Package: MHCtools (via r-universe)

June 2, 2024

Type  Package
Title  Analysis of MHC Data in Non-Model Species
Version  1.5.3

Description  Fifteen tools for bioinformatics processing and analysis of major histocompatibility complex (MHC) data. The functions are tailored for amplicon data sets that have been filtered using the dada2 method (for more information on dada2, visit <https://benjjneb.github.io/dada2/> ), but even other types of data sets can be analyzed. The ReplMatch() function matches replicates in data sets in order to evaluate genotyping success. The GetReplTable() and GetReplStats() functions perform such an evaluation. The CreateFas() function creates a fasta file with all the sequences in the data set. The CreateSamplesFas() function creates individual fasta files for each sample in the data set. The DistCalc() function calculates Grantham, Sandberg, or p-distances from pairwise comparisons of all sequences in a data set, and mean distances of all pairwise comparisons within each sample in a data set. The function additionally outputs five tables with physico-chemical z-descriptor values (based on Sandberg et al. 1998) for each amino acid position in all sequences in the data set. These tables may be useful for further downstream analyses, such as estimation of MHC supertypes. The BootKmeans() function is a wrapper for the kmeans() function of the 'stats' package, which allows for bootstrapping. Bootstrapping k-estimates may be desirable in data sets, where e.g. BIC- vs. k-values do not produce clear inflection points ("elbows"). BootKmeans() performs multiple runs of kmeans() and estimates optimal k-values based on a user-defined threshold of BIC reduction. The method is an automated and bootstrapped version of visually inspecting elbow plots of BIC- vs. k-values. The ClusterMatch() function is a tool for evaluating whether different k-means() clustering models identify similar clusters, and summarize bootstrap model stats as means for different estimated values of k. It is designed to take files produced by the BootKmeans()
function as input, but other data can be analysed if the
descriptions of the required data formats are observed
carefully. The PapaDiv() function compares parent pairs in the
data set and calculate their joint MHC diversity, taking into
account sequence variants that occur in both parents. The
HpltFind() function infers putative haplotypes from families in
the data set. The GetHpltTable() and GetHpltStats() functions
evaluate the accuracy of the haplotype inference. The
CreateHpltOccTable() function creates a binary (logical)
haplotype-sequence occurrence matrix from the output of
HpltFind(), for easy overview of which sequences are present in
which haplotypes. The HpltMatch() function compares haplotypes
to help identify overlapping and potentially identical types.
The NestTablesXL() function translates the output from
HpltFind() to an Excel workbook, that provides a convenient
overview for evaluation and curating of the inferred putative
haplotypes.

License  MIT + file LICENSE
Encoding  UTF-8
LazyData  true
Imports  stats, utils, mgcv, grDevices, graphics, openxlsx
RoxygenNote  7.2.3
NeedsCompilation  no
Author  Jacob Roved [aut, cre]
Maintainer  Jacob Roved <jacob.roved@biol.lu.se>
Depends  R (>= 3.5.0)
Date/Publication  2023-07-08 13:10:02 UTC
Repository  https://jr-evolecol.r-universe.dev
RemoteUrl  https://github.com/cran/MHCtools
RemoteRef  HEAD
RemoteSha  e5d8b4578c9b581d6adf498d49d661702928cf91

Contents

BootKmeans ......................................................... 3
ClusterMatch ....................................................... 5
CreateFas .......................................................... 7
CreateHpltOccTable ............................................... 8
CreateSamplesFas ............................................... 9
DistCalc .......................................................... 10
GetHpltStats ..................................................... 12
GetHpltTable ..................................................... 13
GetReplStats ..................................................... 14
**BootKmeans**

**Description**

*BootKmeans* is a wrapper for the *kmeans()* function of the 'stats' package, which allows for bootstrapping. Bootstrapping k-estimates may be desirable in data sets, where the BIC- vs. k-values do not produce clear inflection points ("elbows").

**Usage**

```r
BootKmeans(
  z1_matrix,
  z2_matrix,
  z3_matrix,
  z4_matrix,
  z5_matrix,
  threshold = 0.01,
  no_scans = 1000,
  max_k = 40,
  iter.max = 1e+06,
  nstart = 200,
  algorithm = "Hartigan-Wong",
  path_out = path_out
)
```
Arguments

- **z1_matrix**
  a matrix with numerical values of the first z-descriptor for each amino acid position in all sequences in the data set.

- **z2_matrix**
  a matrix with numerical values of the second z-descriptor for each amino acid position in all sequences in the data set.

- **z3_matrix**
  a matrix with numerical values of the third z-descriptor for each amino acid position in all sequences in the data set.

- **z4_matrix**
  a matrix with numerical values of the fourth z-descriptor for each amino acid position in all sequences in the data set.

- **z5_matrix**
  a matrix with numerical values of the fifth z-descriptor for each amino acid position in all sequences in the data set.

