Package 'Bios2cor'

October 2, 2020

Type Package

Title From Biological Sequences and Simulations to Correlation Analysis

Version 2.2

Date 2020-10-01

Author
Bruck Taddese [aut], Antoine Garnier [aut], Madeline Deniaud [aut], Julien Pele [ctb], Lea Bellenger [ctb], Jean-Michel Becu [ctb], Marie Chabbert [aut,cre]

Maintainer Marie Chabbert <marie.chabbert@univ-angers.fr>

Depends R (>= 3.5), bio3d, circular, bigmemory, parallel

Imports igraph

Description Utilities for computation and analysis of correlation/covariation in multiple sequence alignments and in side chain motions during molecular dynamics simulations. Features include the computation of correlation/covariation scores using a variety of scoring functions between either sequence positions in alignments or side chain dihedral angles in molecular dynamics simulations and utilities to analyze the correlation/covariation matrix through a variety of tools including network representation and principal components analysis. In addition, several utility functions are based on the R graphical environment to provide friendly tools for help in data interpretation. Examples of sequence covariation analysis are provided in: (1) Pele J, Moreau M, Abdi H, Rodien P, Castel H, Chabbert M (2014) <doi:10.1002/prot.24570> and (2) Taddese B, Deniaud M, Garnier A, Tiss A, Guissouma H, Abdi H, Henrion D, Chabbert M (2018) <doi: 10.1371/journal.pcbi.1006209>. An example of side chain correlated motion analysis is provided in: Taddese B, Garnier A, Abdi H, Henrion D, Chabbert M (2020) <doi: 10.1038/s41598-020-72766-1>. This work was supported by the French National Research Agency (Grant number: ANR-11-BSV2-026) and by GENCI (Grant number: 100567).

License GPL (>= 2)

NeedsCompilation no

Repository CRAN

Date/Publication 2020-10-02 04:22:21 UTC

R topics documented:

`bios2cor-package` *Correlation/covariation Analysis in Biological Sequences and Simulations*

Description

The `bios2cor` package is dedicated to the computation and analysis of correlation/covariation between positions in multiple sequence alignments (MSA) and between side chain dihedral angles during molecular dynamics simulations (MD). Features include the computation of correlation/covariation scores using a variety of scoring functions and their analysis through a variety of

tools including network representation and principal components analysis. In addition, several utility functions are based on the R graphical environment to provide friendly tools for help in data interpretation.

For clarity purpose, version 2 of the package differentiates scoring functions working on MSA and on MD because their arguments are different. Analysis functions are common with auto detection of MSA or MD.

The main functionalities of `bios2cor` are summarized below:

(1) CORRELATION/COVARIATION METHODS : Methods that can be used to analyze sequence alignments and molecular simulations and to calculate a correlation/covariation matrix containing a score for each pair of positions (sequence alignment) or each pair of dihedral angles (molecular simulations).

Methods working with sequence alignments (fasta or msf file is required):

- `omes`: calculates the difference between the observed and expected occurrences of each possible pair of amino acids (x, y) at positions i and j of the alignment.
- `mi` and `mip`: calculate a score based on the probability of joint occurrence of events (MI) and a corrected score by substraction of the average product (MIP), respectively.
- `elsc`: calculates a score based on rigorous statistics of correlation/covariation in a perturbation-based algorithm. It measures how many possible subsets of size n would have the composition found in column j.
- `mcbasc`: relies on a substitution matrix giving a similarity score for each pair of amino acids.

Methods working with molecular simulations (pdb and dcd files are required) :

- `dynamic_circular`: calculates a correlation/covariation score based on a circular version of the Pearson correlation coefficient, between each pair of side chain dihedral angles in a trajectory obtained from molecular dynamics simulations.
- `dynamic_omes`: calculates the difference between the observed and expected occurrences of each possible pair of rotamers (x, y) occuring at side chain dihedral angles i and j in a trajectory.
- `dynamic_mi` and `dynamic_mip`: calculate a score based on the probability of joint occurrence of rotameric states (MI) and a corrected score by substraction of the average product (MIP), respectively.

The methods working with molecular simulations require the following functions :

- `dynamic_structure`: using the result of the `xyz2torsion` function from the `bio3D` package, creates a structure that contains side chain dihedral angle informations for each selected frame of the trajectory.
- `angle2rotamer`: using the result of the `dynamic_structure` function, creates a structure that associates rotameric state to each side chain dihedral angle for each selected frame of the trajectory.

(2) ADDITIONNAL FUNCTIONS : Functions that can be used to analyse the results of the correlation/covariation methods :

Entropy functions :

- `entropy`: calculates an entropy score for each position of the alignment. This score is based on sequence conservation and uses a formula derived from the Shannon's entropy.

- dynamic_entropy: calculates a "dynamic entropy" score for each side chain dihedral angle of a protein during molecular simulations. This score is based on the number of rotameric changes of the dihedral angle during the simulation.

Filters:

- delta_filter: given an entropy object, returns a delta filter for each position of the alignment or each side chain dihedral angle of the protein, based on entropy/dynamic entropy value.

PCA :

- centered_pca: returns a principal component analysis of the double-centered correlation/covariation matrix passed as a parameter. A delta filter can be precised.

(3) OUTPUT FILES : Functions that can be used to produce output files.

Some data structures can be stored in txt/csv files :

- write.scores: Using the result of a correlation/covariation method, creates a file containing the score of each pair of positions (sequence alignment analysis) or of side chain dihedral angles (molecular simulations) and optionaly their entropy/dynamic_entropy score.
- top_pairs_analysis: Using the result of a correlation/covariation method and an integer N, creates two files containing (1) the top N pairs with their scores and (2) the individual elements, their contact counts and their entropy score for the top N pairs. Subsequently, these files can be used for network visualization with the Cytoscape program accessible at *https://cytoscape.org*.
- write.pca: Using the result of the centered_pca function, creates a file that contains the coordinates of each element in the principal component space.
- write.pca.pdb: Using the result of the centered_pca function, creates a pdb file with the PCA coordinates on three principal components along with a pml file for nice visualization with the Pymol molecular visualization program accessible at *https://pymol.org*.

Some data can be visualized in png/pdf files:

- scores.boxplot: Using the result of one or several correlation/covariation methods, creates a boxplot to visualize the distribution of the Z-scores.
- network.plot: Using the result of the top_pairs_analysis function, creates the graph of a network representation of the data.
- scores_entropy.plot: Using the result of a correlation/covariation method and an entropy structure, creates a graph comparing correlation/covariation scores with entropy values. Each pair of elements (i,j) is placed in the graph with (entropy[i] ; entropy[j]) as coordinates. The color code of each point is based on its correlation/covariation score (red/pink color for top values, blue/skyblue for bottom values).
- pca_screeplot: Using the result of the centered_pca function, creates the graph of the eigen values (positive values only).
- pca_2d: Using the result of the centered_pca function, creates a graph with the projection of the elements on two selected components.
- angles.plot: Using pdb and dcd files and the result of a correlation/covariation method, creates graphs to monitor the time evolution of each dihedral angle in the top N pairs

Details

Package:	BioCor
Type:	Package
Version:	2.1
Date:	2020-01-30
License:	GPL

Author(s)

Bruck TADDESE [aut], Antoine Garnier [aut], Madeline DENIAUD [aut], Lea BELLANGER [ctb], Julien PELE[ctb], Jean-Michel BECU [ctb], Marie CHABBERT [cre]. Maintainer: Marie CHABBERT <marie.chabbert@univ-angers.fr>

Examples

```
#File path for output files
wd <- tempdir()
#wd <- getwd()
file <- file.path(wd,"test_seq1")

#Importing multiple sequence alignment
msf <- system.file("msa/toy_align.msf", package = "Bios2cor")
align <- import.msf(msf)

#Creating correlation object with OMES method
omes <- omes(align, gap_ratio = 0.2)

#Creating entropy object
entropy <- entropy(align, gap_ratio = 0.2)

#Creating delta filter based on entropy
filter <- delta_filter(entropy, Smin = 0.3, Smax = 0.8)

#Selecting a correlation matrix
omes <-omes$score

# Creating PCA object for selected correlation matrix and storing eigen values in csv file
pca <- centered_pca(omes, filepathroot= file, pc= NULL, dec_val= 5, filter= filter)
```

`angle2rotamer` *Converts dihedral angles to rotamers*

Description

Given an object of class 'structure' and an angle to rotamer conversion file, associates a rotamer to each dihedral angle value. The object of class 'structure' contains dihedral angle values for each side chain dihedral angle and each frame of the trajectory. The conversion file is a reference file that contains the rotamer to be associated with a dihedral angle value, depending on the residue type and the dihedral angle considered. This function will allow to compute correlation/covariation scores between rotameric states.

Usage

```
angle2rotamer(dynamic_structure,
 conversion_file=system.file("rotamer/dynameomics_rotamers.csv", package= "Bios2cor"))
```

Arguments

`dynamic_structure`
: Dihedral angle structure, result of the `dynamic_structure` function

`conversion_file`
: The file containing the rotamer to be associated with each residue dihedral angle depending of the dihedral angle value.

  Each line contains five fields, separated by ','. The five fields represent the amino acid name ("R", "N",...), the dihedral angle name("chi1", "chi2",...), the associated rotamer ("g+", "t", "g-"), the start and stop angles (between -180 and 180). For example, for the *chi1* angle of the valine residue, a torsion angle between 0 and 120 is associated to rotameric state $g+$.

  Default is the "rotamer/dynameomics_rotamers.csv" conversion file provided with the Bios2cor package from the dynameomics database (`http://dynameomics.org`) .

