Package 'betapart'

October 18, 2018

Type Package

Title Partitioning Beta Diversity into Turnover and Nestedness
Components

Version 1.5.1

Date 2018-10-17

Imports ape, fastmatch, geometry, picante, rcdd

Suggests doParallel, foreach, vegan

Description Functions to compute pair-wise dissimilarities (distance matrices) and multiple-site dissimilarities, separating the turnover and nestedness-resultant components of taxonomic (incidence and abundance based), functional and phylogenetic beta diversity.

License GPL (>= 2)

RoxygenNote 6.1.0

NeedsCompilation no

Author Andres Baselga [aut, cre],
David Orme [aut],
Sebastien Villeger [aut],
Julien De Bortoli [aut],
Fabien Leprieur [aut],
Maxime Logez [ctb],
Renato Henriques-Silva [ctb]

Maintainer Andres Baselga <andres.baselga@usc.es>

Repository CRAN

Date/Publication 2018-10-18 11:40:07 UTC

R topics documented:

---

bbsData   *BBS data by state for two timeslices*

Description

The data consists of binary presence/absence matrices for 569 bird species across 49 US states for two time slices (1980 - 1985 and 2000 - 2005). Only species (identified by AOU number) recorded during both time periods are included. The data are taken from the North American Breeding Bird Survey dataset and from a version of the database downloaded in May 2009.

Usage

```
data(bbsData)
```

Format

Two matrices (bbs1980 and bbs2000) of identical structure showing the presence/absence of the species as binary data.

state US states by USPS two letter codes.

aou Species identity by AOU species ID numbers.

Source

http://www.pwrc.usgs.gov/BBS/

Examples

```
data(bbsData)
str(bbs1980)
str(bbs2000)
```

---

beta.multi	*Multiple-site dissimilarities*

Description

Computes 3 multiple-site dissimilarities accounting for the spatial turnover and the nestedness components of beta diversity, and the sum of both values

Usage

```
beta.multi(x, index.family="sorensen")
```

Arguments

x   data frame, where rows are sites and columns are species. Alternatively x can be a betapart object derived from the betapart.core function

index.family   family of dissimilarity indices, partial match of "sorensen" or "jaccard".

Value

The function returns a list with the three multiple site dissimilarity values.

For index.family="sorensen" the three indices are:

beta.SIM   value of the turnover component, measured as Simpson dissimilarity

beta.SNE   value of the nestedness component, measured as nestedness-resultant fraction of Sorensen dissimilarity

beta.SOR   value of the overall beta diversity, measured as Sorensen dissimilarity

For index.family="jaccard" the three indices are:

beta.JTU   value of the turnover component, measured as turnover fraction of Jaccard dissimilarity

beta.JNE   value of the nestedness component, measured as nestedness-resultant fraction of Jaccard dissimilarity

beta.JAC   value of the overall beta diversity, measured as Jaccard dissimilarity

Author(s)

Andrés Baselga and David Orme

References

Baselga, A. 2010. Partitioning the turnover and nestedness 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

See Also

[beta.pair](), [beta.sample](), [betapart.core](), [beta.temp]()

Examples

```
data(ceram.s)
ceram.beta<-beta.multi(ceram.s, index.family="sor")
```

---

beta.multi.abund   *Abundance-based multiple-site dissimilarities*

Description

Computes 3 multiple-site dissimilarities accounting for the (i) balanced variation and (ii) abundance gradient components of dissimilarity, and the sum of both values (i.e. total abundance-based dissimilarity)

Usage

```
beta.multi.abund(x, index.family="bray")
```

Arguments

x	data frame, where rows are sites and columns are species. Alternatively x can be a betapart.abund object derived from the betapart.core.abund function
index.family	family of dissimilarity indices, partial match of "bray" or "ruzicka".

Value

The function returns a list with the three multiple site dissimilarity values.

For index.family="bray" the three indices are:

beta.BRAY.BAL	value of the balanced variation component of Bray-Curtis multiple-site dissimilarity
beta.BRAY.GRA	value of the abundance-gradient component of Bray-Curtis multiple-site dissimilarity
beta.BRAY	value of the overall dissimilarity, measured as Bray-Curtis multiple-site dissimilarity

For index.family="ruzicka" the three indices are:

beta.RUZ.BAL	value of the balanced variation component of Ruzicka multiple-site dissimilarity
beta.RUZ.GRA	value of the abundance-gradient component of Ruzicka multiple-site dissimilarity
beta.RUZ	value of the overall dissimilarity, measured as Ruzicka multiple-site dissimilarity

Author(s)

Andrés Baselga

References

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

See Also

[beta.pair.abund](), [beta.sample.abund](), [betapart.core.abund](), [beta.multi]()

Examples

```
require(vegan)
data(BCI)
beta.multi.abund(BCI, index.family="bray")
```

| beta.pair | *Incidence-based pair-wise dissimilarities* |

Description

Computes 3 distance matrices accounting for the (i) turnover (replacement), (ii) nestedness-resultant component, and (iii) total dissimilarity (i.e. the sum of both components).

Usage

```
beta.pair(x, index.family = "sorensen")
```

Arguments

| x | data frame, where rows are sites and columns are species. Alternatively x can be a betapart object derived from the betapart.core function |
| index.family | family of dissimilarity indices, partial match of "sorensen" or "jaccard". |

Value

The function returns a list with three dissimilarity matrices. For `index.family="sorensen"` the three matrices are:

`beta.sim`	`dist` object, dissimilarity matrix accounting for spatial turnover (replacement), measured as Simpson pair-wise dissimilarity
`beta.sne`	`dist` object, dissimilarity matrix accounting for nestedness-resultant dissimilarity, measured as the nestedness-fraction of Sorensen pair-wise dissimilarity
`beta.sor`	`dist` object, dissimilarity matrix accounting for total dissimilarity, measured as Sorensen pair-wise dissimilarity (a monotonic transformation of beta diversity)

For `index.family="jaccard"` the three matrices are:

`beta.jtu`	`dist` dissimilarity matrix accounting for spatial turnover, measured as the turnover-fraction of Jaccard pair-wise dissimilarity
`beta.jne`	`dist` object, dissimilarity matrix accounting for nestedness-resultant dissimilarity, measured as the nestedness-fraction of Jaccard pair-wise dissimilarity
`beta.jac`	`dist` object, dissimilarity matrix accounting for beta diversity, measured as Jaccard pair-wise dissimilarity (a monotonic transformation of beta diversity)

Author(s)

Andrés Baselga and David Orme

References

Baselga, A. 2010. Partitioning the turnover and nestedness 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

See Also

[beta.pair.abund](), code[beta.multi](), [beta.sample](), [betapart.core](), [beta.temp]()

Examples

```
data(ceram.s)
ceram.dist<-beta.pair(ceram.s, index.family="jac")
```

---

beta.pair.abund	*Abundance-based pair-wise dissimilarities*

Description

Computes 3 distance matrices accounting for the (i) balanced variation in abundances, (ii) abundance gradients, and (iii) total dissimilarity (i.e. the sum of both components).

Usage

```
beta.pair.abund(x, index.family = "bray")
```

Arguments

x   data frame, where rows are sites and columns are species. Alternatively x can be a betapart.abund object derived from the betapart.core.abund function

index.family   family of dissimilarity indices, partial match of "bray" or "ruzicka".

Value

The function returns a list with three dissimilarity matrices. For index.family="bray" the three matrices are:

beta.bray.bal   dist object, dissimilarity matrix accounting for the dissimilarity derived from balanced variation in abundance between sites

beta.bray.gra   dist object, dissimilarity matrix accounting for the dissimilarity derived from unidirectional abundance gradients

beta.bray   dist object, dissimilarity matrix accounting for total abundance-based dissimilarity between sites, measured as the Bray-Curtis index

For index.family="ruzicka" the three matrices are:

beta.ruz.bal   dist object, dissimilarity matrix accounting for the dissimilarity derived from balanced variation in abundance between sites

beta.ruz.gra   dist object, dissimilarity matrix accounting for the dissimilarity derived from unidirectional abundance gradients

beta.ruz   dist object, dissimilarity matrix accounting for total abundance-based dissimilarity between sites, measured as the Ruzicka index

Author(s)

Andrés Baselga

References

Baselga, A. 2013. Separating the two components of abundance-based dissimilarity: balanced changes in abundance vs. abundance gradients. Methods in Ecology and Evolution 4: 552–557

Legendre, P. 2014. Interpreting the replacement and richness difference components of beta diversity. Global Ecology and Biogeography, 23: 1324–1334

See Also

`beta.multi.abund`, `beta.sample.abund`, `betapart.core.abund`, `beta.pair`

Examples

```
require(vegan)
data(BCI)
BCI.pair<-beta.pair.abund(BCI, index.family="bray")
```

---

beta.sample	*Resampling multiple-site dissimilarity for n sites*

Description

Resamples the 3 multiple-site dissimilarities (turnover, nestedness-resultant fraction, and overall beta diversity) for a subset of sites of the original data frame.

