Package 'TPD'

October 12, 2022

Type Package

Title Methods for Measuring Functional Diversity Based on Trait Probability Density

Version 1.1.0

Date 2019-07-02

Author Carlos P. Carmona <perezcarmonacarlos@gmail.com>

Maintainer Carlos P. Carmona <perezcarmonacarlos@gmail.com>

Description Tools to calculate trait probability density functions (TPD) at any scale (e.g. populations, species, communities). TPD functions are used to compute several indices of functional diversity, as well as its partition across scales. These indices constitute a unified framework that incorporates the underlying probabilistic nature of trait distributions into uni- or multidimensional functional trait-based studies. See Carmona et al. (2016) <doi:10.1016/j.tree.2016.02.003> for further information.

License GPL-3

LazyData TRUE

Depends ggplot2 (>= 1.0.0), ks (>= 1.9.2)

Imports gridExtra (>= 0.9.1), mvtnorm (>= 0.8), graphics, stats

Suggests knitr, testthat, rmarkdown

VignetteBuilder knitr

RoxygenNote 6.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2019-07-02 20:30:03 UTC

R topics documented:

dissim 2
plotTPD 3
Rao 5

redundancy 6
REND 7
TPD 8
TPDc 9
TPDs 10
TPDsMean 12
tSamp 14
uniqueness 15

Index **17**

`dissim` *Overlap-Based Functional Dissimilarity and its Decomposition*

Description

`dissim` calculates the functional dissimilarity between pairs of communities or populations, as well as its decomposition into shared and non-shared trait volume.

Usage

```
dissim(x = NULL)
```

Arguments

`x`	Either an object of class "TPDcomm", generated with the `TPDc` function, containing the TPDc of the considered communities, or an object of class "TPDsp", generated with the `TPDs` or `TPDsMean` functions, containing the TPDs of the considered populations or species.

Value

`dissim` returns the overlap-based functional dissimilarity between all pairs of populations/species/communities, along with the decomposition of dissimilarity between shared and non-shared trait volume.

Examples

```
# 1.  Compute the TPDs of three different species:
traits_iris <- iris[, c("Sepal.Length", "Sepal.Width")]
sp_iris <- iris$Species
TPDs_iris <- TPDs(species = sp_iris, traits_iris)

 #2. Compute the TPDc of three different communities:
abundances_comm_iris <- matrix(c(c(0.9, 0.1, 0), #I. setosa dominates
                         c(0.0, 0.9,  0.1 ), #I. Versic. dominates; setosa absent
                         c(0.0, 0.1,  0.9 )), #I. virg. dominates; setosa absent
                         ncol = 3, byrow = TRUE, dimnames = list(paste0("Comm.",1:3),
                          unique(iris$Species)))
TPDc_iris <- TPDc(TPDs = TPDs_iris, sampUnit = abundances_comm_iris)
```

```
#3. Estimate functional dissimilarity
example_dissimilarity_comm <- dissim (TPDc_iris)
example_dissimilarity_sps <- dissim (TPDs_iris)
```

---

plotTPD *Plotting Trait Probability Distributions*

Description

plotTPD plots a TPD object (created with either TPDs ot TPDc) of 1 or 2 dimensions. In the 1-dimension case, plotTPD displays the trait in the x-axis and the probability associated to each trait value in the y-axis. In the 2-dimensions case, plotTPD displays traits in the x- and y-axes, and probabilities are indicated by a gradient of colors. The function yields a panel for each TPD calculated (one for each species or population in the case of TPDs and one for each community in the case of TPDc).