Details

In the torsion object and in the conversion file, dihedral angle values vary between -180 and 180.

Value

A character matrix containing the rotameric state of each side chain dihedral angle for each frame in the trajectory, depending on the dihedral angle value and on a conversion file.

Author(s)

Antoine GARNIER and Lea BELLENGER

References

Van der Kamp MW, Schaeffer RD, Jonsson AL, Scouras AD, Simms AM, Toofanny RD, Benson NC, Anderson PC, Merkley ED, Rysavy S, Bromley D, Beck DAC, and Daggett V. Dynameomics: A comprehensive database of protein dynamics. Structure, 18: 423-435, 2010.

Examples

```
#Reading pdb and dcd files
pdb <- system.file("rotamer/toy_coordinates.pdb", package= "Bios2cor")
trj <- system.file("rotamer/toy_dynamics.dcd", package= "Bios2cor")

#Reading conversion file
#conversion_file <- system.file("rotamer/dynameomics_rotamers.csv", package= "Bios2cor")

#Creating dynamic_structure object
wanted_frames <- seq(from = 1, to = 40, by = 2)
dynamic_structure <- dynamic_structure(pdb, trj, wanted_frames)

#Creating rotamers object with default conversion_file
rotamers <- angle2rotamer(dynamic_structure)

#Creating rotamers object with conversion_file
#rotamers <- angle2rotamer(dynamic_structure, conversion_file)
```

`angles.plot` *plots the time evolution of the dihedral angles in top pairs*

Description

Given an object of class 'structure' and names of dihedral angles, creates plots of the dihedral angles as a function of frames in a pdf file.

Usage

```
angles.plot(dynamic_structure, angles, filepathroot=NULL)
```

Arguments

`dynamic_structure`	Dihedral angle structure, result of the `dynamic_structure` function
`angles`	A vector containing the names of the dihedral angles to be visualized.Default is NULL (all the torsional angles of the dynamic_structure object are taken into account).
`filepathroot`	Root of the full path name for the output file. Default is NULL. In this case, the output file is named "ANGLES.pdf". When filepathroot is not NULL, a "_ANGLES.pdf" extention is added to the filepathroot name.

Details

The object of class 'structure' contains the side chain dihedral angles (between -180 and 180) for each residue in the protein, for each frame of the molecular simulations. This function allows visualisation of the evolution of selected angles.

Value

returns a pdf file containing the plots of the frame dependance of each element included in argument `angles`.

Author(s)

Antoine GARNIER and Marie CHABBERT

Examples

```
#Indicating file path for output files
wd <- tempdir()
#wd <- getwd()
file <- file.path(wd,"test_dyn1")

#Reading pdb and dcd files
pdb <- system.file("rotamer/toy_coordinates.pdb", package= "Bios2cor")
trj <- system.file("rotamer/toy_dynamics.dcd", package= "Bios2cor")

#Creating dynamic_structure object for wanted frames
wanted_frames <- seq(from= 1, to= 40, by= 2)
dynamic_structure <- dynamic_structure(pdb, trj, wanted_frames)

#Calculating circular correlation between dihedral angles of selected residues
wanted_residues <- c("H","N","Q","F","Y","W")
dihed_corr <- dynamic_circular(dynamic_structure, wanted_residues)

#Selecting a correlation matrix
dihed_corr <- dihed_corr$Zscore

#Selecting angles of interest (here from the "top_pairs_analysis" function)
top_angles <- top_pairs_analysis(dihed_corr, top= 25, file)
my_angles <- unlist(top_angles$positions)

#Creating plots of the time evolution of the dihedral angles
evol_angles <- angles.plot(dynamic_structure, my_angles, file)
```

`centered_pca` *Performs principal component analysis of a correlation/covariation matrix*

Description

Given a correlation/covariation matrix, performs the principal component analysis of the centered matrix.

Usage

```
centered_pca(corr_matrix, filepathroot= NULL, filter = NULL, pc= NULL, dec_val= 5)
```

Arguments

corr_matrix	A score or Zscore matrix created by a correlation/covariation function (omes, mip, elsc, mcbasc, dynamic_circular, dynamic_omes, dynamic_mip)
filepathroot	The root of the full path name for the csv and png files where eigen values are stored or displayed. Default is NULL (Two "EIGEN.csv" and "EIGEN.png" files are created). If not NULL, the extensions "_EIGEN.csv" and "_EIGEN.png" are added to the filepathroot.
filter	A filter calculated by the delta_filter function to limit the analysis to elements within a given entropy/dynamic_entropy range. DEFAULT is NULL (no filter is applied and each element has the same weight equal to the inverse of the number of elements).
pc	A numeric value indicating the number of principal components to be saved. Default is NULL (all the principal components are saved).
dec_val	A numeric value corresponding to the precision when the *round* function is used. Default is 5.

Details

This function performs a principal component analysis of a correlation/covariation matrix after double centering. It is based on the matrix centering algorithm of the mmds.R function from the Bios2mds package. The elements have the same weight except when a delta filter is indicated. In this latter case, only the elements allowed by the delta filter are taken into account.

Value

returns an object of class 'pca' which is a named list of four elements:

eigen	a numeric vector of the eigenvalues
eigen.perc	a numeric vector of the relative eigenvalues (eigenvalues divided by the sum of the absolute eigenvalues)
coord	a numeric matrix representing the coordinates of each element of the correlation/covariation matrix in the PCA space
source	a named list with 2 elements, the correlation/covariation matrix (cor) and the delta filter vector (filter) to limit the analysis to elements within a given entropy range.

returns also two files: a csv file containing eigen values and a png file displaying eigen values.

Author(s)

Antoine GARNIER and Marie CHABBERT

Examples

```
#File path for output files
wd <- tempdir()
#wd <- getwd()
file <- file.path(wd,"test_seq_pca")

#Importing multiple sequence alignment
msf <- system.file("msa/toy_align.msf", package = "Bios2cor")
align <- import.msf(msf)

#Creating correlation object with OMES method
omes <- omes(align, gap_ratio= 0.2)

#Creating entropy object
entropy <- entropy(align, gap_ratio=0.2)

#Creating delta filter based on entropy
filter <- delta_filter(entropy, Smin = 0.2, Smax = 0.6)

#Selecting a correlation/covariation matrix
matrix_omes <-omes$score

#Creating PCA object for selected matrix and storing eigen values in csv file
pca <- centered_pca(matrix_omes, filepathroot= file, filter = filter)
```

---

delta_filter *Creation of an entropy/dynamic score delta filter for each element*

Description

Given an entropy object (result of the entropy or of the dynamic_entropy function), creates a vector with a delta filter of each element based on the entropy value. The vector will be used to limit the analysis to the elements in the given entropy range in the centered_pca and top_pairs_analysis functions.

Usage

```
delta_filter(entropy, Smin = 0, Smax = 1)
```

Arguments

entropy	An object created by the entropy or the dynamic_entropyfunction
Smin	A value indicating the minimum entropy value. (Smin = 0 by default)
Smax	A value indicating the maximum entropy value. (Smax = 1 by default)

Details

The object returned by the entropy or the dynamic_entropy function contains an entropy score for each element. The delta weighting of each element is calculated as follow :

$$weighting[i] = \begin{cases} 1, & Smin < entropy[i] < Smax \\ 0, & otherwise \end{cases}$$

Value

A vector that contains a 0 or 1 weighting score for each element (position in sequence alignment or side chain dihedral angle in trajectory) to limit principal component and top pair analysis to elements within a given entropy range.

Author(s)

Antoine GARNIER

Examples

```
#Importing MSA
align <- import.msf(system.file("msa/toy_align.msf", package = "Bios2cor"))

#Creating entropy object
entropy <- entropy(align)

#Creating delta filter based on entropy
filter <- delta_filter(entropy, Smin = 0.4, Smax = 0.6)
```

---

dynamic_circular *Circular correlation*

Description

Calculates circular correlation/covariation scores between side chain dihedral angles during a molecular dynamics trajectory.

Usage

```
dynamic_circular(
  dynamic_structure,
  res_selection=
   c("C","I","L","M","V","R","H","K","D","E","N","Q","F","Y","W","T","S","P")
)
```

Arguments

dynamic_structure	Object of class 'structure' that is created by the dynamic_structure function
res_selection	List of amino acids that will be taken into account in the correlation/covariation matrix. By default, all the amino acids are taken into account except Gly and Ala, with no side chain dihedral angles.

Details

This function uses the cor.circular function from the circular package based on a circular version of the Pearson coeefficient.

Value

returns a list of four elements which are numeric matrices containing (1) the correlation/covariation scores for each pair of rotamers (score), (2) the Z-scores for each pair of rotamers (Zscore), (3) the correlation/covariation scores for each pair of rotamers with zero values for autocorrelation (correlation within the same side chain) (score_noauto) and (4) the Z-scores calculated without autocorrelation pairs and zero values for autocorrelation pairs (Zscore_noauto).

Author(s)

Bruck TADESSE, Antoine GARNIER, and Marie CHABBERT

References

Circular Statistics, from "Topics in circular Statistics" (2001) S. Rao Jammalamadaka and A. SenGupta, World Scientific.

Examples

```
#Reading pdb and dcd files
pdb <- system.file("rotamer/toy_coordinates.pdb", package= "Bios2cor")
trj <- system.file("rotamer/toy_dynamics.dcd", package= "Bios2cor")

#Creating dynamic_structure object for selected frames
wanted_frames <- seq(from = 1, to = 40, by = 2)
dynamic_structure <- dynamic_structure(pdb, trj, wanted_frames)

#Computing circular correlation between dihedral angles of selected residues
res_selection <- c("H","N","Q","F","Y","W")
dihed_corr <- dynamic_circular(dynamic_structure, res_selection)

#Computing correlation between all dihedral angles
#dihed_corr <- dynamic_circular(dynamic_structure)
```

dynamic_entropy *Variability score*

Description

Measures a "dynamic entropy" or variability score of each dihedral angle based on the number of rotameric changes during the molecular dynamics trajectory.

