Package ‘entropart’

April 11, 2019

Type Package

Title Entropy Partitioning to Measure Diversity

Version 1.6-1

‘entropart’ provides functions to calculate alpha, beta and gamma diversity of communities, including phylogenetic and functional diversity.
Estimation-bias corrections are available.

URL https://github.com/EricMarcon/entropart

BugReports https://github.com/EricMarcon/entropart/issues

License GNU General Public License

Depends ggplot2

Imports ade4, ape, EntropyEstimation, geiger, ggpubr, graphics, grDevices, parallel, reshape2, stats, utils, vegan

Suggests testthat, knitr, pkgdown

LazyData true

VignetteBuilder knitr

SystemRequirements pandoc

Encoding UTF-8

RoxygenNote 6.1.1

NeedsCompilation no

Author Eric Marcon [aut, cre] (<https://orcid.org/0000-0002-5249-321X>), Bruno Herault [aut] (<https://orcid.org/0000-0002-6950-7286>)

Maintainer Eric Marcon <eric.marcon@ecofog.gf>

Repository CRAN

Date/Publication 2019-04-11 20:32:46 UTC
### R topics documented:

- entropart-package ........................................ 3
- AbdFreqCount ........................................... 4
- Accumulation ............................................. 6
- AllenH ..................................................... 9
- Alpha Diversity .......................................... 10
- Alpha Entropy ........................................... 12
- Beta Diversity ........................................... 13
- Beta Entropy .............................................. 15
- ChaoPD ..................................................... 16
- Community Profile ....................................... 18
- Coverage ................................................... 20
- Diversity .................................................. 22
- Div Est ..................................................... 24
- Div Part .................................................... 26
- Div Profile ............................................... 29
- Dqz ......................................................... 32
- Eight Sp Abundance ...................................... 34
- Eight Sp Tree ............................................ 34
- Enq .......................................................... 35
- Entropy CI .................................................. 36
- expq ........................................................ 37
- Gamma Diversity .......................................... 38
- Gamma Entropy ............................................ 40
- Gen Simpson ............................................... 41
- Hqz .......................................................... 43
- HqzBeta ..................................................... 46
- Hurlbert ..................................................... 48
- KLpq ........................................................ 50
- lnq .......................................................... 51
- MC Utilities ............................................... 52
- MC Diversity .............................................. 53
- MC Entropy ............................................... 55
- Meta Community .......................................... 56
- Optimal Similarity ....................................... 58
- Paracou618 dist ........................................... 59
- Paracou618 Functional .................................... 60
- Paracou618 MC ............................................ 61
- Paracou618 Taxonomy ..................................... 61
- PDFD ......................................................... 62
- Phylo Apply ............................................... 63
- Phylo Beta Entropy ........................................ 65
- Phylo Diversity ........................................... 67
- Phylo Entropy .............................................. 70
- Phylo Value ............................................... 73
- PP tree ...................................................... 74
- RAC ........................................................ 75
Description

Functions to calculate alpha, beta and gamma diversity of communities, including phylogenetic and functional diversity.

Estimation-bias corrections are available.

Details

In the entropart package, individuals of different "species" are counted in several "communities" which may (or not) be aggregated to define a "metacommunity". In the metacommunity, the probability to find a species in the weighted average of probabilities in communities. This is a naming convention, which may correspond to plots in a forest inventory or any data organized the same way.

Basic functions allow computing diversity of a community. Data is simply a vector of probabilities (summing up to 1) or of abundances (integer values that are numbers of individuals). Calculate entropy with functions such as `tsallis`, `shannon`, `simpson`, `hurlbert` or `gensimpson` and explicit diversity (i.e. effective number of species) with `diversity` and others. By default, the best available estimator of diversity will be used, according to the data.

Communities can be simulated by `rCommunity`, explicitely declared as a species distribution (`as.AbdVector` or `as.ProbaVector`), and plotted.

Phylogenetic entropy and diversity can be calculated if a phylogenetic (or functional), ultrametric tree is provided. See `PhyloEntropy`, `Rao` for examples of entropy and `PhyloDiversity` to calculate phylodiversity, with the state-of-the-art estimation-bias correction. Similarity-based diversity is calculated with `Dqz`, based on a similarity matrix.

The simplest way to import data is to organize it into two text files. The first file should contain abundance data: the first column named `Species` for species names, and a column for each community.

The second file should contain the community weights in two columns. The first one, named `Weights`, their weights.

Files can be read and data imported by code such as:
AbdFreqCount <- read.csv(file="Abundances.csv", row.names = 1)
Weights <- read.csv(file="Weights.csv")
MC <- MetaCommunity(Abundances, Weights)

The last line of the code calls the MetaCommunity function to create an object that will be used by all metacommunity functions, such as DivPart (to partition diversity), DivEst (to partition diversity and calculate confidence interval of its estimation) or DivProfile (to compute diversity profiles).

A full documentation is available in the vignette. Type: vignette("entropart"). A quick introduction is in vignette("introduction", "entropart").

Author(s)

Eric Marcon, Bruno Herault

References


---

AbdFreqCount | Abundance Frequency Count of a Community

Description

Counts the number of species observed the same number of times.

Usage

AbdFreqCount(Ns, Level = NULL, PCorrection="Chao2015", Unveiling="geom", RCorrection="Rarefy", CheckArguments = TRUE)
AbdFreqCount

Arguments

- **Ns**
  A numeric vector containing species abundances.

- **Level**
  The level of interpolation or extrapolation. It may be an an arbitrary sample size (an integer) or a sample coverage (a number between 0 and 1).

- **PCorrection**
  A string containing one of the possible corrections to estimate a probability distribution in `as.ProbaVector`: "Chao2015" is the default value. Used only for extrapolation.

- **Unveiling**
  A string containing one of the possible unveiling methods to estimate the probabilities of the unobserved species in `as.ProbaVector`: "geom" (geometric: the unobserved species distribution is geometric) is the default value. Used only for extrapolation.

- **RCorrection**
  A string containing a correction recognized by `Richness` to evaluate the total number of species in `as.ProbaVector`. "Rarefy" is the default value to estimate the number of species such that the richness of the asymptotic distribution rarefied to the observed sample size equals the observed number of species in the data. Used only for extrapolation.

- **CheckArguments**
  Logical; if `TRUE`, the function arguments are verified. Should be set to `FALSE` to save time when the arguments have been checked elsewhere.

Details

The Abundance Frequency Count (Chao et al., 2015) is the number of species observed each number of times.

It is a way to summarize the species distribution.

It can be estimated at a specified level of interpolation or extrapolation. Extrapolation relies on the estimation of the estimation of the asymptotic distribution of the community by `as.ProbaVector` and eq. (5) of Chao et al. (2014).

Value

A two-column matrix. The first column contains the number of observations, the second one the number of species observed this number of times.

Author(s)

Eric Marcon <Eric.Maron@ecofog.gf>

References


See Also

PhyloEntropy, ChaoPD

Examples

# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest
# and their taxonomy)
data(Paracou618)
# Ns is the vector of abundances of the first plot
Ns = Paracou618.MC$Ns[, 1]

# Return the abundance frequency count
(AbdFreqCount(Ns) -> afc)
plot(afc, xlab="Number of observations", ylab="Number of species")
lines(afc)

Accumulation

Diversity accumulation.

Description

Diversity and Entropy Accumulation Curves represent the accumulation of entropy with respect to
the sample size.

Usage

as.AccumCurve(x, y, low = NULL, high = NULL)
is.AccumCurve(x)
EntAC(Ns, q = 0, n.seq = 1:sum(Ns), PCorrection="Chao2015", Unveiling="geom",
RCorrection="Rarefy", NumberOfSimulations = 0, Alpha = 0.05,
ShowProgressBar = TRUE, CheckArguments = TRUE)
DivAC(Ns, q = 0, n.seq = 1:sum(Ns), PCorrection="Chao2015", Unveiling="geom",
RCorrection="Rarefy", NumberOfSimulations = 0, Alpha = 0.05,
ShowProgressBar = TRUE, CheckArguments = TRUE)

Arguments

x An object. A numeric vector in as.AccumCurve.
object An object.
**Accumulation**

\[ y \] A numeric vector.

\[ low \] A numeric vector.

\[ high \] A numeric vector.

\[ Ns \] A numeric vector containing species abundances.

\[ q \] A number: the order of diversity. Default is 1.

\[ n.seq \] A sequence of numbers. Accumulation will be calculated at each value.

**PCorrection** A string containing one of the possible corrections to estimate a probability distribution in `as.ProbaVector`: "Chao2015" is the default value. Used only for extrapolation and \( q \) different from 0, 1, 2.

