Package 'CommEcol'

July 21, 2025

Type Package

Title Community Ecology Analyses

Version 1.8.1

Date 2024-06-18

Depends vegan, rncl

Imports ape, picante, adespatial, betapart, gmp

Description Autosimilarity curves, standardization of spatial extent, dissimilarity indexes that overweight rare species, phylogenetic and functional (pairwise and multisample) dissimilarity indexes and nestedness for phylogenetic, functional and other diversity metrics. The methods for phylogenetic and functional nestedness is described in Melo, Cianciaruso and Almeida-Neto (2014) <doi:10.1111/2041-210X.12185>. This should be a complement to available packages, particularly 'vegan'.

License GPL-2

NeedsCompilation no

Maintainer Adriano Sanches Melo <asm.adrimelo@gmail.com>

Author Adriano Sanches Melo [aut, cre]

Repository CRAN

LazyLoad yes

Date/Publication 2024-06-28 09:50:02 UTC

Contents

`CommEcol-package` *Community Ecology Analyses*

Description

Community Ecology Analyses.

Details

Package:	CommEcol
Type:	Package
Version:	1.8.1
Date:	2024-06-18
License:	GPL-2
LazyLoad:	yes

Community Ecology Analyses. This should be a complement to available packages, particularly vegan.

Author(s)

Adriano Sanches Melo
Maintainer: Adriano Sanches Melo <asm.adrimelo@gmail.com>

autosimi *Autosimilarity curve of species community data*

Description

Similarities among two subsamples, each one obtained randomly from the same community dataset. Curves are otained for all subsample sizes from 1 up to half the number of sample units in the dataset. Autosimilarity curves can be used to evaluate sample suficiency when sample size is expressed as number of sampling units such as traps or quadrats. This is particularly suitable when the study involves similarities or dissimilarities among samples such as agglomerative clustering, ordination, Mantel test.

Usage

```
autosimi(comm, method="bray", binary=FALSE, log.transf=FALSE, simi=TRUE, permutations=50)
```

Arguments

`comm`	Dataframe or matrix with samples in rows and species in columns.
`method`	Similarity index obtained from `vegdist`. Similarities are obtained simply as 1-dissimilarity. Accordingly, it only makes sense for indices bounded at 0-1. For indices not bounded at 0-1 (distances), use `simi=FALSE` to obtain autodissimilarity curves.
`binary`	Should data be transformed to presence/absence?
`log.transf`	Transformation `log(x+1)` before calculation of similarities (or dissimilarities)
`simi`	Similarity or dissimilarity curve.
`permutations`	Number of curves randomly calculated and from which the mean autosimilairty curve is obtained.

Details

The function selects randomly and without replacement an even number of sampling units (n = 2,4,6, ...) from the total sampling units. Next, the first n/2 sampling units are pooled (summed) to create a subsample, and the other n/2 sampling units to create another subsample. If `binary=TRUE`, the two subsamples are transformed to presence/absence. If `log.transf=TRUE`, `log(x+1)` of each value is obtained. The similarity between the two subsamples is calculated and stored. The procedure is repeated for larger subsample sizes until half the size of the full dataset (or up to the integer quotient in the case of odd numbers of sample units). The procedure is then repeated for the requested number of curves. The output is the average curve. This function is a different implementation of the procedures described in Schneck & Melo (2010).

Value

A dataframe containing subsample sizes and average similarity values.

Author(s)

Adriano Sanches Melo

References

Cao, Y., D.P. Larsen & R.M. Hughes. 2001. Evaluating sampling sufficiency in fish assemblage surveys: a similarity-based approach. Canadian Jounal of Fisheries and Aquatic Sciences 58: 1782-1793.

Schneck, F. & A.S. Melo. 2010. Reliable sample sizes for estimating similarity among macroinvertebrate assemblages in tropical streams. Annales de Limnologie 46: 93-100.

Weinberg, S. 1978. Minimal area problem in invertebrate communities of Mediterranean rocky substrata. Marine Biology 49: 33-40.

See Also

vegdist

Examples

```
x<-matrix(0,4,4)
diag(x)<-1
x4<-rbind(x,x,x,x)
x4
autosimi(x4, binary=TRUE)
plot(autosimi(x4, binary=TRUE))

data(BCI)
simi<-autosimi(BCI, binary=TRUE, permutations=5)
simi
plot(simi, ylim=c(0.5,1)) # maintain the plot window open for the next curve
simi.log<-autosimi(BCI, binary=FALSE, log.transf=TRUE, permutations=5)
points(simi.log, col="red")
```

---

betaRegDisp *Beta diversity metrics between sites in a moving window along environmental gradients*

Description

The function computes eight metrics of beta diversity according to an informed environmental gradient. It selects a given number of environmentally-neighborhood sites in a moving window to obtain beta diversity.

Usage

```
betaRegDisp(y, x, xy.coords = NULL, ws = 3,
            method.1 = "jaccard", method.2 = "ruzicka",
            method.3 = "ruzicka",
            independent.data = FALSE, illust.plot = FALSE)
```

Arguments

y	Response matrix, where rows are sites and columns are species.
x	Predictor vector. A vector of the environmental gradient under study with the same number of sites as in matrix y. Make sure the order of samples in y and x is the same.
xy.coords	Geographical coordinates. A matrix with two columns of XY decimal degree geographical coordinates, which are used to compute euclidean distance among sites. Rows must be sites in the same order as in y and x.
ws	Window size or number of sites to be used in the computation of the distinct beta-diversity metrics or between-site dissimilarities. It must be a positive integer higher than 2.
method.1	For beta-diversity metrics 1 to 3 (see details). A dissimilarity index available in the vegdist. The options are: "euclidean", "jaccard", "bray", "manhattan", "canberra", "kulczynski", "gower", "morisita", "horn", "mountford", "raup", "binomial", "chao", "altGower", "cao", "mahalanobis".
method.2	For beta-diversity metrics 4 and 5 (see details). A dissimilarity index available in the beta.div. The options are: "hellinger", "chord", "log.chord", "chisquare", "profiles", "percentdiff", "ruzicka", "divergence", "canberra", "whittaker", "wishart", "kulczynski", "jaccard", "sorensen", "ochiai", "ab.jaccard", "ab.sorensen", "ab.ochiai", "ab.simpson", "euclidean".
method.3	For beta-diversity metrics 6, 7, and 8 (see details). A multisample dissimilarity index available in the beta.multi.abund. The options are: "ruzicka" and "bray".
independent.data	Should windows not superpose each other? If independent.data=TRUE, sites do not enter more than once in the dissimilarity calculations, that is, sites are included in a single window.
illust.plot	Should a window plot be open and illustrate how the window moves along the gradient?

Details

The function computes eight beta-diversity metrics among sites included in a set (window) of length ws. See details in Dala-Corte et al. (2019).

