

Package ‘IDSL.UFA’

October 5, 2022

Type Package

Title United Formula Annotation (UFA) for HRMS Data Processing

Version 1.6

Depends R (>= 4.0)

Imports IDSL.MXP (>= 1.4), xml2, RNetCDF, base64enc, IDSL.IPA (>= 2.1), stats, grid, readxl, parallel, doParallel, foreach, GA, ggplot2, gridExtra

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 peaklists from the IDSL.IPA package with molecular formula of a prioritized chemical space using an isotopic profile matching approach. The IDSL.UFA pipeline only requires MS1 for formula annotation.

License MIT + file LICENSE

URL <https://ufa.idsl.me>, <https://github.com/idslme/idsl.ufa>

BugReports <https://github.com/idslme/idsl.ufa/issues>

Encoding UTF-8

LazyData true

Archs i386, x64

NeedsCompilation no

Repository CRAN

Date/Publication 2022-10-04 23:30:19 UTC

R topics documented:

<code>aligned_molecular_formula_annotator</code>	2
<code>detect_formula_sets</code>	3
<code>element_sorter</code>	4
<code>extended_SENIOR_rule_check</code>	5

formula_adduct_calculator	5
formula_vector_generator	6
hill_molecular_formula_printer	7
identification_score	7
ionization_pathway_deconvoluter	8
isotopic_profile_calculator	9
isotopic_profile_molecular_formula_feeder	10
IUPAC_Isotopes	11
molecular_formulas_source_IPDB	12
molecular_formula_annotator	13
molecular_formula_library_generator	13
molecular_formula_library_search	14
monoisotopic_mass_calculator	15
score_coefficients_optimization	15
score_coefficient_evaluation	16
UFA_enumerated_chemical_space	16
UFA_enumerated_chemical_space_xlsxAnalyzer	17
UFA_locate_regex	17
UFA_logRecorder	18
UFA_profile_visualizer	18
UFA_profile_visualizer_xlsxAnalyzer	19
UFA_score_coefficient_corrector	19
UFA_score_coefficient_workflow	20
UFA_score_function_optimization_xlsxAnalyzer	20
UFA_workflow	21
UFA_xlsxAnalyzer	21
zero_score_function	22
Index	23

aligned_molecular_formula_annotator

Aligned Molecular Formula Annotator

Description

This function detects frequent molecular formulas across multiple samples on the aligned peak table matrix.

Usage

aligned_molecular_formula_annotator(PARAM)

Arguments

PARAM a parameter driven from the UFA_xlsxAnalyzer module.

detect_formula_sets *Organic Class Detection by Repeated Unit Patterns*

Description

This function sorts a vector of molecular formulas to aggregate organic compound classes with repeated/non-repeated substructure units. This function only works for molecular formulas with following elements: c("As", "Br", "Cl", "Na", "Se", "Si", "B", "C", "F", "H", "I", "K", "N", "O", "P", "S")

Usage

```
detect_formula_sets(molecular_formulas, ratio_delta_HBrClFI_C = 2,  
mixed.HBrClFI.allowed = FALSE, min_molecular_formula_class = 2,  
max_number_formula_class = 100, number_processing_threads = 1)
```

Arguments

molecular_formulas
a vector of molecular formulas

ratio_delta_HBrClFI_C
c(2, 1/2, 0). 2 to detect structures with linear carbon chains such as PFAS, lipids, chlorinated paraffins, etc. 1/2 to detect structures with cyclic chains such as PAHs. 0 to detect molecular formulas with a fixed structures but changing H/Br/Cl/F/I atoms similar to PCBs, PBDEs, etc.

mixed.HBrClFI.allowed
mixed.HBrClFI.allowed = c(TRUE, FALSE). Select 'FALSE' to detect halogenated-saturated compounds similar to PFOS or select 'TRUE' to detect mixed halogenated compounds with hydrogen.

min_molecular_formula_class
minimum number of molecular formulas in each class. This number should be greater than or equal to 2.

max_number_formula_class
maximum number of molecular formulas in each class

number_processing_threads
Number of processing threads for multi-threaded computations.