Usage

```
beta.sample(x, index.family="sorensen", sites=nrow(x$data), samples = 1)
```

Arguments

x	data frame, where rows are sites and columns are species. Alternatively x can be a betapart object derived from the betapart.core function.
index.family	family of dissimilarity indices, partial match of "sorensen" or "jaccard".
sites	number of sites for which multiple-site dissimilarities will be computed. If not specified, default is all sites.
samples	number of repetitions. If not specified, default is 1.

Value

The function returns a list with a dataframe with the resampled 3 multiple-site dissimilarities (turnover fraction, nestedness-resultant fraction and overall dissimilarity; see `beta.multi`), a vector with the respective means and a vector with the respective standard deviation.

For `index.family="sorensen"`:

sampled.values dataframe containing beta.SIM, beta.SNE and beta.SOR for all samples

mean.values	vector containing the mean values of beta.SIM, beta.SNE and beta.SOR among samples
sd.values	vector containing the sd values of beta.SIM, beta.SNE and beta.SOR among samples

For index.family="jaccard":

sampled.values	dataframe containing beta.JTU, beta.JNE and beta.JAC for all samples
mean.values	vector containing the mean values of beta.JTU, beta.JNE and beta.JAC among samples
sd.values	vector containing the sd values of beta.JTU, beta.JNE and beta.JAC among samples

Author(s)

Andrés Baselga and David Orme

References

Baselga, A. 2010. Partitioning the turnover and nestedness 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

See Also

beta.multi, beta.sample, beta.temp

Examples

```
# Read the data for Northern and Southern European cerambycids
data(ceram.s)
data(ceram.n)

# Resample 100 times the multiple-site dissimilarities
# for 10 countries.
beta.ceram.s<-beta.sample(ceram.s, index.family="sor", sites=10, samples=100)
beta.ceram.n<-beta.sample(ceram.n, index.family="sor", sites=10, samples=100)

# Plot the distributions of beta.SIM in Southern Europe (red)
# and Northern Europe (blue)
plot(density(beta.ceram.s$sampled.values$beta.SIM), col="red", xlim=c(0,1))
lines(density(beta.ceram.n$sampled.values$beta.SIM), col="blue")

# Compute the p-value of difference in beta.SIM between South and North
# (i.e. the probability of finding in the North a higher value than
# in the South)
p.value.beta.SIM<-length(which(beta.ceram.s$sampled.values$beta.SIM<
beta.ceram.n$sampled.values$beta.SIM))/100

p.value.beta.SIM
```

```
# The result is 0 and we used 100 samples, so p<0.01
```

---

beta.sample.abund   *Resampling abundance-based multiple-site dissimilarity for n sites*

Description

Resamples the 3 abundance-based multiple-site dissimilarities (balanced variation fraction,abundance-gradient fraction, and overall dissimilarity) for a subset of sites of the original data frame.

Usage

```
beta.sample.abund(x, index.family="bray", sites = nrow(x), samples = 1)
```

Arguments

x	data frame, where rows are sites and columns are species
index.family	family of dissimilarity indices, partial match of "bray" or "ruzicka".
sites	number of sites for which multiple-site dissimilarities will be computed. If not specified, default is all sites.
samples	number of repetitions. If not specified, default is 1.

Value

The function returns a list with a dataframe with the resampled 3 multiple-site dissimilarities (balanced variation fraction, abundance-gradient fraction and overall dissimilarity; see beta.multi.abund), a vector with the respective means and a vector with the respective standard deviation.

For index.family="bray":

sampled.values	dataframe containing beta.BRAY.BAL, beta.BRAY.GRA and beta.BRAY for all samples
mean.values	vector containing the mean values of beta.BRAY.BAL, beta.BRAY.GRA and beta.BRAY among samples
sd.values	vector containing the sd values of beta.BRAY.BAL, beta.BRAY.GRA and beta.BRAY among samples

For index.family="ruzicka":

sampled.values	dataframe containing beta.RUZ.BAL, beta.RUZ.GRA and beta.RUZ for all samples
mean.values	vector containing the mean values of beta.RUZ.BAL, beta.RUZ.GRA and beta.RUZ among samples
sd.values	vector containing the sd values of beta.RUZ.BAL, beta.RUZ.GRA and beta.RUZ among samples

Author(s)

Andrés Baselga

References

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

See Also

[beta.multi.abund](), [beta.sample]()

Examples

```
require(vegan)
data(BCI)
beta.sample.abund(BCI, index.family="bray", sites=10, samples=100)
```

---

beta.temp	*Temporal change in community composition*

Description

Computes the dissimilarity for each locality between time 1 and time 2, considering the turnover and nestedness components of temporal change, and the sum of both values (overall change)

Usage

```
beta.temp(x, y, index.family="sorensen")
```

Arguments

x	data frame for time 1, where rows are sites and columns are species
y	data frame for time 2, where rows are sites and columns are species. x and y must contain exactly the same sites and species
index.family	family of dissimilarity indices, partial match of "sorensen" or "jaccard".

Value

The function returns a data frame where rows are sites and columns are pairwise dissimilarity values between cell composition in time 1 and time 2. For index.family="sorensen" the indices are beta.sim, beta.sne, and beta.sor. For index.family="jaccard" the indices are beta.jtu, beta.sne, and beta.jac.

Author(s)

Andrés Baselga and David Orme

References

Baselga, A. 2010. Partitioning the turnover and nestedness 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

See Also

beta.multi, beta.pair, beta.sample, betapart.core,

Examples

```
data(bbsData)
bbs.t <- beta.temp(bbs1980, bbs2000, index.family="sor")
```

---

betapart   *Partitioning beta diversity into turnover and nestedness components*

Description

betapart allows computing pair-wise dissimilarities (distance matrices) and multiple-site dissimilarities, separating the turnover and nestedness-resultant components of taxonomic (incidence and abundance based), functional and phylogenetic beta diversity.

Details

The partitioning of incidence-based dissimilarity can be performed for two different families of indices: Sorensen and Jaccard. The pairwise function beta.pair yields 3 distance matrices accounting for the spatial turnover and the nestedness components of beta-diversity. The third distance matrix accounts for the sum of both components, i.e. total dissimilarity (a monotonic transformation of beta diversity). The multiple site function beta.multi yields the spatial turnover and the nestedness components of overall dissimilarity, and the sum of both components, total dissimilarity. The basic calculations for all these multiple-site measures and pairwise dissimilarity matrices can be computed using the function betapart.core, which returns an object of class betapart. This is useful for large datasets as the consuming calculations are done only once, and its result can then be used for computing many indices. The multiple-site values can be randomly sampled a specified number of times for a specified number of sites using the function beta.sample. The aforementioned indices used for assessing spatial patterns can also be used for measuring temporal changes in community composition with the function beta.temp. Likewise, an analogous framework has been implemented for separating the two components of abundance-based dissimilarity (balanced changes in abundance vs. abundance gradients) using commands beta.pair.abund, beta.multi.abund, betapart.core.abund, and beta.sample.abund. The framework has been extended for functional beta diversity with commands functional.betapart.core, functional.beta.pair and functional.beta.multi, and for phylogenetic beta diversity with commands phylo.betapart.core, phylo.beta.pair and phylo.beta.multi. The package also allows fitting negative exponential or power law distance-decay models for assessing the relationship between assemblage (dis)similarity and spatial (or other) distance. decay.model will fit the distance-decay function via GLM, plot.decay

will plot the distance-decay pattern, and `boot.coefs.decay` will bootstrap the paramaters of the distance-decay model.

Author(s)

Andrés Baselga, David Orme, Sébastien Villéger, Julien De Bortoli and Fabien Leprieur

References

Baselga, A. 2010. Partitioning the turnover and nestedness 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

Baselga, A. 2013. Separating the two components of abundance-based dissimilarity: balanced changes in abundance vs. abundance gradients. Methods in Ecology and Evolution, 4: 552-557

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

Baselga A, Leprieur, F. 2015. Comparing methods to separate components of beta diversity. Methods in Ecology and Evolution 6: 1069-1079

Baselga A, Orme CDL. 2012. betapart: an R package for the study of beta diversity. Methods Ecol. Evol. 3: 808-812

Gómez-Rodríguez, C. & Baselga, A. 2018. Variation among European beetle taxa in patterns of distance decay of similarity suggests a major role of dispersal processes. Ecography, in press

Legendre P. 2014. Interpreting the replacement and richness difference components of beta diversity. Global Ecology and Biogeography, 23: 1324–1334

Leprieur F, Albouy C, De Bortoli J, Cowman PF, Belwood DR, Mouillot D. 2012. Quantifying phylogenetic beta diversity: distinguishing between "true" turnover of lineages and phylogenetic diversity gradients. PLoS One 7(8): e42760

Villéger, S. Grenouillet, G., Brosse, S. 2013. Decomposing functional beta-diversity reveals that low functional beta-diversity is driven by low functional turnover in European fish assemblages. Global Ecology and Biogeography, 22: 671-681

---

betapart.core   *Core calculations of betapart*

Description

Computes the basic quantities needed for computing the multiple-site beta diversity measures and pairwise dissimilarity matrices.

