

# Package ‘MetChem’

July 21, 2025

**Version** 0.5

**Date** 2025-06-04

**Maintainer** Stefano Cacciatore <tkcaccia@gmail.com>

**Title** Chemical Structural Similarity Analysis

**Description** A new pipeline to explore chemical structural similarity across metabolites. It allows the metabolite classification in structurally-related modules and identifies common shared functional groups. The KODAMA algorithm is used to highlight structural similarity between metabolites. See Cacciatore S, Tenori L, Luchinat C, Bennett PR, MacIntyre DA. (2017) *Bioinformatics* <[doi:10.1093/bioinformatics/btw705](https://doi.org/10.1093/bioinformatics/btw705)>, Cacciatore S, Luchinat C, Tenori L. (2014) *Proc Natl Acad Sci USA* <[doi:10.1073/pnas.1220873111](https://doi.org/10.1073/pnas.1220873111)>, and Abdel-Shafy EA, Melak T, MacIntyre DA, Zadra G, Zerbini LF, Piazza S, Cacciatore S. (2023) *Bioinformatics Advances* <[doi:10.1093/bioadv/vbad053](https://doi.org/10.1093/bioadv/vbad053)>.

**Depends** R (>= 3.5.0), stats, KODAMA (>= 3.0), httr, XML, fingerprint, rcdk (>= 3.4.3)

**Suggests** knitr, rmarkdown, readxl, impute, pheatmap, RColorBrewer, clinical

**VignetteBuilder** knitr

**SuggestsNote** No suggestions

**License** GPL (>= 2)

**NeedsCompilation** no

**Repository** CRAN

**Author** Ebtessam Abdel-Shafy [aut],  
Tadele Melak [aut],  
David A. MacIntyre [aut],  
Giorgia Zadra [aut],  
Luiz F. Zerbini [aut],  
Silvano Piazza [aut],  
Stefano Cacciatore [aut, cre]

**Date/Publication** 2025-06-04 16:30:02 UTC

## Contents

allbranches	2
chemical.dissimilarity	3
ChemRICH	4
clusters.detection	4
diseasesMet	5
enzymesMet	6
features	7
HFD	8
KODAMA.chem.sim	8
Metabolites	9
nameMet	10
pathwaysMet	11
propertiesMet	11
readMet	12
selectionMet	13
substituentsMet	13
taxonomyMet	14
tree.cutting	15
WMCSA	15
write.cls	16
write.gct	17
write.gmt	17
<b>Index</b>	<b>19</b>

---

allbranches	<i>Cut a Tree into Groups of Data</i>
-------------	---------------------------------------

---

### Description

Cuts a tree as resulting from `hclust` function, into groups (a.k.a. modules).

### Usage

```
allbranches(hh,minlen=5)
```

### Arguments

hh	a tree as produced by <code>hclust</code> function.
minlen	The minimum number of elements in each module.

### Value

A list contains vectors of module memberships.

**See Also**

[cutree](#), [hclust](#), [clusters.detection](#)

**Examples**

```
data(Metabolites)

data=Metabolites$readMet$concentration
hh=hclust(dist(data),method="ward.D")
res=allbranches(hh)
```

---

chemical.dissimilarity

*Chemical dissimilarity.*

---

**Description**

This function calculates the structural dissimilarity between different metabolites using the simplified molecular-input line-entry system (SMILE) of each metabolite as input.

**Usage**

```
chemical.dissimilarity (smiles,method="tanimoto",type="extended")
```

**Arguments**

smiles	A vector of smile notations.
method	The method used to calculate the distance between molecular fingerprint ("tanimoto" as default). For more information see <a href="#">fp.sim.matrix</a> function.
type	The type of fingerprint applied to the SMILES ("extended" as default). For more information see <a href="#">get.fingerprint</a> function.

**Value**

A list contains distance between fingerprints .

