Package 'BALCONY'

February 28, 2019

Type Package

Title Better ALignment CONsensus analYsis

Version 0.2.10

Maintainer Michal Stolarczyk <stolarczyk.michal93@gmail.com>

Description Facilitates the evolutionary analysis and structure conservation study of specified amino acids in proteins.

BugReports https://github.com/michalstolarczyk/BALCONY/issues

License GPL

Encoding UTF-8

LazyData true

Imports seqinr, Rpdb, scales, stats, base, dplyr, Biostrings, readr, progress

RoxygenNote 6.1.1

NeedsCompilation no

Author Michal Stolarczyk [aut, cre], Alicja Pluciennik [aut]

Repository CRAN

Date/Publication 2019-02-28 16:00:03 UTC

R topics documented:

alignment 2
alignment2matrix 3
align_params 4
align_seq_mtx2grs 5
barplotshow 6
calculate_AA_variation 7
calculate_pseudo_counts 8
compare_cons_metrics 9
cons2seqs_ident 10
cons2seqs_sim 11

consensus . . . 12
convert_AA_symbol . . . 13
create_final_CSV . . . 14
create_structure_seq . . . 15
CRE_conservativity . . . 17
delete_isoforms . . . 18
D_matrix . . . 19
Escore_conservativity . . . 20
excl_low_prob_strcts . . . 21
find_consecutive_seq . . . 22
find_seq . . . 23
find_seqid . . . 23
get_pos_based_seq_weights . . . 24
get_prot_entropy . . . 25
get_remarks465_pdb . . . 26
get_seq_names . . . 27
get_seq_weights . . . 28
get_structures_entropy . . . 29
get_structures_idx . . . 30
gonnet . . . 31
is_upper . . . 32
kabat_conservativity . . . 32
kolmogorov_smirnov_test . . . 33
landgraf_conservativity . . . 35
noteworthy_seqs . . . 36
pairwise_alignment_MSA . . . 37
plot_entropy . . . 37
plot_structure_on_protein . . . 38
prepare_structure_profile . . . 40
preprocess_hmm_output . . . 41
read_structure . . . 42
RealValET_conservativity . . . 43
schneider_conservativity . . . 44
shannon_conservativity . . . 45
small_alignment . . . 46
structure . . . 46
substitution_mtx . . . 47

Index **48**

`alignment` *Sample alignment of soluable epoxide hydrolase family*

Description

was performed on a dataset comprising 311 soluble epoxide hydrolase peptide orthologous sequences acquired from UniProtKB. The alignment was performed and edited with MUSCLE algorithm in JALVIEW, respectively.

Format

An alignment object read with `read.alignment` function from seqinr package.

alignment$nb A numeric: number of sequences

alignment$nam A vector of characters: names of the sequences

alignment$seq A vector of characters: amino acid sequences

References

MUSCLE: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-5-113
JALVIEW: https://academic.oup.com/bioinformatics/article/25/9/1189/203460/Jalview-Version-2-a-multi

Examples

```
data("alignment")
alignment
```

---

alignment2matrix *Load alignment into matrix*

Description

The function loads alignment into matrix to facilitate a convenient data manipulation

Usage

```
alignment2matrix(alignment)
```

Arguments

`alignment`	data loaded with `read.alignment` function

Value

`matrix`	Aligned sequences matrix where number of rows equals to number of aligned sequences and number of columns equals to the length length of aligned sequences

Author(s)

Alicja Pluciennik & Michal Stolarczyk

See Also

`align_params`, `read.alignment`

Examples

```
data("alignment")
alignment = delete_isoforms(alignment)
matrix=alignment2matrix(alignment)
```

`align_params` *Get alignment dimensions*

Description

This function returns size of alignment, which facilitates the convenient performing upcoming steps of analysis.

Usage

```
align_params(alignment)
```

Arguments

`alignment`	alignment loaded with `read.alignment`

Details

Function returns list of two elements row_no(number of rows, sequences) and col_no(number of columns,length of aligned sequences)

Value

`row_no`	number of sequences
`col_no`	length of aligned sequences

Author(s)

Alicja Pluciennik & Michal Stolarczyk

References

seqinr

See Also

`read.alignment`

Examples

```
data("alignment")
parameters=align_params(alignment);
```

`align_seq_mtx2grs` *Convert amino acid symbols to groups according to their properties of user's choice.*

Description

This function performs a conversion of amino acid symbols to group symbols according to their properties. Implemented grouping methods are: substitution_matrix (majority of properties taken into account), polarity, size and aromaticity. "GX", where X stands for group number, are group symbols.

Usage

```
align_seq_mtx2grs(aligned_sequences_matrix,grouping_method)
```

Arguments

`aligned_sequences_matrix`
: A matrix that contains aligned sequences. It is an output of `alignment2matrix` function

`grouping_method`
: A string which specifies the grouping method to be used. One of following: 'substitution_matrix', 'polarity', 'size', 'aromaticity'

Value

`grouped_aligned_sequences_matrix`
: A matrix of size of the input matrix but with group symbols instead of amino acid symbols

Author(s)

Alicja Pluciennik & Michal Stolarczyk

See Also

`alignment2matrix`, `read.alignment`

Examples

```
data(alignment)
alignment = delete_isoforms(alignment)
grouping_method = "general"
aligned_sequences_matrix = alignment2matrix(alignment)
grouped = align_seq_mtx2grs(aligned_sequences_matrix,grouping_method)
```

barplotshow *Shows barplot with amino acid variation on the specified position*

Description

This function facilitates a visual inspection of multiple sequence alignment (MSA) position variablity.

Usage

```
barplotshow(position, AA_variation)
```

Arguments

position	A number of column of alignment to be visualized
AA_variation	A percentage frequency of amino acids in the alignment, calculated with calculate_AA_variation function

Value

This function produces a barchart

Author(s)

Alicja Pluciennik & Michal Stolarczyk

See Also

calculate_AA_variation

Examples

```
data("small_alignment")
position = 100
threshold = 0.01
var_aa = calculate_AA_variation(small_alignment,threshold)
barplotshow(position, var_aa)
```

`calculate_AA_variation` *Calculate AA variations on each position of the multiple sequence alignment*

Description

This function calculates AA variations on each position of the alignment which may be further used for the conservativity study of the set of sequences in quiestion

Usage

```
calculate_AA_variation(alignment, threshold, grouped,
  grouping_method, weights, pseudo_counts=F)
```

Arguments

`alignment`	The data loaded with `read.alignment` function
`threshold`	(optional) A number in range 0-1. A of minimal frequency of occurences of amino acids at each position. Default: all the residues are visualized.
`grouped`	(optional) A logical indicating if the grouping of amino acids should be applied. Default: FALSE
`grouping_method`	(optional) A string which specifies the grouping method to be used. One of following: 'substitution_matrix', 'polarity', 'size', 'aromaticity'. Default: 'substitution_matrix'. Default: 'substitution_matrix' if grouped is TRUE.
`weights`	(optional) A vector of length equal number of sequences in the alignment object with weights to overcome the taxonomic bias in the conservation analysis.
`pseudo_counts`	(optional) A logical indicating if pseudo-counts should be added to the MSA. Pseudo-counts can be used only in non-group mode and without weights. Using these options with pseudo-counts will be suppressed. Default: FALSE

Details

The output consists of amino acids and their fractions on each position of alignment. Amino acids with occurence frequencies lower than the threshold of user's choice are excluded.

Value

Returns list of thre matrices with tabelarized symbols of the most common AA in alignment column, percentage values for contributed AA and combined one.

