Package ‘VIProDesign’

July 21, 2025

Type Package

Title A Comprehensive Tool for Protein Design

Version 0.1.0

Description Provides tools for designing virus protein panels through sequence clustering and protein sequence analysis. The package includes functionality for filtering sequences, removing redundancy, identifying outliers, clustering sequences, and calculating entropy to evaluate clustering quality. A publication describing these methods is in preparation and will be added once available.

License Apache License 2.0

Encoding UTF-8

RoxygenNote 7.3.2

Imports Biostrings, DECIPHER, cluster, pathviewr, dbscan, ape

Suggests optparse

NeedsCompilation no

Author Tatsiana Bylund [aut, cre]

Maintainer Tatsiana Bylund <tatsiana.bylund@nih.gov>

Repository CRAN

Date/Publication 2025-05-02 09:30:09 UTC

Contents

entropy_RR 2
filter_sequences 2
phyl_tree_cluster_dbscan 3
phyl_tree_distance_k 4
run_VIProDesign 5

Index 7

entropy_RR *Calculate Entropy for Amino Acid Sequences*

Description

This function calculates the entropy of a set of amino acid sequences. It is used to evaluate the diversity of sequences in a given alignment.

Usage

```
entropy_RR(alignment)
```

Arguments

`alignment`	An 'AAStringSet' object containing the aligned amino acid sequences.

Details

The entropy is calculated based on the frequency of amino acids at each position in the alignment. Higher entropy indicates greater variability at that position.

Value

A numeric vector representing the entropy values for each position in the alignment.

Examples

```
# Example usage:
library(Biostrings)
sequences <- AAStringSet(c("ACDEFGHIK", "ACDEFGHIK", "ACDEFGHIK"))
entropy_values <- entropy_RR(sequences)
print(entropy_values)
```

filter_sequences *Filter Sequences*

Description

Filters sequences to remove non-standard amino acids.

Usage

```
filter_sequences(input_file)
```

Arguments

`input_file`	Path to the input FASTA file.

Value

A ‘AAStringSet‘ object containing the filtered sequences.

`phyl_tree_cluster_dbscan` *Perform DBSCAN Clustering on a Phylogenetic Tree*

Description

This function applies the DBSCAN clustering algorithm on a set of protein sequences to identify clusters and remove outliers based on a distance cutoff.

Usage

```
phyl_tree_cluster_dbscan(input_obj, cutoff, nmin)
```

Arguments

`input_obj`	A ‘AAStringSet‘ object containing protein sequences.
`cutoff`	A numeric value specifying the distance cutoff for clustering.
`nmin`	An integer specifying the minimum number of points required to form a cluster (DBSCAN parameter).

Details

The function uses the DBSCAN algorithm to cluster sequences based on their phylogenetic distances. Sequences identified as outliers are excluded from the final output.

Value

This function returns a ‘AAStringSet‘ object containing protein sequences with outliers removed.

Examples

```
# Example usage:
library(Biostrings)

# Create an AAStringSet object with the sequences
seqs <- AAStringSet(c(
  seq1 = "MKTIIALSYIFCLVFADYKDDDDK",
  seq2 = "MKTIIALSYIFCLVFADYKDLLKDDDD",
  seq3 = "MKTIIALSYIFCLVFADEELYKDDDD",
  seq4 = "MKTIEIALSYIFCLVFADYKDDDD",
  seq5 = "MKTIIKLAAASYIFCLVFADYKDDDD",
  seq6 = "MKTIIALSKIPFCLVFADYKDDDD",
  seq7 = "MKTIIALSYIFiQEERTCLVFADYKDDDD"
))
```

```
# Perform DBSCAN clustering and remove outliers
no_outliers <- phyl_tree_cluster_dbscan(seqs, cutoff = 0.5, nmin = 5)
```

`phyl_tree_distance_k` *Calculate Phylogenetic Tree Distances*

Description

This function calculates the pairwise distances between sequences in a phylogenetic tree and returns a numeric vector of 5th smallest distances for each leaf in the tree.

