Package 'IDSL.UFAx'

September 26, 2022

Type Package

Title Exhaustive Chemical Enumeration for United Formula Annotation

Version 1.6

Depends R (>= 4.0)

Imports IDSL.MXP (>= 1.4), xml2, RNetCDF, base64enc, IDSL.IPA (>= 2.1), IDSL.UFA (>= 1.5), stats, readxl, parallel, doParallel, foreach, RcppAlgos

Author Sadjad Fakouri-Baygi [cre, aut] (<https://orcid.org/0000-0002-6864-6911>), Dinesh Barupal [aut] (<https://orcid.org/0000-0002-9954-8628>)

Maintainer Sadjad Fakouri-Baygi <sadjad.fakouri-baygi@mssm.edu>

Description
A pipeline to annotate a number of peaks from the IDSL.IPA peaklists using an exhaustive chemical enumeration-based approach. This package can perform elemental composition calculations using the following 15 elements : C, B, Br, Cl, K, S, Se, Si, N, H, As, F, I, Na, O, and P.

License MIT + file LICENSE

URL https://ufa.idsl.me/enumerating-chemical-space/exhaustive-enumeration, https://github.com/idslme/idsl.ufax

BugReports https://github.com/idslme/idsl.ufax/issues

Encoding UTF-8

Archs i386, x64

NeedsCompilation no

Repository CRAN

Date/Publication 2022-09-26 11:40:02 UTC

R topics documented:

UFAx_molecular_formula_library_search
Searching Molecular Formula in a Library of Known Compounds

Description

This function searches for detected molecular formula ions in a library of known molecular formulas.

Usage

```
UFAx_molecular_formula_library_search(molecular_formula_ions, IonPathways, Elements,
MF_library, number_processing_threads = 1)
```

Arguments

molecular_formula_ions	A string vector of molecular formula ions.
IonPathways	A vector of ionization pathways. Pathways should be like [Coeff*M+ADD1-DED1+...] where "Coeff" should be an integer between 1-9 and ADD1 and DED1 may be ionization pathways. ex: 'IonPathways <- c("[M]+", "[M+H]+", "[2M-Cl]-", "[3M+CO2-H2O+Na-KO2+HCl-NH4]-")'
Elements	A vector string of the used elements.
MF_library	A library of molecular formulas generated using the 'molecular_formula_library_generator' module.
number_processing_threads	Number of processing threads for multi-threaded processing

Value

A matrix of matched compounds in the reference library.

UFAx_score_coefficient_corrector
Score Coefficient MolecularFormulaAnnotationTable Corrector

Description

This function updates ranking orders of the individual MolecularFormulaAnnotationTable when score coefficients change. This function creates files with similar names to the original files with "_updated" extensions.

Usage

```
UFAx_score_coefficient_corrector(annotated_molf_address, maxNEME,
Score_coeff, number_processing_threads = 1)
```

Arguments

annotated_molf_address	Address of the individual MolecularFormulaAnnotationTables (.Rdata)
maxNEME	Maximum value for Normalized Euclidean Mass Error (NEME) in mDa
Score_coeff	A vector of five numbers representing coefficients of the identification score
number_processing_threads	Number of processing threads for multi-threaded computations

Value

The MolecularFormulaAnnotationTable is saved with an "_updated" extension.

UFAx_workflow *UFAx Workflow*

Description

This function runs the exhaustive chemical enumeration part of the IDSL.UFAx pipeline.

Usage

```
UFAx_workflow(spreadsheet)
```

Arguments

spreadsheet	IDSL.UFAx parameter spreadsheet

Value

The MolecularFormulaAnnotationTable is saved in the assigned folder in the parameter spreadsheet.

Note

You should load the IDSL.UFA package to run the IDSL.UFAx functions.

Examples

```
library(IDSL.UFA) # You should load the IDSL.UFA package to run the IDSL.UFAx functions.
library(IDSL.UFAx)
s_path <- system.file("extdata", package = "IDSL.UFAx")
SSh1 <- paste0(s_path,"/UFAx_parameters.xlsx")
temp_wd <- tempdir() # update this address
temp_wd_zip <- paste0(temp_wd,"/003.mzML_UFAx_testfiles.zip")
spreadsheet <- readxl::read_xlsx(SSh1)
download.file(
  paste0("https://github.com/idslme/IDSL.UFAx/blob/main/UFAx_educational_files/",
         "003.mzML_UFAx_testfiles.zip?raw=true"), destfile = temp_wd_zip, mode = "wb")
```

```
unzip(temp_wd_zip, exdir = temp_wd)
spreadsheet[1, 4] <- temp_wd
spreadsheet[4, 4] <- temp_wd
spreadsheet[6, 4] <- temp_wd
spreadsheet[3, 4] <- "seq(1, 100, 1)" # peak IDs to process
UFAx_results <- UFAx_workflow(spreadsheet)
```

Index