Metrics 1-3 uses dissimilarity indices available in vegdist:

1. Mean pair-wise dissimilarity between sites in a window;

2. Mean dissimilarity between focal site and the other sites in a window. If an odd number is informed in ws, the focal site is the central site in relation to its neighbours in the window. If an even number is informed in ws, the focal site is the first site in the window;

3. Mean distance of sites to their group centroid in a Principal Coordinate (PCoA) space computed using betadisper;

Metrics 4-5 uses dissimilarity indices available in beta.div:

4. Total sum of squares (SS) of the window sites (Legendre and De Caceres, 2013);

5. Local contributions to beta diversity (LCBD; Legendre and De Caceres, 2013);

Metrics 6-8 uses dissimilarity indices available in `beta.multi.abund`:

6. Total multiple-site dissimilarities for a selected window of sites;

7. Nestedness component of multiple-site dissimilarities for a selected window of sites;

8. Turnover component of multiple-site dissimilarities for a selected window of sites.

Value

A matrix with 10 columns (or 12 if `xy.coords` is informed). Values in columns are sorted according to the enviromental gradient, from the lowest to the highest value. Columns correspond to:

1. `grad` - The environmental gradient (predictor vector, `x`);

2. `mean.grad` - Mean value of the environmental gradient of sites selected in each window;

3. `mean.diss.pairs` - Mean pair-wise dissimilarity between sites in a selected window (metric 1);

4. `diss.focal` - Mean dissimilarity between focal site and the other sites (metric 2);

5. `mean.dist.cent` - Mean distance of sites to their group centroid in a Principal Coordinate (PCoA) space (metric 3);

6. `SS.group` - Total sum of squares (SS) of the sites in a window (metric 4);

7. `SS.focal` - Local contributions to beta diversity (LCBD), which represents how much a focal site contributed to the total window SS;

8. `beta.TOT` - Total multiple-site dissimilarity;

9. `beta.NES` - Nestedness component of multiple-site dissimilarity;

10. `beta.TUR` - Turnover component of multiple-site dissimilarity;

11. `mean.geodist` - If `xy.coords` is provided, the mean linear euclidean distance between sites in the a window is returned.

12. `focal.geodist` - If `xy.coords` is provided, the mean linear euclidean distance of the focal site in relation to its neighbours in the window is returned.

Author(s)

Luciano F. Sgarbi, Renato B. Dala-Corte and Adriano S. Melo

References

Anderson, M.J., K.E. Ellingsen and B.H. McArdle. 2006. Multivariate dispersion as a measure of beta diversity. Ecology Letters 9: 683-693.

Baselga, A. 2010. Partitioning the turnover and nestedness components of beta diversity. Global Ecology and Biogeography 19: 134-143.

Baselga, A. 2017. Partitioning abundance-based multiple-site dissimilarity into components: balanced variation in abundance and abundance gradients. Methods in Ecology and Evolution 8: 799-808.

Dala-Corte, R.B., L.F. Sgarbi, F.G. Becker and A.S. Melo. 2019. Beta diversity of stream fish communities along anthropogenic environmental gradients at multiple spatial scales. Environmental Monitoring and Assessment 191:288.

Legendre, P. and M. De Caceres. 2013. Beta diversity as the variance of community data: dissimilarity coefficients and partitioning. Ecology Letters 16: 951-963.

See Also

vegdist, betadisper, beta.div, beta.multi

Examples

```
## Example 1. A simmulated community matrix with a known structure of increasing
##  beta diversity by turnover
# n is the total sample sites
# LocS is the number of spp per site
# MaxS is the total number of spp in the matrix

# All samples will contain LocS species. The first sample will contain presences
# for the first LocS species. The subsequent samples will contain LocS presences
# spread over a increasing set of species. The assignment of presences for the
# second sample to the last sample is done randomly. The last sample will
# contain LocS presences assigned randomly to the MaxS species. Thus, for a
# window size of 3 (ws=3) and a dataset of 10 samples, beta diversity for the
# samples 1-3 will be much lower than for samples 8-10.

SimComm <- function(n = 19, MaxS = 24, LocS = 8){
    s <- seq (LocS, MaxS, length.out = n)
    mat <- matrix(0, n, MaxS, dimnames =
                      list(paste("site", 1:n, sep = "_"),
                           paste("sp", 1:MaxS, sep = "_")))
    for(i in 1:n){
        mat[i, sample(1:s[i], LocS)] <- 1
    }
    mat <- mat[, colSums(mat)!=0]
    return(mat)
}

mat <- SimComm(n = 19, MaxS = 24, LocS = 8)

#Creating an environmental gradient:
grad <- 1:nrow(mat)

b.resu <- betaRegDisp(y = mat, x = grad, xy.coord = NULL, ws = 3,
                      method.1 = "jaccard",
                      method.2 = "ruzicka",
                      method.3 = "ruzicka",
                      independent.data = FALSE, illust.plot = FALSE)

##Ploting all the output of the object for the simmulated community
op <- par(no.readonly = TRUE)
par(mfrow = c(5, 2), oma = c(1, 0, 1, 0.1), mar = c(1.5, 3, .1, .1), cex = 1, las = 0)
for(i in 1:ncol(b.resu)){
  plot(b.resu[, 1], b.resu[, i], ylab = colnames(b.resu)[i], cex.lab = .9,
       cex.axis = 0.9, tcl = -0.2, mgp = c(1.5, .2, 0), pch = 15, col = "grey")
}
mtext("Environmental gradient", cex = 1.3, 1, -0.1, outer = TRUE)
par(op)
```

```
##Example 2
data(varespec)
data(varechem)
grad <- varechem[, "Baresoil"]
resu <- betaRegDisp(y = varespec, x = grad, ws = 3, method.1 = "jaccard",
            method.2 = "ruzicka", method.3 = "ruzicka",
            independent.data = FALSE, illust.plot = FALSE)

#Plotting all the outputs of the function:
op <- par(no.readonly = TRUE)
par(mfrow = c(5, 2), oma = c(1, 0, 1, 0.1), mar = c(1.5, 3, .1, .1), cex = 1, las = 0)
for(i in 1:ncol(resu)){
  plot(resu[, 1], resu[, i], ylab = colnames(resu)[i], cex.lab = .9,
       cex.axis = 0.9, tcl = -0.2, mgp = c(1.5, .2, 0), pch = 15, col = "grey")
}
mtext("Environmental gradient", cex = 1.3, 1, 0, outer = TRUE)
par(op)
```

---

compas *Simulation of species community data along gradients*

Description

The function generates unimodal curves of different shapes according to the specification of different parameter values. For different combinations of parameters, the response function may resemble the Gaussian curve or its skewed and platykurtic variations. Simulated observations are obtained by generating values for parameters and solving the function for defined x-values. In an ecological context, the response curve may be interpreted as densities of a species along an environmental gradient. Solving the function for a given x-value would be equivalent to sampling the species at the coordinate x. If many different curves are generated, solving the function for a given x-value would be equivalent to sampling a community at the coordinate x of the gradient. The idea is easily expanded to include two or more gradients or dimensions. For the case of two gradients, two sets of coordinates, one for each dimension, are used to obtain a community observation. A theory justifying the use of response curves along gradients and a discussion of the shapes of these curves is found in McGill & Collins (2003). The code is mostly based on Minchin (1987a, 1987b).

Usage

```
compas(S, dims, am, clump=1, beta.R, coords, n.quanti=5, n.quali=0.1, add1)
```

Arguments

S	The number of species occurring in the simulated gradients. This IS NOT necessarily the number of species that will appear in the resulting dataset as some species may not be "sampled".
dims	Number of gradients (dimensions).

am	A vector of abundance of species in its modal point (log scale) for each gradient. This(ese) value(s) is(are) used to sample a lognormal distribution with mean(s) am and sd=1. The number of supplied means should be the same of the number of dims.
clump	A non-zero positive integer indicating how strong the species richness gradient is. For clump=1, modal points for species (see Details) are randomly and uniformly distributed in the space and thus species richness are mostly homogeneous. For higher values, more modal points of species (and thus more species) will be clumped on higher values of the gradient(s) (the upper right-hand corner if dims=2. See example.
beta.R	Beta diversity (turnover) parameter.
coords	A matrix-like (or vector if dims=1) object with coordinates of sampling sites. The number of columns (axes) should be the same of the number of dims.
n.quanti	A value higher than 0 indicating how much noise or variability should be added to abudance values sampled from the response curves. Each abundance value is substituted by a random value obtained from a Negative Binomial distribution with mean equal to the respective abundance value and variance proportional to n.quanti.
n.quali	Qualitative Noise. Each specie has probability "1-n.quali" of occurring in a site within its range. The argument control the replacement of n.quali*100% of the abundance values by "0".
add1	A value between 0 and 1. Add (add1*100)% of "marginal/vagrant species" occurring randomly with 1 individual in the entire dataset.

