Package: SYNCSA (via r-universe)

September 6, 2024

Type Package

Title Analysis of Functional and Phylogenetic Patterns in Metacommunities

Version 1.3.5

Date 2020-09-01

Author Vanderlei Julio Debastiani

Maintainer Vanderlei Julio Debastiani
<vanderleidebastiani@yahoo.com.br>

Imports vegan, FD, permute, RcppArmadillo, utils, stats, graphics, parallel

Description Analysis of metacommunities based on functional traits and phylogeny of the community components. The functions that are offered here implement for the R environment methods that have been available in the SYNCSA application written in C++ (by Valerio Pillar, available at <http://ecoqua.ecologia.ufrgs.br/SYNCSA.html>).

License GPL-2

Encoding UTF-8

RoxygenNote 7.1.1

Collate 'CollectNames.R' 'ProgressBAR.R' 'belonging.R' 'cent.norm.R' 'coinertia.partial.syncsa.R' 'coinertia.syncsa.R' 'cor.matrix.R' 'cor.matrix.bf.R' 'cor.matrix.partial.R' 'cor.matrix.rao.R' 'matmult.syncsa.R' 'matrix.p.R' 'matrix.t.R' 'matrix.w.transformation.R' 'matrix.x.R' 'optimal.R' 'organize.syncsa.R' 'part.cor.R' 'pca.R' 'permut.row.matrix.R' 'permut.vector.R' 'plot.pcasyncsa.R' 'print.optimal.R' 'syncsa.R' 'print.syncsa.R' 'pro.matrix.R' 'pro.matrix.bf.R' 'pro.matrix.partial.R' 'pro.matrix.rao.R' 'procrustes.syncsa.R' 'procrustes.partial.syncsa.R' 'rao.diversity.R' 'rv.matrix.R' 'rv.matrix.bf.R' 'rv.matrix.partial.R' 'rv.matrix.rao.R' 'startup.R' 'syncsa.obs.R' 'var.dummy.R' 'var.type.R'

Repository https://vanderleidebastiani.r-universe.dev

RemoteUrl https://github.com/vanderleidebastiani/syncsa

RemoteRef HEAD

RemoteSha 83622ced401714559059aa17895b224d6fb90510

Contents

`ADRS` *Artificial Data for Run SYNCSA*

Description

Artificial data for run SYNCSA.

Usage

```
data(ADRS)
```

Format

A list with:.

commnunity Community data, observations of six species in ten sites.

traits Matrix data of species described by two traits.

phylo Matrix containing phylogenetic distance between species.

envir One environmental variable for each community.

Examples

```
data(ADRS)
```

`belonging` *Degree of belonging of species*

Description

Function to obtain a matrix containing the degrees of belongings of each and every species. The degree of belonging of each species is defined based on its ecological or phylogenetic resemblance to every other species in the community. For more details, see `matrix.p`, `matrix.x` and `syncsa`.

Usage

```
belonging(dis, standardize = TRUE)
```

Arguments

`dis`	Matrix containing distance between species.
`standardize`	Logical argument (TRUE or FALSE) to specify if dis must be standardized in values ranging from 0 to 1 (Default standardize = TRUE).

Value

Standardized matrix containing the degree of belonging of species in relation to each other. Row totals (species) = 1.

Author(s)

Vanderlei Julio Debastiani <vanderleidebastiani@yahoo.com.br>

References

Pillar, V.D.; Duarte, L.d.S. (2010). A framework for metacommunity analysis of phylogenetic structure. Ecology Letters, 13, 587-596.

See Also

`matrix.p`, `matrix.x`, `syncsa`

Examples

```
data(ADRS)
belonging(ADRS$phylo)
```

`cent.norm` *Matrix centralization and standardization*

Description

Function to perform centralization and standardization in a matrix.

Usage

```
cent.norm(x, na.rm = FALSE)
```

Arguments

`x`	A matrix
`na.rm`	Logical argument (TRUE or FALSE) to specify if missing observations are removed (Default na.rm = FALSE).

Author(s)

Vanderlei Julio Debastiani <vanderleidebastiani@yahoo.com.br>

See Also

`syncsa`

`coinertia.partial.syncsa` *Co-inertia and Partial Co-inertia correlations.*

Description

Function to obtain the RV coefficient between two matrices and partial RV coefficient between three matrices.

Usage

```
coinertia.partial.syncsa(x, y, z, scale = FALSE)

coinertia.syncsa(x, y, scale = FALSE)
```

Arguments

`x, y`	Matrix that will be correlated.
`z`	Matrix whose effect will be removed from the correlation between x and y.
`scale`	Standardized variables to unit variance.

Value

RV coefficient (generalization of the Pearson correlation coefficient) between matrices.

Author(s)

Vanderlei Julio Debastiani <vanderleidebastiani@yahoo.com.br>

References

Legendre, P. and Legendre L. (2012). Numerical Ecology. 3nd English edition. Elsevier.

See Also

`syncsa`

`CollectNames` *Collect names an entire list*

Description

Function to collect names an entire list.

Usage

```
CollectNames(l, prefix = NULL)
```

Arguments

`l`	A list.
`prefix`	A prefix to nomes.

Value

The names.

cor.matrix *Function to obtain the correlation between two matrices and partial matrix correlation between three matrices.*

Description

The functions cor.matrix and cor.matrix.partial are similar the function mantel and mantel.partial, although the significance of the statistics is evaluated differently from Mantel. The functions pro.matrix and pro.matrix.partial use symmetric Procrustes as a measure of concordance between data sets. The function cor.mantel is similar to the function mantel, but allows the use of a set of predefined permutation. For more details, see syncsa.

Usage

```
cor.matrix(
  mx1,
  mx2,
  x,
  my1 = NULL,
  my2 = NULL,
  y,
  permute.my2 = FALSE,
  method = "pearson",
  dist = "euclidean",
  permutations = 999,
  norm = FALSE,
  norm.y = FALSE,
  strata = NULL,
  na.rm = FALSE,
  seqpermutation = NULL,
  parallel = NULL,
  newClusters = TRUE,
  CL = NULL
)

cor.matrix.bf(
  dist.x,
  dist.y,
  method = "pearson",
  permutations = 999,
  strata = NULL,
  na.rm = FALSE,
  seqpermutation = NULL,
  parallel = NULL,
  newClusters = TRUE,
  CL = NULL
)
```

```
cor.matrix.partial(
  mx1,
  mx2,
  x,
  my1 = NULL,
  my2 = NULL,
  y,
  mz1 = NULL,
  mz2 = NULL,
  z,
  method = "pearson",
  dist = "euclidean",
  permute.my2 = FALSE,
  permute.mz2 = FALSE,
  permutations = 999,
  norm = FALSE,
  norm.y = FALSE,
  norm.z = FALSE,
  strata = NULL,
  na.rm = FALSE,
  seqpermutation = NULL,
  parallel = NULL,
  newClusters = TRUE,
  CL = NULL
)

cor.matrix.rao(
  mx1,
  mx2,
  x,
  y,
  method = "pearson",
  dist = "euclidean",
  put.together = NULL,
  permutations = 999,
  strata = NULL,
  na.rm = FALSE,
  seqpermutation = NULL,
  parallel = NULL,
  newClusters = TRUE,
  CL = NULL
)

pro.matrix(
  mx1,
  mx2,
  x,
  my1 = NULL,
  my2 = NULL,
  y,
  permute.my2 = FALSE,
  permutations = 999,
  norm = FALSE,
  norm.y = FALSE,
  strata = NULL,
  seqpermutation = NULL,
  parallel = NULL,
  newClusters = TRUE,
  CL = NULL
)

pro.matrix.bf(
  x,
  y,
  permutations = 999,
  strata = NULL,
  na.rm = FALSE,
  seqpermutation = NULL,
  parallel = NULL,
  newClusters = TRUE,
  CL = NULL
)

pro.matrix.partial(
  mx1,
  mx2,
  x,
  my1 = NULL,
  my2 = NULL,
  y,
  mz1 = NULL,
  mz2 = NULL,
  z,
  permute.my2 = FALSE,
  permute.mz2 = FALSE,
  permutations = 999,
  norm = FALSE,
  norm.y = FALSE,
  norm.z = FALSE,
  strata = NULL,
  seqpermutation = NULL,
  parallel = NULL,
  newClusters = TRUE,
  CL = NULL
)
```

```
pro.matrix.rao(
  mx1,
  mx2,
  x,
  y,
  put.together = NULL,
  permutations = 999,
  strata = NULL,
  seqpermutation = NULL,
  parallel = NULL,
  newClusters = TRUE,
  CL = NULL
)

rv.matrix(
  mx1,
  mx2,
  x,
  my1 = NULL,
  my2 = NULL,
  y,
  permute.my2 = FALSE,
  permutations = 999,
  norm = FALSE,
  norm.y = FALSE,
  strata = NULL,
  seqpermutation = NULL,
  parallel = NULL,
  newClusters = TRUE,
  CL = NULL
)

rv.matrix.bf(
  x,
  y,
  permutations = 999,
  strata = NULL,
  na.rm = FALSE,
  seqpermutation = NULL,
  parallel = NULL,
  newClusters = TRUE,
  CL = NULL
)

rv.matrix.partial(
  mx1,
  mx2,
  x,
  my1 = NULL,
  my2 = NULL,
  y,
  mz1 = NULL,
  mz2 = NULL,
  z,
  permute.my2 = FALSE,
  permute.mz2 = FALSE,
  permutations = 999,
  norm = FALSE,
  norm.y = FALSE,
  norm.z = FALSE,
  strata = NULL,
  seqpermutation = NULL,
  parallel = NULL,
  newClusters = TRUE,
  CL = NULL
)

rv.matrix.rao(
  mx1,
  mx2,
  x,
  y,
  put.together = NULL,
  permutations = 999,
  strata = NULL,
  seqpermutation = NULL,
  parallel = NULL,
  newClusters = TRUE,
  CL = NULL
)
```