`var_aa$AA`	A matrix of AA on all alignment positions with decreasing frequencies in columns
`var_aa$per`	The percentage of AA frequencies corresponding to the $AA
`var_aa$matrix`	A combination of this two. The best suited element for visual inspection of the variability at each position

Author(s)

Michal Stolarczyk & Alicja Pluciennik

See Also

align_params,calculate_pseudo_counts

Examples

```
data("small_alignment")
alignment = delete_isoforms(small_alignment)
threshold=10
grouped = FALSE
var_aa=calculate_AA_variation(small_alignment, threshold, grouped)
```

---

calculate_pseudo_counts

Calculate pseudo counts for alignment

Description

This function calculates pseudo-counts (as shown in Henikoff et al. (1996)) for an alignment with the use of substitution matrices. It is recommended to estimate amino acid frequencies for alignments with small number of sequences (in order to calculate reliable entropy scores)

Usage

```
calculate_pseudo_counts(alignment, substitution_mtx)
```

Arguments

alignment	alignment loaded with read.alignment
substitution_mtx	Matrix with amino acids substitution frequencies. Default: GONNET

Value

pseudoCounts	Matrix with pseudo counts of size 21x number of alignment columns

Note

Please note that when using other scoring matrix user needs to make sure that all alignment symbols are present there. Missing symbol will issue an error.

Author(s)

Alicja Płuciennik & Michał Stolarczyk

References

Henikoff et al.(1996) Using substitution probabilities to improve position-specific scoring matrices, Bioinformatics, 12, 135–143
Claverie (1994) Some useful statistical properties of position-weight matrices. Comput. Chem., 18, 287-293

Examples

```
data("alignment")
PC <- calculate_pseudo_counts(alignment)
```

`compare_cons_metrics` *compare_cons_metrics*

Description

This function is designed to compare the conservation metrics used in the analysis. This way the user can notice the significant correlation or differences between these to evaluate their performance in a specific case.

Usage

```
compare_cons_metrics(protein_entropy, structure_profile, pdb_name)
```

Arguments

`protein_entropy`	List of entropy scores values for a whole protein (output of `get_prot_entropy`).
`structure_profile`	Each element is a list of entropy values (matrix of entropy scores) and indices of residues building structure in protein of interest (output of `prepare_structure_profile`).
`pdb_name`	The name of the analyzed protein.

Details

This function allows to show the scatterplots of an entropy scores. The protein is marked as gray points, the structures are marked with symbols. It is useful to visualise differences between entropy scores, and choose the best one for further analysis.

Value

This function produces a set of scatter plots facilitating the visual inspection of entropy metrics dependancies.

Author(s)

Alicja Pluciennik & Michal Stolarczyk

Examples

```
data("alignment")
alignment = delete_isoforms(alignment)
data("structure")
uniprot="P34913"
indices=get_structures_idx(structure)
protein_index = indices$proteinIndices
structure_index = indices$structureIndices
entropy_scores_list=list(
                        Schneider_entropy = schneider_conservativity(alignment),
                        Escore_entropy = Escore_conservativity(alignment)
                        )
structure_entropy=get_structures_entropy(structure_index, entropy_scores_list)

structure_profile = prepare_structure_profile(structure, structure_entropy)
protein_entropy=get_prot_entropy(protein_index, entropy_scores_list)
compare_cons_metrics(protein_entropy, structure_profile, "1CQZ")
```

---

cons2seqs_ident *Identity of each sequence in the alignment to the consensus sequence.*

Description

The function calculates identity of consensus to each sequence in the alignment. It facilitates an assessment of consensus accuracy and identification of outlying sequences in the alignment. Also, it can be used to weight conservativity metrics results in further steps of analysis with BALCONY package.

Usage

```
cons2seqs_ident(alignment, consensus_seq)
```

Arguments

`alignment`	Data loaded with `read.alignment` function
`consensus_seq`	Consensus sequence (output of `consensus` function)

Details

Returned values are percentage of identical symbols (AA and "-") in consensus sequence and aligned sequence.

Value

`percentage`	Numeric vector of identity score (percentage); positions in the numeric vector correspond to sequences in alignment, respectively

Author(s)

Alicja Pluciennik & Michal Stolarczyk

See Also

consensus cons2seqs_sim

Examples

```
data("alignment")
alignment = delete_isoforms(alignment)
threshold=60
consensus=consensus(alignment, threshold)
true_consensus=cons2seqs_ident(alignment, consensus)
```

---

cons2seqs_sim *Group consensus to each sequence in the alignment similarity*

Description

The function calculates similarity of group consensus to each sequence in the alignment. It facilitates an assessment of consensus accuracy and identification of outlying sequences in the alignment. Grouping amino acids allows to check similiarity between sequences by amino acids properties of user's choice.

Usage

```
cons2seqs_sim(grouped_alignment, grouped_consensus_seq)
```

Arguments

grouped_alignment	The output of read.alignment function
grouped_consensus_seq	A string of amino acids, the output of consensus function

Details

AA in consensus sequences and aligned sequences are converted into groups symbols according to method of user's choice. Returned values are percentage of similar amino acids considering the properties in consensus sequence and aligned sequence.

Value

percentage	numeric vector of identity score (percentage); positions in the numeric vector correspond to sequences in alignment, respectively

Author(s)

Alicja Pluciennik & Michal Stolarczyk

See Also

read.alignment, consensus, align_params

Examples

```
data("small_alignment")
alignment = delete_isoforms(small_alignment)
threshold_consensus = 30
grouping_method = "substitution_matrix"
alignment_grouped = align_seq_mtx2grs(alignment2matrix(alignment),grouping_method)
consensus_seq_grouped = consensus(alignment_grouped, threshold_consensus)
consensus_to_seqs_similarity = cons2seqs_sim(alignment_grouped, consensus_seq_grouped)
```

---

consensus *Consensus sequence determination*

Description

Function calculates consensus sequence for given alignment with a threshold of user's choice.

Usage

```
consensus(alignment, threshold)
```

Arguments

alignment	output of of read.alignment function or grouped alignment created with: align_seq_mtx2grs and alignment2matrix
threshold	minimal fraction of amino acids on the certain position in all sequences of the alignment to be taken for consensus letter on this position; number in range 0-100.

Details

If maximum fraction of any amino acid on the certain position is lower than a threshold then "*" is printed instead.

Value

consensus_sequence

A character vector of length of the aligned sequence containing consesus sequence based on the input alignment

Note

Please note that this function masks the seqinr package function consensus

Author(s)

Alicja Pluciennik & Michal Stolarczyk

See Also

s2c

Examples

```
data("alignment")
alignment = delete_isoforms(alignment)
threshold=80 # Set the consensus threshold
consensus_sequence=consensus(alignment, threshold)
```

---

convert_AA_symbol *Amino acids symbols conversion*

Description

This function facilitates the conversion of three letter amino acids' codes to one letter equivalents.

Usage

```
convert_AA_symbol(amino_acids)
```

Arguments

amino_acids	A character or vector of characters with amino acid(s) three letter code(s)

Details

In case a vector of amino acid three letter codes is provided the function returns a vector of their one letter equivalents.

Value

A chracter or vector of characters with amino acids one letter code(s)

Author(s)

Michal Stolarczyk & Alicja Pluciennik

Examples

```
three_letter_codes = c("LEU", "VAL", "ALA")
convert_AA_symbol(three_letter_codes)
```

`create_final_CSV` *Create CSV file to save results*

Description

Create_final_CSV() saves results as table into csv file. Combination of given variation allows to compare protein structure with evolutionary data content from alignment. Each position on alignment has its own column in csv file. If the length of the alignmnet exceeds 1000 characters, the output is divided into separate files with suffixes corresponing to the number of file produced by this function.