Usage

```
dynamic_entropy(rotamers)
```

Arguments

rotamers	A character matrix of type 'rotamers' that is produced by the angle2rotamer function. The matrix indicates the rotameric state of each side chain dihedral angle for each frame of the trajectory.

Details

The "dynamic entropy" score S is computed by summing the number of rotameric changes over all frames, normalized to the number of frames. It is not a "true entropy" score but gives usefull information on variability of the dihedral angle during the MD simulation.

Value

A numeric vector containing a "dynamic entropy" score for each side chain dihedral angle during the trajectory. The score is comprised between 0 (no change in the rotameric state during the trajectory) and 1 (rotameric change for every frame of the trajectory).

Author(s)

Antoine GARNIER, Lea BELLENGER and Marie CHABBERT

Examples

```
#Reading pdb and dcd files
pdb <- system.file("rotamer/toy_coordinates.pdb", package= "Bios2cor")
trj <- system.file("rotamer/toy_dynamics.dcd", package= "Bios2cor")

#Reading conversion file
conversion_file <- system.file("rotamer/dynameomics_rotamers.csv", package= "Bios2cor")

#Creating the dynamic_structure and rotamers objects
wanted_frames <- seq(from = 1, t = 40, by = 5)
dynamic_structure <- dynamic_structure(pdb, trj, wanted_frames)
rotamers <- angle2rotamer(dynamic_structure, conversion_file)

#creating the dynamic_entropy object
dynamic_entropy <- dynamic_entropy(rotamers)
```

dynamic_mi *Mutual Information (MI) function applied to rotamers in molecular dynamics simulations*

Description

Calculates a mutual information score (MI) based on the probability of joint occurrence of events.

Usage

```
dynamic_mi(
  dynamic_structure,
  rotamers,
  res_selection=
   c("C","I","L","M","V","R","H","K","D","E","N","Q","F","Y","W","T","S","P")
)
```

Arguments

dynamic_structure	An object of class 'structure' that is created by the dynamic_structure function
rotamers	A character matrix of type 'rotamers' that is produced by the angle2rotamer function. The matrix indicates the rotameric state of each side chain dihedral angle for each frame of the trajectory.
res_selection	List of amino acids that will be taken into account in the correlation/covariation matrix. By default, all the amino acids are taken into account except Gly and Ala, with no side chain dihedral angles.

Details

The MI score at position [i,j] has been computed with the following formula :

- $MI(i,j) = \sum_{x,y} p_{x,y}(i,j) ln \frac{p_{x,y}(i,j)}{p_x(i) p_y(j)}$

where $p_{x,y}(i,j)$ is the frequency of the rotamer pair (x,y) at dihedral angles i and j.

Value

returns a list of four elements which are numeric matrices containing (1) the correlation/covariation scores for each pair of rotamers (score), (2) the Z-scores for each pair of rotamers (Zscore), (3) the correlation/covariation scores for each pair of rotamers with zero values for autocorrelation (correlation within the same side chain) (score_noauto) and (4) the Z-scores calculated without autocorrelation pairs and zero values for autocorrelation pairs (Zscore_noauto).

Author(s)

Antoine GARNIER and Marie CHABBERT

References

Dunn SD, Wahl LM, Gloor GB. Mutual information without the influence of phylogeny or entropy dramatically improves residue contact prediction. Bioinfor;atics 2008;24:333-340. Martin LC, Gloor GB, Dunn SD, Wahl LM. Using infor;ation theory to search for co-evolving residues in proteins. Bioinformatics 2005;21:4116-4124.

Examples

```
#Reading pdb and dcd files
pdb <- system.file("rotamer/tiny_toy_coordinates.pdb", package= "Bios2cor")
trj <- system.file("rotamer/tiny_toy_dynamics.dcd", package= "Bios2cor")

#Creating dynamic_structure object
wanted_frames <- seq(from = 5, to = 40, by = 15)
dynamic_structure <- dynamic_structure(pdb, trj, wanted_frames)

#Creating rotamers object using conversion_file
conversion_file <- system.file("rotamer/dynameomics_rotamers.csv", package= "Bios2cor")
rotamers <- angle2rotamer(dynamic_structure, conversion_file)

#Creating correlation object for selected residues using MI method
wanted_residues <- c("H","N")
mi_corr <- dynamic_mi(dynamic_structure, rotamers, wanted_residues)
```

`dynamic_mip` *Mutual Information Product (MIP) function applied to rotamers in molecular dynamics simulations*

Description

Calculates a corrected mutual information score (MIP), by substraction of the average product from the probability of joint occurrence of events.

Usage

```
dynamic_mip(
  dynamic_structure,
  rotamers,
  res_selection=
   c("C","I","L","M","V","R","H","K","D","E","N","Q","F","Y","W","T","S","P")
)
```

Arguments

`dynamic_structure`	An object of class 'structure' that is created by the `dynamic_structure` function
`rotamers`	A character matrix of type 'rotamers' that is produced by the `angle2rotamer` function. The matrix indicates the rotameric state of each side chain dihedral angle for each frame of the trajectory.

res_selection List of amino acids that will be taken into account in the correlation/covariation matrix. By default, all the amino acids are taken into account except Gly and Ala, with no side chain dihedral angles.

Details

The MIP score at position [i,j] has been computed with the following formula :

$$MIP(i,j) = MI(i,j) - \frac{MI(i,\bar{j})MI(\bar{i},j)}{< MI >}$$

with :

- $MI(i,j) = \sum_{x,y} p_{x,y}(i,j) ln \frac{p_{x,y}(i,j)}{p_x(i)p_y(j)}$
- $MI(i,\bar{j}) = \frac{1}{n-1} \sum_{j \neq i} MI(i,j)$
- $MI(\bar{i},j) = \frac{1}{n-1} \sum_{i \neq j} MI(i,j)$
- $< MI >= \frac{2}{n(n-1)} \sum_{i,j} MI(i,j)$

and where $p_{x,y}(i,j)$ is the frequency of the rotamer pair (x,y) at dihedral angles i and j.

Value

returns a list of four elements which are numeric matrices containing (1) the correlation/covariation scores for each pair of rotamers (score), (2) the Z-scores for each pair of rotamers (Zscore), (3) the correlation/covariation scores for each pair of rotamers with zero values for autocorrelation (correlation within the same side chain) (score_noauto) and (4) the Z-scores calculated without autocorrelation pairs and zero values for autocorrelation pairs (Zscore_noauto).

Author(s)

Antoine GARNIER and Marie CHABBERT

References

Dunn SD, Wahl LM, Gloor GB. Mutual information without the influence of phylogeny or entropy dramatically improves residue contact prediction. Bioinfor;atics 2008;24:333-340. Martin LC, Gloor GB, Dunn SD, Wahl LM. Using infor;ation theory to search for co-evolving residues in proteins. Bioinformatics 2005;21:4116-4124.

Examples

```
#Reading pdb and dcd files
pdb <- system.file("rotamer/tiny_toy_coordinates.pdb", package= "Bios2cor")
trj <- system.file("rotamer/tiny_toy_dynamics.dcd", package= "Bios2cor")

#Creating dynamic_structure object
wanted_frames <- seq(from = 5, to = 40, by = 15)
dynamic_structure <- dynamic_structure(pdb, trj, wanted_frames)
```

```
#Creating rotamers object using conversion_file
conversion_file <- system.file("rotamer/dynameomics_rotamers.csv", package= "Bios2cor")
rotamers <- angle2rotamer(dynamic_structure, conversion_file)

#Creating correlation object for selected residues with MIP method
wanted_residues <- c("H","N")
mip_corr <- dynamic_mip(dynamic_structure, rotamers, wanted_residues)
```

`dynamic_omes` *OMES (Observed minus Expected Squared) function applied to rotamers in molecular dynamics simulations*

Description

Calculates difference between observed and expected occurrences of each possible pair of rotamers (x, y) for i and j dihedral angles over all frames

Usage

```
dynamic_omes(
  dynamic_structure,
  rotamers,
  res_selection=
   c("C","I","L","M","V","R","H","K","D","E","N","Q","F","Y","W","T","S","P")
)
```

Arguments

`dynamic_structure`	An object of class 'structure' that is created by the `dynamic_structure` function
`rotamers`	A character matrix that is produced by the angle2rotamer function. The matrix indicates the rotameric state of each side chain dihedral angle for each frame of the trajectory.
`res_selection`	List of amino acids that will be taken into account in the correlation/covariation matrix. By default, all the amino acids are taken into account except Gly and Ala, with no side chain dihedral angles.

Details

The OMES score for angles [i,j] has been computed with the following formula :

$$OMES(i,j) = \frac{1}{N}\sum_{x,y}(N^{obs}_{x,y}(i,j) - N^{ex}_{x,y}(i,j))^2$$

with : $N^{ex}_{x,y}(i,j) = p_x(i)p_y(j)N$

where :

- $N_{x,y}^{obs}(i,j)$ is number of times that each (x,y) rotamer pair is observed at angles i and j
- $N_{x,y}^{ex}(i,j)$ is number of times that each (x,y) rotamer pair would be expected, based on the frequency of rotamer x and y at angles i and j
- N is the number of frames
- $p_x(i)$ is the frequency of rotamer x at angle i
- $p_y(j)$ is the frequency of rotamer y at angle j

Value

returns a list of four elements which are numeric matrices containing (1) the correlation/covariation scores for each pair of rotamers (score), (2) the Z-scores for each pair of rotamers (Zscore), (3) the correlation/covariation scores for each pair of rotamers with zero values for autocorrelation (correlation within the same side chain) (score_noauto) and (4) the Z-scores calculated without autocorrelation pairs and zero values for autocorrelation pairs (Zscore_noauto).