**Unveiling** A string containing one of the possible unveiling methods to estimate the probabilities of the unobserved species in `as.ProbaVector`: "geom" (geometric: the unobserved species distribution is geometric) is the default value. Used only for extrapolation and \( q \) different from 0, 1, 2.

**RCorrection** A string containing a correction recognized by `Richness` to evaluate the total number of species in `as.ProbaVector`. "Rarefy" is the default value to estimate the number of species such that the entropy of the asymptotic distribution rarefied to the observed sample size equals the observed entropy of the data. Used only for extrapolation and \( q \) different from 0, 1, 2. If \( q \) is 0 (extrapolation of richness), "Rarefy" is taken for "Jackknife".

**NumberOfSimulations** The number of Simulations to build confidence intervals.

**Alpha** The risk level, 5% by default.

**main** The main title of the plot. If NULL (by default), there is no title.

**xlab** The X axis label, "Rank" by default.

**ylab** The Y axis label. If NULL (by default), "Probability" or "Abundance" is chosen according to the object class.

**ylim** The interval of y values plotted.

**LineWidth** The width of the line that represents the actual profile.

**ShadeColor** The color of the shaded confidence envelope.

**BorderColor** The color of the bounds of the confidence envelope.

**alpha** Opacity of the confidence enveloppe, between 0 and 1.

**ShowProgressBar** If TRUE (default), a progress bar is shown.

**CheckArguments** Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

**Details**

`DivAC` or `EntAC` estimate the diversity or entropy accumulation curve of a distribution. See `Tsallis` for details about the computation of entropy at each level of interpolation and extrapolation. In accumulation curves, extrapolation if done by estimating the asymptotic distribution of the community.
and estimating entropy at different levels by interpolation. The asymptotic richness is adjusted so that the extrapolated part of the accumulation joins the observed value at the sample size.

AccumCurve objects include EntAC and DivAC objects for entropy and diversity accumulation. They generalize the classical Species Accumulation Curves (SAC) which are diversity accumulation of order $q=0$.

as.AccumCurve transforms two vectors (where x is the sample size and y the accumulation) into an object of class AccumCurve.

AccumCurve objects can be plotted with either plot or autoplot methods.

Value

A DivAC or an EntAC object. Both are AccumCurve objects, which are a list:

- **x**: The sample size.
- **y**: The value of entropy or diversity.
- **low**: The lower bound of the confidence envelope of the estimation.
- **high**: The upper bound of the confidence envelope of the estimation.

Attributes "Size" and "Value" contain the actual sample size and the corresponding diversity or entropy.

AccumCurve objects can be summarized and plotted.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>, Bruno Herault <Bruno.Herault@ecofog.gf>

References


See Also

Tsallis, Diversity

Examples

```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Ns is the total number of trees per species
Ns <- as.AbdVector(Paracou618.MC$Ns)
# Accumulation curve of Simpson's diversity
autoplot(DivAC(Ns, q=2))
```
AllenH

Phylogenetic Entropy of a Community

Description

Calculates the phylogenetic diversity of order q of a probability vector.

Usage

AllenH(Ps, q = 1, PhyloTree, Normalize = TRUE, Prune = FALSE, CheckArguments = TRUE)

Arguments

Ps A probability vector, summing to 1.
q A number: the order of entropy. Default is 1.
PhyloTree An object of class hclust, phylo, phylog or PPtree. The tree is not necessarily ultrametric.
Normalize If TRUE (default), diversity is not affected by the height of the tree. If FALSE, it is proportional to the height of the tree.
Prune What to do when some species are in the tree but not in Ps? If TRUE, the tree is pruned to keep species of Ps only. The height of the tree may be changed if a pruned branch is related to the root. If FALSE (default), species with probability 0 are added in Ps.
CheckArguments Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

Details

The phylogenetic entropy is calculated following Allen et al. (2009) for order $q = 1$ and Leinster and Cobold (2011) for other orders. The result is identical to the total entropy calculated by Phylentropy but it is much faster. A single value is returned instead of a Phylentropy object, and no bias correction is available.

The Normalize argument allows normalizing entropy by the height of the tree, similarly to ChaoPD. Diversity can be calculated for non ultrametric trees following Leinster and Cobold (2011) even though the meaning of the result is not so clear.

Value

A named number equal the entropy of the community. The name is "None" to recall that no bias correction is available.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>
References


See Also

PhyloEntropy, ChaoPD

Examples

# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest
# and their taxonomy)
data(Paracou618)
# Ps is the vector of probabilities
Ps <- as.ProbaVector(Paracou618.MC$Ns)

# Calculate the phylogenetic Shannon diversity of the plot
Alenh(Ps, 1, Paracou618.Taxonomy, Normalize=TRUE)

# Calculate it using PhyloEntropy: more powerful but much slower is the tree has many periods
PhyloEntropy(Ps, 1, Paracou618.Taxonomy, Normalize=TRUE) -> phyE
summary(phyE)

### AlphaDiversity

**Reduced-bias alpha diversity of a metacommunity**

Description

Calculates the reduced-bias total alpha diversity of order \( q \) of communities.

Usage

```r
AlphaDiversity(MC, q = 1, Correction = "Best", Tree = NULL, Normalize = TRUE, 
Z = NULL, CheckArguments = TRUE)
```

Arguments

- **MC**
  A *MetaCommunity* object.
- **q**
  A number: the order of diversity. Default is 1 for Shannon diversity.
- **Correction**
  A string containing one of the possible corrections accepted by *AlphaEntropy*
  or "None" or "Best", the default value.
- **Tree**
  An object of class *hclust, phylo, phylog* or *PPTree*. The tree must be ultra-
  metric.
- **Normalize**
  If TRUE (default), diversity is not affected by the height of the tree.
  If FALSE, diversity is proportional to the height of the tree.
A relatedness matrix, *i.e.* a square matrix whose terms are all positive, strictly positive on the diagonal. Generally, the matrix is a similarity matrix, *i.e.* the diagonal terms equal 1 and other terms are between 0 and 1.

**CheckArguments** Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

**Details**
Entropy is calculated by `AlphaEntropy` and transformed into diversity.

**Value**
An `MCdiversity` object containing diversity values of each community and of the metacommunity.

**Author(s)**
Eric Marcon <Eric.Marcon@ecofog.gf>

**References**

**See Also**
`AlphaEntropy`

**Examples**
```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Calculate Simpson alpha diversity
summary(AlphaDiversity(Paracou618.MC, 2))
# Compare without correction
summary(AlphaDiversity(Paracou618.MC, 2, Correction = "None"))
# Estimate phylogenetic Simpson alpha diversity
summary(AlphaDiversity(Paracou618.MC, 2, Tree = Paracou618.Taxonomy) -> e)
plot(e)
```
AlphaEntropy

Reduced-bias alpha entropy of a metacommunity

Description

Calculates the reduced-bias total alpha entropy of order $q$ of communities.

Usage

```r
AlphaEntropy(MC, q = 1, Correction = "Best", Tree = NULL, Normalize = TRUE,
            Z = NULL, CheckArguments = TRUE)
```

Arguments

- **MC**: A `MetaCommunity` object.
- **q**: A number: the order of diversity. Default is 1 for Shannon entropy.
- **Correction**: A string containing one of the possible corrections accepted by the bias-corrected entropy function (see details) or "None" or "Best", the default value.
- **Tree**: An object of class `hclust`, `phylo`, `phylog` or `PPTree`. The tree must be ultra-metric.
- **Normalize**: If `TRUE` (default), the entropy returned by the function is normalized by the height of the tree (it is the weighted average value of the entropy in each slice). If `FALSE`, it is the unnormalized weighted sum of the results.
- **Z**: A relatedness matrix, i.e. a square matrix whose terms are all positive, strictly positive on the diagonal. Generally, the matrix is a similarity matrix, i.e. the diagonal terms equal 1 and other terms are between 0 and 1.
- **CheckArguments**: Logical; if `TRUE`, the function arguments are verified. Should be set to `FALSE` to save time when the arguments have been checked elsewhere.

Details

If `Tree` is not `NULL`, then phylogenetic entropy is calculated by `bcPhyloEntropy`; else, if `Z` is not `NULL`, then similarity-based entropy is calculated by `bCHqz`; else, neutral entropy is calculated by `bcTsallis`.

The alpha entropy of each community is calculated and summed according to community weights. The possible corrections are detailed in `Tsallis`.