Usage

```
create_final_CSV(filename,variations_matrix,structure,sequence_id,alignment,score_list)
```

Arguments

`filename`	name of the output file produced by the function
`variations_matrix`	An object which contains alignment and frequencies of occurences each amino acids on each position of alignment. Output of `calculate_AA_variation`
`structure`	An strcture object - matrix of aligned, examined protein sequence covered by structure markers (S/N). Output of `create_structure_seq`.
`sequence_id`	the Uniprot code of the sequence of interest
`alignment`	the output of `read.alignment` function. A variable containing alignment data. One of the sequences must be the sequence of interest
`score_list`	list of calculated entropy/conservation scores. Optional parameter. If not provided, this rows are not present in the output file

Value

`csv_file`	A comma separated variable file containing information provided to this function. It is also written in the current directory.

Author(s)

Alicja Pluciennik & Michal Stolarczyk

See Also

`create_structure_seq`, `schneider_conservativity`, `Escore_conservativity`, `landgraf_conservativity`, `read.alignment`

Examples

```
data("alignment")
data("structure")
uniprot="P34914"
alignment = delete_isoforms(alignment)
threshold = 1
var_aa=calculate_AA_variation(alignment,threshold)
entropy_data=list(Schneider.entropy=schneider_conservativity(alignment),
                  Escore.entropy = Escore_conservativity(alignment))
create_final_CSV("my_filename",var_aa,structure,uniprot,alignment,entropy_data)
```

`create_structure_seq` *Superimpose structural data of interest on sequence after the alignment*

Description

Create sequence of a protein structure model based on numbers of amino acids given in a text file (list of IDs and numbers in protein)

Usage

```
create_structure_seq(structure_list, sequence_id, alignment,
                     pdb_path = NULL, chain_identifier = NULL, shift = NULL)
```

Arguments

`structure_list`	A list of structure data used for further evolutionary analysis. It can be text file(s) read by the `read_structure` function (text file with 2 columns: numbers of amino acids and 3-letters codes of AA; First row needs to contain markers)
`sequence_id`	The id/name of the target sequence in alignment which will be a base of structure sequence
`alignment`	An alignment object read with `read.alignment` function, must contain the target sequence
`pdb_path`	A string specifying the path to the PDB file with structural information. Optional parameter, required if the structure is incomplete e.g. fragments such as loops are missing
`chain_identifier`	A character specifying the chain of interest e.g. "A" or "B"
`shift`	A numeric value. In case there is a need to adjust the amino acids numeration due to missing amino acids at the beginning of the structure (that are not considered in the PDB file REMARK465 section)

Details

This function is useful to create sequence covered with structural data provided in a .txt file. This sequence can be compared with alignment to check the conservation for interesting amino acid(s). Additionally, if path to the PDB file is provided the function corrects the output accordingly to the information in REMARK465 on missing amino acids.

Value

`structure_matrix`
: A matrix of characters "S" and "N" marking on sequence the structural element; "S" - amino acid forms the analyzed structure, "N" - amino acid which does not form the structure. Number of rows of the matrix corresponds to the number of structures analyzed

`structure_numbers`
: A vector containing the numbers of the amino acids in the sequence of interest (no gaps)

`structure_probabilities`
: A matrix of numeric values: probabilities of corresponding to the structural information from first element of the output, which helps to reduce the effect of non-consistent structural amino acids on the conservativity analysis of the structure of interest

Author(s)

Alicja Pluciennik & Michal Stolarczyk

See Also

`get_remarks465_pdb`, `find_consecutive_seq`, `read_structure`, `read.alignment`

Examples

```
data("alignment")
structure_files = c(system.file("extdata", "T1_4JNC.structure", package = "BALCONY"),
                    system.file("extdata", "T2_4JNC.structure", package = "BALCONY"),
                    system.file("extdata", "T3_4JNC.structure", package = "BALCONY")
                    )
structure_list = read_structure(structure_files)
#creating library uniprot - PDB
lib=list(c("Q84HB8","4I19","4QA9"),
        c("P34913","4JNC"),
        c("P34914","1EK2","1CR6","1EK1","1CQZ"))
pdb_name = "4JNC"
uniprot=find_seqid(pdb_name,lib)
tunnel=create_structure_seq(structure_list,uniprot,alignment)
```

CRE_conservativity *Calculate cumulative relative entropy score*

Description

This function calculates cumulative relative entropy score according to: Hannenhalli and Russell (2000).

Usage

```
CRE_conservativity(alignment, hmmbuild_path=NULL, pairwiseAlignemnt_scores=NULL)
```

Arguments

alignment	An alignment object read with read.alignment function
hmmbuild_path	(optional if running under UNIX) The aboslute path to the hmmbuild binary
pairwiseAlignemnt_scores	(optional) A matrix with pairwise alignment scores. For example created by pairwiseAlignment. If the matrix is not provideded by the user it is calculated automatically by the function (time consuming). The sequences are extracted from the alignemnt object.

Details

PSEUDO-ALGORITHM (According to Hannenhalli and Russell (2000)):

1. (If score matrix is not provided) Run pairwise alignments for all available sequences in the input MSA and save scores to a matrix
2. (If score matrix is not provided) Calculate a distance matrix based off of the alignment scores one
3. Perform hierarchical clustering on the distance matrix (UPGMA method)
4. Get the sequence clusters
5. Divide the alignment into sub_groups which are the clusters
6. Run hmmbuild for whole_alignment without sub-group and sub_group
7. Calculate relative entropy using these two as indicated in the Reference and repeat for each sub_group
8. Calculate the cumulative relative entropy

hmmbuild program:

This function uses hmmbuild program of HMMER suite for HMM profile generation for MSA.

We recommend downloading and installing HMMER by following the instructions and steps in the HMMER installation website .

Value

`score` A vector of length equal to the length of aligned sequences

Author(s)

Michal Stolarczyk & Alicja Pluciennik

References

Hannenhalli, S. S. & Russell, R. B. Analysis and prediction of functional sub-types from protein sequence alignments11Edited by J. Thornton. Journal of Molecular Biology 303, 61–76 (2000).

See Also

`consensus`, `cons2seqs_ident`, `read.alignment`

Examples

```
#No example due to external software requirements
```

`delete_isoforms` *Delete protein isoforms from alignment object*

Description

This function searches for isoforms in the alignment object (entries with "-digitl" in the name) and deletes them

Usage

```
delete_isoforms(alignment)
```

Arguments

`alignment` An object (S3) class alignment read with `read.alignment` function

Details

The isoforms are detected as entries with `"-digit|"` in the sequence name. If no isoforms are detected this function prints a "No isiforms detected" notification instead

Value

Alignment without isoforms - an object (S3) class alignment

Author(s)

Michal Stolarczyk & Alicja Pluciennik

See Also

read.alignment

Examples

```
data("alignment")
delete_isoforms(alignment)
```

D_matrix *Calculate substitution rate matrix between two amino acids*

Description

This function is used to calculate Landgraf conservation metric. D_matrix contains substitution rates between two amino acids in the alignment, according to the following formula:

$$D(a,b) = (d(a,a) - d(a,b))/d(a,a)$$

where:
$d(a,a)$ is a probability of AA substitution by itself
$d(a,b)$ is a probability of substitution of amino acid a with other amino acid.