Value

A matrix of clustered classes of organic molecular formulas.

Examples

```
molecular_formulas <- c("C3F7O3S", "C4F9O3S", "C5F11O3S", "C6F9O3S", "C8F17O3S",  
"C9F19O3S", "C10F21O3S", "C7ClF14O4", "C10ClF20O4", "C11ClF22O4", "C11Cl2F21O4",  
"C12ClF24O4")  
##
```

```
ratio_delta_HBrClFI_C <- 2 # to aggregate polymeric classes
mixed.HBrClFI.allowed <- FALSE # To detect only halogen saturated classes
min_molecular_formula_class <- 2
max_number_formula_class <- 20
##
classes <- detect_formula_sets(molecular_formulas, ratio_delta_HBrClFI_C,
mixed.HBrClFI.allowed, min_molecular_formula_class, max_number_formula_class,
number_processing_threads = 1)
```

element_sorter

Element Sorter

Description

This function sorts 84 elements in the periodic table for molecular formula deconvolution and isotopic profile calculation.

Usage

```
element_sorter(ElementList = "all", ElementOrder = "alphabetical")
```

Arguments

ElementList	A string vector of elements needed for isotopic profile calculation. The default value for this parameter is a vector string of entire elements.
ElementOrder	ElementOrder = c("alphabetical", "same") where "alphabetical" should be used to sort the elements for elemental deconvolution (default value), "same" should be used to keep the input order.

Value

Elements	A string vector of elements (alphabetically sorted or unsorted)
massAbundanceList	A list of isotopic mass and abundance of elements.
Valence	A vector of electron valences.

Examples

```
EL_mass_abundance_val <- element_sorter()
```

extended_SENIOR_rule_check
extended SENIOR rule check

Description

This function checks whether a molecular formula follows the extended SENIOR rule.

Usage

```
extended_SENIOR_rule_check(mol_vec, valence_vec, ionization_correction = 0)
```

Arguments

mol_vec A vector of the deconvoluted molecular formula

valence_vec A vector of the valences from the molecular formula. Valences may be acquired from the 'IUPAC_Isotopes' data.

ionization_correction
A number to compensate for the ionization losses/gains. For example, '-1' for [M+H/K/Na] ionization pathways and '+1' for [M-H] ionization pathway.

Value

rule2 TRUE for when the molecular formula passes the rule and FALSE for when the molecular formula fails to pass the rule.

formula_adduct_calculator
Formula Adduct Calculator

Description

This function takes a formula and a vector of ionization pathways and returns the adduct formulas.

Usage

```
formula_adduct_calculator(molecular_formula, IonPathways)
```

Arguments

molecular_formula
molecular formula

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]-")'

Value

A vector of adduct formulas

Examples

```
molecular_formula = "C15H10O7"  
IonPathways = c("[M+]", "[M+H]", "[M+H2O+H]", "[M+Na]")  
Formula_adducts <- formula_adduct_calculator(molecular_formula, IonPathways)
```

formula_vector_generator

Molecular Formula Vector Generator

Description

This function convert a molecular formulas into a numerical vector

Usage

```
formula_vector_generator(molecular_formula, Elements, L_Elements = length(Elements))
```

Arguments

molecular_formula	molecular formula
Elements	a string vector of elements. This value must be driven from the 'element_sorter' function.
L_Elements	number of elements. To speed up loop calculations, consider calculating the number of elements outside of the loop.

Value

a numerical vector for the molecular formula. This function returns a vector of -Inf values when the molecular formula has elements not listed in the 'Elements' string vector.