Author(s)

Antoine GARNIER, Lea BELLENGER, and Marie CHABBERT

References

Fodor AA and Aldrich RW. Influence of conservation on calculations of amino acid covariance in multiple sequence alignments. Proteins. 2004;56(2):211-21.

Examples

```
#Reading pdb and dcd files
pdb <- system.file("rotamer/toy_coordinates.pdb", package= "Bios2cor")
trj <- system.file("rotamer/toy_dynamics.dcd", package= "Bios2cor")

#Creating dynamic_structure object
wanted_frames <- seq(from = 5, to = 40, by = 10)
dynamic_structure <- dynamic_structure(pdb, trj, wanted_frames)

#Creating rotamers object using conversion_file
conversion_file <- system.file("rotamer/dynameomics_rotamers.csv", package= "Bios2cor")
rotamers <- angle2rotamer(dynamic_structure, conversion_file)

#Creating correlation object for selected residues with OMES method
wanted_residues <- c("W")
omes_corr <- dynamic_omes(dynamic_structure, rotamers, wanted_residues)
```

dynamic_structure *Creates the data structure for the analysis of side chain dihedral angles*

Description

Given a structure pdb file, a trajectory dcd file and frame indices, gathers information on side chain dihedral angles in a unique structure. This structure will be used in correlation/covariation methods aimed at analyzing side chain rotational motions during molecular dynamics simulations.

Usage

```
dynamic_structure(pdb, trj, frames=NULL)
```

Arguments

pdb	Filepath of the pdb file
trj	Filepath of trajectory file (dcd file)
frames	Indicates the selected frames for the analysis, created with the seq function (Usage: frames <-seq(from ,to , by=). Default is NULL (all the frames of the trajectory are taken into account).

Value

Returns a list of class 'structure' with the following elements containing information on the sequence, structure, trajectory and side chain dihedral angles (values and names) of the protein during the simulation:

pdb	an object of class 'pdb' created by the read.pdbfunction from the bio3d package
dcd	A numeric matrix of xyz coordinates with a frame per row and a Cartesian coordinate per column. Created by the read.dcdfunction from the bio3d package
xyz	A numeric matrix of xyz coordinates with a frame per row and a Cartesian coordinate per column. For each frame, the protein coordinates have been fitted on the pdb structure using the fit.xyz from the bio3d package
tor	A numeric matrix of side chain dihedral angles with a frame per row and a dihedral angle per column. Contains only side chain dihedral angles. Created by the xyz2tor function from the bio3d package
tor.names	a character vector with the names of all side chain chi dihedral angles. They are written as "M.chiN" where M is the residue number and N the dihedral angle chi (chi1, chi2,...). Alanine and Glycine residues which do not have side chain dihedral angles are omitted.
tor.resno	a character vector with the residue number M of all side chain chi dihedral angles.

tor.angle	a character vector with the dihedral angle (chiN) of all side chain chi dihedral angles.
nb_torsions	a numeric value indicating the total number of dihedral angles
prot.seq	a character vector with the sequence of the protein
nb_residues	a numeric value indicating the number of residues in the protein
nb_frames	a numeric value indicating the total number of selected frames
frames	a numeric vector indicating the selected frames

Author(s)

Bruck TADDESE, Antoine GARNIER and Marie CHABBERT

Examples

```
#Reading pdb and dcd files
pdb <- system.file("rotamer/toy_coordinates.pdb", package= "Bios2cor")
trj <- system.file("rotamer/toy_dynamics.dcd", package= "Bios2cor")

#Creating dynamic_structure object for selected frames
wanted_frames <- seq(from = 1, to = 40, by = 2)
dynamic_structure <- dynamic_structure(pdb, trj, wanted_frames)

#Creating dynamic_structure object for all the frames
#dynamic_structure <- dynamic_structure(pdb, trj)
```

---

elsc *Explicit Likelihood of Subset Covariation (ELSC) function*

Description

Calculates a score based on rigorous statistics of correlation/covariation in a perturbation-based algorithm. It measures how many possible subsets of size n would have the composition found in column j in the subset alignment defined by the perturbation in column i, and in the ideal subset (i.e., in a subset with the amino acid distribution equal to the total alignment).

Usage

```
elsc(align, gap_ratio = 0.2)
```

Arguments

align	An object of class 'align' created by the import.msf or the import.fasta function from a sequence alignment
gap_ratio	Numeric value between 0 and 1 indicating the maximal gap ratio at a given position in the MSA for this position to be taken into account. Default is 0.2, positions with more than 20 percent of gaps will not be taken into account in the analysis. When gap_ratio is 1 or close to 1, only positions with at least 1 aa are taken into account (positions with only gaps are excluded).

Details

The ELSC score at position [i,j] has been computed with the following formula :

$$ELSC(i,j) = -ln \prod_{y} \frac{\binom{N_{y(j)}}{n_{y(j)}}}{\binom{N_{y(j)}}{m_{y(j)}}}$$

As a reminder, a binomial coefficient $\binom{N}{k}$ is computed as follow :

$$\binom{N}{k} = \frac{N!}{k!(N-k)!}$$

where :

- $N_{y(j)}$ is the number of residues y at position j in the total (unperturbed) sequence alignment
- $n_{y(j)}$ is the number of residues y at position j in the subset alignment defined by the perturbation in column i
- $m_{y(j)}$ is the number of residues y at position j in the ideal subset (i.e., in a subset with the amino acid distribution equal to the total alignment)

Value

A list of two elements which are numeric matrices containing the ELSC scores and Z-scores for each pair of elements.

Author(s)

Madeline DENIAUD and Marie CHABBERT

References

Dekker JP, Fodor A, Aldrich RW, Yellen G. A perturbation-bqsed method for calculating explicit likelihood of evolutionary covariance in multiple sequence alignements. Bioinformatics 2004;20:1565-1572.

Examples

```
#Importing MSA file
align <- import.msf(system.file("msa/toy_align.msf", package = "Bios2cor"))

#Creating correlation object with ELSC method for positions with gap ratio < 0.1
elsc <- elsc(align, gap_ratio = 0.1)

#Creating correlation object with ELSC method for positions with gap_ratio < 0.2 (Default)
#elsc <- elsc(align)
```

`entropy` *Entropy score*

Description

Measures the entropy score of each position in a sequence alignment

Usage

```
entropy(align, gap_ratio=0.2)
```

Arguments

`align`	An object created by the `import.msf` or the `import.fasta` function from a multiple sequence alignment file
`gap_ratio`	Numeric value between 0 and 1 indicating the maximal gap ratio at a given position in the MSA for this position to be taken into account. 1 is excluded (positions with gaps only). Default is 0.2, positions with more than 20 percent of gaps will not be taken into account in the analysis. When gap_ratio is 1 or close to 1, only positions with at least 1 aa are taken into account (positions with only gaps are excluded).

Details

The entropy score S at position i has been computed with a formula derived from the Shannon's entropy as follow :

$$S(i) = -\sum_{x} p_x(i) \log_{20} p_x(i)$$

where :

- *i* is the position in the sequence
- *x* is the sequence index
- $p_x(i)$ represents the frequency of residue x at position i

Value

A vector containing an entropy value for each position in the alignment

Author(s)

Antoine GARNIER and Marie CHABBERT

References

Shannon CE. A mathematical theory of communication. Bell Syst Techn J 1948;27:379-423.

Examples

```
#Importing MSA file
align <- import.msf(system.file("msa/human_gpcr.msf", package = "Bios2cor"))

#creating entropy object for positions with gap ratio < 0.5
entropy <- entropy(align,gap_ratio=0.5)
```

`import.fasta` *Reads a file in FASTA format*

Description

Reads a Multiple Sequence Alignment (MSA) file in FASTA format (.fasta or .fa extension).

Usage

```
import.fasta(file, aa.to.upper = TRUE, gap.to.dash = TRUE, log.file = NULL)
```

Arguments

`file`	a string of characters to indicate the name of the MSA file to be read.
`aa.to.upper`	a logical value indicating whether amino acids should be converted to upper case (TRUE) or not (FALSE). Default is TRUE.
`gap.to.dash`	a logical value indicating whether the dot (.) and tilde ($\sim$) gap symbols should be converted to dash (-) character (TRUE) or not (FALSE). Default is TRUE.
`log.file`	a file containing errors that occurred when trying to read fasta file

Details

Initially, FASTA (for FAST-ALL) was the input format of the FASTA program, used for protein comparison and searching in databases. Presently, FASTA format is a standard format for biological sequences.

The FASTA formatted file of a single sequence displays:

- a single-line description beginning with a greater-than (>) symbol. The following word is the identifier.
- followed by any number of lines, representing biological sequence.

For multiple alignments, the FASTA formatted sequences are concatenated to create a multiple FASTA format.

Value

A object of class 'align', which is a named list whose elements correspond to sequences, in the form of character vectors.

Note

The `import.fasta` function was developed for the `bios2mds` R package (Julien PELE [aut], Jean-Michel BECU [aut], Marie CHABBERT [cre]). .

Author(s)

Julien PELE

References

Pearson WR and Lipman DJ (1988) Improved tools for biological sequence comparison. *Proc Natl Acad Sci U S A* **27**:2444-2448.