Arguments

`mx1`	Matrix that multiplied by mx2 results in the matrix x.
`mx2`	Matrix that when multiplied by mx1 results in the matrix x. See ‘details‘ below.
`x`	Matrix that will be correlated with the matrix y.
`my1`	Matrix that multiplied by my2 results in the matrix y.
`my2`	Matrix that when multiplied by my1 results in the matrix y. See ‘details‘ below.
`y`	Matrix that will be correlated with the matrix x.
`permute.my2`	Logical argument (TRUE or FALSE) to specify if realize parallel permutation in matrix my2.
`method`	Correlation method, as accepted by cor: "pearson", "spearman" or "kendall".

`dist`	Dissimilarity index, as accepted by vegdist: "manhattan", "euclidean", "canberra", "bray", "kulczynski", "jaccard", "gower", "altGower", "morisita", "horn", "mountford", "raup" , "binomial" or "chao".
`permutations`	Number of permutations in assessing significance.
`norm`	Logical argument (TRUE or FALSE) to specify if x is standardized within variables (Default norm = FALSE).
`norm.y`	Logical argument (TRUE or FALSE) to specify if y is standardized within variables (Default norm = FALSE).
`strata`	Argument to specify restricting permutations within species groups (Default strata = NULL).
`na.rm`	Logical argument (TRUE or FALSE) to specify if pairwise deletion of missing observations when computing dissimilarities (Default na.rm = FALSE).
`seqpermutation`	A set of predefined permutation, with the same dimensions of permutations (Default seqpermutation = NULL).
`parallel`	Number of parallel processes. Tip: use parallel::detectCores() (Default parallel = NULL).
`newClusters`	Logical argument (TRUE or FALSE) to specify if make new parallel processes or use predefined socket cluster. Only if parallel is different of NULL (Default newClusters = TRUE).
`CL`	A predefined socket cluster done with parallel package.
`dist.x`	Dissimilarity matrices of class dist.
`dist.y`	Dissimilarity matrices of class dist.
`mz1`	Matrix that multiplied by mz2 results in the matrix z.
`mz2`	Matrix that when multiplied by mz1 results in the matrix z. See ‘details‘ below.
`z`	Matrix whose effect will be removed from the correlation between x and y.
`permute.mz2`	Logical argument (TRUE or FALSE) to specify if realize parallel permutation in matrix mz2.
`norm.z`	Logical argument (TRUE or FALSE) to specify if z is standardized within variables (Default norm = FALSE).
`put.together`	List to specify group of traits. Each group specify receive the same weight that one trait outside any group, in the way each group is considered as unique trait (Default put.together = NULL). This argument must be a list, see examples in `syncsa`.

Details

The null model is based on permutations in the matrix mx2, typically the matrices B, U and Q, except in the function cor.mantel when the permutations are done in one of distance matrix.

Null model described by Pillar et al. (2009) and Pillar & Duarte (2010). For more details on the matrices and the null model, see `syncsa`.

Value

`Obs`	Correlation between matrices.
`p`	Significance level based on permutations.

Author(s)

Vanderlei Julio Debastiani <vanderleidebastiani@yahoo.com.br>

References

Pillar, V.D.; Duarte, L.d.S. (2010). A framework for metacommunity analysis of phylogenetic structure. Ecology Letters, 13, 587-596.

Pillar, V.D., Duarte, L.d.S., Sosinski, E.E. & Joner, F. (2009). Discriminating trait-convergence and trait-divergence assembly patterns in ecological community gradients. Journal of Vegetation Science, 20, 334:348.

See Also

`syncsa`, `organize.syncsa`, `mantel`, `procrustes`

`flona` *Hypothetical data for SYNCSA*

Description

Hypothetical data for running examples.

Usage

```
data(flona)
```

Format

A list with 4 matrices.

commnunity Community data, observations of 59 species in 39 sites.

traits Matrix data of species described by five traits.

phylo Matrix containing phylogenetic distance between species.

environment Three environmental variables for each community.

Examples

```
data(flona)
```

`matmult.syncsa` *Matrix multiplication with missing data*

Description

Function to get the matrix product when missing data (NA) are found in matrix Y. The matrix X must be standardized. See details.

Usage

```
matmult.syncsa(X, Y)
```

Arguments

`X`	A matrix, typically the standardized community matrix (W).
`Y`	A matrix.

Details

The function ignore missing data when found in matrix Y. Before multiplication of matrices the missing data in Y are replaced by 0 and multiplication is performed, and then, an adjustment is performed. This adjustment is done by divide each cell of the product matrix by the sum of proportions of nonzero at X with complete data in Y. In SYNCSA context this adjustment is done by divide each cell of the product matrix by the sum of species proportion with trait data in Y. The matrix X must be standardized, in other words, row totals must be equal to 1.

Value

The matrix product.

Author(s)

Vanderlei Julio Debastiani <vanderleidebastiani@yahoo.com.br>

See Also

`syncsa`, `matmult`,

matrix.p *Matrix P*

Description

Function to obtain a matrix containing phylogeny-weighted species composition. For more details, see `syncsa`.

Usage

```
matrix.p(
  comm,
  phylodist,
  transformation = "standardized",
  spp.weights = NULL,
  notification = TRUE
)
```

Arguments

`comm`	Community data, with species as columns and sampling units as rows. This matrix can contain either presence/absence or abundance data.
`phylodist`	Matrix containing phylogenetic distance between species. Must be a complete matrix (not a diagonal resemblance matrix).
`transformation`	Method to community data transformation, "none", "standardized" or "weights" (Default transformation = "standardized").
`spp.weights`	Vector with 0 or 1 to specify individual species weights (Default spp.weights = NULL).
`notification`	Logical argument (TRUE or FALSE) to specify if notifications for missing observations are to be shown (Default notification = TRUE).

Value

`matrix.w`	Standardized community matrix, where rows are communities and columns species. If default transformation, row totals (communities) = 1.
`matrix.q`	Standardized matrix containing the degree of belonging of species in relation to each other. Row totals (species) = 1.
`matrix.P`	Phylogeny-weighted species composition matrix. If default transformation, row totals (communities) = 1.

Note

IMPORTANT: Species sequence in the community data matrix MUST be the same as the one in the phylogenetic distance matrix or in the spp.weights vector. See `organize.syncsa`.

Author(s)

Vanderlei Julio Debastiani <vanderleidebastiani@yahoo.com.br>

References

Pillar, V.D.; Duarte, L.d.S. (2010). A framework for metacommunity analysis of phylogenetic structure. Ecology Letters, 13, 587-596.

