

Package: RxnSim (via r-universe)

August 20, 2024

Type Package

Title Functions to Compute Chemical and Chemical Reaction Similarity

Version 1.0.4

Date 2023-07-15

Maintainer Varun Giri <varungiri@gmail.com>

Description Methods to compute chemical similarity between two or more reactions and molecules. Allows masking of chemical substructures for weighted similarity computations. Uses packages 'rCDK' and 'fingerprint' for cheminformatics functionality. Methods for reaction similarity and sub-structure masking are as described in: Giri et al. (2015) <[doi:10.1093/bioinformatics/btv416](https://doi.org/10.1093/bioinformatics/btv416)>.

License GPL-3

Depends R (>= 4.3.0)

Imports methods, rJava, fingerprint, data.table, rcdk (>= 3.8.1)

Author Varun Giri [aut, cre]

NeedsCompilation no

Repository <https://varungiri.r-universe.dev>

RemoteUrl <https://github.com/varungiri/rxnsim>

RemoteRef HEAD

RemoteSha e235374befe191864cd6a01fdde37b055a813b4c

Contents

RxnSim-package	2
ms.compute	3
rs.clearCache	5
rs.compute	5
rs.makeDB	9
rs.mask	11

Index	13
--------------	-----------

Description

RxnSim provides methods to compute molecular and reaction similarity. It uses rCDK package (R Interface to the CDK Libraries) and fingerprints package for chemoinformatic routines.

Details

RxnSim provides methods to compute chemical similarity between two or more reactions and molecules. Molecular similarity is computed based on structural features. Reaction similarity is a function of similarities of participating molecules. The package provides multiple methods to extract structural features as fingerprints (or feature vectors) and similarity metrics. It additionally provides functionality to mask chemical substructures for weighted similarity computations. It uses rCDK and fingerprint packages for cheminformatics functionality.

User functions:

`rs.compute` computes similarity between two reactions.

`rs.compute.list` computes similarity between all pairs of reactions from two lists.

`rs.compute.sim.matrix` computes pairwise similarity between all reactions in a list.

`rs.compute.DB` computes similarity of a reaction to those in a reaction database (DB) object read from a text file.

`rs.makeDB` reads a text file containing EC Numbers, Reaction Names and Reaction SMILES and converts it into a reaction DB object.

`ms.compute` computes similarity between two molecules.

`ms.compute.sim.matrix` computes pairwise similarity between all molecules in a list.

`rs.clearCache` clears fingerprint cache.

`rs.mask` substitutes given sub-structure in the molecules of a reaction by a user defined mask.

`ms.mask` substitutes given sub-structure in a molecule by a user defined mask.

Author(s)

Varun Giri <varungiri@gmail.com>

Maintainer: Varun Giri

See Also

`rs.compute`, `ms.compute`

Examples

```
# Reaction similarity
rs.compute('CCCO>>CCC=O', 'CC(O)C>>CC(=O)C')

# Metabolite similarity
ms.compute('CCC=O', 'CC(O)C')
```

ms.compute

*Computes Similarity of Molecules***Description**

Computes chemical similarity between two (or more) input molecules.

Usage

```
ms.compute (molA, molB, format = 'smiles', standardize = TRUE, explicitH = FALSE,
            sim.method = 'tanimoto', fp.type = 'extended', fp.mode = 'bit', fp.depth = 6,
            fp.size = 1024, fpCached = FALSE)
ms.compute.sim.matrix (molA, format = 'smiles', standardize = TRUE, explicitH = FALSE,
                       sim.method = 'tanimoto', fp.type = 'extended', fp.mode = 'bit', fp.depth = 6,
                       fp.size = 1024, clearCache = TRUE)
ms.compute.PCA(molA, format = 'smiles', standardize = TRUE, explicitH = FALSE,
               fp.type = 'extended', fp.mode = 'bit', fp.depth = 6, fp.size = 1024,
               clearCache = TRUE)
```

