Package 'Bios2cor'

September 7, 2017

Type Package

Title From Biological Sequences and Simulations to Correlation Analysis

Version 1.2

Date 2017-08-16

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

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

Depends R (>= 3.1), bio3d, circular, bigmemory, parallel

Imports igraph

Description Utilities for computation and analysis of correlation/co-variation in multiple sequence alignments and in side chain motions during molecular dynamics simulations. Features include the computation of correlation/co-variation scores using a variety of scoring functions between either sequence positions in alignments or side chain dihedral angles in molecular dynamics simulations and to analyze the correlation/co-variation 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 co-variation analysis and utility tools are provided in: Pele J, Moreau M, Abdi H, Rodien P, Castel H, Chabbert M. (2014) <doi:10.1002/prot.24570>. This work was supported by the French National Research Agency (Grant number: ANR-11-BSV2-026).

License GPL (>= 2)

NeedsCompilation no

Repository CRAN

Date/Publication 2017-09-07 10:53:53 UTC

R topics documented:

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

Description

Bios2cor is dedicated to the computation and analysis of correlation/co-variation between positions in multiple sequence alignments and between side chain dihedral angles during molecular dynamics simulations. Features include the ability to compute correlation/co-variation using a variety of scoring functions and to analyze the correlation/co-variation 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.

The main functionalities of `Bios2cor` are summarized below:

(1) CORRELATION/COVARIATION METHODS : Methods that can be used to analyze sequence alignments nad molecular simulations and to calculate a correlation 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.
- `mip`: calculates a score (MI) based on the probability of joint occurrence of events and correct it with the average product correction which is subtracted from the MI score.
- `elsc`: calculates a score based on rigorous statistics of 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) :

- `rotamer_circular`: calculates a correlation 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
- `rotamer_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
- `rotamer_mip`: calculates a score (MI) based on the probability of joint occurrence of rotameric states and correct it with the average product which is subtracted from this MI score.

The methods working with molecular simulations require the following functions :

- `dynamic_struct`: using the result of the `xyz2torsion` function, creates a unique structure that contains side chain dihedral angle informations for each selected frame of the trajectory
- `angle_conversion`: using the result of the `dynamic_struct` 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 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.
- `rotamer_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:

- `gauss_weighting`: given an entropy object, returns a *"gaussian weighting"* for each position of the alignment or each side chain dihedral angle of the protein.
- `sigmoid_weighting`: given an entropy object, returns a *"sigmoidal weighting"* for each position of the alignment or each side chain dihedral angle of the protein.

- `delta_weighting`: given an entropy object, returns a *"delta weighting"* for each position of the alignment or each side chain dihedral angle of the protein.

PCA :

- `centered_pca`: returns a principal component analysis of the correlation matrix passed as a parameter. A weighting filter can be precised, using the weighting functions described above.

(3) OUTPUT FILES : Functions that can be used to produce output files (txt/csv and graphs)

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

- `create_corrfile`: 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)
- `corr_contact`: Using the result of a correlation/covariation method and an integer X, creates a file containing top positions/dihedral angles and their contact counts (number of times the position/dihedral angle appears in the top X pairs with highest scores
- `create_pcafile`: Using the result of the `centered_pca` function, creates a file that contains the coordinates of each position or of each dihedral angle in the principal components
- `write_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 Pymol

Some data can be analyzed thanks to special graphs :

- `create_boxplot`: Using the result of a correlation/covariation method, creates a boxplot to visualize the distribution of the Z-scores
- `create_network`: Using the result of the `corr_contact` function, creates the graph of a network representation of the data with links between correlated/covarying elements
- `entropy_graph`: Using the result of a correlation/covariation method and an entropy structure, creates a graph comparing correlation 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 score (red/pink color for top values, blue/skyblue for bottom values).
- `create_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
- `ang_evo_graph`: 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 X pairs

Details

Package:	BioCor
Type:	Package
Version:	1.5
Date:	2017-03-06
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

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

#Creating OMES object
omes <- omes(align,fileHelix= NULL , diag= 0, fileCSV= NULL, gap_val= 0.8, z_score= TRUE)

#Creating ENTROPY object
entropy <- entropy(align)

#Creating weighting filter
filter <- gauss_weighting(entropy, L= 0.1)

# Creating PCA structures for OMES method and storing in txt file
omes <-omes$normalized
pca <- centered_pca(omes, m= filter, pc= NULL, dec_val= 5,eigenvalues_csv= NULL)
```

---

angle_conversion *Conversion of dihedral angles to rotamers*

Description

Given an object of class 'structure' and an angle 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 to a dihedral angle value, depending on the residue type and the dihedral angle considered. This function will allow to easily compare rotameric changes that occur during the simulations.

Usage

```
angle_conversion(dynamic_struct, conversion_file)
```

Arguments

dynamic_struct	Rotamer structure, result of the ***dynamic_struct*** function

conversion_file

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

Details

In the torsion object and in the conversion file, dihedral angle values are 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.

Author(s)

Antoine GARNIER and Lea BELLENGER

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_rotameres.csv", package= "Bios2cor")

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

my_rotamers <- angle_conversion(dynamic_struct, conversion_file)
```

---

ang_evo_graph *Angle evolution graphs*

Description

Given an object of class 'structure' and names of dihedral angles, creates graphs to monitor the evolution of the dihedral angles along the frames and organizes them in a pdf file.