Usage

```
betapart.core(x)
```

Arguments

x	data frame, where rows are sites and columns are species

Value

The function returns an object of class betapart with the following elements:

sumSi	the sum of the species richness values of all sites
St	the total richness in the dataset
a	the multiple-site analog of the shared species term
shared	a matrix containing the number of species shared between pairs of sites
not.shared	a matrix containing the number of species not shared between pairs of sites: b, c
sum.not.shared	a matrix containing the total number of species not shared between pairs of sites: b+c
max.not.shared	a matrix containing the total maximum number of species not shared between pairs of sites: max(b,c)
min.not.shared	a matrix containing the total minimum number of species not shared between pairs of sites: min(b,c)

Author(s)

Andrés Baselga and David Orme

References

Baselga, A. 2010. Partitioning the turnover and nestedness 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

See Also

[beta.multi](), [beta.pair](), [beta.sample](), [beta.temp](),

Examples

```
data(ceram.s)
ceram.core.s<-betapart.core(ceram.s)
ceram.dist.jac<-beta.pair(ceram.core.s, index.family="jac")
ceram.dist.sor<-beta.pair(ceram.core.s, index.family="sor")
ceram.multi.jac<-beta.multi(ceram.core.s, index.family="jac")
ceram.multi.sor<-beta.multi(ceram.core.s, index.family="sor")
```

betapart.core.abund	*Core calculations of betapart for abundance-based dissimilarity measures*

Description

Computes the basic quantities needed for computing the abundance-based multiple-site dissimilarity measures and pairwise dissimilarity matrices.

Usage

```
betapart.core.abund(x)
```

Arguments

x   data frame, where rows are sites and columns are species

Value

The function returns an object of class `betapart.abund` with the following elements:

`multiple.shared.abund`
the multiple-site intersection component in terms of abundances (AM)

`pair.shared.abund`
a matrix containing the agreement in abundance between pairs of sites (A)

`min.not.shared.abund`
a matrix containing the minimum disagreement in abundance between pairs of sites: min(B,C)

`max.not.shared.abund`
a matrix containing the maximum disagreement in abundance between pairs of sites between pairs of sites: max(B,C)

`pair.not.shared.abund`
a matrix containing the total disagreement in abundance between pairs of sites: B+C

Author(s)

Andrés Baselga

References

Baselga, A. 2013. Separating the two components of abundance-based dissimilarity: balanced changes in abundance vs. abundance gradients. Methods in Ecology and Evolution, 4: 552–557

Legendre, P. 2014. Interpreting the replacement and richness difference components of beta diversity. Global Ecology and Biogeography, 23: 1324–1334

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

See Also

beta.multi.abund, beta.pair.abund, beta.sample.abund, betapart.core

Examples

```
require(vegan)
data(BCI)
core.BCI<-betapart.core.abund(BCI)
pair.BCI<-beta.pair.abund(core.BCI)
multi.BCI<-beta.multi.abund(core.BCI)
```

---

boot.coefs.decay   *Bootstrapping the parameters of distance-decay models computed*
                          with decay.model()

Description

Takes the output of decay.model() and bootstraps the parameters (intercept and slope).

Usage

```
boot.coefs.decay(x, R)
```

Arguments

x	the output of decay.model().
R	the number of bootstrap samples.

Value

The function returns a list with:

model.type	functional form of the model, either negative exponential or power law.
y.type	similarties or dissimilarities.
boot.coefs	a matrix with the coefficients bootstrapped distributions, including intercept values in the first column, and slope values in the second column.
original.coefs	model coefficients as estimated with a GLM using decay.model().
mean.boot	the mean of the bootstrapped distributions.
sd.boot	the standard deviation of the bootstrapped distributions.

Author(s)

Andrés Baselga

References

Gómez-Rodríguez, C. & Baselga, A. 2018. Variation among European beetle taxa in patterns of distance decay of similarity suggests a major role of dispersal processes. Ecography, in press

See Also

decay.model

Examples

```
require(vegan)

data(BCI)
## UTM Coordinates (in metres)
UTM.EW <- rep(seq(625754, 626654, by=100), each=5)
UTM.NS <- rep(seq(1011569,  1011969, by=100), len=50)

spat.dist<-dist(data.frame(UTM.EW, UTM.NS))

dissim.BCI<-beta.pair.abund(BCI)$beta.bray.bal

BCI.decay.pow<-decay.model(dissim.BCI, spat.dist, model.type="pow", perm=100)

boot.coefs.decay(BCI.decay.pow, 50)
```

---

bray.part *Partitioning pair-wise Bray-Curtis dissimilarities*

Description

Computes 3 distance matrices accounting for the balanced variation and abundance gradient components of Bray-Curtis dissimilarity, and the sum of both values (i.e. Bray-Curtis dissimilarity)

Usage

```
bray.part(x)
```

Arguments

x   data frame of species abundances, where rows are sites and columns are species.

Value

The function returns a list with three dissimilarity matrices.

bray.bal   dist object, dissimilarity matrix accounting for the dissimilarity derived from balanced variation in abundance between sites

bray.gra   dist object, dissimilarity matrix accounting for the dissimilarity derived from unidirectional abundance gradients

bray   `dist` object, dissimilarity matrix accounting for total abundance-based dissimi-
                larity between sites, measured as the Bray-Curtis index

Author(s)

Andrés Baselga

References

Baselga, A. in press. Separating the two components of abundance-based dissimilarity: balanced
changes in abundance vs. abundance gradients. Methods in Ecology and Evolution. DOI: 10.1111/2041-
210X.12029

See Also

beta.pair

Examples

```
require(vegan)
data(BCI)
BCI.matrices<-bray.part(BCI)
```

---

ceram.n   *Cerambycidae from Northern European Countries*

Description

The ceram.n data frame has 19 rows and 634 columns. Columns are presence/absence values of 634
species. The variable names are formed from the scientific names. The case names are standard
country abbreviations, excepting RSS (Southern European Russia), RSC (Central European Russia)
and RSN (Northern European Russia).

Usage

```
data(ceram.n)
```

Source

http://www.cerambycidae.net/

References

1. Danilevsky, M. L. 2007. A check-list of Longicorn Beetles (Coleoptera, Cerambycoidea) of
   Europe.Available at http://www.cerambycidae.net/

2. Baselga, A. 2008. Determinants of species richness, endemism and turnover in European
   longhorn beetles. Ecography 31:263-271

---

ceram.s   *Cerambycidae from Southern European Countries*

Description

The ceram.s data frame has 15 rows and 634 columns. Columns are presence/absence values of 634 species. The variable names are formed from the scientific names. The case names are standard country abbreviations, excepting SS (Serbia) and CBH (Croatia and Bosnia-Herzegovina).

Usage

```
data(ceram.s)
```

Source

http://www.cerambycidae.net/

References

1. Danilevsky, M. L. 2007. A check-list of Longicorn Beetles (Coleoptera, Cerambycoidea) of Europe.Available at http://www.cerambycidae.net/

2. Baselga, A. 2008. Determinants of species richness, endemism and turnover in European longhorn beetles. Ecography 31:263-271

---

decay.model   *Fitting distance decay models to pair-wise assemblage similarity*

Description

Fits a negative exponential or power law function (via GLM) describing (i) the decay of assemblage similarity with spatial (or any other) distance, or, equivalently, (ii) the increase of assemblage dissimilarity with distance.

Usage

```
decay.model(y, x, model.type="exponential", y.type="similarities", perm=100)
```

Arguments

y	codedist object, either containing similarities or dissimilarities between pairs of assemblages.
x	codedist object, containing distances (spatial or other) between pairs of assemblages.
model.type	functional form of the model, either negative exponential or power law, partial match of "exponential" or "power".

y.type   polarity of the codedist object (i.e. 1 means total similarity or total dissimilarity),
                  partial match of "similarities" or "dissimilarities".

perm   number of randomizations to assess significance.

Value

The function returns a list with:

data   dataframe containing distances (spatial or other) and similarities (or dissimilar-
                  ities).

model   the fitted GLM.

model.type   functional form of the model, either negative exponential or power law.

y.type   similarties or dissimilarities.

pseudo.r.squared

                  similarties or dissimilarities.

a.intercept   intercept of the model, i.e. similarity or dissimilarity at distance=0.

b.slope   slope of the model, i.e. rate at which similarity decreases with distance, or dis-
                  similarity increases with distance in a negative exponential or power law model.

p.value   significance of the model, as estimated from a randomization test.

