting: `filter_mass_accuracy()`, `filter_mf_data()`, `remove_blanks()`, `subset_known_mf()`, `ume_assign_formulas()`, `ume_filter_formulas()`

Examples

```
filter_int(mfd = calc_norm_int(mfd = mf_data_demo,
normalization = "sum_rank", n_rank = 100), norm_int_min = 1)
```

`filter_mass_accuracy` *Automated filter for mass accuracy*

Description

This function automatically sets a filter for mass accuracy for each individual spectrum.

Usage

```
filter_mass_accuracy(
  mfd,
  ma_col = "ppm",
  file_col = "file_id",
  msg = FALSE,
  ...
)
```

Arguments

`mfd`	data.table with molecular formula data as derived from `ume::assign_formulas`. Column names of elements/isotopes must match names in the `isotope` column of `ume::masses`; values are integers representing counts per formula.
`ma_col`	Name of the column that contains mass accuracy values in ppm (string)
`file_col`	Name of the column that contains file name
`msg`	logical. Deprecated synonym for `verbose`.
`...`	Additional arguments passed to methods.

Value

data.table; subset of original molecular formula table

See Also

Other Formula subsetting: `filter_int()`, `filter_mf_data()`, `remove_blanks()`, `subset_known_mf()`, `ume_assign_formulas()`, `ume_filter_formulas()`

`filter_mf_data` *Filter molecular formula data by mass spectrometric metadata*

Description

This function filters molecular formulas by isotope numbers, element ratios, etc.

Usage

```
filter_mf_data(
  mfd,
  c_iso_check = FALSE,
  n_iso_check = FALSE,
  s_iso_check = FALSE,
  ma_dev = 3,
  dbe_max = 999,
  dbe_o_min = -999,
  dbe_o_max = 999,
  mz_min = 1,
  mz_max = 9999,
  n_min = 0,
  n_max = 999,
  s_min = 0,
  s_max = 999,
  p_min = 0,
  p_max = 999,
  oc_min = 0,
  oc_max = 999,
  hc_min = 0,
  hc_max = 999,
  nc_min = 0,
  nc_max = 99,
  verbose = FALSE,
  ...
)
```

Arguments

`mfd`	data.table with molecular formula data as derived from `ume::assign_formulas`. Column names of elements/isotopes must match names in the `isotope` column of `ume::masses`; values are integers representing counts per formula.
`c_iso_check`	(TRUE / FALSE); check if formulas are verified by the presence of the main daughter isotope
`n_iso_check`	(TRUE / FALSE); check if formulas are verified by the presence of the main daughter isotope

s_iso_check	(TRUE / FALSE); check if formulas are verified by the presence of the main daughter isotope
ma_dev	Deviation range of mass accuracy in +/- ppm (default: 3 ppm)
dbe_max	Maximum number for DBE
dbe_o_min	Minimum number for DBE minus O atoms
dbe_o_max	Maximum number for DBE minus O atoms
mz_min	Minimum of mass to charge value
mz_max	Maximum of mass to charge value
n_min	Minimum number of nitrogen atoms
n_max	Maximum number of nitrogen atoms
s_min	Minimum number of nitrogen atoms
s_max	Maximum number of nitrogen atoms
p_min	Minimum number of nitrogen atoms
p_max	Maximum number of nitrogen atoms
oc_min	Minimum atomic ratio of oxygen / carbon
oc_max	Maximum atomic ratio of oxygen / carbon
hc_min	Minimum atomic ratio of hydrogen / carbon
hc_max	Maximum atomic ratio of hydrogen / carbon
nc_min	Minimum atomic ratio of nitrogen / carbon
nc_max	Maximum atomic ratio of nitrogen / carbon
verbose	logical; if TRUE, show progress messages.
...	Additional arguments passed to methods.

Value

data.table; subset of original molecular formula table

Author(s)

Boris P. Koch

See Also

Other Formula subsetting: filter_int(), filter_mass_accuracy(), remove_blanks(), subset_known_mf(), ume_assign_formulas(), ume_filter_formulas()

Examples

```
filter_mf_data(mfd = mf_data_demo, dbe_o_max = 10)
```

`filter_multi_assignments` *Filter multiple molecular formula assignments by mass-error medians*

Description

Filters multiple molecular formula assignments for the same mass peak using subgroup-specific mass-error distributions.

The function is intended for UME molecular formula data (`mfd`) and applies the false-positive filtering concept of Gao et al. (2024). Multiple assignments are evaluated separately for each mass spectrum, defined by `file_col`. Within each spectrum, assignments belonging to the same peak are compared. The assignment whose subgroup median mass error in mDa is closest to zero is retained.

Usage

```
filter_multi_assignments(
  mfd,
  file_col = "file_id",
  peak_col = "peak_id",
  mf_col = "mf",
  formula_mass_col = "m",
  error_ppm_col = "ppm",
  kmd_col = "kmd",
  kmd_digits = 3,
  kmd_base_mass = 14.01565,
  group_cols = c("nsp_type", "kmd"),
  include_single_assignments = TRUE,
  median_type = 8,
  tie_method = c("first", "all"),
  return = c("filtered", "annotated")
)
```

Arguments

`mfd`	A `data.table` or `data.frame` containing molecular formula assignments.
`file_col`	Character. Column identifying an individual mass spectrum. Default is `"file_id"`.
`peak_col`	Character. Column identifying a mass peak. Default is `"peak_id"`.
`mf_col`	Character. Molecular formula column. Default is `"mf"`.
`formula_mass_col`	Character. Column containing the theoretical formula mass. Default is `"m"`.
`error_ppm_col`	Character. Column containing the relative mass error in ppm. Default is `"ppm"`.
`kmd_col`	Character. Name of the Kendrick mass defect column. If the column does not exist, it is created automatically from `formula_mass_col`. Default is `"kmd"`.
`kmd_digits`	Integer. Number of digits used when automatically calculating the Kendrick mass defect. Default is `3`.

kmd_base_mass	Numeric. Exact mass of the Kendrick base unit. The default is 14.01565, corresponding to CH2.
group_cols	Character vector. Columns used to define subgroups for median mass-error estimation. Default is c("nsp_type", "kmd"). file_col is added internally and should not be included manually.
include_single_assignments	Logical. If TRUE, formulas without multiple assignments are returned together with retained multiple assignments. If FALSE, only retained multiple assignments are returned. Default is TRUE.
median_type	Integer. Quantile algorithm passed to stats::quantile(). Default is 8.
tie_method	Character. How to handle ties when several assignments of the same peak have identical absolute subgroup median mass errors. One of "first" or "all". Default is "first".
return	Character. Either "filtered" or "annotated". If "filtered", only retained rows are returned. If "annotated", all rows are returned with additional UME filter columns.

Details

The default UME column setup assumes:

- file_id: unique mass spectrum identifier
- peak_id: unique peak identifier within a spectrum
- mf: molecular formula
- m: theoretical formula mass
- ppm: relative mass error in ppm
- nsp_type: formula class/group
- kmd: Kendrick mass defect

Mass error in mDa is calculated as:

$$Merr_{mDa} = m \times ppm/1000$$

Multiple assignments are identified by duplicated combinations of file_col and peak_col. Median mass errors are calculated by file_col and group_cols, by default file_id, nsp_type, and kmd.

Value

A data.table.

If return = "filtered", rejected multiple assignments are removed.

If return = "annotated", the returned table contains additional columns:

- ume_multi_assignment
- ume_merr_mda
- ume_median_merr_mda
- ume_abs_median_merr_mda
- ume_assignment_status

References

Gao, S. et al. (2024). Detection and Exclusion of False-Positive Molecular Formula Assignments via Mass Error Distributions in UHR Mass Spectra of Natural Organic Matter. Analytical Chemistry, 96, 10210-10218.

Examples

```
## Not run:
mfd_filtered <- filter_multi_assignments(mfd)

mfd_check <- filter_multi_assignments(
  mfd,
  return = "annotated"
)

mfd_only_multi <- filter_multi_assignments(
  mfd,
  include_single_assignments = FALSE
)

## End(Not run)
```

`get_isotope_info` *Retrieve NIST element and isotope data*

Description

Checks if element/isotope columns are present in `mfd` and lookup of NIST isotope information (based on masses). Can be applied to a formula library and any table having molecular formula data. If only an element name is identified, the symbol and data of the lightest isotope of the element will be returned. For example, the column name "C" will return "12C" isotope data.

Usage

```
get_isotope_info(mfd, masses = ume::masses, verbose = FALSE, ...)
```

Arguments

`mfd`	data.table with molecular formula data as derived from `ume::assign_formulas`. Column names of elements/isotopes must match names in the `isotope` column of `ume::masses`; values are integers representing counts per formula.
`masses`	A data.table. Defaults to `ume::masses` (based on NIST data) containing isotope information for elements, including nominal and exact mass, relative abundance, and Hill system order.
`verbose`	logical; if `TRUE`, show progress messages.
`...`	Additional arguments passed to methods.

Value

A data.table containing information on all isotopes identified in mfd and a column "orig_name" having the original names of the isotope / element columns in `mfd`. Results are ordered according to Hill system.

References

Hill E.A. (1900). On a system of indexing chemical literature; adopted by the classification division of the U. S. patent office. *Journal of the American Chemical Society*, **22**, 478-494. doi:10.1021/ja02046a005

Examples

```
get_isotope_info(mfd = mf_data_demo, verbose = TRUE)
```

`inchi_to_mf` *Extract molecular formula from InChI string*

Description

Extracts the molecular formula from an InChI string by parsing the first layer of the InChI representation. The function performs a fast string-based extraction without requiring external cheminformatics libraries.

Usage

```
inchi_to_mf(inchi)
```

Arguments

`inchi`	A character vector containing InChI strings (e.g., `"InChI=1S/C2H6O/c1-2-3/h3H,2H2,1H3"`).

Details

Extract molecular formula from InChI

The function extracts the molecular formula from the first layer of the InChI string (i.e., the part immediately following `"InChI=1S/"` or `"InChI=1/"` and before the next `/` separator).

This approach is highly efficient because the molecular formula is explicitly encoded in the InChI and does not require interpretation of molecular structure, in contrast to SMILES-based approaches.

Leading and trailing whitespace is ignored. Non-character inputs result in an error.

Value

A character vector of molecular formulas in Hill notation, with the same length and order as `inchi`. Invalid or missing inputs return `NA_character_`.

See Also

https://www.inchi-trust.org/technical-faq/

Examples

```
inchi <- c(
  "InChI=1S/C2H6O/c1-2-3/h3H,2H2,1H3",
  "InChI=1S/H2O/h1H2",
  NA_character_
)

inchi_to_mf(inchi)
# [1] "C2H6O" "H2O"   NA
```

---

is_ume_peaklist *Check whether an object is a UME peaklist*

Description

Check whether an object is a UME peaklist

Usage

```
is_ume_peaklist(x)
```

Arguments

x	Any object

Value

TRUE/FALSE

---

known_mf *Collection of known formulas, for which additional information is available.*

Description

Known formulas; contains formulas for which additional knowledge is available. This can be also calibration lists. Due to size reasons the table is restricted to what is covered by standard UME formula library (mz<=700, elements CHONSP considered). The original version is part of the UME database and transferred to UME using UTF-8 encoding. CRAM molecular formulas are taken from the supplementary material that is provided by Hertkorn et al. (2006).

Usage

```
known_mf
```

Format

A data.table with ~300,000 rows and 14 variables:

mf_kf_id Unique identifier for formulas in known_mf

mf molecular formula

category Grouping of molecular formulas into topics

info1 - 4 Additional information on the formula

reference Reference in which the formula was mentioned

mass Exact neutral mass of the formula

last_update Last update in the source database

Source

taken from www.awi.de

See Also

Other ume data: `lib_demo`, `masses`, `mf_data_demo`, `peaklist_demo`, `ume_name_dictionary`

Examples

```
data(known_mf)
```

---

lib_demo *Demo formula library (200 - 300 Da, neutral mass)*

Description

Contains a small molecular formula library for demonstration and validation purposes. Complete formula libraries are available in the 'ume.formulas' data package.

Usage

```
lib_demo
```

Format

A data.table having ~115,111 rows and 12 variables:

vkey First two digits represent the formula library version; last digits are unique identifiers for each formula

mf Neutral molecular formula (no differentiation of isotopes)

mass Calculated exact neutral mass of a formula (based on ume::masses)

See Also

Other ume data: known_mf, masses, mf_data_demo, peaklist_demo, ume_name_dictionary

Examples

```
data(peaklist_demo)
```

---

masses *Masses: Elements and isotopes*

Description

Contains masses, valences, isotopes and isotope ratios of elements based on data by NIST Physical Measurement Laboratory (https://www.nist.gov/pml).