Details

This implementation is based on the software Compas (Minchin 1987b), described in detail in Minchin (1987a). Simulated parameters are random values obtained from distributions such as the normal and the uniform. Some of these parameters can be modified by users. However, some them are fixed and not modified unless you are able to edit the code. The option to fix some of the parameters should simplify the use of the function.

The number of species in the simulated matrix may be smaller than S because some species may occur outside the gradient. The gradiente is -50 up to 150, but 'sampling' occurs only at the range 0-100. This allows species to have their mode outside the 0-100 gradient. Also, species occurring in the gradient may not be sampled as a result of the quantitative (n.quanti) and qualitative (n.quali) noises added.

Parameters alpha and gamma (not available as arguments), which together determine curve symmetry and kurtosis, are obtained from a uniform distribution bounded by 0.1 and 5.

The abundance at the mode of the curve is determined by random values otained from log-normal distribution with log(mean) am and sd=1 (the last one not available as argument).

The position of the modal point in the gradient (parameter m, not available as argument) will depend on the value of clump. If clump=1 coordinates of the modal points are obtained from a random uniform distribution. For higher values of clump, species modal points will tend to be concentrated on high values of the gradient. In all cases, coordinates are bounded by -50 and +150. For a studied gradient bounded by 0 and 100, the specification of a larger interval allows the position of the modal point to be located outside the gradient.

The parameter beta.R determines the range of occurrence of species in the gradient. In terms of diversity, it determines the turnover (beta diversity) along the gradient, and is expressed as beta.R = 100/mean(r), and r are the ranges of species. Values of r are obtained from a normal distribution with mean 100/beta.R and standard deviation of 0.3*100/beta.R (the last one not available as argument). A beta.R = 2 determines that range of species are, on average, 50 units of the gradient (and 15 SD units). High beta.R determines restricted ranges and, thus, high beta diversity. Old versions of this function included the argument "beta" and expressed, contrary to the name of the argument, as range in the gradient. The code of the new function is equivalent to the old ones (beta=0.5 was translated as 0.5*100 and determined ranges around 50; this can be attained now as beta.R=2 which determines ranges 100/2=50).

Species may be (i) present in densities different from the expected or (ii) absent due to sampling error, historical factors, or the influence of a multitude of environmental factors. Simulated communities are subject to these two types of error or noise to mimic the two phenomena. In order to obtain scattered values of abundances for each species along their gradients, each non-zero value is replaced by a random value obtained from a Negative Binomial distribution (rnbinom) with mean equal to the value to be replaced and variance proportional to n.quanti. The parametrization of the Negative Binomial is done using mu (mean) and size (dispersion parameter), the later obtained as mu^2 / (variance-mu). The n.quanti is used to define the variance as mu*(1+n.quanti). For n.quanti tending to zero the distribution is similar to the Poisson. For values up to 1 it is still similar to the Poisson. For n.quanti >5 it is quite distinct with long right-hand tails and many zeros if the mean is low (e.g. <10). Additional absences are achieved by randomly choosing n.quali*100% of non-zero values and replacing them by zeros.

A third characteristic commonly seen in field datasets is the occurrence of a species outside its regular range or in different habitats. These species are termed vagrant or marginal, and usually appear in the dataset with 1 individual. In order to account for this common finding, the function attaches to the simulated community an additional set of species with 1 individual, each species occurring in one sample unit only. The number of these additional species is set as add1*100% of the number of species sampled in the simulated communities after the inclusion of the two types of errors cited above.

Value

A dataframe of sites (rows) by species (cols) abundances. In case of dims=1, a plot of response species curves is produced. These curves do not include quantitative noise (n.quanti), qualitative noise (n.quali) (see comments above) and 'marginalor vagrant species' (add1) additions. Notice that the number of species in the result dataframe may vary as (i) some species may not be sampled by coords, or (ii) species range is located outside the sampling gradient (0-100).

Note

As the function includes many parameters, many simulated communities will not mimic the real ones. Users should try different set of options to approxiamte real datasets. A starting point may be the examples provided below.

Author(s)

Adriano Sanches Melo

References

McGill, B. & C. Collins. 2003. A unified theory for macroecology based on spatial patterns of abundance. Evolutionary Ecology Research 5: 469-492.

Minchin, P.R. 1987a. Simulation of multidimensional community patterns: towards a comprehensive model. Vegetatio 71: 145-156.

Minchin, P.R. 1987b. COMPAS: a program for the simulation of multidimensional community patterns based on generalized beta functions. Dep. of Biological Sciences, Southern Illinois University Edwardsville, USA.

See Also

vegdist

Examples

```
# 1 dimension.
coo <- seq(10, 90, 10)
compas(S=30, dims=1, am=2, beta.R=2, coords=coo,
       n.quanti=5,  n.quali=0.1, add1=0.1)

# 2 dimensions.
coo2 <- cbind(coo, coo)
compas(S=50, dims=2, am=c(2,2), beta.R=c(1,1), coords=coo2,
       n.quanti=5, n.quali=0.1, add1=0.1)

# 2 dimensions. Homogeneous and clumped species distributions.
# Try a few times.
coo.grid <- expand.grid(seq(10,90,10), seq(10,90,10))
plot(coo.grid, xlim=c(0,100), ylim=c(0,100))

mat1 <- compas(S=60, dims=2, am=c(2,2), clump=1, beta.R=c(2,2), coords=coo.grid,
               n.quanti=5, n.quali=0.1, add1=0.1)
S1 <- rowSums(ifelse(mat1>0,1,0))

mat2 <- compas(S=60, dims=2, am=c(2,2), clump=3,
               beta.R=c(2,2), coords=coo.grid, n.quanti=5, n.quali=0.1, add1=0.1)
S2 <- rowSums(ifelse(mat2>0,1,0))

ind <- coo.grid/10
S1.mat <- matrix(NA, 9, 9)
S2.mat <- matrix(NA, 9, 9)
for(i in 1:length(S1)){
      S1.mat[ind[i,1], ind[i,2]] <- S1[i]
      S2.mat[ind[i,1], ind[i,2]] <- S2[i]
   }

plot(coo.grid, cex=S1/4)
plot(coo.grid, cex=S2/4)

image(x=coo, y=coo, z=S1.mat, col=gray(50:1/50))
image(x=coo, y=coo, z=S2.mat, col=gray(50:1/50))
```

```
filled.contour(x=coo, y=coo, z=S1.mat)
filled.contour(x=coo, y=coo, z=S2.mat)
```

`dis.chao` *Chao et al. dissimilarity indices*

Description

Dissimilarity indices proposed by Chao et al. (2005) that takes into account (rare) unseen shared species. This is done mostly by heavy-weighting shared rare species. The function handle abundance or frequency data and are available for the Jaccard and Sorensen family of indices. Probability versions of the index, that does not heavy-weight rare species, are also included.

Usage

```
dis.chao(comm, index="jaccard", version="rare", freq=NULL)
```

Arguments

`comm`	Dataframe or matrix with samples in rows and species in columns.
`index`	The index formula to be used in Chao dissimilarity. Partial match to "jaccard" or "sorensen".
`version`	The uncorrected probability version of the Chao index or the corrected version that takes into account unseen rare species. Partial match to "probability" or "rare".
`freq`	A numeric vector indicating total number of sampling units composing each sample (row) of the community data for computing incidence-based Chao indices. Length of `freq` must be the same as the number of samples in `comm`.