Arguments

molA	input molecule in SMILES format or name (with path) of MDL MOL file. ms.compute.sim.matrix accepts list of molecules as input.
molB	input molecule in SMILES format or name (with path) of MDL MOL file.
format	specifies format of input molecule(s). Molecule(s) can be provided in one of following formats: 'SMILES' (default) or 'MOL'.
standardize	suppresses all explicit hydrogen if set as TRUE (default).
explicitH	converts all implicit hydrogen to explicit if set as TRUE. It is set as FALSE by default.
sim.method	similarity metric to be used to evaluate molecule similarity. Allowed types include: 'simple', 'jaccard', 'tanimoto' (default), 'russelrao', 'dice', 'rodgerstanimoto', 'achiai', 'cosine', 'kulczynski2', 'mt', 'baroniurbanibuser', 'tversky', 'robust', 'hamann', 'pearson', 'yule', 'mcconnaughey', 'simpson', 'jaccard-count' and 'tanimoto-count'.
fp.type	fingerprint type to use. Allowed types include: 'standard', 'extended' (default), 'graph', 'estate', 'hybridization', 'maccs', 'pubchem', 'kr', 'shortestpath', 'signature' and 'circular'.

fp.mode	fingerprint mode to be used. It can either be set to 'bit' (default) or 'count'.
fp.depth	search depth for fingerprint construction. This argument is ignored for 'pubchem', 'maccs', 'kr' and 'estate' fingerprints.
fp.size	length of the fingerprint bit string. This argument is ignored for 'pubchem', 'maccs', 'kr', 'estate', 'circular' (count mode) and 'signature' fingerprints.
fpCached	boolean that enables fingerprint caching. It is set to FALSE by default.
clearCache	boolean that resets the cache before (and after) processing molecule lists. It is set to TRUE by default. Cache can also be explicitly cleared by using rs.clearCache .

Details

See [rs.compute](#) functions, for details for fingerprints and similarity matrices. `ms.compute` can use fingerprint caching by enabling `fpCached` option. `ms.compute` and `ms.compute.sim.matrix` use same cache as `rs.compute` and other functions in the package. `ms.compute.PCA` computes PCA based on the fingerprints using `prcomp` function.

Value

Returns similarity value(s).

`ms.compute` returns a similarity value.

`ms.compute.sim.matrix` returns a $m \times m$ symmetric matrix of similarity values. m is the length of the input list.

`ms.compute.PCA` returns `prcomp` object.

Note

Fingerprint cache stores fingerprints generated for a molecule index based on its SMILES. When caching is enabled, the fingerprint for a molecule, if present, is retrieved from the cache. The parameters pertaining to fingerprint generation are thus ignored. If the fingerprint for the molecule is not already cached, fingerprint based on the input parameters is generated and stored in the cache.

Author(s)

Varun Giri <varungiri@gmail.com>

See Also

[rs.compute](#), [rs.clearCache](#)

Examples

```
ms.compute('N', '[H]N([H])[H]', standardize = FALSE)
```

rs.clearCache	<i>Clears Fingerprint Cache</i>
---------------	---------------------------------

Description

Clears fingerprint cache. `rs.clearCache` should be called before fingerprint method (or related properties) are modified.

Usage

```
rs.clearCache()
```

Value

No return value

Author(s)

Varun Giri <varungiri@gmail.com>

See Also

[rs.compute](#), [ms.compute](#)

rs.compute	<i>Computes Similarity of Reactions</i>
------------	---

Description

Computes similarity between two (or more) input reactions.

`rs.compute` computes similarity of two reactions.

`rs.compute.list` computes similarity of two lists of reactions.

`rs.compute.sim.matrix` computes similarity of reactions in a list.

`rs.compute.DB` computes similarity of a reaction against a database (parsed from text file).