Usage

```
plotTPD(TPD, whichPlot = NULL, nRowCol = NULL, color1 = "grey60",
  leg = TRUE, leg.text = NULL, leg.pos = "topright", leg.cex = 1)
```

Arguments

TPD	An object of class "TPDsp" or TPDcomm", generated with the TPDs or TPDc functions, respectively, containing the TPDs of the considered populations or species or the TPDc of the considered communities.
whichPlot	A vector indicating the identity of the species, populations or communities to plots. Defaults to NULL, in which case, all cases are plotted.
nRowCol	A vector with two integers indicating the number of rows and columns of the layout. The product of the two numbers must be greater or equal than the length of whichPlot, so that all plots can be included. Defaults to NULL, in which case the layout is automatically selected.
color1	The color used to fill the TPD in the 1-dimension case. Defaults to "grey60".
leg	Logical, indicating whether a legend with the name of the species, population or community should be added in the 1-dimension case.
leg.text	Vector, containing the names to be used in the legend of each plot in the 1-dimension case, or in the plots title in the 2-dimensions case. If provided, it must have the same length as whichPlot. Defaults to NULL, in which case the names of the species/populations or communities are used.
leg.pos	Character, indicating the location of the legend in the plot. Possible values are: "bottomright", "bottom", "bottomleft", "left", "topleft", "top", "topright", "right" and "center". Defaults to "topright".
leg.cex	Numeric, indicating the character expansion factor relative to current par("cex") for the text of the legend.

Details

In the 2-dimensions case, `plotTPD` requires the packages `ggplot2` and `gridExtra` in order to work.

Examples

```
# 1.  Compute the TPDs of five different species. SP3 is in the center of
#   the trait space, and the rest of species in the corners
set.seed(1)
nind <- 10
species_ex <- c(rep("SP1",nind), rep("SP2",nind), rep("SP3",nind),
 rep("SP4",nind), rep("SP5",nind))
traits_ex <- data.frame(trait1 = c(rnorm(nind, 10, 3),
                                   rnorm(nind, 10, 3),
                                   rnorm(nind, 15, 3),
                                   rnorm(nind, 20, 3),
                                   rnorm(nind, 20, 3)),
                        trait2 = c(rnorm(nind, 10, 3),
                                   rnorm(nind, 20, 3),
                                   rnorm(nind, 15, 3),
                                   rnorm(nind, 10, 3),
                                   rnorm(nind, 20, 3)))
## Not run:
species_TPDs_2D <- TPDs (species = species_ex, traits = traits_ex)
# Plot all species
plotTPD(species_TPDs_2D)
# Plot only species 3
plotTPD(species_TPDs_2D, whichPlot = 3)

## End(Not run)
#1 dimension case:
species_TPDs_1D <- TPDs (species = species_ex, traits = traits_ex$trait1)
plotTPD(species_TPDs_1D)

#Now, let us plot communities (TPDc)
#2. three different communities with different abundances of each species
abundances_ex <- matrix(c(c(0.05, 0.05, 0.8,  0.05, 0.05),
                          c(0.9,  0,    0,    0,    0.1 ),
                          c(0,    0,    1,    0,    0   )),
                  ncol = 5, byrow = TRUE, dimnames = list(paste0("Comm.",1:3),
                    unique(species_ex)))
## Not run:
example_TPDc_2D <- TPDc (TPDs = species_TPDs_2D, sampUnit = abundances_ex)
plotTPD(example_TPDc_2D)

## End(Not run)
example_TPDc_1D <- TPDc (TPDs = species_TPDs_1D, sampUnit = abundances_ex)
plotTPD(example_TPDc_1D)
```

Rao *Rao's Quadratic Entropy and its Partition*

Description

Rao

Usage

```
Rao(diss = NULL, TPDc = NULL, regional = TRUE)
```

Arguments

diss	An object of class "OverlapDiss", generated with the dissim function, containing the dissimilarity of the considered populations or species.
TPDc	An object of class "TPDcomm", generated with the TPDc function, containing the containing the TPDc of all the communities whose functional diversity is going to be calculated. Species (or populations) identities and their relative abundance in each community will be extracted from this object.
regional	Logical indicating if the correction by Villeger and Mouillot (2008) is applied or not. Defaults to TRUE.