- **threshold**
  a numerical value between 0 and 1 specifying the threshold of reduction in BIC for selecting a k estimate for each kmeans clustering model. The value specifies a proportion of the max observed reduction in BIC when increasing k by 1 (default 0.01).

- **no_scans**
  an integer specifying the number of k estimation scans to run (default 1,000).

- **max_k**
  an integer specifying the hypothetical maximum number of clusters to detect (default 40). In each k estimation scan, the algorithm runs a kmeans() clustering model for each value of k between 1 and max_k.

- **iter.max**
  an integer specifying the maximum number of iterations allowed in each kmeans() clustering model (default 1,000,000).

- **nstart**
  an integer specifying the number of rows in the set of input matrices that will be chosen as initial centers in the kmeans() clustering models (default 200).

- **algorithm**
  character vector, specifying the method for the kmeans() clustering function, one of c("Hartigan-Wong", "Lloyd", "Forgy", "MacQueen"), default is "Hartigan-Wong".

- **path_out**
  a user defined path to the folder where the output files will be saved.

Details

BootKmeans() performs multiple runs of kmeans() scanning k-values from 1 to a maximum value defined by the user. In each scan, an optimal k-value is estimated using a user-defined threshold of BIC reduction. The method is an automated version of visually inspecting elbow plots of BIC vs. k-values. The number of scans to be performed is defined by the user.

For each k-estimate scan, the algorithm produces a summary of the stats incl. total within SS, AIC, and BIC, an elbow plot (BIC vs. k), and a set of cluster files corresponding to the estimated optimal k-value. It also produces a table summarizing the stats of the final selected kmeans() models corresponding to the estimated optimal k-values of each scan.

After running BootKmeans() on a data set, it is recommended to subsequently evaluate the repeatability of the bootstrapped k-estimation scans with the ClusterMatch() function also included in MHCtools.

Input data format: A set of five z-matrices containing numerical values of the z-descriptors (z1-z5) for each amino acid position in a sequence alignment. Each column should represent an amino acid position and each row one sequence in the alignment.

Value

The function produces three folders in path_out, which contain for each scan the estimated k-clusters saved as .Rdata files, an elbow plot saved as .pdf, and a stats summary table saved as a .csv file. In path_out a summary of all scans performed in the bootstrap run is also saved as .csv. This table is also shown in the console. Should alternative elbow plots be desired, they may be produced manually with the stats presented in the summary tables for each scan.

Note

AIC and BIC are calculated from the kmeans model objects by the following formulae: - AIC = D + 2*m*k - BIC = D + log(n)*m*k in which: - m = ncol(fit$centers) - n = length(fit$cluster) - k = nrow(fit$centers) - D = fit$tot.withinss

See Also

ClusterMatch; DistCalc

Examples

```r
z1_matrix <- z1_matrix
z2_matrix <- z2_matrix
z3_matrix <- z3_matrix
z4_matrix <- z4_matrix
z5_matrix <- z5_matrix
path_out <- tempdir()
BootKmeans(z1_matrix, z2_matrix, z3_matrix, z4_matrix, z5_matrix, threshold=0.01,
no_scans=10, max_k=20, iter.max=10, nstart=10, algorithm="Hartigan-Wong",
path_out=path_out)
```

Description

ClusterMatch is a tool for evaluating whether k-means() clustering models with similar estimated values of k identify similar clusters. ClusterMatch() also summarizes model stats as means for different estimated values of k. It is designed to take files produced by the BootKmeans() function as input, but other data can be analyzed if the descriptions of the data formats given below are observed carefully.
Usage

ClusterMatch(filepath, path_out, k_summary_table)

Arguments

filepath

a user defined path to a folder that contains the set of K-cluster files to be matched against each other. The algorithm will attempt to load all files in the folder, so it should contain only the relevant K-cluster files. If the clusters were generated using the BootKmeans() function, such a folder (named Clusters) was created by the algorithm in the output path given by the user. Each K-cluster file should correspond to the model$cluster object in kmeans() saved as a .Rdata file. Such files are generated as part of the output from BootKmeans(). ClusterMatch() assumes that the file names contain the string "model_" followed by a model number, which must match the corresponding row numbers in k_summary_table. If the data used was generated with the BootKmeans() function, the formats and numbers will match by default.

path_out

a user defined path to the folder where the output files will be saved.

k_summary_table

a data frame summarizing the stats of the kmeans() models that produced the clusters in the K-cluster files. If the data used was generated with the BootKmeans() function, a compatible k_summary_table was produced in the output path with the file name "k_means_bootstrap_summary_stats_<date>.csv". If other data is analyzed, please observe these formatting requirements: The k_summary_table must contain the data for each kmeans() model in rows and as minimum the following columns: - k-value (colname: k.est) - residual total within sums-of-squares (colname: Tot.withinss.resid) - residual AIC (colname: AIC.resid) - residual BIC (colname: BIC.resid) - delta BIC/max BIC (colname: prop.delta.BIC) - delta BIC/k.est (colname: delta.BIC.over.k) It is crucial that the models have the same numbers in the K-cluster file names and in the k_summary_table, and that the rows of the table are ordered by the model number.