Author(s)

Andrés Baselga

References

Nekola, J.C. & McGill, B.J. 2014. Scale dependency in the functional form of the distance decay
relationship. Ecography, 37, 309-320

Gómez-Rodríguez, C. & Baselga, A. 2018. Variation among European beetle taxa in patterns of
distance decay of similarity suggests a major role of dispersal processes. Ecography, in press

See Also

beta.pair, beta.pair.abund

Examples

```
require(vegan)

data(BCI)
## UTM Coordinates (in metres)
UTM.EW <- rep(seq(625754, 626654, by=100), each=5)
UTM.NS <- rep(seq(1011569,  1011969, by=100), len=50)

spat.dist<-dist(data.frame(UTM.EW, UTM.NS))

dissim.BCI<-beta.pair.abund(BCI)$beta.bray.bal
```

```
plot(spat.dist, dissim.BCI, ylim=c(0,1), xlim=c(0, max(spat.dist)))

BCI.decay.exp<-decay.model(dissim.BCI, spat.dist, y.type="dissim", model.type="exp", perm=100)

BCI.decay.pow<-decay.model(dissim.BCI, spat.dist, y.type="dissim", model.type="pow", perm=100)

plot.decay(BCI.decay.exp, col=rgb(0,0,0,0.5))
plot.decay(BCI.decay.exp, col="red", remove.dots=TRUE, add=TRUE)
plot.decay(BCI.decay.pow, col="blue", remove.dots=TRUE, add=TRUE)
```

---

functional.beta.multi   *Multiple-site functional dissimilarities*

Description

Computes 3 multiple-site functional dissimilarities accounting for the spatial turnover and the nestedness components of functional beta diversity, and the sum of both values. Functional dissimilarities are based on volume of convex hulls intersections in a multidimensional functional space.

Usage

```
functional.beta.multi(x, traits, index.family="sorensen", warning.time=TRUE)
```

Arguments

x	data frame, where rows are sites and columns are species. Alternatively x can be a `functional.betapart` object derived from the `functional.betapart.core` function
traits	if x is not a `functional.betapart` object, a data frame, where rows are species and columns are functional space dimensions (i.e. quantitative traits or synthetic axes after PCoA). Number of species in each site must be strictly higher than number of dimensions. Number of dimensions should not exceed 4 and number of sites should not exceed 10. See `Details`.
index.family	family of dissimilarity indices, partial match of "sorensen" or "jaccard".
warning.time	a logical value indicating whether computation of multiple-site dissimilarities would stop if number of dimensions exceeds 4 or if number of sites exceeds 10. If turn to `FALSE`, computation process can be tracked in the `step.fbc.txt` file, see `Details`.

Details

For multiple-site dissimilarities metrics (N>2 sites), the volume of the union of the N convex hulls is computed using the inclusion-exclusion principle (Villéger et al., 2011). It requires to compute the volume of all the intersections between 2 to N convex hulls. Intersection between k>2 convex hulls is computed as the intersection between the two convex hulls shaping intersections between the corresponding k-1 convex hulls, e.g. V(AnBnC)=V((AnB)n(BnC)). For N sites, computing

multiple-site dissimilarity metrics thus requires computing 2^N-(N+1) pair-wise intersections between convex hulls in a multidimensional functional space. Computation time of the intersection between two convex hulls increases with the number of dimensions (D) of the functional space. Therefore, to prevent from running very long computation process warning.time is set by default to stop the function if D>4 or N>10. Computation progress can be tracked in the "step.fbc.txt" file written in the working directory. This table shows proportion of steps completed for computing convex hull volume shaping each site ("FRi") and intersections between them ("intersection_k").

Note that the the betapart package now supports external parallel computing for null models. However, this functionality is only availabe in functional.betapart.core. In this case, use the functional.betapart object as x in this function. See functional.betapart.core for more details.

Value

The function returns a list with the three multiple site functional dissimilarity values.

For index.family="sorensen" the three indices are:

beta.SIM	value of the functional turnover component, measured as Simpson derived functional dissimilarity
beta.SNE	value of the functional nestedness component, measured as nestedness-resultant fraction of Sorensen derived functional dissimilarity
beta.SOR	value of the overall functional beta diversity, measured as Sorensen derived functional dissimilarity

For index.family="jaccard" the three indices are:

beta.JTU	value of the functional turnover component, measured as turnover fraction of Jaccard derived functional dissimilarity
beta.JNE	value of the functional nestedness component, measured as nestedness-resultant fraction of Jaccard derived functional dissimilarity
beta.JAC	value of the overall functional beta diversity, measured as Jaccard derived functional dissimilarity

Author(s)

Sébastien Villéger, Andrés Baselga and David Orme

References

Villéger S., Novack-Gottshal P. & Mouillot D. 2011. The multidimensionality of the niche reveals functional diversity changes in benthic marine biotas across geological time. Ecology Letters 14: 561-568

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

Villéger, S. Grenouillet, G., Brosse, S. 2013. Decomposing functional beta-diversity reveals that low functional beta-diversity is driven by low functional turnover in European fish assemblages. Global Ecology and Biogeography 22: 671–681

See Also

functional.beta.pair, functional.betapart.core, beta.multi

Examples

```
##### 4 communities in a 2D functional space (convex hulls are rectangles)
traits.test<-cbind( c(1,1,1,2,2,3,3,4,4,5,5) , c(1,2,4,1,2,3,5,1,4,3,5) )
dimnames(traits.test)<-list(paste("sp",1:11,sep="") , c("Trait 1","Trait 2") )

comm.test<-matrix(0,4,11,dimnames=list( c("A","B","C","D") , paste("sp",1:11,sep="") ) )
comm.test["A",c(1,2,4,5)]<-1
comm.test["B",c(1,3,8,9)]<-1
comm.test["C",c(6,7,10,11)]<-1
comm.test["D",c(2,4,7,9)]<-1

plot(5,5,xlim=c(0,6), ylim=c(0,6), type="n", xlab="Trait 1",ylab="Trait 2")
points(traits.test[,1],traits.test[,2], pch=21,cex=1.5,bg="black")
rect(1,1,4,4, col="#458B0050", border="#458B00") ; text(2.5,2.5,"B",col="#458B00",cex=1.5)
polygon(c(2,1,3,4), c(1,2,5,4), col="#DA70D650", border="#DA70D6") ;
text(2.5,3,"D",col="#DA70D6",cex=1.5)
rect(1,1,2,2, col="#FF000050" , border="#FF0000") ; text(1.5,1.5,"A",col="#FF0000",cex=1.5)
rect(3,3,5,5, col="#1E90FF50", border="#1E90FF") ; text(4,4.2,"C",col="#1E90FF",cex=1.5)

test.multi<-functional.beta.multi(x=comm.test, traits=traits.test, index.family = "jaccard" )
test.multi

test.multi.ABC<-functional.beta.multi(x=comm.test[c("A","B","C"),], traits=traits.test,
index.family = "jaccard" )
test.multi.ABC

test.multi.ABD<-functional.beta.multi(x=comm.test[c("A","B","D"),], traits=traits.test,
index.family = "jaccard" )
test.multi.ABD
```

---

functional.beta.pair   *Pair-wise functional dissimilarities*

Description

Computes 3 distance matrices accounting for the spatial turnover and nestedness components of functional beta diversity, and the sum of both values. Functional dissimilarities are based on volume of convex hulls intersections in a multidimensional functional space.

Usage

```
functional.beta.pair(x, traits, index.family="sorensen")
```

Arguments

x	data frame, where rows are sites and columns are species. Alternatively x can be a `functional.betapart` object derived from the `functional.betapart.core` function
traits	if x is not a `functional.betapart` object, a data frame, where rows are species and columns are functional space dimensions (i.e. quantitative traits or synthetic axes after PCoA). Number of species in each site must be strictly higher than number of dimensions.
index.family	family of dissimilarity indices, partial match of "sorensen" or "jaccard".

Details

Note that the the `betapart` package now supports external parallel computing for null models. However, this functionality is only availabe in `functional.betapart.core`. In this case, use the `functional.betapart` object as x in this function. See `functional.betapart.core` for more details.

Value

The function returns a list with three functional dissimilarity matrices.