Value

Rao returns a list containing functional diversity at different scales for the whole dataset and for pairs of communities.

Information for the whole dataset include: i) alpha functional diversity of each sampling unit expressed as raw rao values (alpha_rao) and in equivalent numbers alpha_eqv), ii) the average alpha functional diversity of the sampling units, calculated following de Bello et al. (2010) (mean_alpha_eqv), iii) gamma functional diversity for the whole dataset, expressed as raw rao values (gamma_rao) and in equivalent numbers (gamma_eqv), and iv) beta functional diversity for the whole dataset expressed in proportional terms (see de Bello et al. 2010) (beta_prop).

Information for pairs of communities (contained in the element pairwise) include the average alpha (expressed in equivalent numbers) of each pair of communities, gamma of each pair of communities and beta functional diversity for each pair of communities.

Examples

```
# 1.  Compute the TPDs of three different species.
traits_iris <- iris[, c("Sepal.Length", "Sepal.Width")]
sp_iris <- iris$Species
TPDs_iris <- TPDs(species = sp_iris, traits_iris)
#2. Compute the dissimilarity between the three species:
dissim_iris_sp <- dissim(TPDs_iris)
#3. Compute the TPDc of five different communities:
abundances_comm_iris <- matrix(c(c(0.9,  0.1, 0), # setosa dominates
                                 c(0.4,  0.5, 0.1 ),
```

```
                                  c(0.15, 0.7, 0.15), #versicolor dominates
                                  c(0.1,  0.5, 0.4),
                                  c(0,    0.1, 0.9)), #virginica dominates
                            ncol = 3, byrow = TRUE, dimnames = list(paste0("Comm.",1:5),
                              unique(iris$Species)))
TPDc_iris <- TPDc( TPDs = TPDs_iris, sampUnit = abundances_comm_iris)

#4. Compute Rao:
Rao_iris <- Rao(diss = dissim_iris_sp, TPDc = TPDc_iris)
```

---

redundancy *Functional Redundancy of Communities*

Description

redundancy calculates the functional redundancy of communities, considering single or multiple traits. The functional volume (indicated by Functional Richness) occupied by a community with high functional redundancy should not decrease substantially when some species are lost, and vice versa.

Usage

```
redundancy(TPDc = NULL)
```

Arguments

TPDc	An object of class "TPDcomm", generated with the TPDc function, containing the TPDc of the considered communities.

Value

redundancy returns a list containing the functional redundancy values of all the communities from TDPc, along with the number of species of each community. It also returns a vector with the values of relative redundancy (i.e. redundancy divided by richness minus one).

Examples

```
#1. Compute the TPDs of three different species.
traits_iris <- iris[, c("Sepal.Length", "Sepal.Width")]
sp_iris <- iris$Species
TPDs_iris <- TPDs(species = sp_iris, traits_iris)

#2. Compute the TPDc of five different communities:
abundances_comm_iris <- matrix(c(c(0.9,  0.05, 0.05), #I. setosa dominates
                                 c(0.0,  0.5,  0.5 ), #I. setosa absent
                                 c(0.33, 0.33, 0.33), #Equal abundances
                                 c(0.1,  0.45, 0.45), #Versicolor and virginica dominate
                                 c(0.5,  0,    0.5)), #versicolor absent
                           ncol = 3, byrow = TRUE, dimnames = list(paste0("Comm.",1:5),
```

```
                              unique(iris$Species)))
TPDc_iris <- TPDc( TPDs = TPDs_iris, sampUnit = abundances_comm_iris)

#3. Estimate functional redundancy
FRed_iris <- redundancy(TPDc = TPDc_iris)
```

REND *Functional Evenness, Richness and Divergence of Communities, Species or Populations*

Description

REND computes Functional Richness, Functional Evenness and Functional Divergence, the three primary components of functional diversity (Mason et al. 2005) for single or multiple traits. Although these components were originally intended to be calculated for communities, REND also allows to compute them for populations or species. In the case of communities, all the calculations are based on the TPDc of the considered communities; therefore results are independent of any underlying feature of the species that compose the communities.