Usage

```
rs.compute (rxnA, rxnB, format = 'rsmi', standardize = TRUE, explicitH = FALSE,  
           reversible = TRUE, algo = 'msim', sim.method = 'tanimoto',  
           fp.type = 'extended', fp.mode = 'bit', fp.depth = 6, fp.size = 1024,  
           verbose = FALSE, fpCached = FALSE)
```

```
rs.compute.list (rxnA, rxnB, format = 'rsmi', standardize = TRUE, explicitH = FALSE,  
               reversible = TRUE, algo = 'msim', sim.method = 'tanimoto',  
               fp.type = 'extended', fp.mode = 'bit', fp.depth = 6, fp.size = 1024,  
               clearCache = TRUE)
```

```
rs.compute.sim.matrix (rxnA, format = 'rsmi', standardize = TRUE, explicitH = FALSE,
                      reversible = TRUE, algo = 'msim', sim.method = 'tanimoto',
                      fp.type = 'extended', fp.mode = 'bit', fp.depth = 6, fp.size = 1024,
                      clearCache = TRUE)
rs.compute.DB (rxnA, DB, format = 'rsmi', ecrange = '*', reversible = TRUE,
              algo = 'msim', sim.method = 'tanimoto', sort = TRUE, fpCached = FALSE)
```

Arguments

rxnA	input reaction in RSMI format or name (with path) of MDL RXN file. <code>rs.compute.list</code> and <code>rs.compute.sim.matrix</code> accept list of reactions as input.
rxnB	input reaction in RSMI format or name (with path) of MDL RXN file. <code>rs.compute.list</code> accepts list of reactions as input.
DB	parsed database object as returned by <code>rs.makeDB</code> .
format	specifies format of input reaction(s). Reaction(s) can be provided in one of following formats: 'RSMI' (default) or 'RXN'.
ecrange	EC number(s) search pattern while comparing against reaction DB. * is used as wildcard. E.g., 1.2.1.* will restricted search to all reactions with EC numbers starting with 1.2.1.- .
standardize	suppresses all explicit hydrogen if set as TRUE (default).
explicitH	converts all implicit hydrogen to explicit if set as TRUE. It is set as FALSE by default.
reversible	boolean that indicates reversibility of input reaction(s). If set as TRUE (default), reaction(s) are aligned by comparing them in forward direction and by reversing one of them to compute maximum similarity value.
algo	reaction similarity algorithm to be used. One of following algorithms can be used: 'msim' (default), 'msim_max', 'rsim' and 'rsim2'. See description for the details of the algorithms.
sim.method	similarity metric to be used to evaluate reaction similarity. Allowed types include: 'simple', 'jaccard', 'tanimoto' (default), 'russehrao', 'dice', 'rodgerstanimoto', 'achiai', 'cosine', 'kulczynski2', 'mt', 'baroniurbanibuser', 'tversky', 'robust', 'hamann', 'pearson', 'yule', 'mcconnaughey', 'simpson', 'jaccard-count' and 'tanimoto-count'.
fp.type	fingerprnt type to use. Allowed types include: 'standard', 'extended' (default), 'graph', 'estate', 'hybridization', 'maccs', 'pubchem', 'kr', 'shortestpath', 'signature' and 'circular'.
fp.mode	fingerprnt mode to be used. It can either be set to 'bit' (default) or 'count'.
fp.depth	search depth for fingerprint construction. This argument is ignored for 'pubchem', 'maccs', 'kr' and 'estate' fingerprints.
fp.size	length of the fingerprint bit string. This argument is ignored for the 'pubchem', 'maccs', 'kr', 'estate', 'circular' (count mode) and 'signature' fingerprints.

verbose	boolean that enables display of detailed molecule pairing and reaction alignment (and respective similarity values). The argument is ignored for 'rsim2' algorithm.
sort	boolean that enables rs.compute.DB to return data frame sorted based upon decreasing value of similarities.
fpCached	boolean that enables fingerprint caching. It is set to FALSE by default.
clearCache	boolean that resets the cache before (and after) processing reaction lists. It is set to TRUE by default. Cache can also be explicitly cleared using rs.clearCache.