For `index.family="sorensen"` the three matrices are:

funct.beta.sim	dist object, dissimilarity matrix accounting for functional turnover, measured as Simpson derived pair-wise functional dissimilarity
funct.beta.sne	dist object, dissimilarity matrix accounting for nestedness-resultant functional dissimilarity, measured as the nestedness-fraction of Sorensen derived pair-wise functional dissimilarity
funct.beta.sor	dist object, dissimilarity matrix accounting for functional beta diversity, measured as Sorensen derived pair-wise functional dissimilarity

For `index.family="jaccard"` the three matrices are:

funct.beta.jtu	dist object, dissimilarity matrix accounting for functional turnover, measured as the turnover-fraction of Jaccard derived pair-wise functional dissimilarity
funct.beta.jne	dist object, dissimilarity matrix accounting for nestedness-resultant functional dissimilarity, measured as the nestedness-fraction of Jaccard derived pair-wise functional dissimilarity
funct.beta.jac	dist object, dissimilarity matrix accounting for functional beta diversity, measured as Jaccard derived pair-wise functional dissimilarity

Author(s)

Sébastien Villéger, Andrés Baselga and David Orme

References

Villéger S., Novack-Gottshal P. & Mouillot D. 2011. The multidimensionality of the niche reveals functional diversity changes in benthic marine biotas across geological time. Ecology Letters 14: 561-568

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

Villéger, S. Grenouillet, G., Brosse, S. 2013. Decomposing functional beta-diversity reveals that low functional beta-diversity is driven by low functional turnover in European fish assemblages. Global Ecology and Biogeography 22: 671–681

See Also

functional.beta.multi, functional.betapart.core, beta.pair

Examples

```
##### 4 communities in a 2D functional space (convex hulls are rectangles)
traits.test<-cbind( c(1,1,1,2,2,3,3,4,4,5,5) , c(1,2,4,1,2,3,5,1,4,3,5) )
dimnames(traits.test)<-list(paste("sp",1:11,sep="") , c("Trait 1","Trait 2") )

comm.test<-matrix(0,4,11,dimnames=list( c("A","B","C","D") , paste("sp",1:11,sep="") ) )
comm.test["A",c(1,2,4,5)]<-1
comm.test["B",c(1,3,8,9)]<-1
comm.test["C",c(6,7,10,11)]<-1
comm.test["D",c(2,4,7,9)]<-1

plot(5,5,xlim=c(0,6), ylim=c(0,6), type="n", xlab="Trait 1",ylab="Trait 2")
points(traits.test[,1],traits.test[,2], pch=21,cex=1.5,bg="black")
rect(1,1,4,4, col="#458B0050", border="#458B00") ; text(2.5,2.5,"B",col="#458B00",cex=1.5)
polygon(c(2,1,3,4), c(1,2,5,4), col="#DA70D650", border="#DA70D6") ;
text(2.5,3,"D",col="#DA70D6",cex=1.5)
rect(1,1,2,2, col="#FF000050" , border="#FF0000") ; text(1.5,1.5,"A",col="#FF0000",cex=1.5)
rect(3,3,5,5, col="#1E90FF50", border="#1E90FF") ; text(4,4.2,"C",col="#1E90FF",cex=1.5)

test.pair<-functional.beta.pair(x=comm.test, traits=traits.test, index.family = "jaccard" )
lapply(test.pair,round,2)
```

---

functional.betapart.core

Core calculations of functional dissimilarities metrics

Description

Computes the basic quantities needed for computing the multiple-site functional beta diversity measures and pairwise functional dissimilarity matrices. This version of the function now supports external parallelization for null models.

Usage

```
functional.betapart.core(x, traits, multi=TRUE, warning.time=TRUE, return.details=FALSE,
fbc.step=FALSE, core.ident=NULL)
```

Arguments

x	data frame, where rows are sites and columns are species.
traits	data frame, where rows are species and columns are functional space dimensions (i.e. quantitative traits or synthetic axes after PCoA). Number of species in each site must be strictly higher than number of dimensions.
multi	a logical value indicating whether basic quantities for multiple-site functional beta-diversity should be computed. See `Details`.
warning.time	a logical value indicating whether computation of multiple-site dissimilarities would stop if number of dimensions exceeds 4 or if number of sites exceeds 10. If turn to `FALSE`, computation process can be tracked in the `step.fbc.txt` file, see `Details`.
return.details	a logical value indicating whether volume and coordinates of vertices of convex hulls shaping each site and their intersections in the functional space should be returned.
fbc.step	a logical value indicating whether the computation progress tracking file "step.fbc.txt" should be created; Setting it to FALSE will speed up the function.
core.ident	computer core identifier to allow for external parallelization of this function (e.g., for null models),see `Examples`.

Details

For multiple-site dissimilarities metrics (N>2 sites), the volume of the union of the N convex hulls is computed using the inclusion-exclusion principle (Villéger et al., 2011). It requires to compute the volume of all the intersections between 2 to N convex hulls. Intersection between k>2 convex hulls is computed as the intersection between the two convex hulls shaping intersections between the corresponding k-1 convex hulls, e.g. V(AnBnC)=V((AnB)n(BnC)). For N sites, computing multiple-site dissimilarity metrics thus requires computing 2^N-(N+1) pair-wise intersections between convex hulls in a multidimensional functional space. Computation time of the intersection between two convex hulls increases with the number of dimensions (D) of the functional space. Therefore, to prevent from running very long computation process `warning.time` is set by default to stop the function if D>4 or N>10.

If `fbc.step` is set to TRUE, computation progress can be tracked in the "step.fbc.txt" file written in the working directory. In the case of external parallelization, each core will write a unique computation progress tracking file "step.fbc_core.ident.txt" using the `core.ident` argument. This table shows proportion of steps completed for computing convex hull volume shaping each site ("FRi") and intersections between them ("intersection_k").

Value

The function returns an object of class `betapart` with the following elements:

sumFRi	the sum of the functional richness values of all sites

FRt	the total functional richness in the dataset
a	the multiple-site analog of the shared functional richness term
shared	a matrix containing the functional richness shared between pairs of sites
not.shared	a matrix containing the functional richness not shared between pairs of sites: b, c
sum.not.shared	a matrix containing the total functional richness not shared between pairs of sites: b+c
max.not.shared	a matrix containing the total maximum functional richness not shared between pairs of sites: max(b,c)
min.not.shared	a matrix containing the total minimum functional richness not shared between pairs of sites: min(b,c)
details	if return.details=TRUE a list of two lists: $CH a list with a vector (FRi) of functional richness in each site (i.e. convex hull volume) and coord_vertices a list of N matrices with the coordinates of species being vertices in the D-dimensions functional space. $intersections a list of 3 lists: $combinations, N-1 matrices with all combinations of 2 to N sites (numbers are rank of sites in x) ; $volumes, N-1 vectors with the volume inside the intersection between each combination of sites ; $coord_vertices, list of N-1 matrices with the coordinates of the vertices shaping these intersections (NA if no intersection).

Author(s)

Sébastien Villéger, Andrés Baselga, David Orme, Renato Henriques-Silva, Maxime Logez

References

Villéger S., Novack-Gottshal P. & Mouillot D. 2011. The multidimensionality of the niche reveals functional diversity changes in benthic marine biotas across geological time. Ecology Letters. 14, 561-568

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

Villéger, S. Grenouillet, G., Brosse, S. 2012. Decomposing functional beta-diversity reveals that low functional beta-diversity is driven by low functional turnover in European fish assemblages. Global Ecology and Biogeography, in press