Usage

```
REND(TPDc = NULL, TPDs = NULL)
```

Arguments

TPDc	An object of class "TPDcomm", generated with the TPDc function, containing the TPDc of the considered communities.
TPDs	An object of class "TPDsp", generated with the TPDs function, containing the TPDs of the considered populations or species.

Value

REND returns a list with an element for each of the provided parameters (ie. communities and/or populations/species). These lists contain in turn one element for the Functional Richness of each unit, one for Functional Evenness, and one for Functional Divergence.

References

Mason, NWH, Mouillot, D, Lee, WG and Wilson, JB (2005), Functional richness, functional evenness and functional divergence: the primary components of functional diversity. *Oikos*, 111: 112–118.

Examples

```
# 1.  Compute the TPDs of five different species. SP3 is in the center of
#   the trait space, and the rest of species in the corners
set.seed(1)
species_ex <- c(rep("SP1",20), rep("SP2",20), rep("SP3",20), rep("SP4",20),
             rep("SP5",20))
traits_ex <- data.frame(trait1 = c(rnorm(20, 10, 1),
                                   rnorm(20, 10, 1),
                                   rnorm(20, 15, 1),
                                   rnorm(20, 20, 1),
                                   rnorm(20, 20, 1)),
                        trait2 = c(rnorm(20, 10, 1),
                                   rnorm(20, 20, 1),
                                   rnorm(20, 15, 1),
                                   rnorm(20, 10, 1),
                                   rnorm(20, 20, 1)))
species_TPDs <- TPDs (species = species_ex, traits = traits_ex)
#2. Five different communities with different abundances of each species
abundances_ex <- matrix(c(0.05, 0.05, 0.8,  0.05, 0.05, # 1. Low divergence
                  0.9,  0,    0,    0,    0.1,  # 2. High divergence
                  0,    0,    1,    0,    0,    # 3. Low Richness
                  0.2,  0.2,  0.2,  0.2,  0.2,  # 4. High Evenness
                  0.8,  0.05, 0.05, 0.05, 0.05), # 5. Low Evenness
                  ncol = 5, byrow = TRUE, dimnames = list(paste0("Comm.",1:5),
                  unique(species_ex)))

example_TPDc <- TPDc (TPDs = species_TPDs, sampUnit = abundances_ex)
#3. Estimate functional richness, evenness and divergence
example_RicEveDiv <- REND (TPDc = example_TPDc)
```

TPD *TPD: Functional Diversity based on Trait Probability Density*

Description

TPD is a package that includes different methods, based on the computation of the trait probability densities of populations and species (TPDs), and its extension to communities (TPDc). These methods allow estimating several components of functional diversity and to partition it across scales (within and between populations or species, within and between communities, etc). The collection of methods constitute a unified framework that incorporates the underlying probabilistic nature of trait distributions in a uni- or multivariate trait space.

Author(s)

Carlos P Carmona, <perezcarmonacarlos@gmail.com>

TPDc *Trait Probability Density of Communities*

Description

TPDc computes the trait probability density functions (TPD) of communities, for single or multiple traits. A TPDc for each community is calculated based on the TPDs of the species (or populations) present in the community and their relative abundances. The TPDs of all species should have been calculated beforehand using the TPDs function.

Usage

```
TPDc(TPDs, sampUnit)
```

Arguments

TPDs	An object of class "TPDsp" calculated with the function TPDs, containing the TPDs of all the species or populations present in the communities whose TPDc is going to be computed.
sampUnit	A matrix or data.frame containing the abundances of each species (in columns) in each community (in rows). The names of the species must be given in 'colnames(sampUnit)', whereas the names of the species must appear in 'rownames(sampUnit)'. Species names must match the names of the species used to build the TPDs object provided. In addition, in the cases in which there are different populations of the same species, the names of the communities must also match the names of the 'samples' argument provided to TPDs. In case that there is some combination of species x community with abundance greater than 0 that is not present in 'TPDs', the function will fail. NA values are not allowed in the matrix nor in the row or column names.