Usage

```
masses
```

Format

A data.table having 288 rows and 23 variables:

label Element symbol including nominal isotopic mass

symbol Element symbol in upper case

nm Nominal mass of the isotope

exact_mass Exact mass of the isotope

mole_fraction Mole fraction compared to all isotopes of an element

relative_abundance Relative abundance compared to the main (most abundant) isotope

valence Valence at standard conditions

valence2 Alternative valence at standard conditions

hill_order Rank in Hill Order for molecular formulas (cf. https://en.wikipedia.org/wiki/Chemical_formula)

Source

https://www.nist.gov/pml/atomic-weights-and-isotopic-compositions-relative-atomic-masses

References

Hill E.A. (1900). On a system of indexing chemical literature; adopted by the classification division of the U. S. patent office. *Journal of the American Chemical Society*, **22**, 478-494. doi:10.1021/ja02046a005

See Also

Other ume data: known_mf, lib_demo, mf_data_demo, peaklist_demo, ume_name_dictionary

Examples

```
data(masses)
```

`mf_data_demo` *mf_data_demo*

Description

Contains molecular formula data and metainformation on formulas. The metainformation

Usage

```
mf_data_demo
```

Format

A data.table with ~9245 rows (formulas) and 65 variables:

file_id Unique ID (integer) for each analysis

file A unique label for a mass spectrum or sample (character)

peak_id Unique ID (integer) for each mass peak in the peak list 'pl'

mz Mass to charge ratio of the singly charged molecular ion (numeric)

i_magnitude Measured mass peak magnitude of the singly charged molecular ion (numeric)

norm_int Normalized intensity as calculated by calc_norm_int()

m Neutral measured mass of the molecular ion

m_cal Neutral calculated mass of the assigned formula

ppm Realtive mass accuracy of measured mass compared to m_cal (in ppm)

nm Nominal mass of the neutral molecule

mf molecular formula (no differentiation of isotopes)

dbe Double bond equivalent

`12C` Number of carbon atoms (12C)

`1H` Number of hydrogen atoms

hc hydrogen / carbon ratio in a molecular formula

oc oxygen / carbon ratio in a molecular formula

nc nitrogen / carbon ratio in a molecular formula

sc sulfur / carbon ratio in a molecular formula

ai Aromaticity index according to Koch and Dittmar (2008, 2016)

z z score according to Stenson et al. (2003)

kmd Kendrick mass defect (based on CH2-units) according to Kendrick (1963)

ppm_filt Calculated threshold value for relative mass accuracy (in ppm) that can be used for formular filtering

mf_id Identifier for each unique molecular formula identified in the unfiltered dataset

int13c Measured relative peak magnitude of the 13C1 isotope compared to the parent ion (0 if isotope was not existing)

int15n Measured relative peak magnitude of the 15N1 isotope compared to the parent ion (0 if isotope was not existing)

int34s Measured relative peak magnitude of the 34S1 isotope compared to the parent ion (0 if isotope was not existing)

s_n Signal to noise ratio of the molecular ion (numeric)

res Mass resolution of the peak / ion (numeric)

del Mass difference between the calculated and measured mass

dev_n_c Deviation of the 12C/13C isotope ratio represented in carbon numbers according to Koch et al. (2007)

dbe_o DBE minus O

nosc Nominal oxidation state of carbon according to LaRowe & Van Cappellen (2011)

delg0_cox Standard molal Gibbs energies of the oxidation half reactions of organic compounds according to LaRowe & Van Cappellen (2011)

relint13c_calc Calculated relative intensity of the 13C1-isotope compared to the parent ion

int13c_calc Calculated absolute intensity of the 13C1-isotope compared to the parent ion

relint34s_calc Calculated relative intensity of the 34S1-isotope compared to the parent ion

int34s_calc Calculated absolute intensity of the 34S1-isotope compared to the parent ion

snp_check Checks if a formula contains S, N, or P

nsp_type Lists the number of N, S, and P atoms in a formula

co_tot Total number of carbon and oxygen atoms in a molecular formula

nsp_tot Total number of nitrogen, sulfur, and phosphorus atoms in a molecular formula

n_occurrence_orig Number of occurrences of a molecular formula in the entire unfiltered set of formulas

n_assignments_orig Number of molecular formula assignments per molecular mass in the unfiltered set of formulas

int_bp Peak magnitude (intensity) of the base peak in a spectrum

n_assignments Number of molecular formula assignments per molecular mass after filter process

int_ref Reference intensity that was used for calculating the normalized intensity (norm_int)

int_bp Magnitude of the base peak in a mass spectrum

int_bp Total magnitude of the reference that was used for normalization (cf. calc_norm_int())

Source

taken from www.awi.de

See Also

Other ume data: `known_mf`, `lib_demo`, `masses`, `peaklist_demo`, `ume_name_dictionary`

Examples

```
data(mf_data_demo)
```

order_columns *Order columns*

Description

Take most prominent columns required for data evaluation first - followed by all other columns.

Usage

```
order_columns(mfd, col_order = NULL, ...)
```

Arguments

mfd	data.table with molecular formula data as derived from ume::assign_formulas. Column names of elements/isotopes must match names in the isotope column of ume::masses; values are integers representing counts per formula.
col_order	A list of column names that defines the order of columns of mfd. Default is: cols = c("sample_tag", "sample_id", "file", "file_id", "peak_id", "i_magnitude", "norm_int", "m", "m_cal", "ppm", "nm", "mf", "dbe", "c", "h", "n", "o", "p", "s", "hc", "oc", "nc", "sc", "ai", "z", "kmd") If "col_order" is NULL the default order is applied.
...	Additional arguments passed to methods.

Value

A data.table containing isotope data for those isotopes present in mfd.

See Also

Other tools: add_missing_element_columns()

Examples

```
order_columns(mfd = mf_data_demo)
```

peaklist_demo *Demo peak list*

Description

Contains parts of the peaklist (200 - 300 m/z) from mass spectra to use as demonstration and validation dataset. The sample mass spectra contain one blank, three replicates of North Sea water, and three Arctic fjord samples as triplicates.

Usage

```
peaklist_demo
```

Format

A data.table having 31,091 rows and 7 variables:

file_id A unique identifier for a mass spectrum (integer)

file A unique label for a mass spectrum or sample (character)

peak_id A unique identifier for a peak in the entire peak list (integer)

mz Mass to charge ratio of the singly charged molecular ion (numeric)

i_magnitude Peak magnitude of the molecular ion (numeric)

s_n Signal to noise ratio of the molecular ion (numeric)

res Mass resolution of the peak / ion (numeric)

Source

taken from www.awi.de

See Also

Other ume data: `known_mf`, `lib_demo`, `masses`, `mf_data_demo`, `ume_name_dictionary`

Examples

```
data(peaklist_demo)
```

---

remove_blanks *Remove molecular formulas detected in blanks*

Description

Remove all molecular formulas that were detected in one or more blank analyses (identified via `blank_file_ids`). Matching is always on `mf`. If a retention-time column is present (or provided using `ret_time_col`), removal is restricted to the corresponding LC segment.

Usage

```
remove_blanks(
  mfd,
  blank_file_ids = NULL,
  blank_prevalence = 0.5,
  ret_time_col = NULL,
  verbose = FALSE,
  ...
)
```

Arguments

mfd	data.table with molecular formula data as derived from ume::assign_formulas. Column names of elements/isotopes must match names in the isotope column of ume::masses; values are integers representing counts per formula.
blank_file_ids	Integer vector of file_id values that represent blank analyses.
blank_prevalence	Numeric between 0 and 1. Threshold for blank filtering: the proportion of blanks in which a molecular formula must occur before it is excluded from the sample data. For example, blank_prevalence = 0 (default) removes any formula detected in at least one blank, while blank_prevalence = 0.5 removes formulas detected in 50% or more of the blanks.
ret_time_col	Character scalar. Name of the retention-time column that contains the beginning of the retention time segment that corresponds to the mass spectrum. If NULL (default), the function will auto-detect the first column in c("ret_time_min","retention_time","rt", that exists in mfd. If none is found, blanks are removed ignoring retention time.
verbose	logical; if TRUE, show progress messages.
...	Additional arguments passed to methods.

Details

- Requires a unique integer file_id per analysis in mfd.
- Minimal required columns in mfd: mf, file_id.
- Optional column: a retention-time column (e.g. "ret_time_min").
- If a retention-time column is used, formulas present in blanks are only removed for rows whose mf **and** retention time match
- The input mfd is **not** modified by reference; a subset is returned.

Value

data.table; subset of the original molecular formula table (mfd) with blank formulas removed (globally or LC-segment-wise).

Backward compatibility

The argument LCMS is deprecated and no longer used. Retention-time-aware removal is now enabled automatically when a retention-time column is present or explicitly provided via ret_time_col.

Author(s)

Boris P. Koch

See Also

Other Formula subsetting: filter_int(), filter_mass_accuracy(), filter_mf_data(), subset_known_mf(), ume_assign_formulas(), ume_filter_formulas()

Examples

```
# Presence/absence removal, no retention time:
remove_blanks(mfd = mf_data_demo,
              remove_blank_list = "Blank",
              verbose = TRUE)
```

`remove_empty_columns` *Remove empty columns*

Description

Removes columns that contain only `NA` values from a `data.table`. Columns listed in `excl_cols` are retained even if they are empty.

Usage

```
remove_empty_columns(df, excl_cols = NULL, ...)
```

Arguments

`df`	A `data.table` from which empty columns should be removed.
`excl_cols`	Optional character vector of column names that must be preserved, even if all values in those columns are missing.
`...`	Additional arguments passed to methods.

Value

A `data.table` containing all original non-empty columns, plus any columns listed in `excl_cols`, regardless of whether they are empty. Columns that contain only `NA` values and are *not* explicitly preserved are removed from the output.

Examples

```
dt <- data.table::data.table(
  c = c(2, 2, 2),
  x = c(NA, NA, NA),
  y = c(NA, NA, NA)
)
remove_empty_columns(dt, excl_cols = "y")
```

remove_id_columns *Remove columns that contain ID's*

Description

This functions removes columns ID columns ('_id') and hierarchical search columns ('_lft', '_rgt') from a table. Only exceptions are "sample_id" and "bottle_id that are always kept in the output table.

Usage

```
remove_id_columns(df, ...)
```

Arguments

df	data.table that contains ID columns
...	Additional arguments passed to methods.

See Also

Other Clean data output: remove_unknown_columns()

remove_unknown_columns *Remove columns that only have one specific value*

Description

This function removes columns that exclusively contain the value defined in 'search_term' (such as " unknown" (default)).

Usage

```
remove_unknown_columns(df, excl_cols = NULL, search_term = " unknown", ...)
```

Arguments

df	data.table that contains empty columns
excl_cols	List of column names that should not be removed, even if all values contain search_term
search_term	String that uniquely occurs in one column
...	Additional arguments passed to methods.

See Also

Other Clean data output: remove_id_columns()

subset_known_mf *Subsetting known molecular formula categories*

Description

Subset all molecular formulas that are present in one or more categories of ume::known_mf. Based on presence / absence.

Usage

```
subset_known_mf(
  mfd,
  select_category = NULL,
  exclude_category = NULL,
  verbose = FALSE,
  ...
)
```

Arguments

mfd	data.table with molecular formula data as derived from ume::assign_formulas. Column names of elements/isotopes must match names in the isotope column of ume::masses; values are integers representing counts per formula.
select_category	List of category names that should be selected
exclude_category	List of category names that should be ignored
verbose	logical; if TRUE, show progress messages.
...	Additional arguments passed to methods.

Value

data.table; subset of original molecular formula data.table (mfd)

See Also

Other Formula subsetting: filter_int(), filter_mass_accuracy(), filter_mf_data(), remove_blanks(), ume_assign_formulas(), ume_filter_formulas()

Examples

```
subset_known_mf(category_list = c("marine_dom"), mfd = mf_data_demo, verbose = TRUE)
```

theme_uplots *theme_uplots*

Description

Applies a clean UME-style theme used across all uplot_* visualisations.

Usage

```
theme_uplots(base_size = 12, base_family = "")
```

Arguments

base_size	Numeric base font size.
base_family	Font family.

Details

Unified UME Theme for All uplot_* Functions

Value

A ggplot2 theme object.

ume_assign_formulas *Complete formula assignment (wrapper function)*

Description

Assigns molecular formulas to neutral molecular masses and calculates all parameters required for data evaluation, such as a posteriori filtering of molecular formulas, plotting, and statistics. The function uses a pre-build molecular formula library.

Usage

```
ume_assign_formulas(pl, formula_library, verbose = FALSE, ...)
```

Arguments

pl	data.table containing peak data. Mandatory columns include neutral molecular mass (mass), peak magnitude (i_magnitude), and a peak identifier (peak_id).

`formula_library`
: Molecular formula library: a predefined data.table used for assigning molecular formulas to a peak list and for mass calibration. The library requires a fixed format, including mass values for matching. Predefined libraries are available in the R package *ume.formulas* and further described in Leefmann et al. (2019). A standard library for marine dissolved organic matter is `ume.formulas::lib_02`. New libraries can be built using `ume::create_ume_formula_library()`.

`verbose`
: logical; if `TRUE`, show progress messages.

`...`
: Arguments passed on to `calc_ma_abs`, `calc_neutral_mass`, `assign_formulas`, `eval_isotopes`, `calc_eval_params`, `add_known_mf`, `calc_norm_int`

  `m` Measured mass

  `ma_dev` Mass accuracy in +/- parts per million (ppm)

  `pol` Character: `"neg"`, `"pos"`, or `"neutral"`.

  `mz` String. Name of the column that contains mass-to-charge information (default: `"mz"`).

  `remove_isotopes` If set to TRUE (default), all entries for isotopologues are removed from mfd. The main isotope information for each parent ion is still maintained in the "intxy"-columns.

  `mfd` data.table with molecular formula data as derived from `ume::assign_formulas`. Column names of elements/isotopes must match names in the `isotope` column of `ume::masses`; values are integers representing counts per formula.

  `mf_col` Character. Name of the molecular formula column if `mfd` is a table. Default is `"mf"`.

  `known_mf` A `data.table` containing known molecular formulas and their assigned categories. Default is `ume::known_mf`.

  `category_dictionary` A `data.table` containing molecular formula category definitions. Default is `ume::ume_name_dictionary`.

  `wide` Logical.
  If `FALSE` (default), return a single comma-separated `categories` column.
  If `TRUE`, create one column per category (`CRAM`, `surfactant`, ...).

  `ms_id` Character; name of the column identifying individual spectra (default: `"file_id"`).

  `peak_id` Character; name of the column identifying unique peaks (default: `"peak_id"`).

  `peak_magnitude` Character; name of the column containing peak intensity values (default: `"i_magnitude"`).

  `normalization` Character; normalization method to apply. One of `"bp"`, `"sum"`, `"sum_ubiq"`, `"sum_rank"`, `"none"`. Default is `"bp"`.

  `n_rank` Integer; number of top-ranked peaks to use for `"sum_rank"` normalization (default: 200).

Details

All function arguments: args(filter_mf_data) args(filter_int)

Value

A data.table having molecular formula assignments for each mass.