Usage

```
ang_evo_graph(dynamic_struct, top_positions, filepath)
```

Arguments

`dynamic_struct`	Dihedral angle structure, result of the ***dynamic_struct*** function
`top_positions`	A vector containing the names of the dihedral angles in the top X pairs; Result of *corr_contact* function
`filepath`	Filepath for the output file

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 graphs of the frame dependance of each element included in argument `top_positions`

Author(s)

Antoine GARNIER

Examples

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


#output file
filepath <- "angles.pdf"

#Calculating dynamic data (dihedral angles)
wanted_frames <- seq(from= 1, to= 40, by= 2)
dynamic_struct <- dynamic_struct(pdb, trj, wanted_frames)

wanted_residues <- c("H","N","Q","F","Y","W")


dihed_corr <- rotamer_circular(dynamic_struct, wanted_residues)
dihed_corr <- dihed_corr$normalized

#Select positions of interest (calculated from the "corr_contact" function or list)
top_angles <- corr_contact(dihed_corr, top= 25, "contact_file.txt","top_scores.txt")
my_angles <- unlist(top_angles$positions)

#Creating ang_evol graph
evol_angles <- ang_evo_graph(dynamic_struct, my_angles, filepath)
```

centered_pca *Principal component analysis of a correlation/covariation matrix*

Description

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

Usage

```
centered_pca(mat, m = NULL, pc= 3, dec_val= 3, eigenvalues_csv= NULL)
```

Arguments

mat	A matrix created by a correlation function (omes, mip, elsc, mcbasc, rotamer_circular, rotamer_omes, rotamer_mip)
m	A weighting filter calculated by a weighting function (gauss_weighting, sigmoid_weighting or delta_weighting). 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.
dec_val	A numeric value corresponding to the precision when the *round* function is used.
eigenvalues_csv	A full path name for the csv file where eigen values are stored. Default is NULL (csv file is not created). If not NULL, a png file will also be created with the .csv extension replaced by .png

Details

This function performs a principal component analysis of a correlation/covariation matrix after double centering. It is based on the matrix centering of the *mmds.R* function from the *Bios2mds* package.The elements may have the same weight or different weights. In this case, a weighting filter, corresponding to the relative weight of each matrix element, is passed as a parameter The weighting filter is calculated by a weighting function based on the element entropy.

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 matrix in the PCA space
source	a named list with 2 elements, the correlation matrix (**cor**) and the vector of mass **m** which is used to give a different weight to each element.

Author(s)

Antoine GARNIER

Examples

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

#Creating OMES object
omes <- omes(align,fileHelix= NULL , diag= 0, fileCSV= NULL, gap_val= 0.8, z_score= TRUE)

#Creating ENTROPY object
entropy <- entropy(align)

#Creating weighting filter
filter <- gauss_weighting(entropy, L= 0.1)

# Creating PCA structures for OMES method and storing in txt file
omes <-omes$normalized
pca <- centered_pca(omes, m= filter, pc= NULL, dec_val= 5,eigenvalues_csv= NULL)
#create_pcafile(pca, "pca_results.txt",10,"pc1.txt","pc2.txt")
```

---

corr_contact *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

```
corr_contact(corr, top= 25, contact_filepath= NULL, score_top_filepath= NULL)
```

Arguments

corr	An object created by a correlation function (omes, mip, elsc, mcbasc, rotamer_circular, rotamer_omes, rotamer_mip).
top	A integer indicating the number of top pairs used for this analysis
contact_filepath	The full path name of the output contact file
score_top_filepath	The full path name of the output top score file

Details

This function sorts element pairs by correlation score and analyses the top X pairs to determine the number of pairs (contacts) each element of the top X pairs is involved in.

Value

A vector containing top elements names

Author(s)

Antoine GARNIER

Examples

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

#Creating OMES object
omes <- omes(align,fileHelix= NULL , diag= 0, fileCSV= NULL, gap_val= 0.8, z_score= TRUE)

omes <-omes$normalized

contact <- corr_contact(omes, top= 25, "contact_file.txt","top_scores.txt")
```

---

create_boxplot *Creates boxplot file*

Description

Given correlation objects, creates a boxplot file to compare themselves

Usage

```
create_boxplot(correlation_objects, objects_names, elite, high, filepath)
```

Arguments

correlation_objects	Correlation objects to be compared
objects_names	Correlation objects names
elite	Integer that will define the number of top and bottom values of correlation objects to be taken into account (e.g. top and bottom 25)
high	Integer that will define the number of next top and bottom values of correlation objects to be taken into account (e.g. top and bottom 275)
filepath	The full path name of the output file

Details

Correlation structures contain correlation/covariation scores for each pair of elements [i,j]. The boxplot will compare these structures 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 boxplot comparing different correlation objects

Author(s)

Julien PELE and Antoine GARNIER

Examples

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

  #Creating OMES object
  omes <- omes(align,fileHelix= NULL , diag= 0, fileCSV= NULL, gap_val= 0.8, z_score= TRUE)
  omes <-omes$normalized

  #Creating list for boxplots
    corr_objects <- list(omes)
    corr_objects_names <- c("omes")
    create_boxplot(corr_objects, corr_objects_names, 25, 275,"boxplot.png")
```

---

create_corrfile *Creates a correlation file*

Description

Given a correlation object, stores the correlation score of each pair of elements in a txt file.