Usage

```
D_matrix(substitution_matrix)
```

Arguments

substitution_matrix	A matrix with probablity of substitutions, e.g. Gonnet substitution matrix

Value

distance	A matrix of substitution probablities for all amino acids

Author(s)

Alicja Pluciennik & Michal Stolarczyk

Examples

```
data("gonnet")
distance=D_matrix(gonnet)
```

`Escore_conservativity` *Calculate the Escore conservation metric*

Description

This function facilitates the calculation of Escore conservation metric (in amino acid or group mode)

Usage

```
Escore_conservativity(alignment, grouping_method = NULL, weights = NULL,pseudo_counts=F)
```

Arguments

`alignment`	Alignment data read with `read.alignment` function
`grouping_method`	(optional) A string which specifies the grouping method to be used. One of following: 'substitution_matrix', 'polarity', 'size', 'aromaticity', default: NULL
`weights`	(optional) A vector of length equal number of sequences in the alignment object with weights to overcome the taxonomic bias in the conservation analysis.
`pseudo_counts`	(optional) A logical indicating if pseudo-counts should be added to the MSA. Pseudo-counts can be used only in non-group mode and without weights. Using these options with pseudo-counts will be suppressed. Default: FALSE

Details

The conservativity score is calculated according to the following formula:

$$P(i) = max(p(i))/n(i)$$

$$Pnorm(i) = P(i)/max(P)$$

$$score = -ln(P_norm(i))/max(-ln(P_norm))$$

where:
$p(i)$ - amino acids frequency on i-th position where gaps are included
$n(i)$ - amino acids count on i-th position where gaps are excluded

Value

`conservation_score`
A vector of length equal to the length of aligned sequences

Note

Also, this function originally calculates the entropy values which can be used to estimate the conservativity score according to the following formula:

$$conservation = 1 - entropy$$

Author(s)

Alicja Pluciennik & Michal Stolarczyk

Examples

```
data("small_alignment")
conservation_score = Escore_conservativity(alignment)
```

---

excl_low_prob_strcts *Exclude low probability structural data*

Description

This function facilitates the exclusion of low probability structural data from the downstream conservativity analysis, which helps to reduce the effect of non-consistent structural amino acids on the conservativity analysis of the structure of interest

Usage

```
excl_low_prob_strcts(structure, threshold)
```

Arguments

structure	A structure object generated with create_structure_seq function
threshold	The threshold for the structural data exclusion

Value

structure_matrix
: A matrix of characters "S" and "N" marking on sequence the structural element; "S" - amino acid forms the analyzed structure, "N" - amino acid which does not form the structure. Number of rows of the matrix corresponds to the number of structures analyzed

structure_numbers
: A vector containing the numbers of the amino acids in the sequence of interest (no gaps)

structure_probabilities
: A matrix of numeric values: probabilities of corresponding to the structural information from first element of the output

Author(s)

Michal Stolarczyk & Alicja Pluciennik

See Also

create_structure_seq

Examples

```
data("alignment")
structure_files = c(system.file("extdata", "T1_4JNC.structure", package = "BALCONY"),
                    system.file("extdata", "T2_4JNC.structure", package = "BALCONY"),
                    system.file("extdata", "T3_4JNC.structure", package = "BALCONY")
                    )
structure_list = read_structure(structure_files)
#creating library uniprot - PDB
lib=list(c("Q84HB8","4I19","4QA9"),
        c("P34913","4JNC"),
        c("P34914","1EK2","1CR6","1EK1","1CQZ"))
pdb_name = "4JNC"
uniprot=find_seqid(pdb_name,lib)
tunnel=create_structure_seq(structure_list,uniprot,alignment)
tunnel_excluded = excl_low_prob_strcts(tunnel, 0.5)
```

---

find_consecutive_seq *Find sequences of numbers in a numeric vector*

Description

This function finds sequences of consecutive numbers in numeric vectors

Usage

```
find_consecutive_seq(vector)
```

Arguments

vector	A numeric vector to be analyzed

Details

Out of the following vector: 1,2,3,4,5,6,7,20,21,140,141 the function will find values starting the sequences: 1,20,140 and their lengths 7,2,2 respectively

Value

values	A vector of values starting the consecutive sequences
lengths	A vector of lengths of identified sequences

Author(s)

Michal Stolarczyk & Alicja Pluciennik

Examples

```
find_consecutive_seq(c(1,2,3,4,5,6,7,20,21,140,141,300,301,302))
```

find_seq *Find sequence by id in alignment.*

Description

This function allows to search for a sequence with its id. Useful for browsing a larg multiple sequence alignment data or for automatization purposes.

Usage

```
find_seq(sequence_id, alignment)
```

Arguments

sequence_id	identifier of desired sequence from alignment
alignment	alignment file loaded with read.alignment

Value

sequence	A string, the desired aligned sequence form alignment

Author(s)

Alicja Pluciennik & Michal Stolarczyk

Examples

```
data("alignment")
#creating library uniprot - PDB
lib=list(  c("Q84HB8","4I19","4QA9"),
  c("P34913","4JNC"),
  c("P34914","1EK2","1CR6","1EK1","1CQZ"))
sequence_id=find_seqid("1CQZ",lib)
sequence=find_seq(sequence_id, alignment)
```

find_seqid *Find sequence identifier by other sequence identifier in given alignment within a specified library*

Description

This function allows to find sequence id from alignment file corresponding to the given sequence id. Function requires library of equivalent sequences id defined by user and it is useful to find sequences from other databases in alignment for examined sequence from other database (like PDB sequence for structure and UniProt sequences in alignment).

Usage

```
find_seqid(sequence_id, library)
```

Arguments

sequence_id	A string. An ID of e.g. PDB structure identifier
library	A list of vectors which contain a defined by user library e.g. of UniProt ids <-> PDB ids. See examples

Value

seqid	A string. The equivalent ID to the one provided as the input.

Author(s)

Alicja Pluciennik & Michal Stolarczyk

Examples

```
#creating library uniprot - PDB
lib=list(  c("Q84HB8","4I19","4QA9"),
  c("P34913","4JNC"),
  c("P34914","1EK2","1CR6","1EK1","1CQZ"))
PDB_name = "1CQZ"
find_seqid(PDB_name,lib)
```

---

get_pos_based_seq_weights

Get position based weights of sequences in alignment

Description

This function calculates position based weights of sequences based on Heinkoff & Heinkoff (1994) for given MSA. The score is calculated as sum of scores for each sequence position c. Score for position c is equal 1/r if there is r different residues at column c in MSA but 1/rs if r symbol is repeated in s sequences.

Usage

```
get_pos_based_seq_weights(alignment, gap=TRUE, normalized=TRUE)
```

Arguments

alignment	alignment loaded with read.alignment
gap	(optional) a logical parameter, if TRUE(default) the gaps in MSA are included
normalized	(optional) logical parameter, if TRUE (default) weights for all sequences are divided by number of columns in alignment (when gap = TRUE weights sum up to 1)

Details

The weights might be calculated only for amino acids symbols or for all symbols (including gaps). Also weights can be normalized by number of columns in MSA, then the sum of weights for all sequences is 1.

Value

`weights`	a vector of position based weights for each sequence in given alignment

Author(s)

Alicja Pluciennik & Michal Stolarczyk

References

Henikoff, S. & Henikoff, J. G. Position-based sequence weights. Journal of Molecular Biology 243, 574–578 (1994).

Examples

```
data("small_alignment")
pos_based_weights <- get_pos_based_seq_weights(small_alignment)
```

`get_prot_entropy` *Get MSA-based calculated entropy for chosen protein.*

Description

This function allows to obtain vector of entropies for one complete protein sequence from MSA (gaps introduced in alignment are omitted)

Usage

```
get_prot_entropy(protein_index, score_list)
```

Arguments

`protein_index`	Indices of given protein aminoacids in aligned sequence
`score_list`	A list of entropy scores calculated for MSA

Details

This function can be used on list of entropies or list with one element for one entropy score.

