Package 'AntibodyForests'

July 17, 2025

Title Delineating Inter- And Intra-Antibody Repertoire Evolution

Version 1.1.0

Maintainer Daphne van Ginneken <daphne.v.ginneken@gmail.com>

Description The generated wealth of immune repertoire sequencing data requires software to investigate and quantify inter- and intra-antibody repertoire evolution to uncover how B cells evolve during immune responses. Here, we present 'AntibodyForests', a software to investigate and quantify inter- and intra-antibody repertoire evolution.

License GPL-2

Depends R (>= 4.0.0)

Imports ape, Biostrings, dplyr, graphics, grDevices, gtools, igraph, magrittr, parallel, pwalign, rlang, scales, seqinr, stats, stringdist, stringr, tidyr, utils, viridis

Suggests alakazam, base64enc, bio3d, combinat, devtools, DT, fpc, ggplot2, ggrepel, ggsignif, htmltools, knitr, msa, phangorn, pheatmap, philentropy, Peptides, Rcompadre, rmarkdown, RPANDA, swipeR

VignetteBuilder knitr

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

NeedsCompilation no

Author Daphne van Ginneken [aut, cre],
Alexander Yermanos [aut],
Valentijn Tromp [aut],
Tudor-Stefan Cotet [ctb]

Repository CRAN

Date/Publication 2025-07-17 15:30:02 UTC

Contents

`Af_add_node_feature` *Function to add node features to an AntibodyForests-object*

Description

Function to add node features to an AntibodyForests-object

Usage

```
Af_add_node_feature(AntibodyForests_object, feature.df, feature.names)
```

Arguments

AntibodyForests_object	AntibodyForests-object, output from Af_build()
feature.df	Dataframe with features for each node. Must contain columns sample_id, clonotype_id, barcode and the features to be added.
feature.names	Character vector with the names of the features to be added.

Value

Returns an AntibodyForests-object with the features added to the nodes.

Examples

```
af <- Af_add_node_feature(AntibodyForests::small_af,
                          feature.df = AntibodyForests::small_vdj,
                          feature.names = c("VDJ_dgene", "VDJ_jgene"))
```

Af_build *Function to infer B cell evolutionary networks for all clonotypes in VDJ dataframe as obtained from the 'VDJ_build()' function.*

Description

This function takes a VDJ dataframe and uses the specified sequence columns to build a tree/network for each clonotype and stores them in an AntibodyForests object, together with the sequences and other specified features. These trees/networks provide insights into the evolutionary relationships between B cell sequences from each clonotype. The resulting object of class 'AntibodyForests' can be used for downstream analysis as input for...

Usage

```
Af_build(
  VDJ,
  sequence.columns,
  germline.columns,
  concatenate.sequences,
  node.features,
  string.dist.metric,
  dna.model,
  aa.model,
  codon.model,
  construction.method,
  IgPhyML.output.file,
```

```
  resolve.ties,
  remove.internal.nodes,
  include,
  parallel,
  num.cores
)
```

Arguments

VDJ	dataframe - VDJ object as obtained from the VDJ_build() function in Platypus, or object of class dataframe that contains the columns 'sample_id', 'clonotype_id', and the columns specified in 'sequence.columns', 'germline.columns', and 'node.features'.
sequence.columns	string or vector of strings - denotes the sequence column(s) in the VDJ dataframe that contain the sequences that will be used to infer B cell lineage trees. Nodes in the trees will represent unique combinations of the selected sequences. Defaults to 'c("VDJ_sequence_nt_trimmed", "VJ_sequence_nt_trimmed")'.
germline.columns	string or vector of strings - denotes the germline column(s) in the VDJ dataframe that contain the sequences that will be used as starting points of the trees. The columns should be in the same order as in 'sequence.columns'. Defaults to 'c("VDJ_germline_nt_trimmed", "VJ_germline_nt_trimmed")'.
concatenate.sequences	bool - if TRUE, sequences from multiple sequence columns are concatenated into one sequence for single distance matrix calculations / multiple sequence alignments, else, a distance matrix is calculated / multiple sequence alignment is performed for each sequence column separately. Defaults to FALSE.
node.features	string or vector of strings - denotes the column name(s) in the VDJ dataframe from which the node features should be extracted (which can, for example, be used for plotting of lineage trees later on). Defaults to 'isotype" (if present).
string.dist.metric	string - denotes the metric that will be calculated with the 'stringdist::stringdistmatrix()' function to measure (string) distance between sequences. Options: 'lv', 'dl', 'osa', 'hamming', 'lcs', 'qgram', 'cosine', 'jaccard', and 'jw'. Defaults to 'lv' (Levenshtein distance / edit distance). 'lv' : Levensthein distance (also known as edit distance) equals to the minimum number of single-element edits (insertions, deletions, or substitutions) required to transformer one string into another. 'dl' : Damerau-Levenshtein distance is similar to the Levenshtein distance, but also allows transpositions of adjacent elements as a single-edit operation. 'osa' : Optimal String Alignment distance is similar to the Damerau-Levensthein distance, but does not allow to apply multiple transformations on a same substring. 'hamming' : Hamming distance equals to the number of positions at which the corresponding elements differ between two strings (applicable only to strings of equal length). 'lcs' : Longest Common Subsequence distance is similar to the Levenshtein distance, but only allowing insertions and deletions as single-edit operations. 'qgram' : Q-gram distance equal to the number of distinct q-grams

that appear in either string but not both, whereby q-grams are all possible substrings of length q in both strings (q defaults to 1). 'cosine' : cosine distance equals to 1 - cosine similarity (the strings are converted into vectors containing the frequency of all single elements (A and B), whereby the cosine similarity (Sc) equals to the dot product of these vectors divided by the product of the magnitude of these vectors, which can be written in a formula as Sc(A, B) = A . B / (||A|| x ||B||)). 'jaccard' : Jaccard distance equals to 1 - Jaccard index (the strings are converted into sets of single elements (A and B), whereby the Jaccard index (J) equals to the size of the intersection of the two sets divided by the size of the union of the sets 'jw' : Jaro-Winkler distance equals to 1 - Jaro-Winkler similarity (Jaro-Winkler similary is calculated with the following formulas: Sw = Sj + P * L * (1-Sj) in which Sw is the Jaro-Winkler similary, Sj is the Jaro similarity, P is the scaling factor (defaults to 0), and L is the length of the matching prefix; and Sj = 1/3 * (m/|s1| + m/|s2| + (m-t)/m) in which Sj is the Jaro similarity, m is the number of matching elements, |s1| and |s2|are the lengths of the strings, and t is the number of transpositions).

`dna.model` string or vector of strings - specifies the DNA model(s) to be used during distance calculation or maximum likelihood tree inference. When using one of the distance-based construction methods ('phylo.network.default', 'phylo.network.mst', or 'phylo.tree.nj'), an evolutionary model can be used to compute a pairwise distance matrix from DNA sequences using the 'ape::dist.dna()' function. Available DNA models: 'raw', 'N', 'TS', 'TV', 'JC69', 'K80', 'F81', 'K81', 'F84', 'BH87', 'T92', 'TN93', 'GG95', 'logdet', 'paralin', 'indel', and 'indelblock'. When using the 'phylo.tree.ml' construction method, models are compared with each other with the 'phangorn::modelTest()' function, of which the output will be used as input for the 'phangorn::pml_bb()' function to infer the maximum likelihood tree. The best model according to the BIC (Bayesian information criterion) will be used to infer the tree. Defaults to "all" (when nucleotide sequences are found in the specified 'sequence.columns' and the 'germline.columns'). Available DNA models: 'JC', 'F81', 'K80', 'HKY', 'TrNe', 'TrN', 'TPM1', 'K81', 'TPM1u', 'TPM2', 'TPM2u', 'TPM3', 'TPM3u', 'TIM1e', 'TIM1', 'TIM2e', 'TIM2', 'TIM3e', 'TIM3', 'TVMe', 'TVM', 'SYM', and 'GTR'.