Details


Value

The function returns a summary table, which for each estimated number of clusters (i.e. the k-values of the models) lists: - number of models that found i clusters - mean residual total within sums-of-squares - mean residual AIC - mean residual BIC - mean delta BIC/max BIC - mean delta BIC/k - mean number of allele assignments that fall outside of the i most abundant clusters across all pairwise comparisons between the models that found i clusters - mean proportion of allele assignments that fall outside of the i most abundant clusters across all pairwise comparisons
CreateFas

between the models that found i clusters The summary table is also saved as a .csv file in the output path.

See Also

BootKmeans

Examples

filepath <- system.file("extdata/ClusterMatch", package="MHCTools")
path_out <- tempdir()
k_summary_table <- k_summary_table
ClusterMatch(filepath, path_out, k_summary_table)

---

CreateFas

CreateFas() function

Description

CreateFas creates a FASTA file with all the sequences in a 'dada2' sequence table.

Usage

CreateFas(seq_table, path_out)

Arguments

seq_table seq_table is a sequence table as output by the 'dada2' pipeline, which has samples in rows and nucleotide sequence variants in columns.

path_out is a user defined path to the folder where the output files will be saved.

Details


Value

A FASTA file with all the sequences in a 'dada2' sequence table. The sequences are named in the FASTA file by an index number corresponding to their column number in the sequence table.

See Also

CreateSamplesFas; for more information about 'dada2' visit <https://benjjneb.github.io/dada2/>
CreateHpltOccTable

Examples

```r
seq_table <- sequence_table_fas
path_out <- tempdir()
CreateFas(seq_table, path_out)
```

Description

`CreateHpltOccTable` is designed to create a haplotype-sequence occurrence matrix from the set of R lists with putative haplotypes output by the `HpltFind()` function. `CreateHpltOccTable()` assumes that data originated from a diploid species.

Usage

```r
CreateHpltOccTable(seq_table, filepath, path_out)
```

Arguments

- `seq_table` is a sequence table as output by the 'dada2' pipeline, which has samples in rows and nucleotide sequence variants in columns.
- `filepath` is a user defined path to the folder where the output files from the `HpltFind()` function have been saved.
- `path_out` is a user defined path to the folder where the output files will be saved.

Details


Value

A binary (logical) occurrence matrix with the data set sequences (inherited from `seq_table`) in columns and the putative haplotypes inferred by the `HpltFind()` function in rows.

See Also

`HpltFind`; for more information about 'dada2' visit <https://benjjneb.github.io/dada2/>
CreateSamplesFas

Examples

```r
seq_table <- sequence_table
filepath <- system.file("extdata/HpltFindOut/", package="MHCtools")
path_out <- tempdir()
CreateHpltOccTable(seq_table, filepath, path_out)
```

CreateSamplesFas

Description

CreateSamplesFas creates a set of FASTA files with the sequences present in each sample in a `dada2` sequence table.

Usage

CreateSamplesFas(seq_table, path_out)

Arguments

- `seq_table` is a sequence table as output by the `dada2` pipeline, which has samples in rows and nucleotide sequence variants in columns.
- `path_out` is a user defined path to the folder where the output files will be saved.

Details


Value

A set of FASTA files with the sequences present in each sample in the sequence table. The sequences are named in the FASTA files by an index number corresponding to their column number in the sequence table, thus identical sequences will have identical sample names in all the FASTA files.

See Also

CreateFas; for more information about `dada2` visit <https://benjjneb.github.io/dada2/>

Examples

```r
seq_table <- sequence_table_fas
path_out <- tempdir()
CreateSamplesFas(seq_table, path_out)
```
DistCalc() function

Description

DistCalc calculates Grantham distances, Sandberg distances, or p-distances from pairwise comparisons of aligned sequences.

Usage

DistCalc(
  seq_file,
  path_out,
  input_fasta = NULL,
  input_seq = "aa",
  aa_dist = NULL,
  codon_pos = NULL,
  dist_type = "G"
)

Arguments

seq_file is a sequence occurrence table as output by the 'dada2' pipeline, which has samples in rows and nucleotide sequence variants in columns. Optionally, a fasta file can be supplied as input in the format rendered by read.fasta() from the package 'seqinr'.

path_out is a user defined path to the folder where the output files will be saved.

input_fasta optional, a logical (TRUE/FALSE) that indicates whether the input file is a fasta file (TRUE) or a 'dada2'-style sequence table (NULL/FALSE). The default is NULL/FALSE.

input_seq defines the type of sequences in seq_file. It may take the values 'nucl' or 'aa'.