Usage

```
create_corrfile(corr_struct, entropy= NULL, auto_pairing= FALSE, filepath)
```

Arguments

corr_struct	An object created by a correlation function (omes, mip, elsc, mcbasc, rotamer_circular, rotamer_omes, rotamer_mip
entropy	An object created by the ***entropy*** function
auto_pairing	Option that allows/removes correlation scores between dihedral angles of the same residue in the correlation matrix (valid only for side chain correlations in molecular simulations)
filepath	The full path name of the output file

Details

The correlation structure contains a score for each pair of elements [i,j]. Elements represent positions in the sequence alignments and side chain dihedral angles in molecular simulations. If entropy = NULL, each line of the correlation file will contain element i, element j and score [i,j]. If entropy is not NULL, each line of the correlation file will contain element i, element j, entropy[i], entropy[j] and score [i,j]

Value

A txt file containing the correlation scores and optionally the entropies

Author(s)

Antoine GARNIER

Examples

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

#Creating OMES object
omes <- omes(align,fileHelix= NULL , diag= 0, fileCSV= NULL, gap_val= 0.8, z_score= TRUE)
omes <-omes$normalized

create_corrfile(omes, entropy= NULL, auto_pairing= FALSE, "corr_results.txt")
```

---

create_entropyfile *Stores entropy values*

Description

Given an entropy object, stores each element (position or dihedral angle) and its entropy value in a csv file.

Usage

```
create_entropyfile(entropy, filepath)
```

Arguments

entropy	An object created by the ***entropy*** or the rotamer_entropyfunction which required an alignment file name
filepath	The full path name of the output file

Details

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

Value

A csv file containing entropy scores

Author(s)

Antoine GARNIER

Examples

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

#Creating ENTROPY object (as an example)
entropy <- entropy(align)

create_entropyfile(entropy, "results_entropy.txt")
```

---

create_network *Creates network structure of top elements*

Description

Given a top_contact structure (result of the corr_contact function), creates a network to visualize the elements involved in the top scoring pairs and their links

Usage

```
create_network(top_contact_struct, filepath)
```

Arguments

top_contact_struct	An object created by the corr_contact function
filepath	The full path name of the output file

Value

A network representing links between elements in the top scoring pairs

Author(s)

Antoine GARNIER

Examples

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

#Creating OMES object
omes <- omes(align,fileHelix= NULL , diag= 0, fileCSV= NULL, gap_val= 0.8, z_score= TRUE)
omes <-omes$normalized

contact <- corr_contact(omes, top= 25, "contact_file.txt","top_scores.txt")

create_network(contact, "network_file.pdf")
```

`create_pcafile` *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

```
create_pcafile(
  pca_struct,
  entropy= NULL,
  filepath,
  top_positions= 10,
  top_pca1= NULL,
  top_pca2= NULL
)
```

Arguments

`pca_struct`	An object created by the `centered_pca` function
`entropy`	An object created by the ***entropy*** or the `rotamer_entropy` function
`filepath`	The full path name of the output file where all coordinates on all components are stored
`top_positions`	An integer corresponding to the number of elements with highest coordinates on the first or on second principal component that will be save if top_pca1 or top_pca2 are not FALSE. DEFAULT is 10
`top_pca1`	The full path name of the output file where the PC1 and PC2 coordinates of the elements with top PC1 coordinates are saved
`top_pca2`	The full path name of the output file where the PC1 and PC2 coordinates of the elements with top PC2 coordinates are saved

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 NaN value for X, Y or Z coordinates will be ignored. Optionally, the function also returns the top 'top_positions' elements with highest absolute coordinates on PC1 and PC2

Value

returns a file containing the coordinates of each element in PC space and, optionally, top elements on PC1 or PC2 with their coordinates on PC1 and PC2

Author(s)

Antoine GARNIER

Examples

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

#Creating OMES object
omes <- omes(align,fileHelix= NULL , diag= 0, fileCSV= NULL, gap_val= 0.8, z_score= TRUE)
omes <-omes$normalized

pca <- centered_pca(omes, m = NULL, pc= NULL, dec_val= 5,eigenvalues_csv= NULL)

create_pcafile(pca, entropy = NULL,"pca_results.txt",10,"pc1.txt","pc2.txt")
```