`aa.model` string or vector of strings - specifies the AA model(s) to be used during distance calculation or maximum likelihood tree inference. When using one of the distance-based construction methods ('phylo.network.default', 'phylo.network.mst', or 'phylo.tree.nj'), an evolutionary model can be used to compute a pairwise distance matrix from AA sequences using the 'phangorn::dist.ml()' function. Available AA models: '"WAG", "JTT", "LG", "Dayhoff", "VT", "Dayhoff_DCMut", "JTT-DCMut" When using the 'phylo.tree.ml' construction method, models are compared with each other with the 'phangorn::modelTest()' function, of which the output will be used as input for the 'phangorn::pml_bb' function to infer the maximum likelihood tree. The best model according to the BIC (Bayesian information criterion) will be used to infer the tree. Defaults to the following models: (when protein sequences are found in the specified 'sequence.columns' and the 'germline.columns'). Available AA models: "WAG", "JTT", "LG", "Dayhoff", "VT", "Dayhoff_DCMut", "JTT-DCMut"

`codon.model` string or vector of strings - specifies the codon substitution models to compare with each other with the 'phangorn::codonTest()' function (only possible

when the 'construction.method' paramter is set to 'phylo.tree.ml' and when colums with DNA sequences are selected). The best model according to the BIC (Bayesian information criterion) will be used to infer the tree, and this tree will replace the tree inferred with the best model of the model specified in the 'dna.models' parameter. Defaults to NA. Available codon models: 'M0'.

`construction.method`

string - denotes the approach and algorithm that will be used to convert the distance matrix or multiple sequence alignment into a lineage tree. There are two approaches two construct a lineage tree: a tree can be constructed from a network/graph (phylo.network) or from a phylogenetic tree (phylo.tree). There are three algorithm options that take a pairwise distance matrix as input: 'phylo.network.default', 'phylo.network.mst', and 'phylo.tree.nj'. There are two algorithm options that take a multiple sequence alignment as input: 'phylo.tree.ml', and 'phylo.tree.mp'. Defaults to 'phylo.network.default' (mst-like algorithm). 'phylo.network.default': mst-like tree evolutionary network algorithm in which the germline node is positioned at the top of the tree, and nodes with the minimum distance to any existing node in the tree are linked iteratively. 'phylo.network.mst' : minimum spanning tree (MST) algorithm from 'ape::mst()' constructs networks with the minimum sum of edge lengths/weights, which involves iteratively adding edges to the network in ascending order of edge weights, while ensuring that no cycles are formed, after which the network is reorganized into a germline-rooted lineage tree. 'phylo.tree.nj' : neighbor-joining (NJ) algorithm from 'ape::nj()' constructs phylogenetic trees by joining pairs of nodes with the minimum distance, creating a bifurcating tree consisting of internal nodes (representing unrecovered sequences) and terminal nodes (representing the recovered sequences). 'phylo.tree.mp' : maximum-parsimony (MP) algorithm from 'phangorn::pratchet()' constructs phylogenetic trees by minimizing the total number of edits required to explain the observed differences among sequences. 'phylo.tree.ml' : maximum-likelihood (ML) algorithm from 'phangorn::pml_bb()' constructs phylogenetic trees by estimating the tree topology and branch lengths that maximize the likelihood of observing the given sequence data under a specified evolutionary model. 'phylo.tree.IgPhyML' : no trees/network are inferred, but trees are directly imported from

`IgPhyML.output.file`

string - specifies the path to the IgPhyML output file, from which the trees will be imported (if 'construction.method' is set to 'phylo.tree.IgPhyML').

`resolve.ties` string or vector of strings - denotes the way ties are handled during the conversion of the distance matrix into lineage trees by the 'phylo.network.tree' algorithm (in the event where an unlinked node, that is to be linked to the tree next, shares identical distances with multiple previously linked nodes in the lineage tree). Options: 'min.expansion', 'max.expansion', 'min.germline.dist', 'max.germline.dist', 'min.germline.edges', 'max.germline.edges', and 'random'. If a vector is provided, ties will be resolved in a hierarchical manner. Defaults to 'c("max.expansion", "close.germline.dist", "close.germline.edges", "random")'. 'min.expansion' : the node(s) having the smallest size is/are selected. 'max.expansion' : the node(s) having the biggest size is/are selected. 'min.germline.dist' : the node(s) having the smallets string distance to the germline node is/are selected. 'max.germline.dist' : the node(s) having the biggest string distance to the germline

node is/are selected. 'min.germline.edges' : the node(s) having the lowest possible number of edges to the germline node is/are selected. 'max.germline.edges : the node(s) having the highest possible number of edges to the germline node is/are selected. 'min.descendants' : the node(s) having the smallest number of descendants is/are selected. 'max.descendants' : the node(s) having the biggest number of descendants is/are selected. 'random' : a random node is selected.

`remove.internal.nodes`
: string - denotes if and how internal nodes should be removed when the 'construction.method' is set to 'phylo.tree.nj', 'phylo.tree.mp', 'phylo.tree.ml' or 'phylo.tree.IgPhyML'. Options: 'zero.length.edges.only', 'connect.to.parent', 'minimum.length', and 'minimum.cost'. Defaults to 'minimum.cost', when 'construction.method' is set to 'phylo.tree.nj'. Defautls to 'connect.to.parent', when 'construction.method' is set to 'phylo.tree.mp', 'phylo.tree.ml', or 'phylo.tree.IgPhyML'. 'zero.length.edges.only' : only internal nodes with a distance of zero to a terminal node are removed by replacing it with the terminal node. 'connect.to.parent' : connects all terminal nodes to the first parental sequence-recovered node upper in the tree, resulting in a germline-directed tree. 'minimum.length' : iteratively replaces internal nodes with terminal nodes that are linked by an edge that has the minimum length. 'minimum.cost' : iteratively replaces internal nodes with terminal nodes which results in the minimum increase in the sum of all edges (this increase is referred to as the 'cost').

`include`
: string or vector of strings - specifies the objects to be included in the output object for each clonotype (if created). Options: 'nodes', 'dist.matrices', 'msa', 'phylo', 'igraph', 'igraph.with.inner.nodes', 'metrics', or 'all' to select all objects. Defaults to 'all'. 'nodes' : nested list wherein for each node, all information is stored (sequences, barcodes, selected column in 'node.features'). 'dist' : pairwise string distance matrices calculated using the specified 'string.dist.metric', one for each column selected in 'sequence.columns', or only one if 'concatenate_sequences' is set to TRUE. 'msa' : multiple sequence alignments, one for each column selected in 'sequence.columns', or only one if 'concatenate_sequences' is set to TRUE. 'phylo' : object of class 'phylo' that is created when 'construction.method' is set to 'phylo.tree.nj', 'phylo.tree.mp', or 'phylo.tree.ml', and when the clonotype contains at least three sequences. 'igraph' : object of class 'igraph' that represent the B cell lineage tree, which is used for plotting by the 'plot_lineage_tree()' function. 'igraph.with.inner.nodes' : object of class 'igraph' that represent the B cell lineage tree before the removal of internal nodes (if 'remove.internal.nodes' is set to 'connect.to.parent' or 'all'). 'edges' : dataframe with the three columns 'upper.node', 'lower.node', and 'edge.length', whereby each row in the dataframe represent an edge in the lineage tree. 'edges.with.inner.nodes' : dataframe with the three columns 'upper.node', 'lower.node', and 'edge.length', whereby each row in the dataframe represent an edge in the lineage tree. 'metrics' : list of tree metrics that can only be calculated during the construction of the lineage tree, which includes a 'tie.resolving' matrix, indicating which options were used to handle ties (when 'construction.method' is set to 'phylo.network.default'), and a 'model' string, indicating which model was used to infer the maximum likelihood tree (if 'construction.method' is set to 'phylo.tree.ml').

`parallel`
: bool - if TRUE, the per-clone network inference is executed in parallel (paral-

lelized across samples). Defaults to FALSE.

num.cores — integer - number of cores to be used when parallel = TRUE. Defaults to all available cores - 1 or the number of samples in the VDJ dataframe (depending which number is smaller).

Value

An object of class 'AntibodyForests', structured as a nested list where each outer list represents a sample, and each inner list represents a clonotype. Each clonotype list contains the output objects specified in the 'include' parameter. For example, `AntibodyForests[[1]][[2]]` contains the list of output objects for the first sample and third clonotype (which would be equivalent to something like AntibodyForests$S1$clonotype3).