See Also

functional.beta.multi, functional.beta.pair, betapart.core

Examples

```
##### 4 communities in a 2D functional space (convex hulls are rectangles)
traits.test<-cbind( c(1,1,1,2,2,3,3,4,4,5,5) , c(1,2,4,1,2,3,5,1,4,3,5) )
dimnames(traits.test)<-list(paste("sp",1:11,sep="") , c("Trait 1","Trait 2") )

comm.test<-matrix(0,4,11,dimnames=list( c("A","B","C","D") , paste("sp",1:11,sep="") ) )
comm.test["A",c(1,2,4,5)]<-1
```

```
comm.test["B",c(1,3,8,9)]<-1
comm.test["C",c(6,7,10,11)]<-1
comm.test["D",c(2,4,7,9)]<-1

plot(5,5,xlim=c(0,6), ylim=c(0,6), type="n", xlab="Trait 1",ylab="Trait 2")
points(traits.test[,1],traits.test[,2], pch=21,cex=1.5,bg="black")
rect(1,1,4,4, col="#458B0050", border="#458B00") ; text(2.5,2.5,"B",col="#458B00",cex=1.5)
polygon(c(2,1,3,4), c(1,2,5,4), col="#DA70D650", border="#DA70D6")
text(2.5,3,"D",col="#DA70D6",cex=1.5)
rect(1,1,2,2, col="#FF000050" , border="#FF0000") ; text(1.5,1.5,"A",col="#FF0000",cex=1.5)
rect(3,3,5,5, col="#1E90FF50", border="#1E90FF") ; text(4,4.2,"C",col="#1E90FF",cex=1.5)


test.core<-functional.betapart.core(x=comm.test, traits=traits.test, multi=TRUE,
return.details=FALSE)
test.core

# using functional.betapart.core to get details on intersections
# when only pairwise dissimilarity is computed
test.core.pair<-functional.betapart.core(x=comm.test, traits=traits.test, multi=FALSE,
return.details=TRUE)
test.core.pair

# for multiple dissimilarity
test.core.multi<-functional.betapart.core(x=comm.test, traits=traits.test, multi=TRUE,
return.details=TRUE)
test.core.multi

# using core outputs to compute pairwise and multiple functional dissimilarities
functional.beta.pair(x=test.core.pair, index.family = "jaccard" )
functional.beta.multi(x=test.core.multi, index.family = "jaccard" )


# Null model using parallel computing

# Example 1: pairwise functional beta diversity (functional.beta.pair)
# Note that this is an example with a small number of samples and null model
# permutations for illustration.
# Real null model analyses should have a much greater number of samples and permutations.

##### 4 communities in a 3D functional space

comm.test<-matrix(0,4,11,dimnames=list( c("A","B","C","D") , paste("sp", 1:11, sep = "")))
comm.test["A",c(1,2,4,5)]<-1
comm.test["B",c(1,3,8,9)]<-1
comm.test["C",c(6,7,10,11)]<-1
comm.test["D",c(2,4,7,9)]<-1

set.seed(1)
traits.test<-matrix(rnorm(11*3, mean=0, sd=1), 11, 3)
dimnames(traits.test)<-list(paste("sp", 1:11, sep="") , c("Trait 1","Trait 2", "Trait 3"))

# Required packages
```

```
require(doParallel)
require(picante)
require(fastmatch)
require(foreach)


# define number of cores
# Use parallel::detectCores() to determine number of cores available in your machine
nc <- 2

# create cluster
cl <- parallel::makeCluster(nc)

# register parallel backend
registerDoParallel(cl)

# define number of permutations for the null model (the usual is 1000)
# make sure that nperm/nc is a whole number so that all cores have the same number
# of permutations to work on
nperm <- 10

# compute observed values for multiple functional dissimilarities
test.score <- functional.betapart.core(x = comm.test, traits = traits.test,
multi = FALSE, warning.time = FALSE, return.details = FALSE,
fbc.step = FALSE, core.ident = NULL)

obs.pair.func.dis <- functional.beta.pair(x = test.score, index.family = "sorensen")

# number of possible pairs of sites
pairs <- combn(nrow(comm.test), 2)

# create strings for each pair of site
pair_names <- NULL
for (i in 1:ncol(pairs)){
  pair_id <- paste(attr(obs.pair.func.dis$funct.beta.sim, "Labels")[pairs[1,i]],
  attr(obs.pair.func.dis$funct.beta.sim, "Labels")[pairs[2,i]], sep="-")
  pair_names<-c(pair_names, pair_id)
}

# transform functional.beta.pair results into a matrix
obs.pair.func.dis <- do.call(rbind, lapply(obs.pair.func.dis, function(x) c(x)))

# set names for each pair of site using the set of strings created previsouly
colnames(obs.pair.func.dis) <- pair_names

# export necessary variables and functions to the cluster of cores
parallel::clusterExport(cl = cl, c("nperm", "nc", "comm.test", "traits.test"),
envir = environment())

null.pair.func.dis <- foreach::foreach(n = 1:nc, .combine = c,
.packages=c("picante","betapart","fastmatch")) %dopar% {

# number of tasks per core (i.e., permutations per core)
```

```
nt <- nperm/nc

#create a list of lists where the results from "nt" permutations will be stored
  null.pair.temp <- replicate(nt,list())

  # for each core "n" perform "nt" permutations
  for (j in 1:nt){

    # randomize community with chosen null model
    # for this particular example we used the "independent swap algorithm"
    # but the user can choose other types of permutation, or create it's own null model
    null.comm.test <- randomizeMatrix(comm.test, null.model="independentswap",
iterations=1000)

    # execute functional.betapart.core function identifying each "n" core with the
  # core.ident argument
    null.test.score <- try(functional.betapart.core(null.comm.test,
  traits = traits.test, multi = FALSE, warning.time = FALSE, return.details = FALSE,
  fbc.step = FALSE, core.ident = n), silent=TRUE)

    # in this artificial example there are a few combinations of species that the convex hull
  # cannot be calculated due to
    # some odd geometric combination so we need to re-permute the community matrix
    while(inherits(null.test.score, "try-error")){

    null.comm.test <- randomizeMatrix(comm.test, null.model="independentswap",
iterations=1000)
    null.test.score <- try(functional.betapart.core(x = null.comm.test,
traits = traits.test, multi = FALSE,
    warning.time = FALSE, return.details = FALSE, fbc.step = FALSE,
core.ident = n), silent=TRUE)
    }

    # compute the pairwise beta-diversity null values and input them in the temporary
# result matrix
    null.pair.temp[[j]] <- functional.beta.pair(x = null.test.score,
index.family = "sorensen" )

  }
  #retrieve the results from each core
  null.pair.temp
}

# stop cluster
parallel::stopCluster(cl)

# transform the results from each permutation into a matrix
null.pair.func.dis <- lapply(null.pair.func.dis, function(x) do.call(rbind,
lapply(x, function(x) c(x))))

# compute the mean, standard deviation and p-values of dissimilary metrics
# for each pair of site
```

```
mean.null.pair.func <-matrix(numeric(),ncol = ncol(obs.pair.func.dis),
nrow = nrow(obs.pair.func.dis))
sd.null.pair.func <-matrix(numeric(),ncol = ncol(obs.pair.func.dis),
nrow = nrow(obs.pair.func.dis))
p.pair.func.dis<-matrix(numeric(),ncol = ncol(obs.pair.func.dis),
nrow = nrow(obs.pair.func.dis))

# for each one of the 3 null dissimilarity metrics (SIN, SNE and SOR)
for (j in 1:nrow(obs.pair.func.dis)){

  #for each pair of sites
  for (i in 1:ncol(obs.pair.func.dis)){

# group the null values from each metric of each pair of site into into
# it's own matrix
null.pair.temp <- do.call(rbind, lapply(null.pair.func.dis, `[`,j,i))

# compute mean of all null values
mean.null.pair.func[j,i] <- mean(null.pair.temp)

# compute standard deviation of all null values
sd.null.pair.func[j,i] <- sd(null.pair.temp)

# compute the p-values
p.pair.func.dis[j,i] <-(length(which(obs.pair.func.dis[j,i]<null.pair.temp))+1)/nperm
# the +1 is to take into account that the observed value is one of the possibilities
  }
}

# compute standardized effect sizes
ses.pair.func.dis <- (obs.pair.func.dis - mean.null.pair.func)/sd.null.pair.func

# result matrix
result <- as.data.frame(matrix(numeric(), ncol = 9,nrow = ncol(obs.pair.func.dis),
    dimnames = list(pair_names, c("obs.SIM","ses.SIM","p.SIM","obs.SNE",
    "ses.SNE","p.SNE","obs.SOR","ses.SOR","p.SOR"))))

result$obs.SIM <- obs.pair.func.dis[1,]
result$obs.SNE <- obs.pair.func.dis[2,]
result$obs.SOR <- obs.pair.func.dis[3,]

result$ses.SIM <- ses.pair.func.dis[1,]
result$ses.SNE <- ses.pair.func.dis[2,]
result$ses.SOR <- ses.pair.func.dis[3,]

result$p.SIM <- p.pair.func.dis[1,]
result$p.SNE <- p.pair.func.dis[2,]
result$p.SOR <- p.pair.func.dis[3,]

result

## Not run:
# Example 2: multiple functional beta diversity (functional.beta.multi)
```

```
# Note that this is an example with a small number of samples and null model
# permutations for illustration.
# Real null model analyses should have a much greater number of samples
# and permutations.

##### 4 communities in a 3D functional space

comm.test<-matrix(0,4,11,dimnames=list( c("A","B","C","D"),
paste("sp", 1:11, sep = "")))
comm.test["A",c(1,2,4,5)]<-1
comm.test["B",c(1,3,8,9)]<-1
comm.test["C",c(6,7,10,11)]<-1
comm.test["D",c(2,4,7,9)]<-1

set.seed(1)
traits.test<-matrix(rnorm(11*3, mean=0, sd=1), 11, 3)
dimnames(traits.test)<-list(paste("sp", 1:11, sep=""),
c("Trait 1","Trait 2", "Trait 3"))

# Required packages
require(doParallel)
require(picante)
require(fastmatch)
require(foreach)


# define number of cores
# Use parallel::detectCores() to determine number of cores available in your machine
nc <- 2

# create cluster
cl <- parallel::makeCluster(nc)

# register parallel backend
registerDoParallel(cl)

# define number of permutations for the null model (the usual is 1000)
# make sure that nperm/nc is a whole number so that all cores have the same number
# of permutations to work on
nperm <- 10

# compute observed values for multiple functional dissimilarities
test.score <- functional.betapart.core(x = comm.test, traits = traits.test,
multi = TRUE, warning.time = FALSE, return.details = FALSE, fbc.step = FALSE,
core.ident = NULL)
obs.multi.func.dis <- t(as.matrix(unlist(functional.beta.multi(x=test.score,
index.family = "sorensen"))))

# export necessary variables and functions to the cluster of cores
parallel::clusterExport(cl = cl, c("nperm", "nc", "comm.test", "traits.test"),
envir=environment())

null.multi.func.dis <- foreach::foreach(n = 1:nc, .combine = rbind,
```

```
    .packages = c("picante","betapart","fastmatch")) %dopar% {

  # for each core, create temporary matrix to store 3 null multiple functional
  # dissimilarity indices (SIN, SNE,SOR)
  null.multi.temp <- matrix(numeric(), ncol = 3, nrow = nperm/nc,
  dimnames = list(NULL, c("funct.beta.SIM", "funct.beta.SNE", "funct.beta.SOR")))

  # number of tasks per core (i.e., permutations per core)
  nt <- nperm/nc

      # for each core "n" perform "nt" permutations
      for (j in 1:nt) {

      # randomize community matrix with chosen null model (for this example
  # we chose the "independent swap" algorithm)
      null.comm.test <- picante::randomizeMatrix(comm.test,
  null.model="independentswap", iterations=1000)

      # execute functional.betapart.core function identifying each "n" core
      # with the core.ident argument for external parallelization,
      null.test.score <- try(functional.betapart.core.modif(x = null.comm.test,
  traits = traits.test,
      multi = TRUE,  warning.time = FALSE, return.details = FALSE, fbc.step = FALSE,
  core.ident = n), silent = TRUE)

      # in this artificial example there are a few combinations of species
  # that the convex hull
      # cannot be calculated due to some odd geometric combination so we
  # need to re-permute the community matrix

      while(inherits(null.test.score, "try-error")){
      null.comm.test <- randomizeMatrix(comm.test, null.model="independentswap",
  iterations=1000)
      null.test.score <- try(functional.betapart.core(x = null.comm.test,
  traits = traits.test,
  multi = TRUE,  warning.time = FALSE, return.details = FALSE, fbc.step = FALSE,
  core.ident = n), silent = TRUE)
                        }
      # input null values in the temporary result matrix
      null.multi.temp[j,] <- unlist(functional.beta.multi(x = null.test.score,
  index.family = "sorensen"))
      }
# recover results from each core
null.multi.temp
}

# close cluster
parallel::stopCluster(cl)

# result matrix
result <- matrix(numeric(), ncol = 3, nrow = 3, dimnames = list(c("obs","ses","p"),
colnames(obs.multi.func.dis)))
```

```
# input observed values for the multiple functional dissimilarity indices (SIN, SNE,SOR)
result[1,] = obs.multi.func.dis

# compute standardized effect sizes (ses) for the multiple functional
# dissimilarity indices (SIN, SNE,SOR)
result[2,] <- (obs.multi.func.dis-colMeans(null.multi.func.dis, na.rm=TRUE))/
apply(null.multi.func.dis,2,sd, na.rm=TRUE)

# compute p-values for the multiple functional dissimilarity indices (SIN, SNE,SOR)
# SIM
result[3,1] <- (length(which(obs.multi.func.dis[1]<null.multi.func.dis[,1])) + 1)/nperm
# SNE
result[3,2] <- (length(which(obs.multi.func.dis[2]<null.multi.func.dis[,2])) + 1)/nperm
# SOR
result[3,3] <- (length(which(obs.multi.func.dis[3]<null.multi.func.dis[,3])) + 1)/nperm
# the +1 is to take into account that the observed value is one of the possibilities

result
###

## End(Not run)
```