Value

An `MEntropy` object containing entropy values of each community and of the metacommunity.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>
References


See Also

`bcTsallis`

Examples

```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Calculate Simpson alpha entropy
summary(AlphaEntropy(Paracou618.MC, 2))
# Compare without correction
summary(AlphaEntropy(Paracou618.MC, 2, Correction = "None"))
# Estimate phylogenetic Simpson alpha entropy
summary(AlphaEntropy(Paracou618.MC, 2, Tree = Paracou618.Taxonomy) -> e)
plot(e)
```

**BetaDiversity**

*Reduced-bias beta diversity of a metacommunity*

Description

Calculates the reduced-bias beta diversity of order $q$ between communities.

Usage

```r
BetaDiversity(MC, q = 1, Correction = "Best", Tree = NULL, Normalize = TRUE, Z = NULL, CheckArguments = TRUE)
```

Arguments

- `MC`: A `MetaCommunity` object.
- `q`: A number: the order of diversity. Default is 1 for Shannon diversity.
- `Correction`: A string containing one of the possible corrections accepted by `bcTsallisBeta` or "None" or "Best", the default value.
- `Tree`: An object of class `hclust`, `phylo`, `phylog` or `PPTree`. The tree must be ultrametric.
- `Normalize`: If TRUE (default), diversity is not affected by the height of the tree. If FALSE, diversity is proportional to the height of the tree.
A relatedness matrix, \textit{i.e.} a square matrix whose terms are all positive, strictly positive on the diagonal. Generally, the matrix is a similarity matrix, \textit{i.e.} the diagonal terms equal 1 and other terms are between 0 and 1.

CheckArguments Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

Details

Entropy is calculated by \texttt{BetaEntropy} and transformed into diversity.

Diversity values of communities are not defined: community entropies are averaged to obtain the metacommunity entropy which is transformed into diversity (Marcon et al., 2014).

Value

An \texttt{MCdiversity} object containing diversity value of the metacommunity.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


See Also

\texttt{BetaEntropy}

Examples

\begin{verbatim}
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Estimate Shannon beta diversity
summary(BetaDiversity(Paracou618$MC, 1))
# Compare without correction
summary(BetaDiversity(Paracou618$MC, 1, Correction = "None"))
# Estimate phylogenetic Shannon beta diversity
summary(BetaDiversity(Paracou618$MC, 1, Tree = Paracou618$Taxonomy) -> e)
\end{verbatim}
**BetaEntropy**

*Reduced-bias beta entropy of a metacommunity*

**Description**
Calculates the reduced-bias beta entropy of order $q$ between communities.

**Usage**

```r
BetaEntropy(MC, q = 1, Correction = "Best", Tree = NULL, Normalize = TRUE, 
Z = NULL, CheckArguments = TRUE)
```

**Arguments**

- **MC**
  A `MetaCommunity` object.

- **q**
  A number: the order of diversity. Default is 1 for Shannon entropy.

- **Correction**
  A string containing one of the possible corrections accepted by the bias-corrected entropy function (see details) or "None" or "Best", the default value.

- **Tree**
  An object of class `hclust`, `phylo`, `phylog` or `PPTree`. The tree must be ultra-metric.

- **Normalize**
  If `TRUE` (default), the entropy returned by the function is normalized by the height of the tree (it is the weighted average value of the entropy in each slice). If `FALSE`, it is the unnormalized weighted sum of the results.

- **Z**
  A relatedness matrix, *i.e.* a square matrix whose terms are all positive, strictly positive on the diagonal. Generally, the matrix is a similarity matrix, *i.e.* the diagonal terms equal 1 and other terms are between 0 and 1.

- **CheckArguments**
  Logical; if `TRUE`, the function arguments are verified. Should be set to `FALSE` to save time when the arguments have been checked elsewhere.

**Details**

If `Tree` is not `NULL`, then phylogenetic entropy is calculated by `bcPhyloBetaEntropy`; else, if `Z` is not `NULL`, then similarity-based entropy is calculated by `bcHqzBeta`; else, neutral entropy is calculated by `bcTsallisBeta`.

The reduced-bias beta entropy of each community is calculated and summed according to community weights.

Note that beta entropy is related to alpha entropy (if $q$ is not 1) and cannot be compared across communities (Jost, 2007). Do rather calculate the `BetaDiversity` of the metacommunity.

**Value**

An `MEntropy` object containing entropy values of each community and of the metacommunity.

**Author(s)**

Eric Marcon <Eric.Marcon@ecofog.gf>
References


See Also

`bctsandisiBeta, BetaDiversity`

Examples

```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Estimate Shannon beta entropy
summary(BetaEntropy(Paracou618.MC, 1))
# Compare without correction
summary(BetaEntropy(Paracou618.MC, 1, Correction = "None"))
# Estimate phylogenetic Shannon beta entropy
summary(BetaEntropy(Paracou618.MC, 1, Tree = Paracou618.Taxonomy) -> e)
plot(e)
```

ChaoPD

*Phylogenetic Diversity of a Community*

Description

Calculates the phylogenetic diversity of order \( q \) of a probability vector.

Usage

`ChaoPD(Ps, q = 1, PhyloTree, Normalize = TRUE, Prune = FALSE, CheckArguments = TRUE)`

Arguments

- **Ps**: A probability vector, summing to 1.
- **q**: A number: the order of diversity. Default is 1.
- **PhyloTree**: An object of class *hclust, phylo, phylog* or *PPTree*. The tree is not necessarily ultrametric.
- **Normalize**: If TRUE (default), diversity is not affected by the height of the tree. If FALSE, it is proportional to the height of the tree.
- **Prune**: What to do when some species are in the tree but not in \( Ps \)? If TRUE, the tree is pruned to keep species of \( Ps \) only. The height of the tree may be changed if a pruned branch is related to the root. If FALSE (default), species with probability 0 are added in \( Ps \).
CheckArguments  Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

Details

The phylogenetic diversity is calculated following Chao et al. (2010). The result is identical to the total diversity calculated by Phylodiversity but it is much faster. A single value is returned instead of a Phylodiversity object, and no bias correction is available.

The Normalize arguments allows calculating either $q\bar{D}(T)$ (if TRUE) or $qPD(T)$ if FALSE.

Diversity can be calculated for non ultrametric trees following Chao et al. (2010) even though the meaning of the result is not so clear (Leinster and Cobold, 2011).

Value

A named number equal the diversity of the community. The name is "None" to recall that no bias correction is available.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


See Also

PhyloDiversity, AllenH

Examples

# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest and their taxonomy)
data(Paracou618)
# Ps is the vector of probabilities
Ps <- Paracou618.MC$Ps

# Calculate the phylogenetic Simpson diversity of the plot
(ChaoPD(Paracou618.MC$Ps, 2, Paracou618.Taxonomy, Normalize=TRUE))

# Calculate it using PhyloDiversity
# (more powerful but much slower if the tree has many periods)
PhyloDiversity(Paracou618.MC$Ps, 2, Paracou618.Taxonomy, Normalize=TRUE) -> phyD
summary(phyD)
CommunityProfile

Diversity or Entropy Profile of a community

Description

Calculates the diversity or entropy profile of a community, applying a community function to a vector of orders.

Usage

CommunityProfile(FUN, NorP, q.seq = seq(0, 2, 0.1),
  NumberOfSimulations = 0, Alpha = 0.05, BootstrapMethod = "Chao2015",
  size = 1, ..., ShowProgressBar = TRUE, CheckArguments = TRUE)
as_CommunityProfile(x, y, low = NULL, high = NULL)
is_CommunityProfile(x)
## S3 method for class 'CommunityProfile'
plot(x, ..., main = NULL,
  xlab = "Order of Diversity", ylab = "Diversity", ylim = NULL,
  LineWidth = 2, ShadeColor = "grey75", BorderColor = "red")
## S3 method for class 'CommunityProfile'
autoplot(object, ..., main = NULL,
  xlab = "Order of Diversity", ylab = "Diversity",
  ShadeColor = "grey75", alpha = 0.3, BorderColor = "red")
CEEnvelope(Profile, LineWidth = 2, ShadeColor = "grey75", BorderColor = "red", ...)

Arguments

- **FUN**
  - The function to be applied to each value of q.seq. Any function accepting a numeric vector (or a two-column matrix) and a number as first two arguments and an argument named CheckArguments is acceptable (other arguments of the functions are passed by...). See *Details* for useful entropy and diversity functions and *Examples* for an ad-hoc one.

- **NorP**
  - A numeric vector. Contains either abundances or probabilities.

- **q.seq**
  - A numeric vector: the sequence of diversity orders to address. Default is from 0 to 2.

- **NumberOfSimulations**
  - The number of simulations to run, 0 by default.

- **Alpha**
  - The risk level, 5% by default.

- **BootstrapMethod**
  - The method used to obtain the probabilities to generate bootstrapped communities from observed abundances. See rCommunity.