aa_dist is optional, a logical (TRUE/FALSE) that determines whether nucleotide sequences should be translated to amino acid sequences before distance calculation, default is NULL/FALSE. Note that aa_dist must be set to TRUE, if Grantham or Sandberg distances are calculated from an alignment of nucleotide sequences.

codon_pos is optional, a vector of comma separated integers specifying which codons to include in distance calculations. If omitted, distance calculations are made using all codons. Note: When calculating nucleotide P-distances, codon_pos should be specified as a vector of nucleotide positions.

dist_type is used to specify which kind of distances that are calculated. It takes the values 'G' for Grantham distances, 'S' for Sandberg distances, or 'P' for p-distances. The argument is optional with 'G' as default setting.
Details

The DistCalc() function takes a fasta file or a ‘dada2’-style sequence occurrence table (with aligned sequences as column names and samples in rows) as input and produces a matrix with pairwise distances for all sequences in the data set. If calculation of Sandberg distances is specified, the function additionally outputs five tables with physico-chemical z-descriptor values (based on Sandberg et al. 1998) for each amino acid position in all sequences in the data set. These tables may be useful for further downstream analyses, such as estimation of MHC supertypes. If a sequence occurrence table is provided as input, the DistCalc() function furthermore produces a table with the mean distances from all pairwise comparisons of the sequences in each sample in the data set. (Note: The mean distance will be NA for samples that have 0 or 1 sequence(s).)

Grantham distances and Sandberg distances are calculated as described in Pierini & Lenz 2018. The Grantham distances produced by DistCalc() are simply the mean Grantham distances (Grantham 1974) between all amino acid codons in sequence pairs. When calculating Sandberg distances, DistCalc() first computes Euclidian distances between all amino acid pairs based on the five physico-chemical z-descriptors defined in Sandberg et al. 1998. Sandberg distances are then calculated as the mean Euclidian distances between all amino acid codons in sequence pairs. P-distances calculated by DistCalc() are simply the proportion of varying codons between pairs of sequences.

The DistCalc() function includes an option for the user to specify which codons to compare, which is useful e.g. if conducting the analysis only on codons involved in specific functions, such as peptide binding of an MHC molecule. Note: When calculating nucleotide P-distances, codon_pos is applied directly on the nucleotide sequences. This allows the user to calculate divergence in e.g. first, second, or third codon positions. Hence, codon_pos should be specified as a vector of nucleotide positions when calculating nucleotide P-distances.

DistCalc() also accepts calculating amino acid distances directly from protein-coding DNA sequences using the standard genetic code.


It accepts gaps defined by ‘-’. Nucleotide triplets containing gaps are translated to ‘X’, if amino acid distances are calculated directly from DNA nucleotide sequences. Please note that ‘-’ or ‘X’ are treated as unique characters in p-distance calculations. The function will not accept ‘X’ or gaps in Grantham or Sandberg distance calculations. If you wish to exclude codons with ‘X’ or gaps from distance calculations, please use the codon_pos option to specify which codons to compare.


The function returns a matrix with distances from all pairwise sequence comparisons, where n is the number of sequences. If a sequence occurrence table is given as input file, the function additionally returns a table with the mean distance for each sample in the data set. If a sequence occurrence table is given as input file, the sequences are named in the output matrix by an index number that corresponds to their column number in the input file. If calculation of Sandberg distances is specified, the function additionally outputs five tables with physico-chemical z-descriptor values for each amino acid position in all sequences in the data set. All output tables are saved as .csv files in the output path.

See Also

For more information about 'dada2', visit <https://benjjneb.github.io/dada2/>

Examples

```r
seq_file <- sequence_table_fas
path_out <- tempdir()
DistCalc(seq_file, path_out, input_fasta=NULL, input_seq="nucl", aa_dist=NULL, codon_pos=c(1,2,3,4,5,6,7,8), dist_type="P")
```

Description

**GetHpltStats** uses the output files produced by the HpltFind() function to calculate the mean of the mean proportion of incongruent sequences across all nests in the data set.

Usage

`GetHpltStats(filepath)`

Arguments

- `filepath` is a user defined path to the folder where the output files from the HpltFind() function have been saved.

Details

### Description

*GetHpltTable* uses the output files produced by the *HpltFind()* function to produce a table with the mean proportion of incongruent sequences for each nest. If the mean proportion of incongruent sequences is generally low, but certain nests have many incongruent sequences, biological reasons may be causing the mismatches, e.g. extra-pair fertilizations or recombination events.

### Usage

```r
GetHpltTable(filepath)
```

### Arguments

- `filepath` is a user defined path to the folder where the output files from the *HpltFind()* function have been saved.

### Details


### Value

A table with the mean proportion of incongruent sequences for each nest.

### See Also

*HpltFind; GetHpltStats*
GetReplStats

Examples

```r
filepath <- system.file("extdata/HpltFindOut/", package="MHCtools")
GetHpltTable(filepath)
```

GetReplStats

GetReplStats function

Description

GetReplStats uses the output files produced by the ReplMatch() function to calculate statistics on the agreement between replicated samples in the sequencing experiment.