Examples

```
# Importing MSA file in FASTA format
aln <- import.fasta(system.file("msa/toy2_align.fa", package = "Bios2cor"))
```

`import.msf` *Reads a multiple sequence alignment file in MSF format*

Description

Reads a Multiple Sequence Alignment (MSA) file in MSF format (.msf extension).

Usage

```
import.msf(file, aa.to.upper = TRUE, gap.to.dash = TRUE, log.file = NULL)
```

Arguments

`file`	a string of characters to indicate the name of the MSA file to be read.
`aa.to.upper`	a logical value indicating whether amino acids should be converted to upper case (TRUE) or not (FALSE). Default is TRUE.
`gap.to.dash`	a logical value indicating whether the dot (.) and tilde ($\sim$) gap symbols should be converted to the dash (-) character (TRUE) or not (FALSE). Default is TRUE.
`log.file`	a file containing errors that occurred when trying to read fasta file

Details

Initially, Multiple Sequence Format (MSF) was the multiple sequence alignment format of the Wisconsin Package (WP) or GCG (Genetic Computer Group). This package is a suite of over 130 sequence analysis programs for database searching, secondary structure prediction or sequence alignment. Presently, numerous multiple sequence alignment editors (Jalview and GeneDoc for example) can read and write MSF files.

MSF file displays several specificities:

- a header containing sequence identifiers and characteristics (length, check and weight).
- a separator symbolized by 2 slashes (//).
- sequences of identifiers, displayed by consecutive blocks.

Value

A object of class 'align', which is a named list whose elements correspond to sequences, in the form of character vectors.

Note

The `import.msf` function was developed for the `bios2mds` R package (Julien PELE [aut], Jean-Michel BECU [aut], Marie CHABBERT [cre]).

It checks the presence of duplicated identifiers in header. Sequences whose identifiers are missing in header are ignored.

Author(s)

Julien PELE

See Also

`read.alignment` function from `seqinr` package.
`read.GDoc` function from `aaMI` package (archived).

Examples

```
#Importing MSA file in MSF format
aln <- import.msf(system.file("msa/toy_align.msf", package = "Bios2cor"))
```

---

mcbasc *McBASC (McLachlan Based Substitution Correlation) function*

Description

Calculates a score for each pair of residus in the sequenec alignment. It relies on a substitution matrix giving a similarity score for each pair of amino acids.

Usage

```
mcbasc(align, gap_ratio= 0.2)
```

Arguments

`align`	An object of class 'align' created by the `import.msf` or the `import.fasta` function from a sequence alignment
`gap_ratio`	Numeric value between 0 and 1 indicating the maximal gap ratio at a given position in the MSA for this position to be taken into account. Default is 0.2, positions with more than 20 percent of gaps will not be taken into account in the analysis. When gap_ratio is 1 or close to 1, only positions with at least 1 aa are taken into account (positions with only gaps are excluded).

Details

The McBASC score at position [i,j] has been computed with a formula which was initially proposed by Valencia and coworkers(1) as follow :

$$McBASC(i,j) = \frac{1}{N^2 \sigma(i)\sigma(j)} \sum_{k,l} (SC_{k,l}(i) - SC(i))(SC_{k,l}(j) - SC(j))$$

where :

- $SC_{k,l}(i)$ is the score for the amino acid pair present in sequences k and l at position i
- $SC_{k,l}(j)$ is the score for the amino acid pair present in sequences k and l at position j
- $SC(i)$ is the average of all the scores $SC_{k,l}(i)$
- $SC(j)$ is the average of all the scores $SC_{k,l}(j)$
- $\sigma(i)$ is the standard deviation of all the scores $SC_{k,l}(i)$
- $\sigma(j)$ is the standard deviation of all the scores $SC_{k,l}(j)$

Value

A list of two elements which are numeric matrices containing the mcbasc scores and Z-scores for each pair of elements.

Author(s)

Madeline DENIAUD and Marie CHABBERT

References

(1) Gobel U, Sander C, Schneider R, Valencia A. Correlated mutations and residue contacts in proteins. Proteins 1994;18:309-317.

Examples

```
#Importing MSA file
align <- import.msf(system.file("msa/toy_align.msf", package = "Bios2cor"))

#Creating correlation object with McBASC method for positions with gap_ratio < 0.2 (Default)
mcbasc <- mcbasc(align, gap_ratio = 0.2)
```

mi *Mutual Information (MI) function*

Description

Calculates a mutual information score (MI) based on the probability of joint occurrence of events.

Usage

```
mi(align, gap_ratio= 0.2)
```

Arguments

align	An object of class 'align' created by the import.msf or the import.fasta function from a sequence alignment
gap_ratio	Numeric value between 0 and 1 indicating the maximal gap ratio at a given position in the MSA for this position to be taken into account. Default is 0.2, positions with more than 20 percent of gaps will not be taken into account in the analysis. When gap_ratio is 1 or close to 1, only positions with at least 1 aa are taken into account (positions with only gaps are excluded).

Details

The MI score at position [i,j] has been computed with the following formula :

- $MI(i,j) = \sum_{x,y} p_{x,y}(i,j) ln \frac{p_{x,y}(i,j)}{p_x(i)p_y(j)}$

and where $p_{x,y}(i,j)$ is the frequency of the amino acid pair (x,y) at positions i and j.

N.B. this formula has been widely applied in the field of sequence correlation/covariation but favors pairs with high entropy.

Value

A list of two elements which are numeric matrices containing the MI scores and Z-scores for each pair of elements.

Author(s)

Madeline DENIAUD and Marie CHABBERT

References

Dunn SD, Wahl LM, Gloor GB. Mutual information without the influence of phylogeny or entropy dramatically improves residue contact prediction. Bioinfor;atics 2008;24:333-340. Martin LC, Gloor GB, Dunn SD, Wahl LM. Using infor;ation theory to search for co-evolving residues in proteins. Bioinformatics 2005;21:4116-4124.

Examples

```
#Importing MSA file
align <- import.fasta(system.file("msa/toy2_align.fa", package = "Bios2cor"))

#Creating correlation object with MI method for positions with gap ratio < 0.2 (Default)
mi <- mi(align)
```

`mip` *Mutual Information product (MIP) function*

Description

Calculates a corrected mutual information score (MIP), by substraction of the average product from the probability of joint occurrence of events.

Usage

```
mip(align, gap_ratio = 0.2)
```

Arguments

`align`	An object of class 'align' created by the `import.msf` or the `import.fasta` function from a sequence alignment
`gap_ratio`	Numeric value between 0 and 1 indicating the maximal gap ratio at a given position in the MSA for this position to be taken into account. Default is 0.2, positions with more than 20 percent of gaps will not be taken into account in the analysis. When gap_ratio is 1 or close to 1, only positions with at least 1 aa are taken into account (positions with only gaps are excluded).

Details

The MIp score at position [i,j] has been computed with the following formula :

$$MIp(i,j) = MI(i,j) - \frac{MI(i,\bar{j})MI(\bar{i},j)}{< MI >}$$

with :

- $MI(i,j) = \sum_{x,y} p_{x,y}(i,j) ln \frac{p_{x,y}(i,j)}{p_x(i)p_y(j)}$
- $MI(i,\bar{j}) = \frac{1}{n-1} \sum_{j \neq i} MI(i,j)$
- $MI(\bar{i},j) = \frac{1}{n-1} \sum_{i \neq j} MI(i,j)$
- $< MI >= \frac{2}{n(n-1)} \sum_{i,j} MI(i,j)$

and where $p_{x,y}(i,j)$ is the frequency of the amino acid pair (x,y) at positions i and j.

N.B. this formula has been widely applied in the field of sequence correlation/covariation but favors pairs with high entropy.

Value

A list of two elements which are numerical matrices containing the MIP scores and Z-scores for each pair of elements.

Author(s)

Madeline DENIAUD and Marie CHABBERT

References

Dunn SD, Wahl LM, Gloor GB. Mutual information without the influence of phylogeny or entropy dramatically improves residue contact prediction. Bioinfor;atics 2008;24:333-340. Martin LC, Gloor GB, Dunn SD, Wahl LM. Using infor;ation theory to search for co-evolving residues in proteins. Bioinformatics 2005;21:4116-4124.

Examples

```
#Importing MSA file
align <- import.fasta(system.file("msa/toy2_align.fa", package = "Bios2cor"))

#Creating correlation object with MIP method for positions with gap ratio < 0.2 (Default)
mip <- mip(align)
```

---

network.plot *Creates network structure of top elements*

Description

Given a top_pairs object (result of the top_pairs_analysis function), creates a network to visualize the elements involved in the top scoring pairs and their links

Usage

```
network.plot(top_pairs, filepathroot=NULL)
```

Arguments

top_pairs	An object of class 'top_pairs' created by the top_pairs_analysis function
filepathroot	The root of the full path name for the output file. Default is NULL (a "NETWORK.pdf" file will be created). If not NULL, the filepathroot will have the "_NETWORK.pdf" extension.

Value

A network representing links between elements in the top scoring pairs

Author(s)

Antoine GARNIER

Examples

```
#File path for output file
wd <- tempdir()
#wd <-getwd()
file <- file.path(wd,"test_omes")

#Importing MSA file
msf <- system.file("msa/toy_align.msf", package = "Bios2cor")
align <- import.msf(msf)

#Creating OMES correlation object for positions with gap ratio < 0.2
omes <- omes(align, gap_ratio= 0.2)

#Selecting a correlation matrix
omes <-omes$Zscore

#Analyzing pairs with top scores and creating 'top_pairs' object
top_pairs <- top_pairs_analysis(omes, top = 25, file)

#Plotting the network structure of top pairs in pdf file
network.plot(top_pairs, file)
```

---

omes *OMES (Observed minus Expected Squared) function*

Description

Calculates the difference between the observed and expected occurrences of each possible pair of amino acids (x, y) at positions i and j of the alignment

Usage

```
omes(align, gap_ratio = 0.2)
```

Arguments

align	An object of class 'align' created by the import.msf or the import.fasta function from a sequence alignment
gap_ratio	Numeric value between 0 and 1 indicating the maximal gap ratio at a given position in the MSA for this position to be taken into account. Default is 0.2, positions with more than 20 percent of gaps will not be taken into account in the analysis. When gap_ratio is 1 or close to 1, only positions with at least 1 aa are taken into account (positions with only gaps are excluded).