Value

TPDc returns an object of class "TPDcomm", which is a list containing the following components:

data: A list containing information used to perform the calculations, including the coordinates –in trait space– in which the TPD function has been evaluated, the volume –in trait units– of each cell of the grid, the length of each edges of the cells of the grid, the original trait data, the names of the species, the name of the populations in case sample is not NULL, the alpha level specified by the user, the traits of the individuals of each population, the type of TPDs calculates, which can be either "species" or "populations" depending on whether sample is or not NULL.

TPDc: A list containing information related with the TPDc of each community, including the species present in each community, the abundance of those species in each community, the abundance-rescaled TPDs of each species in each community, and the TPDc of each community, which is the probability associated to each cell of the grid in in which the trait space has been divided.

Examples

```
# 1.  Compute the TPDs of three different species
traits_iris <- iris[, c("Sepal.Length", "Sepal.Width")]
sp_iris <- iris$Species
example_TPDs <- TPDs(species = sp_iris, traits = traits_iris)

#2. Three different communities with different abundances of each species
example_abundances <- matrix(c(c(0.5, 0.3, 0.2,
                                 0.1, 0.8, 0.1,
                                 0.5, 0,   0.5)), #I. virg. dominates; setosa absent
                         ncol = 3, byrow = TRUE, dimnames = list(paste0("Comm.",1:3),
                         unique(iris$Species)))
example_TPDc <- TPDc (TPDs = example_TPDs, sampUnit = example_abundances)
```

TPDs *Trait Probability Density of Populations*

Description

TPDs computes the trait probability density functions (TPD) of populations. A TPD for each population is calculated using kernel density estimators around each trait value provided TPDs can be used for single or multiple traits (up to four traits at the present time).

Usage

```
TPDs(species, traits, samples = NULL, weight = NULL, alpha = 0.95,
  trait_ranges = NULL, n_divisions = NULL, tolerance = 0.05)
```

Arguments

species	A vector containing the names of the species whose TPD is to be calculated. It must have the same length than 'traits'; therefore, repeated names are allowed. NA values are not allowed.
traits	A matrix or data.frame containing the trait values of each individual of each species. Individuals are in rows and trait values in columns, with one column for each trait. NA values are not allowed.
samples	A vector containing the identity of the sampling unit in which each individual was present. Defaults to NULL, in which case, a TPDs is calculated for each species. If it is not NULL, a TPDs is calculated for each combination of 'species x sampling unit' (ie, a TPDs for each population). NA values are not allowed.
weight	A vector containing the weights to apply to each observation. It can be useful when individuals differ in their biomass. Defaults to NULL, in which case, all individuals are given the same weight.
alpha	A number between 0 and 1, indicating the proportion of the probability density function of each population to include. A value of 1 includes the whole density function, but may be sensitive to the presence of outliers. Defaults to 0.95.

`trait_ranges` A vector or a list indicating the range of trait values that will be considered in the calculations. If a vector is provided, each element should indicate the percentage (0-Inf) by which the range of each trait should be expanded in each direction. If a list is provided, it should contain the range (minimum and maximum) of trait values that will be considered. Each element of the vector or list corresponds with one trait. The order of the traits must be the same as the order of the columns in the 'traits' arguments. Defaults to NULL, in which case, the ranges of all the traits are automatically calculated by expanding the range of the columns in 'traits' by 15% in each direction. Trait ranges that are too short may result in an inadequate characterization of TPDs.