See Also

`syncsa`, `organize.syncsa`, `belonging`, `matrix.t`, `matrix.x`

Examples

```
data(ADRS)
matrix.p(ADRS$community, ADRS$phylo)
```

`matrix.t` *Matrix T*

Description

Function to obtain a matrix containing trait averages at community level. For more details, see `syncsa`.

Usage

```
matrix.t(
  comm,
  traits,
  scale = TRUE,
  ranks = TRUE,
  transformation = "standardized",
  spp.weights = NULL,
  notification = TRUE
)
```

Arguments

`comm`	Community data, with species as columns and sampling units as rows. This matrix can contain either presence/absence or abundance data.
`traits`	Matrix or data frame of species described by traits, with traits as columns and species as rows.
`scale`	Logical argument (TRUE or FALSE) to specify if the traits are measured on different scales (Default scale = TRUE). When scale = TRUE traits are measured on different scales and the matrix T is subjected to standardization within each trait. When scale = FALSE traits are measured on the same scale the matrix T is not subjected to standardization.

`ranks`	Logical argument (TRUE or FALSE) to specify if ordinal variables are convert to ranks (Default ranks = TRUE).
`transformation`	Method to community data transformation, "none", "standardized" or "weights" (Default transformation = "standardized").
`spp.weights`	Vector with 0 or 1 to specify individual species weights (Default spp.weights = NULL).
`notification`	Logical argument (TRUE or FALSE) to specify if notifications of missing observations are shown (Default notification = TRUE).

Value

`matriz.w`	Standardized community matrix, where rows are communities and columns species. If default transformation, row totals (communities) = 1.
`matriz.b`	Matrix of traits, exactly the same data input.
`matriz.T`	Matrix containing trait averages at community level. If Scale = TRUE the matrix T is standardized within the traits.

Note

IMPORTANT: The sequence species show up in community data matrix MUST be the same as they show up in traits matrix or in the spp.weights vector. See `organize.syncsa`.

Author(s)

Vanderlei Julio Debastiani <vanderleidebastiani@yahoo.com.br>

References

Pillar, V.D.; Duarte, L.d.S. (2010). A framework for metacommunity analysis of phylogenetic structure. Ecology Letters, 13, 587-596.

Pillar, V.D., Duarte, L.d.S., Sosinski, E.E. & Joner, F. (2009). Discriminating trait-convergence and trait-divergence assembly patterns in ecological community gradients. Journal of Vegetation Science, 20, 334:348.

See Also

`syncsa`, `organize.syncsa`, `matrix.p`, `matrix.x`

Examples

```
data(ADRS)
matrix.t(ADRS$community, ADRS$traits)
```

`matrix.w.transformation` *Community data transformation*

Description

Function to transformation community data and replace missing data. See details.

Usage

```
matrix.w.transformation(
  comm,
  transformation = "standardized",
  spp.weights = NULL,
  reference = NULL,
  type = 0,
  include = TRUE,
  notification = TRUE
)
```

Arguments

`comm`	Community data, with species as columns and sampling units as rows. This matrix can contain either presence/absence or abundance data.
`transformation`	Method to community data transformation, "none", "standardized", "weights" or "beals" (Default transformation = "standardized").
`spp.weights`	Vector with 0 or 1 to specify individual species weights (Default spp.weights = NULL).
`reference`	Data to compute joint occurrences in `beals` function. If NULL, comm is used as reference to compute the joint occurrences (Default reference = NULL).
`type`	Method to specifies how abundance values are used in `beals` function, type = 0 presence/absence, type = 1 abundances are used to compute conditioned probabilities, type = 2 abundances are used to compute weighted averages of conditioned probabilities or type = 3 abundances are used to compute both conditioned probabilities and weighted averages (Default type = 0).
`include`	Logical argument (TRUE or FALSE) to specify if target species are included when computing the mean of the conditioned probabilities in `beals` (Default include = TRUE).
`notification`	Logical argument (TRUE or FALSE) to specify if notifications for missing observations are to be shown (Default notification = TRUE).

Details

The function applies standardization/transformation for community data. The options are: "none" no transformation is applied; "standardized" the community data is standardized to row totals will be 1; "weights" community data is first "standardized" and when, individual species weights are multiplied to each species entries; and "beals" option applies Beals smoothing using the function `beals`. The arguments "reference", "type" and "include" are the same used in `beals` function. Missing data are replaced by 0 after the standardization/transformation.

Value

Transformed community matrix, where rows are communities and columns species.

Note

IMPORTANT: Species sequence in the community data matrix MUST be the same as the one in the reference matrix or in spp.weights vector. See `organize.syncsa`.

Author(s)

Vanderlei Julio Debastiani <vanderleidebastiani@yahoo.com.br>

References

Pillar, V.D.; Duarte, L.d.S. (2010). A framework for metacommunity analysis of phylogenetic structure. Ecology Letters, 13, 587-596. De Cáceres, M.; Legendre, P. 2008. Beals smoothing revisited. Oecologia 156: 657–669.

See Also

`syncsa`, `organize.syncsa`, `matrix.p`, `matrix.t`, `matrix.x`

Examples

```
data(ADRS)
matrix.w.transformation(ADRS$community)
```

`matrix.x` *Matrix X*

Description

Function to obtain a matrix containing trait-weighted species composition. For more details, see `syncsa`.

Usage

```
matrix.x(
  comm,
  traits,
  scale = TRUE,
  ranks = TRUE,
  transformation = "standardized",
  spp.weights = NULL,
  notification = TRUE,
  ...
)
```

Arguments

`comm`	Community data, with species as columns and sampling units as rows. This matrix can contain either presence/absence or abundance data.
`traits`	Matrix or data frame of species described by traits, with traits as columns and species as rows.
`scale`	Logical argument (TRUE or FALSE) to specify if the traits are measured on different scales (Default scale = TRUE). When scale = TRUE traits are measured on different scales the matrix of traits is subjected to standardization within each trait, and Gower Index is used to calculate the degree of belonging to the species. When scale = FALSE traits are measured on the same scale the matrix of traits is not subjected to standardization, and Euclidean distance is calculated to determine the degree of belonging to the species.
`ranks`	Logical argument (TRUE or FALSE) to specify if ordinal variables are convert to ranks (Default ranks = TRUE).
`transformation`	Method to community data transformation, "none", "standardized" or "weights" (Default transformation = "standardized").
`spp.weights`	Vector with 0 or 1 to specify individual species weights (Default spp.weights = NULL).
`notification`	Logical argument (TRUE or FALSE) to specify if notifications of missing observations are shown (Default notification = TRUE).
`...`	Parameters for `gowdis` function.

Value

`matriz.w`	Standardized community matrix, where rows are communities and columns species. If default transformation, row totals (communities) = 1.
`matriz.u`	Standardized matrix containing the degree of belonging of each species in relation to each other species. Row totals (species) = 1.
`matriz.X`	Trait-weighted species composition matrix. If default transformation, row totals (communities) = 1.

Note

IMPORTANT: The sequence species show up in community data matrix MUST be the same as they show up in traits matrix or in the spp.weights vector. See `organize.syncsa`.

Author(s)

Vanderlei Julio Debastiani <vanderleidebastiani@yahoo.com.br>

References

Pillar, V.D.; Duarte, L.d.S. (2010). A framework for metacommunity analysis of phylogenetic structure. Ecology Letters, 13, 587-596.

Pillar, V.D., Duarte, L.d.S., Sosinski, E.E. & Joner, F. (2009). Discriminating trait-convergence and trait-divergence assembly patterns in ecological community gradients. Journal of Vegetation Science, 20, 334:348.

See Also

`syncsa`, `organize.syncsa`, `belonging`, `matrix.t`, `matrix.p`, `gowdis`

Examples

```
data(ADRS)
matrix.x(ADRS$community, ADRS$traits)
```

`optimal` *Searching for optimal traits*

Description

Maximize trait-convergence assembly patterns (TCAP = roTE), trait-divergence assembly patterns (TDAP = roXE.T), maximize both trait-divergence assembly patterns and trait-convergence assembly patterns (TCAP.TDAP = roXE) or alpha divergence (roRE) For more details, see `syncsa`.