Details

The OMES score at position [i,j] has been computed with the following formula :

$$OMES(i,j) = \frac{1}{N(i,j)} \sum_{x,y} (N^{obs}_{x,y}(i,j) - N^{ex}_{x,y}(i,j))^2$$

with : $N^{ex}_{x,y}(i,j) = p_x(i)p_y(j)N$

where :

- $N^{obs}_{x,y}(i,j)$is number of times that each (x,y) pair is observed at positions i and j
- $N^{ex}_{x,y}(i,j)$is number of times that each (x,y) pair would be expected, based on the frequency of residues x and y at positions i and j
- $N(i,j)$is the number of sequences in the alignment with non-gapped residues at positions i and j
- $p_x(i)$is the frequency of amino acid x at position i
- $p_y(j)$is the frequency of amino acid y at position j

Value

A list of two elements which are numerical matrices containing the OMES scores and Z-scores for each pair of elements.

Author(s)

Jean-Miche BECU, Madeline DENIAUD, and Marie CHABBERT

References

Fodor AA and Aldrich RW. Influence of conservation on calculations of amino acid covariance in multiple sequence alignments. Proteins. 2004;56(2):211-21.

Examples

```
#Importing MSA file
msf <- system.file("msa/toy_align.msf", package = "Bios2cor")
align <- import.msf(msf)

#Creating correlation object with OMES method for positions with gap ratio < 0.2 (Default)
omes <- omes(align, gap_ratio = 0.2)
#omes <- omes(align)
```

pca_2d *PCA projection on two dimensions*

Description

Given an object of class 'pca' (result of the centered_pca function), draws a graph of the projection of the elements on two dimensions (first and second components by default)

Usage

```
pca_2d(pca_struct, abs= 1, ord= 2, filepathroot=NULL)
```

Arguments

pca_struct	An object created by the centered_pca function
abs	An integer which corresponds to the x axis of the projection plane. Default is 1 (first component)
ord	An integer which corresponds to the y axis of the projection plane. Default is 2 (second component)
filepathroot	The root of the full path name for the output file. Default is NULL (a "PCA_abs_ord.png" file will be created). If not NULL, the filepathroot will have the "_PCA_abs_ord.png" extension.

Value

A 2D graph of the projection of the elements on selected two principal components

Author(s)

Antoine GARNIER

Examples

```
#File path for output files
wd <- tempdir()
#wd <-getwd()
file <- file.path(wd,"test_seq2")

#Importing MSA file
msf <- system.file("msa/toy_align.msf", package = "Bios2cor")
align <- import.msf(msf)

#Creating OMES correlation object
omes <- omes(align, gap_ratio = 0.2)

#Creating entropy object
#entropy <- entropy(align)
```

```
#Selecting correlation matrix
omes <-omes$Zscore

#Creating PCA object for selected correlation matrix and saving eigen values
pca <- centered_pca(omes, filepathroot= file, pc= NULL, dec_val= 5, filter= NULL)

#Creating 2D plot of elements projected on selected [abs,ord] plane
pca_2d(pca, abs = 1, ord = 3, file)

#Creating 2D plot of elements projected on default [1,2] plane
#pca_2d(pca)
```

`pca_screeplot` *Creates PCA screeplot*

Description

Given a PCA structure (result of the `centered_pca` function), creates a screeplot of the positive eigen values

Usage

```
pca_screeplot(pca_struct, filepathroot=NULL)
```

Arguments

`pca_struct`	An object created by the `centered_pca` function
`filepathroot`	The root of the full path name for the output file. Default is NULL (a "EIGEN.png" file will be created). If not NULL, the filepathroot will have the "_EIGEN.png" extension.

Value

A screeplot of positive eigen values

Author(s)

Antoine GARNIER

Examples

```
#File path for output files
wd <- tempdir()
#wd <-getwd()
file <- file.path(wd,"test_seq3")

#Importing MSA file
msf <- system.file("msa/toy_align.msf", package = "Bios2cor")
align <- import.msf(msf)
```

```
#Creating OMES correlation object
omes <- omes(align, gap_ratio = 0.2)

#Selecting correlation matrix
omes <-omes$Zscore

#Creating PCA object for selected correlation matrix and saving eigen values
pca <- centered_pca(omes, filepathroot= file, pc= NULL, dec_val= 5, filter = NULL)

#Plotting scree plot
pca_screeplot(pca, file)
#pca_screeplot(pca)
```

`random.msa` *Random Alignment*

Description

Builds a multiple sequence alignment (MSA) of random sequences.

Usage

```
random.msa(nb.seq = 100, id = "SEQ", nb.pos = 100, gap = FALSE,
aa.strict = FALSE, align = NULL, align.replace = TRUE)
```

Arguments

`nb.seq`	a numeric value indicating the number of sequences in the random MSA. Default is 100.
`id`	a string of characters used to tag each sequence name. Default is "SEQ". An incremented number is attached to this tag to name each sequence.
`nb.pos`	a numeric value indicating the length of each sequence in the random MSA. Default is 100.
`gap`	a logical value indicating whether the gap character should be considered as a supplementary symbol (TRUE) or not (FALSE). Default is FALSE.
`aa.strict`	a logical value indicating whether only strict amino acids should be taken into account (TRUE) or not (FALSE). Default is FALSE.
`align`	an object of class 'align', obtained from `import.fasta` or `import.msf` function. If this parameter is not NULL, the composition of the output sequences is based on the composition of the input sequences. Default is NULL.
`align.replace`	a logical value indicating random drawing with replacement (TRUE) or without replacement (FALSE) of characters present in `align`. Default is FALSE.

Details

random.msa may be used to compare a reference MSA to a random MSA. The random MSA must have the same characteristics as the reference MSA (same number of sequences of same length).

A procedure can be applied to the random MSA to assess the amount of variance due to random mutations in the reference MSA.

The subset function is used for random selection of the amino acids. If a truly random procedure is needed, see random package.

Value

A named list whose objects correspond to random sequences.

Note

This function has been initially developped in the bios2mds R package (Julien PELE [aut], Marie CHABBERT [cre]).

Author(s)

Julien PELE

References

For an application of random MSA see :

Pele J, Abdi H, Moreau M, Thybert D and Chabbert M (2011) Multidimensional scaling reveals the main evolutionary pathways of class A G-protein-coupled receptors. *PLoS ONE* **6**: e19094. doi:10.1371.

See Also

permutation and synsequence functions from seqinr package.

Examples

```
#Importing MSA file
aln <- import.fasta(system.file("msa/toy2_align.fa", package = "Bios2cor"))

#Generating a random sequence alignment with the same characterics as input file
nb.seq <- length(aln)
nb.pos <- length(aln[[1]])
aln.random <- random.msa(nb.seq = nb.seq, nb.pos = nb.pos)
```

scores.boxplot *Creates boxplots of correlation/covariation scores*

Description

Given a list of correlation/covariation matrices, build boxplots for comparative purposes.

Usage

```
scores.boxplot(corr_matrix_list, name_list, filepathroot=NULL, elite=25, high=275)
```

Arguments

corr_matrix_list	A list of correlation/covariation matrices to be compared
name_list	The names of the correlation/covariation matrices
filepathroot	The root of the full path name for the output file. Default is NULL (a BOXPLOT.png file will be created). If not NULL, the "_BOXPLOT.png" extension is added to the filepathroot.
elite	An integer to determine the number of pairs with the highest and lowest scores (e.g. 25: pairs ranked 1 to 25 in decreasing or increasing order) to be colored with the "elite" color codes. Default is 25.
high	An integer to determine the number of pairs with the next highest and lowest scores (e.g. 275: pairs ranked 26 to 275 in decreasing or increasing order) to be colored with the "high" color codes. Default is 275.

Details

The correlation/covariation matrices contain the correlation/covariation scores for each pair of elements [i,j]. The boxplots will allow comparing these scores using color codes : the highest values are dark blue, the next highest values are light blue, the lowest values are red and the next lowest values are pink.

Value

A pdf figure with boxplots to compare correlation/covariation scores

Author(s)

Julien PELE and Antoine GARNIER

References

For an application of these boxplots, see :

Pele J, Abdi H, Moreau M, Thybert D and Chabbert M (2011) Multidimensional scaling reveals the main evolutionary pathways of class A G-protein-coupled receptors. *PLoS ONE* **6**: e19094. doi:10.1371.

Examples

```
#File path for output file
wd <- tempdir()
#wd <-getwd()
file <- file.path(wd,"test_seq")

#Importing MSA file
msf <- system.file("msa/toy_align.msf", package = "Bios2cor")
align <- import.msf(msf)

#Creating OMES correlation object
omes <- omes(align, gap_ratio = 0.2)

#Creating MIP correlation object
#mip <- mip(align, gap_ratio = 0.2)

#Selecting correlation matrices
omes <-omes$Zscore
#mip <- mip$Zscore

#Creating a list of matrices and plotting the boxplots in a graph
#Two matrices
#corr_matrix_list <- list(omes,mip)
#name <- c("omes","mip")
#One matrix
corr_matrix_list <- list(omes)
name <- c("omes")
scores.boxplot(corr_matrix_list, name, file, 25, 275)
```

---

scores_entropy.plot *Scores versus entropy graph*

Description

The aim of this graph is the visualization of the entropy values of the elements in top and bottom pairs.