`n_divisions` The number of equal-length parts in which each trait should be divided to calculate the grid in which calculations are based. Note the number of cells composing the grid increases exponentially as dimensionality increases, which can result in high computation times. Defaults to NULL, in which case one trait is divided into 1000 parts, two traits are divided into 200 parts (40,000 cells), three traits are divided into 50 parts (125,000 cells), and 4 traits are divided into 25 parts (390,625 cells).

`tolerance` A number between 0 and 1, giving the admissible proportion of deviation from 1 in the integral of the TPDs of each population. Integrals can be lower than 1 when the extent of the evaluation grid is not enough to capture all the probability density function of the species. These problems are usually solved by increasing 'trait_ranges'. When the absolute deviation is greater than 'tolerance', a warning message is produced, but the TPDs function does not fail. Defaults to 0.05.

Value

`TPDs` returns an object of class "TPDsp", which is a list containing the following components:

data: A list containing information used to perform the calculations, including the coordinates –in trait space– in which the TPD function has been evaluated, the volume –in trait units– of each cell of the grid, the length of each edges of the cells of the grid, the original trait data, the names of the species, the name of the populations in case sample is not NULL, the alpha level specified by the user, the traits of the individuals of each population, and the type of TPDs calculated.

TPDs: A list, with one element per species or population, containing the probability associated to each cell of the grid in which the trait space has been divided.

Examples

```
# 1.  Compute the TPDs of three different species
traits_iris <- iris[, c("Sepal.Length", "Sepal.Width")]
sp_iris <- iris$Species
example_sp <- TPDs(species = sp_iris, traits = traits_iris)

# 2.  Two different populations of each species
samples_ex <- rep(c(rep(1, 25), rep(2, 25)), 3)
example_pop <- TPDs (species = sp_iris, traits = traits_iris,
     samples = samples_ex)
```

TPDsMean *Creating TPDs without individual observations*

Description

TPDsMean estimates the TPDs of species using the mean trait values and covariance matrix of traits. It is most useful when there is no trait information at the individual level, but the mean and variance (and optionally covariance) of traits are known.

Usage

```
TPDsMean(species, means, sds, covar = FALSE, alpha = 0.95,
  samples = NULL, trait_ranges = NULL, n_divisions = NULL,
  tolerance = 0.05)
```

Arguments

species	A vector containing the names of the species whose TPD is to be calculated. The length of 'species' must match the number of rows of 'traits', and the length of 'sigmas'. NA values are not allowed.
means	A matrix or data.frame containing the average trait values of each species (or population). Species are in rows and traits in columns, with one column for each trait. NA values are not allowed.
sds	A matrix or data.frame containing the standard deviation of trait values of each species (or population). Species are in rows and traits in columns, with one column for each trait. NA values are not allowed.
covar	Logical; if TRUE, the covariance between traits is calculated using the mean trait values and considered in the construction of the multivariate normal distributions. Defaults to FALSE.
alpha	A number between 0 and 1, indicating the proportion of the probability density function of each population to include. A value of 1 includes the whole density function, but may be sensitive to the presence of outliers. Defaults to 0.95.
samples	A vector containing the identity of the sampling unit in which each individual was present. Defaults to NULL, in which case, a TPDs is calculated for each species. If it is not NULL, a TPDs is calculated for each combination of 'species x sampling unit' (ie, a TPDs for each population). NA values are not allowed.
trait_ranges	A vector or a list indicating the range of trait values that will be considered in the calculations. If a vector is provided, each element should indicate the interval (in number of standard deviations) by which the range of each trait should be expanded in each direction. In that case, an interval of n standard deviations is calculated around each species mean trait value, and the absolute maximum and minimum across all species are selected as the range for each trait.If a list is provided, it should contain the range (minimum and maximum) of trait values that will be considered.Each element of the vector or list corresponds with one trait. The order of the traits must be the same as the order of the columns in the

'traits' arguments. Defaults to NULL, in which case, the ranges of all the traits are automatically calculated by expanding the range of the columns in 'means' by 5 times the values in 'sds' in each direction. Trait ranges that are too short may result in an inadequate characterization of TPDs.