- **size**
  - The size of simulated communities used to compute the bootstrap confidence envelope. 1 (default) means that the actual size must be used.

- **object**
  - An object.
CommunityProfile

x  An object to be tested or plotted or the vector of orders of community profiles in as.CommunityProfile.
y  Entropy or diversity values of each order, corresponding to x values.
low  Entropy or diversity lower bound of the confidence envelope, corresponding to x values.
high  Entropy or diversity higher bound of the confidence envelope, corresponding to x values.
Profile  An CommunityProfile to be plotted.
...  Additional arguments to be passed to FUN in CommunityProfile, to plot in plot.CommunityProfile or to lines in CEvelope.
main  The main title of the plot.
xlab  The x axis label of the plots.
ylab  The y axis label of the plot.
ylim  The interval of y values plotted.
LineWidth  The width of the line that represents the actual profile.
ShadeColor  The color of the shaded confidence envelope.
BorderColor  The color of the bounds of the confidence envelope.
alpha  Opacity of the confidence envelope, between 0 and 1.
ShowProgressBar  If TRUE (default), a progress bar is shown.
CheckArguments  Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

Details

The function CommunityProfile is used to calculate diversity or entropy profiles based on community functions such as Tsallis or ChaopD. The first two arguments of the function must be a probability or abundance vector and a number (q). Additional arguments cannot be checked. Unexpected results may be returned if FUN is not used properly.

If NumberOfSimulations is greater than 0, a bootstrap confidence interval is produced by simulating communities with rCommunity and calculating their profiles. The size of those communities may be that of the actual community or specified by size. CommunityProfile objects can be plotted. They can also be added to the current plot by CEvelope.

Value

A CommunityProfile, which is a list:

x  The order q values
y  The entropy or diversity values returned by FUN
low  The lower bound of the confidence interval
high  The upper bound of the confidence interval
Coverage

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>, Bruno Herault <Bruno.Herault@ecofog.gf>

Examples

```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Plot diversity estimated without bias correction
plot(CommunityProfile(Diversity, Paracou618$MC$Ps, seq(0, 2, 0.2)),
     lty=3, ylim=c(50, 350))
# Estimate diversity, with a confidence envelope
# (only 10 simulations to save time, should be 1000)
Profile <- CommunityProfile(Diversity, as.AbdVector(Paracou618$MC$Ns),
                             seq(0, 2, 0.2), Correction="Unveil", NumberOfSimulations=10)
# Complete the plot, and add the legend
CEnvelope(Profile, main="Paracou Plots Diversity")
legend("topright", c("Bias Corrected", "Biased"), lty=c(1,3), inset=0.01)
```

```r
# Advanced use with beta-diversity functions :
# Profile of the beta entropy of the first community of Paracou618.
# Observed and expected probabilities are bound into a 2-column matrix
# An intermediate function is necessary to separate them before calling TsallisBeta
# The CheckArguments is mandatory but does not need to be set: CommunityProfile() sets it to FALSE
CommunityProfile(Function(PandPexp, q, CheckArguments)
                  (TsallisBeta(PandPexp[, 1], PandPexp[, 2], q)),
                  NorP=cbind(Paracou618$MC$Psi[, 1], Paracou618$MC$Ps), q.seq=seq(0, 2, 0.2))
```

<table>
<thead>
<tr>
<th>Coverage</th>
<th>Sample coverage of a community</th>
</tr>
</thead>
</table>

Description

"Coverage" calculates an estimator of the sample coverage of a community described by its abundance vector. "Coverage2Size" estimates the sample size corresponding to the chosen sample coverage.

Usage

```r
Coverage(Ns, Estimator = "Best", Level = NULL, CheckArguments = TRUE)
Coverage2Size(Ns, SampleCoverage, CheckArguments = TRUE)
```

Arguments

- **Ns**
  - A numeric vector containing species abundances.
- **Estimator**
  - A string containing one of the possible estimators: "ZhangHuang", "Chao", "Turing", "Good". "Best" is for "Chao".
- **Level**
  - The level of interpolation or extrapolation, i.e. an abundance.
- **SampleCoverage**
  - The target sample coverage.
CheckArguments Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

Details

The sample coverage $C$ of a community is the total probability of occurrence of the species observed in the sample. $1 - C$ is the probability for an individual of the whole community to belong to a species that has not been sampled.

The historical estimator is due to Turing (Good, 1953). It only relies on singletons (species observed only once). Chao’s (Chao and Shen, 2010) estimator uses doubletons too and Zhang-Huang’s (Chao et al., 1988; Zhang and Huang, 2007) uses the whole distribution.

If level is not null, the sample coverage is interpolated or extrapolated. Interpolation by the Good estimator relies on the equality between sampling deficit and the generalized Simpson entropy (Good, 1953). The Chao (2014) estimator allows extrapolation, reliable up a level equal to the double size of the sample.

Value

"Coverage" returns a named number equal to the calculated sample coverage. The name is that of the estimator used. "Coverage2Size" returns a number equal to the sample size corresponding to the chosen sample coverage.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


Examples

```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Ns is the vector of abundances of the metacommunity
Ns <- Paracou618.MC$Ns
# Calculate the sample coverage of the metacommunity
Coverage(Ns)  # Stored in Paracou618.SampleCoverage
```
Diversity

Hill number of a community

Description

Calculates the HCDT (generalized) diversity of order $q$ of a probability vector.

Usage

Diversity(NorP, q = 1, ...)
bcDiversity(Ns, q = 1, Correction = "Best", CheckArguments = TRUE)
## S3 method for class 'ProbaVector'
Diversity(NorP, q = 1, ..., CheckArguments = TRUE, Ps = NULL)
## S3 method for class 'AbdVector'
Diversity(NorP, q = 1, Correction = "Best", Level = NULL,
PCorrection="Chao2015", Unveiling="geom", RCorrection="Rarefy", ..., 
CheckArguments = TRUE, Ns = NULL)
## S3 method for class 'integer'
Diversity(NorP, q = 1, Correction = "Best", Level = NULL,
PCorrection="Chao2015", Unveiling="geom", RCorrection="Rarefy", ..., 
CheckArguments = TRUE, Ns = NULL)
## S3 method for class 'numeric'
Diversity(NorP, q = 1, Correction = "Best", Level = NULL,
PCorrection="Chao2015", Unveiling="geom", RCorrection="Rarefy", ..., 
CheckArguments = TRUE, Ps = NULL, Ns = NULL)

Arguments

- **Ps**
  A probability vector, summing to 1.

- **Ns**
  A numeric vector containing species abundances.

- **NorP**
  A numeric vector, an integer vector, an abundance vector (`AbdVector`) or a probability vector (`ProbaVector`). Contains either abundances or probabilities.

- **q**
  A number: the order of diversity. Default is 1.

- **Correction**
  A string containing one of the possible asymptotic estimators: "None" (no correction), "ChaoShen", "GenCov", "Grassberger", "Holste", "Bonachela", "ZhangGrabchak", or "ChaoJost", "Marcon", "UnveiliC", "UnveiliJ", "UnveilJ" or "Best", the default value. Currently, "Best" is "UnveilJ".

- **Level**
  The level of interpolation or extrapolation. It may be an a chosen sample size (an integer) or a sample coverage (a number between 0 and 1).

- **PCorrection**
  A string containing one of the possible corrections to estimate a probability distribution in `as.ProbaVector`: "Chao2015" is the default value. Used only for extrapolation.
Unveiling  A string containing one of the possible unveiling methods to estimate the probabilities of the unobserved species in `as.ProbaVector`: "geom" (the unobserved species distribution is geometric) is the default value. Used only for extrapolation.

RCorrection  A string containing a correction recognized by `Richness` to evaluate the total number of species in `as.ProbaVector`. "Rarefy" is the default value to estimate the number of species such that the diversity of the asymptotic distribution rarefied to the observed sample size equals the observed diversity of the data. Used only for extrapolation.

...  Additional arguments. Unused.

CheckArguments  Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

Details

Diversity calls `Tsallis` to calculate entropy and transforms it into diversity by calculating its deformed exponential.

Bias correction requires the number of individuals to estimate sample Coverage. See `Tsallis` for details.

The functions are designed to be used as simply as possible. Diversity is a generic method. If its first argument is an abundance vector, an integer vector or a numeric vector which does not sum to 1, the bias corrected function `bcDiversity` is called.

Diversity can be estimated at a specified level of interpolation or extrapolation, either a chosen sample size or sample coverage (Chao et al., 2014), rather than its asymptotic value. See `Tsallis` for details.