Usage

```
optimal(
  comm,
  traits = NULL,
  envir = NULL,
  checkdata = TRUE,
  subset.min = 1,
  subset.max = ncol(traits),
  pattern = NULL,
  ro.method = "mantel",
  dist = "euclidean",
  method = "pearson",
  scale = TRUE,
  scale.envir = TRUE,
  ranks = TRUE,
  asym.bin = NULL,
  ord = "metric",
  put.together = NULL,
  na.rm = FALSE,
  notification = TRUE,
  progressbar = FALSE
)

## S3 method for class 'optimal'
print(x, ...)
```

Arguments

`comm`	Community data, with species as columns and sampling units as rows. This matrix can contain either presence/absence or abundance data. Alternatively comm can be an object of class metacommunity.data, an alternative way to set all data.frames/matrices. When you use the class metacommunity.data the arguments traits, envir and put.together must be null. See details.
`traits`	Matrix data of species described by traits, with traits as columns and species as rows (Default traits = NULL).
`envir`	Environmental variables for each community, with variables as columns and sampling units as rows (Default envir = NULL).
`checkdata`	Logical argument (TRUE or FALSE) to check if species sequence in the community data follows the same order as the one in the trait and if sampling units in the community data follows the same order as the one in the environmental matrices (Default checkdata = TRUE).
`subset.min`	Minimum of traits in each subset (Default subset.min = 1).
`subset.max`	Maximum of traits in each subset (Default subset.max = ncol(traits)).
`pattern`	Patterns for maximize correlation, "tcap", "tdap", "tcap.tdap" or "rao" (Default pattern = NULL).
`ro.method`	Method to obtain the correlation, "mantel" or "procrustes" (Default ro.method = "mantel").
`dist`	Dissimilarity index, as accepted by vegdist: "manhattan", "euclidean", "canberra", "bray", "kulczynski", "jaccard", "gower", "altGower", "morisita", "horn", "mountford", "raup" , "binomial" or "chao".
`method`	Correlation method, as accepted by cor: "pearson", "spearman" or "kendall".
`scale`	Logical argument (TRUE or FALSE) to specify if the traits are measured on different scales (Default Scale = TRUE). When scale = TRUE traits are measured on different scales and the matrix T is subjected to standardization within each trait. When scale = FALSE if traits are measured on the same scale and the matrix T is not subjected to standardization. Furthermore, if scale = TRUE the matrix of traits is subjected to standardization within each trait, and Gower Index is used to calculate the degree of belonging to the species, and if scale

	= FALSE the matrix of traits is not subjected to standardization, and Euclidean distance is calculated to determine the degree of belonging to the species.
`scale.envir`	Logical argument (TRUE or FALSE) to specify if the environmental variables are measured on different scales (Default scale = TRUE). If the enviromental variables are measured on different scales, the matrix is subjected to centralization and standardization within each variable.
`ranks`	Logical argument (TRUE or FALSE) to specify if ordinal variables are convert to ranks (Default ranks = TRUE).
`asym.bin`	Vector listing the asymmetric binary traits, see `gowdis` (Default asym.bin = NULL).
`ord`	Method to be used for ordinal traits, see `gowdis` (Default ord = "metric").
`put.together`	List to specify group traits that are added or removed together (Default put.together = NULL). This argument must be a list, see examples.
`na.rm`	Logical argument (TRUE or FALSE) to specify if pairwise deletion of missing observations when computing dissimilarities (Default na.rm = FALSE).
`notification`	Logical argument (TRUE or FALSE) to specify if notifications of missing observations are shown (Default notification = TRUE).
`progressbar`	Logical argument (TRUE or FALSE) to specify if display a progress bar on the R console (Default progressbar = FALSE).
`x`	An object of class optimal.
`...`	Other parameters for the respective functions.

Details

Package **SYNCSA** requires that the species and community sequence in the data.frame or matrix must be the same for all dataframe/matrices. The function `organize.syncsa` organizes the data for the functions of the package, placing the matrices of community, traits and environmental varibles in the same order. The function use of function organize.syncsa is not requered for run the functions, but is recommended. In this way the arguments comm, traits, envir, as well as the argument put.together, can be specified them as normal arguments or by passing them with the object returned by the function `organize.syncsa` using, in this case only the argument comm. Using the object returned by organize.syncsa, the comm argument is used as an alternative way of entering to set all data.frames/matrices, and therefore the other arguments (traits, envir, and put.together) must be null.

Value

`Subset`	Subset of traits that maximizes the correlation.
`ro`	Correlation for the subset of traits.

Note

IMPORTANT: The sequence species show up in community data matrix MUST be the same as they show up in traits matrix. See details and `organize.syncsa`.

Author(s)

Vanderlei Julio Debastiani <vanderleidebastiani@yahoo.com.br>

References

Pillar, V.D.; Duarte, L.d.S. (2010). A framework for metacommunity analysis of phylogenetic structure. Ecology Letters, 13, 587-596.

Pillar, V.D., Duarte, L.d.S., Sosinski, E.E. & Joner, F. (2009). Discriminating trait-convergence and trait-divergence assembly patterns in ecological community gradients. Journal of Vegetation Science, 20, 334:348.

See Also

`syncsa`, `organize.syncsa`

Examples

```
data(flona)
optimal(flona$community, flona$traits, flona$environment, subset.min = 3,
   subset.max = 5, pattern = "tcap")
optimal(flona$community, flona$traits, flona$environment, subset.min = 3,
   subset.max = 5, pattern = "tdap")
optimal(flona$community, flona$traits, flona$environment, subset.min = 3,
   subset.max = 5, pattern = "tcap.tdap")
put.together <- list(c("fol", "sem"), c("tam", "red"))
put.together
optimal(flona$community, flona$traits, flona$environment, subset.min = 1,
   subset.max = 3, pattern = "tcap", put.together = put.together)
```

`organize.syncsa` *Function for organize data for Package SYNCSA*

Description

Package **SYNCSA** requires that the species and community sequence in the data.frame or matrix must be the same for all data.frame/matrices.

Usage

```
organize.syncsa(
  comm,
  traits = NULL,
  phylodist = NULL,
  envir = NULL,
  strata = NULL,
  spp.weights = NULL,
  check.comm = TRUE,
  convert.traits = FALSE,
  ranks = TRUE
)
```

Arguments

`comm`	Community data, with species as columns and sampling units as rows.
`traits`	Matrix or data.frame of species described by traits, with traits as columns and species as rows (Default traits = NULL).
`phylodist`	Matrix containing phylogenetic distance between species. Must be a complete matrix (not a half diagonal matrix).This matrix can be larger than community data (more species) as long as it has at least all species that are in community data (Default phylodist = NULL).
`envir`	Environmental variables for each community, with variables as columns and sampling units as rows (Default envir = NULL).
`strata`	Strata named vector to specify restricting permutations within species groups (Default strata = NULL).
`spp.weights`	Named vector to specify individual species weights (Default spp.weights = NULL).
`check.comm`	Logical argument (TRUE or FALSE) to remove sampling units and species with total sums equal or less than zero (Default check.comm = TRUE).
`convert.traits`	Logical argument (TRUE or FALSE) to convert factor traits in dummy traits and/or convert ordinal variables in numeric (see ranks argument) (Default convert.traits = FALSE).
`ranks`	Logical argument (TRUE or FALSE) to specify if ordinal variables are convert to ranks. If ranks = TRUE all ordered variable are replaced by their ranks and if ranks = FALSE all ordinal variables are simply treated as continuous variables (Default ranks = TRUE).

Details

The function organizes the data for the functions of the package SYNCSA, placing the matrices of community, traits, phylogenetic distance, environmental varibles, species weights and strata vectors in the same order. The function use as reference the community data for organize all data.frame or matrices in the same order that the sampling units names and species names found in community data set. For this all data sets entered must be correctly named, with rows and columns named. The matrices phylodist, traits, envir can be larger than community data (more species and/or more sampling units) as long as it has at least all species and/or sampling units that are in community data. The function organizes the data despite the absence of one of the data.frames or matrices, provided that the community data had been entered. Unspecified data will appear as NULL.

When trait is a data.frame with different types of variables correctly identified and the argument convert.traits is TRUE factor traits are expanded in dummy traits and ordinal variables are converted in numeric according to ranks argument.