Examples

```
molecular_formula <- "C12H2Br5Cl3O"  
Elements_molecular_formula <- c("C", "H", "O", "Br", "Cl")  
EL <- element_sorter(ElementList = Elements_molecular_formula)  
Elements <- EL[[1]]  
L_Elements <- length(Elements)  
mol_vec <- formula_vector_generator(molecular_formula, Elements, L_Elements)
```

```
hill_molecular_formula_printer
```

Print Hill Molecular Formula

Description

This function produces molecular formulas from a list numerical vectors in the Hill notation system

Usage

```
hill_molecular_formula_printer(Elements, MolVecMat, number_processing_threads = 1)
```

Arguments

Elements	A vector string of the used elements.
MolVecMat	A matrix of numerical vectors of molecular formulas in each row.
number_processing_threads	Number of processing threads for multi-threaded processing

Value

A vector of molecular formulas

Examples

```
Elements <- c("C", "H", "O", "N", "Br", "Cl")
MoleFormVec1 <- c(2, 6, 1, 0, 0, 0) # C2H6O
MoleFormVec2 <- c(8, 10, 2, 4, 0, 0) # C8H10N4O2
MoleFormVec3 <- c(12, 2, 1, 0, 5, 3) # C12H2Br5Cl3O
MolVecMat <- rbind(MoleFormVec1, MoleFormVec2, MoleFormVec3)
H_MolF <- hill_molecular_formula_printer(Elements, MolVecMat)
```

```
identification_score
```

Multiplicative Identification Score for the IDSL.UFA pipeline

Description

This function calculates the score values to rank candidate molecular formulas for a mass spectrometry-chromatography peak.

Usage

```
identification_score(Score_coefficients, N_isotopologues, PCS, RCS, NEME,
maxNEME, R13C_PL, R13C_IP)
```

Arguments

Score_coefficients	A vector of seven numbers equal or greater than 0
N_isotopologues	Number of isotopologues in the theoretical isotopic profiles.
PCS	PCS (per mille)
RCS	RCS (percentage)
NEME	NEME (mDa)
maxNEME	maximum NEME (mDa)
R13C_PL	R13C of the peak from IDSL.IPA peaklists
R13C_IP	R13C from theoretical isotopic profiles

ionization_pathway_deconvoluter

Ionization Pathway Deconvoluter

Description

This function deconvolutes ionization pathways into a coefficient and a numerical vector to simplify prediction ionization pathways.

Usage

```
ionization_pathway_deconvoluter(IonPathways, Elements)
```

Arguments

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

Value

A list of adduct calculation values for each ionization pathway.

Examples

```
Elements <- element_sorter()[["Elements"]]
IonPathways <- c("[M]+", "[M+H]+", "[2M-Cl]-", "[3M+CO2-H2O+Na-KO2+HCl-NH4]-")
Ion_DC <- ionization_pathway_deconvoluter(IonPathways, Elements)
```

isotopic_profile_calculator
Isotopic Profile Calculator

Description

This function was designed to calculate isotopic profile distributions for small molecules with masses ≤ 1200 Da. Details of the equations used in this function are available in the reference[1]. In this function, neighboring isotopologues are merged using the satellite clustering merging (SCM) method described in the reference[2].

Usage

```
isotopic_profile_calculator(MoleFormVec, Elements_mass_abundance, peak_spacing,  
intensity_cutoff, UFA_IP_memeory_variables = c(1e30, 1e-12, 10))
```

Arguments

MoleFormVec A numerical vector of the molecular formula
Elements_mass_abundance A list of isotopic mass and abundance of elements obtained from the ‘element_sorter’ function
peak_spacing A maximum space between two isotopologues in Da
intensity_cutoff A minimum intensity threshold for isotopic profiles in percentage
UFA_IP_memeory_variables A vector of three variables. Default values are c(1e30, 1e-12, 10). Memory may be an issue when the entire isotopologues are calculated; therefore, UFA_IP_memeory_variables[1] is used to adjust memory usage. UFA_IP_memeory_variables[2] indicates the minimum relative abundance (RA calculated by eq(1) in the reference [1]) of an isotopologue to include in the isotopic profile calculations. UFA_IP_memeory_variables[3] is the maximum elapsed time to calculate the isotopic profile on the ‘setTimeLimit’ function of base R.

Value

A matrix of isotopic profile. The first and second column represents the mass and intensity profiles, respectively.