Value

A named number equal to the calculated diversity. The name is that of the bias correction used.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


See Also

`Tsallis`, `expq`, `AbdVector`, `ProbaVector`
Examples

# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Ns is the total number of trees per species
Ns <- as.AbdVector(Paracou618.MC$Ns)
# Species probabilities
Ps <- as.ProbaVector(Paracou618.MC$Ns)
# Whittaker plot
plot(Ns)
# Calculate diversity of order 1, i.e. Shannon's diversity
Diversity(Ps, q=1)
# Calculate it with estimation bias correction (asymptotic estimator)
Diversity(Ns, q=1)
# Extrapolate it up to 99.9% sample coverage (close to the asymptotic estimator)
Diversity(Ns, q=1, Level=0.999)
# Rarefy it to half the sample size
Diversity(Ns, q=1, Level=sum(Ns)/2)

---

DivEst  
Diversity Estimation of a metacommunity

Description

Estimates diversity of a metacommunity.

Usage

DivEst(q = 0, MC, Biased = TRUE, Correction = "Best", Tree = NULL, 
Normalize = TRUE, Z = NULL, Simulations = 100, 
ShowProgressBar = TRUE, CheckArguments = TRUE)

Arguments

q  
A number: the order of diversity.

MC  
A MetaCommunity object.

Biased  
Logical; if FALSE, a bias correction is applied.

Correction  
A string containing one of the possible corrections. The correction must be accepted by DivPart. "Best" is the default value.
Tree  An object of class `hclust`, `phylo`, `phylog` or `PPTree`. The tree must be ultrametric.

Normalize If TRUE (default), diversity is not affected by the height of the tree.
If FALSE, diversity is proportional to the height of the tree.

Z  A relatedness matrix, i.e. a square matrix whose terms are all positive, strictly positive on the diagonal. Generally, the matrix is a similarity matrix, i.e. the diagonal terms equal 1 and other terms are between 0 and 1.

Simulations The number of simulations to build confidence intervals.

ShowProgressBar If TRUE (default), a progress bar is shown.

CheckArguments Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

x  An object to be tested or plotted.

main  The title of the plot.

Which May be "Alpha", "Beta" or "Gamma" to respectively plot the metacommunity’s alpha, beta or gamma diversity. If "All" (default), all three plots are shown.

labels  Vector of labels to be added to multiple plots. "auto" is the same as c("a", "b", "c", "d").

font.label  A list of arguments to customize labels. See `ggarrange`.

object  A `MCdiversity` object to be summarized or plotted.

...  Additional arguments to be passed to the generic methods.

Details

`DivEst` estimates the diversity of the metacommunity and partitions it into alpha and beta components.

If `Tree` is provided, the phylogenetic diversity is calculated else if `Z` is not NULL, then similarity-based entropy is calculated.

Bootstrap confidence intervals are calculated by drawing simulated communities from a multinomial distribution following the observed frequencies (Marcon et al, 2012; 2014).

Value

A `DivEst` object which is a `DivPart` object with an additional item in its list:

SimulatedDiversity

A matrix containing the simulated values of alpha, beta and gamma diversity.

`DivEst` objects can be summarized and plotted.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf> and Bruno Herault <Bruno.Herault@ecofog.gf>
References


See Also

DivPart

Examples

```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Estimate Shannon diversity.
Estimation <- DivEst(q = 1, Paracou618.MC, Biased = FALSE, Correction = "Unveil", Simulations = 20)
plot(Estimation)
summary(Estimation)
```

DivPart **Diversity Partition of a metacommunity**

Description

Partitions the diversity of a metacommunity into alpha and beta components.

Usage

```r
DivPart(q = 1, MC, Biased = TRUE, Correction = "Best", Tree = NULL,
       Normalize = TRUE, Z = NULL, CheckArguments = TRUE)
```

Arguments

- **q** A number: the order of diversity. Default is 1.
- **MC** A `MetaCommunity` object.
- **Biased** Logical; if FALSE, a bias correction is applied.
Correction A string containing one of the possible corrections.
The correction must be accepted by AlphaEntropy, BetaEntropy and GammaEntropy. "Best" is the default value.

Tree An object of class hclust, phylo, phylog or PPtrree. The tree must be ultra-metric.

Normalize If TRUE (default), diversity is not affected by the height of the tree.
If FALSE, diversity is proportional to the height of the tree.

Z A relatedness matrix, i.e. a square matrix whose terms are all positive, strictly positive on the diagonal. Generally, the matrix is a similarity matrix, i.e. the diagonal terms equal 1 and other terms are between 0 and 1.

CheckArguments Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

x An object to be tested or plotted.

object A MCdiversity object to be summarized or plotted.

... Additional arguments to be passed to the generic methods.

Details

DivPart partitions the diversity of the metacommunity into alpha and beta components. It supports estimation-bias correction.

If Tree is provided, the phylogenetic diversity is calculated else if Z is not NULL, then similarity-based entropy is calculated.
Beta diversity/entropy is calculated from Gamma and Alpha when bias correction is required, so community values are not available.

Value

A DivPart object. It is a list:

MetaCommunity The name of the MetaCommunity object containing inventory data.
Order The value of q.
Biased Logical. If FALSE, bias corrected values of diversity have been computed.
Correction The estimation bias correction used to calculate diversity.
Method The method used to calculate entropy ("HCDT", "Similarity-based").
Tree The phylogenetic or functional tree used to calculate phylodiversity.
Normalized Logical. Indicates whether phylodiversity is normalized or proportional to the height of the tree.
Z The matrix used to calculate similarity-based entropy.
TotalAlphaDiversity The alpha diversity of communities.
TotalBetaDiversity The beta diversity of communities.
GammaDiversity The gamma diversity of the metacommunity.
CommunityAlphaDiversities
   A vector containing the alpha diversity of each community.

TotalAlphaEntropy
   The alpha entropy of communities.

TotalBetaEntropy
   The beta entropy of communities.

GammaEntropy
   The gamma entropy of the metacommunity.

CommunityAlphaEntropies
   A vector containing the alpha entropy of each community.

DivPart objects can be summarized and plotted.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf> and Bruno Herault <Bruno.Herault@ecofog.gf>

References


See Also

   DivProfile

Examples

# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Estimate Shannon diversity.
summary(DivPart(q = 1, Paracou618.MC, Biased = FALSE) -> dp)
plot(dp)
DivProfile

Diversity Profile of a metacommunity

Description

Calculate the diversity profiles (alpha, beta, gamma) of a metacommunity.

Usage

DivProfile(q.seq = seq(0, 2, 0.1), MC, Biased = TRUE, Correction = "Best",
        Tree = NULL, Normalize = TRUE, Z = NULL,
        NumberOfSimulations = 0, Alpha = 0.05,
        ShowProgressBar = TRUE, CheckArguments = TRUE)

is.DivProfile(x)

## S3 method for class 'DivProfile'
plot(x, ..., main = NULL, xlab = "Order of Diversity",
     ylab = NULL, Which = "All",
     LineWidth = 2, ShadeColor = "grey75", BorderColor = "red")

## S3 method for class 'DivProfile'
autoplot(object, ..., main = NULL, xlab = "Order of Diversity",
         ylab = NULL, Which = "All", ShadeColor = "grey75", alpha = 0.3, BorderColor = "red",
         labels = NULL, font.labels = list(size=11, face="plain"))

## S3 method for class 'DivProfile'
summary(object)

Arguments

q.seq A numeric vector.
MC A MetaCommunity object.
Biased Logical; if FALSE, a bias correction is applied.
Correction A string containing one of the possible corrections.
The correction must be accepted by AlphaEntropy, BetaEntropy and GammaEntropy.
"Best" is the default value.
Tree An object of class hclust, phylo, phylog or PPTree. The tree must be ultra-
metric.
Normalize If TRUE (default), diversity is not affected by the height of the tree.
If FALSE, diversity is proportional to the height of the tree.
Z A relatedness matrix, i.e. a square matrix whose terms are all positive, strictly
positive on the diagonal. Generally, the matrix is a similarity matrix, i.e. the
diagonal terms equal 1 and other terms are between 0 and 1.
NumberOfSimulations The number of simulations to run, 0 by default.
Alpha The risk level, 5% by default.
ShowProgressBar If TRUE (default), a progress bar is shown.
CheckArguments Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

x An object to be tested or plotted.

main The main title of the plot. Ignored if Which = "All".

xlab The x axis label of the plots.

ylab The y axis label of the plot. Ignored if Which = "All".

Which May be "Communities", "Alpha", "Beta" or "Gamma" to respectively plot the alpha diversity of communities or the metacommunity's alpha, beta or gamma diversity. If "All" (default), all four plots are shown.