Details

Communities usually are composed of many rare and only a few common/frequent species. Although rare, a species may provide a valuable information in the estimation of resemblance between samples. However, the use of raw abundances makes the contribution of rare species to be negligible to the resulting index value. For instance, dropping a common species from the community data table usually will make much larger differences in the dissimilarity matrix than dropping a rare species.

There are a few indices that give more importance to an individual belonging to a rare than to a common/frequent species (see `dis.goodall` and `dis.nness`). Notice, however, that this differential weight of species can also be obtained, for instance, by log-transforming or standardizing data (e.g. dividing by maximum within each species) (Melo, in preparation). In this sense, an extreme case in which rare and common species have the same weight is in the use of presence/absence data.

The "probability" version of the abundance-based Chao-Jaccard and Chao-Sorensen indices are distinct from those traditional formulae available, for instance, in `vegdist` (Chao et al. 2005).

These probability indices do not overweight rare species and are included here for comparability. The version that takes into account unseeen rare species simplifies to the probability version if no rare species (singleton or doubleton for abundance data; unique or duplicate for incidence data) is present (see example below).

Value

A "dist" object.

Author(s)

Adriano Sanches Melo

References

Chao, A., R.L. Chazdon, R.K. Colwell & T. Shen. 2005. A new statistical approach for assessing similarity of species composition with incidence and abundance data. Ecology Letters 8: 148-159.

Melo, A.S. in preparation. Is it possible to improve recovery of multivariate groups by weighting rare species in similarity indices?

See Also

vegdist, dis.goodall, dis.nness.

Examples

```
library(vegan)
### Example 1: Rare species are heavier:
aa <- c(1, 2, 4, 5)
bb <- c(1, 2, 0, 5)
cc <- c(0, 2, 3, 3)
dat3 <- rbind(aa, bb, cc)
colnames(dat3) <- c("sp1", "sp2", "sp3", "sp4")
dat3

vegdist(dat3, method='jaccard', binary=FALSE)
# Notice dissimilarity between the pair aa-bb is the same of the pair aa-cc.
# In fact, bb and cc differ from aa in the same way (one species, and
# 4 exclusive individuals).

dis.chao(dat3, index="jaccard", version="prob")
# The probability version of the Chao index, however, produce different
# dissimilarities for the pairs aa-bb and aa-cc
# (aa-cc is less dissimilar than aa-bb).

dis.chao(dat3, index="jaccard", version="rare")
# The dissimilarity for the pair aa-cc is the same as that obtained using the
# probability version. However, the dissimilarity for the pair aa-bb decreased.
# The reason is that aa-bb shares two rare species (sp1, sp2),
# whereas the pair aa-cc shares a single rare species (sp2).
```

```
### Example 2: "rare" version of the Chao index simplifies to the
# "probability" version if no rare species (with 1 or 2 individuals) is present.
data(japi)
dim(japi)
# 75 sampling units (stones) and 66 morphospecies of stream macroinvertebrates.
japi.m <- as.matrix(japi)
japi.2 <- ifelse(japi.m==1, 3, japi.m)
# no singletons.
japi.3 <- ifelse(japi.2==2, 3, japi.2)
# no doubletons.

sort(
     dis.chao(japi.3, index='jac', version='rare')
   - dis.chao(japi.3, index='jac', version='prob'))


### Example 3: frequency data
# Stones in the japi dataset were sampled from downstream to upstream direction.
# Consecutive stones are spaced 1-6 m. The set of the first 25 stones should be
# more dissimilar to the last set of 25 stones than the middle set
# (simply because of spatial autocorrelation).
japi.pa <- ifelse(japi.m > 0, 1, 0)
japi.1st <- japi.pa[ 1:25, ]
japi.2nd <- japi.pa[26:50, ]
japi.3rd <- japi.pa[51:75, ]

japi.inc <- rbind(
                 colSums(japi.1st),
                 colSums(japi.2nd),
                 colSums(japi.3rd)
)
# species frequency of occurrence in the three sets of stones.

dis.chao(japi.inc, index="jaccard", freq=c(25, 25, 25) )
```

`dis.goodall` *Goodall dissimilarity index*

Description

This is a probability index in which rare species (or descriptors) are overweighted. Accordingly, sharing rare species makes pairs of samples (or objects) more similar than sharing common/frequent species (or descriptors). As Legendre & Legendre (1998) put it when describing this index: "... it is less likely for two sites to both contain the same rare species than a more frequent species. In this sense, agreement for a rare species should be given more importance than for a frequent species, when estimating the similarity between sites.".

Usage

```
dis.goodall(comm, p.simi="steinhaus", approach="proportion")
```

Arguments

comm	Dataframe or matrix with samples in rows and species in columns.
p.simi	The partial similarity index to be used in the Goodall index. Partial match to "steinhaus" (raw abundance data), or "gower" (normalized abundance data).
approach	The two approaches to compute Goodall index. Partial match to "proportion" or "chisquare".

Details

Communities are usually composed of many rare and only a few common/frequent species. Although rare, a species may provide a valuable information in the estimation of resemblance between samples. However, the use of raw abundances makes the contribution of rare species to be negligible to the resulting index value. For instance, dropping a common species from the community data table usually will make much larger differences in the dissimilarity matrix than dropping a rare species.

There are a few indices that give more importance to an individual belonging to a rare than to a common/frequent species (see dis.chao and dis.nness). Notice, however, that this differential weight of species can also be obtained, for instance, by log-transforming or standardizing data (e.g. dividing by maximum within each species) (Melo in preparation). In this sense, an extreme case in which rare and common species have the same weight is in the use of presence/absence data.

The implementation of Goodall index (Goodall 1966) follows Legendre & Legendre (1998, pp. 269-273). They suggest to use the "steinhaus" index for raw species abundance data and the "gower" partial similarity index (p.simi) for normalized data. The index can be calculated in two ways or approaches ("proportion" or "chisquare") and values produced by them should be very different, although this is mostly due to scale; they are monotonically correlated. Notice Legendre & Legendre (1998) present a similarity version of this index. The one produced by this function is a dissimilarity, computed simply as 1-similarity. See the examples below for some behaviors of the index.

One important issue made clear by Legendre & Legendre (1998, pp. 270-271) is that the dissimilarity value for a pair of sites depnds on other sites present in the data matrix. If changes are made to some sites, the resulting dissimilarity values for all remaining pairs are modified (see example below).

Value

A "dist" object.

Author(s)

Adriano Sanches Melo

References

Goodall, D.W. 1966. A new similarity index based on probability. Biometrics 22: 882-907.

Legendre, P & L. Legendre. 1998. Numerical Ecology. 2nd ed. Elsevier.

Melo, A.S. (in preparation) Is it possible to improve recovery of multivariate groups by weighting rare species in similarity indices?

See Also

vegdist, dis.chao, dis.nness.

Examples

```
library(vegan)
a <- c(1, 1, 0)
b <- c(2, 1, 0)
c <- c(0, 1, 1)
d <- c(0, 1, 2)
e <- c(0, 1, 3)
dat5 <- rbind(a,b,c,d,e)
colnames(dat5) <- c("sp1","sp2","sp3")
dat5

# Notice the samples in the pair a-b differ from each other exactly in the same
# way as samples in the pair c-d. However, a-b shares a rare species (sp1),
# whereas c-d shares a frequent species (sp3, which is also present in e). Thus,
# the dissimilarity a-b is the same of c-d using Bray-Curtis, but not using
# Goodall index:
vegdist(dat5, "bray")
dis.goodall(dat5)

# As the importance of a species for the Goodall index depends on its overall
# frequency in the community data, the deletion of a sample changes results:
dis.goodall(dat5[-5,])
```

---

dis.nness *NNESS and NESS dissimilarity indices*

Description

These dissimilarities indices overweight rare species to a desired degree. If rare species are heavy-weighthed, individuals of a rare species are more important than individuals of a common species. In the extreme, a species represented by a single individual will have the same weight of a common species, which is equivalent to the use of presence-absence data.