The individual species weights (spp.weights) and strata must be named vectors. The strata vector is a vector for restrict permutation within species groups, insofar as the SYNCSA package the null models are based in permutation of species rather than permutation of sample units.

Value

A object of class metacommunity.data (also of the class list) with the data.frames or matrices:

`call`	The arguments used.
`community`	Community data.
`traits`	Traits data.
`phylodist`	Phylogenetic distance.
`environmental`	Environmental variables.
`community.var.type`	Type of each varible in community data, where 'c' to continuous/numeric, 'o' to ordinal, 'b' to binary and 'f' to factor. Nominal are not allowed.
`traits.var.type`	Type of each varible in traits data. See labels above.
`phylodist.var.type`	Type of each varible in phylodist. See labels above.
`environmental.var.type`	Type of each varible in environmental data. See labels above.
`strata`	The strata vector for permutations.
`put.together`	A list with suggestion to group of traits that are analyzed together, only if convert.traits is TRUE and if some traits are of factor class.
`spp.weights`	The individual species weights vector.
`list.warning`	A list of warning.

Author(s)

Vanderlei Julio Debastiani <vanderleidebastiani@yahoo.com.br>

See Also

`syncsa`, `var.dummy`, `var.type`

Examples

```
data(ADRS)
organize.syncsa(ADRS$community, ADRS$traits, ADRS$phylo, ADRS$envir)
```

`part.cor` *First-order partial correlation coefficient*

Description

Function to obtain the first-order partial correlation coefficient.

Usage

```
part.cor(rxy, rxz, ryz)
```

Arguments

`rxy`	Correlation between xy
`rxz`	Correlation between xz
`ryz`	Correlation between yz

Value

The first-order partial correlation coefficient.

Author(s)

Vanderlei Julio Debastiani <vanderleidebastiani@yahoo.com.br>

See Also

`syncsa`, `cor.matrix`,

`pca` *Principal Components Analysis (PCA) with NA (missing data)*

Description

The function use the option "pairwise.complete.obs" (in function `cor`) for calculate the correlation. The correlation between each pair of variables is computed using all complete pairs of observations on those variables.

Usage

```
pca(data)

## S3 method for class 'pcasyncsa'
plot(
  x,
  show = c("variables", "individuals"),
  axis = c(1, 2),
  xlab = axis[1],
  ylab = axis[2],
  arrows = TRUE,
  text = TRUE,
  points = FALSE,
  ...
)
```

Arguments

`data`	A data frame or matrix with individuals in rows and variables in columns.
`x`	A object of class pcasyncsa.
`show`	Draw "variables" or "individuals".
`axis`	Axis for draw, must have length equal to two (Default axis = c(1, 2)).
`xlab`	Text for x label (Default xlab = axis[1]).
`ylab`	Text for y label (Default ylab = axis[2]).
`arrows`	Logical argument (TRUE or FALSE) to specify if arrows are showed for variables (Default arrows = TRUE).
`text`	Logical argument (TRUE or FALSE) to specify if text are showed for individuals (Default text = TRUE).
`points`	Logical argument (TRUE or FALSE) to specify if points are showed for individuals (Default points = FALSE).
`...`	Parameters for `plot` function.

Value

`decomposition`	list with the results of decomposition of correlation matrix.
`eigenvalues`	Data frame containing all the eigenvalues, the percentage of inertia and the cumulative percentage of inertia.
`individuals`	Coordinates for the individuals.
`variables`	Correlation between original variables and axes.

Author(s)

Vanderlei Julio Debastiani <vanderleidebastiani@yahoo.com.br>

See Also

syncsa syncsa

Examples

```
data(ADRS)
traits<-ADRS$traits
# Some NA
traits[c(1,5),1]<-NA
traits[3,2]<-NA
traits
res<-pca(traits)
res
plot(res, show = "variables", arrows = TRUE)
plot(res, show = "individuals", axis = c(1, 2), text = TRUE)
plot(res, show = "individuals", text = FALSE, points = TRUE)
```

---

permut.row.matrix *Permutate rows in a matrix*

Description

Function to permutate rows in a matrix.

Usage

```
permut.row.matrix(data, strata = NULL, seqpermutation = NULL)
```

Arguments

`data`	A matrix.
`strata`	Strata vector to specify restricting permutations in rows, must be the same length of number of rows in data matrix (Default strata = NULL).
`seqpermutation`	A set of predefined permutation vector (Default seqpermutation = NULL).

Value

`permut.matrix`	The matrix permuted.
`samp`	The sequence of permutation.

Author(s)

Vanderlei Julio Debastiani <vanderleidebastiani@yahoo.com.br>

See Also

syncsa, permut.vector

`permut.vector` *Permutate a vector*

Description

Function to permutate a vector of size n using the function `shuffleSet`.

Usage

```
permut.vector(n, strata = NULL, nset = 999)
```

Arguments

`n`	The length of vector.
`strata`	A vector to specify restricting permutations.
`nset`	The number of permutations to generate for the set (Default strata = NULL).

Value

A matrix of permutations, where each row is a separate permutation.

Author(s)

Vanderlei Julio Debastiani <vanderleidebastiani@yahoo.com.br>

See Also

`syncsa`, `permut.row.matrix`

`procrustes.syncsa` *Procrustes and Partial Procrustes correlations.*

Description

Function to obtain the Procrustes correlation between two matrices and partial Procrustes correlation between three matrices.

Usage

```
procrustes.syncsa(x, y)

procrustes.partial.syncsa(x, y, z)
```

Arguments

`x, y`	Matrix that will be correlated.
`z`	Matrix whose effect will be removed from the correlation between x and y.

Details

The function procrustes.syncsa is a small change in the function `procrustes`. The function uses a correlation-like statistic derived from the symmetric Procrustes sum of squares. Partial Procrustes correlations is obtained by Procrustes correlation of residuals between each variable in z being used as linear predictor of all variables in x and y. For more details, see `syncsa`.

Value

Procrustes correlation between matrices.

Author(s)

Vanderlei Julio Debastiani <vanderleidebastiani@yahoo.com.br>

References

Peres-Neto, P.R. and Jackson, D.A. (2001). How well do multivariate data sets match? The advantages of a Procrustean superimposition approach over the Mantel test. Oecologia 129: 169-178.

See Also

`syncsa`, `cor.matrix`

`ProgressBAR` *Text Progress Bar*

Description

Function to display a progress bar in the R console. See `txtProgressBar`.

Usage

```
ProgressBAR(n, N, ...)
```

Arguments

`n`	Number of the current progress.
`N`	Total number of cases.
`...`	Other parameters for the txtProgressBar function.

Author(s)

Vanderlei Julio Debastiani <vanderleidebastiani@yahoo.com.br>

`rao.diversity` *Rao's quadratic entropy*

Description

Calculates Rao's quadratic entropy, functional and phylogenetic redundancy.

Usage

```
rao.diversity(
  comm,
  traits = NULL,
  phylodist = NULL,
  checkdata = TRUE,
  ord = "metric",
  put.together = NULL,
  standardize = TRUE,
  transformation = "standardized",
  spp.weights = NULL,
  ...
)
```

Arguments

`comm`	Community data, with species as columns and sampling units as rows. This matrix can contain either presence/absence or abundance data. Alternatively comm can be an object of class metacommunity.data, an alternative way to set all data.frames/matrices. When you use the class metacommunity.data the arguments traits, phylodist and put.together must be null. See details.
`traits`	Data frame or matrix data of species described by traits, with traits as columns and species as rows (Default traits = NULL).
`phylodist`	Matrix containing phylogenetic distance between species (Default phylodist = NULL).
`checkdata`	Logical argument (TRUE or FALSE) to check if species sequence in the community data follows the same order as the one in the trait and in the phylodist matrices (Default checkdata = TRUE).
`ord`	Method to be used for ordinal variables, see `gowdis` (Default ord = "metric").
`put.together`	List to specify group of traits. Each group specify receive the same weight that one trait outside any group, in the way each group is considered as unique trait (Default put.together = NULL). This argument must be a list, see examples in `syncsa`.
`standardize`	Logical argument (TRUE or FALSE) to specify if standardize phylogenetic distance to range into range 0 to 1. (Default standardize = TRUE).
`transformation`	Method to transformation, "none", "standardized", "weights" or "max.weights" (Default transformation = "standardized").