Examples

```
af <- Af_build(VDJ = AntibodyForests::small_vdj,
               sequence.columns = c("VDJ_sequence_aa_trimmed","VJ_sequence_aa_trimmed"),
               germline.columns = c("VDJ_germline_aa_trimmed","VJ_germline_aa_trimmed"),
                node.features = c("VDJ_vgene", "isotype"),
                string.dist.metric = "lv",
                construction.method = "phylo.network.default")
```

Af_cluster_metrics *Function to make a grouped boxplot of metrics from clusters of clonotypes*

Description

Function to compare metrics between clusters of clontoypes

Usage

```
Af_cluster_metrics(
  input,
  clusters,
  metrics,
  min.nodes,
  colors,
  text.size,
  significance,
  parallel,
  num.cores
)
```

Arguments

input	• list - AntibodyForests-object as output from Af_build()
clusters	• named integer - The clusters as output from Af_compare_within_repertoires()
metrics	• string - The metrics to be calculated per tree 'nr.nodes' : The total number of nodes 'nr.cells' : The total number of cells in this clonotype 'mean.depth' : Mean of the number of edges connecting each node to the germline 'mean.edge.length' : Mean of the edge lengths between each node and the germline 'group.depth' : Mean of the number of edges connecting each node per group (node.features of the AntibodyForests-object) to the germline. (default FALSE) 'sackin.index' : Sum of the number of nodes between each terminal node and the germline, normalized by the total number of terminal nodes. 'spectral.density' : Metrics of the spectral density profiles (calculated with package RPANDA) – peakedness : Tree balance – asymmetry : Shallow or deep branching events – principal eigenvalue : Phylogenetic diversity – modalities : The number of different structures within the tree
min.nodes	The minimum number of nodes for a tree to be included in this analysis (this included the germline). This should be the same as for the Af_compare_within_repertoires() functions.
colors	• string - Optionally specific colors for the clusters
text.size	Font size in the plot (default 20).
significance	• boolean - If TRUE, the significance of a T test between the groups is plotted (default FALSE)
parallel	If TRUE, the metric calculations are parallelized across clonotypes. (default FALSE)
num.cores	Number of cores to be used when parallel = TRUE. (Defaults to all available cores - 1)

Value

- list - A list with boxplots per metric

Examples

```
plot <- Af_cluster_metrics(input = AntibodyForests::small_af,
                           clusters = AntibodyForests::compare_repertoire[["clustering"]],
                              metrics = "mean.depth",
                              min.nodes = 8)
plot$mean.depth
```

`Af_cluster_node_features` *Function to create a barplot of the cluster composition of selected features from each tree in an AntibodyForests-object*

Description

Function to create a barplot of the cluster composition of selected features from each tree in an AntibodyForests-object

Usage

```
Af_cluster_node_features(
  input,
  features,
  clusters,
  fill,
  colors,
  text.size,
  significance
)
```

Arguments

`input`	AntibodyForests-object(s), output from Af_build()
`features`	Character vector of features to include in the barplot. (these features need to be present in the nodes of the trees)
`clusters`	Named vector with the cluster assignments of the trees, output from Af_compare_within_repertoires().
`fill`	identify each unique feature per tree (unique, default), or assign the most observed feature to the tree (max)
`colors`	Color palette to use for the features.
`text.size`	Size of the text in the plot. Default is 12.
`significance`	Logical, whether to add Chi-squared Test p-value to the plot. Default is FALSE.

Value

A list with barplots for each provided feature.

Examples

```
plot <- Af_cluster_node_features(input = AntibodyForests::small_af,
                          clusters = AntibodyForests::compare_repertoire[["clustering"]],
                                  features = "isotype",
                                  fill = "max")
plot$isotype
```

`Af_compare_across_repertoires` *A function to compare dynamics of B cell evolution across different repertoires.*

Description

Compare tree topology metrics across different (groups) of AntibodyForests objects.

Usage

```
Af_compare_across_repertoires(
  AntibodyForests_list,
  metrics,
  plot,
  text.size,
  colors,
  significance,
  parallel,
  num.cores
)
```

Arguments

`AntibodyForests_list`	A list of AntibodyForests objects to compare.
`metrics`	Which metrics to use for comparison. Options are: . betweenness : The number of shortest paths that pass through each node (Default) . degree : The number of edges connected to each node (Default) 'nr.nodes' : The total number of nodes 'nr.cells' : The total number of cells in this clonotype 'mean.depth' : Mean of the number of edges connecting each node to the germline 'mean.edge.length' : Mean of the edge lengths between each node and the germline 'sackin.index' : Sum of the number of nodes between each terminal node and the germline, normalized by the number of terminal nodes 'spectral.density' : Metrics of the spectral density profiles (calculated with package RPANDA) • peakedness : Tree balance • asymmetry : Shallow or deep branching events • principal eigenvalue : Phylogenetic diversity • modalities : The number of different structures within the tree
`plot`	What kind of plot to make. boxplot (default) freqpoly
`text.size`	Font size in the plot (default 20).
`colors`	Optionally specific colors for the groups. If not provided, the default ggplot2 colors are used.
`significance`	If TRUE, the significance of a T test between the groups is plotted in the boxplot (default FALSE)

parallel	If TRUE, the metric calculations are parallelized (default FALSE)
num.cores	Number of cores to be used when parallel = TRUE. (Defaults to all available cores - 1)

Value

Plots to compare the repertoires on the supplied metrics.

Examples

```
boxplots <- Af_compare_across_repertoires(list("S1" = AntibodyForests::small_af[1],
            "S2" = AntibodyForests::small_af[2]),
            metrics = c("betweenness", "degree"),
            plot = "boxplot")
boxplots$betweenness
```

Af_compare_methods *Function to compare trees created with different algorithms from the same clonotype.*

Description

Function to compare different trees from the same clonotype to compare various graph construction and phylogenetic reconstruction methods.

Usage

```
Af_compare_methods(
  input,
  min.nodes,
  include.average,
  distance.method,
  depth,
  clustering.method,
  visualization.methods,
  parallel,
  num.cores
)
```

Arguments

input	A list of AntibodyForests-objects as output from the function Af_build(). These objects should contain the same samples/clonotypes. For easy interpretation of the results, please name the objects in the list according to their tree-construction method.
min.nodes	The minimum number of nodes in a tree to include in the comparison, this includes the germline. Default is 2 (this includes all trees).

include.average
: If TRUE, the average distance matrix and visualizations between the trees is included in the output (default FALSE)

distance.method
: The method to calculate the distance between trees (default euclidean) 'euclidean' : Euclidean distance between the depth of each node in the tree 'GBLD' : Generalized Branch Length Distance, derived from Mahsa Farnia & Nadia Tahiri, Algorithms Mol Biol 19, 22 (2024). https://doi.org/10.1186/s13015-024-00267-1

depth
: If distance.methods is 'euclidean', method to calculate the germline-to-node depth (default edge.count) 'edge.count' : The number of edges between each node and the germline 'edge.length' : The sum of edge lengths between each node and the germline

clustering.method
: Method to cluster trees (default NULL) NULL : No clustering 'mediods' : Clustering based on the k-mediods method. The number of clusters is estimated based on the optimum average silhouette.

visualization.methods
: The methods to analyze similarity (default NULL) NULL : No visualization 'PCA' : Scatterplot of the first two principal components. 'MDS' : Scatterplot of the first two dimensions using multidimensional scaling. "heatmap' : Heatmap of the distance

parallel
: If TRUE, the depth calculations are parallelized across clonotypes (default FALSE)

num.cores
: Number of cores to be used when parallel = TRUE. (Defaults to all available cores - 1)

Value

A list with all clonotypes that pass the min.nodes threshold including the distance matrix, possible clustering and visualization

Examples

```
plot <- Af_compare_methods(input = list("Default" = AntibodyForests::af_default,
                                        "MST" = AntibodyForests::af_mst,
                                        "NJ" = AntibodyForests::af_nj),
                           depth = "edge.count",
                           visualization.methods = "heatmap",
                           include.average = TRUE)
plot$average
```

Af_compare_PLM *Function to compare the distributions of the Protein Language Model probabilities or ranks of the mutations along the edges of B cell lineage trees across repertoires using the Jensen-Shannon divergence.*

Description

Function to compare the distributions of the Protein Language Model probabilities or ranks of the mutations along the edges of B cell lineage trees across repertoires using the Jensen-Shannon divergence.

Usage

```
Af_compare_PLM(PLM_dataframe, values, font.size, output.file)
```

Arguments

PLM_dataframe	Dataframe resulting from Af_PLM_dataframe(). This contains the Protein Language Model probabilities and ranks of the mutations along the edges of B cell lineage trees.
values	What values to compare "substitution_rank" will compare the rank of the mutation along the edge of the tree (Highest probability is rank 1). "substitution_probability" will copmare the probability of the mutation along the edge of the tree. "original_rank" will compare the rank of the original amino acid at the site of mutation along the edge of the tree (Highest probability is rank 1). "original_probability" will compare the probability of the original amino acid at the site of mutation along the edge of the tree. "unmutated_rank" will compare the rank of the unmutated amino acids along the edge of the tree (Highest probability is rank 1). "unmutated_probability" will compare the probabilities of the unmutated amino acids along the edge of the tree.
font.size	Font size for the plot. Default is 16.
output.file	string - specifies the path to the output file (PNG of PDF). Defaults to NULL.