Usage

```
dis.nness(comm, m=NULL, ness=FALSE)
dis.nness.find.m(comm, ness=FALSE)
```

Arguments

comm	Dataframe or matrix with samples in rows and species in columns.

m	The number of individuals to be sampled in the estimation of species shared in NNESS and NESS indices. m must be an integer and lower than half (NESS) or the total (NNESS) abundance of the smallest sample in the pair under comparison. Higher m gives more importance to rare species. If no value is provided, the function will use dis.nness.find.m to find a value that is sensitive to both rare and abundant species (see Details below).
ness	Compute NESS index.

Details

Communities usually are composed of many rare and only a few common/frequent species. Although rare, a species may provide a valuable information in the estimation of resemblance between samples. However, the use of raw abundances makes the contribution of rare species to be negligible to the resulting index value. For instance, dropping a common species from the community data table usually will make much larger differences in the dissimilarity matrix than dropping a rare species.

There are a few indices that give more importance to an individual belonging to a rare than to a common/frequent species (see dis.chao and dis.goodall). Notice, however, that this differential weight of species can also be obtained, for instance, by log-transforming or standardizing data (e.g. dividing by maximum within each species) (Melo, in preparation). In this sense, an extreme case in which rare and common species have the same weight is in the use of presence/absence data.

NNESS is a modified or New version of the Normalized Expected Species Shared (NESS). NESS was proposed by Grassle & Smith (1976) and estimates similarity based on the number of species shared between random samples of size m individuals. Higher m gives more weight to rare species. For m = 1, NESS is the same as Morisita and NNESS is the same as Morisita-Horn dissimilarities. As higher values of m are used, dissimilarities tend to converge to presence-absence Sorensen index (when a rare and a common species have the same weight). NNESS was proposed by Trueblood et al. (1994) and circumvents the inability of NESS to handle samples composed exclusively by singletons (species with 1 individual).

Trueblood et al. (1994) also suggested to use an m value that is sensitive to both rare and abundant species. The NESS or NNESS index is calculated using values of m ranging from 1 up to half (NESS) or the total (NNESS) number of individuals in the smallest sample. The Kendall correlation is then calculated for each pair of triangular dissimilarity matrices. The selected m value is the one which produces a correlation with the matrix obtained with m = 1 that is most similar to the correlation with m = max, where max is the maximum value m may assume (abundance of the smallest sample for NNESS or half of that abundance for NESS). As this procedure may be slow for large datasets, up to 30 m values are used to compute dissimilarity matrices. NESS and NNESS are most suitable for raw abundance data and, thus, to data expressed as integers. This implementation, however, will work on non-integer data. NESS and NNESS formulae are for similarities and are computed here simply as 1-similarity.

The calculation of NESS and NNESS involves the use of binomial coefficients. For samples with many individuals (e.g. 1100) and high m values (e.g. 490), the resulting value is too large to be stored as double precision. Accordingly, this function uses the chooseZ of the Multiple Precision Arithmetic package when total abundance of at least one sample contains more than 1000 individuals. If no sample contains more than 1000 individuals, computations are performed using the simpler choose.

NESS and NNESS are most suitable to abundance data. However, choose is able to handle positive non-integers and calculations will be done accordingly. However, chooseZ is not able to handle non-integers and, thus, for datasets in which at least one sample contains more than 1000 individuals, non-integer abundances will be rounded up to next upper integer.

Value

A "dist" object for `dis.nness` and a integer for `dis.nness.find.m`.

Author(s)

Adriano Sanches Melo

References

Grassle, J.F. & W. Smith. 1976. A similarity measure sensitive to the contribution of rare species and its use in investigation of variation in marine benthic communities. Oecologia 25: 13-22.

Melo, A.S. Submitted. Is it possible to improve recovery of multivariate groups by weighting rare species in similarity indices?

Trueblood, D.D., E.D. Gallagher & D.M.Gould. Three stages of seasonal succession on the Savin Hill Cove mudflat, Boston Harbor. Limnology and Oceanography 39: 1440-1454.

See Also

vegdist, dis.chao, dis.goodall.

Examples

```
library(vegan)
aa <- c(1, 2, 4, 5)
bb <- c(1, 2, 0, 5)
cc <- c(0, 2, 3, 3)
dat3 <- rbind(aa, bb, cc)
colnames(dat3) <- c("sp1", "sp2", "sp3", "sp4")
dat3

#  NESS using m=1 is the same as Morisita and
# NNESS using m=1 is the same as Morisita-Horn:
dis.nness(dat3, m=1, ness=TRUE)
vegdist(dat3, method="morisita")

dis.nness(dat3, m=1, ness=FALSE)
vegdist(dat3, method="horn")

# The dissimilarity for the pair aa-bb is reduced if more weight is given to
# rare species (higher m). The reason is that aa-bb shares two rare
# species (sp1, sp2), whereas the pair aa-cc shares a single rare species (sp2).
dis.nness(dat3, m=1, ness=FALSE)
dis.nness(dat3, m=8, ness=FALSE)
```

japi — *Macroinvertebrate morphospecies living on stones of a stream in southeast Brazil*

Description

Abundance data of 66 macroinvertebrate morphospecies (columns) living on 75 stones (rows) of a stream in southeast Brazil. Stones were around 20 cm diameter and were sampled from downstream (pedra1) to upstream (pedra75) direction. Consecutive stones are spaced 1-6 m. Details can be found in Melo & Froehlich (2001).

Usage

```
data(japi)
```

Format

A data frame with 75 observations (stones) on 66 variables (morphospecies).

References

Melo, A.S. & C.G. Froehlich. 2001.Evaluation of methods for estimating macroinvertebrate species richness using individual stones in tropical streams. Freshwater Biology 46: 711-721.

Examples

```
data(japi)
japi
```

part.m.tree — *Partition of multi-sample dissimilarity indices using phylogenetic/functional data*

Description

This function computes the partition of multi-sample Sorensen and Jaccard dissimilarity indices described by Baselga (2012), but adapted for phylogenetic/functional data. The two resulting components are dissimilarities due to turnover and nestednes.

Usage

```
part.m.tree(comm, tree, index.family="sorensen")
```

Arguments

comm	Dataframe or matrix with samples in rows and species in columns.
tree	A phylogenetic/functional tree containing all species listed in comm
.	
index.family	The family of dissimilarity indices to be partitionated. Partial match to "sorensen" or "jaccard".

Value

A list including the number of samples and three dissimilarity values. The nomenclature of the dissimilarities follows Baselga (2012). For the jaccard:

JAC	Multi-sample Jaccard dissimilarity index.
JTU	The turnover component of the Jaccard index.
JNE	The nestedness component of the Jaccard index.

For the sorensen:

SOR	Multi-sample Sorensen dissimilarity index.
SIM	The turnover component of the Sorensen index (Simpson index).
SNE	The nestedness component of the Sorensen index.

Author(s)

Adriano Sanches Melo, but part of the code borrowed from beta.multi (package betapart) and phylosor (package picante)

References

Baselga, A. 2010. Partitioning the turnover and nestdness components of beta diversity. Global Ecology and Biogeography 19, 134-143.

Baselga, A. 2012. The relationship between species replacement, dissimilarity derived from nestedness and nestedness. Global Ecology and Biogeography 21, 1223-1232.