`spp.weights`	Vector with 0 or 1 to specify individual species weights (Default spp.weights = NULL).
`...`	Parameters for `gowdis` function.

Details

Rao's quadratic entropy is a measure of diversity of ecological communities defined by Rao (1982) and is based on the proportion of the abundance of species present in a community and some measure of dissimilarity among them. The dissimilarity range from 0 to 1 and is based on a set of specified functional traits or in the phylogenetic dissimilarity.

For the trait data, the function calculates the square root of the one-complement of Gower‘s similarity index, in order to have a dissimilarity matrix with Euclidean metric properties. Gower‘s index ranges from 0 to 1 and can handle traits measured indifferent scales. When the species are completely different in terms of their traits, Rao quadratic entropy is equivalent to the Gini-Simpson index. Traits data can be numeric, factor or ordered factor. For this be considered traits data must be of data.frame class and containing each variable type determined. For additional details and requirements of function please see `gowdis`.

Functional redundancy is defined purely as the difference between species diversity and Rao‘s quadratic entropy based on their functional dissimilarity (de Bello et al. 2007). The same definition is used for phylogenetic redundancy.

By default, the community data is standardization to row totals will be 1. The options are: "none" no transformation is applied; "standardized" the community data is standardized to row totals will be 1; and "weights" community data is first "standardized" and when, individual species weights are multiplied to each species entries. The argument transformation also allow an additional method to weights individual species, called "max.weights". In this method community data is standardization as default, however dissimilarity matrix is weighted indeed. The argument spp.weights specify the weights to target species and the dissimilarity weights are 1 if at least one species in the pair belongs to the target community, otherwise weights to pair are 0.

Package **SYNCSA** requires that the species and community sequence in the data.frame or matrix must be the same for all dataframe/matrices. The function `organize.syncsa` organizes the data for the functions of the package, placing the matrices of community, traits, phylogenetic distance in the same order. The function use of function organize.syncsa is not requered for run the functions, but is recommended. In this way the arguments comm, traits, phylodist, as well as the argument put.together and spp.weights, can be specified them as normal arguments or by passing them with the object returned by the function `organize.syncsa` using, in this case only the argument comm. Using the object returned by organize.syncsa, the comm argument is used as an alternative way of entering to set all data.frames/matrices, and therefore the other arguments (traits, phylodist, put.together and spp.weights) must be null.

Value

`Simpson`	Gini-Simpson index within each community (equivalent to Rao quadratic entropy with null, crisp, similarities).
`FunRao`	Rao quadratic entropy within each community, considering trait distance.
`FunRedundancy`	Functional redundancy in each community.
`PhyRao`	Rao quadratic entropy within each community, considering phylogenetic distance.

`PhyRedundancy`	Phylogenetic redundancy in each community.

Note

IMPORTANT: The sequence species show up in community data matrix MUST be the same as they show up in traits and phylodist matrices as well as in the species weights vector. See details and `organize.syncsa`.

Author(s)

Vanderlei Julio Debastiani <vanderleidebastiani@yahoo.com.br>

References

de Bello, F.; Leps, J.; Lavorel, S. & Moretti, M. (2007). Importance of species abundance for assessment of trait composition: an example based on pollinator communities. Community Ecology, 8, 163:170.

Pillar, V.D.; Blanco, C.C.; Muler, S.C.; Sosinski, E.E.; Joner, F. & Duarte, L.d.S. (2013). Functional redundancy and stability in plant communities. Journal of Vegetation Science, 24, 963:974.

Rao, C.R. (1982). Diversity and dissimilarity coefficients: a unified approach. Theoretical Population Biology, 21, 24:43.

See Also

`organize.syncsa`, `gowdis`, `syncsa`

Examples

```
data(ADRS)
rao.diversity(ADRS$community)
rao.diversity(ADRS$community, traits = ADRS$traits)
```

`syncsa` *SYNCSA*

Description

This function integrates several steps for the analysis of phylogenetic assembly patterns and their links to traits and ecological processes in a metacommunity (Pillar et al. 2009, Pillar & Duarte 2010, Debastiani & Pillar 2012). The function implement methods that have been available in the SYNCSA application written in C++ (by Valerio Pillar, available at http://ecoqua.ecologia.ufrgs.br/SYNCSA.html). See details.

Usage

```
syncsa(
  comm,
  traits = NULL,
  phylodist = NULL,
  envir = NULL,
  checkdata = TRUE,
  ro.method = "mantel",
  method = "pearson",
  dist = "euclidean",
  scale = TRUE,
  scale.envir = TRUE,
  ranks = TRUE,
  asym.bin = NULL,
  ord = "metric",
  put.together = NULL,
  na.rm = FALSE,
  transformation = "standardized",
  spp.weights = NULL,
  strata = NULL,
  permutations = 999,
  parallel = NULL,
  notification = TRUE
)

## S3 method for class 'syncsa'
print(x, ...)

syncsa.obs(
  comm,
  traits = NULL,
  phylodist = NULL,
  envir = NULL,
  scale = TRUE,
  scale.envir = TRUE,
  ranks = TRUE,
  asym.bin = NULL,
  ord = "metric",
  put.together = NULL,
  na.rm = FALSE,
  transformation = "standardized",
  spp.weights = NULL,
  notification = TRUE
)
```

Arguments

`comm`	Community data, with species as columns and sampling units as rows. This matrix can contain either presence/absence or abundance data. Alternatively comm can be an object of class metacommunity.data, an alternative way to set all data.frames/matrices. When you use the class metacommunity.data the arguments traits, phylodist, envir and put.together must be null. See details.
`traits`	Data frame or matrix data of species described by traits, with traits as columns and species as rows (Default traits = NULL).
`phylodist`	Matrix containing phylogenetic distance between species. Must be a complete matrix, not a half diagonal matrix (Default phylodist = NULL).
`envir`	Environmental variables for each community, with variables as columns and sampling units as rows (Default envir = NULL).
`checkdata`	Logical argument (TRUE or FALSE) to check if species sequence in the community data follows the same order as the one in the trait and in the phylodist matrices and if sampling units in the community data follows the same order as the one in the environmental data (Default checkdata = TRUE).
`ro.method`	Method to obtain the correlation, "mantel", "procrustes", "coinertia", "none" or "nullmodel" (Default ro.method = "mantel").
`method`	Mantel correlation method, as accepted by cor: "pearson", "spearman" or "kendall" (Default method = "pearson").
`dist`	Dissimilarity index used for Mantel correlation, as accepted by vegdist: "manhattan", "euclidean", "canberra", "bray", "kulczynski", "jaccard", "gower", "altGower", "morisita", "horn", "mountford", "raup" , "binomial" or "chao" (Default dist = "euclidean").
`scale`	Logical argument (TRUE or FALSE) to specify if the traits are measured on different scales (Default Scale = TRUE). When scale = TRUE traits are measured on different scales the the matrix T is subjected to standardization within each trait. When scale = FALSE traits are measured on the same scale and the matrix T is not subjected to standardization. Furthermore, if scale = TRUE the matrix of traits is subjected to standardization within each trait, and Gower Index is used to calculate the degree of belonging to the species, and if scale = FALSE the matrix of traits is not subjected to standardization, and Euclidean distance is calculated to determine the degree of belonging to the species.
`scale.envir`	Logical argument (TRUE or FALSE) to specify if the environmental variables are measured on different scales. If the enviromental variables are measured on different scales, the scale.envir = TRUE the matrix with enviromental variables is subjected to centralization and standardization within each variable. (Default scale.envir = TRUE).
`ranks`	Logical argument (TRUE or FALSE) to specify if ordinal variables are convert to ranks (Default ranks = TRUE).
`asym.bin`	Vector listing the asymmetric binary traits, see `gowdis` (Default asym.bin = NULL).
`ord`	Method to be used for ordinal traits, see `gowdis` (Default ord = "metric").