---

create_screeplot *Creates PCA screeplot*

Description

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

Usage

```
create_screeplot(pca_struct, pca_index= NULL, filepath)
```

Arguments

pca_struct	An object created by the centered_pca function
filepath	The full path name of the output file
pca_index	An integer indicating the number of selected eigenvalues. Its value must not exceed correlation matrix length. DEFAULT is NULL (all positive eigen values are plotted)

Value

A screeplot of selected eigen values

Author(s)

Antoine GARNIER

Examples

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

#Creating OMES object
omes <- omes(align,fileHelix= NULL , diag= 0, fileCSV= NULL, gap_val= 0.8, z_score= TRUE)
omes <-omes$normalized

pca <- centered_pca(omes, m = NULL, pc= NULL, dec_val= 5,eigenvalues_csv= NULL)

create_screeplot(pca, pca_index= NULL, "screeplot.png")
```

---

cyto_entropy *Creation of a entropy file in Cytoscape format*

Description

Given an entropy object, stores entropy scores in a file formatted as a Cytoscape node attribute file

Usage

```
cyto_entropy(entropy, contact= NULL, filepath)
```

Arguments

entropy	An object created by an entropy function
contact	Result of corr_contact function that contains top positions to be taken into account
filepath	The full path name of the output file

Details

This function purpose is storing the entropy scores in a file that will read by Cytoscape as node attributes

Author(s)

Antoine GARNIER

Examples

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

#Creating ENTROPY object and writing results in cytoscape format
entropy <- entropy(align)

cyto_ent <- cyto_entropy(entropy, contact = NULL, "cyto_entropy.txt")
```

`cyto_zscore` *Creation of a Z-score file in Cytoscape format*

Description

Given a correlation object, stores Z-scores in a file formatted as a Cytoscape edge attribute file

Usage

```
cyto_zscore(corr_struct, contact= NULL, filepath)
```

Arguments

`corr_struct`	An object created by a correlation function (`omes`, `mip`, `elsc`, `mcbasc`, `rotamer_circular`, `rotamer_omes`, `rotamer_mip`)
`contact`	Result of `corr_contact` function that contains top positions to be taken into account
`filepath`	The path where the file will be created

Details

This function purpose is storing Z-scores in a file that will read by Cytoscape as node attributes

Author(s)

Antoine GARNIER

Examples

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

#Creating OMES object and writing Z-scores in cytoscape format
omes <- omes(align)

results <- cyto_zscore(omes, contact = NULL, "cyto_Zscore.txt")
```

delta_weighting *Creation of a Delta weighting filter for each element*

Description

Given an entropy object (result of the entropy or of the rotamer_entropyfunction), creates a vector with a delta weighting of each element based on the entropy value. The vector will be used to give different weights to each element in the centered_pca function.

Usage

```
delta_weighting(entropy, Smin= 0.4, Smax= 0.6)
```

Arguments

entropy	An object created by the entropy function
Smin	A value indicating the minimum entropy value. (Smin= 0.4 by default)
Smax	A value indicating the maximum entropy value. (Smax= 0.6 by default)

Details

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

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

Value

A vecto that contains a weighting score for each element (position in sequence alignment or side chain dihedral angle in trajectory).

Author(s)

Antoine GARNIER

Examples

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

#Creating ENTROPY object
entropy <- entropy(align)

filter <- delta_weighting(entropy, Smin= 0.4, Smax= 0.6)
```

`dynamic_struct` *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_struct(pdb, trj, frames)
```

Arguments

`pdb`	Filepath of protein description (pdb file)
`trj`	Filepath of trajectory file (dcd file)
`frames`	Stride length indicating the ratio of selected frames in dcd file

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.pdb`function 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.dcd`function 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,...). Positions for Alanine and Glycine residues which do not have side chain dihedral angles are omitted.
`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 vector indicating the frames that are used (first, last, stride)

Author(s)

Bruck TADDESE and Antoine GARNIER

Examples

```
 #Input files
pdb <- system.file("rotamer/toy_coordinates.pdb", package= "Bios2cor")
trj <- system.file("rotamer/toy_dynamics.dcd", package= "Bios2cor")
conversion_file <- system.file("rotamer/dynameomics_rotameres.csv", package= "Bios2cor")

wanted_frames <- seq(from= 1, to= 40, by= 2)
dynamic_struct <- dynamic_struct(pdb, trj, wanted_frames)
```

---

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

Description

Calculates a score based on rigorous statistics of 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,
  fileHelix= NULL,
  diag= 0,
  fileCSV= NULL,
  gap_val= 0.8,
  double_passing= FALSE,
  z_score= TRUE
)
```

Arguments

align	An object of class 'align' created by the ***import.msf*** or the ***import.fasta*** function from a sequence alignment
fileHelix	A string of characters that indicates the file containing the positions of the anchor residues in the sequence alignment. To be used for the analysis of GPCR sequences. Default is NULL.
diag	A numeric value indicating the score of the diagonal elements in the scoring matrix. Default is 0.
fileCSV	A string of characters indicating the name of the csv file where the output matrix will be saved. Default is NULL.

gap_val	Numeric value indicating the gap ratio at a given position for this position to be taken into account. This value must be between 0 and 0.8. Default is 0.8, which means that positions with more than 80 percent of gaps will not be taken into account.
double_passing	Boolean to calculate correlation score twice : once from first position to last position then from last to first. Results are summed then divided by 2. DEfault is FALSE.
z_score	A boolean for Z-score normalisation of the covariation matrix. Default is TRUE.