References

- [1] Fakouri Baygi, S., Crimmins, B.S., Hopke, P.K. Holsen, T.M. (2016). Comprehensive emerging chemical discovery: novel polyfluorinated compounds in Lake Michigan trout. *Environmental Science and Technology*, 50(17), 9460-9468, doi: [10.1021/acs.est.6b01349](https://doi.org/10.1021/acs.est.6b01349).
- [2] Fakouri Baygi, S., Fernando, S., Hopke, P.K., Holsen, T.M. and Crimmins, B.S. (2019). Automated Isotopic Profile Deconvolution for High Resolution Mass Spectrometric Data (APGC-QToF) from Biological Matrices. *Analytical chemistry*, 91(24), 15509-15517, doi: [10.1021/acs.analchem.9b03335](https://doi.org/10.1021/acs.analchem.9b03335).

See Also

<https://ipc.idsl.me/>

Examples

```
EL <- element_sorter()
Elements <- EL[[1]]
Elements_mass_abundance <- EL[[2]]
peak_spacing <- 0.005 # mDa
intensity_cutoff <- 1 # (in percentage)
MoleFormVec <- formula_vector_generator("C8H10N4O2", Elements)
IP <- isotopic_profile_calculator(MoleFormVec, Elements_mass_abundance,
                                peak_spacing, intensity_cutoff)
```

isotopic_profile_molecular_formula_feeder

Isotopic Profile Molecular Formula Feeder

Description

A function to calculate IPDBs from a vector of molecular formulas

Usage

```
isotopic_profile_molecular_formula_feeder(molecular_formula, peak_spacing = 0,
intensity_cutoff_str = 1, UFA_IP_memory_variables = c(1e30, 1e-12, 10),
IonPathways = "[M]+", number_processing_threads = 1)
```

Arguments

molecular_formula
A vector string of molecular formulas

peak_spacing
A maximum space between isotopologues in Da to merge neighboring isotopologues.

intensity_cutoff_str
A minimum intensity threshold for isotopic profiles in percentage. This parameter may be a string piece of R commands using c, b, br, cl, k, s, se, and si variables corresponding to the same elements.

UFA_IP_memory_variables
A vector of three variables. Default values are c(1e30, 1e-12, 10). Memory may be an issue when the entire isotopologues are calculated; therefore, UFA_IP_memory_variables[1] is used to adjust memory usage. UFA_IP_memory_variables[2] indicates the minimum relative abundance (RA calculated by eq(1) in the reference [1]) of an isotopologue to include in the isotopic profile calculations. UFA_IP_memory_variables[3] is the maximum elapsed time to calculate the isotopic profile on the 'setTimeLimit' function of base R.

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]-)'

number_processing_threads
number of processing cores for multi-threaded computations.

Value

A list of isotopic profiles

References

[1] Fakouri Baygi, S., Crimmins, B.S., Hopke, P.K. Holsen, T.M. (2016). Comprehensive emerging chemical discovery: novel polyfluorinated compounds in Lake Michigan trout. *Environmental Science and Technology*, 50(17), 9460-9468, doi: [10.1021/acs.est.6b01349](https://doi.org/10.1021/acs.est.6b01349).

See Also

<https://ipc.idsl.me/>

Examples

```
library(IDSL.UFA, attach.required = TRUE)
molecular_formula <- c("C13F8N8O2", "C20H22", "C8HF16ClSO3", "C12Cl10")
peak_spacing <- 0.005 # in Da for QToF instruments
# Use this piece of code for intensity cutoff to preserve significant isotopologues
intensity_cutoff_str <- "if (s>0 & si>0) {min(c(c, 10, si*3, s*4))}
else if (s>0 & si==0) {min(c(c, 10, s*4))}
else if (s==0 & si>0) {min(c(c, 10, si*3))}
else if (s==0 & si==0) {min(c(c, 10))}"
UFA_IP_memeory_variables <- c(1e30, 1e-12)
IonPathways <- c("[M+H]+", "[M+Na]+", "[M-H2O+H]+")
number_processing_threads <- 2
listIsoProDataBase <- isotopic_profile_molecular_formula_feeder(molecular_formula,
peak_spacing, intensity_cutoff_str, UFA_IP_memeory_variables, IonPathways,
number_processing_threads)
save(listIsoProDataBase, file = "listIsoProDataBase.Rdata")
```

IUPAC_Isotopes

IUPAC Isotopes

Description

This data consists of element, mass, abundance and valence of 289 stable isotopes for 84 elements.