Usage

```r
GetReplStats(filepath)
```

Arguments

- `filepath` is a user defined path to the folder where the output files from the ReplMatch() function have been saved.

Details


Value

A list containing the number of replicate sets with zero incongruent sequences, the proportion of replicate sets with zero incongruent sequences, the mean of the mean proportion of incongruent sequences across all replicate sets, and the repeatability of the sequencing experiment.

See Also

ReplMatch; GetReplTable

Examples

```r
filepath <- system.file("extdata/ReplMatchOut/", package="MHCtools")
GetReplStats(filepath)
```
GetReplTable function

Description

GetReplTable uses the output files produced by the ReplMatch() function to produce a table with the replicate sets and their respective mean proportion of incongruent sequences.

Usage

GetReplTable(filepath)

Arguments

filepath is a user defined path to the folder where the output files from the ReplMatch() function have been saved.

Details


Value

A table with the mean proportion of incongruent sequences for each replicate set.

See Also

ReplMatch; GetReplStats

Examples

filepath <- system.file("extdata/ReplMatchOut/", package="MHCtools")
GetReplTable(filepath)
HpltFind

Description

HpltFind is designed to automatically infer major histocompatibility complex (MHC) haplotypes from the genotypes of parents and offspring in families (defined as nests) in non-model species, where MHC sequence variants cannot be identified as belonging to individual loci. HpltFind() assumes that data originated from a diploid species. The functions GetHpltTable(), GetHpltStats(), and NestTablesXL() are designed to evaluate the output files.

Usage

HpltFind(nest_table, seq_table, alpha = 0.8, path_out)

Arguments

nest_table is a table containing the sample names of parents and offspring in each nest. This table should be organized so that the individual names are in the first column (Sample_ID), and the nest number is in the second column (Nest). For each nest, the first two rows should be the parents, followed immediately by the offspring in the subsequent rows, and then followed by the next nest, and so on. It is assumed that nests are numbered consecutively beginning at 1.

seq_table seq_table is a sequence table as output by the 'dada2' pipeline, which has samples in rows and nucleotide sequence variants in columns.

alpha a numerical value between 0 and 1 (default 0.8) specifying a threshold by which a set of sequences overlapping between a chick and a parent will be assigned to the putative parental A haplotype or passed to the B haplotype. Typical values are in the range 0.6-0.9. In data sets with many different MHC alleles per individual (i.e. many MHC gene copies), alpha may be set high. In data sets with fewer MHC alleles per individual, it should be set lower. A range of alpha values may be tested to find the optimal setting for a given data set, e.g. by evaluating the mean proportion of incongruent sequences across the data set using GetHpltStats().

path_out is a user defined path to the folder where the output files will be saved.

Details

HpltMatch

Value

A set of R lists containing for each nest the putative haplotypes, the names of sequences that could not be resolved with certainty in each parent, the names of the sequences that were incongruent in the genotypes of the nest, and the mean proportion of incongruent sequences (which is a measure of the haplotype inference success and largely influenced by the exactness of the genotyping experiment). The sequences are named in the output by an index number corresponding to their column number in the sequence table, thus identical sequences will have identical sample names in all the output files. These files can be reopened in R e.g. using the readRDS() function in the base package. Note: HpltFind() will overwrite any existing files with the same output file names in path_out.

See Also

GetHpltTable; GetHpltStats; NestTablesXL; CreateHpltOccTable; for more information about 'dada2' visit <https://benjjneb.github.io/dada2/>

Examples

nest_table <- nest_table
seq_table <- sequence_table
path_out <- tempdir()
HpltFind(nest_table, seq_table, alpha=0.8, path_out)

HpltMatch

HpltMatch() function

Description

Putative haplotypes may be identical to each other, or they may differ only by incongruent or unresolved sequences. It is therefore useful to curate putative haplotypes by comparing them to identify potentially overlapping types as candidates for further investigation. HpltMatch calculates the proportion of matching sequences between pairs of haplotypes and produces a .csv table with values in a lower left matrix. If a threshold value is specified, a list of haplotype matches where the proportion of matching sequences exceeds the threshold will be produced.

Usage

HpltMatch(hplt_occ_matrix, path_out, threshold = NULL)

Arguments

hplt_occ_matrix

A binary (logical) occurrence matrix with the data set sequences in columns and the putative haplotypes in rows, as produced by the CreateHpltOccTable() function.

path_out

a user defined path to the folder where the output file(s) will be saved.

threshold

a numerical value between 0 and 1 (default NULL) specifying a threshold for the proportion of matching sequences between haplotypes.
Details

Note: The NestTablesXL() function provides a useful format for further investigation of potentially overlapping haplotypes.