See Also

Other Formula assignment: add_known_mf(), calc_eval_params(), check_formula_library(), eval_isotopes()

Other Formula subsetting: filter_int(), filter_mass_accuracy(), filter_mf_data(), remove_blanks(), subset_known_mf(), ume_filter_formulas()

Other ume wrapper: ume_filter_formulas()

Examples

```
ume_assign_formulas(pl = peaklist_demo, formula_library = lib_demo, pol = "neg", ma_dev = 0.2)
```

---

ume_filter_formulas *Complete Formula subsetting / filtering (wrapper)*

Description

A wrapper function to filter molecular formulas according to a evaluation parameters.

Usage

```
ume_filter_formulas(mfd, verbose = FALSE, ...)
```

Arguments

mfd	data.table with molecular formula data as derived from ume::assign_formulas. Column names of elements/isotopes must match names in the isotope column of ume::masses; values are integers representing counts per formula.
verbose	logical; if TRUE, show progress messages.
...	Arguments passed on to filter_mf_data, subset_known_mf, calc_norm_int, filter_int, remove_blanks c_iso_check (TRUE / FALSE); check if formulas are verified by the presence of the main daughter isotope n_iso_check (TRUE / FALSE); check if formulas are verified by the presence of the main daughter isotope s_iso_check (TRUE / FALSE); check if formulas are verified by the presence of the main daughter isotope ma_dev Deviation range of mass accuracy in +/- ppm (default: 3 ppm) dbe_max Maximum number for DBE dbe_o_min Minimum number for DBE minus O atoms dbe_o_max Maximum number for DBE minus O atoms mz_min Minimum of mass to charge value

`mz_max` Maximum of mass to charge value

`n_min` Minimum number of nitrogen atoms

`n_max` Maximum number of nitrogen atoms

`s_min` Minimum number of nitrogen atoms

`s_max` Maximum number of nitrogen atoms

`p_min` Minimum number of nitrogen atoms

`p_max` Maximum number of nitrogen atoms

`oc_min` Minimum atomic ratio of oxygen / carbon

`oc_max` Maximum atomic ratio of oxygen / carbon

`hc_min` Minimum atomic ratio of hydrogen / carbon

`hc_max` Maximum atomic ratio of hydrogen / carbon

`nc_min` Minimum atomic ratio of nitrogen / carbon

`nc_max` Maximum atomic ratio of nitrogen / carbon

`select_category` List of category names that should be selected

`exclude_category` List of category names that should be ignored

`ms_id` Character; name of the column identifying individual spectra (default: `"file_id"`).

`peak_id` Character; name of the column identifying unique peaks (default: `"peak_id"`).

`peak_magnitude` Character; name of the column containing peak intensity values (default: `"i_magnitude"`).

`normalization` Character; normalization method to apply. One of `"bp"`, `"sum"`, `"sum_ubiq"`, `"sum_rank"`, `"none"`. Default is `"bp"`.

`n_rank` Integer; number of top-ranked peaks to use for `"sum_rank"` normalization (default: 200).

`norm_int_min` Lower threshold (>=) of (normalized) peak magnitude

`norm_int_max` Upper threshold (<=) of (normalized) peak magnitude

`blank_file_ids` Integer vector of `file_id` values that represent blank analyses.

`blank_prevalence` Numeric between 0 and 1. Threshold for blank filtering: the proportion of blanks in which a molecular formula must occur before it is excluded from the sample data. For example, `blank_prevalence = 0` (default) removes any formula detected in at least one blank, while `blank_prevalence = 0.5` removes formulas detected in 50% or more of the blanks.

`ret_time_col` Character scalar. Name of the retention-time column that contains the beginning of the retention time segment that corresponds to the mass spectrum. If `NULL` (default), the function will auto-detect the first column in `c("ret_time_min","retention_time","rt","RT")` that exists in `mfd`. If none is found, blanks are removed ignoring retention time.

Value

A data.table having molecular formula assignments for each mass. ume_filter_formulas(mfd = mf_data_demo, dbe_o_max = 15, norm_int_min = 2)

See Also

Other Formula subsetting: `filter_int()`, `filter_mass_accuracy()`, `filter_mf_data()`, `remove_blanks()`, `subset_known_mf()`, `ume_assign_formulas()`

Other ume wrapper: `ume_assign_formulas()`

`ume_name_dictionary` *UME name dictionary*

Description

A compact dictionary table used internally by UME for column-name normalization, mapping external software names to UME names, and assigning human-readable labels in tables, plots, and reports.

The object is generated from the MarChem database table `tab_admin_name_dictionary`.

Usage

```
ume_name_dictionary
```

Format

A `data.table` with the following columns:

source_system Character. Source system of the name, e.g. `"UME"` or `"lambda_miner"`.

source_name Character. Original name as used in the source system.

label Character. Canonical UME name.

synonym_name Character. Alternative or external name mapped to `label`.

nice_label Character. Human-readable label used for plotting, tables, and reports.

semantic_type Character. Type of mapped concept, e.g. `"col_name"` or `"formula_category"`.

target Character. Intended target context, e.g. `"ume_labels"`, `"known_mf"`, or `"mfd"`.

priority Integer. Mapping priority used when multiple mappings are available.

unit Character. Optional unit associated with the variable.

Details

UME name dictionary

For internal UME labels, `source_name`, `synonym_name`, and `label` are often identical. For external sources, `synonym_name` stores the external column name and `label` stores the corresponding UME name.

Examples include:

- `peak_mz` -> `mz`
- `formula_error_relative` -> `ppm`
- `measurement_name` -> `file_id`

Formula category labels such as `CRAM`, `ideg_neg`, or `marine_dom` are also stored in this object and can be used to create readable column names after reshaping operations.

See Also

as_peaklist()

Other ume data: known_mf, lib_demo, masses, mf_data_demo, peaklist_demo

---

uplot_cluster *uplot_cluster*

Description

This function plots the results of a cluster analysis and a multi-dimensional scaling (MDS) plot based on molecular formula data.

Usage

```
uplot_cluster(
  mfd,
  grp = "file_id",
  int_col = "norm_int",
  dist_method = NULL,
  hclust_method = "average",
  scale_distance = FALSE,
  ...
)
```

Arguments

mfd	data.table with molecular formula data as derived from ume::assign_formulas. Column names of elements/isotopes must match names in the isotope column of ume::masses; values are integers representing counts per formula.
grp	Character vector. Names of columns (e.g., sample or file identifiers) used to aggregate results.
int_col	Character. The name of the column that contains the intensity values to be used (e.g. for clustering or color coding). Default usually is "norm_int" for normalized intensity values.
dist_method	Character or NULL. Distance method passed to vegan::vegdist(). If NULL, "euclidean" is used for int_col = "logratio_value" and "bray" otherwise.
hclust_method	Character. Clustering method passed to stats::hclust(). Default is "average".
scale_distance	Logical. If TRUE, distances are multiplied by 100. Default is FALSE.
...	Additional arguments passed to vegan::vegdist().

Details

Plot Cluster Analysis and Multi-Dimensional Scaling

The input data are first converted to a wide sample-by-feature matrix using `as_ume_matrix()`. A dissimilarity matrix is then calculated with `vegan::vegdist()`, followed by hierarchical clustering and non-metric MDS.

The default distance method is `"euclidean"`, which is suitable for log-ratio transformed data such as `logratio_value` and also valid for normalized intensities. For non-negative intensity data such as `norm_int`, `"bray"` can be a useful alternative.

Value

A named list with two elements:

`dendrogram` A `recordedplot` object containing the hierarchical clustering dendrogram.

`mds` A `plotly` object representing the two-dimensional MDS scatter plot.

Note

For CLR/ALR-transformed log-ratio values, use `dist_method = "euclidean"`. For non-negative normalized intensities, `dist_method = "bray"` is also commonly used.

See Also

Other uplots: `uplot_cvm()`, `uplot_dbe_minus_o_freq()`, `uplot_dbe_vs_c()`, `uplot_dbe_vs_ma()`, `uplot_dbe_vs_o()`, `uplot_freq_ma()`, `uplot_freq_vs_ppm()`, `uplot_hc_vs_m()`, `uplot_heteroatoms()`, `uplot_isotope_precision()`, `uplot_kmd()`, `uplot_lcms()`, `uplot_ma_vs_mz()`, `uplot_ms()`, `uplot_n_mf_per_sample()`, `uplot_pca()`, `uplot_ratios()`, `uplot_reproducibility()`, `uplot_ri_vs_sample()`, `uplot_vk()`

Examples

```
out <- uplot_cluster(
  mfd = mf_data_demo,
  grp = "file_id",
  int_col = "norm_int"
)

out$dendrogram
out$mds

## Not run:
mfd_lr <- calc_logratio(
  mfd = mf_data_demo,
  method = "clr",
  zero = "min"
)

out_lr <- uplot_cluster(
  mfd = mfd_lr,
  grp = "file_id",
```

```
  int_col = "logratio_value",
  dist_method = "euclidean"
)

## End(Not run)
```

---

uplot_cvm *Carbon vs Mass (CvM) Diagram*

Description

Generates a scatter plot of nominal molecular mass (nm) versus carbon count (12C), coloured by the median a supplied variable (z_var), following Reemtsma (2010).

Usage

```
uplot_cvm(
  mfd,
  z_var = "co_tot",
  fun = median,
  palname = "redblue",
  tf = FALSE,
  size_dots = 1.5,
  ...
)
```

Arguments

mfd	data.table with molecular formula data as derived from ume::assign_formulas. Column names of elements/isotopes must match names in the isotope column of ume::masses; values are integers representing counts per formula.
z_var	Character. Column name for variable used for color-coding. Content of column should be numeric.
fun	Function used to aggregate z_var for identical combinations. Default is median.
palname	Character. Name of the palette. Available palettes: "black", "redblue", "ratios", "rainbow", "awi", "viridis", "inferno", "terrain.colors", "gray".
tf	Logical. If TRUE, applies a transformation to the color scale (default is FALSE).
size_dots	Numeric. Size of the dots in the plot (default = 0.5).
...	Arguments passed on to uplot_wrapper title Optional character string. Plot title. Set title_show = FALSE to suppress the title entirely. title_show Logical. Display the plot title? Default: TRUE. title_size Numeric. Font size of the title (points). ume_logo Logical. Add the UME package logo? Default: TRUE.

`ume_label` Logical. Add the vertical UME branding label? Default: `TRUE`.

`map_labels` A list specifying which variables should get mapped to human-readable labels using `uplots_map_labels()`. Expected elements: `x`, `y`, `colour`, `fill`, `size`. May be `NULL` to suppress mapping.

`p` A ggplot object created by a uplot_* function.

`col_bar` Logical. Show colour bar?

`colour_scale` Character. Controls how the colour aesthetic is handled. One of:

- `"auto"` (default): Automatically chooses a continuous scale for numeric variables and a discrete scale for categorical variables.
- `"continuous"`: Forces a continuous colour scale.
- `"discrete"`: Forces a discrete colour scale.
- `"none"`: Do not modify the colour scale.

`x_npc_logo,y_npc_logo` NPC coordinates for logo placement.

`x_npc_label,y_npc_label` NPC coordinates for label placement.

`x_label_angle` Controls the orientation of labels on a discrete x-axis. Use `"auto"` to choose the angle based on the number and length of labels, or supply a numeric angle between 0 and 90 degrees. Default is `"auto"`.

`interactive` Logical. Return plotly object?

`plotly` Logical. If TRUE, return interactive plotly object.

`text_size` Numeric font size (in points).

Details

Carbon vs Mass (CvM) Diagram

Value

A ggplot or plotly object.

References

Reemtsma, T. (2010). The carbon versus mass diagram to visualize and exploit FTICR-MS data of natural organic matter. *Journal of Mass Spectrometry*, 45(4), 382–390. doi:10.1002/jms.1722

See Also

Other uplots: `uplot_cluster()`, `uplot_dbe_minus_o_freq()`, `uplot_dbe_vs_c()`, `uplot_dbe_vs_ma()`, `uplot_dbe_vs_o()`, `uplot_freq_ma()`, `uplot_freq_vs_ppm()`, `uplot_hc_vs_m()`, `uplot_heteroatoms()`, `uplot_isotope_precision()`, `uplot_kmd()`, `uplot_lcms()`, `uplot_ma_vs_mz()`, `uplot_ms()`, `uplot_n_mf_per_sample()`, `uplot_pca()`, `uplot_ratios()`, `uplot_reproducibility()`, `uplot_ri_vs_sample()`, `uplot_vk()`

Examples

```
uplot_cvm(mfd = mf_data_demo, z_var = "co_tot", ume_logo = FALSE)
uplot_cvm(mfd = mf_data_demo, z_var = "norm_int", palname = "viridis")
```

```
## Not run:
uplot_cvm(mfd = mf_data_demo, z_var = "co_tot", interactive = TRUE)
uplot_cvm(mf_data_demo, base_size = 11, palname = "awi", tf = TRUE,
  title_show = FALSE, col_bar = FALSE)

## End(Not run)
```

`uplot_dbe_minus_o_freq` *Frequency Plot of DBE - O atoms*

Description

Bar plot showing the frequency distribution of double bond equivalents (dbe) minus the number of oxygen atoms in a molecular formula (`dbe_o`). The unified UME plotting system is applied (theme, labels, logo, hover text, plotly).

The formula assignment strategy follows chemically motivated constraints and group-wise decision criteria based on DBE and oxygen content to distinguish reliable from equivocal molecular formulas.

Usage

```
uplot_dbe_minus_o_freq(mfd, ...)
```

Arguments

`mfd`	data.table with molecular formula data as derived from `ume::assign_formulas`. Column names of elements/isotopes must match names in the `isotope` column of `ume::masses`; values are integers representing counts per formula.
`...`	Arguments passed on to `uplot_wrapper`
	`title` Optional character string. Plot title. Set `title_show = FALSE` to suppress the title entirely.
	`title_show` Logical. Display the plot title? Default: `TRUE`.
	`title_size` Numeric. Font size of the title (points).
	`ume_logo` Logical. Add the UME package logo? Default: `TRUE`.
	`ume_label` Logical. Add the vertical UME branding label? Default: `TRUE`.
	`map_labels` A list specifying which variables should get mapped to human-readable labels using `uplots_map_labels()`. Expected elements: `x`, `y`, `colour`, `fill`, `size`. May be `NULL` to suppress mapping.
	`p` A ggplot object created by a uplot_* function.
	`palname` Colour palette name passed to f_colorz().
	`col_bar` Logical. Show colour bar?