`n_divisions` The number of equal-length parts in which each trait should be divided to calculate the grid in which calculations are based. Note the number of cells composing the grid increases exponentially as dimensionality increases, which can result in high computation times. Defaults to NULL, in which case one trait is divided into 1000 parts, two traits are divided into 200 parts (40,000 cells), three traits are divided into 50 parts (125,000 cells), and 4 traits are divided into 25 parts (390,625 cells).

`tolerance` A number between 0 and 1, giving the admissible proportion of deviation from 1 in the integral of the TPDs of each population. Integrals can be lower than 1 when the extent of the evaluation grid is not enough to capture all the probability density function of the species. These problems are usually solved by increasing 'trait_ranges'. When the absolute deviation is greater than 'tolerance', a warning message is produced, but the TPDsMean function does not fail. Defaults to 0.05.

Value

`TPDsMean` returns an object of class "TPDsp", which is a list containing the following components:

data: A list containing information used to perform the calculations, including the coordinates –in trait space– in which the TPD function has been evaluated, the volume –in trait units– of each cell of the grid, the length of each edges of the cells of the grid, the original trait data (means and sds matrices), the names of the species, the alpha level specified by the user, and the type of TPDs calculated.

TPDs: A list, with one element per species or population, containing the probability associated to each cell of the grid in which the trait space has been divided.

Examples

```
# 1.  Compute the TPDs of three different species (1 dimension)
sp_ex <- unique(iris$Species)
mt1 <- tapply(iris[, "Sepal.Length"], iris$Species, mean)
means_ex <- matrix(c(mt1), ncol=1)
st1 <- tapply(iris[, "Sepal.Length"], iris$Species, sd)
sds_ex <- matrix(c(st1), ncol=1)
TPDs_iris<- TPDsMean(species = sp_ex, means = means_ex, sds = sds_ex)

# 2.  Compute the TPDs of three different species (2 dimensions)
sp_ex <- unique(iris$Species)
mt1 <- tapply(iris[, "Sepal.Length"], iris$Species, mean)
mt2 <- tapply(iris[, "Sepal.Width"], iris$Species, mean)
means_ex <- matrix(c(mt1, mt2), ncol=2)
st1 <- tapply(iris[, "Sepal.Length"], iris$Species, sd)
st2 <- tapply(iris[, "Sepal.Width"], iris$Species, sd)
sds_ex <- matrix(c(st1, st2), ncol=2)
TPDs_iris<- TPDsMean(species = sp_ex, means = means_ex, sds = sds_ex)
```

```
# 3.  Two different populations of each species
samples_aux <- rep(c(rep(1, 25), rep(2, 25)), 3)
sp_ex <- rep(unique(iris$Species), each=2)
mt1 <- tapply(iris[, "Sepal.Length"], (paste0(iris$Species,samples_aux)), mean)
mt2 <- tapply(iris[, "Sepal.Width"], (paste0(iris$Species,samples_aux)), mean)
means_ex <- matrix(c(mt1, mt2), ncol=2)
st1 <- tapply(iris[, "Sepal.Length"], (paste0(iris$Species,samples_aux)), sd)
st2 <- tapply(iris[, "Sepal.Width"], (paste0(iris$Species,samples_aux)), sd)
sds_ex <- matrix(c(st1, st2), ncol=2)
samples_ex<- rep(c("Comm.1","Comm.2"),3)
TPDs_iris_pop <- TPDsMean (species = sp_ex, means = means_ex, sds = sds_ex,
   samples = samples_ex)
```

`tSamp` *Trait Values Sampling*

Description

`tSamp` samples (with replacement) trait values from populations, species or communities. The probability of sampling each trait –or combination of traits in multidimensional cases– is proportional to the value of TPDs or TPDc for the corresponding cell (the trait space is divided in a grid composed of cells, see `TPDs` for further information).