Leprieur, F., C. Albouy, J.D. Bortoli, P.F. Cowman, D.R. Bellwood and D. Mouillot. 2012. Quantifying phylogenetic beta diversity: distinguishing between 'true' turnover of lineages and phylogenetic diversity gradients. PLoS ONE 7(8), e42760. doi:10.1371/journal.pone.0042760

See Also

part.p.tree for pair-wise partitions.

Examples

```
library(picante)
data(sites4.6)
data(tree6)
part.m.tree(comm=sites4.6, tree=tree6, index.family="sorensen")
```

part.p.tree *Partition of pair-wise dissimilarity indices using phylogenetic/functional data*

Description

This function computes the partition of pair-wise Sorensen and Jaccard dissimilarity indices described by Baselga (2011, 2012), but adapted for phylogenetic/functional data. The two resulting components are dissimilarities due to turnover and nestednes

Usage

```
part.p.tree(comm, tree, index.family = "sorensen")
```

Arguments

comm	Dataframe or matrix with samples in rows and species in columns.
tree	A phylogenetic/functional tree containing all species listed in comm.
index.family	The family of dissimilarity indices to be partitionated. Partial match to "sorensen" or "jaccard".

Value

A list including three dissimilarity matrices. The nomenclature of the matrices follows Baselga (2012). For the jaccard:

jac	Jaccard dissimilarity matrix.
jtu	The turnover component of the Jaccard index.
jne	The nestedness (or richness difference) component of the Jaccard index.

For the sorensen:

sor	Sorensen dissimilarity matrix.
sim	The turnover component of the Sorensen index (Simpson index).
sne	The nestedness (or richness difference) component of the Sorensen index.

Author(s)

Adriano Sanches Melo, with part of the code borrowed from phylosor (package picante).

References

Baselga, A. 2010. Partitioning the turnover and nestdness components of beta diversity. Global Ecology and Biogeography 19, 134-143.

Baselga, A. 2012. The relationship between species replacement, dissimilarity derived from nestedness and nestedness. Global Ecology and Biogeography 21, 1223-1232.

Leprieur, F., C. Albouy, J.D. Bortoli, P.F. Cowman, D.R. Bellwood and D. Mouillot. 2012. Quantifying phylogenetic beta diversity: distinguishing between 'true' turnover of lineages and phylogenetic diversity gradients. PLoS ONE 7(8), e42760. doi:10.1371/journal.pone.0042760

See Also

`part.m.tree` for multi-sample partitions.

Examples

```
library(picante)
data(sites4.6)
data(tree6)
part.p.tree(comm=sites4.6, tree=tree6, index.family="sorensen")
```

`select.window` *Select cells inside a defined window of a gridded dataset*

Description

Selection of cells (or sites) around a focal cell. Cells must have have xy coordinates. Users should define a 'radius' from the focal cell that will define the window

Usage

```
select.window(xf, yf, radius = 1, xydata)
```

Arguments

`xf`	The x-coordinate of the focal cell.
`yf`	The y-coordinate of the focal cell.
`radius`	The radius that define the window to select neighbor cells.
`xydata`	A matrix-like object. The first and second columns of data must have x and y coordinates, respectively. The remaining columns must contain species or other atributes to be selected.

Details

The function is intended to be used in gridded data, but should work on sites irregularly scattered in the xy-space.

Value

A matrix-like object in which the first and second columns are the x and y coordinates. The remaining columns includes species (or attributes) observed in the selected cells.

Author(s)

Adriano Sanches Melo

Examples

```
x <- rep(1:5,each=5)
y <- rep(1:5,5)
spp <- matrix(1:100,25,4)
colnames(spp) <- c("sp1","sp2","sp3","sp4")
xyspp <- cbind(x,y,spp)
resu <- select.window(xf=3, yf=3, radius=1.1, xydata=xyspp)
resu

plot(x,y)
# maintain the plot window open.
points(resu[,1:2],col=2,cex=2 )
# cells of the selected window in red.

# A reduced number of cells will be selected for focal cells located in margins.
resu <- select.window(xf=5, yf=5, radius=1.1, xydata=xyspp)
plot(x,y)
# maintain the plot window open.
points(resu[,1:2],col=2,cex=2 )
# cells of the selected window in red.

# Unrecorded species in the selected window are removed from resulting
#  dataframe (or matrix):
spp<-matrix(rep(0:1,each=50),25,4)
colnames(spp)<-c("sp1","sp2","sp3","sp4")
xyspp<-cbind(x,y,spp)
select.window(xf=3, yf=3, radius=1.1, xydata=xyspp)
```

---

sites4.6 *Artificial dataset containing 4 samples [rows] and 6 species[columns].*

Description

Artificial dataset containing 4 samples [rows] and 6 species[columns].

Usage

```
data(sites4.6)
```

Format

A data frame with 4 observations on the following 6 variables.

`taxon_1` a numeric vector

`taxon_2` a numeric vector

`taxon_3` a numeric vector

`taxon_4` a numeric vector

`taxon_5` a numeric vector

`taxon_6` a numeric vector

Examples

```
data(sites4.6)
sites4.6
```

---

sites5.6 *Artificial dataset containing 5 samples [rows] and 6 species[columns].*

Description

Artificial dataset containing 5 samples [rows] and 6 species[columns].

Usage

```
data(sites5.6)
```

Format

A data frame with 5 observations on the following 6 variables.

`taxon_1` a numeric vector

`taxon_2` a numeric vector

`taxon_3` a numeric vector

`taxon_4` a numeric vector

`taxon_5` a numeric vector

`taxon_6` a numeric vector

Examples

```
data(sites5.6)
sites5.6
```

sites6.6 *Artificial dataset containing 6 samples [rows] and 6 species[columns].*

Description

Artificial dataset containing 6 samples [rows] and 6 species[columns].

Usage

```
data(sites6.6)
```

Format

A data frame with 6 observations on the following 6 variables.

`taxon_1` a numeric vector

`taxon_2` a numeric vector

`taxon_3` a numeric vector

`taxon_4` a numeric vector

`taxon_5` a numeric vector

`taxon_6` a numeric vector

Examples

```
data(sites6.6)
sites6.6
```

stairs6 *Example of phylogenetic tree containing 6 species.*

Description

Example of phylogenetic tree containing 6 species

Usage

```
data(stairs6)
```

Format

The format is: List of 5 $ edge : int [1:10, 1:2] 7 8 9 10 11 11 10 9 8 7 ... $ Nnode : int 5 $ tip.label : chr [1:6] "taxon_1" "taxon_2" "taxon_3" "taxon_4" ... $ edge.length: num [1:10] 1 1 1 1 1 1 2 3 4 5 $ root.edge : num 1 - attr(*, "class")= chr "phylo" - attr(*, "order")= chr "cladewise"

Examples

```
data(stairs6)
plot(stairs6)
```

`standExtent` *Standardization of spatial extent for two metacommunities by subsampling*

Description

Communities far from each other tend to differ more than those nearby. Accordingly, spatial extent usually affects species richness and other metacommunity properties. Comparison between two metacommunities may be biased if spatial extent differs. One potential solution is to standardize spatial extent by subsampling of the metacommunity more spread in space.

Usage

```
standExtent(d.large, d.small, ini.large=NULL, ini.small=NULL)
```

Arguments

`d.large`	A square (symmetrical) or triangular distance matrix of the metacommunity with large spatial extent.
`d.small`	A square (symmetrical) or triangular distance matrix of the metacommunity with small spatial extent.
`ini.large`	The first community to be used in the agregation process of the metacommunity with large spatial extent. Make sure it is present in `d.large`. If absent, a random site will be used.
`ini.small`	The first community to be used in the agregation process of the metacommunity with small spatial extent. Make sure it is present in `d.small`. If absent, a random site will be used.