`colour_scale` Character. Controls how the colour aesthetic is handled. One of:

- `"auto"` (default): Automatically chooses a continuous scale for numeric variables and a discrete scale for categorical variables.
- `"continuous"`: Forces a continuous colour scale.
- `"discrete"`: Forces a discrete colour scale.
- `"none"`: Do not modify the colour scale.

`x_npc_logo,y_npc_logo` NPC coordinates for logo placement.

`x_npc_label,y_npc_label` NPC coordinates for label placement.

`x_label_angle` Controls the orientation of labels on a discrete x-axis. Use `"auto"` to choose the angle based on the number and length of labels, or supply a numeric angle between 0 and 90 degrees. Default is `"auto"`.

`interactive` Logical. Return plotly object?

`plotly` Logical. If TRUE, return interactive plotly object.

`size_dots` Numeric. Size of the dots in the plot (default = 0.5).

`text_size` Numeric font size (in points).

Details

Frequency Plot of DBE - O

Value

ggplot or plotly object

References

Herzsprung, P., Hertkorn, N., von Tümpling, W., Harir, M., Friese, K., & Schmitt-Kopplin, P. (2014). Understanding molecular formula assignment of Fourier transform ion cyclotron resonance mass spectrometry data of natural organic matter from a chemical point of view. *Analytical and Bioanalytical Chemistry*, 406(30), 7977–7987. doi:10.1007/s002160148249y

See Also

Other uplots: `uplot_cluster()`, `uplot_cvm()`, `uplot_dbe_vs_c()`, `uplot_dbe_vs_ma()`, `uplot_dbe_vs_o()`, `uplot_freq_ma()`, `uplot_freq_vs_ppm()`, `uplot_hc_vs_m()`, `uplot_heteroatoms()`, `uplot_isotope_precision()`, `uplot_kmd()`, `uplot_lcms()`, `uplot_ma_vs_mz()`, `uplot_ms()`, `uplot_n_mf_per_sample()`, `uplot_pca()`, `uplot_ratios()`, `uplot_reproducibility()`, `uplot_ri_vs_sample()`, `uplot_vk()`

Examples

```
uplot_dbe_minus_o_freq(mf_data_demo)
uplot_dbe_minus_o_freq(mf_data_demo, interactive = TRUE, ume_logo = FALSE, title_show = FALSE)
```

uplot_dbe_vs_c *Plot DBE vs Carbon Atoms*

Description

Creates a scatter plot of DBE (double bond equivalents) vs. number of carbon atoms. Points are color-coded by a selected variable (z_var). The plot follows the same stylistic conventions as the other uplot_* functions, including the unified theme and optional UME caption.

This approach follows the DBE/C concept introduced for identifying aromatic sub-structures in a molecular formula.

Usage

```
uplot_dbe_vs_c(
  mfd,
  z_var = "norm_int",
  fun = median,
  palname = "redblue",
  tf = FALSE,
  size_dots = 1.5,
  ...
)
```

Arguments

mfd	data.table with molecular formula data as derived from ume::assign_formulas. Column names of elements/isotopes must match names in the isotope column of ume::masses; values are integers representing counts per formula.
z_var	Character. Column name for variable used for color-coding. Content of column should be numeric.
fun	Function used to aggregate z_var for identical combinations. Default is median.
palname	Character. Name of the palette. Available palettes: "black", "redblue", "ratios", "rainbow", "awi", "viridis", "inferno", "terrain.colors", "gray".
tf	Logical. If TRUE, applies a transformation to the color scale (default is FALSE).
size_dots	Numeric. Size of the dots in the plot (default = 0.5).
...	Arguments passed on to uplot_wrapper title Optional character string. Plot title. Set title_show = FALSE to suppress the title entirely. title_show Logical. Display the plot title? Default: TRUE. title_size Numeric. Font size of the title (points). ume_logo Logical. Add the UME package logo? Default: TRUE. ume_label Logical. Add the vertical UME branding label? Default: TRUE.

`map_labels` A list specifying which variables should get mapped to human-readable labels using `uplots_map_labels()`. Expected elements: `x`, `y`, `colour`, `fill`, `size`. May be `NULL` to suppress mapping.

`p` A ggplot object created by a uplot_* function.

`col_bar` Logical. Show colour bar?

`colour_scale` Character. Controls how the colour aesthetic is handled. One of:

- `"auto"` (default): Automatically chooses a continuous scale for numeric variables and a discrete scale for categorical variables.
- `"continuous"`: Forces a continuous colour scale.
- `"discrete"`: Forces a discrete colour scale.
- `"none"`: Do not modify the colour scale.

`x_npc_logo,y_npc_logo` NPC coordinates for logo placement.

`x_npc_label,y_npc_label` NPC coordinates for label placement.

`x_label_angle` Controls the orientation of labels on a discrete x-axis. Use `"auto"` to choose the angle based on the number and length of labels, or supply a numeric angle between 0 and 90 degrees. Default is `"auto"`.

`interactive` Logical. Return plotly object?

`plotly` Logical. If TRUE, return interactive plotly object.

`text_size` Numeric font size (in points).

Value

A ggplot2 object or a plotly object (if `plotly = TRUE`).

References

Hockaday, W. C., Grannas, A. M., Kim, S., & Hatcher, P. G. (2006). Direct molecular evidence for the degradation and mobility of black carbon in soils from ultrahigh-resolution mass spectral analysis of dissolved organic matter from a fire-impacted forest soil. *Organic Geochemistry*, 37(4), 501–510. doi:10.1016/j.orggeochem.2005.11.003

See Also

Other uplots: `uplot_cluster()`, `uplot_cvm()`, `uplot_dbe_minus_o_freq()`, `uplot_dbe_vs_ma()`, `uplot_dbe_vs_o()`, `uplot_freq_ma()`, `uplot_freq_vs_ppm()`, `uplot_hc_vs_m()`, `uplot_heteroatoms()`, `uplot_isotope_precision()`, `uplot_kmd()`, `uplot_lcms()`, `uplot_ma_vs_mz()`, `uplot_ms()`, `uplot_n_mf_per_sample()`, `uplot_pca()`, `uplot_ratios()`, `uplot_reproducibility()`, `uplot_ri_vs_sample()`, `uplot_vk()`

Examples

```
uplot_dbe_vs_c(mf_data_demo, z_var = "norm_int")
```

uplot_dbe_vs_ma *Plot DBE vs ppm with Option for Interactive Plot*

Description

This function generates a scatter plot of DBE (Double Bond Equivalent) versus parts per million (ppm) from the provided data. It also provides the option to customize the appearance and to return an interactive plotly plot.

Usage

```
uplot_dbe_vs_ma(
  mfd,
  z_var = "norm_int",
  fun = median,
  palname = "redblue",
  tf = FALSE,
  size_dots = 1.5,
  ...
)
```

Arguments

mfd	data.table with molecular formula data as derived from ume::assign_formulas. Column names of elements/isotopes must match names in the isotope column of ume::masses; values are integers representing counts per formula.
z_var	Character. Column name for variable used for color-coding. Content of column should be numeric.
fun	Function used to aggregate z_var for identical combinations. Default is median.
palname	Character. Name of the palette. Available palettes: "black", "redblue", "ratios", "rainbow", "awi", "viridis", "inferno", "terrain.colors", "gray".
tf	Logical. If TRUE, applies a transformation to the color scale (default is FALSE).
size_dots	Numeric. Size of the dots in the plot (default = 0.5).
...	Arguments passed on to uplot_wrapper title Optional character string. Plot title. Set title_show = FALSE to suppress the title entirely. title_show Logical. Display the plot title? Default: TRUE. title_size Numeric. Font size of the title (points). ume_logo Logical. Add the UME package logo? Default: TRUE. ume_label Logical. Add the vertical UME branding label? Default: TRUE. map_labels A list specifying which variables should get mapped to human-readable labels using uplots_map_labels(). Expected elements: x, y, colour, fill, size. May be NULL to suppress mapping. p A ggplot object created by a uplot_* function.

col_bar Logical. Show colour bar?

colour_scale Character. Controls how the colour aesthetic is handled. One of:

- "auto" (default): Automatically chooses a continuous scale for numeric variables and a discrete scale for categorical variables.
- "continuous": Forces a continuous colour scale.
- "discrete": Forces a discrete colour scale.
- "none": Do not modify the colour scale.

x_npc_logo,y_npc_logo NPC coordinates for logo placement.

x_npc_label,y_npc_label NPC coordinates for label placement.

x_label_angle Controls the orientation of labels on a discrete x-axis. Use "auto" to choose the angle based on the number and length of labels, or supply a numeric angle between 0 and 90 degrees. Default is "auto".

interactive Logical. Return plotly object?

plotly Logical. If TRUE, return interactive plotly object.

text_size Numeric font size (in points).

Value

A ggplot or plotly object.

See Also

Other uplots: uplot_cluster(), uplot_cvm(), uplot_dbe_minus_o_freq(), uplot_dbe_vs_c(), uplot_dbe_vs_o(), uplot_freq_ma(), uplot_freq_vs_ppm(), uplot_hc_vs_m(), uplot_heteroatoms(), uplot_isotope_precision(), uplot_kmd(), uplot_lcms(), uplot_ma_vs_mz(), uplot_ms(), uplot_n_mf_per_sample(), uplot_pca(), uplot_ratios(), uplot_reproducibility(), uplot_ri_vs_sample(), uplot_vk()

Examples

```
uplot_dbe_vs_ma(mfd = mf_data_demo, size_dots = 1)
```

---

uplot_dbe_vs_o *Plot DBE vs Oxygen Atoms (cf. Herzsprung et al. 2014) with Option for Interactive Plot*

Description

This function generates a scatter plot of Double Bond Equivalent (DBE) versus the number of oxygen atoms (o). It allows for optional customization of colors based on a specified variable (z_var) and offers the option to convert the plot to an interactive plotly object.

Usage

```
uplot_dbe_vs_o(
  mfd,
  z_var = "norm_int",
  fun = median,
  palname = "redblue",
  tf = FALSE,
  size_dots = 1.5,
  ...
)
```

Arguments

mfd	data.table with molecular formula data as derived from ume::assign_formulas. Column names of elements/isotopes must match names in the isotope column of ume::masses; values are integers representing counts per formula.
z_var	Character. Column name for variable used for color-coding. Content of column should be numeric.
fun	Function used to aggregate z_var for identical combinations. Default is median.
palname	Character. Name of the palette. Available palettes: "black", "redblue", "ratios", "rainbow", "awi", "viridis", "inferno", "terrain.colors", "gray".
tf	Logical. If TRUE, applies a transformation to the color scale (default is FALSE).
size_dots	Numeric. Size of the dots in the plot (default = 0.5).
...	Arguments passed on to uplot_wrapper

title Optional character string. Plot title. Set title_show = FALSE to suppress the title entirely.

title_show Logical. Display the plot title? Default: TRUE.

title_size Numeric. Font size of the title (points).

ume_logo Logical. Add the UME package logo? Default: TRUE.

ume_label Logical. Add the vertical UME branding label? Default: TRUE.

map_labels A list specifying which variables should get mapped to human-readable labels using uplots_map_labels(). Expected elements: x, y, colour, fill, size. May be NULL to suppress mapping.

p A ggplot object created by a uplot_* function.

col_bar Logical. Show colour bar?

colour_scale Character. Controls how the colour aesthetic is handled. One of:

- "auto" (default): Automatically chooses a continuous scale for numeric variables and a discrete scale for categorical variables.
- "continuous": Forces a continuous colour scale.
- "discrete": Forces a discrete colour scale.
- "none": Do not modify the colour scale.

x_npc_logo,y_npc_logo NPC coordinates for logo placement.

x_npc_label,y_npc_label NPC coordinates for label placement.

`x_label_angle` Controls the orientation of labels on a discrete x-axis. Use `"auto"` to choose the angle based on the number and length of labels, or supply a numeric angle between 0 and 90 degrees. Default is `"auto"`.

`interactive` Logical. Return plotly object?

`plotly` Logical. If TRUE, return interactive plotly object.

`text_size` Numeric font size (in points).

Value

A ggplot or plotly object.

See Also

Other uplots: `uplot_cluster()`, `uplot_cvm()`, `uplot_dbe_minus_o_freq()`, `uplot_dbe_vs_c()`, `uplot_dbe_vs_ma()`, `uplot_freq_ma()`, `uplot_freq_vs_ppm()`, `uplot_hc_vs_m()`, `uplot_heteroatoms()`, `uplot_isotope_precision()`, `uplot_kmd()`, `uplot_lcms()`, `uplot_ma_vs_mz()`, `uplot_ms()`, `uplot_n_mf_per_sample()`, `uplot_pca()`, `uplot_ratios()`, `uplot_reproducibility()`, `uplot_ri_vs_sample()`, `uplot_vk()`

`uplot_freq` *Frequency Plot of a Selected Variable*

Description

Creates a frequency plot (bar plot) for a selected variable in a molecular formula dataset. Values are grouped and counted, then visualized as bars. A unified UME plot theme is applied for consistent styling across all uplot_* functions.

Usage

```
uplot_freq(
  mfd,
  var = "14N",
  col = "grey",
  space = 0.5,
  width = 0.3,
  logo = TRUE,
  gg_size = 12,
  plotly = FALSE,
  ...
)
```

Arguments

`mfd` data.table with molecular formula data as derived from `ume::assign_formulas`. Column names of elements/isotopes must match names in the `isotope` column of `ume::masses`; values are integers representing counts per formula.

var	Character. Name of the variable for which the frequency distribution should be plotted (e.g. "14N").
col	Bar fill color.
space	Not used (kept for backward compatibility).
width	Bar width.
logo	Logical. If TRUE, adds a UME caption.
gg_size	Base text size for theme_uplots(). Default = 12.
plotly	Logical. If TRUE, return interactive plotly object.
...	Additional arguments passed to methods.