Usage

```
data("IUPAC_Isotopes")
```

Format

A data frame with 289 observations on the following 4 variables.

element a character vector

mass a numerical vector

abundance a numerical vector

valence a numerical vector

Note

The PubChem source for isotopes abundance and mass data is IUPAC.

References

Kim S, Chen J, Cheng T, Gindulyte A, He J, He S, Li Q, Shoemaker BA, Thiessen PA, Yu B, Zaslavsky L, Zhang J, Bolton EE. PubChem in 2021: new data content and improved web interfaces. *Nucleic Acids Res.* 2021 Jan 8; 49(D1):D1388–D1395. doi:10.1093/nar/gkaa971.

Examples

```
data(IUPAC_Isotopes)
```

molecular_formulas_source_IPDB

IPDB from a Molecular Formulas Source

Description

This function produces IPDB from a molecular formulas source (A csv or txt file).

Usage

```
molecular_formulas_source_IPDB(PARAM_SF)
```

Arguments

PARAM_SF PARAM_SF is an internal variable of the IDSL.UFA package.

molecular_formula_annotator
Molecular Formula Annotator

Description

This module annotate candidate molecular formulas in the peaklists from the IDSL.IPA pipeline using isotopic profiles.

Usage

```
molecular_formula_annotator(IPDB, spectralList, peaklist,
mass_accuracy, maxNEME, minPCS, minNDCS, minRCS, Score_coeff,
number_processing_threads)
```

Arguments

IPDB	An isotopic profile database produced by the IDSL.UFA functions.
spectralList	spectraList from the 'MS_deconvoluter' function of the IDSL.IPA package
peaklist	Peaklist from the IDSL.IPA pipeline
mass_accuracy	Mass accuracy in Da
maxNEME	Maximum value for Normalized Euclidean Mass Error (NEME) in mDa
minPCS	Minimum value for Profile Cosine Similarity (PCS)
minNDCS	Minimum value for Number of Detected Chromatogram Scans (NDCS)
minRCS	Minimum value for Ratio of Chromatogram Scans (RCS) in percentage
Score_coeff	A vector of five numbers representing coefficients of the identification score
number_processing_threads	Number of processing threads for multi-threaded processing

Value

A dataframe of candidate molecular formulas

molecular_formula_library_generator
Molecular Formula Database Producer

Description

This function generates an efficient database for molecular formula matching against a database.

Usage

```
molecular_formula_library_generator(entire_molecular_formulas)
```

Arguments

entire_molecular_formulas
A string vector of molecular formulas (redundancy is allowed)

Value

A vector of frequency of molecular formulas in the database.

Examples

```
entire_molecular_formulas <- c("C2H6O", "C2H6O", "C2H6O", "C2H6O", "CH4O", "CH4O",
"CH4O", "NH4", "C6H12O6")
db <- molecular_formula_library_generator(entire_molecular_formulas)
freq <- db[c("C6H12O6", "CH4O")]
```

molecular_formula_library_search

Molecular Formula Library Search

Description

This function attempts to match candidate molecular formulas against a library of molecular formulas using a set of ionization pathways.