Usage

```
scores_entropy.plot(entropy, corr_matrix,
  filepathroot=NULL,
  elite=25,
  high=275,
  filter=NULL)
```

Arguments

entropy	An object created by the entropy (sequence alignment) or the dynamic_entropy (trajectory) function.

corr_matrix	A correlation/covariation matrix created by one of the correlation/covariation functions:
filepathroot	The root of the full path name of the graph that will be created. Default is NULL (a "ei_ej.pdf" file will be created). If not NULL, the filepathroot will have the "_ei_ej.pdf" extension.
elite	An integer to determine the number of pairs with the highest and lowest scores (e.g. 25: pairs ranked 1 to 25 in decreasing or increasing order) to be colored with the "elite" color codes. Default is 25.
high	An integer to determine the number of pairs with the next highest and lowest scores (e.g. 275: pairs ranked 26 to 275 in decreasing or increasing order) to be colored with the "high" color codes. Default is 275.
filter	A vector created by the delta_filter function, to limit the analysis to elements within a given entropy range. When filter is not NULL, only top elite and high pairs are visualized. Default is NULL.

Details

Using the result of a correlation/covariation method and an entropy structure, creates a graph comparing correlation/covariation scores with entropy values. Each pair of elements (i,j) is placed in the graph with (entropy[i] ; entropy[j]) as coordinates. In the absence of filter, the color code of each point is based on its correlation/covariation score (dark and light blue for top elite and high values, red and pink for bottom elite and high values). In the presence of an entropy based filter, only top elite and high scores are visualized in dark and light blue, respectively,

Value

A graph showing the entropy values of the elements in pairs with top and bottom entropy scores

Author(s)

Julien PELE, Antoine GARNIER and Marie CHABBERT

References

For an application of these graphs see :

Pele J, Abdi H, Moreau M, Thybert D and Chabbert M (2011) Multidimensional scaling reveals the main evolutionary pathways of class A G-protein-coupled receptors. *PLoS ONE* **6**: e19094. doi:10.1371.

Examples

```
##Example with MSA
  #File path for output file
  wd <- tempdir()
  #wd <-getwd()
  file <- file.path(wd,"test_seq7")

  #Importing MSA file
```

```
  msf <- system.file("msa/toy_align.msf", package = "Bios2cor")
  align <- import.msf(msf)

  #Creating OMES correlation object and selecting a correlation matrix
  correlation <- omes(align, gap_ratio = 0.2)
  corr_matrix <- correlation$Zscore

  #Creating entropy object
  entropy <- entropy(align)

  #Creating a delta filter based on entropy
  filter <- delta_filter(entropy, Smin = 0.4, Smax = 0.7)

  #Creating the entropy graph
  scores_entropy.plot(entropy, corr_matrix, filepathroot = file, filter=filter)

##Example with MD
#  #File path for output file
#  wd <- tempdir()
#  #wd <-getwd()
#  file <- file.path(wd,"test_dyn7")

#  #Reading pdb and dcd files
#  pdb <- system.file("rotamer/toy_coordinates.pdb", package= "Bios2cor")
#  trj <- system.file("rotamer/toy_dynamics.dcd", package= "Bios2cor")

#  #Creating dynamic_structure object for selected frames
#  wanted_frames <- seq(from = 1, to = 40, by = 1)
#  dynamic_structure <- dynamic_structure(pdb, trj, wanted_frames)
#
#  #Creating rotamers object using conversion_file
# conversion_file <- system.file("rotamer/dynameomics_rotamers.csv", package= "Bios2cor")
#  rotamers <- angle2rotamer(dynamic_structure, conversion_file)
#
#  #Creating the dynamic_entropy and filter objects
#  entropy <- dynamic_entropy(rotamers)
#  filter <- delta_filter(entropy, Smin = 0.0, Smax = 0.1)

#  #Creating correlation object
#  #dyn_cor <- dynamic_circular(dynamic_structure)
#  dyn_cor <- dynamic_omes(dynamic_structure,rotamers)

#  #selection correlation matrix
#  corr_matrix <- dyn_cor$Zscore_noauto
#
#  #Creating the entropy graph
#  scores_entropy.plot(entropy, corr_matrix, filepathroot = file, filter=filter)
```

`top_pairs_analysis` *Calculates the number of contacts for each element of the top pairs*

Description

Given a correlation object, calculates the number of pairs (contacts) each element in the top X pairs are involved in

Usage

```
top_pairs_analysis(corr_matrix, filepathroot=NULL, top=25, entropy=NULL, filter=NULL)
```

Arguments

corr_matrix	One of the matrices created by a correlation/covariation function (omes, mip, elsc, mcbasc, dynamic_circular, dynamic_omes, dynamic_mip).
filepathroot	The root of the full path names of the output files for top_pairs_analysis. Default is NULL (two csv files are created : TOPn_CONTACTS.csv and TOPn_SCORES.csv, where n is the number of top pairs). If not NULL, extentions "-TOPn_CONTACTS.csv" and "_TOPn_SCORES.csv" are added to the root name.
top	A integer indicating the number of top pairs used for this analysis. Default is 25.
entropy	An object created by the entropy or dynamic_entropy function. Default is NULL.
filter	A vector created by the delta_filter function to limit the analysis to elements in the given entropy/dynamic_entropy range. DEfault is NULL.

Details

This function sorts element pairs by correlation/covariation scores and analyzes the top X pairs to determine the number of pairs (contacts) each element of the top X pairs is involved in. If filter is TRUE, only the scores of elements in the delta filter defined entropy range are taken into account. Results are written as .csv files.

Value

returns an object of class 'top_pairs' which is a named list of four elements for subsequent network representation with the network.plot function:

pair_i	a vector containing the name of element i in the ordered top pairs
pair_j	a vector containing the name of element j in the ordered top pairs
positions	a vector containing the positions in the top n pairs
contacts	a vector containing the number of contacts of the positions in the top n pairs

returns also two .csv files containing scores and contacts of the top n pairs for subsequent network representation with Cytoscape.

Author(s)

Antoine GARNIER and Marie CHABBERT

Examples

```
#File path for output files
wd <- tempdir()
#wd <-getwd()
file <- file.path(wd,"test_seq")

#Importing MSA file
msf <- system.file("msa/toy_align.msf", package = "Bios2cor")
align <- import.msf(msf)

#Creating entropy object
entropy <- entropy(align)

#Creating OMES correlation object
omes <- omes(align, gap_ratio = 0.2)

#Selecting correlation matrix
omes <-omes$Zscore

#Creating top_pairs object and writing scores and contacts to csv files
top_pairs <- top_pairs_analysis(omes, file, top = 25, entropy=entropy)
```

`write.entropy` *Writes and displays entropy values*

Description

Given an entropy/dynamic_entropy object, writes each element (position or dihedral angle) and its entropy/dynamic_entropy value in a csv file, and displays the histogram.

Usage

```
write.entropy(entropy, filepathroot = NULL)
```

Arguments

`entropy`	An object created by the `entropy` or the `dynamic_entropy` function.
`filepathroot`	The root of the full path name for the entropy output file. Default is NULL (a "ENTROPY.csv" file is created). If filepathroot is not NULL, a "_ENTROPY.csv" extension is added to the file path root.

Details

The `entropy` function calculates entropy score for each position of an alignment file. The `dynamic_entropy`function calculate a "dynamic entropy" score for each side chain dihedral angle of a protein during a molecular simulations.

Value

A csv file containing the elements and their scores. A png file displaying the histogram of the scores.

Author(s)

Antoine GARNIER

Examples

```
#File path for output files
wd <- tempdir()
#wd <-getwd()
file <- file.path(wd,"test_seq")

#Importing multiple sequence alignment
align <- import.msf(system.file("msa/human_gpcr.msf", package = "Bios2cor"))

#Creating entropy object
entropy <- entropy(align, gap_ratio = 0.2)

#Writing results to csv file
write.entropy(entropy, file)
```

---

write.pca *Creates a file of coordinates in PCA space*

Description

Given an object of class 'pca' (result of the centered_pca function), stores the coordinates of each element in the PC space in a txt file

Usage

```
write.pca(corr_pca,filepathroot=NULL, pc=NULL, entropy= NULL)
```

Arguments

corr_pca	An object created by the centered_pca function from a correlation/covariation matrix
filepathroot	The root for the full path name of the output file where all coordinates on all components are stored. Default is NULL (a "PCA_COORD.csv" file is created). If not NULL, the "_PCA_COORD.csv extension is added to the root name.
pc	An integer corresponding to the number of principal components for which coordinates of the elements are saved. By default, this number corresponds to the number of components with positive eigen values.
entropy	An object created by the entropy or the dynamic_entropy function. DEfault is NULL.

Details

The object returned by the centered_pca function contains coordinates in the PC space for each element. Each line of the pca file will contain the name of the current element and its coordinates. Any line that contains Na value for X, Y or Z coordinates will be ignored.

Value

returns a file containing the coordinates of each element in PC space.