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 : a matrix containing the ELSC scores for each pair of elements and, optionally, a matrix containing the Z-scores

Author(s)

Madeline DENIAUD and Antoine GARNIER

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

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

#Creating ELSC object
elsc <- elsc(align)
```

`entropy` *Entropy score*

Description

Measures the entropy score of each position in a sequence alignment

Usage

```
entropy(align, fileHelix= NULL)
```

Arguments

`align`	An object created by the ***import.msf*** function which required an alignment file name
`fileHelix`	A file that contains the positions of the anchor residues in the sequence alignment. Developped for analysis of GPCR sequence alignments

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

References

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

Examples

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

#creating entropy object
#entropy <- entropy(align, fileHelix = helix)
entropy <- entropy(align)
```

---

entropy_graph *Entropy graph*

Description

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

Usage

```
entropy_graph(entropy, corr, filter= NULL, elite, high, csv, name)
```

Arguments

entropy	An object created by the ***entropy*** (sequence alignment) or the ***rotamer_entropy*** (trajectory) function.
corr	An object created by one of the correlation functions:
filter	A vector created by one of the weighting functions (gaussian, delta or sigmoid filter)
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.
high	An integer to determine the number of pairs with the next highest and lowest scores (e.g. 275: pairs ranked 26 to 300 in decreasing or increasing order) to be colored with the "high" color codes.
csv	Logical value that indicates if information must be stored in a csv file
name	The full path name of the graph that will be created

Details

Using the result of a correlation/covariation method and an entropy structure, creates a graph comparing correlation 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 score (blue/red color for elite values, light blue/pink for bottom values).

Value

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

Author(s)

Julien PELE and Antoine GARNIER

Examples

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

#Creating OMES object
omes <- omes(align,fileHelix= NULL , diag= 0, fileCSV= NULL, gap_val= 0.8, z_score= TRUE)
omes <- omes$normalized

#Creating ENTROPY object
entropy <- entropy(align)

#creating entropy graph
entropy_graph(entropy, omes, filter = NULL, elite = 25, high = 275, csv = TRUE,
        name = "entropy_graph.pdf")
```

---

gauss_weighting *Creation of a gaussian weighting filter for each element*

Description

Given an entropy object (result of the entropy or of the rotamer_entropyfunction), creates a vector with a gaussian weighting of each element based on the entropy value. The vector will be used to give different weights to each element in the centered_pca function.

Usage

```
gauss_weighting(entropy, L= 0.1, lambda= 0)
```

Arguments

entropy	An object created by the entropy function
L	A value indicating the gaussian curve's width. It is recommanded to be between 0.05 and 0.2. (0.1 by default)
lambda	A value between 0 and 1 that will be added to the gaussian weighting score to avoid zero problems

Details

The object returned by the entropy function contains an entropy score for each position. The weighting of each position is calculated as follows :

$$weighting[i] = e^{\frac{-(entropy[i]-0.5)^2}{L}}$$

Then the lambda constant is applied as follow :

$$weighting = lambda + (1 - lambda) * weigthing$$

Value

A numeric matrix that contains a weighting score for each position in the alignment based on gaussian function.

Author(s)

Antoine GARNIER

Examples

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

 #Creating ENTROPY object
 entropy <- entropy(align)

 filter <- gauss_weighting(entropy, L= 0.1, lambda= 1)
```

`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.

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

```
# reading of the multiple sequence alignment of human GPCRS 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 (∼) 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

`import.msf` 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

```
# reading of the multiple sequence alignment of human GPCRs 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, fileHelix= NULL, diag= 0, fileCSV= NULL, gap_val= 0.8, z_score= TRUE)
```

Arguments

align	An object of class 'align' created by the ***import.msf*** or the ***import.fasta*** function from a sequence alignment
fileHelix	A string of characters that indicates the file containing the positions of the anchor residues in the sequence alignment. To be used for the analysis of GPCR sequences. Default is NULL.
diag	A numeric value indicating the score of the diagonal elements in the scoring matrix. Default is 0.
fileCSV	A string of characters indicating the name of the csv file where the output matrix will be saved. Default is NULL.
gap_val	Numeric value indicating the gap ratio at a given position for this position to be taken into account. This value must be between 0 and 0.8. Default is 0.8, which means that positions with more than 80 percent of gaps will not be taken into account.
z_score	A boolean for Z-score normalisation of the covariation matrix. Default is TRUE.

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 : a matrix containing a correlation score for each pair of rotamer and its normalized version

Author(s)

Madeline DENIAUD and Antoine GARNIER

References

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

Examples

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

#Creating McBASC object
mcbasc <- mcbasc(align)
```

---

mip *MIp(Mutual Information product) function*

Description

Calculates a mutual information score (MI) based on the probability of joint occurrence of events and corrects it with the average product which is subtracted from the MI score.