Value

`entropy`	A list where each element is a vector of entropy values provided in entropy_scores_list

Author(s)

Alicja Pluciennik & Michal Stolarczyk

Examples

```
data("structure")
data("alignment")
pdb_name = "1CQZ" #A string with path to PDB file
uniprot="P43914"
chain_identifier = "B"
structure_index=get_structures_idx(structure)
entropy_scores_list=list(Schneider_entropy = schneider_conservativity(alignment),
                         Escore_entropy = Escore_conservativity(alignment))
prot_entropy=get_prot_entropy(structure_index$proteinIndices, entropy_scores_list)

# In case of one entropy score
entropy_scores_list = list()
entropy_scores_list[[1]] = Schneider_entropy = schneider_conservativity(alignment)
prot_entropy=get_prot_entropy(structure_index$proteinIndices, entropy_scores_list)
```

---

get_remarks465_pdb *Get "REMARK 465" data from PDB file*

Description

This function extracts the data concerning missing amino acids in PDB protein structure from the PDB file

Usage

```
get_remarks465_pdb(pdb_path, chain_identifier)
```

Arguments

pdb_path	A string specifying the path tp the PDB file
chain_identifier	A character specifying the chain to be considered

Value

aa_numbers	A numeric vector of indices of missing amino acids
chain	A character specifying the chain which was considered in remark 465 data extraction

Author(s)

Michal Stolarczyk & Alicja Pluciennik

See Also

read.pdb

Examples

```
require(Rpdb)
chain_identifier = "A"
pdb_path = system.file("extdata", "4jnc.pdb", package = "BALCONY")
print(pdb_path)
#pdb_file_path = "path_to_file"
remark465_data = get_remarks465_pdb(pdb_path,chain_identifier)
```

---

get_seq_names *Get names of sequences from alignment*

Description

This function allows to get sequence names/identifires from alignment.

Usage

```
get_seq_names(alignment)
```

Arguments

alignment	The alignment object read with read.alignment function

Value

names	A vector of characters with names of each sequence from the alignment

Author(s)

Alicja Pluciennik & Michal Stolarczyk

Examples

```
data("alignment")
sequences_names=get_seq_names(alignment)
```

`get_seq_weights` *Get sequences weights*

Description

This function returns weights of the sequneces in the alignment object

Usage

```
get_seq_weights(alignment)
```

Arguments

`alignment`	alignment loaded with `read.alignment`

Details

The weights are calculated as shown in: Valdar and Thronton (2001)

According to the following formulas:

$$W_j = \frac{\sum_{k \neq j}^{N} Dist(s_j, s_k))}{N - 1}$$

where:
W_j is the weight of sequence s_j, and is defined as the average evolutionary distance between s_j and all other sequences in the alignment
N is the number of sequences in the alignment.

$$Dist(s_j, s_k)) = 1 = \frac{\sum_{i \epsilon Aligned_{jk}} Mut(s_j, s_k))}{n(Aligned_{jk}))}$$

where:
$Dist(s_j, s_k)$, the evolutionary distance between sequences s_j and s_k
$Aligned_{jk}$ is the set of all non-gap positions in s_j or s_k, $n(Aligned_{jk})$ is the number of such positions.

$$Mut(a, b) = \frac{m(a, b) - min(m)}{max(m) - min(m)}$$

where:
$Mut(a, b)$ measures the similarity between amino acids a and b as derived from a mutation data matrix m

Value

A vector with weights of length equal to the number of sequences in the alignment

Author(s)

Michal Stolarczyk & Alicja Pluciennik

References

Valdar, W. S. J. & Thornton, J. M. Protein–protein interfaces: Analysis of amino acid conservation in homodimers. Proteins: Structure, Function, and Bioinformatics 42, 108–124 (2001).

Examples

```
data("small_alignment")
alignment = small_alignment
weights = get_seq_weights(alignment)
```

---

get_structures_entropy

Get entropy of amino acids (for region of interest) in given protein

---

Description

This function allows to get values of entropy/conservation for amino acids dispersed in sequence of given protein. It works well with a list of dispersed amino acids in one protein.

Usage

```
get_structures_entropy(structure_index,score_list)
```

Arguments

structure_index	A is a list of indices in alignment of protein and structures. Output output of get_structures_idx function
score_list	A list of entropies for whole alignment

Details

This function allows to obtain entropy (calculated on MSA) for dispersed amino acids in protein e.g. surface, binding site, tunnels etc. The input is a list of few structure indices in given protein sequence. Function calculates position of those in aligned sequence and returns a vector/matrix or a list of matrices with entropy values.

Value

structure_entropies	A list of matrices. Rows are entropy scores, columns are

Author(s)

Alicja Pluciennik & Michal Stolarczyk

See Also

create_structure_seq, read_structure

Examples

```
data("structure")
data("alignment")

#creating library uniprot - PDB
uniprot="P34914"
tunnel=create_structure_seq(structure,uniprot,alignment)
indices=get_structures_idx(structure)
protein_index = indices$proteinIndices
structure_index = indices$structureIndices
entropy_scores_list=list(Schneider_entropy = schneider_conservativity(alignment),
                         Escore_entropy = Escore_conservativity(alignment))
structure_entropy=get_structures_entropy(structure_index, entropy_scores_list)
```

---

get_structures_idx *Get IDs of structure(s) elements from aligned sequences (MSA)*

Description

This function allows to obtain positions in aligned sequences for analyzed structure (e.g. functionally related amino acids dispersed in sequence) based on sequence corresponding to the crystal structure.

Usage

```
get_structures_idx(structure)
```

Arguments

structure	The output of create_structure_seq() function

Details

It facilitates the management and oparation on the entropy values calculated for given MSA.

Value

Output is a list of two elements:

proteinIndices	A sorted vector of amino acids of analyzed sequence in MSA
strucureIndices	A list of sorted vectors of amino acids indices in aligned sequence for each structure

Author(s)

Alicja Pluciennik & Michal Stolarczyk

Examples

```
data("structure")

#creating library uniprot - PDB
lib=list(c("Q84HB8","4I19","4QA9"),
        c("P34913","4JNC"),
        c("P34914","1EK2","1CR6","1EK1","1CQZ"))
pdb_name = "1CQZ" #A string with path to PDB file
uniprot=find_seqid(pdb_name,lib)
tunnel=create_structure_seq(structure,uniprot,alignment)
structure_index=get_structures_idx(tunnel)
```

---

gonnet *Gonnet substitution matrix*

Description

This dataset comprises the Gonnet substitution matrix which facilitates e.g. the calculation of Landgraf conservation score

Usage

```
data("gonnet")
```

Format

A data frame with 0 observations on the following 2 variables.

AA names Names of amino acids included in the matrix

matrix The substitution matrix itself

Source

http://imed.med.ucm.es/Tools/sias_help.html

Examples

```
data("gonnet")
```

is_upper *Check if the letter is uppercase.*

Description

This function facilitates the detection of uppercase strings/characters.

Usage

```
is_upper(string)
```

Arguments

string	A string or character

Details

All letters of a string must be uppercase for the string to be identified as an uppercase one

Value

A logical value indicating if the string/character is an uppercase one

Author(s)

Michal Stolarczyk & Alicja Pluciennik

Examples

```
string = "ABCD"
is_upper(string)
```

kabat_conservativity *Calculate Kabat conservation metric*

Description

This function facilitates the calculation of Kabat conservation metric.