Value

A pheatmap of the Jensen-Shannon distance between repertoires

Examples

```
Af_compare_PLM(PLM_dataframe = AntibodyForests::PLM_dataframe,
            values = "original_probability")
```

Af_compare_within_repertoires

Function to compare tree topology of B cell lineages

Description

Function to compare trees of clonotypes.

Usage

```
Af_compare_within_repertoires(
  input,
  min.nodes,
  distance.method,
  distance.metrics,
  clustering.method,
  visualization.methods,
  plot.label,
  text.size,
  point.size = 2,
  parallel,
  num.cores
)
```

Arguments

`input`
- list - An AntibodyForests-object, output from Af_build()

`min.nodes`
- integer - The minimum number of nodes in a tree to include in the comparison

`distance.method`
- string - The method to calculate distance (default ...) 'none' : No distance metric, analyze similarity directly from distance.metrics 'euclidean' : 'jensen-shannon' : Jensen-Shannon distance between spectral density profiles of trees.

`distance.metrics`
- string - If distance.method is "none" or "euclidean", these metrics will be used to calculate clusters and PCA/MDS dimensions and are used for plotting. (Default is mean.depth and nr.nodes) 'nr.nodes' : The total number of nodes 'nr.cells' : The total number of cells in this clonotype 'mean.depth' : Mean of the number of edges connecting each node to the germline 'mean.edge.length' : Mean of the edge lengths between each node and the germline 'group.depth' : Mean of the number of edges connecting each node per group (node.features of the AntibodyForests-object) to the germline. (default FALSE) 'sackin.index' : Sum of the number of nodes between each terminal node and the germline, normalized for the number of terminal nodes. 'spectral.density' : Metrics of the spectral density profiles (calculated with package RPANDA)
  - peakedness : Tree balance
  - asymmetry : Shallow or deep branching events
  - principal eigenvalue : Phylogenetic diversity
  - modalities : The number of different structures within the tree

`clustering.method`
- string - Method to cluster trees (default none) 'none' : No clustering 'mediods' : Clustering based on the k-mediods method. The number of clusters is estimated based on the optimum average silhouette.

visualization.methods
- string - The methods to analyze similarity (default PCA) 'PCA' : Scatterplot of the first two principal components. This is usefull when distance.method is "none". 'MDS' : Scatterplot of the first two dimensions using multidimensional scaling. Usefull for all distance methods 'heatmap' : A (clustered) heatmap of the distance between clonotypes. If distance.method is "none", euclidean distance will be calculated.

plot.label
- boolean - Label clonotypes in the PCA/MDS plot (default FALSE)

text.size
- integer - Size of the text in the plots (default 12)

point.size
- integer - Size of the points in the plots (default 2)

parallel — If TRUE, the metric calculations are parallelized (default FALSE)

num.cores — Number of cores to be used when parallel = TRUE (Defaults to all available cores - 1)

Value

- list - Returns a distance matrix, clustering, and various plots based on visualization.methods

Examples

```
compare_repertoire <- Af_compare_within_repertoires(input = AntibodyForests::small_af,
                                    min.nodes = 8,
                                    distance.method = "euclidean",
                                    distance.metrics = c("mean.depth", "sackin.index"),
                                    clustering.method = "mediods",
                                    visualization.methods = "PCA")
#Plot the PCA clusters
compare_repertoire$plots$PCA_clusters
```

---

af_default

Small AntibodyForests object with default algorithm for function testing purposes

---

Description

Small AntibodyForests object with default algorithm for function testing purposes

Usage

```
af_default
```

Format

An object of class `list` of length 1.

`Af_distance_boxplot` *Function to make a grouped boxplot of distance between nodes from specific groups and the germline of lineage trees constructed with AntibodyForests.*

Description

Function to compare trees.

Usage

```
Af_distance_boxplot(
  AntibodyForests_object,
  distance,
  min.nodes,
  groups,
  node.feature,
  unconnected,
  colors,
  text.size,
  x.label,
  group.order,
  significance,
  parallel,
  output.file
)
```

Arguments

`AntibodyForests_object`	AntibodyForests-object, output from Af_build()
`distance`	• string - How to calculate the distance to the germline. 'node.depth' : Average of the sum of edges on the shortest parth between germline and nodes from this group. 'edge.length' : Average of the sum of edge length of the shortest path between germline and nodes from this group. (Default)
`min.nodes`	The minimum number of nodes for a tree to be included in this analysis (this included the germline)
`groups`	Which groups to compare. These groups need to be in the node features of the AntibodyForests-object. Set to NA if all features should displayed. (default is NA) If you want to compare IgM and IgG for example, groups should be c("IgM, "IgG") (not "Isotypes")
`node.feature`	Node feature in the AntibodyForests-object to compare.
`unconnected`	If TRUE, trees that don't have all groups will be plotted, but not included in significance analysis. (default FALSE)
`colors`	Optionally specific colors for the group (Will be matched to the groups/names on alphabetical order).

text.size	Font size in the plot (default 20).
x.label	Label for the x-axis (default is the node feature).
group.order	Order of the groups on the x-axis. (default is alphabetical/numerical)
significance	If TRUE, the significance of the difference (paired t-test) between the groups is plotted. (default FALSE)
parallel	If TRUE, the metric calculations are parallelized across clonotypes. (default FALSE)
output.file	string - specifies the path to the output file (PNG of PDF). Defaults to NULL.

Value

A ggplot2 object with the boxplot.

Examples

```
Af_distance_boxplot(AntibodyForests::small_af,
                    distance = "edge.length",
                    min.nodes = 5,
                    groups = c("IGHA", "IgG1"),
                    node.feature = "isotype",
                    unconnected = TRUE)
```

Af_distance_scatterplot

Function to scatterplot the distance to the germline to a numerical node feature of the AntibodyForests-object

---

Description

Function to scatterplot the distance to the germline to a numerical node feature of the AntibodyForests-object

Usage

```
Af_distance_scatterplot(
  AntibodyForests_object,
  node.features,
  distance,
  min.nodes,
  color.by,
  color.by.numeric,
  correlation,
  geom_smooth.method,
  color.palette,
  font.size,
```

```
  ylabel,
  point.size,
  output.file
)
```

Arguments

AntibodyForests_object	AntibodyForests-object, output from Af_build()
node.features	Node features in the AntibodyForests-object to compare (needs to be numerical)
distance	string - How to calculate the distance to the germline. 'node.depth' : The sum of edges on the shortest parth between germline and each node 'edge.length' : The sum of edge length of the shortest path between germline and each node (Default)
min.nodes	The minimum number of nodes for a tree to be included in this analysis (this included the germline). Default is 2.
color.by	Color the scatterplot by a node.feature in the AntibodyForests-object, by the sample, or no color ("none). Default is "none".
color.by.numeric	Logical. If TRUE, the color.by feature is treated as a numerical feature. Default is FALSE.
correlation	"pearson", "spearman", "kendall", or "none"
geom_smooth.method	"none", lm" or "loess". Default is "none".
color.palette	The color palette to use for the scatterplot. Default for numerical color.by is "viridis".
font.size	The font size of the plot. Default is 12.
ylabel	The labels of the y-axis, in the same order as the node.features. Default is the node.features
point.size	The size of the points in the scatterplot. Default is 1.
output.file	string - specifies the path to the output file (PNG of PDF). Defaults to NULL.

Value

A ggplot2 object with the scatterplot

Examples

```
Af_distance_scatterplot(AntibodyForests_object = AntibodyForests::small_af,
                        node.features = "size",
                        distance = "edge.length",
                        min.nodes = 5,
                        color.by = "sample",
                        color.by.numeric = FALSE,
                        geom_smooth.method = "lm",
                        correlation = "pearson")
```

`Af_edge_RMSD` *Function to calculate the RMSD between sequences over each edge in the AntibodyForest object*

Description

This function calculates the RMSD between sequences over each edge in the AntibodyForest object.

Usage

```
Af_edge_RMSD(
  AntibodyForests_object,
  VDJ,
  pdb.dir,
  file.df,
  sequence.region,
  sub.sequence.column,
  chain,
  font.size,
  point.size,
  color,
  output.file
)
```

Arguments

`AntibodyForests_object`
: AntibodyForests-object, output from Af_build()

`VDJ`
: The dataframe with V(D)J information such as the output of Platypus::VDJ_build() that was used to create the AntibodyForests-object. Must contain columns sample_id, clonotype_id, barcode.