Usage

```
mip(align, fileHelix= NULL, diag= 0, fileCSV= NULL, gap_val= 0.8, z_score= TRUE)
```

Arguments

`align`	An object of class 'align' created by the ***import.msf*** or the ***import.fasta*** function from a sequence alignment
`fileHelix`	A string of characters that indicates the file containing the positions of the anchor residues in the sequence alignment. To be used for the analysis of GPCR sequences. Default is NULL.
`diag`	A numeric value indicating the score of the diagonal elements in the scoring matrix. Default is 0.
`fileCSV`	A string of characters indicating the name of the csv file where the output matrix will be saved. Default is NULL.

gap_val — Numeric value indicating the gap ratio at a given position for this position to be taken into account. This value must be between 0 and 0.8. Default is 0.8, which means that positions with more than 80 percent of gaps will not be taken into account.

z_score — A logical value to perform a Z-score normalisation of the covariation matrix (TRUE) or not (FALSE). Default is TRUE.

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 covariation but favors pairs with high entropy.

Value

A list of two elements : a matrix containing the correlation scores for each pair of positions and, optionally, a second matrix with the Z-scores

Author(s)

Madeline DENIAUD and Antoine GARNIER

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

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

#Creating MIP object
mip <- mip(align)
```

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, fileHelix= NULL , diag= 0, fileCSV= NULL, gap_val= 0.8, z_score= TRUE)
```

Arguments

align	An object of class 'align' created by the ***import.msf*** or the ***import.fasta*** function from a sequence alignment
fileHelix	A string of characters that indicates the file containing the positions of the anchor residues in the sequence alignment. To be used for the analysis of GPCR sequences. Default is NULL.
diag	A numeric value indicating the score of the diagonal elements in the scoring matrix. Default is 0.
fileCSV	A string of characters indicating the name of the csv file where the output matrix will be saved. Default is NULL.
gap_val	Numeric value indicating the gap ratio at a given position for this position to be taken into account. This value must be between 0 and 0.8. Default is 0.8, which means that positions with more than 80 percent of gaps will not be taken into account.
z_score	A logical value to perform a Z-score normalisation of the covariation matrix (TRUE) or not (FALSE). Default is TRUE.

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 : a matrix containing a correlation score for each pair of elements and, optionally, a matrix containing the Z-scores

Author(s)

Madeline DENIAUD and Antoine GARNIER

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

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

#Creating OMES object
#omes <- omes(align,fileHelix= NULL , diag= 0, fileCSV= NULL, gap_val= 0.8, z_score= TRUE)
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, filepath)
```

Arguments

pca_struct	An object created by the centered_pca function
abs	An integer which corresponds to the component that will be projected onto the x axis. Default is 1 (first component)
ord	An integer which corresponds to the component that will be projected onto the y axis. Default is 2 (second component)
filepath	The full path name of the png file that will be creared

Value

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

Author(s)

Antoine GARNIER

Examples

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

#Creating OMES object
omes <- omes(align)

#Creating ENTROPY object
entropy <- entropy(align)

# Creating PCA structures for OMES method and storing in txt file
omes <-omes$normalized
pca <- centered_pca(omes, m= NULL, pc= NULL, dec_val= 5,eigenvalues_csv= NULL)
pca_2d(pca, abs= 1, ord= 2, "pca_2d.png")
```

---

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

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

Value

A named list whose objects correspond to random sequences.

Note

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

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

```
# generating a random sequence alignment with the same characterics
# as human GPCRs:
aln <- import.fasta(system.file("msa/toy2_align.fa", package = "Bios2cor"))
nb.seq <- length(aln)
nb.pos <- length(aln[[1]])
aln.random <- random.msa(nb.seq = nb.seq, nb.pos = nb.pos)
```

rotamer_circular *Circular correlation*

Description

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

Usage

```
rotamer_circular(
  dynamic_struct,
  res_selection=
   c("C","I","L","M","V","R","H","K","D","E","N","Q","F","Y","W","T","S","P"),
  z_score= TRUE,
  auto_pairing= FALSE
)
```

Arguments

dynamic_struct	Object of class 'structure' that is created by the ***dynamic_struct*** function
res_selection	Selection of amino acid types that will be taken into account in the covariation matrix. Allows to limit the analysis to a limited selection of amino acid types.
z_score	A logical value for Z-score normalisation of the covariation matrix. Default is TRUE.
auto_pairing	A logical value that maintains (TRUE) or removes (FALSE) covariation scores between dihedral angles within a same residue in the covariation matrix. DEFAULT is FALSE.

Details

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

Value

A list of two elements : a matrix containing a covariation score for each pair of rotamer and its normalized version

Author(s)

Bruck TADESSE and Antoine GARNIER

References

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

Examples

```
#Reading pdb 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_rotameres.csv", package= "Bios2cor")

wanted_residues <- c("H","N","Q","F","Y","W")

wanted_frames <- seq(from= 1, to= 40, by= 2)
dynamic_struct <- dynamic_struct(pdb, trj, wanted_frames)

dihed_corr <- rotamer_circular(dynamic_struct, wanted_residues)
```

`rotamer_entropy` *Entropy score*

Description

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

Usage

```
rotamer_entropy(rotamers)
```

Arguments

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

Details

The entropy score S is computed by summing the number of rotameric changes over all frames, normalized to the number of frames.