Details

RxnSim implements four algorithms to compute reaction similarity, namely `msim`, `msim_max`, `rsim` and `rsim2`.

`msim` is based on individual similarities of molecules in two reactions. First, each reactant (product) of a reaction is paired with an equivalent (similar) reactant (product) of the other reaction based on pairwise similarity values using hierarchical grouping. A 0 similarity value is assigned to each unpaired molecule. Reaction similarity is then computed by averaging the similarity values for each pair of equivalent molecule(s) and unpaired molecule(s). Molecule equivalences computed can be reviewed using verbose mode in `rs.compute`.

`msim_max` reaction similarity is computed in the same way as described for `msim` except that the unpaired molecules are not used for computing average.

`rsim` is based on cumulative features of reactant(s) and product(s) of two reactions. Each reaction is represented by two fingerprints, one each for the reactants and another for products. Reaction similarity is computed by averaging similarity values obtained by comparing reactants fingerprint and products fingerprints.

`rsim2` is based on cumulative features of all molecules in a reaction forming a reaction fingerprint. Reaction similarity is computed based on the reaction fingerprints of two reactions.

For reversible reactions (`reversible = TRUE`), apart from comparing reactions in the forward direction they are also compared by reversing one of the reactions. The greater of the two similarity values is reported.

Fingerprint Caching

`rs.compute` and `rs.compute.DB` functions can use fingerprint caching. If `fpCached` is set as `TRUE`, cache is queried first before generating fingerprints. Any new fingerprint generated is stored in the cache. Setting `fpCached = FALSE` makes no change to cache. Cache can be cleared by calling [rs.clearCache](#).

`rs.compute.list` and `rs.compute.sim.matrix` functions internally use caching. To ensure consistency of fingerprints, [rs.clearCache](#) is called internally. Use `clearCache = FALSE` to override this behaviour; it will use current state of cache and add new fingerprints to it.

Same cache is used for all functions.

Similarity metric included in RxnSim. These metric (except jaccard-count and tanimoto-count) are derived from [fingerprint package](#).

ID	Name	Remarks
simple	Sokal & Michener	bit

jaccard	Jaccard	bit
tanimoto	Tanimoto (bit)	bit and count
jaccard-count	Jaccard (count)	count
tanimoto-count	Tanimoto (count)	count ^
dice	Dice (bit)	bit and count
russelrao	Russel And Rao	bit
rodgerstanimoto	Roger And Tanimoto	bit
achiai	Ochiai	bit
cosine	Cosine	bit
kulczynski2	Kulczynski 2	bit
mt	Modified Tanimoto	bit
baroniurbanibuser	Baroni-Urbani/Buser	bit
robust	Robust (bit)	bit and count
tversky	Tversky*	bit
hamann	Hamann	bit
pearson	Pearson	bit
yule	Yule	bit
mcconnaughey	McConnaughey	bit
simpson	Simpson	bit

*Tversky coefficients can be specified by combining them into a vector, e.g., `c('tversky', a, b)`.

`tanimoto (bit)`, `dice (bit)` and `robust (bit)` compute similarity of feature vectors (count mode) by translating them to equivalent fingerprint vectors. Default similarity metric used is `tanimoto`.

List of fingerprints included in RxnSim. These are derived from [rCDK package](#).

ID	Name of the Fingerprint	Mode
standard	Standard	bit
extended	Extended	bit
estate	EState	bit
graph	Graphonly	bit
hybridization	Hybridization	bit
maccs	MACCS	bit
pubchem	Pubchem	bit
kr	Klekota-Roth	bit
shortestpath	Shortestpath	bit
signature	Signature	count
circular	Circular	bit and count

Value

`rs.compute` returns a similarity value.