**See Also**

[fp.sim.matrix](#), [get.fingerprint](#),

**Examples**

```
data(Metabolites)
d=chemical.dissimilarity(Metabolites$SMILES[1:50])
```

---

ChemRICH

*ChemRICH Dataset*

---

### Description

This dataset consists of a list of the metabolites names download from <https://chemrich.fiehnlab.ucdavis.edu/>. HMDB IDs were retrieved from PubChem Identifier Exchange Service (<https://pubchem.ncbi.nlm.nih.gov/idxchange/idxch>) and manually curated.

### Usage

```
data(ChemRICH)
```

### Value

A list with the following elements in the variable ChemRICH:

name	A vector of metabolite's names.
SMILES	A vector of SMILES representation of each metabolite.
HMDB	A vector containing HMDB IDs of each metabolite.

### Examples

```
data(ChemRICH)
```

---

clusters.detection

*Detection of clusters.*

---

### Description

This function calculates the structural similarity between different metabolites, performs hierarchical clustering using the KODAMA algorithm, and detects the optimal number of clusters. The procedure is repeated to ensure the robustness of the detection.

### Usage

```
clusters.detection (smiles,  
                    k=50,  
                    seed=12345,  
                    max_nc = 30,  
                    dissimilarity.parameters=list(),  
                    kodama.matrix.parameters=list(),  
                    kodama.visualization.parameters=list(),  
                    hclust.parameters=list(method="ward.D"),  
                    verbose = TRUE)
```

**Arguments**

smiles	A list of smile notations for the study metabolites dataset.
k	The number of components of multidimensional scaling.
seed	Seed for the generation of random numbers.
max_nc	Maximum number of clusters.
dissimilarity.parameters	Optional parameters for <a href="#">chemical.dissimilarity</a> function.
kodama.matrix.parameters	Optional parameters for <a href="#">KODAMA.matrix</a> function.
kodama.visualization.parameters	Optional parameters for <a href="#">KODAMA.visualization</a> function.
hclust.parameters	Optional parameters for <a href="#">hclust</a> function.
verbose	If verbose is TRUE, it displays the progress for each iteration.

**Value**

A list contains all results of KODAMA chemical similarity analysis and hierarchical clustering.

**See Also**

[KODAMA.matrix](#), [KODAMA.visualization](#)

**Examples**

```
data(Metabolites)
res=clusters.detection(Metabolites$SMILES)
```

---

diseasesMet

*Metabolite-associated Diseases*

---

**Description**

This function correlates metabolites to associated diseases.

**Usage**

```
diseasesMet(doc)
```

**Arguments**

doc	A list of metabolites information produced by <a href="#">readMet</a> function.
-----	---

**Value**

A data frame contains the diseases associated with each metabolite.

**See Also**

[pathwaysMet](#), [taxonomyMet](#), [enzymesMet](#)

**Examples**

```
data(Metabolites)
dis=diseasesMet(Metabolites$readMet)
```

---

enzymesMet

*Metabolite-associated Enzymes*

---

**Description**

This function finds the metabolite related enzymes.

**Usage**

```
enzymesMet(doc)
```

**Arguments**

doc                    A list of metabolites information produced by [readMet](#) function.

**Value**

A data frame contains the enzymes associated with each metabolite.

**See Also**

[pathwaysMet](#) , [taxonomyMet](#), [diseasesMet](#)

**Examples**

```
data(Metabolites)
enz=enzymesMet(Metabolites$readMet)
```

---

features	<i>Cluster features extraction</i>
----------	------------------------------------

---

## Description

This function finds features associated with each cluster.

## Usage

```
features(doc, cla, cl, HMDB_ID)
```

## Arguments

doc	The output of the <a href="#">readMet</a> function.
cla	The output of <a href="#">diseasesMet</a> , <a href="#">enzymesMet</a> , <a href="#">pathwaysMet</a> , <a href="#">propertiesMet</a> , <a href="#">substituentsMet</a> , or <a href="#">taxonomyMet</a> functions.
cl	The output of the <a href="#">allbranches</a> function containing the module memberships.
HMDB_ID	A vector of HMDB IDs associated with their chemical name.