Details

The standardized of spatial extent is done by selecting the same number of sites in both metacommunities using the criterion that within-metacommunity average distance among sites is similar.

The procedure consists of: 1) selection of a focal site `ini.large` in the metacommunity more spread in space, 2) selection of a second site that is nearest to the focal one, 3) selection of a third site using the criterion of minimal average distance among them, 4) repeat step 3 until all sites are included. The procedure then 5) selects a focal site `ini.small` in the metacommunity less spread in space, 6) selects a second site in the metacommunity less spread in space using the criterion of the distance between the focal site and this second one is the most similar to the distance in step 2 above, 7) selects a third site using the criterion that the distance among the three sites is most similar to the average distance in step 3 above, and 8) repeats step 7 until all sites in the less spread metacommunity are included.

Notice that after some sites are accumulated, the difference between the average distances in the two sets of metacommunities will increase abruptly. This will occur when most sites in the periphery of the less spread metacommunity are included.

Users may specify initial sites in the accumulation process (steps 1 and 5 above). If not specified, randomly selected sites will be used.

The percentage of difference of the two spatial extents is provided in the form: (d.mean.large - d.mean.small)/d.mean.large, where d.mean.large and d.mean.small are, respectively, the average within-metacommunity distances of the more and less spread metacommunities. This difference may be helpful to decide for the number and identity of sites to be used.

Value

A dataframe containing the names of the accumulated sites in both metacommunities, their within-metacommunity average distances and the percentage of difference in spatial extents.

Author(s)

Adriano Sanches Melo

References

Heino, J., A.S. Melo, J. Jyrkankallio-Mikkola, D.K. Petsch, V.Sa. Saito, K.T. Tolonen, L.M. Bini, T.S.F. Silva, V. Pajunen, J. Soininen & T. Siqueira. (in preparation) Higher richness but lower abundance of stream insects in tropical than boreal regions hold at regional, basin and local scales.

Examples

```
### Data
large.lat <- seq(2, 16, 2)
large.lon <- seq(2, 16, 2)
large.coo <- expand.grid(large.lon, large.lat)
large.coo[, 1] <- large.coo[, 1] + rnorm(64, sd=0.4)
large.coo[, 2] <- large.coo[, 2] + rnorm(64, sd=0.4)
rownames(large.coo) <- as.character(paste("large", 1:64, sep=""))
large.distances <- dist(large.coo)

small.lat <- 1:8
# Notice the spatial extent here is much smaller than that of
#  the "large" set above.
small.lon <- 1:8
small.coo <- expand.grid(small.lon, small.lat)
small.coo[, 1] <- small.coo[, 1] + rnorm(64, sd=0.4)
small.coo[, 2] <- small.coo[, 2] + rnorm(64, sd=0.4)
rownames(small.coo) <- as.character(paste("small", 1:64, sep=""))
small.distances <- dist(small.coo)

### Example 1 - Graphical demonstration of the subsampling process.
resu <- standExtent(d.large = large.distances, d.small=small.distances,
                    ini.large="large52", ini.small="small45")

plot(2:64, resu[2:64, "percent.dif"])
# Notice the subit increase of the curve. This is because the average distance
# in the small metacommunity cannot follow the increase in its large metacommunity.

op <- par(no.readonly = TRUE)
par(mfrow=c(2,1))
par(mar=c(1.7, 2, 1.2, 1.2))
```

```
plot(large.coo, xlim=c(0, 17), ylim=c(0, 17))
text(10, 16, "Large")
points(large.coo[resu[1, "site.large"],], col="blue", pch=15)
count <- 1
threshold <- 0
while(threshold <= 5 | count <= 7){
# Using a threshold of 5 percent or at least 7 sites are accumulated. The later
#   is due to large variations when number of sites is low.
   count <- count+1
   points(large.coo[resu[count,"site.large"],], col="red", pch=15)
   threshold <- abs(resu[count, "percent.dif"])
   Sys.sleep(0.05)
}
# Wait a little bit and watch the plot until the plotting of red dot stops.

par(mar=c(1.7, 2, 1.2, 1.2))
plot(small.coo, xlim=c(0, 17), ylim=c(0, 17))
# notice the reduced spatil extent.
text(10, 16, "Small")
points(small.coo[resu[1,"site.small"],], col="blue", pch=15)
count <- 1
threshold <- 0
while(threshold <= 5 | count <= 7){
   count <- count+1
   points(small.coo[resu[count,"site.small"],], col="red", pch=15)
   threshold <- abs(resu[count, "percent.dif"])
   Sys.sleep(0.05)
}
par(op)

### Example 2 - Generation of 3 paired standardized spatial extents
### using each of the three first sites in the larger spatial extent
### as ini.large.
names.large <- rownames(large.coo)[1:3]
resu.list <- vector(mode="list", length=3)
names(resu.list) <- names.large
for(i in names.large){
   print(i)
   resu.list[[i]] <- standExtent(d.large = large.distances,
                                 d.small=small.distances,
                                 ini.large=i)
}
resu.list
```

---

tree6 *Example of phylogenetic tree containing 6 species.*

Description

Example of phylogenetic tree containing 6 species

Usage

```
data(tree6)
```

Format

The format is: List of 4 $ edge : int [1:10, 1:2] 7 8 9 9 8 10 10 7 11 11 ... $ Nnode : int 5 $ tip.label : chr [1:6] "taxon_1" "taxon_2" "taxon_4" "taxon_3" ... $ edge.length: num [1:10] 1 1 1 1 1 1 1 1 2 2 - attr(*, "class")= chr "phylo"

Examples

```
data(tree6)
plot(tree6)
```

---

treeNodf *Tree-like Nestedness of Ecological Metacommunities*

Description

This function implements a generalization of the NODF (Nested Overlap and Decreasing Fill; Almeida-Neto et al. 2008) to quantify nestedness in metacommunities that takes into account relatedness among objects expressed as a tree-like object (Melo, Cianciaruso and Almeida-Neto, submitted).

Usage

```
treeNodf    (comm, col.tree, order.rows=FALSE, row.tree, order.cols=FALSE)
treeNodfTest(comm, col.tree, order.rows=FALSE, row.tree, order.cols=FALSE,
                                    null.model="perm.rows", permutations=999)
```

Arguments

comm	Dataframe or matrix with samples in rows and species in columns.
col.tree	A tree-like object containing all species listed in comm. This tree will be used to quantify (and test) nestedness among objects in rows (e.g. sites).
order.rows	Should rows of comm be ordered by decreasing Branch-Length (BL) Diversity. See details below.
row.tree	A tree-like object containing all row objects (e.g. sites) listed in comm. This tree will be used to quantify (and test) nestedness among objects in columns (e.g. species).
order.cols	Should columns of comm be ordered by decreasing Branch-Length (BL) Diversity. See details below.
null.model	Seven null models are currently implemented: "perm.rows", "perm.cols", "perm.rc", "perm.tip.cols", "perm.tip.rows", "perm.tip.rc" and "ff". See details.
permutations	Number of permutations of the null.model to assess significance.

Details

This is a direct extension of the NODF metric used to quantify nestedness in metacommunities using presence-absence data. NODF measures the proportion of the species richness present in a species-poor community which is present in a species-rich community. The tree-like version of this metric uses Branch-Length (BL) as a measure of diversity (instead of species richness used in NODF). For phylogenetic trees (a cladogram), BL is the Phylogenetic Diversity (PD, sensu Faith 1992). In this case, treeNODF (or phyloNODF) measures the proportion of the BL (or PD) in a BL-poor (or PD-poor) community that is present in a BL-rich (or PD-rich) community. The same reasoning applies to other tree-like objects such as functional dendrograms (Petchey & Gaston 2006).