LineWidth The width of the line that represents the actual profile.

ShadeColor The color of the shaded confidence envelope.

BorderColor The color of the bounds of the confidence envelope.

alpha Opacity of the confidence envelope, between 0 and 1.

labels Vector of labels to be added to multiple plots. "auto" is the same as c("a", "b", "c", "d").

font.label A list of arguments to customize labels. See ggarrange.

object A MCDiversity object to be summarized or plotted.

Details

If Tree is provided, the phylogenetic diversity is calculated.

DivPart partitions the diversity of the metacommunity into alpha and beta components. It supports estimation-bias correction.

If Tree is provided, the phylogenetic diversity is calculated else if Z is not NULL, then similarity-based entropy is calculated.

Beta diversity/entropy is calculated from Gamma and Alpha when bias correction is required, so community values are not available.

If NumberOfSimulations is greater than 0, a bootstrap confidence interval is produced by simulating communities from a multinomial distribution following the observed frequencies (Marcon et al, 2012; 2014) and calculating their profiles.

Value

A DivProfile object. It is a list:

MetaCommunity The name of the MetaCommunity object containing inventory data.

Order A vector containing the values of q.

Biased Logical. If FALSE, bias corrected values of diversity have been computed.

Correction The estimation bias correction used to calculate diversity. Usually a string, but it may be a list if different corrections have been used in the estimation of phylodiversity.

Method The method used to calculate entropy ("HCDT", "Similarity-based").
The phylogenetic or functional tree used to calculate phylodiversity.

Normalized Logical. Indicates whether phylodiversity is normalized or proportional to the height of the tree.

Z The matrix used to calculate similarity-based entropy.

CommunityAlphaDiversities A matrix containing the alpha diversity of each community.

TotalAlphaDiversity A vector containing the alpha diversity of communities for each order.

BetaDiversity A vector containing the beta diversity of communities for each order.

GammaDiversity A vector containing the gamma diversity of the metacommunity for each order.

CommunityAlphaEntropies A matrix containing the alpha entropy of each community.

TotalAlphaEntropy A vector containing the alpha entropy of communities for each order.

BetaEntropy A vector containing the beta entropy of communities for each order.

GammaEntropy A vector containing the gamma entropy of the metacommunity for each order.

Confidence envelopes Total Alpha, Beta and Gamma Entropy and Diversity may come with a confidence envelope whose value is stored in twelve more vectors named suffixed Low or High, such as GammaEntropyLow

DivProfile objects can be summarized and plotted.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf> and Bruno Herault <Bruno.Herault@ecofog.gf>

References


See Also

DivPart

Examples

# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)

# Estimate diversity.
Profile <- DivProfile(q.seq = seq(0, 2, 0.1), Paracou618.MC, Biased = FALSE)
plot(Profile)
Similarity-based diversity of a community

Description
Calculates the diversity of order $q$ of a probability vector according to a similarity matrix.

Usage
Dqz(NorP, q = 1, Z = diag(length(NorP)), ...)
bcdqz(Ns, q = 1, Z = diag(length(Ns)), Correction = "Best", CheckArguments = TRUE)
## S3 method for class 'ProbaVector'
Dqz(NorP, q = 1, Z = diag(length(NorP)), ..., 
    CheckArguments = TRUE, Ps = NULL)
## S3 method for class 'AbdVector'
Dqz(NorP, q = 1, Z = diag(length(NorP)), Correction = "Best", ..., 
    CheckArguments = TRUE, Ns = NULL)
## S3 method for class 'integer'
Dqz(NorP, q = 1, Z = diag(length(NorP)), Correction = "Best", ..., 
    CheckArguments = TRUE, Ns = NULL)
## S3 method for class 'numeric'
Dqz(NorP, q = 1, Z = diag(length(NorP)), Correction = "Best", ..., 
    CheckArguments = TRUE, Ps = NULL, Ns = NULL)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ps</td>
<td>A probability vector, summing to 1.</td>
</tr>
<tr>
<td>Ns</td>
<td>A numeric vector containing species abundances.</td>
</tr>
<tr>
<td>NorP</td>
<td>A numeric vector, an integer vector, an abundance vector (AbdVector) or a probability vector (ProbaVector). Contains either abundances or probabilities.</td>
</tr>
<tr>
<td>q</td>
<td>A number: the order of diversity. Default is 1.</td>
</tr>
<tr>
<td>Z</td>
<td>A relatedness matrix, i.e. a square matrix whose terms are all positive, strictly positive on the diagonal. Generally, the matrix is a similarity matrix, i.e. the diagonal terms equal 1 and other terms are between 0 and 1. Default is the identity matrix to calculate neutral diversity.</td>
</tr>
<tr>
<td>Correction</td>
<td>A string containing one of the possible corrections: &quot;None&quot; (no correction), &quot;HorvitzThomson&quot;, &quot;MarconZhang&quot; or &quot;Best&quot;, the default value. The &quot;MarconZhang&quot; correction assumes a similarity matrix.</td>
</tr>
<tr>
<td>...</td>
<td>Additional arguments. Unused.</td>
</tr>
<tr>
<td>CheckArguments</td>
<td>Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.</td>
</tr>
</tbody>
</table>
Details

Diversity is calculated following Leinster and Cobbold (2012): it is the reciprocal of the (generalized) average (of order q) of the community species ordinariness.

A similarity matrix is used (as for Dqz), not a distance matrix as in Ricotta and Szeidl (2006). See the example.

Bias correction requires the number of individuals. Use bcHqz and choose the Correction. Correction techniques are from Marcon et al. (2014).

Currently, the "Best" correction is the max value of "HorvitzThomson" and "MarconZhang".

The functions are designed to be used as simply as possible. Dqz is a generic method. If its first argument is an abundance vector, an integer vector or a numeric vector which does not sum to 1, the bias corrected function bcDqz is called. Explicit calls to bcDqz (with bias correction) or to Dqz.ProbaVector (without correction) are possible to avoid ambiguity. The .integer and .numeric methods accept Ps or Ns arguments instead of NorP for backward compatibility.

Value

A named number equal to the calculated diversity. The name is that of the bias correction used.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


See Also

hqz, PhyloDiversity

Examples

```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Prepare the similarity matrix
DistanceMatrix <- as.matrix(Paracou618.dist)
# Similarity can be 1 minus normalized distances between species
Z <- 1 - DistanceMatrix/max(DistanceMatrix)
# Calculate diversity of order 2
Dqz(Paracou618.MC$Ns, 2, Z)
```
EightSpAbundance     Abundances of 8 species to run examples.

Description

This dataset is a light-weight example.

Usage

data(Paracou618)

Format

A named vector.

Examples

data(Paracou618)
EightSpAbundance

EightSpTree     Functional tree with 8 species.

Description

This dataset is a light-weight example.

Usage

data(Paracou618)

Format

An object of class phylog containing a functional tree.

Examples

data(Paracou618)
# Preprocess the tree to be able to plot it
# without loading ade4 package
plot(Preprocess.Tree(EightSpTree), hange=-0.01)
Grassberger's expectation of $n^q$

Description

Expected value of $n^q$ when $n$ follows a Poisson law.

Usage

Enq(n, q)

Arguments

n  A positive integer vector.
q  A positive number.

Details

The expectation of $n^q$ when $n$ follows a Poisson distribution has been derived by Grassberger (1988).

Value

A vector of the same length as $n$ containing the transformed values.

Note

The function is computed using the beta.function.

Its value is 0 for $n - q + 1 < 0$.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


Examples

# Compare
n <- c(2,3)
Enq(n, q=2)
# with
n^n

# Result is 1
Enq(n, q=0)
EntropyCI

Description

Resamples a community by Monte-Carlo simulations of a multinomial distribution and returns a vector of entropy values to calculate confidence intervals.

Usage

```r
EntropyCI(FUN, Simulations = 100, Ns, BootstrapMethod = "Chao2015", 
ShowProgressBar = TRUE, ..., CheckArguments = TRUE)
```

Arguments

- **FUN**: The entropy function to be applied to each simulated community. May be any entropy function accepting a vector of species abundances, such as `bcTsallis`, `bcShannon`, `bcSimpson` or `bcPhyloEntropy`.
- **Simulations**: The number of simulations to build confidence intervals.
- **Ns**: A numeric vector containing species abundances.
- **BootstrapMethod**: The method used to obtain the probabilities to generate bootstrapped communities from observed abundances. See `rCommunity`.
- **...**: Additional arguments to be passed to `FUN`.
- **ShowProgressBar**: If TRUE (default), a progress bar is shown.
- **CheckArguments**: Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

Details

This function is used to obtain the distribution of entropy and eventually calculate confidence intervals. It draws simulated communities according to a multinomial distribution with the same number of individuals and probabilities as the actual community. It calculates the entropy of each simulated community. Last, it recenters the distribution of entropy values around the actual value of entropy according to Marcon et al. (2012): the estimation bias of simulated communities entropy can not be corrected analytically, but it does not affect the distribution shape.