Usage

```
kabat_conservativity(alignment, weights = NULL,pseudo_counts=F)
```

Arguments

alignment	Alignment data read with read.alignment function
weights	(optional) A vector of length equal number of sequences in the alignment object with weights to overcome the taxonomic bias in the conservation analysis.
pseudo_counts	(optional) A logical indicating if pseudo-counts should be added to the MSA. Pseudo-counts can be used only without weights. Using this option with pseudo-counts will be suppressed. Default: FALSE

Value

conservation_score
: A vector of length equal to the length of aligned sequences

Note

Please note that the Kabat matric formula can be found in the paper listed in "See Also" section below. Also, this function originally calculates the entropy values which can be used to estimate the conservativity score according to the following formula:

$$conservation = 1 - entropy$$

Author(s)

Alicja Plucennik & Michal Stolarczyk

See Also

http://onlinelibrary.wiley.com/doi/10.1002/prot.10146/abstract

Examples

```
data("small_alignment")
conservation_score = kabat_conservativity(alignment)
```

---

kolmogorov_smirnov_test

Perform Kolmogorov-Smirnov test for structural data

Description

This function facilitates the comparison of conservativity of structure of interest with the rest of the protein. For example comparison of tunnel conservativity with overall protein conservativity.

Usage

```
kolmogorov_smirnov_test(protein_entropy, structure_entropy, alternative,
                        pdb_name = "Reference", range = NULL, make_plot = NULL)
```

Arguments

protein_entropy	A list of calculated entropy scores (vectors of numeric values). The output of get_prot_entropy function
structure_entropy	A list where each element is a list of structure indices in the protein and matrix with corresponding entropy values (each row is a separate score metric)
alternative	A numeric value indicating the character of alternative hypothesis of the test to be performed: 1 - two sided test, 2 - less, 3 - greater, following the generic ks.test function.
pdb_name	(optional) A string with name of the reference protein, default: "Reference"
range	(optional) A numeric vector with region of reference protein to be excluded from the data set. Useful when protein consists of additional chains with outstandingly low/high entropy values which may distort result of the test, default: NULL
make_plot	(optional) A logical indicating if cumulative distribution functions should be displayed, default: TRUE

Value

A matrix of p-values for each entropy metric (rows) and structure (columns)

Author(s)

Michal Stolarczyk & Alicja Pluciennik

See Also

ecdf, ks.test

Examples

```
data("alignment")
data("structure")
entropy_data=list(Schneider.entropy=schneider_conservativity(alignment),
             Escore.entropy = Escore_conservativity(alignment),
             Kabat.entropy =  kabat_conservativity(alignment))
indices=get_structures_idx(structure)
protein_index = indices$proteinIndices
structure_index = indices$structureIndices
prot_cons=get_prot_entropy(protein_index,entropy_data)
stru_entropy=get_structures_entropy(structure_index,entropy_data)
profiles_for_structure=prepare_structure_profile(structure, stru_entropy)
EQUAL=kolmogorov_smirnov_test(protein_entropy = prot_cons,
                              structure_entropy = profiles_for_structure,
                              alternative = 1,range = c(1:233),make_plot = TRUE)
```

landgraf_conservativity *Calculate Landgraf conservation score*

Description

This function calculates Landgraf conservarion score

Usage

```
landgraf_conservativity(matrix_name = NULL, alignment, weights)
```

Arguments

matrix_name	A string with path to the file woith substitution matrix to be used to calculate the Landgraf conservation score. Optional parameter, if not provided the Gonnet substitution matrix is used (according to author's suggestion)
alignment	An alignment object read with read.alignment function
weights	A vector with weight for each sequence in the alignment to be used to calculate the Landgraf conservation score e.g. each sequence similarity to the consensus sequence from the alignment - output from cons2seqs_ident fuction

Value

score	A vector of length equal to the length of aligned sequences

Note

Please note that the Shannon matric formula can be found in the paper listed in "See Also" section below. Also, this function originally calculates the entropy values which can be used to estimate the conservativity score according to the following formula:

$$conservation = 1 - entropy$$

Author(s)

Michal Stolarczyk & Alicja Pluciennik

See Also

consensus, cons2seqs_ident, read.alignment

http://onlinelibrary.wiley.com/doi/10.1002/prot.10146/abstract

Examples

```
data("small_alignment")
alignment = small_alignment
threshold_consensus = 30
consensus_seq=consensus(alignment, threshold_consensus);
consensus_sequences_identity=cons2seqs_ident(alignment, consensus_seq)
score = landgraf_conservativity(alignment = alignment, weights = consensus_sequences_identity)
```

`noteworthy_seqs` *Find noteworthy sequences in the dataset (aligned sequences)*

Description

This function detects noteworthy sequences (most common, closest to the consensus and most different from the consesus) to facilitate convenient detection of outlying sequences that might be excluded from the further analysis.

Usage

```
noteworthy_seqs(percentage, alignment)
```

Arguments

`percentage`	The identity of each sequence in the alignment to the consensus sequence. Output of the `cons2seqs_ident` function
`alignment`	Alignment loaded with `read.alignment` function

Value

`best_consensus`	Sequence closest to the consensus
`worst_consensus`	Sequence most different to the consensus
`most_common`	Most common sequence in the alignment

Author(s)

Alicja Pluciennik & Michal Stolarczyk

Examples

```
data("alignment")
consensus_seq = consensus(alignment, 30)
consensus_to_sequences_identity=cons2seqs_ident(alignment,consensus_seq)
noteworthy_seqs(consensus_to_sequences_identity, alignment)
```

pairwise_alignment_MSA *Calculate pairwise alignment for whole MSA*

Description

For given alignment calculate pariwise alignments and returns alignment score.

Usage

```
pairwise_alignment_MSA(alignment)
```

Arguments

alignment	An alignment object read with read.alignment function

Value

score_mtx	Matrix of alignment scores

Author(s)

Michal Stolarczyk & Alicja Pluciennik

Examples

```
data("small_alignment")
pairwiseAlignemnt_scores=pairwise_alignment_MSA(small_alignment)
```

plot_entropy *Plot entropies for protein*

Description

This function plots entropies of protein. Plots might be superimposed or not.

Usage

```
plot_entropy(protein_conservation, colors, impose = NULL,
             prot_name = NULL, legend_pos = NULL)
```

Arguments

protein_conservation	A list or a vectors of protein conservation/entropies. The output of get_prot_entropy function
colors	(optional) A vector of colors for each plot, default: rainbow
impose	(optional) A boolean, if True/T plots are superimposed, if False/F plots are printed separately, default: T
prot_name	(optional) A string with structure name (to be used in the tile of the plot), default: none
legend_pos	(optional) A string witch legend position - one of following: "bottomright", "bottom", "bottomleft", "left", "topleft", "top", "topright", "right" and "center". Default: "bottomleft"

Details

This function produces plots for given values, on X axis are amino acids, on Y axis are values of entropy/conservation. Legend contains score names for description values.

Value

This function produces plots

Author(s)

Alicja Pluciennik & Michal Stolarczyk

Examples

```
data("alignment")
data("structure")
uniprot="P34914"
structure_index=get_structures_idx(structure)
entropy_scores_list=list(Schneider_entropy = schneider_conservativity(alignment),
                         Escore_entropy = Escore_conservativity(alignment))
prot_entropy=get_prot_entropy(structure_index$proteinIndices, entropy_scores_list)

plot_entropy(prot_entropy, colors = c("red","green","blue"),
             impose = TRUE, prot_name = "Murine Epoxide Hydrolase",
             legend_pos = "bottomright")
```

---

plot_structure_on_protein

Plot structure entropy on protein background

Description

This function enables to visually asses the stucture(s) entropy in comparison with protein's entropy

Usage

```
plot_structure_on_protein(protein_entropy, structure_profiles,
                          pdb_name, colors, structure_names, legend_pos)
```

Arguments

protein_entropy	A list of entropy values for protein of interest. Output of get_prot_entropy function
structure_profiles	Output of prepare_structure_profile function
pdb_name	(optional) A string with protein's name e.g. PDB ID, default: none
colors	(optional) A vector of strings with colors to be used to plot the stucture markers of length equal to number of structures in structure profiles, default: rainbow()
structure_names	(optional) A vector of strings to be displayed as names in the legend, default: "stru <no>"
legend_pos	(optional) A string witch legend position - one of following: "bottomright", "bottom", "bottomleft", "left", "topleft", "top", "topright", "right" and "center". Default: "bottomleft"

Details

For each entropy score from structure_profiles (these must correspond to prot_entropy) this function plots separate plots. Each plot presents entropy score for whole protein each structure is marked as one of 21 symbols available in generic plot function.