---

phylo.beta.multi   *Multiple-site phylogenetic dissimilarities*

Description

Computes 3 distance values accounting for the multiple-site phylogenetic turnover and nestedness components of phylogenetic beta diversity, and the sum of both values. Phylogenetic dissimilarities are based on Faith's phylogenetic diversity.

Usage

```
phylo.beta.multi(x, tree, index.family="sorensen")
```

Arguments

x	a community matrix or data frame, where rows are sites and columns are species. Alternatively x can be a phylo.betapart object derived from the phylo.betapart.core function
tree	a phylogenetic tree of class phylo with tips names identic to species names from the community matrix.
index.family	family of dissimilarity indices, partial match of "sorensen" or "jaccard".

Details

The Sorensen dissimilarity index allows computing the PhyloSor index (Bryant et al. 2008) whereas the Jaccard dissimilarity index allows computing the UniFrac index (Lozupone & Knight 2005).

Value

The function returns a list with three phylogenetic dissimilarity values.

For `index.family="sorensen"` the three values are:

phylo.beta.sim   `dist` object, dissimilarity value accounting for phylogenetic turnover, measured as Simpson derived multiple-site phylogenetic dissimilarity

phylo.beta.sne   `dist` object, dissimilarity value accounting for nestedness-resultant phylogenetic dissimilarity, measured as the nestedness-fraction of Sorensen derived multiple-site phylogenetic dissimilarity

phylo.beta.sor   `dist` object, dissimilarity value accounting for phylogenetic beta diversity, measured as Sorensen derived multiple-site phylogenetic dissimilarity

For `index.family="jaccard"` the three values are:

phylo.beta.jtu   `dist` object, dissimilarity value accounting for phylogenetic turnover, measured as the turnover-fraction of Jaccard derived multiple-site phylogenetic dissimilarity

phylo.beta.jne   `dist` object, dissimilarity value accounting for nestedness-resultant phylogenetic dissimilarity, measured as the nestedness-fraction of Jaccard derived multiple-site phylogenetic dissimilarity

phylo.beta.jac   `dist` object, dissimilarity value accounting for phylogenetic beta diversity, measured as Jaccard derived multiple-site phylogenetic dissimilarity

Author(s)

Julien De Bortoli (juldebortoli@yahoo.fr), Fabien Leprieur(fabien.leprieur@univ-montp2.fr), Andrés Baselga and David Orme

References

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

Bryant JA, Lamanna C, Morlon H, Kerkhoff AJ, Enquist BJ, et al. (2008) Microbes on mountainsides: Contrasting elevational patterns of bacterial and plant diversity. Proceedings of the National Academy of Sciences of the United States of America 105: 11505-11511.

Faith DP, Lozupone CA, Nipperess D, Knight R (2009) The Cladistic Basis for the Phylogenetic Diversity (PD) Measure Links Evolutionary Features to Environmental Gradients and Supports Broad Applications of Microbial Ecology's "Phylogenetic Beta Diversity" Framework. Int J Mol Sci 10: 4723-4741. doi: 10.3390/ijms10114723.

Leprieur F, Albouy C, De Bortoli J, Cowman PF, Bellwood DR, et al. (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

Lozupone C, Knight R (2005) UniFrac: a new phylogenetic method for comparing microbial communities. Applied and Environmental Microbiology 71: 8228-8235.

See Also

phylo.betapart.core, beta.multi

Examples

```
# toy tree for 6 species (sp1 to sp6)
require(ape)
toy.tree<-read.tree(text="(((sp1:1,sp2:1):5,(sp3:3,sp4:3):3):2,(sp5:7,sp6:7):1);")
plot(toy.tree)

# toy community table with 6 assemblages (A to F) with 6 species (sp1 to sp6)
toy.comm<-matrix(nrow=6, ncol=6)
rownames(toy.comm)<-c("A","B","C","D","E","F")
colnames(toy.comm)<-c("sp1","sp2","sp3","sp4","sp5","sp6")
toy.comm[1,]<-c(1,1,1,0,0,0)
toy.comm[2,]<-c(0,1,1,1,0,0)
toy.comm[3,]<-c(0,0,1,1,1,0)
toy.comm[4,]<-c(0,0,1,1,1,1)
toy.comm[5,]<-c(0,0,0,1,1,1)
toy.comm[6,]<-c(1,0,0,1,1,1)

toy.phylobetamulti<-phylo.beta.multi(toy.comm, toy.tree, index.family="sor")
toy.betamulti<-beta.multi(toy.comm, index.family="sor")
```

---

phylo.beta.pair   *Pair-wise phylogenetic dissimilarities*

Description

Computes 3 distance matrices accounting for the phylogenetic turnover and nestedness components of phylogenetic beta diversity, and the sum of both values. Phylogenetic dissimilarities are based on Faith's phylogenetic diversity.

Usage

```
phylo.beta.pair(x, tree, index.family="sorensen")
```

Arguments

x	a community matrix or data frame, where rows are sites and columns are species. Alternatively x can be a phylo.betapart object derived from the phylo.betapart.core function
tree	a phylogenetic tree of class phylo with tips names identic to species names from the community matrix.
index.family	family of dissimilarity indices, partial match of "sorensen" or "jaccard".

Details

The Sorensen dissimilarity index allows computing the PhyloSor index (Bryant et al. 2008) whereas the Jaccard dissimilarity index allows computing the UniFrac index (Lozupone & Knight 2005).

Value

The function returns a list with three phylogenetic dissimilarity matrices.