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

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_rotameres.csv", package= "Bios2cor")

 #Creating angle_conversion object
 wanted_frames <- seq(from= 1, to= 40, by= 5)
 dynamic_struct <- dynamic_struct(pdb, trj, wanted_frames)

 my_rotamers <- angle_conversion(dynamic_struct, conversion_file)

 #creating rotamer entropy object
 rotamer_entropy <- rotamer_entropy(my_rotamers)
```

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

Description

Calculates a mutual information score (MI) based on the probability of joint occurrence of events and corrects it with the average product which is subtracted from the MI score.

Usage

```
rotamer_mip(
  dynamic_struct,
  rotamers,
  res_selection=
   c("C","I","L","M","V","R","H","K","D","E","N","Q","F","Y","W","T","S","P"),
  z_score= TRUE,
  auto_pairing= FALSE
)
```

Arguments

`dynamic_struct`	An object of class 'structure' that is created by the ***dynamic_struct*** function
`rotamers`	A character matrix of type 'rotamers' that is produced by the angle_conversion function. The matrix indicates the rotameric state of each side chain dihedral angle for each frame of the trajectory.
`res_selection`	Selection of amino acid types that will be taken into account in the covariation matrix. Allows to limit the analysis to a limited selection of amino acid types.
`z_score`	A logical value for Z-score normalisation of the covariation matrix. Default is TRUE.

auto_pairing A logical value that maintains (TRUE) or removes (FALSE) covariation scores between dihedral angles within a same residue in the covariation matrix. DEFAULT is FALSE.

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.

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

Value

A list of two elements : a matrix containing a correlation score for each pair of rotamer and its normalized version

Author(s)

Antoine GARNIER and Madeline DENIAUD

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

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

conversion_file <- system.file("rotamer/dynameomics_rotameres.csv", package= "Bios2cor")

wanted_residues <- c("H","N")

wanted_frames <- seq(from= 5, to= 40, by= 15)
```

```
dynamic_struct <- dynamic_struct(pdb, trj, wanted_frames)

my_angles <- angle_conversion(dynamic_struct, conversion_file)

#Creating MIP object
mip_corr <- rotamer_mip(dynamic_struct, my_angles, wanted_residues)
```

`rotamer_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

```
rotamer_omes(
  dynamic_struct,
  rotamers,
  res_selection=
   c("C","I","L","M","V","R","H","K","D","E","N","Q","F","Y","W","T","S","P"),
  z_score= TRUE,
  auto_pairing= FALSE
)
```

Arguments

`dynamic_struct`	An object of class 'structure' that is created by the ***dynamic_struct*** function
`rotamers`	A character matrix that is produced by the angle_conversion function. The matrix indicates the rotameric state of each side chain dihedral angle for each frame of the trajectory.
`res_selection`	Selection of amino acid types that will be taken into account in the covariation matrix. Allows to limit the analysis to a limited selection of amino acid types.
`z_score`	A logical value for Z-score normalisation of the covariation matrix. Default is TRUE.
`auto_pairing`	A logical value that maintains (TRUE) or removes (FALSE) covariation scores between dihedral angles within a same residue in the covariation matrix. DEFAULT is FALSE.

Details

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

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

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

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 one or two elements which are a numeric matrix containing the correlation scores for each pair of rotamers and, optionaly, a numeric matrix containing the Z-scores

Author(s)

Antoine GARNIER, Madeline DENIAUD and Lea Bellenger

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

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

conversion_file <- system.file("rotamer/dynameomics_rotameres.csv", package= "Bios2cor")

wanted_residues <- c("W")

nb_frames_wanted <- 40
wanted_frames <- seq(from= 5, to= nb_frames_wanted, by= 10)
dynamic_struct <- dynamic_struct(pdb, trj, wanted_frames)

my_angles <- angle_conversion(dynamic_struct, conversion_file)

#Creating OMES object
omes_corr <- rotamer_omes(dynamic_struct, my_angles, wanted_residues, z_score= FALSE)
```

shuffle_all *Amino acids shuffle*

Description

Given an alignment object, created by the `import.fasta` or `import.msf` function, shuffles all the residues from all the sequences to create a random multiple sequence alignment and calculates a correlation matrix with the selected method.

Usage