`rs.compute.list`

returns a $m \times n$ matrix of similarity values. m and n are the length of two input lists respectively.

`rs.compute.sim.matrix`
returns a $m \times m$ symmetric matrix of similarity values. m is the length of the input list.

`rs.compute.DB` returns a data frame.

Note

While using fingerprint caching (by setting `fpCached = TRUE` in `rs.compute` and `rs.compute.DB` or `clearCache = FALSE` in `rs.compute.list` and `rs.compute.sim.matrix`), ensure that the fingerprints are generated using same parameters values (`fp.type`, `fp.mode`, `fp.depth` and `fp.size`). To reset cache, call `rs.clearCache`.

`rs.compute.DB` uses same parameter values for creating fingerprint as used for (and stored with) DB object (created using `rs.makeDB`) passed as argument.

Author(s)

Varun Giri <varungiri@gmail.com>

References

^ Carbonell, P., Planson, A-G., Fichera, D., & Faulon J-L. (2011) A retrosynthetic biology approach to metabolic pathway design for therapeutic production. *BMC Systems Biology*, 5:122.

See Also

[rs.makeDB](#), [rs.clearCache](#), [ms.compute](#)

Examples

```
# Reaction similarity using msim algorithm
rs.compute(rct1, rct2, verbose = TRUE)
```

`rs.makeDB`

Converts Text File to Reaction Database

Description

Reads and parses input text file containing reaction smiles into reaction database object. The reaction database is used for querying reaction similarity of candidate reactions.

Usage

```
rs.makeDB (txtFile, header = FALSE, sep = '\t', standardize = TRUE, explicitH = FALSE,
fp.type = 'extended', fp.mode = 'bit', fp.depth = 6, fp.size = 1024,
useMask = FALSE, maskStructure, mask, recursive = FALSE)
```

Arguments

txtFile	input file containing EC numbers, reaction name and RSMI. See description for format of input file.
header	boolean to indicate if the input file contains a header. It is set to FALSE by default.
sep	the field separator character to be used while reading the input file.
standardize	suppresses all explicit hydrogen if set as TRUE (default).
explicitH	converts all implicit hydrogen to explicit if set as TRUE. It is set as FALSE by default.
fp.type	Fingerprint type to use. Allowed types include: 'standard', 'extended' (default), 'graph', 'estate', 'hybridization', 'maccs', 'pubchem', 'kr', 'shortestpath', 'signature' and 'circular'.
fp.mode	fingerprint mode to be used. It can either be set to 'bit' (default) or 'count'.
fp.depth	search depth for fingerprint construction. This argument is ignored for 'pubchem', 'maccs', 'kr' and 'estate' fingerprints.
fp.size	length of the fingerprint bit string. This argument is ignored for 'pubchem', 'maccs', 'kr', 'estate', 'circular' (count mode) and 'signature' fingerprints.
useMask	boolean to indicate use of masking. If TRUE, each reaction is processed to mask given substructure. See rs.mask for details.
maskStructure	SMILES or SMARTS of the structure to be searched and masked.
mask	SMILES of structure to be used as mask.
recursive	if TRUE, all the occurrences of input substructure are replaced recursively.

Details

The parameters used to generate fingerprints are stored in the database object and returned with the parsed data. Same parameter values are used while parsing input reaction in [rs.compute.DB](#).

The input text file should contain following three fields, separated with TAB (or any appropriate field separator). A field can be left blank.

```
[EC Number] [Reaction Name] [Reaction SMILES (RSMI)]
```

The package comes with a sample reaction database file extracted from Rhea database (Morgat et al., 2015). If no `txtfile` is provided, default sample database file is used:

```
rs.makeDB()
```

A larger dataset containing all reactions from Rhea database (v.83) is also provided with the package.

Value

Returns a list, containing parsed input data, reaction fingerprints.