Diversity can not be recentered this way so diversity function should not be used. Unexpected results will be obtained if inappropriate functions are used.

Value

A numeric vector containing the entropy value of each simulated community.
Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


Examples

```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Abundance (all estimators will include bias correction)
Ns <- as.AbdVector(Paracou618.MC$Ns)
q <- 1
# Estimate entropy and transform it into diversity
RealEst <- expq(Tsallis(Ns, q), q)
# Transform the distribution of Tsallis entropy into diversity
SimulatedDiversity <- expq(EntropyCI(Tsallis, Simulations=50, Ns, q=q), q)
# Figure
plot(density(SimulatedDiversity), col="black", lwd=2, main="", xlab="Diversity")
abline(v=RealEst, col="red", lwd=2, lty=2)
abline(v=quantile(SimulatedDiversity, probs = 0.025), col="black", lwd=1, lty=3)
abline(v=quantile(SimulatedDiversity, probs = 0.975), col="black", lwd=1, lty=3)
legend("topright", c("Real value", "Confidence interval"), lty=c(2,3),
col=c("red", "black"), inset=0.01)
# Print results
cat("Estimated Diversity: ", RealEst)
quantile(SimulatedDiversity, probs = c(0.025, 0.975))
```

expq

*Exponential of order q*

Description

Calculates the deformed exponential of order $q$.

Usage

```r
expq(x, q)
expq.CommunityProfile(Profile)
```

Arguments

- `x` A numeric vector.
- `Profile` A `CommunityProfile`.
- `q` A number.
Details

The deformed exponential is defined as \((x(1 - q) + 1)^{1/(1-q)}\).

For \(q > 1\), \(\ln_q (+\infty) = \frac{1}{q-1}\) so \(\exp_q(x)\) is not defined for \(x > \frac{1}{q-1}\).

\(\exp q\).CommunityProfile calculates the deformed exponential of a CommunityProfile. Its \$x\) item (the order of diversity) is kept unchanged whilst other items are set to their exponential of order \$x\).
Thus, an entropy profile is transformed into a diversity profile.

Value

A vector of the same length as \(x\) containing the transformed values or a CommunityProfile.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


See Also

\(\exp q\)

Examples

\begin{verbatim}
curve(exp(x), -5, 0, lty=3)
curve(expq(x, 2), -5, 0, lty=2, add=TRUE)
curve(expq(x, 3), -5, 0, lty=1, add=TRUE)
legend("topleft", legend = c("exp(x)", "exp2(x)", "exp3(x)")
\end{verbatim}

Description

Calculates the reduced-bias diversity of order \(q\) of a metacommunity.

Usage

\begin{verbatim}
GammaDiversity(MC, q = 1, Correction = "Best", Tree = NULL, Normalize = TRUE,
              Z = NULL, CheckArguments = TRUE)
\end{verbatim}
Arguments

- `MC` A `MetaCommunity` object.
- `q` A number: the order of diversity. Default is 1.
- `Correction` A string containing one of the possible corrections accepted by `AlphaEntropy` or "None" or "Best", the default value.
- `Tree` An object of class `hclust`, `phylo`, `phylog` or `PPTree`. The tree must be ultra-metric.
- `Normalize` If TRUE (default), diversity is not affected by the height of the tree. If FALSE, diversity is proportional to the height of the tree.
- `Z` A relatedness matrix, i.e. a square matrix whose terms are all positive, strictly positive on the diagonal. Generally, the matrix is a similarity matrix, i.e. the diagonal terms equal 1 and other terms are between 0 and 1.
- `CheckArguments` Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

Details

Entropy is calculated by `GammaEntropy` and transformed into diversity.

Value

The metacommunity's gamma entropy.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


See Also

`GammaEntropy`

Examples

```
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Calculate Simpson gamma diversity
GammaDiversity(Paracou618$MC, 2)
# Compare without correction
```
GammaEntropy

Description

Calculates the reduced-bias Tsallis entropy of order $q$ of a metacommunity.

Usage

```
GammaEntropy(MC, q = 1, Correction = "Best", Tree = NULL, Normalize = TRUE,
             Z = NULL, PhyloDetails = FALSE, CheckArguments = TRUE)
```

Arguments

- **MC**: A `MetaCommunity` object.
- **q**: A number: the order of entropy. Default is 1.
- **Correction**: A string containing one of the possible corrections accepted by the bias-corrected entropy function (see details) or "None" or "Best", the default value.
- **Tree**: An object of class `hclust`, `phylo`, `phylog` or `PPTree`. The tree must be ultrametric.
- **Normalize**: If `TRUE` (default), the entropy returned by the function is normalized by the height of the tree (it is the weighted average value of the entropy in each slice). If `FALSE`, it is the unnormalized weighted sum of the results.
- **Z**: A relatedness matrix, i.e. a square matrix whose terms are all positive, strictly positive on the diagonal. Generally, the matrix is a similarity matrix, i.e. the diagonal terms equal 1 and other terms are between 0 and 1.
- **PhyloDetails**: If `FALSE` (default), the function always returns a number. If `TRUE` and `Tree` is not `NULL` then a `PhyloValue` object is returned with all details. That is used internally by `DivPart` to obtain the corrections used to estimate gamma entropy along the tree and apply them to the estimation of alpha diversity.
- **CheckArguments**: Logical; if `TRUE`, the function arguments are verified. Should be set to `FALSE` to save time when the arguments have been checked elsewhere.

Details

If `Tree` is not `NULL`, then phylogenetic entropy is calculated by `bcPhyloEntropy`.
Else, if `Z` is not `NULL`, then similarity-based entropy is calculated by `bcHqz`.
Else, neutral entropy is calculated by `bcTsallis`.

Value

A number equal to the calculated entropy.
Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


See Also

*bctSallis, bcPhyloEntropy*

Examples

```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Calculate Simpson gamma entropy
GammaEntropy(Paracou618.MC, 2)
# Compare without correction
GammaEntropy(Paracou618.MC, 2, Correction = "None")
# Estimate phylogenetic Simpson gamma entropy
GammaEntropy(Paracou618.MC, 2, Tree = Paracou618.Taxonomy)
```

Description

Calculates the Generalized Simpson’s entropy of order $r$ of a probability or abundance vector, and its effective number of species.

Usage

```r
GenSimpson(NorP, r = 1, ...)
bcGenSimpson(Ns, r = 1, CheckArguments = TRUE)
## S3 method for class 'ProbaVector'
GenSimpson(NorP, r = 1, ...
  CheckArguments = TRUE, Ps = NULL)
## S3 method for class 'AbdVector'
GenSimpson(NorP, r = 1, ...
  CheckArguments = TRUE, Ns = NULL)
## S3 method for class 'integer'
GenSimpson(NorP, r = 1, ...,
```
Arguments

Ps  A probability vector, summing to 1.
Ns  A numeric vector containing species abundances.
NorP A numeric vector, an integer vector, an abundance vector (AbdVector) or a probability vector (Probavector). Contains either abundances or probabilities.
r  A number: the order of diversity. Default is 1 for Simpson's diversity.
... Additional arguments. Unused.
CheckArguments Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

Details

The Generalized Simpson's Entropy (Zhang and Zhou, 2010) of order \( r \) is, in the species accumulation curve, the probability for the individual sampled in rank \( r + 1 \) to belong to a new species. It is a measure of diversity so long as \( r \) is lower than the number of species (Grabchak et al., 2016).

Bias correction requires the number of individuals. Use bcGenSimpson. It is limited to orders \( r \) less than or equal to the number of individuals in the community.

The effective number of species GenSimpsonD (explicit diversity) has been derived by Grabchak et al. (2016).

The functions are designed to be used as simply as possible. GenSimpson is a generic method. If its first argument is an abundance vector, an integer vector or a numeric vector which does not sum to 1, the bias corrected function bcGenSimpson is called. Explicit calls to bcGenSimpson (with bias correction) or to GenSimpson.ProbaVector (without correction) are possible to avoid ambiguity. The .integer and .numeric methods accept Ps or Ns arguments instead of NorP for backward compatibility.
**Value**

A named number equal to the calculated index or diversity. The name is either "Biased" or "Unbiased", depending on the estimator used.

**Note**

The unbiased estimator is calculated by the `GenSimp.z` function of the `EntropyEstimation` package.