Value

A table specifying the proportions of matching sequences between pairs of haplotypes (in a lower left matrix). If a threshold value is specified, a list of haplotype matches where the proportion of matching sequences exceeds the threshold will be printed to the console. The list will also be saved in the output path, and can be reopened in R e.g. using the readRDS() function in the base package. Note: HpltMatch() will overwrite any existing files with the same output file names in path_out.

See Also

HpltFind; CreateHpltOccTable; NestTablesXL

Examples

hplt_occ_matrix <- hplt_occurrence_matrix
path_out <- tempdir()
HpltMatch(hplt_occ_matrix, path_out, threshold=NULL)

hplt_occurrence_matrix

Data hplt_occurrence_matrix

Description

hplt_occurrence_matrix is an example of a binary occurrence matrix derived from a randomized real major histocompatibility complex (MHC) data set. The matrix was generated using the CreateHpltOccTable() function.

Usage

hplt_occurrence_matrix

Format

hplt_occurrence_matrix is a data frame with 136 putative haplotypes in rows and 329 MHC sequence variants in columns.
**k_summary_table**

**Source**

original data.

---

**k_summary_table**  
**k_summary_table.rda**

**Description**

*k_summary_table* contains the results from a bootstrapped kmeans clustering analysis performed on the test data in the tables z1_matrix_test_data, z2_matrix_test_data, z3_matrix_test_data, z4_matrix_test_data, and z5_matrix_test_data using BootKmeans().

**Usage**

*k_summary_table*

**Format**

*k_summary_table* is a data frame with observations from 10 k-estimation scans in rows and their respective stats in 11 columns.

**Source**

original data.

---

**NestTablesXL**  
**NestTablesXL() function**

**Description**

*NestTablesXL* reads the R lists output by the HpltFind() function and translates them to an Excel workbook for more convenient evaluation of the inferred haplotypes and curation of unresolved and incongruent sequences. The workbook contains separate tabs for each nest in the data set and provides an overview of the genotypes of the samples in each nest and the inferred haplotypes.

**Usage**

*NestTablesXL(nest_table, seq_table, filepath, path_out)*
Arguments

nest_table is a table containing the sample names of parents and offspring in each nest. This table should be organized so that the individual names are in the first column (Sample_ID), and the nest number is in the second column (Nest). For each nest, the first two rows should be the parents, followed immediately by the offspring in the subsequent rows, and then followed by the next nest, and so on. It is assumed that nests are numbered consecutively beginning at 1. Please use the same table as was used to generate the haplotypes using HpltFind().

seq_table seq_table is a sequence table as output by the 'dada2' pipeline, which has samples in rows and nucleotide sequence variants in columns. Please use the same table as was used to generate the haplotypes using HpltFind().

filepath is a user defined path to the folder where the output files from the HpltFind() function have been saved.

path_out is a user defined path to the folder where the output file will be saved.

Details


Value

An Excel workbook with individual tabs for each nest in nest_table. Each tab contains a binary (logical) occurrence matrix with the samples from each nest in columns and sequences (inherited from seq_table) in rows. The order of the samples is derived from nest_table, with parents in the two leftmost columns. Each tab also lists the putative haplotypes inferred by the HpltFind() function and provides lists of unresolved sequences in haplotypes, sequences with unidentified decent (i.e., present in parents but not in offspring), sequences not assigned to haplotypes, and sequences with unidentified origin (i.e., present in offspring but not in parents). Note: NestTablesXL() will overwrite any existing file with the output file name in path_out.

See Also

HpltFind; for more information about 'dada2' visit <https://benjjneb.github.io/dada2/>

Examples

```r
nest_table <- nest_table
seq_table <- sequence_table_hplt
filepath <- system.file("extdata/HpltFindOut/", package="MHCtools")
path_out <- tempdir()
NestTablesXL(nest_table, seq_table, filepath, path_out)
```
**nest_table**

**Data nest_table**

---

**Description**

`nest_table`, `parents_table`, and `sequence_table` comprise a randomized dataset derived from a real major histocompatibility complex (MHC) genotyping experiment with data from parents and offspring. The nucleotide sequences have been replaced with randomly generated sequences, and sample names have been anonymized.

**Usage**

`nest_table`

**Format**

`nest_table` is a data frame with 213 samples in rows and 2 columns:

- **Sample_ID**: Sample ID
- **Nest**: Nest index number

**Source**

original data.

---

**PapaDiv**

**PapaDiv() function**

---

**Description**

`PapaDiv` calculates the joint major histocompatibility complex (MHC) diversity in parent pairs, taking into account alleles that are shared between the parents. The joint diversity in parent pairs is often of interest in studies of mate choice, fitness, and heritability.