Data	data frame containing EC Numbers, Reaction Names and RSMI as read from the input file. MaskedRSMI are also included if masking is used.
FP	list of molecular fingerprints for each reaction in the input file. These fingerprints are further processed based on the reaction similarity algorithm.

It also contains the parameter values used for generating fingerprints, viz., `standardize`, `explicitH`, `fp.type`, `fp.mode`, `fp.depth` and `fp.size`.

Author(s)

Varun Giri <varungiri@gmail.com>

References

Morgat, A., Lombardot, T., Axelsen, K., Aimo, L., Niknejad, A., Hyka-Nouspikel, N., Coudert, E., Pozzato, M., Pagni, M., Moretti, S., Rosanoff, S., Onwubiko, J., Bougueleret, L., Xenarios, I., Redaschi, N., Bridge, A. (2017) Updates in Rhea - an expert curated resource of biochemical reactions. *Nucleic Acids Research*, **45**:D415-D418; doi: 10.1093/nar/gkw990

See Also

[rs.compute.DB](#), [rs.mask](#)

rs.mask

Masks a Sub-structure in Input Molecule or Reaction

Description

Replaces a sub-structure, provided as SMILES or SMARTS, with the given mask.

`rs.mask` masks input sub-structure in reaction.

`ms.mask` masks input sub-structure in molecule.

Usage

```
rs.mask (substructure, mask, reaction, format = 'rsmi', standardize = TRUE,  
        explicitH = FALSE, recursive = FALSE)
```

```
ms.mask (substructure, mask, molecule, format = 'smiles', standardize = TRUE,  
        explicitH = FALSE, recursive = FALSE)
```

Arguments

substructure	SMILES or SMARTS of the structure to be searched and masked.
mask	SMILES of structure to be used as mask.
reaction	Input reaction to be processed.
molecule	Input molecule to be processed.
format	specifies format of input reaction/molecule. It can be one of following for a reaction: 'RSMI' or 'RXN'; for a molecule: 'SMILES' or 'MOL'.
standardize	suppresses all explicit hydrogen if set as TRUE (default).
explicitH	converts all implicit hydrogen to explicit if set as TRUE. It is set as FALSE by default.
recursive	if TRUE, all the occurrences of input sub-structure are replaced.

Details

The sub-structure is searched in input reaction/molecule and replaced with the mask. All the bonds between identified sub-structure and the remaining atoms are mapped to the mask. If mask contains more than one atom, all the bonds are connected to the last atom in mask. By default, the first identified sub-structure is replaced. To replace all occurrences, recursive should be set to TRUE. Valence is not checked for the mask atom and the final structure.

Value

Returns SMILES with mask.

Note

Aromatic form of SMILES of the query sub-structure should be used for masking aromatic structures. Automatic aromaticity perception is not done on query structures.

Author(s)

Varun Giri <varungiri@gmail.com>

See Also

[rs.makeDB](#)

Examples

```
ms.mask('OP(=O)O', '[Cs]', 'O=P(O)(O)OP(=O)(O)OP(=O)(O)OCC3OC(n2cnc1c(ncnc12)N)C(O)C3O')
```

Index

fingerprint package, [7](#)

ms.compute, [2](#), [3](#), [5](#), [9](#)

ms.compute.sim.matrix, [2](#)

ms.mask, [2](#)

ms.mask (rs.mask), [11](#)

rCDK package, [8](#)

rs.clearCache, [2](#), [4](#), [5](#), [7](#), [9](#)

rs.compute, [2](#), [4](#), [5](#), [5](#)

rs.compute.DB, [2](#), [10](#), [11](#)

rs.compute.list, [2](#)

rs.compute.sim.matrix, [2](#)

rs.makeDB, [2](#), [6](#), [9](#), [9](#), [12](#)

rs.mask, [2](#), [10](#), [11](#), [11](#)

RxnSim (RxnSim-package), [2](#)

rxnsim (RxnSim-package), [2](#)

RxnSim-package, [2](#)