Value

A ggplot object, or a plotly object when plotly = TRUE.

---

uplot_freq_ma *Histogram of Mass Accuracy*

Description

Creates a histogram of mass accuracy values (ppm). Includes summary statistics (median, 2.5% and 97.5% quantiles). Follows general uplot behavior:

- returns a ggplot2 object by default
- converts to plotly *only if* plotly = TRUE
- uses caption-style UME logo

Usage

```
uplot_freq_ma(mfd, ma_col = "ppm", bins = NULL, ...)
```

Arguments

mfd	data.table with molecular formula data as derived from ume::assign_formulas. Column names of elements/isotopes must match names in the isotope column of ume::masses; values are integers representing counts per formula.
ma_col	String. Name of the column having mass accuracy values.
bins	Numeric. Number of bins(e.g. for the x-scale in a histogram)
...	Arguments passed on to uplot_wrapper title Optional character string. Plot title. Set title_show = FALSE to suppress the title entirely. title_show Logical. Display the plot title? Default: TRUE. title_size Numeric. Font size of the title (points). ume_logo Logical. Add the UME package logo? Default: TRUE.

ume_label Logical. Add the vertical UME branding label? Default: TRUE.

map_labels A list specifying which variables should get mapped to human-readable labels using uplots_map_labels(). Expected elements: x, y, colour, fill, size. May be NULL to suppress mapping.

p A ggplot object created by a uplot_* function.

palname Colour palette name passed to f_colorz().

col_bar Logical. Show colour bar?

colour_scale Character. Controls how the colour aesthetic is handled. One of:

- "auto" (default): Automatically chooses a continuous scale for numeric variables and a discrete scale for categorical variables.
- "continuous": Forces a continuous colour scale.
- "discrete": Forces a discrete colour scale.
- "none": Do not modify the colour scale.

x_npc_logo,y_npc_logo NPC coordinates for logo placement.

x_npc_label,y_npc_label NPC coordinates for label placement.

x_label_angle Controls the orientation of labels on a discrete x-axis. Use "auto" to choose the angle based on the number and length of labels, or supply a numeric angle between 0 and 90 degrees. Default is "auto".

interactive Logical. Return plotly object?

plotly Logical. If TRUE, return interactive plotly object.

size_dots Numeric. Size of the dots in the plot (default = 0.5).

text_size Numeric font size (in points).

Value

ggplot or plotly object

See Also

Other uplots: uplot_cluster(), uplot_cvm(), uplot_dbe_minus_o_freq(), uplot_dbe_vs_c(), uplot_dbe_vs_ma(), uplot_dbe_vs_o(), uplot_freq_vs_ppm(), uplot_hc_vs_m(), uplot_heteroatoms(), uplot_isotope_precision(), uplot_kmd(), uplot_lcms(), uplot_ma_vs_mz(), uplot_ms(), uplot_n_mf_per_sample(), uplot_pca(), uplot_ratios(), uplot_reproducibility(), uplot_ri_vs_sample(), uplot_vk()

---

uplot_freq_vs_ppm *Mass Accuracy Frequency Histogram*

Description

Creates a histogram showing the frequency distribution of mass accuracy values (ppm). Displays median and quantile statistics in the title and optionally adds a UME caption (logo). The plot uses the unified UME theme (theme_uplots()), ensuring visual consistency across all uplot_* functions.

Usage

```
uplot_freq_vs_ppm(
  df,
  col = "grey",
  width = 0.01,
  gg_size = 12,
  logo = TRUE,
  plotly = FALSE
)
```

Arguments

df	A data.table or data.frame containing columns: • ppm — mass accuracy in ppm • 14N, 32S, 31P, dbe_o — required for consistency with UME QC tools
col	Character. Histogram bar color. Default "grey".
width	Numeric. Histogram bin width (not used when bins = 100).
gg_size	Base text size for theme_uplots(). Default = 12.
logo	Logical. If TRUE, adds a UME caption.
plotly	Logical. If TRUE, return interactive plotly object.

Details

This plot is useful for visual inspection of mass accuracy performance. The required additional columns (14N, 32S, 31P, dbe_o) ensure that the dataset is a complete UME molecular formula table and can be compared to other quality-control plots.

Value

A **ggplot2 histogram**, or a **plotly object** if plotly = TRUE.

See Also

Other uplots: uplot_cluster(), uplot_cvm(), uplot_dbe_minus_o_freq(), uplot_dbe_vs_c(), uplot_dbe_vs_ma(), uplot_dbe_vs_o(), uplot_freq_ma(), uplot_hc_vs_m(), uplot_heteroatoms(), uplot_isotope_precision(), uplot_kmd(), uplot_lcms(), uplot_ma_vs_mz(), uplot_ms(), uplot_n_mf_per_sample(), uplot_pca(), uplot_ratios(), uplot_reproducibility(), uplot_ri_vs_sample(), uplot_vk()

Examples

```
uplot_freq_vs_ppm(mf_data_demo)
```

uplot_hc_vs_m *H/C vs Molecular Mass Plot*

Description

Creates a scatter plot of the hydrogen-to-carbon ratio (H/C) versus molecular mass (nm). Points are color-coded according to a selected intensity or property column (int_col). This visualization follows the conceptual design in Schmitt-Kopplin et al. (2010).

Usage

```
uplot_hc_vs_m(
  mfd,
  int_col = "norm_int",
  palname = "redblue",
  size_dots = 1.2,
  plotly = FALSE,
  ...
)
```

Arguments

mfd	data.table with molecular formula data as derived from ume::assign_formulas. Column names of elements/isotopes must match names in the isotope column of ume::masses; values are integers representing counts per formula.
int_col	Character. The name of the column that contains the intensity values to be used (e.g. for clustering or color coding). Default usually is "norm_int" for normalized intensity values.
palname	Character. Name of the palette. Available palettes: "black", "redblue", "ratios", "rainbow", "awi", "viridis", "inferno", "terrain.colors", "gray".
size_dots	Numeric. Size of the dots in the plot (default = 0.5).
plotly	Logical. If TRUE, return interactive plotly object.
...	Arguments passed on to uplot_wrapper title Optional character string. Plot title. Set title_show = FALSE to suppress the title entirely. title_show Logical. Display the plot title? Default: TRUE. title_size Numeric. Font size of the title (points). ume_logo Logical. Add the UME package logo? Default: TRUE. ume_label Logical. Add the vertical UME branding label? Default: TRUE. map_labels A list specifying which variables should get mapped to human-readable labels using uplots_map_labels(). Expected elements: x, y, colour, fill, size. May be NULL to suppress mapping. p A ggplot object created by a uplot_* function. col_bar Logical. Show colour bar?

colour_scale Character. Controls how the colour aesthetic is handled. One of:

- "auto" (default): Automatically chooses a continuous scale for numeric variables and a discrete scale for categorical variables.
- "continuous": Forces a continuous colour scale.
- "discrete": Forces a discrete colour scale.
- "none": Do not modify the colour scale.

x_npc_logo,y_npc_logo NPC coordinates for logo placement.

x_npc_label,y_npc_label NPC coordinates for label placement.

x_label_angle Controls the orientation of labels on a discrete x-axis. Use "auto" to choose the angle based on the number and length of labels, or supply a numeric angle between 0 and 90 degrees. Default is "auto".

interactive Logical. Return plotly object?

text_size Numeric font size (in points).

Value

A ggplot2 object, or a plotly object if plotly = TRUE.

See Also

Other uplots: uplot_cluster(), uplot_cvm(), uplot_dbe_minus_o_freq(), uplot_dbe_vs_c(), uplot_dbe_vs_ma(), uplot_dbe_vs_o(), uplot_freq_ma(), uplot_freq_vs_ppm(), uplot_heteroatoms(), uplot_isotope_precision(), uplot_kmd(), uplot_lcms(), uplot_ma_vs_mz(), uplot_ms(), uplot_n_mf_per_sample(), uplot_pca(), uplot_ratios(), uplot_reproducibility(), uplot_ri_vs_sample(), uplot_vk()

Examples

```
uplot_hc_vs_m(mf_data_demo, int_col = "norm_int")
```

---

uplot_heteroatoms *Heteroatom Combination vs Mass Accuracy*

Description

Produces a boxplot visualizing the distribution of mass accuracy (ppm) for different heteroatom combinations (nsp_type) defined by the number of nitrogen (N), sulfur (S), and phosphorus (P) atoms in each formula.

The plot can be returned as either a ggplot object or as an interactive plotly object (plotly = TRUE). An optional "UltraMassExplorer" watermark can be added.

Usage

```
uplot_heteroatoms(df, col = "grey", gg_size = 12, logo = TRUE, plotly = FALSE)
```

Arguments

df	A data.table containing at least: • nsp_type: character or factor indicating heteroatom combinations • ppm: numeric mass accuracy values
col	Character. Box color. Default "grey".
gg_size	Base text size for theme_uplots(). Default = 12.
logo	Logical. If TRUE, adds a UME caption.
plotly	Logical. If TRUE, return interactive plotly object.

Value

A ggplot or plotly interactive boxplot.

See Also

Other uplots: uplot_cluster(), uplot_cvm(), uplot_dbe_minus_o_freq(), uplot_dbe_vs_c(), uplot_dbe_vs_ma(), uplot_dbe_vs_o(), uplot_freq_ma(), uplot_freq_vs_ppm(), uplot_hc_vs_m(), uplot_isotope_precision(), uplot_kmd(), uplot_lcms(), uplot_ma_vs_mz(), uplot_ms(), uplot_n_mf_per_sample(), uplot_pca(), uplot_ratios(), uplot_reproducibility(), uplot_ri_vs_sample(), uplot_vk()

Examples

```
uplot_heteroatoms(mf_data_demo)
```

---

uplot_isotope_precision

Precision of Isotope Abundance

Description

Isotope precision describes how reliably the instrument reproduces the expected intensity of the naturally occurring ^{13}C isotope peak relative to its corresponding monoisotopic ^{12}C peak.

Usage

```
uplot_isotope_precision(
  mfd,
  z_var = "nsp_tot",
  int_col = "norm_int",
  size_dots = 1.5,
  bins = 100,
  data_reduction = FALSE,
  tf = FALSE,
```

```
  logo = TRUE,
  plotly = FALSE,
  cex.axis = 1,
  cex.lab = 1.4
)
```

Arguments

mfd	data.table with molecular formula data as derived from ume::assign_formulas. Column names of elements/isotopes must match names in the isotope column of ume::masses; values are integers representing counts per formula.
z_var	Column used for color mapping (default: "nsp_tot")
int_col	Intensity column (default: "norm_int")
size_dots	Numeric. Size of the dots in the plot (default = 0.5).
bins	Number of bins used when data_reduction = TRUE
data_reduction	Logical. If TRUE, bins the data and uses bin medians (recommended for very large datasets; speeds up rendering massively).
tf	Logical. If TRUE, applies a transformation to the color scale (default is FALSE).
logo	Logical. If TRUE, adds a UME caption.
plotly	Logical. Return a plotly object instead of ggplot.
cex.axis	Numeric. Size of axis text (default is 1).
cex.lab	Numeric. Size of axis labels (default is 1.4).

Details

The measured $^{13}\mathrm{C}$ signal provides an *intrinsic* validation of molecular formula assignments.

For a molecule containing n carbon atoms with a natural abundance of 1.07% for $^{13}\mathrm{C}$, the theoretical relative intensity of the isotope peak $^{13}\mathrm{C}_1{}^{12}\mathrm{C}_{n-1}$ is:

$$I_{theo} = n \times 0.0107$$

The measured intensity I_{meas} provides an independent estimate of the number of carbon atoms:

$$n_{calc} = \frac{I_{meas}}{0.0107}$$

From this, the deviation in carbon number can be defined:

$$C_{dev} = n_{assigned} - n_{calc}$$

A value of $C_{dev} = 0$ indicates perfect agreement between the formula assignment and the isotope-based estimate. Negative values indicate that the measured isotope abundance is lower than expected.

Isotope precision is assessed by evaluating the distribution of C_{dev} across peaks with sufficient signal quality. C_{dev} becomes small and stable at higher signal-to-noise ratios (S/N). Therefore,

isotopic peak ratios for intense mass signals provide an internal metric for validating molecular formula assignments. The function visualizes the deviation between measured and theoretical ^{13}C isotope ratios. Supports optional data reduction (binning) to enhance interactive rendering speed in `Plotly`.

Value

A `ggplot` or `plotly` object.

See Also

Other uplots: `uplot_cluster()`, `uplot_cvm()`, `uplot_dbe_minus_o_freq()`, `uplot_dbe_vs_c()`, `uplot_dbe_vs_ma()`, `uplot_dbe_vs_o()`, `uplot_freq_ma()`, `uplot_freq_vs_ppm()`, `uplot_hc_vs_m()`, `uplot_heteroatoms()`, `uplot_kmd()`, `uplot_lcms()`, `uplot_ma_vs_mz()`, `uplot_ms()`, `uplot_n_mf_per_sample()`, `uplot_pca()`, `uplot_ratios()`, `uplot_reproducibility()`, `uplot_ri_vs_sample()`, `uplot_vk()`

Other isotopes: `calc_isotope_pattern()`, `create_isotope_expanded_table()`, `eval_isotopes()`

---

uplot_kmd *Kendrick Mass Defect (KMD) vs. Nominal Mass Plot*

Description

This function generates a scatter plot of Kendrick Mass Defect (KMD) versus nominal mass (`nm`), with color-coding based on a specified variable (`z_var`). Optionally, the plot can be returned as an interactive Plotly object.