`pdb.dir`
: a directory containing PDB files.

`file.df`
: a dataframe of pdb filenames (column file_name) to be used and sequence IDs (column sequence) corresponding to the the barcodes in the AntibodyForests-object

`sequence.region`
: a character vector of the sequence region to be used to calculate properties. Default is "full.sequence".

  - full.sequence: the full sequence(s) in the PDB file
  - sub.sequence: part of the full sequence, for example the CDR3 region in the PDB file. This sub sequence must be a column in the VDJ dataframe.
  - binding.residues: the binding residues in the PDB file

`sub.sequence.column`
: a character vector of the column name in the VDJ dataframe containing the sub sequence to be used to calculate properties. Default is NULL.

chain	a character vector of the chain to be used to calculate properties. Default is both heavy and light chain Assuming chain "A" is heavy chain, chain "B" is light chain, and possible chain "C" is the antigen. • HC+LC: both heavy and light chain • HC: heavy chain, assuming chain A is the heavy chain. • LC: light chain, assuming chain B is the light chain. • AG: antigen, assuming chain C is the antigen. • whole.complex: the whole complex of antibody-antigen (all available chains in the pdb file).
font.size	The font size of the plot. Default is 12.
point.size	The size of the points in the scatterplot. Default is 1.
color	The color of the dots in the scatterplot. Default is "black".
output.file	string - specifies the path to the output file (PNG of PDF). Defaults to NULL.

Value

A list with the edge dataframe and a ggplot2 object

Examples

```
## Not run:
rmsd_df <- Af_edge_RMSD(AntibodyForests::small_af,
                        VDJ = AntibodyForests::small_vdj,
                        pdb.dir = "~/path/PDBS_superimposed/",
                        file.df = files,
                        sequence.region = "full.sequence",
                        chain = "HC+LC")
## End(Not run)
```

Af_get_sequences *Function to get the sequences from the nodes in an AntibodyForest object*

Description

Function to get the sequences from the nodes in an AntibodyForest object

Usage

```
Af_get_sequences(AntibodyForests_object, sequence.name, min.nodes, min.edges)
```

Arguments

AntibodyForests_object	AntibodyForests-object, output from Af_build()
sequence.name	character, name of the sequence column in the AntibodyForests object (example VDJ_sequence_aa_trimmed)
min.nodes	integer, minimum number of nodes in the tree (not including germline)
min.edges	integer, minimum number of edges in the tree (not including edges to the germline)

Value

A dataframe with the sequences and sequence identifiers

Examples

```
sequence_df <- Af_get_sequences(AntibodyForests::small_af,
               sequence.name = "VDJ_sequence_aa_trimmed")
```

Af_metrics *Function to calculate metrics for each tree in an AntibodyForests-object*

Description

Function to calculate metrics for each tree in an AntibodyForests-object

Usage

```
Af_metrics(
  input,
  min.nodes,
  node.feature,
  group.node.feature,
  multiple.objects,
  metrics,
  parallel,
  num.cores,
  output.format
)
```

Arguments

input	AntibodyForests-object(s), output from Af_build()
min.nodes	The minimum number of nodes in a tree to calculate metrics (including the germline).
node.feature	The node feature to be used for the group.edge.length, group.node.size or group.nodes.depth metric.

`group.node.feature`
: The groups in the node feature to be plotted. Set to NA if all features should displayed. (default NA)

`multiple.objects`
: If TRUE: input should contain multiple AntibodyForests-objects (default FALSE)

`metrics`
: The metrics to be calculated (default mean.depth and nr.nodes) 'nr.nodes' : The total number of nodes 'nr.cells' : The total number of cells in this clonotype 'mean.depth' : Mean of the number of edges connecting each node to the germline 'mean.edge.length' : Mean of the edge lengths between each node and the germline 'group.node.depth' : Mean of the number of edges connecting each node per group (node.features of the AntibodyForests-object) to the germline. 'group.edge.length' : Mean of the sum of edge length of the shortest path between germline and nodes per group (node.features of the AntibodyForests-object) 'group.node.size' : Mean of the node size per group normalized by the clone size (node.features of the AntibodyForests-object) 'sackin.index' : Sum of the number of nodes between each terminal node and the germline, normalized by the total number of terminal nodes. 'spectral.density' : Metrics of the spectral density profiles (calculated with package RPANDA)

  - peakedness : Tree balance
  - asymmetry : Shallow or deep branching events
  - principal eigenvalue : Phylogenetic diversity
  - modalities : The number of different structures within the tree

`parallel`
: If TRUE, the metric calculations are parallelized (default FALSE)

`num.cores`
: Number of cores to be used when parallel = TRUE. (Defaults to all available cores - 1)

`output.format`
: The format of the output. If set to "dataframe", a dataframe is returned. If set to "AntibodyForests", the metrics are added to the AntibodyForests-object. (default "dataframe")

Value

Returns either a dataframe where the rows are trees and the columns are metrics or an AntibodyForests-object with the metrics added to trees

Examples

```
metric_df <- Af_metrics(input = AntibodyForests::small_af,
                        metrics = c("mean.depth", "sackin.index"),
                        min.nodes = 8)
head(metric_df)
```

af_mst *Small AntibodyForests object with MST algorithm for function testing purposes*

Description

Small AntibodyForests object with MST algorithm for function testing purposes

Usage

```
af_mst
```

Format

An object of class `list` of length 1.

af_nj *Small AntibodyForests object with NJ algorithm for function testing purposes*

Description

Small AntibodyForests object with NJ algorithm for function testing purposes

Usage

```
af_nj
```

Format

An object of class `list` of length 1.

`Af_node_size_boxplot` *Function to make a grouped boxplot of the normalized average node sizes (number of cells with the exact same sequence) from specific groups of lineage trees constructed with AntibodyForests.*

Description

Function to compare trees.

Usage

```
Af_node_size_boxplot(
  AntibodyForests_object,
  min.nodes,
  groups,
  node.feature,
  colors,
  text.size,
  x.label,
  group.order,
  significance,
  parallel,
  output.file
)
```

Arguments

`AntibodyForests_object`	AntibodyForests-object, output from Af_build()
`min.nodes`	The minimum number of nodes for a tree to be included in this analysis (this included the germline)
`groups`	Which groups to compare. These groups need to be in the node features of the AntibodyForests-object. Set to NA if all features should displayed. (default is NA) If you want to compare IgM and IgG for example, groups should be c("IgM, "IgG") (not "Isotypes")
`node.feature`	Node feature in the AntibodyForests-object to compare.
`colors`	Optionally specific colors for the group (Will be matched to the groups/names on alphabetical order).
`text.size`	Font size in the plot (default 20).
`x.label`	Label for the x-axis (default is the node feature).
`group.order`	Order of the groups on the x-axis. (default is alphabetical/numerical)
`significance`	If TRUE, the significance of the difference (paired t-test) between the groups is plotted. (default FALSE)
`parallel`	If TRUE, the metric calculations are parallelized across clonotypes. (default FALSE)
`output.file`	string - specifies the path to the output file (PNG of PDF). Defaults to NULL.

Value

A ggplot2 object with the boxplot.

Examples

```
Af_node_size_boxplot(AntibodyForests::small_af,
                     min.nodes = 5,
                     groups = c("IGHA", "IgG1"),
                     node.feature = "isotype")
```

`Af_PLM_dataframe` *Function to create a dataframe of the Protein Language Model probabilities and ranks of the mutations along the edges of B cell lineage trees.*

Description

Function to create a dataframe of the Protein Language Model probabilities and ranks of the mutations along the edges of B cell lineage trees.

Usage

```
Af_PLM_dataframe(AntibodyForests_object, sequence.name, path_to_probabilities)
```

Arguments

`AntibodyForests_object`	AntibodyForests-object, output from Af_build()
`sequence.name`	character, name of the sequence column in the AntibodyForests object (example VDJ_sequence_aa_trimmed)
`path_to_probabilities`	character, path to the folder containing probability matrices for all sequences. Probability matrices should be in CSV format and the filename should include sampleID_clonotypeID_nodeNR, matching the AntibodyForests-object.