For `index.family="sorensen"` the three matrices are:

`phylo.beta.sim` dist object, dissimilarity matrix accounting for phylogenetic turnover, measured as Simpson derived pair-wise phylogenetic dissimilarity

`phylo.beta.sne` dist object, dissimilarity matrix accounting for nestedness-resultant phylogenetic dissimilarity, measured as the nestedness-fraction of Sorensen derived pair-wise phylogenetic dissimilarity

`phylo.beta.sor` dist object, dissimilarity matrix accounting for phylogenetic beta diversity, measured as Sorensen derived pair-wise phylogenetic dissimilarity

For `index.family="jaccard"` the three matrices are:

`phylo.beta.jtu` dist object, dissimilarity matrix accounting for phylogenetic turnover, measured as the turnover-fraction of Jaccard derived pair-wise phylogenetic dissimilarity

`phylo.beta.jne` dist object, dissimilarity matrix accounting for nestedness-resultant phylogenetic dissimilarity, measured as the nestedness-fraction of Jaccard derived pair-wise phylogenetic dissimilarity

`phylo.beta.jac` dist object, dissimilarity matrix accounting for phylogenetic beta diversity, measured as Jaccard derived pair-wise phylogenetic dissimilarity

Author(s)

Julien De Bortoli (juldebortoli@yahoo.fr), Fabien Leprieur(fabien.leprieur@univ-montp2.fr), Andrés Baselga and David Orme

References

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

Bryant JA, Lamanna C, Morlon H, Kerkhoff AJ, Enquist BJ, et al. (2008) Microbes on mountainsides: Contrasting elevational patterns of bacterial and plant diversity. Proceedings of the National Academy of Sciences of the United States of America 105: 11505-11511.

Faith DP, Lozupone CA, Nipperess D, Knight R (2009) The Cladistic Basis for the Phylogenetic Diversity (PD) Measure Links Evolutionary Features to Environmental Gradients and Supports Broad Applications of Microbial Ecology's "Phylogenetic Beta Diversity" Framework. Int J Mol Sci 10: 4723-4741. doi: 10.3390/ijms10114723.

Leprieur F, Albouy C, De Bortoli J, Cowman PF, Bellwood DR, et al. (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

Lozupone C, Knight R (2005) UniFrac: a new phylogenetic method for comparing microbial communities. Applied and Environmental Microbiology 71: 8228-8235.

See Also

[phylo.betapart.core](#), [beta.pair](#)

Examples

```
# toy tree for 6 species (sp1 to sp6)
require(ape)
toy.tree<-read.tree(text="(((sp1:1,sp2:1):5,(sp3:3,sp4:3):3):2,(sp5:7,sp6:7):1);")
plot(toy.tree)

# toy community table with 6 assemblages (A to F) with 6 species (sp1 to sp6)
toy.comm<-matrix(nrow=6, ncol=6)
rownames(toy.comm)<-c("A","B","C","D","E","F")
colnames(toy.comm)<-c("sp1","sp2","sp3","sp4","sp5","sp6")
toy.comm[1,]<-c(1,1,1,0,0,0)
toy.comm[2,]<-c(0,1,1,1,0,0)
toy.comm[3,]<-c(0,0,1,1,1,0)
toy.comm[4,]<-c(0,0,1,1,1,1)
toy.comm[5,]<-c(0,0,0,1,1,1)
toy.comm[6,]<-c(1,0,0,1,1,1)


toy.phylobetapair<-phylo.beta.pair(toy.comm, toy.tree, index.family="sor")
toy.betapair<-beta.pair(toy.comm, index.family="sor")
plot(toy.betapair$beta.sim,toy.phylobetapair$phylo.beta.sim)
plot(toy.betapair$beta.sne,toy.phylobetapair$phylo.beta.sne)
```

---

phylo.betapart.core   *Core calculations of phylogenetic dissimilarities metrics*

Description

Computes the basic quantities needed for computing the multiple-site phylogenetic beta diversity measures and pairwise phylogenetic dissimilarity matrices.

Usage

```
phylo.betapart.core(x, tree)
```

Arguments

x	a community matrix or data frame, where rows are sites and columns are species.
tree	a phylogenetic tree of class phylo with tips names identic to species names from the community matrix.

Value

The function returns a list with:

sumSi	the sum of the phylogenetic diversity values of all sites
St	the total phylogenetic diversity in the dataset
shared	a matrix containing the phylogenetic diversity shared between pairs of sites
sum.not.shared	a matrix containing the total phylogenetic diversity not shared between pairs of sites: b+c
max.not.shared	a matrix containing the total maximum phylogenetic diversity not shared between pairs of sites: max(b,c)
min.not.shared	a matrix containing the total minimum phylogenetic diversity not shared between pairs of sites: min(b,c)

Author(s)

Julien De Bortoli (juldebortoli@yahoo.fr), Fabien Leprieur(fabien.leprieur@univ-montp2.fr), Andrés Baselga and David Orme

References

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

Bryant JA, Lamanna C, Morlon H, Kerkhoff AJ, Enquist BJ, et al. (2008) Microbes on mountainsides: Contrasting elevational patterns of bacterial and plant diversity. Proceedings of the National Academy of Sciences of the United States of America 105: 11505-11511.

Faith DP, Lozupone CA, Nipperess D, Knight R (2009) The Cladistic Basis for the Phylogenetic Diversity (PD) Measure Links Evolutionary Features to Environmental Gradients and Supports Broad Applications of Microbial Ecology's "Phylogenetic Beta Diversity" Framework. Int J Mol Sci 10: 4723-4741. doi: 10.3390/ijms10114723.

Leprieur F, Albouy C, De Bortoli J, Cowman PF, Bellwood DR, et al. (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

Lozupone C, Knight R (2005) UniFrac: a new phylogenetic method for comparing microbial communities. Applied and Environmental Microbiology 71: 8228-8235.

See Also

phylo.beta.pair, phylo.beta.multi

Examples

```
# toy tree for 6 species (sp1 to sp6)
require(ape)
toy.tree<-read.tree(text="(((sp1:1,sp2:1):5,(sp3:3,sp4:3):3):2,(sp5:7,sp6:7):1);")
plot(toy.tree)
```

```
# toy community table with 6 assemblages (A to F) with 6 species (sp1 to sp6)
toy.comm<-matrix(nrow=6, ncol=6)
rownames(toy.comm)<-c("A","B","C","D","E","F")
colnames(toy.comm)<-c("sp1","sp2","sp3","sp4","sp5","sp6")
toy.comm[1,]<-c(1,1,1,0,0,0)
toy.comm[2,]<-c(0,1,1,1,0,0)
toy.comm[3,]<-c(0,0,1,1,1,0)
toy.comm[4,]<-c(0,0,1,1,1,1)
toy.comm[5,]<-c(0,0,0,1,1,1)
toy.comm[6,]<-c(1,0,0,1,1,1)

toy.phylocore<-phylo.betapart.core(toy.comm, toy.tree)
```

---

plot.decay	*Plotting distance decay curves from models computed with decay.model()*

Description

Takes the output of decay.model() and plots a distance-decay curve, either a negative exponential or power law function as estimated with decay.model().

Usage

```
## S3 method for class 'decay'
plot(x, xlim=c(0,max(x$data[,1])), ylim=c(0,1),
add=FALSE, remove.dots=FALSE, col="black", pch=1, lty=1, lwd=5, cex=1, ...)
```

Arguments

x	the output of decay.model().
xlim	the range of spatial distances to be plotted, default is from 0 to the maximum distance in the data.
ylim	the range of assemblage similarities or dissimilarities to be plotted, default is from 0 to 1.
add	add to the previous plot.
remove.dots	remove the dots from the plot, thus retaining just the decay curve.
col	colour used.
pch	symbol used for points.
lty	line type.
lwd	line width.
cex	scale of text and symbols.
...	other parameters for plotting functions.

Author(s)

Andrés Baselga

References

Gómez-Rodríguez, C. & Baselga, A. 2018. Variation among European beetle taxa in patterns of distance decay of similarity suggests a major role of dispersal processes. Ecography, in press

See Also

decay.model

Examples

```
require(vegan)

data(BCI)
## UTM Coordinates (in metres)
UTM.EW <- rep(seq(625754, 626654, by=100), each=5)
UTM.NS <- rep(seq(1011569,  1011969, by=100), len=50)

spat.dist<-dist(data.frame(UTM.EW, UTM.NS))

dissim.BCI<-beta.pair.abund(BCI)$beta.bray.bal

plot(spat.dist, dissim.BCI, ylim=c(0,1), xlim=c(0, max(spat.dist)))

BCI.decay.exp<-decay.model(dissim.BCI, spat.dist, model.type="exp", perm=100)

BCI.decay.pow<-decay.model(dissim.BCI, spat.dist, model.type="pow", perm=100)

plot.decay(BCI.decay.exp, col=rgb(0,0,0,0.5))
plot.decay(BCI.decay.exp, col="red", remove.dots=TRUE, add=TRUE)
plot.decay(BCI.decay.pow, col="blue", remove.dots=TRUE, add=TRUE)
```

Index