Usage

```
uplot_kmd(
  mfd,
  z_var = "norm_int",
  fun = median,
  palname = "redblue",
  tf = FALSE,
  size_dots = 1,
  ...
)
```

Arguments

`mfd`	data.table with molecular formula data as derived from `ume::assign_formulas`. Column names of elements/isotopes must match names in the `isotope` column of `ume::masses`; values are integers representing counts per formula.
`z_var`	Character. Column name for variable used for color-coding. Content of column should be numeric.
`fun`	Function used to aggregate `z_var` for identical combinations. Default is `median`.

palname Character. Name of the palette. Available palettes: "black", "redblue", "ratios", "rainbow", "awi", "viridis", "inferno", "terrain.colors", "gray".

tf Logical. If TRUE, applies a transformation to the color scale (default is FALSE).

size_dots Numeric. Size of the dots in the plot (default = 0.5).

... Arguments passed on to uplot_wrapper

title Optional character string. Plot title. Set title_show = FALSE to suppress the title entirely.

title_show Logical. Display the plot title? Default: TRUE.

title_size Numeric. Font size of the title (points).

ume_logo Logical. Add the UME package logo? Default: TRUE.

ume_label Logical. Add the vertical UME branding label? Default: TRUE.

map_labels A list specifying which variables should get mapped to human-readable labels using uplots_map_labels(). Expected elements: x, y, colour, fill, size. May be NULL to suppress mapping.

p A ggplot object created by a uplot_* function.

col_bar Logical. Show colour bar?

colour_scale Character. Controls how the colour aesthetic is handled. One of:

- "auto" (default): Automatically chooses a continuous scale for numeric variables and a discrete scale for categorical variables.
- "continuous": Forces a continuous colour scale.
- "discrete": Forces a discrete colour scale.
- "none": Do not modify the colour scale.

x_npc_logo,y_npc_logo NPC coordinates for logo placement.

x_npc_label,y_npc_label NPC coordinates for label placement.

x_label_angle Controls the orientation of labels on a discrete x-axis. Use "auto" to choose the angle based on the number and length of labels, or supply a numeric angle between 0 and 90 degrees. Default is "auto".

interactive Logical. Return plotly object?

plotly Logical. If TRUE, return interactive plotly object.

text_size Numeric font size (in points).

Details

Kendrick Mass Defect (KMD) vs. Nominal Mass Plot

Value

A ggplot or plotly object.

References

Kendrick E. (1963). A mass scale based on CH_2 = 14.0000 for high resolution mass spectrometry of organic compounds. *Analytical Chemistry*, **35**, 2146–2154.

Hughey C.A., Hendrickson C.L., Rodgers R.P., Marshall A.G., Qian K.N. (2001). Kendrick mass defect spectrum: A compact visual analysis for ultrahigh-resolution broadband mass spectra. *Analytical Chemistry*, **73**, 4676–4681. doi:10.1021/ac010560w

See Also

Other uplots: uplot_cluster(), uplot_cvm(), uplot_dbe_minus_o_freq(), uplot_dbe_vs_c(), uplot_dbe_vs_ma(), uplot_dbe_vs_o(), uplot_freq_ma(), uplot_freq_vs_ppm(), uplot_hc_vs_m(), uplot_heteroatoms(), uplot_isotope_precision(), uplot_lcms(), uplot_ma_vs_mz(), uplot_ms(), uplot_n_mf_per_sample(), uplot_pca(), uplot_ratios(), uplot_reproducibility(), uplot_ri_vs_sample(), uplot_vk()

Examples

```
uplot_kmd(mf_data_demo, z_var = "norm_int")
```

---

uplot_lcms *Plot LC-MS Spectrum (or fallback MS if no RT available)*

Description

Creates a 3D LC–MS plot (RT x m/z x intensity) **when retention time is available**. If no retention-time column exists (e.g., with DI-FTMS demo data), the function gracefully falls back to `uplot_ms()` and issues an informative message.

Usage

```
uplot_lcms(
  pl,
  mass = "mz",
  peak_magnitude = "i_magnitude",
  retention_time = "ret_time_min",
  label = "file_id",
  logo = FALSE,
  ...
)
```

Arguments

pl	data.table containing peak data. Mandatory columns include neutral molecular mass (mass), peak magnitude (i_magnitude), and a peak identifier (peak_id).
mass	Column containing m/z values (default "mz").
peak_magnitude	Column containing intensity (default "i_magnitude").
retention_time	Column with retention time (default "ret_time_min").
label	Sample/group labeling column (default "file_id").
logo	Logical. If TRUE, adds a UME caption.
...	Additional arguments passed to methods.

Value

A plotly 3D visualization (LC-MS) or a 2D MS spectrum fallback.

See Also

Other uplots: uplot_cluster(), uplot_cvm(), uplot_dbe_minus_o_freq(), uplot_dbe_vs_c(), uplot_dbe_vs_ma(), uplot_dbe_vs_o(), uplot_freq_ma(), uplot_freq_vs_ppm(), uplot_hc_vs_m(), uplot_heteroatoms(), uplot_isotope_precision(), uplot_kmd(), uplot_ma_vs_mz(), uplot_ms(), uplot_n_mf_per_sample(), uplot_pca(), uplot_ratios(), uplot_reproducibility(), uplot_ri_vs_sample(), uplot_vk()

---

uplot_ma_vs_mz *Plot Mass Accuracy vs m/z*

Description

Generates a UME-style scatter plot showing mass accuracy (ppm) versus mass-to-charge ratio (m/z). Summary statistics (median, 2.5% and 97.5% quantiles) are displayed as horizontal reference lines and an annotation panel.

Usage

```
uplot_ma_vs_mz(mfd, ma_col = "ppm", plotly = FALSE, ...)
```

Arguments

mfd	data.table with molecular formula data as derived from ume::assign_formulas. Column names of elements/isotopes must match names in the isotope column of ume::masses; values are integers representing counts per formula.
ma_col	Character. Column containing mass accuracy (ppm).
plotly	Logical. If TRUE, return interactive plotly object.
...	Arguments passed on to uplot_wrapper title Optional character string. Plot title. Set title_show = FALSE to suppress the title entirely. title_show Logical. Display the plot title? Default: TRUE. title_size Numeric. Font size of the title (points). ume_logo Logical. Add the UME package logo? Default: TRUE. ume_label Logical. Add the vertical UME branding label? Default: TRUE. map_labels A list specifying which variables should get mapped to human-readable labels using uplots_map_labels(). Expected elements: x, y, colour, fill, size. May be NULL to suppress mapping. p A ggplot object created by a uplot_* function. palname Colour palette name passed to f_colorz(). col_bar Logical. Show colour bar?

colour_scale Character. Controls how the colour aesthetic is handled. One of:

- "auto" (default): Automatically chooses a continuous scale for numeric variables and a discrete scale for categorical variables.
- "continuous": Forces a continuous colour scale.
- "discrete": Forces a discrete colour scale.
- "none": Do not modify the colour scale.

x_npc_logo,y_npc_logo NPC coordinates for logo placement.

x_npc_label,y_npc_label NPC coordinates for label placement.

x_label_angle Controls the orientation of labels on a discrete x-axis. Use "auto" to choose the angle based on the number and length of labels, or supply a numeric angle between 0 and 90 degrees. Default is "auto".

interactive Logical. Return plotly object?

size_dots Numeric. Size of the dots in the plot (default = 0.5).

text_size Numeric font size (in points).

Value

A ggplot or plotly object.

See Also

Other uplots: uplot_cluster(), uplot_cvm(), uplot_dbe_minus_o_freq(), uplot_dbe_vs_c(), uplot_dbe_vs_ma(), uplot_dbe_vs_o(), uplot_freq_ma(), uplot_freq_vs_ppm(), uplot_hc_vs_m(), uplot_heteroatoms(), uplot_isotope_precision(), uplot_kmd(), uplot_lcms(), uplot_ms(), uplot_n_mf_per_sample(), uplot_pca(), uplot_ratios(), uplot_reproducibility(), uplot_ri_vs_sample(), uplot_vk()

Examples

```
uplot_ma_vs_mz(mf_data_demo, ma_col = "ppm")
```

---

uplot_ms *Plot Mass Spectrum*

Description

Plots a mass spectrum, showing peak magnitude versus mass-to-charge ratio (m/z).

Optionally reduces the dataset by selecting the most abundant peaks per spectrum.

Usage

```
uplot_ms(
  pl,
  mass = "mz",
  peak_magnitude = "i_magnitude",
  label = "file_id",
  logo = FALSE,
  plotly = TRUE,
  data_reduction = 1,
  ...
)
```

Arguments

pl	data.table containing peak data. Mandatory columns include neutral molecular mass (mass), peak magnitude (i_magnitude), and a peak identifier (peak_id).
mass	Character. Name of the column containing mass-to-charge or mass information (default = "mz").
peak_magnitude	Character. Name of the column containing peak magnitude (default = "i_magnitude").
label	Character. Name of the column identifying individual spectra (default = "file_id").
logo	Logical. If TRUE, adds a UME caption.
plotly	Logical. If TRUE, return interactive plotly object.
data_reduction	Numeric between 0 and 1. Fraction of the most abundant peaks to retain per spectrum. Default = 1 (no reduction). If set to 0, a minimum of 0.01 is used to ensure some data is displayed.
...	Additional arguments passed to methods.

Value

A ggplot object or a plotly object if plotly = TRUE.

See Also

Other uplots: uplot_cluster(), uplot_cvm(), uplot_dbe_minus_o_freq(), uplot_dbe_vs_c(), uplot_dbe_vs_ma(), uplot_dbe_vs_o(), uplot_freq_ma(), uplot_freq_vs_ppm(), uplot_hc_vs_m(), uplot_heteroatoms(), uplot_isotope_precision(), uplot_kmd(), uplot_lcms(), uplot_ma_vs_mz(), uplot_n_mf_per_sample(), uplot_pca(), uplot_ratios(), uplot_reproducibility(), uplot_ri_vs_sample(), uplot_vk()

Examples

```
uplot_ms(pl = peaklist_demo, data_reduction = 0.1, plotly = TRUE, label = "file")
uplot_ms(pl = mf_data_demo, data_reduction = 1, plotly = FALSE, label = "file")
```

`uplot_n_mf_per_sample` *Number of Molecular Formulas per Sample Plot*

Description

Creates a bar plot showing how many molecular formulas were assigned per sample (`file_id`). The plot title contains the mean and standard deviation of assigned molecular formulas across samples. Optionally, the plot can be converted to an interactive Plotly plot or display the UltraMassExplorer logo.

Usage

```
uplot_n_mf_per_sample(
  df,
  col = "grey",
  logo = TRUE,
  width = 0.3,
  gg_size = 12,
  plotly = FALSE
)
```

Arguments

`df`	A data.table containing at least a `file_id` column.
`col`	Character. Fill color for the bars (default `"grey"`).
`logo`	Logical. If TRUE, adds a UME caption.
`width`	Numeric. Width of bars (default `0.3`).
`gg_size`	Base text size for `theme_uplots()`. Default = 12.
`plotly`	Logical. If TRUE, return interactive plotly object.

Details

Number of Molecular Formulas per Sample / File

Value

A `ggplot` object, or a `plotly` object if `plotly = TRUE`.

See Also

Other uplots: `uplot_cluster()`, `uplot_cvm()`, `uplot_dbe_minus_o_freq()`, `uplot_dbe_vs_c()`, `uplot_dbe_vs_ma()`, `uplot_dbe_vs_o()`, `uplot_freq_ma()`, `uplot_freq_vs_ppm()`, `uplot_hc_vs_m()`, `uplot_heteroatoms()`, `uplot_isotope_precision()`, `uplot_kmd()`, `uplot_lcms()`, `uplot_ma_vs_mz()`, `uplot_ms()`, `uplot_pca()`, `uplot_ratios()`, `uplot_reproducibility()`, `uplot_ri_vs_sample()`, `uplot_vk()`

Examples

```
uplot_n_mf_per_sample(mf_data_demo)
```

`uplot_pca` *Plot PCA Results*

Description

Performs Principal Component Analysis (PCA) on molecular formula intensity or log-ratio data and visualizes the results as a PCA score plot and a Van Krevelen plot colored by PC1 loadings.

Usage

```
uplot_pca(
  mfd,
  grp,
  int_col = "norm_int",
  scale_pca = NULL,
  fill = NULL,
  palname = "viridis",
  col_bar = TRUE,
  ...
)
```

Arguments

`mfd`	data.table with molecular formula data as derived from `ume::assign_formulas`. Column names of elements/isotopes must match names in the `isotope` column of `ume::masses`; values are integers representing counts per formula.
`grp`	Character. Name of the column used to define rows/samples in the PCA matrix.
`int_col`	Character. Name of the value column used for PCA (default = `"norm_int"`).
`scale_pca`	Logical or `NULL`. If `NULL`, PCA scaling is disabled for `int_col = "logratio_value"` and enabled otherwise.
`fill`	Numeric or `NULL`. Fill value used when converting to a wide matrix. If `NULL`, missing combinations are filled with `NA` for `int_col = "logratio_value"` and with `0` otherwise.
`palname`	Character. Name of the color palette passed to `uplot_vk()` (default = `"viridis"`).
`col_bar`	Logical. If `TRUE`, show the color bar in the Van Krevelen plot.
`...`	Additional arguments passed to `uplot_vk()`.

Details

Principal Component Analysis (PCA) Plotting

The PCA is performed on a wide matrix with one row per group defined by `grp` and one column per molecular formula (`mf`). Values are aggregated using the mean if multiple values occur for the same combination of `grp` and `mf`.

Columns with zero variance are removed before PCA because they cannot be scaled. If `int_col = "logratio_value"`, PCA scaling is disabled by default because CLR/ALR values are already log-ratio transformed.

Value

A list containing:

pca The PCA model object returned by `stats::prcomp()`.

t_score A `data.table` with PCA scores for each group.

fig_vk A Van Krevelen plot colored by PC1 loadings.

fig_pca A PCA score plot of PC1 versus PC2.

mfd The input molecular formula data augmented with PC1/PC2 scores and PC1/PC2 loadings.

Note

For `logratio_value`, `scale_pca = FALSE` is used by default. For normalized intensities such as `norm_int`, `scale_pca = TRUE` is used by default.