Usage

```
molecular_formula_library_search(MolecularFormulaAnnotationTable,
IPDB, MF_library, IonPathways, number_processing_threads = 1)
```

Arguments

MolecularFormulaAnnotationTable	A molecular formula annotation table from the 'molecular_formula_annotator' module.
IPDB	A list of isotopic profiles database for the targeted compounds.
MF_library	A library of molecular formulas generated using the 'molecular_formula_library_generator' module.
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]-")'
number_processing_threads	Number of processing threads for multi-threaded processing

`monoisotopic_mass_calculator`*Monoisotopic Mass Calculator*

Description

This function calculates monoisotopic mass of a molecular formula

Usage

```
monoisotopic_mass_calculator(MoleFormVec, Elements_mass_abundance)
```

Arguments

MoleFormVec A numerical vector molecular formula

Elements_mass_abundance

A list of isotopic mass and abundance of elements obtained from the 'element_sorter' function

Value

The monoisotopic mass

Examples

```
Elements <- c("C", "H", "O")
MoleFormVec <- c(2, 6, 1) # C2H6O
EL_mass_abundance <- element_sorter(ElementList = Elements, ElementOrder = "alphabetical")
Elements_mass_abundance <- EL_mass_abundance[["massAbundanceList"]]
MImass <- monoisotopic_mass_calculator(MoleFormVec, Elements_mass_abundance)
```

`score_coefficients_optimization`*Coefficients Score Optimization*

Description

This function optimizes the coefficients of the score function.

Usage

```
score_coefficients_optimization(PARAM_SFT)
```

Arguments

PARAM_SFT

PARAM_SFT is a variable derived from the 'UFA_score_function_optimization_xlsxAnalyzer' function

score_coefficient_evaluation

Score Coefficient Evaluation

Description

This function evaluates the efficiency of the optimization process.

Usage

```
score_coefficient_evaluation(PARAM_SFT)
```

Arguments

PARAM_SFT PARAM_SFT is a variable derived from the 'UFA_coefficient_xlsxAnalyzer' function

UFA_enumerated_chemical_space

IPDBs from UFA Enumerated Chemical Space (ECS) approach

Description

This function produces the isotopic profile database using the UFA enumerated chemical space (ECS) approach.

Usage

```
UFA_enumerated_chemical_space(PARAM_MF)
```

Arguments

PARAM_MF A dataframe of the molecular formula constraints in the UFA spreadsheet

UFA_enumerated_chemical_space_xlsxAnalyzer

IPDBs from UFA Enumerated Chemical Space (ECS) xlsx Analyzer

Description

This function evaluates the molecular formula generation constraints in the spreadsheet to create the isotopic profile database.

Usage

```
UFA_enumerated_chemical_space_xlsxAnalyzer(PARAM_MF)
```

Arguments

PARAM_MF A dataframe of the molecular formula constraints in the UFA spreadsheet

UFA_locate_regex *UFA Locate regex*

Description

Locate indices of the pattern in the string

Usage

```
UFA_locate_regex(string, pattern)
```

Arguments

string a string as character
pattern a pattern to screen

Details

This function returns 'NA' when no matches are detected for the pattern.

Value

A 2-column matrix of location indices. The first and second columns represent start and end positions, respectively.

Examples

```
pattern <- "Cl"  
string <- "NaCl.5HCl"  
Location_Cl <- UFA_locate_regex(string, pattern)
```

UFA_logRecorder	<i>UFA_logRecorder</i>
-----------------	------------------------

Description

UFA_logRecorder

Usage

```
UFA_logRecorder(messageQuote, printMessage = TRUE)
```

Arguments

messageQuote messageQuote

printMessage printMessage

UFA_profile_visualizer	<i>UFA Profile Visualizer</i>
------------------------	-------------------------------

Description

This function creates mass spectra comparison figures for a list of HRMS files and a vector of molecular formulas at specific retention times.

Usage

```
UFA_profile_visualizer(PARAM_SA)
```

Arguments

PARAM_SA PARAM_SA is a variable derived from the 'UFA_profile_visualizer_xlsxAnalyzer' function.

UFA_profile_visualizer_xlsxAnalyzer

UFA Spectra Analysis xlsxAnalyzer

Description

This function processes the spreadsheet of the UFA parameters to ensure the parameter inputs are in agreement with the UFA requirements.

Usage

```
UFA_profile_visualizer_xlsxAnalyzer(spreadsheet)
```

Arguments

spreadsheet UFA spreadsheet

Value

This function returns the UFA parameters to feed the 'UFA_profile_visualizer' function.