## Value

A list of p-value calculated using Fisher test for cluster associated features.

## See Also

[KODAMA.chem.sim](#), [tree.cutting](#), [substituentsMet](#)

## Examples

```
data(Metabolites)
SMILES=Metabolites$SMILES
names(SMILES)=Metabolites$name
HMDB=Metabolites$HMDB
names(HMDB)=Metabolites$name
res=KODAMA.chem.sim(SMILES)
cl=allbranches(res$hclust)
cla=substituentsMet(Metabolites$readMet)
f=features(Metabolites$readMet, cla, cl, HMDB)
```

HFD

*HFD Dataset*

---

**Description**

This dataset is dataframe of metabolite dataset contains only chemical information.

**Usage**

```
data(HFD)
```

**Value**

A list with the following elements in the variable HFD:

SMILES	A vector of SMILES representation of each metabolite.
CHEMICAL_ID	A vector of chemical ID number or each metabolite.
PUBCHEM	A vector of identifier ID number from PUBCHEM database for chemical molecules and their activities in biological assays.
CHEMSPIDER	A vector of a unique identifier from CHEMSPIDER database each molecule.
HMDB	A vector containing HMDB IDs of each metabolite.

**Examples**

```
data(HFD)
```

---

KODAMA.chem.sim

*KODAMA chemical similarity.*

---

**Description**

This function calculates the structural similarity between different metabolites and performs hierarchical clustering using the KODAMA algorithm.

**Usage**

```
KODAMA.chem.sim (smiles,  
                 d=NULL,  
                 k=50,  
                 dissimilarity.parameters=list(),  
                 kodama.matrix.parameters=list(),  
                 kodama.visualization.parameters=list(),  
                 hclust.parameters=list(method="ward.D"))
```



**Arguments**

smiles	A list of smile notations for the study metabolites dataset.
d	A distance structure such as that returned by <code>dist</code> or a full symmetric matrix containing the dissimilarities. If <code>NULL</code> (default), then the dissimilarity matrix will be generated by <code>chemical.dissimilarity</code> function. Otherwise, <code>d</code> will be considered as the dissimilarity matrix.
k	A number of components of multidimensional scaling.
dissimilarity.parameters	Optional parameters for <code>chemical.dissimilarity</code> function.
kodama.matrix.parameters	Optional parameters for <code>KODAMA.matrix</code> function.
kodama.visualization.parameters	Optional parameters for <code>KODAMA.visualization</code> function.
hclust.parameters	Optional parameters for <code>hclust</code> function.

**Value**

A list contains all results of KODAMA chemical similarity analysis and hierarchical clustering for the KODAMA dimensions.

**See Also**

[KODAMA.matrix](#), [KODAMA.visualization](#),

**Examples**

```
data(Metabolites)

res=KODAMA.chem.sim(Metabolites$SMILES)
plot(res$kodama$visualization)
```

---

Metabolites

*Metabolomic Dataset*

---

**Description**

This dataset consists of a list of the metabolites as returned by the function `readMet` and concentration value of each metabolites.

**Usage**

```
data(Metabolites)
```

**Value**

A list with the following elements in the variable `Metabolites`:

<code>concentration</code>	A matrix containing the concentration of each metabolites.
<code>name</code>	A vector of metabolite's names.
<code>SMILES</code>	A vector of SMILES representation of each metabolite.
<code>HMDB</code>	A vector containing HMDB IDs of each metabolite.
<code>readMet</code>	A list of metabolites information produced by <a href="#">readMet</a> function.

**Examples**

```
data(Metabolites)
```

---

<code>nameMet</code>	<i>Name of metabolites</i>
----------------------	----------------------------

---

**Description**

This function extracts the metabolite's names from the list generated by [readMet](#) function.

**Usage**

```
nameMet(doc)
```

**Arguments**

`doc` A list of metabolites information produced by [readMet](#) function.

**Value**

A data frame contains the names of each metabolite.