See Also

`uplot_vk()`, `calc_logratio()`

Other uplots: `uplot_cluster()`, `uplot_cvm()`, `uplot_dbe_minus_o_freq()`, `uplot_dbe_vs_c()`, `uplot_dbe_vs_ma()`, `uplot_dbe_vs_o()`, `uplot_freq_ma()`, `uplot_freq_vs_ppm()`, `uplot_hc_vs_m()`, `uplot_heteroatoms()`, `uplot_isotope_precision()`, `uplot_kmd()`, `uplot_lcms()`, `uplot_ma_vs_mz()`, `uplot_ms()`, `uplot_n_mf_per_sample()`, `uplot_ratios()`, `uplot_reproducibility()`, `uplot_ri_vs_sample()`, `uplot_vk()`

Examples

```
res <- uplot_pca(
  mfd = mf_data_demo,
  grp = "file_id",
  int_col = "norm_int"
)

res$fig_pca
res$fig_vk

## Not run:
mfd_lr <- calc_logratio(
  mfd = mf_data_demo,
  method = "clr",
```

```
  zero = "min"
)

res_lr <- uplot_pca(
  mfd = mfd_lr,
  grp = "file_id",
  int_col = "logratio_value"
)

## End(Not run)
```

---

uplot_ppm_avg *Plot Median of Mass Accuracy per Sample (ppm)*

Description

This function generates a bar plot showing the median of mass accuracy (ppm) for each sample. It also provides the option to convert the plot into an interactive `plotly` object.

Usage

```
uplot_ppm_avg(df, cex.axis = 12, cex.lab = 15, plotly = FALSE, ...)
```

Arguments

`df`	A data frame containing the data. The columns `ppm` (ppm values) and `file_id` (sample identifiers) should be present in the data.
`cex.axis`	Numeric. Size of axis text (default is `1`).
`cex.lab`	Numeric. Size of axis labels (default is `1.4`).
`plotly`	Logical. If TRUE, return interactive plotly object.
`...`	Additional arguments passed to methods.

Value

A `ggplot` object or a `plotly` object depending on the `plotly` argument.

uplot_ratios *Molecular Formula Ratio Plot (Sample vs Control)*

Description

Computes the intensity ratio between a sample and a control group and visualizes it in a Van Krevelen diagram. Optionally highlights unique molecular formulas and plots the ratio distribution.

Usage

```
uplot_ratios(
  mfd,
  upper = 90,
  lower = -90,
  grp = "file_id",
  int_col = "norm_int",
  control,
  sample,
  uniques = FALSE,
  palname = "ratios",
  distrib = TRUE,
  main = NA,
  ...
)
```

Arguments

mfd	A data.table containing at least columns: mf, oc, hc, grouping variable grp, and intensity column int_col.
upper, lower	Ratio filtering limits (default 90 / -90)
grp	Column defining sample/control grouping
int_col	Intensity column to use
control	Character: control group name
sample	Character: sample group name
uniques	Logical: highlight uniquely present formulas
palname	Color palette for the Van Krevelen projection
distrib	Logical: include ratio distribution plot
main	Optional main title
...	Additional arguments passed to methods.

Details

Ratio Plot in Van Krevelen Space

Value

A list with:

- `ratio_table`
- `plot_ratio_vk`
- `plot_ratio_distr`

See Also

Other uplots: `uplot_cluster()`, `uplot_cvm()`, `uplot_dbe_minus_o_freq()`, `uplot_dbe_vs_c()`, `uplot_dbe_vs_ma()`, `uplot_dbe_vs_o()`, `uplot_freq_ma()`, `uplot_freq_vs_ppm()`, `uplot_hc_vs_m()`, `uplot_heteroatoms()`, `uplot_isotope_precision()`, `uplot_kmd()`, `uplot_lcms()`, `uplot_ma_vs_mz()`, `uplot_ms()`, `uplot_n_mf_per_sample()`, `uplot_pca()`, `uplot_reproducibility()`, `uplot_ri_vs_sample()`, `uplot_vk()`

Examples

```
out <- uplot_ratios(
  mfd     = mf_data_demo,
  grp     = "file",
  control = "Nsea_a",
  sample  = "Fjord 01a"
)
out$plot_ratio_vk
out$plot_ratio_distr
```

`uplot_reproducibility` *Check Reproducibility of Sample Analyses*

Description

Computes reproducibility of sample analyses based on the relative intensity column (`norm_int`). For each molecular formula (`mf`), the function calculates:

- number of occurrences (`N`)
- median relative intensity (`ri`)
- relative standard deviation (RSD = sd/median × 100)

It also bins `ri` into integer bins and calculates the median RSD per bin.

The function returns:

- processed tables
- two **ggplot2** objects:
  - intensity vs RSD scatter plot
  - binned median RSD plot

Usage

```
uplot_reproducibility(mfd, ri = "norm_int")
```

Arguments

mfd	data.table with molecular formula data as derived from ume::assign_formulas. Column names of elements/isotopes must match names in the isotope column of ume::masses; values are integers representing counts per formula.
ri	Character string: name of the intensity column. Default: "norm_int".

Value

A list containing:

tmp Summary table by molecular formula

tmp2 Binned median RSD table

plot_rsd Scatter plot of RI vs RSD (ggplot2)

plot_bins Median RSD per bin (ggplot2)

See Also

Other uplots: uplot_cluster(), uplot_cvm(), uplot_dbe_minus_o_freq(), uplot_dbe_vs_c(), uplot_dbe_vs_ma(), uplot_dbe_vs_o(), uplot_freq_ma(), uplot_freq_vs_ppm(), uplot_hc_vs_m(), uplot_heteroatoms(), uplot_isotope_precision(), uplot_kmd(), uplot_lcms(), uplot_ma_vs_mz(), uplot_ms(), uplot_n_mf_per_sample(), uplot_pca(), uplot_ratios(), uplot_ri_vs_sample(), uplot_vk()

Examples

```
out <- uplot_reproducibility(mf_data_demo, ri = "norm_int")
out$plot_rsd
out$plot_bins
```

---

uplot_ri_vs_sample *Median Relative Intensity per Sample*

Description

Creates a bar plot showing the median intensity for each sample or spectrum, grouped by the column specified in grp.

Groups are ordered by their median intensity. The dataset-wide median and standard deviation of the selected intensity variable are included in the plot title.

Plot styling, axis-label orientation, branding, and optional Plotly conversion are handled by uplot_wrapper().

Usage

```
uplot_ri_vs_sample(
  mfd,
  int_col = "norm_int",
  grp = "file_id",
  col = "grey",
  width = 0.3,
  ...
)
```

Arguments

mfd	data.table with molecular formula data as derived from ume::assign_formulas. Column names of elements/isotopes must match names in the isotope column of ume::masses; values are integers representing counts per formula.
int_col	Character string specifying the column containing the intensity values. Default is "norm_int".
grp	Character string specifying the sample or spectrum grouping column. Default is "file_id".
col	Character string specifying the fill colour of the bars. Default is "grey".
width	Numeric value specifying the bar width. Default is 0.3.
...	Arguments passed on to uplot_wrapper

title Optional character string. Plot title. Set title_show = FALSE to suppress the title entirely.

title_show Logical. Display the plot title? Default: TRUE.

title_size Numeric. Font size of the title (points).

ume_logo Logical. Add the UME package logo? Default: TRUE.

ume_label Logical. Add the vertical UME branding label? Default: TRUE.

map_labels A list specifying which variables should get mapped to human-readable labels using uplots_map_labels(). Expected elements: x, y, colour, fill, size. May be NULL to suppress mapping.

p A ggplot object created by a uplot_* function.

palname Colour palette name passed to f_colorz().

col_bar Logical. Show colour bar?

colour_scale Character. Controls how the colour aesthetic is handled. One of:

- "auto" (default): Automatically chooses a continuous scale for numeric variables and a discrete scale for categorical variables.
- "continuous": Forces a continuous colour scale.
- "discrete": Forces a discrete colour scale.
- "none": Do not modify the colour scale.

x_npc_logo,y_npc_logo NPC coordinates for logo placement.

x_npc_label,y_npc_label NPC coordinates for label placement.

`x_label_angle` Controls the orientation of labels on a discrete x-axis. Use `"auto"` to choose the angle based on the number and length of labels, or supply a numeric angle between 0 and 90 degrees. Default is `"auto"`.

`interactive` Logical. Return plotly object?

`plotly` Logical. If TRUE, return interactive plotly object.

`size_dots` Numeric. Size of the dots in the plot (default = 0.5).

`text_size` Numeric font size (in points).

Details

Plot Median Relative Intensity per Sample

Value

A `ggplot` object, or a `plotly` object when `interactive = TRUE` is passed through `...`.

See Also

Other uplots: `uplot_cluster()`, `uplot_cvm()`, `uplot_dbe_minus_o_freq()`, `uplot_dbe_vs_c()`, `uplot_dbe_vs_ma()`, `uplot_dbe_vs_o()`, `uplot_freq_ma()`, `uplot_freq_vs_ppm()`, `uplot_hc_vs_m()`, `uplot_heteroatoms()`, `uplot_isotope_precision()`, `uplot_kmd()`, `uplot_lcms()`, `uplot_ma_vs_mz()`, `uplot_ms()`, `uplot_n_mf_per_sample()`, `uplot_pca()`, `uplot_ratios()`, `uplot_reproducibility()`, `uplot_vk()`

Examples

```
uplot_ri_vs_sample(
  mfd = mf_data_demo,
  int_col = "norm_int",
  grp = "file"
)
```

`uplot_vk` *uplot_vk*

Description

Creates a Van Krevelen diagram (H/C vs O/C).

Usage

```
uplot_vk(
  mfd,
  z_var = "norm_int",
  title = "Van Krevelen Diagram",
  projection = TRUE,
  palname = "viridis",
```

```
  median_vK = TRUE,
  col_median = "white",
  ai = TRUE,
  size_dots = 3,
  col_bar = TRUE,
  tf = FALSE,
  ...
)
```

Arguments

mfd	data.table with molecular formula data as derived from ume::assign_formulas. Column names of elements/isotopes must match names in the isotope column of ume::masses; values are integers representing counts per formula.
z_var	Character. Column name for variable used for color-coding. Content of column should be numeric.
title	Character. Plot title. Default "Van Krevelen Diagram".
projection	If TRUE, median z-values per (oc, hc) are used.
palname	Character. Name of the palette. Available palettes: "black", "redblue", "ratios", "rainbow", "awi", "viridis", "inferno", "terrain.colors", "gray".
median_vK	Add median VK point.
col_median	Color of the marker for the median O/C and H/C value (Default = "white")
ai	Add aromaticity index threshold lines.
size_dots	Numeric. Size of the dots in the plot (default = 0.5).
col_bar	Logical. If TRUE, adds a color legend (default is TRUE).
tf	Logical. If TRUE, applies a transformation to the color scale (default is FALSE).
...	Arguments passed on to f_colorz z Numeric vector. Values whose colors should be computed. col_num Integer. Number of colors in the palette (default: 100). verbose logical; if TRUE, show progress messages.

Details

Plot Van Krevelen Diagram

Value

ggplot or plotly object

See Also

Other uplots: uplot_cluster(), uplot_cvm(), uplot_dbe_minus_o_freq(), uplot_dbe_vs_c(), uplot_dbe_vs_ma(), uplot_dbe_vs_o(), uplot_freq_ma(), uplot_freq_vs_ppm(), uplot_hc_vs_m(), uplot_heteroatoms(), uplot_isotope_precision(), uplot_kmd(), uplot_lcms(), uplot_ma_vs_mz(), uplot_ms(), uplot_n_mf_per_sample(), uplot_pca(), uplot_ratios(), uplot_reproducibility(), uplot_ri_vs_sample()

Examples

```
uplot_vk(mf_data_demo, z_var = "norm_int")

# Custom title, e.g. used internally by uplot_ratios()
uplot_vk(mf_data_demo, z_var = "norm_int", title = "Custom VK title")
```

`ustats_outlier` *Outlier detection using multiple statistical tests*

Description

This function computes an `out_score` for each value in a selected column. The score increases when a value is flagged as an outlier by one or more tests: IQR test, quantile cutoffs, and Hampel filter.

Usage

```
ustats_outlier(dt, check_col = "ppm", verbose = FALSE, ...)
```

Arguments

`dt`	A `data.table` or `data.frame`.
`check_col`	A character string naming the column to test for outliers.
`verbose`	Logical; print summary statistics when TRUE.
`...`	Additional arguments passed to methods.

Value

A `data.table` containing new columns: `out_score`, `out_box`, `out_quantile`, and `out_hampel`.

Examples

```
ustats_outlier(mf_data_demo, check_col = "ppm")
```

zero_filling *zero_filling*

Description

Replaces zero values in a numeric sample-by-feature matrix before log-ratio analysis or other multivariate statistics.

Usage

```
zero_filling(
  x,
  method = c("min", "median", "CZM", "GBM", "none"),
  factor = 0.65,
  reclose = TRUE,
  remove_empty = TRUE,
  z_warning = 1
)
```

Arguments

x	Numeric matrix or data.frame. Rows are samples and columns are features.
method	Character. Zero-replacement method. One of "min", "median", "CZM", "GBM", or "none".
factor	Numeric. Multiplication factor for method "min". Default is 0.65.
reclose	Logical. If TRUE, rows are normalized to unit sum after zero replacement. Default is TRUE.
remove_empty	Logical. If TRUE, columns containing only zeros are removed before zero replacement. Default is TRUE.
z_warning	Numeric. Passed to zCompositions::cmultRepl() as z.warning. Default is 1.

Details

Replace zeros in compositional feature matrices

The input is expected to contain samples in rows and features, such as molecular formulas, in columns. This is the same structure as returned by as_ume_matrix().

Zero values are problematic for log-ratio analysis because logarithms of zero are undefined. This function provides simple UME-internal zero replacement methods as well as optional access to methods from the zCompositions package.

The default method "min" replaces zeros in each feature column by factor times the smallest positive value observed in that feature. With the default factor = 0.65, zeros are replaced by 65 percent of the feature-specific minimum detected value.

Method "median" replaces zeros in each feature column by the feature-specific median of positive values.