Value

A dataframe with the sample, clonotype, node numbers, number of substitutions, mean substitution rank and mean substitution probability

Examples

```
## Not run:
PLM_dataframe <- Af_PLM_dataframe(AntibodyForests_object = AntibodyForests::small_af,
                 sequence.name = "VDJ_sequence_aa_trimmed",
                 path_to_probabilities = "/directory/ProbabilityMatrix")

## End(Not run)
```

`Af_plot_PLM` *Function to create a distribution plot of the Protein Language Model probabilities and ranks of the mutations along the edges of B cell lineage trees.*

Description

Function to create a distribution plot of the Protein Language Model probabilities and ranks of the mutations along the edges of B cell lineage trees.

Usage

```
Af_plot_PLM(PLM_dataframe, values, group_by, colors, font.size, output.file)
```

Arguments

`PLM_dataframe`	Dataframe resulting from Af_PLM_dataframe(). This contains the Protein Language Model probabilities and ranks of the mutations along the edges of B cell lineage trees.
`values`	What values to plot. Can be "rank" (default) or "probability". "substitution_rank" will plot the rank of the mutation along the edge of the tree (Highest probability is rank 1). "substitution_probability" will plot the probability of the mutation along the edge of the tree. "original_rank" will plot the rank of the original amino acid at the site of mutation along the edge of the tree (Highest probability is rank 1). "original_probability" will plot the probability of the original amino acid at the site of mutation along the edge of the tree.
`group_by`	Plot a seperate line per group or everything together (default). "sample_id" per sample "n_subs" number of substitutions "none" no grouping
`colors`	Color to use for the lines. When group_by = "sample_id": This should be a vector of the same length as the number of samples.
`font.size`	Font size for the plot. Default is 16.
`output.file`	string - specifies the path to the output file (PNG of PDF). Defaults to NULL.

Value

A ggplot2 object of the PLM plot

Examples

```
Af_plot_PLM(PLM_dataframe = AntibodyForests::PLM_dataframe,
            values = "original_probability",
            group_by = "sample_id")
```

`Af_plot_PLM_mut_vs_cons` *Function to create a boxplot of the Protein Language Model probabilities*

Description

Function to create a boxplot of the Protein Language Model probabilities and ranks of the mutating vs. conserved residues along the edges of B cell lineage trees.

Usage

```
Af_plot_PLM_mut_vs_cons(
  PLM_dataframe,
  values,
  dots,
  group_by,
  colors,
  font.size,
  output.file
)
```

Arguments

`PLM_dataframe`	Dataframe resulting from Af_PLM_dataframe(). This contains the Protein Language Model probabilities and ranks of the mutations along the edges of B cell lineage trees.
`values`	What values to plot. Can be "rank" (default) or "probability". "rank" will plot the rank of the amino acid (Highest probability is rank 1). "probability" will plot the probability of the amino acid.
`dots`	Whether to plot the individual points. Can be "none" (default), "all_edges", "sample_average"
`group_by`	Color the dots on a group. Can be "none" (default), "sample_id", or "n_subs".
`colors`	Color to use for the dots. When group_by = "sample_id": This should be a vector of the same length as the number of samples.
`font.size`	Font size for the plot. Default is 16.
`output.file`	string - specifies the path to the output file (PNG of PDF). Defaults to NULL.

Value

A ggplot2 object of the PLM boxplot

Examples

```
Af_plot_PLM_mut_vs_cons(PLM_dataframe = AntibodyForests::PLM_dataframe,
            values = "probability")
```

`Af_plot_tree` *Plots lineage tree of clonotype from AntibodyForests object*

Description

This function retrieves the igraph object from the provided AntibodyForests object for the specified clone within the specified sample and plots the lineage tree using the specified plotting parameters.

Usage

```
Af_plot_tree(
  AntibodyForests_object,
  sample,
  clonotype,
  show.inner.nodes,
  x.scaling,
  y.scaling,
  color.by,
  label.by,
  node.size,
  node.size.factor,
  node.size.scale,
  node.size.range,
  node.color,
  node.color.gradient,
  node.color.range,
  node.label.size,
  arrow.size,
  edge.width,
  edge.label,
  show.color.legend,
  show.size.legend,
  main.title,
  sub.title,
  color.legend.title,
  size.legend.title,
  font.size,
  output.file
)
```

Arguments

`AntibodyForests_object`	AntibodyForests object - AntibodyForests object as obtained from the 'Af_build()' function in Platypus.
`sample`	string - denotes the sample that contains the clonotype.
`clonotype`	string - denotes the clonotype from which the lineage tree should be plotted.

show.inner.nodes
: boolean - if TRUE, the tree with inner nodes is plotted (only present when the trees are created with the 'phylo.tree.nj', 'phylo.tree.mp', phylo.tree.ml', or 'phylo.tree.IgPhyML' construction algorithm). Defaults to FALSE.

x.scaling
: float - specifies the range of the x axis and thereby scales the horizontal distance between the nodes. Defaults to a scaling in which the minimum horizontal space between two nodes equals 20% of the radius of the smallest node present in the tree (calculated using the 'calculate_optimal_x_scaling()' function).

y.scaling
: float - specifies the range of the y axis and thereby scales the vertical distance between the nodes. Defaults to a scaling in which the vertical space between the centers of two nodes equals 0.25 points in the graph.

color.by
: string - specifies the feature of the nodes that will be used for coloring the nodes. This sublist should be present in each sublist of each node in the 'nodes' objects within the AntibodyForests object. For each unique value for the selected feature, a unique color will be selected using the 'grDevices::rainbow()' function (unless a color gradient is created, see 'node.color.gradient' parameter). Defaults to 'isotype' (if present as feature of all nodes), otherwise defaults to NULL.

label.by
: string - specifies what should be plotted on the nodes. Options: 'name', 'size', a feature that is stored in the 'nodes' list, and 'none'. Defaults to 'name'.

node.size
: string or integer or list of integers - specifies the size of the nodes. If set to 'expansion', the nodes will get a size that is equivalent to the number of cells that they represent. If set to an integer, all nodes will get this size. If set to a list of integers, in which each item is named according to a node, the nodes will get these sizes. Defaults to 'expansion'.

node.size.factor
: integer - factor by which all node sizes are multiplied. Defaults to 1.

node.size.scale
: vector of 2 integers - specifies the minimum and maximum node size in the plot, to which the number of cells will be scaled. Defaults to 10 and 30.

node.size.range
: vector of 2 integers - specifies the the range of the node size scale. Defaults to the minimum and maximum node size.

node.color
: string or list of strings - specifies the color of nodes. If set to 'default', and the 'color.by' parameter is not specified, all the seqeuence-recovered nodes are colored lightblue. If set to 'default', and the 'color.by' parameter is set to a categorical value, the sequence-recovered nodes are colored If set to a color (a color from the 'grDevices::color()' list or a valid HEX code), all the sequence-recovered nodes will get this color. If set to a list of colors, in which each item is named to a node, the nodes will get these colors. Defaults to 'default'.

node.color.gradient
: vector of strings - specifies the colors of the color gradient, if 'color.by' is set to a numerical feature. The minimum number of colors that need to be specified are 2. Defaults to 'viridis'.

node.color.range
: - vector of 2 floats - specifies the range of the color gradient. Defaults to the minimum and maximum value found for the feature selected by the 'color.by' parameter.

`node.label.size`
: float - specifies the font size of the node label. Default scales to the size of the nodes.

`arrow.size`
: float - specifies the size of the arrows. Defaults to 1.

`edge.width`
: float - specifies the width of the edges. Defaults to 1.

`edge.label`
: string - specifies what distance between the nodes is shown as labels of the edges. Options: 'original' (distance that is stored in the igraph object), 'none' (no edge labels are shown), 'lv' (Levensthein distance), 'dl' (Damerau-Levenshtein distance), 'osa' (Optimal String Alignment distance), and 'hamming' (Hamming distance). Defaults to 'none'.

`show.color.legend`
: boolean - if TRUE, a legend is plotted to display the values of the specified node feature matched to the corresponding colors. Defaults to TRUE if the 'color.by' parameter is specified.

`show.size.legend`
: boolean - if TRUE, a legend is plotted to display the node sizes and the corresponding number of cells represented. Defaults to TRUE if the 'node.size' parameter is set to 'expansion'.

`main.title`
: string - specifies the main title of the plot (to be plotted in a bold font). Defaults to NULL.

`sub.title`
: string - specifies the sub title of the plot (to be plotted in an italic font below the main title). Defaults to NULL.

`color.legend.title`
: string - specifies the title of the legend showing the color matching. Defaults to the (capitalized) name of the feature specified in the 'color.by' parameter (converted by the 'stringr::str_to_title()' function).

