Package ‘betapart’

February 22, 2018

Type Package

Title Partitioning Beta Diversity into Turnover and Nestedness Components

Version 1.5.0

Date 2018-02-22

Author Andres Baselga, David Orme, Sebastien Villeger, Julien De Bortoli and Fabien Leprieur

Maintainer Andres Baselga <andres.base1ga@usc.es>

Imports ape, geometry, picante, rcdd

Suggests 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)

R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>bbsData</td>
<td>2</td>
</tr>
<tr>
<td>beta.multi</td>
<td>3</td>
</tr>
<tr>
<td>beta.multi.abund</td>
<td>4</td>
</tr>
<tr>
<td>beta.pair</td>
<td>5</td>
</tr>
<tr>
<td>beta.pair.abund</td>
<td>6</td>
</tr>
<tr>
<td>beta.sample</td>
<td>8</td>
</tr>
<tr>
<td>beta.sample.abund</td>
<td>9</td>
</tr>
<tr>
<td>beta.temp</td>
<td>11</td>
</tr>
<tr>
<td>betapart</td>
<td>12</td>
</tr>
<tr>
<td>betapart.core</td>
<td>13</td>
</tr>
<tr>
<td>betapart.core.abund</td>
<td>14</td>
</tr>
</tbody>
</table>
bbsData

2

boot.coefs.decay .............................................................. 16
bray.part ................................................................. 17
ceram.n ................................................................. 18
ceram.s ................................................................. 19
decay.model .......................................................... 19
functional.beta.multi .................................................. 21
functional.beta.pair .................................................. 23
functional.betapart.core .............................................. 25
phylo.beta.multi ....................................................... 28
phylo.beta.pair ....................................................... 30
phylo.betapart.core .................................................... 32
plot.decay .............................................................. 34

Index

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)
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
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
beta.pair

Author(s)
Andrés Baselga

References

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

See Also
beta.pair.abund, codebeta.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".
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


See Also

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

Examples

```r
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


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
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 \texttt{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


See Also

\texttt{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**

```r
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**


**See Also**

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

**Examples**

```r
data(bbsdata)
bbs.t <- beta.temp(bbs1980, bbs2000, index.family="sor")
```
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.beta.pair` and `functional.beta.multi`, and for phylogenetic beta diversity with commands `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 parameters of the distance-decay model.

Author(s)

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

References


---

**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.
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

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")

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.core.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: \( \max(B,C) \)
- `pair.not.shared.abund`: a matrix containing the total disagreement in abundance between pairs of sites: \( B+C \)

Author(s)

Andrés Baselga

References


See Also

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

Examples

```r
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**

```r
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`: similarities 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**


**See Also**

- `decay.model`
**Examples**

```r
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, 100)
```

---

**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**

```r
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 dissimilarity between sites, measured as the Bray-Curtis index.

**Author(s)**

Andrés Baselga
References


See Also

beta.pair

Examples

```r
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

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

Source

http://www.cerambycidae.net/

References


**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**

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

**Source**

http://www.cerambycidae.net/

**References**


**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**

```r
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 dissimilarities).
model  the fitted GLM.
model.type  functional form of the model, either negative exponential or power law.
y.type  similarities or dissimilarities.
pseudo.r.squared  similarities 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 dissimilarity 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

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
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( (An)\cap (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").

**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**


**See Also**

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

##### 4 communities in a 2D functional space (convex hulls are rectangles)

```r
traits.test <- cbind(c(1,1,2,3,4,4,5,5), 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"))
```

```r
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
```

```r
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="#45800050", border="#458000")
text(2.5,2.5,"B", col="#458000", 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="#E90FF50", border="#E90FF")
text(4.4,2,"C", col="#E90FF", cex=1.5)
```

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

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