**See Also**

[readMet](#)

**Examples**

```
data(Metabolites)
nam=nameMet(Metabolites$readMet)
```

---

pathwaysMet

*Metabolic Pathways*

---

### Description

This function finds the metabolite related pathways.

### Usage

```
pathwaysMet(doc)
```

### Arguments

doc                    A list of metabolites information produced by [readMet](#) function.

### Value

A data frame contains the pathways associated with each metabolite.

### See Also

[readMet](#), [taxonomyMet](#), [enzymesMet](#), [diseasesMet](#)

### Examples

```
data(Metabolites)
pat=pathwaysMet(Metabolites$readMet)
```

---

propertiesMet

*Physical Proprieties of metabolites*

---

### Description

This function finds the Physical Proprieties of metabolites.

### Usage

```
propertiesMet(doc)
```

### Arguments

doc                    A list of metabolites information produced by [readMet](#) function.

**Value**

A data frame contains the properties associated with each metabolite.

**See Also**

[readMet](#), [taxonomyMet](#), [substituentsMet](#), [propertiesMet](#)

**Examples**

```
data(Metabolites)
pro=propertiesMet(Metabolites$readMet)
```

---

readMet

*Metabolite Cards Reading*

---

**Description**

This function extract metabocards of your metabolites dataset from <http://www.hmdb.ca/metabolites/> database and store all of this information in a list.

**Usage**

```
readMet(ID, address = c("http://www.hmdb.ca/metabolites/"), remove=TRUE)
```

**Arguments**

ID	A vector containing the HMDBcodes (i.e., metabolite IDs) of metabolites dataset.
address	Optional address where the MetaboCards are located. The default address is <a href="http://www.hmdb.ca/metabolites/">http://www.hmdb.ca/metabolites/</a> .
remove	A logic value. If true, missing and wrong HMDB IDs are removed.

**Value**

A list containing all the information related to the metabocards.

**See Also**

[nameMet](#)

**Examples**

```
ID=c("HMDB0000122", "HMDB0000124", "HMDB0000243", "HMDB0000263")
doc=readMet(ID)
```

---

selectionMet	<i>Metabolites selection</i>
--------------	------------------------------

---

**Description**

This function select metabolites from the list generated by [readMet](#) function.

**Usage**

```
selectionMet(doc, sel)
```

**Arguments**

doc	A list of metabolites information produced by <a href="#">readMet</a> function.
sel	A vector of metabolite's HMDBcode that will be selected

**Value**

doc	A doc list contains only the selcted metabolites.
-----	---

**See Also**

[readMet](#), [nameMet](#)

**Examples**

```
data(Metabolites)
doc=selectionMet(Metabolites$readMet,c("HMDB0000299", "HMDB0000881"))
nameMet(doc)
```

---

substituentsMet	<i>Metabolite substituents</i>
-----------------	--------------------------------

---

**Description**

This function finds the metabolite related substituents.

**Usage**

```
substituentsMet(doc)
```

**Arguments**

doc	A list of metabolites information produced by <a href="#">readMet</a> function
-----	--

**Value**

A data frame contains the substituents of each metabolite.

**See Also**

[readMet](#), [nameMet](#), [propertiesMet](#)

**Examples**

```
data(Metabolites)
sub=substituentsMet(Metabolites$readMet)
```

---

taxonomyMet

*Metabolite Taxonomy*

---

**Description**

This function finds the metabolite related taxonomy.

**Usage**

```
taxonomyMet(doc)
```

**Arguments**

doc                   A list of metabolites information produced by [readMet](#) function.

**Value**

A data frame contains the taxonomy of each metabolite.

**See Also**

[readMet](#), [propertiesMet](#), [enzymesMet](#), [diseasesMet](#)

**Examples**

```
data(Metabolites)
tax=taxonomyMet(Metabolites$readMet)
```

---

tree.cutting	<i>Optimal cluster number calculation.</i>
--------------	--

---

**Description**

This function helps to estimate the optimal cluster number that fit the metabolites dataset. It applies different optimal cluster number calculating algorithms to cut clustering tree of `hclust` function. and return a list contains index corresponding to each cluster number.