`put.together`	List to specify group of traits. Each group specify receive the same weight that one trait outside any group, in the way each group is considered as unique trait (Default put.together = NULL). This argument must be a list, see examples.
`na.rm`	Logical argument (TRUE or FALSE) to specify if pairwise distances should be deleted in cases of missing observations (Default na.rm = FALSE).
`transformation`	Method to transformation, "none", "standardized" or "weights" (Default transformation = "standardized").
`spp.weights`	Vector with 0 or 1 to specify individual species weights (Default spp.weights = NULL).
`strata`	Strata named vector to specify restricting permutations within species groups (Default strata = NULL).
`permutations`	Number of permutations in assessing significance.
`parallel`	Number of parallel processes. Tip: use parallel::detectCores() (Default parallel = NULL).
`notification`	Logical argument (TRUE or FALSE) to specify if notification of missing observations should to be shown (Default notification = TRUE).
`x`	An object of class syncsa.
`...`	Other parameters for the respective functions.

Details

Package **SYNCSA** requires that the species and community sequence in the data.frame or matrix must be the same for all dataframe/matrices. The function `organize.syncsa` organizes the data for the functions of the package, placing the matrices of community, traits, phylogenetic distance, environmental varibles and strata vector in the same order. The function use of function organize.syncsa is not requered for run the functions, but is recommended. It requires data organized into the following matrices: (1) the presences or abundances of species in a set of communities (**W**); (2) the phylogenetic pairwise dissimilarities of these species (**DF**); (3) a set of functional traits describing the species (**B**), which may be a mixture of binary and quantitative traits (continual and ordinal), but not nominal ones (these should be expanded into binary traits); and (4) the ecological gradient of interest, which may be one or more factors to which the communities respond or ecosystem effects of the communities (**E**). In this way the arguments comm, traits, phylodist, envir, as well as the arguments put.together and strata, can be specified them as normal arguments or by passing them with the object returned by the function `organize.syncsa` using, in this case only the argument comm. Using the object returned by organize.syncsa, the comm argument is used as an alternative way of entering to set all data.frames/matrices, and therefore the other arguments (traits, phylodist, envir, put.together and strata) must be null.

Correlations

The function computes several correlations (Mantel or Procrustes) that express trait-convergence assembly patterns (TCAP), trait-divergence assembly patterns (TDAP), and phylogenetic signal in functional traits at the species pool level and at the metacomunity level. This function also generates P-values by permutation testing based on null models (Pillar et al. 2009, Pillar & Duarte 2010).

ro(TE)

This correlation refers to trait-convergence assembly patterns related to the ecological gradient (TCAP, Pillar et al. 2009). For evaluating TCAP, by matrix multiplication we define **T = WB**, which with previous standardization of **W** to unit column totals will contain the trait averages in each community. The elements in **T** are community weighted mean values or community functional parameters (Violle et al. 2007). Standardization of the traits (rows) in **T** is needed if the trait set contains traits measured with different scales. By using matrix correlation, we evaluate how the trait patterns in **T** are associated to ecological gradients in **E**. For relating **T** to **E**, using Mantel correlation we define a distance matrix of the communities (**DT**) using **T**, and another distance matrix of the community sites (**DE**) using **E**. The matrix correlation ro(**TE**) = ro(**DT**;**DE**) measures the level of congruence between TCAP and **E**. A strong correlation ro(**TE**) indicates the factors directly or indirectly represented in **E** are involved in ecological filtering of species that, at least for the traits considered in the analysis, consistently produce trait-convergence assembly patterns along the gradient comprising the metacommunity.

ro(XE) and ro(XE.T)

These matrix correlations refer to trait-divergence assembly patterns related to the ecological gradient (TDAP, Pillar et al. 2009). For the identification of TDAP, in a first step the species pairwise similarities (in the range 0 to 1) in matrix **SB** based on traits in **B** are used to define matrix **U** with degrees of belonging of species to fuzzy sets. By matrix multiplication **X = WU** will contain the species composition of the communities after fuzzy-weighting by their trait similarities (each row in **X** will refer to a species). Matrix **X** expresses both TCAP and TDAP (Pillar et al. 2009). By using matrix correlation, we evaluate how the trait patterns in **X** (TCAP and TDAP) are associated to ecological gradients in **E**. For relating **X** to **E**, we define a distance matrix of the communities (**DX**) using **X**, and another distance matrix of the community sites (**DE**) using **E**. The matrix correlation ro(**XE**) = ro(**DX**;**DE**) between **X** and **E** is defined. We then remove the trait-convergence component ro(**TE**) from ro(**XE**) by computing the partial matrix correlation ro(**XE.T**), which measures the level of congruence between TDAP and **E**. Trait-divergence assembly patterns (TDAP, Pillar et al. 2009) may result from community assembly processes related to biotic interactions (Stubbs & Wilson 2004; Wilson 2007).

ro(PE)

This matrix correlation refers to the phylogenetic structure related to the ecological gradient comprising the metacommunity. The phylogenetic pairwise dissimilarities in **DF** are transformed into similarities and used to define degrees of belonging qij to fuzzy sets. This is analogous to the definition of functional fuzzy sets (Pillar & Orloci 1991; Pillar et al. 2009). Based on the phylogenetic similarities, every species i among s species in the metacommunity specifies a fuzzy set to which every species j (j = 1 to s species, including species i) belongs with a certain degree of belonging in the interval [0, 1]. In our definition, each row in matrix **Q** with the degrees of belonging must add to unit, i.e., the degrees of belonging of a given species across the fuzzy sets are standardized to unit total. By matrix multiplication **P = WQ** will contain the composition of the communities after fuzzy-weighting of species presences or abundances by the species‘ phylogenetic similarities. Each column in matrix **P** holds the phylogenetic structure of a community. The standardization of **Q** is essential for the community totals in each column in **W** remaining the same in **P**. Further, matrix **W** is adjusted to unit column totals prior to multiplication, so that the total richness or abundance within each community in **W** will be standardized. Matrix correlation ro(**PE**) = ro(**DP**;**DE**) measures the strength of the association between community distances based on their phylogenetic structure in **DP** and distances based on their ecological conditions (**DE**). Further, **P** can be explored for phylogenetic patterns at the metacommunity level by using, e.g., ordination techniques.

ro(PT) and ro(PX.T)

These matrix correlations measure phylogenetic signal at the metacommunity level related to TCAP and to TDAP. We define phylogenetic signal at the metacommunity level related to TCAP (PSMT) as the correlation between the phylogenetic structure described in matrix **P** and the trait-convergence structure described in matrix **T**. For this, a proper distance matrix (e.g. Euclidean distances) of communities (**DP**) is computed using **P** and another distance matrix of the same communities (**DT**) is computed using **T**. Then matrix correlation ro(**PT**) = ro(**DP**;**DT**) will measure the level of congruence between variation in **P** and **T**, which is a measure of PSMT. A strong phylogenetic signal at the metacommunity level is expected when communities that are more similar in terms of phylogenetic structure are also similar regarding their average trait values. We also define phylogenetic signal at the metacommunity level related to TDAP (PSMX.T) as the partial matrix correlation ro(**PX.T**) = ro(**DP**;**DX.DT**) between community distances DP computed on phylogenetic structure and community distances **DX** computed on species composition after fuzzy-weighting by the species, or trait similarities, removing the effect of TCAP (**DT**). This is analogous to TDAP.

ro(BF)

This matrix correlation measures phylogenetic signal at the species pool level (PSS, Pillar & Duarte 2010). We define PSS as the matrix correlation ro(**FB**) = ro(**DF**;**DB**) between species phylogenetic dissimilarities (already defined as matrix **DF**) and species trait dissimilarities (derived from already defined matrix **SB**) computed on any number of traits from matrix **B**. The species pool refers to the species present in the metacommunity.

Additional matrix correlations

The matrix correlations ro(**TE.P**) and ro(**XE.P**) are also computed, which may be useful for evaluating causal models in path analysis.

Mantel correlations

The Mantel and Partial Mantel statistics are calculated simply as the correlation entries the dissimilarity matrices, using standard Mantel test (see `mantel` and `cor.matrix`). Partial Mantel statistic use paired correlation between the three matrices and obtains the partial correlation using the formula of first-order partial correlation coefficient. The significances are obtained using a different procedure than standard Mantel test, see section Testing against null models below.