Methods "CZM" and "GBM" use zCompositions::cmultRepl() and require the suggested package zCompositions.

If reclose = TRUE, rows are normalized to unit sum after zero replacement. This is usually appropriate for compositional data analysis.

Value

A numeric matrix with zero values replaced. Removed feature names are stored in the attribute "removed_features". The chosen zero-replacement method is stored in the attribute "zero_method".

Examples

```
mat <- as_ume_matrix(
  mfd = mf_data_demo,
  grp = "file_id",
  value_col = "i_magnitude"
)

mat_filled <- zero_filling(mat, method = "min")
```

Index

∗ **Clean data output**
remove_id_columns, 65
remove_unknown_columns, 65
∗ **Formula assignment**
add_known_mf, 4
calc_eval_params, 15
eval_isotopes, 47
ume_assign_formulas, 67
∗ **Formula subsetting**
filter_int, 48
filter_mass_accuracy, 49
filter_mf_data, 50
remove_blanks, 62
subset_known_mf, 66
ume_assign_formulas, 67
ume_filter_formulas, 69
∗ **ID**
remove_id_columns, 65
∗ **Remove**
remove_id_columns, 65
remove_unknown_columns, 65
∗ **calculations**
calc_data_summary, 12
calc_dbe, 14
calc_eval_params, 15
calc_exact_mass, 17
calc_ideg, 18
calc_ma, 26
calc_neutral_mass, 27
calc_nm, 28
calc_norm_int, 29
calc_number_assignment, 31
∗ **check ume objects**
as_mfd, 7
as_peaklist, 8
∗ **chemistry**
check_neutral_mf, 36
convert_data_table_to_molecular_formulas, 40
convert_molecular_formula_to_data_table, 42
∗ **columns.**
remove_id_columns, 65
∗ **columns**
remove_unknown_columns, 65
∗ **contain**
remove_unknown_columns, 65
∗ **datasets**
known_mf, 56
lib_demo, 57
masses, 58
mf_data_demo, 59
peaklist_demo, 61
ume_name_dictionary, 71
∗ **exclusively**
remove_unknown_columns, 65
∗ **filter_int**
ume_filter_formulas, 69
∗ **helper**
classify_files, 38
∗ **indeces**
calc_iterr, 22
∗ **index calculations**
calc_iterr, 22
∗ **isotopes**
calc_isotope_pattern, 19
create_isotope_expanded_table, 43
eval_isotopes, 47
uplot_isotope_precision, 89
∗ **misc**
add_known_mf, 4
calc_ma, 26
calc_ma_abs, 27
calc_number_assignment, 31
filter_int, 48
filter_mass_accuracy, 49
filter_mf_data, 50
get_isotope_info, 54

order_columns, 61
remove_blanks, 62
subset_known_mf, 66
ume_assign_formulas, 67
ustats_outlier, 107
∗ **missing**
remove_unknown_columns, 65
∗ **molecular formula functions**
check_neutral_mf, 36
convert_data_table_to_molecular_formulas, 40
convert_molecular_formula_to_data_table, 42
∗ **molecular-formula**
calc_isotope_pattern, 19
check_neutral_mf, 36
convert_data_table_to_molecular_formulas, 40
convert_molecular_formula_to_data_table, 42
∗ **that**
remove_unknown_columns, 65
∗ **tools**
add_missing_element_columns, 6
order_columns, 61
∗ **ume data**
known_mf, 56
lib_demo, 57
masses, 58
mf_data_demo, 59
peaklist_demo, 61
ume_name_dictionary, 71
∗ **ume wrapper**
ume_assign_formulas, 67
ume_filter_formulas, 69
∗ **uplots**
uplot_cluster, 72
uplot_cvm, 74
uplot_dbe_minus_o_freq, 76
uplot_dbe_vs_c, 78
uplot_dbe_vs_ma, 80
uplot_dbe_vs_o, 81
uplot_freq_ma, 84
uplot_freq_vs_ppm, 85
uplot_hc_vs_m, 87
uplot_heteroatoms, 88
uplot_isotope_precision, 89
uplot_kmd, 91
uplot_lcms, 93
uplot_ma_vs_mz, 94
uplot_ms, 95
uplot_n_mf_per_sample, 97
uplot_pca, 98
uplot_ratios, 101
uplot_reproducibility, 102
uplot_ri_vs_sample, 103
uplot_vk, 105
∗ **utils**
calc_number_assignment, 31
∗ **values.**
remove_unknown_columns, 65

add_known_mf, 4, *68*
add_known_mf(), *16*, *47*, *69*
add_missing_element_columns, 6
add_missing_element_columns(), *61*
as_mfd, 7
as_mfd(), *9*
as_peaklist, 8
as_peaklist(), *8*
as_ume_matrix, 9
as_ume_matrix(), *25*, *73*, *108*
assign_formulas, 10, *68*

calc_data_summary, 12
calc_data_summary(), *15–17*, *19*, *26*, *28–31*
calc_dbe, 14
calc_dbe(), *13*, *16, 17*, *19*, *26*, *28–31*
calc_eval_params, 15, *68*
calc_eval_params(), *6*, *13*, *15*, *17*, *19*, *26*, *28–31*, *47*, *69*
calc_exact_mass, 17
calc_exact_mass(), *13*, *15, 16*, *19*, *26*, *28–31*
calc_ideg, 18
calc_ideg(), *13*, *15–17*, *26*, *28–31*
calc_isotope_pattern, 19
calc_isotope_pattern(), *44*, *47*, *91*
calc_iterr, 22
calc_logratio, 24
calc_logratio(), *99*
calc_ma, 26
calc_ma(), *13*, *15–17*, *19*, *28–31*
calc_ma_abs, *11*, 27, *68*
calc_ma_abs(), *10*
calc_neutral_mass, *11*, 27, *68*
calc_neutral_mass(), *10*, *13*, *15–17*, *19*, *26*, *29–31*

calc_nm, 28
calc_nm(), *13*, *15–17*, *19*, *26*, *28*, *30, 31*
calc_norm_int, 29, *48*, *68, 69*
calc_norm_int(), *13*, *15–17*, *19*, *26*, *28, 29*, *31*
calc_number_assignment, 31
calc_number_assignment(), *13*, *15–17*, *19*, *26*, *28–30*
calc_number_occurrence(), *13*, *15–17*, *19*, *26*, *28–31*
calc_pielou_evenness, 32
calc_recalibrate_ms(), *13*, *15–17*, *19*, *26*, *28–31*
calc_shannon_index, 33
calc_simpson_index, 34
calc_time_flattened_mfd, 35
check_formula_library(), *6*, *8, 9*, *16*, *47*, *69*
check_mfd(), *8, 9*
check_neutral_mf, 36
check_neutral_mf(), *41–43*
classify_files, 38
color.palette, 39
convert_data_table_to_molecular_formulas, 40
convert_data_table_to_molecular_formulas(), *37*, *43*
convert_molecular_formula_to_data_table, 42
convert_molecular_formula_to_data_table(), *37*, *41*, *43*
create_isotope_expanded_table, 43
create_isotope_expanded_table(), *21*, *47*, *91*
create_time_flattened_mfd *(*calc_time_flattened_mfd*)*, 35
create_ume_formula_library, 44

download_library, 46

eval_isotopes, 47, *68*
eval_isotopes(), *6*, *16*, *21*, *44*, *69*, *91*

f_colorz, *106*
filter_int, 48, *69*
filter_int(), *49*, *51*, *63*, *66*, *69*, *71*
filter_mass_accuracy, 49
filter_mass_accuracy(), *48*, *51*, *63*, *66*, *69*, *71*
filter_mf_data, 50, *69*
filter_mf_data(), *48, 49*, *63*, *66*, *69*, *71*
filter_multi_assignments, 52

get_isotope_info, 54
get_isotope_info(), *41*

inchi_to_mf, 55
is_ume_peaklist, 56

known_mf, 56, *58*, *60*, *62*, *72*

lib_demo, 57, *57, 58*, *60*, *62*, *72*

masses, *54*, *57*, 58, *58*, *60*, *62*, *72*
mf_data_demo, *57, 58*, 59, *62*, *72*

order_columns, 61
order_columns(), *7*

peaklist_demo, *57, 58*, *60*, 61, *72*

remove_blanks, 62, *69*
remove_blanks(), *48, 49*, *51*, *66*, *69*, *71*
remove_empty_columns, 64
remove_id_columns, 65
remove_id_columns(), *65*
remove_unknown_columns, 65
remove_unknown_columns(), *65*

stats::hclust(), *72*
stats::prcomp(), *99*
subset_known_mf, 66, *69*
subset_known_mf(), *48, 49*, *51*, *63*, *69*, *71*

theme_uplots, 67

ume_assign_formulas, 67
ume_assign_formulas(), *6*, *16*, *47–49*, *51*, *63*, *66*, *71*
ume_filter_formulas, 69
ume_filter_formulas(), *48, 49*, *51*, *63*, *66*, *69*
ume_name_dictionary, *57, 58*, *60*, *62*, 71
uplot_cluster, 72
uplot_cluster(), *75*, *77*, *79*, *81*, *83*, *85, 86*, *88, 89*, *91*, *93–97*, *99*, *102, 103*, *105, 106*
uplot_cvm, 74
uplot_cvm(), *73*, *77*, *79*, *81*, *83*, *85, 86*, *88, 89*, *91*, *93–97*, *99*, *102, 103*, *105, 106*
uplot_dbe_minus_o_freq, 76

uplot_dbe_minus_o_freq(), *73*, *75*, *79*, *81*, *83*, *85*, *86*, *88*, *89*, *91*, *93–97*, *99*, *102*, *103*, *105*, *106*
uplot_dbe_vs_c, 78
uplot_dbe_vs_c(), *73*, *75*, *77*, *81*, *83*, *85*, *86*, *88*, *89*, *91*, *93–97*, *99*, *102*, *103*, *105*, *106*
uplot_dbe_vs_ma, 80
uplot_dbe_vs_ma(), *73*, *75*, *77*, *79*, *83*, *85*, *86*, *88*, *89*, *91*, *93–97*, *99*, *102*, *103*, *105*, *106*
uplot_dbe_vs_o, 81
uplot_dbe_vs_o(), *73*, *75*, *77*, *79*, *81*, *85*, *86*, *88*, *89*, *91*, *93–97*, *99*, *102*, *103*, *105*, *106*
uplot_freq, 83
uplot_freq_ma, 84
uplot_freq_ma(), *73*, *75*, *77*, *79*, *81*, *83*, *86*, *88*, *89*, *91*, *93–97*, *99*, *102*, *103*, *105*, *106*
uplot_freq_vs_ppm, 85
uplot_freq_vs_ppm(), *73*, *75*, *77*, *79*, *81*, *83*, *85*, *88*, *89*, *91*, *93–97*, *99*, *102*, *103*, *105*, *106*
uplot_hc_vs_m, 87
uplot_hc_vs_m(), *73*, *75*, *77*, *79*, *81*, *83*, *85*, *86*, *89*, *91*, *93–97*, *99*, *102*, *103*, *105*, *106*
uplot_heteroatoms, 88
uplot_heteroatoms(), *73*, *75*, *77*, *79*, *81*, *83*, *85*, *86*, *88*, *91*, *93–97*, *99*, *102*, *103*, *105*, *106*
uplot_isotope_precision, 89
uplot_isotope_precision(), *21*, *44*, *47*, *73*, *75*, *77*, *79*, *81*, *83*, *85*, *86*, *88*, *89*, *93–97*, *99*, *102*, *103*, *105*, *106*
uplot_kmd, 91
uplot_kmd(), *73*, *75*, *77*, *79*, *81*, *83*, *85*, *86*, *88*, *89*, *91*, *94–97*, *99*, *102*, *103*, *105*, *106*
uplot_lcms, 93
uplot_lcms(), *73*, *75*, *77*, *79*, *81*, *83*, *85*, *86*, *88*, *89*, *91*, *93*, *95–97*, *99*, *102*, *103*, *105*, *106*
uplot_ma_vs_mz, 94
uplot_ma_vs_mz(), *73*, *75*, *77*, *79*, *81*, *83*, *85*, *86*, *88*, *89*, *91*, *93*, *94*, *96*, *97*, *99*, *102*, *103*, *105*, *106*
uplot_ms, 95
uplot_ms(), *73*, *75*, *77*, *79*, *81*, *83*, *85*, *86*, *88*, *89*, *91*, *93–95*, *97*, *99*, *102*, *103*, *105*, *106*
uplot_n_mf_per_sample, 97
uplot_n_mf_per_sample(), *73*, *75*, *77*, *79*, *81*, *83*, *85*, *86*, *88*, *89*, *91*, *93–96*, *99*, *102*, *103*, *105*, *106*
uplot_pca, 98
uplot_pca(), *73*, *75*, *77*, *79*, *81*, *83*, *85*, *86*, *88*, *89*, *91*, *93–97*, *102*, *103*, *105*, *106*
uplot_ppm_avg, 100
uplot_ratios, 101
uplot_ratios(), *73*, *75*, *77*, *79*, *81*, *83*, *85*, *86*, *88*, *89*, *91*, *93–97*, *99*, *103*, *105*, *106*
uplot_reproducibility, 102
uplot_reproducibility(), *73*, *75*, *77*, *79*, *81*, *83*, *85*, *86*, *88*, *89*, *91*, *93–97*, *99*, *102*, *105*, *106*
uplot_ri_vs_sample, 103
uplot_ri_vs_sample(), *73*, *75*, *77*, *79*, *81*, *83*, *85*, *86*, *88*, *89*, *91*, *93–97*, *99*, *102*, *103*, *106*
uplot_vk, 105
uplot_vk(), *73*, *75*, *77*, *79*, *81*, *83*, *85*, *86*, *88*, *89*, *91*, *93–97*, *99*, *102*, *103*, *105*
uplot_wrapper, *74*, *76*, *78*, *80*, *82*, *84*, *87*, *92*, *94*, *104*
ustats_outlier, 107

zero_filling, 108
zero_filling(), *24*, *25*