```
shuffle_all(
  align,
  method,
  fileHelix= NULL,
  gap_val= 0.8,
  z_score= TRUE,
  nb_iterations= 5
)
```

Arguments

`align`	An object created by the ***import.msf*** or ***import.fasta*** function which requires an alignment file name
`method`	A string corresponding to selected correlation method. This should be one out of "OMES", "MIP","ELSC" and "MCBASC"
`fileHelix`	A file that contains the positions of the anchor residues in the sequence alignment. For use with GPCRs
`gap_val`	Authorized gap proportion per position. This value can be between 0 and 0.8 (0.8 by default), which means that positions with more than 80 percent of gaps will not be considered
`z_score`	A logical value to perform a Z-score normalisation of the covariation matrix (TRUE) or not (FALSE). Default is TRUE
`nb_iterations`	Number of times the alignment will be shuffled. Default is 5

Details

Sequences from the initial alignment will be shuffled to yield a random alignment with the same average composition as the initial alignment. Ther correlation matrix of this random alignment will be calculated "nb_iterations" times, then averaged.

Value

Returns a numeric matrix which is the averaged matrix obtained from re-itering "nb_iterations" times the computation of the correlation matrix from a random alignment with the same composition as the initial alignment

Author(s)

Antoine GARNIER

Examples

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

#Shuffling alignment
corr <- shuffle_all(align, method = "OMES", fileHelix= NULL, gap_val= 0.8, z_score= TRUE,
      nb_iterations= 3)
```

`shuffle_positions` *Amino acids shuffle*

Description

Given an alignment object, created by the `import.fasta` or `import.msf` function, for each position in the alignment, shuffles the residues at this position to create a random alignment, and calculates the correlation matrix with the selected method.

Usage

```
shuffle_positions(
  align,
  method,
  fileHelix= NULL,
  gap_val= 0.8,
  z_score= TRUE,
  nb_iterations= 5
)
```

Arguments

`align`	An object created by the ***import.msf*** or ***import.fasta*** function which requires an alignment file name
`method`	A string corresponding to selected correlation method. This should be one out of "OMES", "MIP","ELSC" and "MCBASC"
`fileHelix`	AA file that contains the positions of the anchor residues in the sequence alignment. For use with GPCRs
`gap_val`	Authorized gap proportion per position. This value can be between 0 and 0.8 (0.8 by default), which means that positions with more than 80 percent of gaps will not be considered
`z_score`	A logical value to perform a Z-score normalisation of the covariation matrix (TRUE) or not (FALSE). Default is TRUE
`nb_iterations`	Number of times that alignment must be shuffled. Default is 5

Details

At each position, sequences from the initial alignment will be shuffled to yield a random alignment with the same average composition at each position as the initial alignment and the same sequence entropy profile. The correlation matrix of the resulting alignment will be calculated "nb_iterations" times, then averaged.

Value

Returns a numeric matrix which is the averaged matrix obtained from re-itering "nb_iterations" times the computation of the correlation matrix from a random alignment with the same composition as the initial alignment

Author(s)

Antoine GARNIER

Examples

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

#Shuffling alignment
corr <- shuffle_positions(align, method="OMES", fileHelix= NULL, gap_val= 0.8, z_score= TRUE,
     nb_iterations= 3)
```

---

sigmoid_weighting *Sigmoid weighting for each position*

Description

Given an entropy object (result of the entropy or of the rotamer_entropyfunction), creates a vector with a sigmoid weighting of each element based on the entropy value. The vector will be used to give different weights to each element in the centered_pca function.

Usage

```
sigmoid_weighting(entropy, L= 10, inf_pt= 0.1, stability= TRUE)
```

Arguments

entropy	An object created by the entropy function
L	A value indicating the slope on the inflexion point. A value between 5 and 10 is recommanded. Default is 10
inf_pt	A value between 0 and 1 corresponding to the sigmoid curve inflexion point. Default is 0.1
stability	Logical value indicating whether sigmoid function favors low (TRUE) or high (FALSE) entropy values. Default is TRUE

Details

The object returned by the entropy function contains an entropy score for each position. The weighting of each position is calculated as follows :

$$weighting[i] = \begin{cases} \frac{1}{(1+e^{(-L*(entropy[i]-inf_pt)))}}, & "stability" = FALSE \\ 1 - (\frac{1}{(1+e^{(-L*(entropy[i]-inf_pt)))}}), & "stability" = TRUE \end{cases}$$

This option is useful to limit analysis to side chain dihedral angles with slow or fast moving angles (TRUE or FALSE stability option, respectively).

Value

A numeric matrix that contains a weighting score for each position in the alignment based on sigmoid function.

Author(s)

Antoine GARNIER

Examples

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

#Creating ENTROPY object
entropy <- entropy(align)

filter <- sigmoid_weighting(entropy, L= 10, inf_pt= 0.1, stability= TRUE)
```

---

write_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_pdb(pca, trio_comp= c(1:3), pdb_filepath, pml_filepath)
```

Arguments

pca	An object created by the centered_pca function
trio_comp	A numeric vector of length 3 indicating the principal components to be displayed. Default is c(1, 2, 3).
pdb_filepath	A string of characters indicating the full path name of the output PDB file.
pml_filepath	a string of characters indicating the full path name of the output PML file for visualization with Pymol.

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

Examples

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

#Creating OMES object
omes <- omes(align)
omes <- omes$normalized

# Creating PCA structures for OMES method
pca <- centered_pca(omes, m=NULL, pc= NULL, dec_val= 5,eigenvalues_csv= NULL)

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

---

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