Procrustes correlations

The Procrustes correlation uses symmetric Procrustes as a measure of concordance between the data matrices (see `procrustes` and `procrustes.syncsa`). Procrustes procedure use rotation, translation, and rescaling for minimizing sum of squared differences between two data sets. The correlation of Procrustes is calculated as the statistic derived from the symmetric Procrustes sum of squares, representing the optimal fit between the two data matrices. Partial Procrustes correlation is obtained by Procrustes correlation between residuals matrices. Firstly one Principal Components Analysis (PCA, see `prcomp`) is performed in the matrix Z for dimensionality reduction. The max number of axis kept in the analysis is the number of sampling units divided by 2, this axes of PCA represent the total variation in the Z matrix. After the kept axes are used as predictor in one linear model for each variable of the matrices X and Y. For this a linear model is build using as response one variable of X (same via for Y matrix) and as predictor all remaining axes of PCA, after model fitted and the residual are extracted with the aim of form the residual matrix. The linear model is repeated in the other variables, only with the changed the response variable. The same procedure is performed in the matrix Y. Both residual matrices are submitted to Procrustes analysis and the statistic is returned as a partial correlation, the Partial Procrustes statistic. The significances are obtained using the same procedure than Mantel test, see section Testing against null models below.

Co-inertia correlations

Co-inertia Analysis uses RV coefficient as a measure of correlation between the data matrices. The procedure is equal to Procrustes correlation, however using the RV coefficient.

The ro.method "none" or "nullmodel"

When ro.method is equal to "none" only observed matrices are returned, already when it is equal to "nullmodel" observed matrices and null matrices generated according the null models are returned. This null matrices can be used to apply any statistical method to test relation between matrices not include in the package (e.x. linear models).

Testing against null models

All the matrix correlations are tested against null models. The null model is defined accoding to the correlation being tested. Usually in the SYNCSA package the null models are based in permutation of species rather than permutation of sample units. For ro(**TE**),each permutation generates a random matrix **T** calculated after the permutation among the species vectors in matrix **B**. For ro(**XE**) and ro(**XE.T**), each permutation generates a random matrix **X** after the permutation among species fuzzy sets (rows) in matrix **U**. For ro(**PE**), ro(**PT**), and ro(**PX.T**), each permutation generates a random matrix **P** after the permutation among species fuzzy sets (rows) in matrix **Q**. For ro(**BF**), a conventional Mantel test is performed with dissimilarity matrices **DF** and **DB**. Analogous null models are used for testing the additional matrix correlations; that is, the same null model for ro(**TE**) is used for ro(**TE.P**), the same model for ro(**XE**) is used for ro(**XE.P**). The permutation can be restrict within species groups specifying the strata argument.

Traits types

Traits data can be numeric, factor or ordered factor. For this be considered in the analyses traits data must be of data.frame class and containing each variable type determined. Gower index is used to calculate the similarity between species, using the function gowdis of package FD. For additional details and requirements of function please see `gowdis`.

Community data standardization

By default the community data is standardization to row totals will be 1. The options are: "none" no transformation is applied; "standardized" the community data is standardized to row totals will be 1; and "weights" community data is first "standardized" and when, individual species weights are multiplied to each species entries. An additional method to weights individual species is included only `rao.diversity` function. Current version of package does not allow choose standardization/transformation method in the `optimal` function.

Missing data

The functions ignore missing data when specified. In the case of direct multiplication of matrices the missing data are replaced by 0, ignoring the cell with missing value. For the matrix **T = WB** an adjustment is done by divide each cell of the product matrix (**T**) by the sum of species proportion with trait data in **B**. Result matrices are shown without missing values. Where the matrices are calculated using a dissimilarity index (matrix **U** and correlations between matrices) the missing data are ignored as in `vegdist` function. In some cases the dissimilarity matrices obtained by the function `vegdist` still contain some missing values. In these cases the rest of the procedure will be affected. In these cases you can find solutions in impute the missing values.

Error messenger and options

The data pass by several ckeck points that can produce error messenger. The matrices or data frames must be contain only numeric, binary or ordinal variables, in the way that nominal variable should be expanded into binary (see `var.dummy`). For enhance the code speed some functions use by default matrix algebra, this option can produce error under certain circumstances. This global option can be changed using options("SYNCSA.speed" = FALSE). If SYNCSA.speed = TRUE for use matrix algebra and if SYNCSA.speed = FALSE use not another function of same procedure.

Value

`call`	The arguments used.
`notes`	Some notes about the statistics.
`statistics`	Correlations roTE, roXE, roPE, roPT, roPX.T, roXE.T, roTE.P, roXE.P and roBF, and their significance levels based on permutations.
`matrices`	A list with observed matrices produced for the functions, see details.
`matrices.null`	A list of list with null matrices generated according the null model, only if ro.method is "nullmodel". The number of matrices returned depends of the number of permutations.
`FunRao`	Rao quadratic entropy within each community, considering trait distance.
`traits.weights`	Weight for each trait.

Note

The function calculates the correlations despite the lack of one of the matrices, provided that community data had been entered. Correlations including unspecified matrices will appear with NA.

IMPORTANT: The sequence of species in the community data matrix MUST be the same as that in the phylogenetic distance matrix and in traits matrix. Similarly, the sequence of communities in the community data matrix MUST be the same as that in the environmental data. See details and `organize.syncsa`.

Author(s)

Vanderlei Julio Debastiani <vanderleidebastiani@yahoo.com.br>

References

Debastiani, V.J & Pillar, V.D., (2012). SYNCSA-R tool for analysis of metacommunities based on functional traits and phylogeny of the community components. Bioinformatics, 28(15), 2067–2068.

Pillar, V.D.; Duarte, L.d.S., (2010). A framework for metacommunity analysis of phylogenetic structure. Ecology Letters, 13, 587:596.

Pillar, V.D., Duarte, L.d.S., Sosinski, E.E. & Joner, F. (2009). Discriminating trait-convergence and trait-divergence assembly patterns in ecological community gradients. Journal of Vegetation Science, 20, 334:348.

Pillar, V.D. & Orloci, L., (1991). Fuzzy components in community level comparisons. In: Computer Assisted Vegetation Analysis (eds Feoli, E. & Orloci, L.). Kluwer, Dordrecht, 87:93.

Stubbs, W.J. & Wilson, J.B., (2004). Evidence for limiting similarity in a sand dune community. Journal of Ecology, 92, 557:567.

Violle, C., Navas, M.L., Vile, D., Kazakou, E., Fortunel, C., Hummel, I. & Garnier, E., (2007). Let the concept of trait be functional! Oikos, 116, 882:892.

Wilson, J.B., (2007). Trait-divergence assembly rules have been demonstrated: limiting similarity lives! A reply to Grime. Journal of Vegetation Science, 18, 451:452.

See Also

organize.syncsa, matrix.t, matrix.x, matrix.p, optimal, rao.diversity, cor.matrix, var.type, var.dummy

Examples

```
data(ADRS)
syncsa(ADRS$community, ADRS$traits, ADRS$phylo, ADRS$envir, permutations = 99)
data(flona)
put.together<-list(c("fol","sem"), c("tam", "red"))
put.together
res<-syncsa(flona$community, flona$traits, envir = flona$environment,
   put.together = put.together, permutations = 99)
res$spp.weights
```

`var.dummy` *Generate dummy variable*

Description

Function to expand factor variables in dummy variables in a data.frame. See details.

Usage

```
var.dummy(data)
```

Arguments

`data`	A data.frame.

Details

Variables in the data.frame of class factor is expanded in dummy variables. Each level of factor produce a new column in the dataframe, with presence (1) or absense (0) of level. The name of columns is a combination of orginal variable name plus the level separate by underscore (_). Ordered factor and character class are not expanded.

Value

`data`	The data with all variables of class factor expanded.
`together`	A list with sugestion to group of traits that are analysed together.

Author(s)

Vanderlei Julio Debastiani <vanderleidebastiani@yahoo.com.br>

See Also

syncsa, organize.syncsa, var.type

var.type *Check the type of variables*

Description

Function to check the type of variables in a data.frame or matrix. This function was extracted and slightly modified of the function `gowdis`.

Usage

```
var.type(data)
```

Arguments

`data`	A data.frame or matrix.

Value

A vector with the variable types, where 'c' is continuous/numeric, 'o' is ordinal, 'b' is binary, 'n' is nominal and 'f' is factor.

Author(s)

Vanderlei Julio Debastiani <vanderleidebastiani@yahoo.com.br>

See Also

`syncsa`, `organize.syncsa`, `var.dummy`

Index