**Usage**

```
tree.cutting (res,max_nc=20)
```

**Arguments**

res	A list produced by <code>KODAMA.chem.sim</code> function.
max_nc	The maximum number of cluster (default = 20).

**Value**

A list contains the calculation for each clustering of Rousseeuw's Silhouette index.

**See Also**

[KODAMA.chem.sim](#), [WMCSA](#)

**Examples**

```
data(Metabolites)

res=KODAMA.chem.sim(Metabolites$SMILES)
clu=tree.cutting(res,max_nc = 30)
plot(clu$min_nc:clu$max_nc,clu$res.S)
```

---

WMCSA	<i>Weighted Metabolite Chemical Structural Analysis</i>
-------	---

---

**Description**

Summarize metabolites concentration in each of identified clusters using the module eigenvalue (eigen-metabolite) for calculating module membership measures.

**Usage**

```
WMCSA(data,cl)
```

**Arguments**

data            dataset of different metabolite concentration in differnt samples.  
cl              The output of the [allbranches](#) function containing the module memberships.

**Value**

This function returns a matrix as output represent similarity score of metabolites within the same module among different samples.

**See Also**

[KODAMA.chem.sim](#), [tree.cutting](#)

**Examples**

```
data(Metabolites)

SMILES=Metabolites$SMILES
names(SMILES)=Metabolites$name
res=KODAMA.chem.sim(SMILES)
cl=allbranches(res$hclust)
ww=WMCSA(Metabolites$concentration,cl)
```

---

write.cls

*Write a CLS file*

---

**Description**

This function write a file in the format CLS defined by GenePattern.

**Usage**

```
write.cls(es, address)
```

**Arguments**

es              A matrix.  
address        The address of the file should be saved.

**Value**

No return value. If an invalid address is inserted, the function will generate an error.



**See Also**

[write.gmt](#), [write.gct](#)

---

write.gct

*Write a GCT file*

---

**Description**

This function write a file in the format GCT defined by GenePattern.

**Usage**

```
write.gct(es, address)
```

**Arguments**

es	A matrix.
address	The address of the file should be saved.

**Value**

No return value. If an invalid address is inserted, the function will generate an error.

**See Also**

[write.gmt](#), [write.cls](#)

---

write.gmt

*Write a GMT file*

---

**Description**

This function write a file containing the Metabolite Set information in the format GMT defined by GenePattern.

**Usage**

```
write.gmt(sub, address, min_entry=2, max_entry=50)
```

**Arguments**

sub	A matrix.
address	The address of the file should be saved.
min_entry	The minimum number of metabolites for each metabolite set.
max_entry	The maximum number of metabolites for each metabolite set.

**Value**

No return value. If an invalid address is inserted, the function will generate an error.

**See Also**

[write.gct](#), [write.cls](#)

# Index

## \* datasets

ChemRICH, 4  
HFD, 8  
Metabolites, 9

allbranches, 2, 7, 16

chemical.dissimilarity, 3, 5, 9  
ChemRICH, 4  
clusters.detection, 3, 4  
cutree, 3

diseasesMet, 5, 6, 7, 11, 14

enzymesMet, 6, 6, 7, 11, 14

features, 7  
fp.sim.matrix, 3

get.fingerprint, 3

hclust, 2, 3, 5, 9, 15  
HFD, 8

KODAMA.chem.sim, 7, 8, 15, 16  
KODAMA.matrix, 5, 9  
KODAMA.visualization, 5, 9

Metabolites, 9

nameMet, 10, 12–14

pathwaysMet, 6, 7, 11  
propertiesMet, 7, 11, 12, 14

readMet, 5–7, 10–12, 12, 13, 14

selectionMet, 13  
substituentsMet, 7, 12, 13

taxonomyMet, 6, 7, 11, 12, 14  
tree.cutting, 7, 15, 16

WMCSA, 15, 15  
write.cls, 16, 17, 18  
write.gct, 17, 17, 18  
write.gmt, 17, 17