**Usage**

`PapaDiv(parents_table, seq_table, path_out)`

**Arguments**

- **parents_table**: is a table containing the sample names of the parents in each nest. This table should be organized so that each row represents one nest, with the individual names of the mothers in the first column (Mother), and the individual names of the fathers in the second column (Father).
- **seq_table**: `seq_table` is a sequence table as output by the 'dada2' pipeline, which has samples in rows and nucleotide sequence variants in columns.
- **path_out**: is a user defined path to the folder where the output files will be saved.
Details

The PapaDiv() function outputs a set of R lists containing for the joint diversity of each parent pair, the proportion of sequences that are shared between the parents, the diversity of each of the parents, the observed sequence variants in each parent, the matched sequence variants, and the incongruent sequence variants in each parent.

In addition, PapaDiv() produces a summary table with the names of the parents in a pair, their respective MHC diversities, and the joint parent pair diversity.


Value

a set of R lists containing for the joint diversity of each parent pair, the proportion of sequences that are shared between the parents, the diversity of each of the parents, the observed sequence variants in each parent, the matched sequence variants, and the incongruent sequence variants in each parent. The sequences are named in the output by an index number corresponding to their column number in the sequence table, thus identical sequences will have identical sample names in all the output files. These files are saved in a sub folder in the output path called Parent_pairs (created by PapaDiv()) and can be reopened in R e.g. using the readRDS() function in the base package. For downstream data analysis, the PapaDiv() function also produces a summary table with the names of the parents in a pair, their respective MHC diversities, and the joint parent pair diversity. This table is saved as a .csv file in the output path.

See Also

For more information about 'dada2' visit <https://benjjneb.github.io/dada2/>

Examples

```r
parents_table <- parents_table
seq_table <- sequence_table
path_out <- tempdir()
PapaDiv(parents_table, seq_table, path_out)
```

Description

`nest_table, parents_table, and sequence_table` comprise a randomized dataset derived from a real major histocompatibility complex (MHC) genotyping experiment with data from parents and offspring. The nucleotide sequences have been replaced with randomly generated sequences, and sample names have been anonymized.
parents_table

Format
parents_table is a data frame with 57 parent pairs in rows and 2 columns:

- **Mother**  Mother ID
- **Father**  Father ID

Source
original data.

replicates_table

Usage
replicates_table

Format
replicates_table is a data frame with 111 technical replicate samples in rows and 2 columns:

- **Sample_ID**  Technical replicate sample ID
- **Replic_set**  Index number of replicate set

Source
original data.
ReplMatch

Description

In amplicon filtering it is sometimes valuable to compare technical replicates in order to estimate the accuracy of a genotyping experiment. This may be done both to optimize filtering settings and to estimate repeatability to report in a publication. ReplMatch is designed to automatically compare technical replicates in an amplicon filtering data set and report the proportion of mismatches. The functions GetReplTable() and GetReplStats() are designed to evaluate the output files.

Usage

ReplMatch(repl_table, seq_table, path_out)

Arguments

- **repl_table**: is a table containing the sample names of technical replicates in the data set. This table should be organized so that the individual names are in the first column (Sample_ID), and the index number of the replicate set is in the second column (Replic_set). Replicate sets may contain more than two replicates, but sets must be numbered consecutively beginning at 1.
- **seq_table**: seq_table is a sequence table as output by the ‘dada2’ pipeline, which has samples in rows and nucleotide sequence variants in columns.
- **path_out**: is a user defined path to the folder where the output files will be saved.

Details

Note: ReplMatch() will throw a warning if all samples in a replicate set have 0 sequences. In that case, the mean_props for that replicate set and the repeatability for the data set will be NaN, and ReplMatch() will report which replicate set is problematic and suggest to remove it from the repl_table. If removing replicate sets, beware that the replicate sets in repl_table must be numbered consecutively beginning at 1.


Value

A set of R lists containing for each replicate set the observed sequence variants, the names of the sequences that were incongruent in the replicates, and the mean proportion of incongruent sequences (if 100 matches are expected between the replicates, this is equivalent of an error rate in the sequencing process). The sequences are named in the output by an index number corresponding to their
column number in the sequence table, thus identical sequences will have identical sample names in all the output files. These files can be reopened in R e.g. using the readRDS() function in the base package.

See Also

GetReplTable; GetReplStats; for more information about 'dada2' visit <https://benjjneb.github.io/dada2/>

Examples

repl_table <- replicates_table
data_table <- sequence_table_repl
path_out <- tempdir()
ReplMatch(repl_table, data_table, path_out)

sequence_table

Data sequence_table

Description

nest_table, parents_table, and sequence_table comprise a randomized dataset derived from a real major histocompatibility complex (MHC) genotyping experiment with data from parents and offspring. The nucleotide sequences have been replaced with randomly generated sequences, and sample names have been anonymized.

Usage

sequence_table

Format

sequence_table is a data frame with 334 samples in rows and 329 DNA sequence variants in columns.

Source

original data.
sequence_table_fas

Data sequence_table_fas

Description
sequence_table_fas is a randomized dataset derived from a real major histocompatibility complex (MHC) genotyping experiment. The nucleotide sequences have been replaced with randomly generated sequences, and sample names have been anonymized.