Value

This function produces plot

Author(s)

Alicja Pluciennik & Michal Stolarczyk

Examples

```
data("alignment")
data("structure")
indices=get_structures_idx(structure)
protein_index = indices$proteinIndices
structure_index = indices$structureIndices
entropy_scores_list=list(Schneider_entropy = schneider_conservativity(alignment),
                         Escore_entropy = Escore_conservativity(alignment))
structure_entropy=get_structures_entropy(structure_index, entropy_scores_list)
structure_profile = prepare_structure_profile(structure, structure_entropy)
prot_entropy=get_prot_entropy(protein_index, entropy_scores_list)

plot_structure_on_protein(prot_entropy, structure_profile)
```

prepare_structure_profile *This function combines the entropy data for structure building amino acids with their indices*

Description

This function combines the entropy data for structure building amino acids with its indices. It prepares the data for convenient visualization or processing.

Usage

```
prepare_structure_profile(structure, structure_entropy)
```

Arguments

structure	A structure data, the output of create_structure_seq function
structure_entropy	The entropy values for the structure building residues, the output of get_structures_entropy function

Value

List of structures
: Each element is a list of entropy values (matrix of entropy scores) and indices of residues building structure in protein of interest.

Author(s)

Alicja Pluciennik & Michal Stolarczyk

Examples

```
data("alignment")
data("structure")
uniprot="P34914"
indices=get_structures_idx(structure)
protein_index = indices$proteinIndices
structure_index = indices$structureIndices
entropy_scores_list=list(Schneider_entropy = schneider_conservativity(alignment),
                         Escore_entropy = Escore_conservativity(alignment))
structure_entropy=get_structures_entropy(structure_index, entropy_scores_list)
structure_profile = prepare_structure_profile(structure, structure_entropy)
```

`preprocess_hmm_output` *preprocess HMM output*

Description

Preprocessing od HMMER output file to calculate CRE.

Usage

```
preprocess_hmm_output(hmm_out)
```

Arguments

`hmm_out`	path to ouptut file

Value

Returns list of

`probabilities`	probabilities extracted from file
`alignment_positions`	index of each alignment position

Author(s)

Michal Stolarczyk & Alicja Pluciennik

See Also

CRE_conservativity()

Examples

```
#No example due to external software requirements
```

read_structure *Read structure data from a text file*

Description

By using this function you can read text file and create an structure list which can be used in further evolutionary analysis with BALCONY package. Text file should comprise 2 or 3 columns: first one should contain indices (positions) of amino acids in the protein, the second one should contain amino acid symbols on specified positions and the third one (optionally) the numeric property of given residue.

Usage

```
read_structure(file_names)
```

Arguments

file_names	A vector of strings with structure file(s) name(s)

Details

The files should be formatted as follows:

2 ASP 100
6 TYR 80
11 PHE 30
6 TYR 30

Value

strucure_list	A list with read structure data. Number of elements of this list equals to the number of files specified.

Author(s)

Alicja Pluciennik & Michal Stolarczyk

Examples

```
#Generating exemplary input files for the function

fileConn<-file("exemplary_input1.txt")
writeLines(c("2 TYR 100","3 LEU 100", "7 VAL 50", "10 PHE 30", "20 SER 20"), fileConn)
close(fileConn)
fileConn<-file("exemplary_input2.txt")
writeLines(c("5 ALA 100","6 ILE 100", "18 GLY 100", "40 PHE 100"), fileConn)
close(fileConn)
```

```
structure_list = read_structure(file_names = c("exemplary_input1.txt", "exemplary_input2.txt"))
```

`RealValET_conservativity` *Calculate real-value Evolutionary Trace (ET)*

Description

This function allows to calculate real-valued ET for MSA.

Usage

```
RealValET_conservativity(alignment)
```

Arguments

`alignment`	Alignment data read with read.alignment() function

Details

Here, the real-valued ET is calculated using an evolutionary tree calculated for given alignment. The tree is calculated using UPGMA method. Real-valued ET score can be used as complimentary analysis of evolutionary entropy measures.

Value

\ itemxA vector of real valued ET score corresponding to each MSA column

Author(s)

Alicja Plucennik & Michal Stolarczyk

References

Mihalek, Res, Lichtarge, 2004

Examples

```
data("small_alignment")
alignment = small_alignment
weights = get_seq_weights(alignment)
```

schneider_conservativity *Calculate Schneider conservation metric*

Description

This function facilitates the calculation of Schneider conservation metric.

Usage

```
schneider_conservativity(alignment, weights = NULL,pseudo_counts=F)
```

Arguments

alignment	Alignment data read with read.alignment function
weights	(optional) A vector of length equal number of sequences in the alignment object with weights to overcome the taxonomic bias in the conservation analysis.
pseudo_counts	(optional) A logical indicating if pseudo-counts should be added to the MSA. Pseudo-counts can be used only without weights. Using this option with pseudo-counts will be suppressed. Default: FALSE

Value

conservation_score
A vector of length equal to the length of aligned sequences

Note

Please note that the Schneider matric formula can be found in the paper listed in "See Also" section below. Also, this function originally calculates the entropy values which can be used to estimate the conservativity score according to the following formula:

$$conservation = 1 - entropy$$

Author(s)

Alicja Plucennik & Michal Stolarczyk

See Also

http://onlinelibrary.wiley.com/doi/10.1002/prot.10146/abstract

Examples

```
data("small_alignment")
conservation_score = schneider_conservativity(alignment)
```

shannon_conservativity *Calculate Shannon conservation metric*

Description

This function facilitates the calculation of Shannon conservation metric.

Usage

```
shannon_conservativity(alignment, weights = NULL,pseudo_counts=F)
```

Arguments

alignment	Alignment data read with read.alignment function
weights	(optional) A vector of length equal number of sequences in the alignment object with weights to overcome the taxonomic bias in the conservation analysis.
pseudo_counts	(optional) A logical indicating if pseudo-counts should be added to the MSA. Pseudo-counts can be used only withouth weights. Using these option with pseudo-counts will be suppressed. Default: FALSE

Value

conservation_score
A vector of length equal to the length of aligned sequences

Note

Please note that the Shannon matric formula can be found in the paper listed in "See Also" section below. Also, this function originally calculates the entropy values which can be used to estimate the conservativity score according to the following formula:

$$conservation = 1 - entropy$$

Author(s)

Alicja Plucennik & Michal Stolarczyk

See Also

http://onlinelibrary.wiley.com/doi/10.1002/prot.10146/abstract

Examples

```
data("small_alignment")
conservation_score = shannon_conservativity(alignment)
```

small_alignment *Sample small alignment of soluable epoxide hydrolase family*

Description

This alignment consists of 10 proteins which belong to the soluable epoxide hydrolase family. The amino acid sequences were aligned using MUSCLE algorithm with default settings.

Format

An alignment object read with `read.alignment` function from seqinr package.

alignment$nb A numeric: number of sequences

alignment$nam A vector of characters: names of the sequences

alignment$seq A vector of characters: amino acid sequences

Details

This is a smaller version of sample alignment which facilitates faster presentation of the functions capabilities.

Examples

```
data("small_alignment")
small_alignment
```

structure *A sample structure data*

Description

This sample structure data consists of the amino acids names forming tunnels and their numbers is analyzed protein. The data is a result of CAVER which is a software tool for analysis and visualization of tunnels and channels in protein structures.

