

Package ‘MetaboCoreUtils’

October 14, 2021

Title Core Utils for Metabolomics Data

Version 1.0.0

Description MetaboCoreUtils defines metabolomics-related core functionality provided as low-level functions to allow a data structure-independent usage across various R packages. This includes functions to calculate between ion (adduct) and compound mass-to-charge ratios and masses or functions to work with chemical formulas. The package provides also a set of adduct definitions and information on some commercially available internal standard mixes commonly used in MS experiments.

Depends R (>= 4.1)

Imports stringr, utils

Suggests BiocStyle, testthat, knitr, rmarkdown

License Artistic-2.0

LazyData no

VignetteBuilder knitr

BugReports <https://github.com/RforMassSpectrometry/MetaboCoreUtils/issues>

URL <https://github.com/RforMassSpectrometry/MetaboCoreUtils>

biocViews Infrastructure, Metabolomics, MassSpectrometry

Roxygen list(markdown=TRUE)

RoxygenNote 7.1.1

git_url <https://git.bioconductor.org/packages/MetaboCoreUtils>

git_branch RELEASE_3_13

git_last_commit 0c30b89

git_last_commit_date 2021-05-19

Date/Publication 2021-10-14

Author Johannes Rainer [aut, cre] (<<https://orcid.org/0000-0002-6977-7147>>),
Michael Witting [aut] (<<https://orcid.org/0000-0002-1462-4426>>)

Maintainer Johannes Rainer <Johannes.Rainer@eurac.edu>

R topics documented:

| | |
|------------------------------------|----|
| addElements | 2 |
| adductNames | 3 |
| containsElements | 3 |
| countElements | 4 |
| internalStandardMixNames | 5 |
| internalStandards | 5 |
| mass2mz | 6 |
| mz2mass | 7 |
| pasteElements | 8 |
| standardizeFormula | 9 |
| subtractElements | 10 |

| | |
|--------------|-----------|
| Index | 11 |
|--------------|-----------|

| | |
|-------------|----------------------------------|
| addElements | <i>Combine chemical formulae</i> |
|-------------|----------------------------------|

Description

addElements Add one chemical formula to another.

Usage

```
addElements(x, y = NA_character_)
```

Arguments

| | |
|---|---|
| x | character Vector with 1 or more chemical formulae to be added |
| y | character Vector with 1 or more chemical formulae to be added |

Value

character Resulting formula

Author(s)

Michael Witting

Examples

```
addElements("C6H12O6", "Na")
addElements("C6H12O6", c("Na", "H2O"))
```

| | |
|-------------|--|
| adductNames | <i>Retrieve names of supported adducts</i> |
|-------------|--|

Description

adductNames returns all supported adduct definitions that can be used by `mass2mz()` and `mz2mass()`.
adducts returns a `data.frame` with the adduct definitions.

Usage

```
adductNames(polarity = c("positive", "negative"))
```

```
adducts(polarity = c("positive", "negative"))
```

Arguments

polarity character(1) defining the ion mode, either "positive" or "negative".

Value

for adductNames: character vector with all valid adduct names for the selected ion mode. For
adducts: `data.frame` with the adduct definitions.

Author(s)

Michael Witting, Johannes Rainer

Examples

```
## retrieve names of adduct names in positive ion mode  
adductNames(polarity = "positive")
```

```
## retrieve names of adduct names in negative ion mode  
adductNames(polarity = "negative")
```

| | |
|------------------|---|
| containsElements | <i>Check if one formula is contained in another</i> |
|------------------|---|

Description

containsElements checks if one sum formula is contained in another.

Usage

```
containsElements(x, y)
```

Arguments

x character Single string with a chemical formula
y character Single string with a chemical formula that shall be contained in x

Value

logical TRUE if y is contained in x

Author(s)

Michael Witting

Examples

```
containsElements("C6H12O6", "H2O")  
containsElements("C6H12O6", "NH3")
```

countElements *Count elements in a chemical formula*

Description

countElements parses a string representing a chemical formula into a named vector of element counts.

Usage

```
countElements(x)
```

Arguments

x character(1) representing a chemical formula.

Value

integer with the element counts (names being elements).

Author(s)

Michael Witting

See Also

[pasteElements\(\)](#)

Examples

```
countElements("C6H12O6")  
countElements("C11H12N2O2")
```

`internalStandardMixNames`*Get names of internal standard mixes provided by the package*

Description

`internalStandardMixNames` returns available names of internal standard mixes provided by the `MetaboCoreUtils` package.

Usage

```
internalStandardMixNames()
```

Value

character names of available IS mixes

Author(s)

Michael Witting

Examples

```
internalStandardMixNames()
```

`internalStandards`*Get definitions for internal standards*

Description

`internalStandards` returns a table with metabolite standards available in commercial internal standard mixes. The returned data frame contains the following columns:

- "name": the name of the standard
- "formula_salt": chemical formula of the salt that was used to produce the standard mix
- "formula_metabolite": chemical formula of the metabolite in free form
- "smiles_salt": SMILES of the salt that was used to produced the standard mix
- "smiles_metabolite": SMILES of the metabolite in free form
- "mol_weight_salt": molecular (average) weight of the salt (can be used for calculation of molar concentration, etc.)
- "exact_mass_metabolite": exact mass of free metabolites
- "conc": concentration of the metabolite in ug/mL (of salt form)
- "mix": name of internal standard mix

Usage

```
internalStandards(mix = "QReSS")
```

Arguments

mix character(1) Name of the internal standard mix that shall be returned. One of [internalStandardMixNames\(\)](#).

Value

data.frame data on internal standards

Author(s)

Michael Witting

See Also

[internalStandardMixNames\(\)](#) for provided internal standard mixes.