Usage

```
tSamp(TPDc = NULL, TPDs = NULL, size = 1)
```

Arguments

`TPDc`	An object of class "TPDcomm", generated with the `TPDc` function, containing the TPDc of the considered communities.
`TPDs`	An object of class "TPDsp", generated with the `TPDs` function, containing the TPDs of the considered populations or species.
`size`	Non-negative integer giving the number of observations to choose. Defaults to 1.

Value

`tSamp` returns a list containing sampled trait values for each community of TPDc or species/populations from TPDs.

Examples

```
# 1.  Compute the TPDs of three different species
traits_iris <- iris[, c("Sepal.Length", "Sepal.Width")]
sp_iris <- iris$Species
example_TPDs <- TPDs(species = sp_iris, traits = traits_iris)

#2. Three different communities with different abundances of each species
example_abundances <- matrix(c(c(0.5, 0.3, 0.2,
                                 0.1, 0.8, 0.1,
                                 0.5, 0,   0.5)), #I. virg. dominates; setosa absent
                         ncol = 3, byrow = TRUE, dimnames = list(paste0("Comm.",1:3),
                         unique(iris$Species)))
example_TPDc <- TPDc (TPDs = example_TPDs, sampUnit = example_abundances)

#3. Sample 1,000 trait values from each species and community
example_sampling <- tSamp(TPDc = example_TPDc, TPDs = example_TPDs,
                                 size = 1000)
```

`uniqueness` *Functional Uniqueness of Ecological Units*

Description

`uniqueness` estimates the functional uniqueness of species, communities by comparing the TPD of lower levels (i.e. species), with that of higher levels (i.e. communities). TPD's are compared by means of overlap. High overlap means low uniqueness (i.e. the species traits are frequent in the community), whereas low overlap means high uniqueness. Uniqueness is then estimated as 1-overlap. The function is hence basically the same as 'dissim', with some slight modifications. Despite functional uniqueness can be estimated at any scale, current implementation is limited to species within communities (although communities can be easily created to represent regions, or regional pools of species).

Usage

```
uniqueness(TPDs = NULL, TPDc = NULL)
```

Arguments

`TPDs`	An object of class "TPDsp", generated with the `TPDs` function, containing the TPDs of the considered species
`TPDc`	An object of class "TPDcomm", generated with the `TPDc` function, containing the TPDc of the considered communities.

Value

`uniqueness` returns a matrix, with the communities in rows and the species in columns. The values in the matrix represent the functional uniqueness of each species in each community. Very unique species will have values close to 1, whereas non-unique species will have values close to 0.

Examples

```
# 1.  Compute the TPDs of three different species
traits_iris <- iris[, c("Sepal.Length", "Sepal.Width")]
sp_iris <- iris$Species
example_TPDs <- TPDs(species = sp_iris, traits = traits_iris)

#2. Three different communities with different abundances of each species
example_abundances <- matrix(c(c(0.5, 0.3, 0.2,
                                 0.1, 0.8, 0.1,
                                 0.5, 0,   0.5)), #I. virg. dominates; setosa absent
                         ncol = 3, byrow = TRUE, dimnames = list(paste0("Comm.",1:3),
                         unique(iris$Species)))
example_TPDc <- TPDc (TPDs = example_TPDs, sampUnit = example_abundances)

#3. Calculate the uniqueness of each species in each community
example_uniqueness <- uniqueness (TPDs = example_TPDs, TPDc = example_TPDc)
```

Index

dissim, 2, *5*

plotTPD, 3

Rao, 5
redundancy, 6
REND, 7

TPD, 8
TPD-package *(*TPD*)*, 8
TPDc, *2, 3*, *5–7*, 9, *14, 15*
TPDs, *2, 3*, *7*, *9*, 10, *14, 15*
TPDsMean, *2*, 12
tSamp, 14

uniqueness, 15