```r
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**

```r
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. quantitive 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".

Value

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

funct.beta.sin 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


See Also

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

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

comm.test <- matrix(0,4,4,dimnames=list(c("A","B","C","D"),paste("sp",1:4,sep="")))
comm.test["A",c(1,2,3,4)] <- 1
comm.test["B",c(1,3,6,9)] <- 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="#45800058", border="#458000")
rect(2.5,2.5, "B", col="#458000", cex=1.5)
rect(1,1,4,4, col="#DA70D650", border="#DA70D6")
rect(2.5,2.5, "D", col="#DA70D6", cex=1.5)

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="#FF000058", border="#FF0000")
rect(2.5,2.5, "B", col="#FF0000", cex=1.5)
rect(1,1,4,4, col="#E90FF50", border="#E90FF")
rect(2.5,2.5, "D", col="#E90FF", cex=1.5)

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="#1E00FF58", border="#1E00FF")
rect(2.5,2.5, "B", col="#1E00FF", cex=1.5)

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="#0000FF58", border="#0000FF")
rect(2.5,2.5, "B", col="#0000FF", cex=1.5)
```

---

**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.

### Usage

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

### 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.

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").

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: FRi 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 and David Orme
References


See Also

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

Examples

```r
# 4 communities in a 2D functional space (convex hulls are rectangles)
traits <- cbind( c(1,1,1,2,2,3,3,4,4,4,5,5) , c(1,2,4,1,2,3,5,1,4,3,5) )
dimnames(traits) <- 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(0,1), ylim=c(0,1), type="n", xlab="Trait 1",ylab="Trait 2")
points(comm.test[,1],traits.test[,2], pch=21, cex=1.5, bg="black")
rect(1,1,4,4, col="#458B0050", border="#458B0040" )
rect(1,1,2,2, col="#FF000050", border="#FF000040" )
rect(3,3,5,5, col="#1E90FF50", border="#1E90FF40" )

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")
```

---

**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.beta.part object derived from the phylo.beta.part.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.multi

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


See Also

phylo.betapart.core, beta.multi

Examples

# toy tree for 6 species (sp1 to sp6)
require(ape)
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(1,1,0,0,0)
**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**

```r
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

```r
toy.comm[2,]<-c(0,1,1,0,0)
toy.comm[3,]<-c(0,0,1,1,0)
toy.comm[4,]<-c(0,0,1,1,1)
toy.comm[5,]<-c(0,0,1,1,1)
toy.comm[6,]<-c(1,0,0,1,1)
toy.phylobetamulti<-phylo.beta.multi(toy.comm, toy.tree, index.family="sor")
toy.betamulti<-beta.multi(toy.comm, index.family="sor")
```
phylo.beta.pair

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


See Also

phylo.betapart.core, beta.pair

Examples

# toy tree for 6 species (sp1 to sp6)
require(ape)
plot(toy.tree)

# toy community table with 6 assemblages (A to F) with 6 species (sp1 to sp6)
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 identical 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


See Also

phylo.beta.pair, phylo.beta.multi

Examples

# toy tree for 6 species (sp1 to sp6)
require(ape)
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)
toy.comm[2,]<-c(0,1,1,1,0)
toy.comm[3,]<-c(0,0,1,1,1)
toy.comm[4,]<-c(0,0,1,1,1)
toy.comm[5,]<-c(0,0,1,1,1)
toy.comm[6,]<-c(1,0,1,1,1)
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

```r
## 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


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(0,0,0.5))
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

*Topic datasets
  bbsData, 2

bbs1980 (bbsData), 2
bbs2000 (bbsData), 2
bbsData, 2
beta.multi, 3, 5, 6, 8, 9, 11, 14, 22, 29
beta.multi.abund, 4, 7, 10, 15
beta.pair, 4, 5, 7, 11, 14, 18, 20, 24, 31
beta.pair.abund, 5, 6, 6, 15, 20
beta.sample, 4, 6, 8, 9–11, 14
beta.sample.abund, 5, 7, 9, 15
beta.temp, 4, 6, 9, 11, 14
betapart, 12
betapart-package (betapart), 12
betapart.core, 4, 6, 11, 13, 15, 27
betapart.core.abund, 5, 7, 14
boot.coefs.decay, 16
bray.part, 17

ceram.n, 18
ceram.s, 19
decay.model, 16, 19, 34

functional.beta.multi, 21, 24, 27
functional.beta.pair, 22, 23, 27
functional.betapart.core, 22, 24, 25

phylo.beta.multi, 28, 33
phylo.beta.pair, 30, 33
phylo.betapart.core, 29, 31, 32
plot.decay, 34