`size.legend.title`
: string - specifies the title of the legend showing the node sizes. Defaults to 'Expansion (# cells)'.

`font.size`
: float - specifies the font size of the text in the plot. Defaults to 1.

`output.file`
: string - specifies the path to the output file (PNG of PDF). Defaults to NULL.

Value

No value returned, plots the lineage tree for the specified clonotype on the device or saves it to the output.file.

Examples

```
Af_plot_tree(AntibodyForests::small_af,
                     sample = "S1",
                     clonotype = "clonotype1",
                     main.title = "Lineage tree",
                     sub.title = "Sample 1 - clonotype 1")
```

`Af_sync_nodes` *Function to synchronize the node labels/names of all clonotypes within all samples of two AntibodyForests objects.*

Description

The nodes of each clonotype within each sample of the subject AntibodyForests object will be named according to the names of the nodes of the clonotypes within the samples of the reference AntibodyForests object. The node names present in all the objects within the Therefore, the sample IDs and clonotype IDs should be the same. Note: if a node in the reference AntibodyForests object is divided over two nodes in the subject AntibodyForests object, the nodes will get a letter as suffix (for example, 'node2' in the reference object would become 'node2A' and 'node2B' in the subject object). Note: if multiple nodes in the reference AntibodyForests object are together in one node in the subject AntibodyForests object, the number of the nodes are pasted together with a '+' (for example, 'node5' and 'node6' in the reference object would become 'node5+6' in the subject object).

Usage

```
Af_sync_nodes(reference, subject)
```

Arguments

`reference`	AntibodyForests object - AntibodyForests object as obtained from the 'Af_build()' function in Platypus. This object will be used as a reference.
`subject`	AntibodyForests object - AntibodyForests object as obtained from the 'Af_build()' function in Platypus. For each clonotype, the names of the nodes will be synced with the names of the nodes in the reference AntibodyForests object, by matching the barcodes.

Value

Returns the subject AntibdoyForests object in which all nodes of each clonotypes within all samples are renamed.

Examples

```
af_mst <- Af_sync_nodes(reference = AntibodyForests::af_default,
                        subject = AntibodyForests::af_mst)
```

`Af_to_newick` *Saves an AntibodyForests-object into a newick file*

Description

Saves an AntibodyForests-object into a newick file. The node labels will have the format node\@size where size is the size of the node.

Usage

```
Af_to_newick(AntibodyForests_object, min.nodes, output.file)
```

Arguments

`AntibodyForests_object`	AntibodyForests-object, output from Af_build()
`min.nodes`	The minimum number of nodes in a tree to calculate metrics (including the germline).
`output.file`	string - specifies the path to the output file

Value

No value returned, saves the newick format to the output.file

Examples

```
## Not run:
Af_to_newick(AntibodyForests_object = AntibodyForests::small_af,
             min.nodes = 2,
             output.file = "output.newick")

## End(Not run)
```

`calculate_GBLD` *Calculate the GBLD distance between trees in an AntibodyForests object. Code is derived from https://github.com/tahiri-lab/ClonalTreeClustering/blob/main/src/Python/GBLD_Metric_Final.ipynb Farnia, M., Tahiri, N. New generalized metric based on branch length distance to compare B cell lineage trees. Algorithms Mol Biol 19, 22 (2024). https://doi.org/10.1186/s13015-024-00267-1*

Description

Calculate the GBLD distance between trees in an AntibodyForests object. Code is derived from https://github.com/tahiri-lab/ClonalTreeClustering/blob/main/src/Python/GBLD_Metric_Final.ipynb

Usage

```
calculate_GBLD(AntibodyForests_object, min.nodes)
```

Arguments

AntibodyForests_object	AntibodyForests-object, output from AntibodyForests()
min.nodes	• integer - The minimum number of nodes (including the germline) in a tree to include in the analysis. Default is 3.

Value

A matrix with the GBLD distances between trees in the AntibodyForests object.

Examples

```
GBLD_matrix <- calculate_GBLD(AntibodyForests_object = AntibodyForests::small_af)
GBLD_matrix[1:5, 1:5]
```

---

compare_repertoire *Example output from Af_compare_within_repertoires() for function testing purposes*

Description

Example output from Af_compare_within_repertoires() for function testing purposes

Usage

```
compare_repertoire
```

Format

An object of class `list` of length 1.

igraph_to_phylo *Converts an igraph network into a phylogenetic tree as a phylo object.*

Description

Converts an igraph network into a phylogenetic tree as a phylo object.

Usage

```
igraph_to_phylo(tree, solve_multichotomies)
```

Arguments

tree	igraph object
solve_multichotomies	boolean - whether to remove multichotomies in the resulting phylogenetic tree using ape::multi2di

Value

phylogenetic tree

Examples

```
phylo_object <- igraph_to_phylo(AntibodyForests::small_af[["S1"]][["clonotype1"]]$igraph)
```

newick_to_Af *Converts files with phylogenetic trees in newick format into an AntibodyForests object.*

Description

Converts files with phylogenetic trees in newick format into an AntibodyForests object. Make sure that the germline node is called "germline" and that every line represents a new tree in the newick file. All trees in the same file should be from the same sample.

Usage

```
newick_to_Af(file.list, file.dir)
```

Arguments

file.list	list - list of newick files to be converted to AntibodyForests object. Could be a named list where the names correspond to sample IDs.
file.dir	directory - directory where the newick files are stored. If provided, the function will read all newick files in the directory.

Value

AntibodyForests object

Examples

```
## Not run:
af <- newick_to_Af(file.list = list("S1" = "path/to/sample1.nwk", "S2" = "path/to/sample2.nwk"))

## End(Not run)
```

PLM_dataframe *Small PLM dataframe for function testing purposes*

Description

Small PLM dataframe for function testing purposes

Usage

```
PLM_dataframe
```

Format

An object of class `tbl_df` (inherits from `tbl`, `data.frame`) with 60 rows and 11 columns.

---

small_af *Small AntibodyForests object for function testing purposes*

Description

Small AntibodyForests object for function testing purposes

Usage

```
small_af
```

Format

An object of class `AntibodyForests` of length 5.

small_vdj *Small VDJ dataframe for function testing purposes*

Description

Small VDJ dataframe for function testing purposes

Usage

```
small_vdj
```

Format

An object of class `data.frame` with 3671 rows and 70 columns.

VDJ_3d_properties *Function to calculate 3D-structure propoperties such as the average charge and hydrophobicity, pKa shift, free energy, RMSD of PDB files and add them to an AntibodyForests-object*

Description

Function to calculate protein 3D-structure properties of antibodies (or antibody-antigen complexes) and integrate them into an AntibodyForests-object.

Usage

```
VDJ_3d_properties(
  VDJ,
  pdb.dir,
  file.df,
  properties,
  sequence.region,
  chain,
  propka.dir,
  free_energy_pH,
  sub.sequence.column,
  germline.pdb,
  foldseek.dir
)
```

Arguments

VDJ	a dataframe with V(D)J information such as the output of Platypus::VDJ_build(). Must contain columns sample_id, clonotype_id, barcode.
pdb.dir	a directory containing PDB files.
file.df	a dataframe of pdb filenames (column file_name) to be used and sequence IDs (column sequence) corresponding to the the barcodes column of the VDJ dataframe.
properties	a vector of properties to be calculated. Default is c("charge", "hydrophobicity"). • charge: The net electrical charge at pH 7.0 • hydrophobicity: The hypdrophobicity of each amino acid, devided by the sequence length. • RMSD_germline: the root mean square deviation to the germline structure (needs the germline pdb) • 3di_germline: the edit distance between the 3di sequence of each sequences and the germline sequence (needs foldseek output). • pKa_shift: the acid dissociation constant shift upon binding of the antibody to the antigen (needs Propka output) • free_energy: the free energy of binding of the antibody to the antigen at a certain pH (needs Propka output) • pLDDT: the pLDDT score of the model
sequence.region	a character vector of the sequence region to be used to calculate properties. Default is "full.sequence". • full.sequence: the full sequence(s) in the PDB file • sub.sequence: part of the full sequence, for example the CDR3 region in the PDB file. This sub sequence must be a column in the VDJ dataframe. • binding.residues: the binding residues in the PDB file
chain	a character vector of the chain to be used to calculate properties. Default is both heavy and light chain Assuming chain "A" is heavy chain, chain "B" is light chain, and possible chain "C" is the antigen. • HC+LC: both heavy and light chain • HC: heavy chain, assuming chain A is the heavy chain. • LC: light chain, assuming chain B is the light chain. • AG: antigen, assuming chain C is the antigen. • whole.complex: the whole complex of antibody-antigen (all available chains in the pdb file).
propka.dir	a directory containing Propka output files. The propka filenames should be similar to the PDB filenames.
free_energy_pH	the pH to be used to calculate the free energy of binding. Default is 7.
sub.sequence.column	a character vector of the column name in the VDJ dataframe containing the sub sequence to be used to calculate properties. Default is NULL.
germline.pdb	PDB filename of the germline. Default is NULL.