Usage

```
phyl_tree_distance_k(input_obj)
```

Arguments

`input_obj`	A ‘AAStringSet‘ object containing sequences.

Details

The function uses the ‘ape‘ package to construct a phylogenetic tree and calculate pairwise distances between sequences. The results are returned as a numeric vector.

Value

A numeric vector containing the 5th smallest distances (‘Distances_k‘) for each leaf in the phylogenetic tree.

Examples

```
# Input file
input_file <- system.file("extdata", "input.fasta", package = "VIProDesign")
seqs <- Biostrings::readAAStringSet(input_file)
distances_k <- phyl_tree_distance_k(seqs)
```

`run_VIProDesign` *Run VIProDesign Workflow*

Description

This function performs the VIProDesign workflow for clustering and analyzing protein sequences. It includes steps for filtering sequences, removing redundancy, identifying and removing outliers, and clustering sequences using PAM (Partitioning Around Medoids). This function requires the ‘cd-hit‘ executable to be installed and accessible in the system’s PATH if ‘use_cd_hit = TRUE‘. If ‘cd-hit‘ is not available, the workflow will skip redundancy removal and proceed with the filtered sequences.

Usage

```
run_VIProDesign(
  file,
  output_prefix,
  max_cluster_number = NULL,
  predefined_cluster_number = NULL,
  use_cd_hit = TRUE,
  cd_hit_path,
  cutoff = 0.99,
  remove_outliers = TRUE,
  verbose = FALSE
)
```

Arguments

`file`	A string specifying the path to the input FASTA file containing protein sequences.
`output_prefix`	A string specifying the prefix for output files generated by the workflow.
`max_cluster_number`	An integer specifying the maximum number of clusters to evaluate (optional).
`predefined_cluster_number`	An integer specifying a predefined number of clusters for PAM clustering (optional).
`use_cd_hit`	A logical value indicating whether to remove redundant sequences using ‘cd-hit‘ (default: TRUE).
`cd_hit_path`	A string specifying the path to the ‘cd-hit‘ executable (default: "cd-hit").
`cutoff`	A numeric value specifying the redundancy cutoff for ‘cd-hit‘ (default: 0.99).
`remove_outliers`	A logical value indicating whether to identify and remove outliers using DBSCAN clustering (default: TRUE).
`verbose`	A logical value indicating whether to print detailed messages during execution (default: FALSE).

Details

To install ‘cd-hit‘, you can use conda: “‘ conda install -c bioconda cd-hit “‘ Or download it from the official website: http://weizhong-lab.ucsd.edu/cd-hit/

The workflow includes the following steps: - Filtering sequences. - Removing redundancy using ‘cd-hit‘. - Identifying and removing outliers using DBSCAN clustering. - Performing PAM clustering to identify representative sequences. - Calculating entropy to evaluate clustering quality.

Value

A list containing the following elements:

- `filtered_file`: A file containing the filtered sequences (if redundancy removal was performed).
- `non_redundant_file`: A file containing the non-redundant sequences (if redundancy removal was performed).
- `no_outlier_obj`: A ‘AAStringSet‘ object containing the sequences with outliers removed (if outlier removal was performed).
- `clustering_info`: Clustering information generated by PAM clustering.
- `final_panel`: The final representative sequences selected by the workflow.

Examples

```
# Example usage:
temp_dir <- tempdir()
temp_prefix <- file.path(temp_dir, "output")
input_file <- system.file("extdata", "input.fasta", package = "VIProDesign")
run_VIProDesign(
  file = input_file,
  output_prefix = temp_prefix,
  max_cluster_number = 5,
  use_cd_hit = TRUE,
  cd_hit_path = "/data/kiryst/conda/envs/VIProDesign/bin/cd-hit",
  cutoff = 0.99,
  remove_outliers = TRUE
)
# Clean up
unlink(list.files(temp_dir, full.names = TRUE))
```

Index

entropy_RR, 2

filter_sequences, 2

phyl_tree_cluster_dbscan, 3
phyl_tree_distance_k, 4

run_VIProDesign, 5