Author(s)

Antoine GARNIER

Examples

```
#File path for output files
wd <- tempdir()
#wd <-getwd()
file <- file.path(wd,"test_seq4")

#Importing MSA file
msf <- system.file("msa/toy_align.msf", package = "Bios2cor")
align <- import.msf(msf)

#Creating OMES correlation object and selecting correlation matrix
omes <- omes(align, gap_ratio = 0.2)
omes <-omes$Zscore

#Creating PCA object for selected matrix
pca <- centered_pca(omes, filepathroot= file, filter = NULL, pc= NULL, dec_val= 5)

#Saving coordinates of elements in csv file
write.pca(pca, file, pc = 10, entropy = NULL)
```

---

write.pca.pdb *PDB and PML file creation for 3D representation of PCA analysis*

Description

Given a PCA structure, creates .pdb and .pml files for 3D visualization with Pymol

Usage

```
write.pca.pdb(corr_pca, filepathroot=NULL, trio_comp= c(1:3))
```

Arguments

corr_pca	An object created by the centered_pca function from a correlation/covariation matrix
filepathroot	The root for the full path name of the output PDB and PML files. Default is NULL (Two PCA_l_m_n.pdb and PCA_l_m_n.pml files are created where l, m, n are the 3 selected components). If not null, '_PCA_l_m_n.pdb' and '_PCA_l_m_n.pml' extensions are added to the root name.
trio_comp	A numeric vector of length 3 indicating the principal components to be displayed. Default is c(1, 2, 3).

Details

This function creates PDB and PML files to visualize the positions of the elements (sequence positions or dihedral angles) in a 3D space corresponding to three selected components of the PCA space. The PDB file can be viewed in any molecular graphics softaware. The PML file allows a nice representation with Pymol (axis, background color, size of points and for GPCRs, color code for helices).

Value

Returns two files: a PDB file which contains three PCA coordinates for each element in PDB format and a PML file for nice visualization with Pymol.

Author(s)

Antoine GARNIER and Marie CHABBERT

Examples

```
# Example for MSA
  #File path for output files
  wd <- tempdir()
  #wd <-getwd()
  file <- file.path(wd,"test_seq5")

  #Importing MSA file
  align <- import.msf(system.file("msa/toy_align.msf", package = "Bios2cor"))

  #Creating OMES correlation object and selecting correlation matrix
  omes <- omes(align, gap_ratio = 0.2)
  cor_omes <- omes$Zscore

  #Creating PCA object for selected  matrix
 pca <- centered_pca(cor_omes, filepathroot = file, filter = NULL, pc = NULL, dec_val = 5)

  #Creating PDB and PML files (open PDB file with Pymol then "File > Run" PML file)
  indices <- c(1,2,3)
  write.pca.pdb(pca, file, indices)
```

```
### Example for MD
  #File path for output files
  wd <- tempdir()
  #wd <-getwd()
  file <- file.path(wd,"test_dyn5")

  #Redaing pdb and dcd files
  pdb <- system.file("rotamer/toy_coordinates.pdb", package= "Bios2cor")
  trj <- system.file("rotamer/toy_dynamics.dcd", package= "Bios2cor")

  #Creating dynamic_structure object for selected frames
  nb_frames_wanted <- 40
  wanted_frames <- seq(from = 5, to= nb_frames_wanted, by = 10)
  dynamic_structure <- dynamic_structure(pdb, trj, wanted_frames)

  #Creating rotamers object
 conversion_file <- system.file("rotamer/dynameomics_rotamers.csv", package= "Bios2cor")
  rotamers <- angle2rotamer(dynamic_structure, conversion_file)

  #Creating OMES correlation object and selecting correlation matrix
  wanted_residues <- c("W","Y","F","N")
  omes <- dynamic_omes(dynamic_structure, rotamers, wanted_residues)
  cor_omes <- omes$Zscore_noauto

  #Creating PCA object for selected matrix
  pca <- centered_pca(cor_omes, file, filter = NULL, pc = NULL, dec_val = 5)

  #Creating PDB and PML files (open PDB file with Pymol then "File > Run" PML file)
  indices <- c(1,2,3)
  write.pca.pdb(pca, file, indices)
```

`write.scores` *Creates a correlation/covariation ouput file*

Description

Given a correlation object, writes the score, the Zscore and, optionally the entropy, for each pair of elements in a csv file.

Usage

```
write.scores(correlation, entropy= NULL, filepathroot=NULL)
```

Arguments

`correlation`	An object created by a correlation/covariation function (`omes`, `mi`, `mip`, `elsc`, `mcbasc`, `dynamic_circular`, `dynamic_omes`, `dynamic_mi`, `dynamic_mip`

entropy	An object created by the entropy function
filepathroot	The root of the full path name for the output file. DEfault is NULL (a "CORR_SCORES.csv" file is created). If not NULL, the "_CORR_SCORES.csv" extension is added to the root name.

Details

Elements represent positions in multiple sequence alignments and side chain dihedral angles in molecular dynamic simulations (MD).

In sequence analysis, the correlation object contains two matrices with the correlation/covariation scores and Z-scores for each pair of elements [i,j]. If entropy = NULL, each line of the output file will contain element i, element j, score[i,j], and Z-score[i,j]. If entropy is not NULL, each line of the output file will contain element i, element j, score[i,j], Zscore[i,j], entropy[i], and entropy[j].

In MD analysis, the correlation object contains four matrices with (1) the correlation/covariation scores for each pair of rotamers (score), (2) the Z-scores for each pair of rotamers (Zscore), (3) the correlation/covariation scores for each pair of rotamers with zero values for autocorrelation (correlation within the same side chain) (score_noauto) and (4) the Z-scores calculated without autocorrelation pairs and zero values for autocorrelation pairs (Zscore_noauto). If entropy = NULL, each line of the output file will contain element i, element j, score[i,j], Zscore[i,j], score_noauto[i,j], and Zscore_noauto[i,j]. If entropy is not NULL, each line of the output file will contain element i, element j, score[i,j], Zscore[i,j], score_noauto[i,j], Zscore_noauto[i,j], entropy[i], and entropy[j].

Value

A csv file containing the correlation/covariation scores and optionally the entropies.

Author(s)

Antoine GARNIER and Marie CHABBERT

Examples

```
#Example for MSA
  #File path for output files
  wd <- tempdir()
  #wd <-getwd()
  file <- file.path(wd,"test_seq")

  #Importing MSA file
  msf <- system.file("msa/toy_align.msf", package = "Bios2cor")
  align <- import.msf(msf)

  #Creating correlation and entropy objects
  correlation <- omes(align, gap_ratio= 0.2)
  entropy <- entropy(align)

  #Writing results to csv file
  write.scores(correlation, entropy, file)
```

```
###Example for MD
  #File path for output files
  wd <- tempdir()
  #wd <-getwd()
  file <- file.path(wd,"test_dyn")

  #Reading the pdb and dcd files and the angles to rotamers conversion file
  pdb <- system.file("rotamer/toy_coordinates.pdb", package= "Bios2cor")
  trj <- system.file("rotamer/toy_dynamics.dcd", package= "Bios2cor")
 conversion_file <- system.file("rotamer/dynameomics_rotamers.csv", package= "Bios2cor")

  #Creating dynamic_structure and rotamers objects
  wanted_frames <- seq(from = 5, to = 40, by = 10)
  dynamic_structure <- dynamic_structure(pdb, trj, wanted_frames)
  rotamers <- angle2rotamer(dynamic_structure, conversion_file)

  #Creating correlation and entropy objects
  wanted_residues <- c("F","H","N","Y","W")
  #dyn_corr <- dynamic_omes(dynamic_structure, rotamers, wanted_residues)
  dyn_corr <- dynamic_circular(dynamic_structure, wanted_residues)
  dyn_entropy <- dynamic_entropy(rotamers)

  #Writing results to csv file
  write.scores(dyn_corr, dyn_entropy, file)
```

`xyz2torsion` *Convert Cartesian Coordinates to Torsion Angles*

Description

Convert cartesian coordinate matrix to torsion angles with function `torsion.pdb`.

Usage

```
xyz2torsion(pdb, xyz, tbl = c("basic", "mainchain",
    "sidechain", "all", "phi", "psi", paste("chi", 1:5, sep="")), ncore = NULL)
```

Arguments

`pdb`	A PDB structure object as obtained from `read.pdb`.
`xyz`	Cartesian coordinates as a Mx(3N) matrix.
`tbl`	Names of torsion angles to calculate.
`ncore`	Number of CPU cores used to do the calculation. By default (NULL), use all detected CPU cores.

Details

Available values for the argument 'tbl' include:

- Basic: "phi", "psi", "chi1".
- Mainchain: "phi", "psi".
- Sidechain: "chi1", "chi2", "chi3", "chi4", "chi5".
- All: all of the above.
- Each individual angle.

Value

Returns a MxP matrix, where M is the number of frames and P the number of valid torsion angles.

Note

New function from the bio3d package, available at <https://github.com/Grantlab/bio3d/blob/master/new_funs/xyz2torsion.R>

Author(s)

Xin-Qiu Yao

References

Grant, B.J. et al. (2006) *Bioinformatics* **22**, 2695–2696.

See Also

torsion.xyz, torsion.pdb

Examples

```
pdb <- system.file("rotamer/toy_coordinates.pdb", package= "Bios2cor")
trj <- system.file("rotamer/toy_dynamics.dcd", package= "Bios2cor")

pdb <- read.pdb(pdb)
trj <- read.dcd(trj)

#Selecting only "CA" atoms
ca.inds <- atom.select(pdb, elety = "CA")

#Getting xyz coordinates using fit.xyz form bio3d package
xyz <- fit.xyz(fixed = pdb$xyz, mobile = trj,fixed.inds=ca.inds$xyz,mobile.inds=ca.inds$xyz)

frames <- seq(from= 1, to= 40, by= 2)

#Creating torsion object for side chains using xyz2torsion function from bio3d package
tor <- xyz2torsion(pdb, xyz[frames,], tbl = "sidechain", ncore= 1)
```

Index