UFA_score_coefficient_corrector

Score Coefficient Corrector for MolecularFormulaAnnotationTable

Description

This function updates ranking orders of the individual MolecularFormulaAnnotationTable when score coefficients changed.

Usage

```
UFA_score_coefficient_corrector(input_annotated_molf_address,  
output_annotated_molf_address, IPDB_address, maxNEME, Score_coeff,  
number_processing_threads = 1)
```

Arguments

input_annotated_molf_address Address to load the individual MolecularFormulaAnnotationTables.
output_annotated_molf_address Address to save the individual MolecularFormulaAnnotationTables.
IPDB_address Address of the IPDB (.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 function.
number_processing_threads Number of processing threads for multi-threaded computations.

UFA_score_coefficient_workflow
UFA Score Coefficient Workflow

Description

This function runs the score optimization workflow.

Usage

UFA_score_coefficient_workflow(spreadsheet)

Arguments

spreadsheet The parameter spreadsheet in the .xlsx format.

UFA_score_function_optimization_xlsxAnalyzer
UFA Score Optimization xlsx Analyzer

Description

This function evaluates the parameter spreadsheet for score coefficients optimization.

Usage

UFA_score_function_optimization_xlsxAnalyzer(spreadsheet)

Arguments

spreadsheet The parameter spreadsheet in the .xlsx format.

`UFA_workflow`*UFA Workflow*

Description

This function executes the UFA workflow in order.

Usage

```
UFA_workflow(spreadsheet)
```

Arguments

spreadsheet UFA spreadsheet

Value

This function organizes the UFA file processing for better performance using the template spreadsheet.

See Also

<https://ufa.idsl.me/home>

`UFA_xlsxAnalyzer`*UFA xlsx Analyzer*

Description

This function processes the spreadsheet of the UFA parameters to ensure the parameter inputs are consistent with the requirements of the IDSL.UFA pipeline.

Usage

```
UFA_xlsxAnalyzer(spreadsheet)
```

Arguments

spreadsheet UFA spreadsheet

Value

This function returns the UFA parameters to feed the UFA_workflow function.

zero_score_function *Zero Score Function*

Description

This function generates the input for the score optimization.

Usage

```
zero_score_function(PARAM_SFT)
```

Arguments

PARAM_SFT	PARAM_SFT is a variable derived from the 'UFA_coefficient_xlsxAnalyzer' function
-----------	--

Index

* datasets

- IUPAC_Isotopes, [11](#)
- aligned_molecular_formula_annotator, [2](#)
- detect_formula_sets, [3](#)
- detect_formula_sets
(detect_formula_sets), [3](#)
- element_sorter, [4](#)
- extended_SENIOR_rule_check, [5](#)
- formula_adduct_calculator, [5](#)
- formula_vector_generator, [6](#)
- hill_molecular_formula_printer, [7](#)
- identification_score, [7](#)
- ionization_pathway_deconvoluter, [8](#)
- isotopic_profile_calculator, [9](#)
- isotopic_profile_molecular_formula_feeder,
[10](#)
- IUPAC_Isotopes, [11](#)
- molecular_formula_annotator, [13](#)
- molecular_formula_library_generator,
[13](#)
- molecular_formula_library_search, [14](#)
- molecular_formulas_source_IPDB, [12](#)
- monoisotopic_mass_calculator, [15](#)
- score_coefficient_evaluation, [16](#)
- score_coefficients_optimization, [15](#)
- UFA_enumerated_chemical_space, [16](#)
- UFA_enumerated_chemical_space_xlsxAnalyzer,
[17](#)
- UFA_locate_regex, [17](#)
- UFA_logRecorder, [18](#)
- UFA_profile_visualizer, [18](#)
- UFA_profile_visualizer_xlsxAnalyzer,
[19](#)
- UFA_score_coefficient_corrector, [19](#)
- UFA_score_coefficient_workflow, [20](#)
- UFA_score_function_optimization_xlsxAnalyzer,
[20](#)
- UFA_workflow, [21](#)
- UFA_xlsxAnalyzer, [21](#)
- zero_score_function, [22](#)