`foldseek.dir` a directory containing dataframes with the Foldseek 3di sequence per chain for each sequence. Filenames should be similar to the PDB filenames and it needs to have column "chain" containing the 'A', 'B', and/or 'C' chain. Default is NULL.

Value

the input VDJ dataframe with the calculated 3D-structure properties.

Examples

```
## Not run:
vdj_structure_antibody <- VDJ_3d_properties(VDJ = AntibodyForests::small_vdj,
                          pdb.dir = "~/path/PDBS_superimposed/",
                          file.df = files,
                          properties = c("charge", "3di_germline", "hydrophobicity"),
                          chain = "HC+LC",
                          sequence.region = "full.sequence",
                          propka.dir = "~/path/Propka_output/",
                        germline.pdb = "~/path/PDBS_superimposed/germline_5_model_0.pdb",
                          foldseek.dir = "~/path/3di_sequences/")

## End(Not run)
```

VDJ_import_IgBLAST_annotations

Function to import the annotations and alignments from IgBLAST output into the VDJ dataframe.

Description

Imports the IgBLAST annotations and alignments from IgBLAST output files, stored in the output folders of Cell Ranger, into a VDJ dataframe obtained from the VDJ_build() function in Platypus.

Usage

```
VDJ_import_IgBLAST_annotations(VDJ, VDJ.directory, file.path.list, method)
```

Arguments

`VDJ` dataframe - VDJ object as obtained from the VDJ_build() function in Platypus.

`VDJ.directory` string - path to parent directory containing the output folders (one folder for each sample) of Cell Ranger. This pipeline assumes that the sample IDs and contigs IDs have not been modified and that the IgBLAST output file names have not been changed from the default changeo settings. Each sample directory should contain a 'filtered_contig_igblast_db-pass.tsv' file.

`file.path.list` list - list containing the paths to the 'filtered_contig_igblast_db-pass.tsv' files, in which the names of each item should refer to an sample ID.

method — string - denotes the way the IgBLAST germline annotations from the 'filtered_contig_igblast_db-pass.tsv' files should be appended to the VDJ dataframe. Options: 'replace' or 'attach'. Defaults to 'append'. 'replace' : The original annotation columns in the VDJ dataframe are replaced with the IgBLAST annotations. The original columns are kept with the suffix '_10x'. 'append' : The IgBLAST annotation columns are stored in columns with the suffix '_IgBLAST'.

Value

The VDJ dataframe with the appended or replaced IgBLAST annotations and alignments.

Examples

```
## Not run:
VDJ <- VDJ_import_IgBLAST_annotations(VDJ = AntibodyForests::small_vdj,
                                      VDJ.directory = "path/to/VDJ_directory")

## End(Not run)
```

VDJ_integrate_bulk *A function to integrate bulk and single cell data*

Description

Integrate bulk and single-cell data by reannotating the germline genes and integrating the bulk sequences into the existing single-cell clonotypes.

Usage

```
VDJ_integrate_bulk(
  sc.VDJ,
  bulk.tsv,
  bulk.tsv.sequence.column,
  bulk.tsv.sample.column,
  bulk.tsv.barcode.column,
  bulk.tsv.isotype.column,
  organism,
  scRNA_seqs_annotations,
  bulkRNA_seqs_annotations,
  igblast.dir,
  trim.FR1,
  tie.resolvement,
  seq.identity
)
```

Arguments

sc.VDJ	VDJ dataframe of the single cell data created with Platypus VDJ_build function.
bulk.tsv	A tab separated file of the bulk sequences with the at least columns containing the sequence, a sample ID, a barcode, and the isotype.
bulk.tsv.sequence.column	column name of the bulk tsv that contains the nucleotide sequence
bulk.tsv.sample.column	column name of the bulk tsv that contains the sample_id that matches the sample_id in sc_VDJ
bulk.tsv.barcode.column	column name of the bulk tsv that contains the barcode/identifier of the recovered sequence
bulk.tsv.isotype.column	column name of the bulk tsv that contains the isotype of the recovered sequence
organism	"human" or "mouse"
scRNA_seqs_annotations	A tab separated file of the reannotated single-cell sequences using Change-O AssignGenes.py. If NULL, this function will run Change-O AssignGenes.py (Make sure to have this installed, including igblast.dir). Default is NULL.
bulkRNA_seqs_annotations	A tab separated file of the reannotated bulk sequences using Change-O AssignGenes.py. If NULL, this function will run Change-O AssignGenes.py (Make sure to have this installed, including igblast.dir). Default is NULL.
igblast.dir	directory where the igblast executables are located. For example: use the instruction to set up IgPhyML environment in the AntibodyForests vignette ($(conda info –base)/envs/igphyml/share/igblast)
trim.FR1	• boolean - whether to trim the FR1 region from the sequences and germline, this is recommended to account for variation in primer design during sequencing (Default is TRUE)
tie.resolvement	How to resolve a bulk sequence for which multiple clonotypes match. "all" - assign the bulk sequence to all matching clonotypes (Default) "none" - do not assign the bulk sequence to any clonotype "random" - randomly assign the bulk sequence to one of the matching clonotypes
seq.identity	sequence identity threshold for clonotype assignment (Default: 0.85)

Value

The VDJ dataframe of both the bulk and single-cell data

Examples

```
## Not run:
VDJ <- VDJ_integrate_bulk(sc_VDJ = AntibodyForests::small_vdj,
  bulk_tsv = "bulk_rna.tsv",
  bulk_tsv_sequence_column = "sequence",
```

```
  bulk_tsv_sample_column = "sample_id",
  bulk_tsv_barcode_column = "barcode",
  bulk_tsv_isotype_column = "isotype",
  organism = "human",
  igblast_dir = "anaconda3/envs/igphyml/share/igblast",
  tie_resolvement = "random",
  seq_identity = 0.85)

## End(Not run)
```

VDJ_to_AIRR *Function to convert VDJ dataframe into an AIRR-formatted TSV file.*

Description

Takes a VDJ dataframe along with the imported IgBLAST annotations and alignments and converts it into a tab-separated values (TSV) file formatted according to the AIRR (Adaptive Immune Receptor Repertoire) guidelines.

Usage

```
VDJ_to_AIRR(
  VDJ,
  include,
  columns,
  complete.rows.only,
  filter.rows.with.stop.codons,
  output.file
)
```

Arguments

VDJ	dataframe - VDJ object as obtained from the 'VDJ_build()' function in Platypus, together with the imported IgBLAST annotations and alignments, as obtained from the 'import_IgBLAST_annotations' function in AntibodyForests.
include	list - a nested list specifying the samples and their associated clonotypes to include in the output TSV file. Each sublist represents a sample, where the sublist name is the sample name and the elements within the sublist are the clonotypes of that sample. If not provided, all samples and clonotypes are included.
columns	list - a list specifying the columns to include in the output TSV file. At minimum, the following columns must be specified: 'sequence_id', 'clone_id', 'sequence', 'sequence_alignment', 'germline_alignment', 'v_call', 'v_sequence_start', 'v_sequence_end', 'v_germline_start', 'v_germline_end', 'j_call', 'j_sequence_start', 'j_sequence_end', 'j_germline_start', and 'j_germline_end'. The items in this list should correspond to the column names in the VDJ dataframe, while the names of the items in this list should refer to the column names of the output TSV file.

`complete.rows.only`
: bool - if TRUE, only complete rows (without any missing values) are included in the output TSV file. If FALSE, rows with missing values are retained in the output. Defaults to TRUE.

`filter.rows.with.stop.codons`
: bool - if TRUE, rows containing sequences with stop codons (TAA, TAG, TGA) in the 'sequence_alignment' and 'germline_alignment' columns are filtered out from the output TSV file. Defaults to TRUE.

`output.file`
: string - string specifying the path to the output file. If no path is specified, the output is written to 'airr_rearrengement.tsv' in the current working directory.

Value

None

Examples

```
## Not run:
VDJ_to_AIRR(VDJ = VDJ_IgBLAST, output.file = "path/to/output.tsv")

## End(Not run)
```

Index