Examples

```
internalStandards(mix = "QReSS")
internalStandards(mix = "UltimateSplashOne")
```

mass2mz

Calculate mass-to-charge ratio

Description

mass2mz calculates the m/z value from a neutral mass and an adduct definition.

Custom adduct definitions can be passed to the adduct parameter in form of a data.frame. This data.frame is expected to have columns "mass_add" and "mass_multi" defining the *additive* and *multiplicative* part of the calculation. See [adducts\(\)](#) for examples.

Usage

```
mass2mz(x, adduct = "[M+H]+")
```

Arguments

x numeric neutral mass for which the adduct m/z shall be calculated.

adduct either a character specifying the name(s) of the adduct(s) for which the m/z should be calculated or a data.frame with the adduct definition. See [adductNames\(\)](#) for supported adduct names and the description for more information on the expected format if a data.frame is provided.

Value

numeric matrix with same number of rows than elements in `x` and number of columns being equal to the length of `adduct` (adduct names are used as column names). Each column thus represents the m/z of `x` for each defined adduct.

Author(s)

Michael Witting, Johannes Rainer

See Also

[mz2mass\(\)](#) for the reverse calculation, [adductNames\(\)](#) for supported adduct definitions.

Examples

```
exact_mass <- c(100, 200, 250)
adduct <- "[M+H]+"
```

Calculate m/z of [M+H]⁺ adduct from neutral mass

```
mass2mz(exact_mass, adduct)
```

```
exact_mass <- 100
adduct <- "[M+Na]+"
```

Calculate m/z of [M+Na]⁺ adduct from neutral mass

```
mass2mz(exact_mass, adduct)
```

Calculate m/z of multiple adducts from neutral mass

```
mass2mz(exact_mass, adduct = adductNames())
```

Provide a custom adduct definition.

```
adds <- data.frame(mass_add = c(1, 2, 3), mass_multi = c(1, 2, 0.5))
rownames(adds) <- c("a", "b", "c")
mass2mz(c(100, 200), adds)
```

mz2mass

Calculate neutral mass

Description

`mz2mass` calculates the neutral mass from a given m/z value and adduct definition.

Custom adduct definitions can be passed to the `adduct` parameter in form of a `data.frame`. This `data.frame` is expected to have columns `"mass_add"` and `"mass_multi"` defining the *additive* and *multiplicative* part of the calculation. See [adducts\(\)](#) for examples.

Usage

```
mz2mass(x, adduct = "[M+H]+")
```

Arguments

| | |
|--------|---|
| x | numeric m/z value for which the neutral mass shall be calculated. |
| adduct | either a character specifying the name(s) of the adduct(s) for which the m/z should be calculated or a <code>data.frame</code> with the adduct definition. See adductNames() for supported adduct names and the description for more information on the expected format if a <code>data.frame</code> is provided. |

Value

numeric matrix with same number of rows than elements in x and number of columns being equal to the length of adduct (adduct names are used as column names. Each column thus represents the neutral mass of x for each defined adduct.

Author(s)

Michael Witting, Johannes Rainer

See Also

[mass2mz\(\)](#) for the reverse calculation, [adductNames\(\)](#) for supported adduct definitions.

Examples

```
ion_mass <- c(100, 200, 300)
adduct <- "[M+H]+"
```

Calculate m/z of [M+H]+ adduct from neutral mass

```
mz2mass(ion_mass, adduct)
```

```
ion_mass <- 100
adduct <- "[M+Na]+"
```

Calculate m/z of [M+Na]+ adduct from neutral mass

```
mz2mass(ion_mass, adduct)
```

Provide a custom adduct definition.

```
adds <- data.frame(mass_add = c(1, 2, 3), mass_multi = c(1, 2, 0.5))
rownames(adds) <- c("a", "b", "c")
mz2mass(c(100, 200), adds)
```

pasteElements

Create chemical formula from a named vector

Description

pasteElements creates a chemical formula from element counts (such as returned by [countElements\(\)](#)).

Usage

```
pasteElements(x)
```

Arguments

x integer with element counts, names being individual elements.

Value

character(1) with the chemical formula.

Author(s)

Michael Witting

See Also

[countElements\(\)](#)

Examples

```
elements <- c("C" = 6, "H" = 12, "O" = 6)
pasteElements(elements)
```

standardizeFormula *Standardize a chemical formula*

Description

standardizeFormula standardizes a supplied chemical chem_formula according to the Hill notation system.

Usage

```
standardizeFormula(x)
```

Arguments

x character Single string with the chemical formula to standardize.

Value

character Single string with the standardized chemical formula.

Author(s)

Michael Witting

See Also

[pasteElements\(\)](#) [countElements\(\)](#)

Examples

```
standardizeFormula("C606H12")
```

subtractElements *subtract two chemical formula*

Description

subtractElements subtracts one chemical formula from another.

Usage

```
subtractElements(x, y)
```

Arguments

| | | |
|---|-----------|---|
| x | character | Single string with chemical formula |
| y | character | Single or multiple strings with chemical formula that should be subtracted from x |

Value

character Resulting formula

Author(s)

Michael Witting

Examples

```
subtractElements("C6H12O6", "H2O")
```

```
subtractElements("C6H12O6", "NH3")
```

Index

addElement, 2
adductNames, 3
adductNames(), 6–8
adducts (adductNames), 3
adducts(), 6, 7

containsElements, 3
countElements, 4
countElements(), 8–10

internalStandardMixNames, 5
internalStandardMixNames(), 6
internalStandards, 5

mass2mz, 6
mass2mz(), 3, 8
mz2mass, 7
mz2mass(), 3, 7

pasteElements, 8
pasteElements(), 4, 10

standardizeFormula, 9
subtractElements, 10