Usage
sequence_table_fas

Format
sequence_table_fas is a data frame with 100 samples in rows and 166 DNA sequence variants in columns.

Source
original data.

sequence_table_hplt

Data sequence_table_hplt

Description
sequence_table_hplt is a randomized test data set derived from a real major histocompatibility complex (MHC) genotyping experiment. This table differs from the sequence_table by having the nucleotide sequences replaced with sequence names. Sample names have been anonymized from the real data set.

Usage
sequence_table_hplt

Format
sequence_table_hplt is a data frame with 334 samples in rows and 329 sequence variants in columns.

Source
original data.
sequence_table_repl

Data sequence_table_repl

Description

replicates_table and sequence_table_repl comprise a randomized dataset derived from a real major histocompatibility complex (MHC) genotyping experiment with technical replicates. The nucleotide sequences have been replaced with randomly generated sequences, and sample names have been anonymized.

Usage

sequence_table_repl

Format

sequence_table_repl is a data frame with 412 samples in rows and 511 DNA sequence variants in columns.

Source

original data.

z1_matrix

z1_matrix.rda

Description

z1_matrix comprise a randomized dataset derived from 70 nucleotide sequences of a real major histocompatibility complex (MHC) genotyping experiment. z1-descriptor values have been extracted from a subset of 8 amino acid codons and sequence names have been anonymized.

Usage

z1_matrix

Format

z1_matrix is a data frame with 70 sequences in rows and z1-descriptor variables for 8 sequence codons in columns.

Source

original data.
z2_matrix

**Description**

z2_matrix comprise a randomized dataset derived from 70 nucleotide sequences of a real major histocompatibility complex (MHC) genotyping experiment. z2-descriptor values have been extracted from a subset of 8 amino acid codons and sequence names have been anonymized.

**Usage**

z2_matrix

**Format**

z2_matrix is a data frame with 70 sequences in rows and z2-descriptor variables for 8 sequence codons in columns.

**Source**

original data.

---

z3_matrix

**Description**

z3_matrix comprise a randomized dataset derived from 70 nucleotide sequences of a real major histocompatibility complex (MHC) genotyping experiment. z3-descriptor values have been extracted from a subset of 8 amino acid codons and sequence names have been anonymized.

**Usage**

z3_matrix

**Format**

z3_matrix is a data frame with 70 sequences in rows and z3-descriptor variables for 8 sequence codons in columns.

**Source**

original data.
**z4_matrix**

Description

*z4_matrix* comprise a randomized dataset derived from 70 nucleotide sequences of a real major histocompatibility complex (MHC) genotyping experiment. *z4*-descriptor values have been extracted from a subset of 8 amino acid codons and sequence names have been anonymized.

Usage

```r
z4_matrix
```

Format

*z4_matrix* is a data frame with 70 sequences in rows and *z4*-descriptor variables for 8 sequence codons in columns.

Source

original data.

---

**z5_matrix**

Description

*z5_matrix* comprise a randomized dataset derived from 70 nucleotide sequences of a real major histocompatibility complex (MHC) genotyping experiment. *z5*-descriptor values have been extracted from a subset of 8 amino acid codons and sequence names have been anonymized.

Usage

```r
z5_matrix
```

Format

*z5_matrix* is a data frame with 70 sequences in rows and *z5*-descriptor variables for 8 sequence codons in columns.

Source

original data.
Index

* datasets
  hplt_occurrence_matrix, 18
  k_summary_table, 19
  nest_table, 21
  parents_table, 22
  replicates_table, 23
  sequence_table, 25
  sequence_table_fas, 26
  sequence_table_hplt, 26
  sequence_table_repl, 27
  z1_matrix, 27
  z2_matrix, 28
  z3_matrix, 28
  z4_matrix, 29
  z5_matrix, 29

BootKmeans, 3, 3, 7
ClusterMatch, 5, 5
CreateFas, 7, 7, 9
CreateHpltOccTable, 8, 8, 17, 18
CreateSamplesFas, 7, 9, 9

DistCalc, 5, 10, 10

GetHpltStats, 12, 12, 13, 17
GetHpltTable, 13, 13, 17
GetReplStats, 14, 14, 15, 25
GetReplTable, 14, 15, 15, 25

hplt_occurrence_matrix, 18
HpltFind, 8, 13, 16, 16, 18, 20
HpltMatch, 17, 17

k_summary_table, 19

nest_table, 21
NestTablesXL, 17–19, 19

PapaDiv, 21, 21
parents_table, 22

replicates_table, 23
ReplMatch, 14, 15, 24, 24

sequence_table, 25
sequence_table_fas, 26
sequence_table_hplt, 26
sequence_table_repl, 27

z1_matrix, 27
z2_matrix, 28
z3_matrix, 28
z4_matrix, 29
z5_matrix, 29