Format

A structure object with three elements:

structure_matrix A matrix of characters "S" and "N" marking on sequence the structural element; "S" - amino acid forms the analyzed structure, "N" - amino acid which does not form the structure. Number of rows of the matrix corresponds to the number of structures analyzed

structure_numbers A vector containing the numbers of the amino acids in the sequence of interest (no gaps)

structure_probabilities A matrix of numeric values: probabilities of corresponding to the structural information from first element of the output, which helps to reduce the effect of non-consistent structural amino acids on the conservativity analysis of the structure of interest

Details

The tunnel analysis with CAVER was performed on human epoxide hydrolase structure (PDB ID: 4JNC) 50ns MD simulation.

See Also

CAVER: http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1002708
4JNC: http://www.sciencedirect.com/science/article/pii/S0960894X13004885

Examples

```
data("structure")
structure
```

---

substitution_mtx *Read a substitution matrix*

Description

This function facilitates reading of substitution matrices for further use

Usage

```
substitution_mtx(matrix_name)
```

Arguments

matrix_name	A string with path to the substitution matrix in a text file to be read

Value

names	A vector of characters with amino acid names included in the matrix
matrix	A numeric matrix with values

Author(s)

Michal Stolarczyk & Alicja Pluciennik

Examples

```
path = system.file("extdata", "GONNET.txt", package = "BALCONY")
sub_mat = substitution_mtx(path)
```

Index

∗Topic **Consensus**
noteworthy_seqs, 36
∗Topic **Landgraf**
landgraf_conservativity, 35
∗Topic **PDB file**
create_structure_seq, 15
∗Topic **REMARK 465**
get_remarks465_pdb, 26
∗Topic **Sequences**
noteworthy_seqs, 36
∗Topic **alignment**
align_params, 4
alignment2matrix, 3
consensus, 12
delete_isoforms, 18
Escore_conservativity, 20
find_seq, 23
get_pos_based_seq_weights, 24
get_seq_names, 27
kabat_conservativity, 32
schneider_conservativity, 44
shannon_conservativity, 45
∗Topic **amino acids variation**
barplotshow, 6
∗Topic **amino acids**
calculate_AA_variation, 7
convert_AA_symbol, 13
∗Topic **amino acid**
cons2seqs_sim, 11
∗Topic **barplot**
barplotshow, 6
∗Topic **consecutive**
find_consecutive_seq, 22
∗Topic **consensus_sequence**
consensus, 12
∗Topic **consensus**
cons2seqs_ident, 10
cons2seqs_sim, 11
consensus, 12
∗Topic **conservation**
CRE_conservativity, 17
Escore_conservativity, 20
kabat_conservativity, 32
landgraf_conservativity, 35
RealValET_conservativity, 43
schneider_conservativity, 44
shannon_conservativity, 45
∗Topic **conversion**
alignment2matrix, 3
∗Topic **convert**
convert_AA_symbol, 13
∗Topic **cumulative relative entropy**
CRE_conservativity, 17
∗Topic **datasets**
alignment, 2
gonnet, 31
small_alignment, 46
structure, 46
∗Topic **delete**
delete_isoforms, 18
∗Topic **detect**
is_upper, 32
∗Topic **dimension**
align_params, 4
∗Topic **entropy**
get_prot_entropy, 25
get_structures_entropy, 29
plot_entropy, 37
∗Topic **find**
find_consecutive_seq, 22
find_seqid, 23
∗Topic **get**
get_prot_entropy, 25
∗Topic **groups**
align_seq_mtx2grs, 5
∗Topic **hmmer**
preprocess_hmm_output, 41
∗Topic **identity**

cons2seqs_ident, 10
∗Topic **incomplete structure correction**
create_structure_seq, 15
∗Topic **indices**
get_structures_idx, 30
∗Topic **isoforms**
delete_isoforms, 18
∗Topic **lengths**
find_consecutive_seq, 22
∗Topic **matrix**
align_seq_mtx2grs, 5
alignment2matrix, 3
∗Topic **mertic**
kabat_conservativity, 32
schneider_conservativity, 44
shannon_conservativity, 45
∗Topic **metric**
Escore_conservativity, 20
∗Topic **missing amino acids**
get_remarks465_pdb, 26
∗Topic **output**
create_final_CSV, 14
∗Topic **pairwiseAlignment**
pairwise_alignment_MSA, 37
∗Topic **plot**
plot_entropy, 37
plot_structure_on_protein, 38
∗Topic **profile**
plot_structure_on_protein, 38
prepare_structure_profile, 40
∗Topic **properties**
cons2seqs_sim, 11
∗Topic **protein structure**
create_structure_seq, 15
∗Topic **pseudo counts**
calculate_pseudo_counts, 8
∗Topic **read**
read_structure, 42
substitution_mtx, 47
∗Topic **save**
create_final_CSV, 14
∗Topic **seqid**
find_seqid, 23
∗Topic **sequences names**
get_seq_names, 27
∗Topic **sequences**
find_consecutive_seq, 22
∗Topic **sequence**
find_seq, 23
∗Topic **structure file**
read_structure, 42
∗Topic **structure**
excl_low_prob_strcts, 21
get_structures_entropy, 29
get_structures_idx, 30
kolmogorov_smirnov_test, 33
prepare_structure_profile, 40
∗Topic **substitution matrix**
calculate_pseudo_counts, 8
substitution_mtx, 47
∗Topic **symbols**
convert_AA_symbol, 13
∗Topic **test**
kolmogorov_smirnov_test, 33
∗Topic **uppercase**
is_upper, 32
∗Topic **variation**
calculate_AA_variation, 7
∗Topic **weights**
get_pos_based_seq_weights, 24
RealValET_conservativity, 43

align_params, *3*, 4, *8*, *12*
align_seq_mtx2grs, 5, *12*
alignment, 2
alignment2matrix, 3, *5*, *12*

barplotshow, 6

calculate_AA_variation, *6*, 7, *14*
calculate_pseudo_counts, *8*, 8
compare_cons_metrics, 9
cons2seqs_ident, 10, *18*, *35, 36*
cons2seqs_sim, *11*, 11
consensus, *10–12*, 12, *13*, *18*, *35*
convert_AA_symbol, 13
CRE_conservativity, 17
create_final_CSV, 14
create_structure_seq, *14*, 15, *21*, *30*, *40*

D_matrix, 19
delete_isoforms, 18

ecdf, *34*
Escore_conservativity, *14*, 20
excl_low_prob_strcts, 21

find_consecutive_seq, *16*, 22
find_seq, 23
find_seqid, 23

get_pos_based_seq_weights, 24
get_prot_entropy, *9*, 25, *34*, *38*, *39*
get_remarks465_pdb, *16*, 26
get_seq_names, 27
get_seq_weights, 28
get_structures_entropy, 29, *40*
get_structures_idx, *29*, 30
gonnet, 31

is_upper, 32

kabat_conservativity, 32
kolmogorov_smirnov_test, 33
ks.test, *34*

landgraf_conservativity, *14*, 35

noteworthy_seqs, 36

pairwise_alignment_MSA, 37
pairwiseAlignment, *17*
plot, *39*
plot_entropy, 37
plot_structure_on_protein, 38
prepare_structure_profile, *9*, *39*, 40
preprocess_hmm_output, 41

read.alignment, *3–5*, *7*, *8*, *10–12*, *14–20*, *23*, *24*, *27*, *28*, *33*, *35–37*, *44–46*
read.pdb, *27*
read_structure, *15*, *16*, *30*, 42
RealValET_conservativity, 43

s2c, *13*
schneider_conservativity, *14*, 44
shannon_conservativity, 45
small_alignment, 46
structure, 46
substitution_mtx, 47