The treeNODF follows the same approach of NODF and are calculated for each pair of objects in a given matrix dimension. For instance, for a dataset including 5 sites in its rows, 10 (5*4/2) pairs of values are calculated. The treeNODF for rows is simply the average of these 10 values. The same can be applied for columns. For instance, if columns represent species, one may assess whether there are nestedness in the environmental conditions where they occur (envNODF). For envNODF, users should supply a tree-like object that depicts resemblance among sites (i.e. a dendrogram where sites are classified according to environmental variables of interest). Finally, the treeNODF can be calculated for both rows and columns. In this case, two tree-like objects must be provided. The option to test rows (sites), columns (species) or both will depend exclusively on the hypothesis the user have raised regarding the nestedness structure of his study system.

The treeNODF can be partitionated into two components. The first one, called S.fraction, is the proportion of the species richness (or species incidence in the case of columns) of the BL-poor community that is shared with the BL-rich community. The second one is obtained as topoNODF = treeNODF - S.fraction and measures the effect of tree topology to treeNODF (see Melo et al. submitted).

NODF and its implementation in R `nestednodf` allows one to order rows and columns by frequencies (species richness for sites and species incidences for species). However, evidence of nestedness in a matrix automatically ordered by frequencies does not allows a proper inference of the mechanism generating nestedness (see Almeida-Neto et al 2008, Ulrich et al. 2009). Accordingly, better inferences of mechanisms generating nestedness and tree-nestedness should be done by ordering communities (rows) or species (columns) accordingly to a hypothesis raised *a priori* (and not by their S or BLs).

The "perm.rows" null model permute rows (sites) and is useful to test a site by species matrix in which rows (sites) were ordered by an *a priori* hypothesis (e.g. island area, Lomolino 1996). The "perm.cols" null model permute columns (species) and is useful to test a site by species matrix in which columns (species) were ordered by an *a priori* hypothesis (e.g. species body size). These two models must not be used if diversity (S or BL) is used to order communtiy data. This is because simulated values of the statistic will lower or equal to the observed one, but never higher. Although termed 'null models' here, notice these procedures constitutes a permutation test (rows or columns are permuted, not elements).

The "perm.tip.cols" null model shuffles species (columns) labels across tips of the tree-like object. This null model should be useful to test the effect of resemblance among species (columns) to the treeNODF result. Similarly, the "perm.tip.rows" model shuffles tip labels of the row.tree (usually, sites). The "perm.tip.rc" model shuffles tip labels of both trees.

The "ff" null model is a popular choice in the literature on species co-occurrence analysis. It maintains sum of rows (species richness of sites) and sum of columns (species frequencies) fixed. Notice, however, that treeNODF makes inferences about BL and this type of "diversity" is affected not only

by species richness but also by the resemblance among species (or sites) present in communities. The function commsimulator is used to shuffle matrices according to the "ff" null model using the "quasiswap" option.

Value

For the treeNodf, a list including:

rows — The average of tree.nodf.rows.dist and the averages of its two components for rows.

cols — The average of tree.nodf.cols.dist and the averages of its two components for columns.

mat — The average of tree.nodf.rows.dist and tree.nodf.cols.dist when treeNODF for both rows and columns are calculated. This is the treeNODF for the entire dataset. Averages for the two component metrics are also provided.

tree.nodf.rows.dist — A triangular matrix of treeNODFs for each pair of rows (sites). Each value represents the treeNODF of the site showed in its row in relation to the site showed in its column. Thus, values in the first column represents the treeNODFs of each community in relation to the community supposed to have the highest BL.

tree.nodf.cols.dist — A triangular matrix of treeNODFs for each pair of columns (species). Each value represents the treeNODF of the species showed in its column in relation to the site showed in its row. Thus, values in the first column represents the treeNODFs of each species in relation to the species supposed to have the highest BL (that is, the species supposedly to occurr in the widest set of conditions).

s.fraction.rows.dist — A triangular matrix of s.fration values for each pair of rows (sites).

s.fraction.cols.dist — A triangular matrix of s.fraction values for each pair of columns (species).

topo.nodf.rows.dist — A triangular matrix of topo.nodf values for each pair of rows (sites).

topo.nodf.cols.dist — A triangular matrix of topo.nodf values for each pair of columns (species).

For the treeNodfTest, a list containing the same list provided by treeNODF for the observed values and:

rows.aleats — treeNODF, S.fraction and topoNODF values for rows obtained using the chosen null model.

cols.aleats — treeNODF, S.fraction and topoNODF values for colums obtained using the chosen null model.

mat.aleats — treeNODF, S.fraction and topoNODF values for the entire matrix obtained using the chosen null model.

permutations — Number of permutations used in the test.

Author(s)

Adriano Sanches Melo

References

Almeida-Neto M., P. Guimaraes, P.R. Guimaraes, R.D. Loyola & W. Ulrich. 2008. A consistent metric for nestedness analysis in ecological systems: reconciling concept and measurement. Oikos 117:1227-1239.

Faith, D.P. 1992. Conservation evaluation and phylogenetic diversity. Biological Conservation 61:1-10.

Lomolino, M.V. 1996. Investigatin causality of nestedness of insular communities: selective immigration or extinctions? Journal of Biogeography 23:699-703.

Melo, A.S., M.V. Cianciaruso & M. Almeida-Neto. 2014. treeNODF: nestedness to phylogenetic, functional and other tree-based diversity metrics. Methods in Ecology and Evolution 5: 563-572.

Petchey, O.L. & K.J. Gaston. 2006. Functional diversity: back to basics and looking forward. Ecology Letters 9:741-758.

Ulrich, W., M. Almeida-Neto & N.J. Gotelli. 2009. A comsumer's guide to nestedness analysis. Oikos 118:3-17.

See Also

pd (package picante), beta.multi (package betapart)

Examples

```
library(picante)
data(sites5.6)
data(tree6)
treeNodf(sites5.6, col.tree=tree6)
# You likely will need more runs in your permutation test.
treeNodfTest(sites5.6, col.tree=tree6, null.model="perm.rows", permutations=9)

alt5 <- vegdist(1:5, method="euclidean")
alt5 <- hclust(alt5)
alt5 <- as.phylo(alt5)
alt5$tip.label<-rownames(sites5.6)
treeNodf    (sites5.6, row.tree=alt5)

treeNodfTest(sites5.6, row.tree=alt5, null.model="perm.cols",permutations=99)

treeNodf     (sites5.6, col.tree=tree6, row.tree=alt5)

treeNodfTest(sites5.6, col.tree=tree6, row.tree=alt5, null.model="perm.rc", permutations=99)

treeNodfTest(sites5.6, col.tree=tree6, row.tree=alt5, null.model="ff", permutations=99)

# In the example below treeNodf is zero because PD of the first sample is
#   lower than that of the second sample. Notice samples have the same species
#   richness and higher PD for the first sample is due to the presence of a
```

```
#   distinctive species (e).
tree <- read.tree(text="(((a:1,b:1):1,(c:1,d:1):1):2, e:4);")
plot(tree)
mat <- matrix(c(1,1,1,1,0, 1,1,1,0,1), byrow=TRUE, nrow=2, dimnames=list(1:2, letters[1:5]))
mat
pd(mat, tree)
treeNodf(mat, tree)

# Here treeNodf is non-zero... but you would only do that if you have
#    an 'a priori' hypothesis to order rows of your matrix this way!
treeNodf(mat[2:1, ], tree)
```

Index