**Author(s)**

Eric Marcon <Eric.Marcon@ecofog.gf>

**References**


**Examples**

```
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou6418)
# Ns is the total number of trees per species
Ns <- as.AbdVector(Paracou6418.MC$Ns)
# Species probabilities
Ps <- as.ProbaVector(Paracou6418.MC$Ns)
# Whittaker plot
plot(Ns)

# Calculate GenSimpson entropy of order 1, equal to Simpson's index of diversity
GenSimpson(Ps, 1)
# Calculate an unbiased estimator of GenSimpson diversity of order 100
GenSimpsonD(Ns, 100)
```

---

**Similarity-based entropy of a community**

**Description**

Calculates the entropy of order $q$ of a probability vector according to a similarity matrix.
Usage

HQz(NorP, q = 1, Z = diag(length(NorP)), ...)  
bchqz(Ns, q = 1, Z = diag(length(Ns)), Correction = "Best", SampleCoverage = NULL,  
  CheckArguments = TRUE)  
## S3 method for class 'ProbaVector'  
HQz(NorP, q = 1, Z = diag(length(NorP)),  
  ..., CheckArguments = TRUE, Ps = NULL)  
## S3 method for class 'AbdVector'  
HQz(NorP, q = 1, Z = diag(length(NorP)), Correction = "Best",  
  ..., CheckArguments = TRUE, Ns = NULL)  
## S3 method for class 'integer'  
HQz(NorP, q = 1, Z = diag(length(NorP)), Correction = "Best",  
  ..., CheckArguments = TRUE, Ns = NULL)  
## S3 method for class 'numeric'  
HQz(NorP, q = 1, Z = diag(length(NorP)), Correction = "Best",  
  ..., CheckArguments = TRUE, Ps = NULL, Ns = NULL)

Arguments

Ps A probability vector, summing to 1.
Ns A numeric vector containing species abundances.
NorP A numeric vector, an integer vector, an abundance vector (AbdVector) or a probability vector (ProbaVector). Contains either abundances or probabilities.
q A number: the order of entropy. Default is 1.
Z A relatedness matrix, i.e. a square matrix whose terms are all positive, strictly positive on the diagonal. Generally, the matrix is a similarity matrix, i.e. the diagonal terms equal 1 and other terms are between 0 and 1. Default is the identity matrix to calculate neutral entropy.
Correction A string containing one of the possible corrections: "None" (no correction), "ChaoShen", "MarconZhang" or "Best", the default value. The "MarconZhang" correction assumes a similarity matrix.
SampleCoverage The sample coverage of Ns calculated elsewhere. Used to calculate the gamma diversity of meta-communities, see details.
... Additional arguments. Unused.
CheckArguments Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

Details

Entropy is calculated following Leinster and Cobbold (2012) after Ricotta and Szeidl (2006): it is the entropy of order q of the community, using species ordinariness as the information function.

A similarity matrix is used (as for Dqz), not a distance matrix as in Ricotta and Szeidl (2006). See the example.

Bias correction requires the number of individuals. Use bcHQz and choose the Correction. Correction techniques are from Marcon et al. (2014).
Currently, the "Best" correction is the max value of "ChaoShen" and "MarconZhang". The functions are designed to be used as simply as possible. Hqz is a generic method. If its first argument is an abundance vector, an integer vector or a numeric vector which does not sum to 1, the bias corrected function bcHqz is called. Explicit calls to bcHqz (with bias correction) or to Hqz.ProbaVector (without correction) are possible to avoid ambiguity. The .integer and .numeric methods accept Ps or Ns arguments instead of NorP for backward compatibility. The size of a metacommunity (see MetaCommunity) is unknown so it has to be set according to a rule which does not ensure that its abundances are integer values. Then, classical bias-correction methods do not apply. Providing the SampleCoverage argument allows applying the "ChaoShen" correction to estimate quite well the entropy. DivPart and GammaEntropy functions use this tweak.

Value
A named number equal to the calculated entropy. The name is that of the bias correction used.

Author(s)
Eric Marcon <Eric.Marcon@ecofog.gf>

References

See Also
Dqz, PhyloEntropy

Examples
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Prepare the similarity matrix
DistanceMatrix <- as.matrix(EightSpTree$Wdist^2/2)
# Similarity can be 1 minus normalized distances between species
Z <- 1 - DistanceMatrix/max(DistanceMatrix)
# Calculate diversity of order 2
Ps <- EightSpAbundance/sum(EightSpAbundance)
Hqz(Ps, 2, Z)
# Equal to normalized Rao quadratic entropy when q=2
Rao(Ps, EightSpTree)/max(DistanceMatrix)
# But different from PhyloEntropy for all other q, e.g. 1
Hqz(Ps, 1, Z)
summary(PhyloEntropy(Ps, 1, EightSpTree))
Description

Calculates the similarity-based beta entropy of order \( q \) of a community belonging to a metacommunity.

Usage

\[
\text{HqzBeta}(\text{NorP}, \text{NorPexp} = \text{NULL}, q = 1, Z = \text{diag(length(NorP))}, \ldots) \\
\text{bchqzBeta}(\text{Ns}, \text{Nexp} = \text{NULL}, q = 1, Z = \text{diag(length(Ns))}, \text{Correction} = "\text{Best}" , \\
\text{CheckArguments} = \text{TRUE}) \\
\]

## S3 method for class 'ProbaVector'

\[
\text{HqzBeta}(\text{NorP}, \text{NorPexp} = \text{NULL}, q = 1, Z = \text{diag(length(NorP))}, \ldots, \text{CheckArguments} = \text{TRUE}, \text{Ps} = \text{NULL}, \text{Pexp} = \text{NULL}) \\
\]

## S3 method for class 'AbdVector'

\[
\text{HqzBeta}(\text{NorP}, \text{NorPexp} = \text{NULL}, q = 1, Z = \text{diag(length(NorP))}, \text{Correction} = "\text{Best}" , \\
\ldots, \text{CheckArguments} = \text{TRUE}, \text{Ns} = \text{NULL}, \text{Nexp} = \text{NULL}) \\
\]

## S3 method for class 'integer'

\[
\text{HqzBeta}(\text{NorP}, \text{NorPexp} = \text{NULL}, q = 1, Z = \text{diag(length(NorP))}, \text{Correction} = "\text{Best}" , \\
\ldots, \text{CheckArguments} = \text{TRUE}, \text{Ns} = \text{NULL}, \text{Nexp} = \text{NULL}) \\
\]

## S3 method for class 'numeric'

\[
\text{HqzBeta}(\text{NorP}, \text{NorPexp} = \text{NULL}, q = 1, Z = \text{diag(length(NorP))}, \text{Correction} = "\text{Best}" , \\
\ldots, \text{CheckArguments} = \text{TRUE}, \text{Ps} = \text{NULL}, \text{Ns} = \text{NULL}, \text{Pexp} = \text{NULL}, \text{Nexp} = \text{NULL}) \\
\]

Arguments

- **Ps**: The probability vector of species of the community.
- **Pexp**: The probability vector of species of the metacommunity.
- **Ns**: A numeric vector containing species abundances of the community.
- **Nexp**: A numeric vector containing species abundances of the metacommunity.
- **NorP**: A numeric vector, an integer vector, an abundance vector (AbdVector) or a probability vector (ProbaVector). Contains either abundances or probabilities of the community.
- **NorPexp**: A numeric vector, an integer vector, an abundance vector (AbdVector) or a probability vector (ProbaVector). Contains either abundances or probabilities of the metacommunity.
- **q**: A number, the order of diversity. Default is 1.
- **Z**: A relatedness matrix, \textit{i.e.} a square matrix whose terms are all positive, strictly positive on the diagonal. Generally, the matrix is a similarity matrix, \textit{i.e.} the diagonal terms equal 1 and other terms are between 0 and 1. Default is the identity matrix to calculate neutral entropy.
- **Correction**: A string containing one of the possible corrections: currently, no correction is available so "Best", the default value, is equivalent to "None".
Additional arguments. Unused.

CheckArguments Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

Details

The derivation of similarity-based beta entropy can be found in Marcon et al. (2014).

Bias correction requires the number of individuals.

Note that beta entropy value is related to alpha entropy (if q is not 1) and cannot be compared across communities (Jost, 2007). Beta entropy of a community is not meaningful in general, do rather calculate the BetaDiversity of the metacommunity.

The functions are designed to be used as simply as possible. HqzBeta is a generic method. If its first argument is an abundance vector, an integer vector or a numeric vector which does not sum to 1, the bias corrected function bchqzBeta is called. Explicit calls to bchqzBeta (with bias correction) or to HqzBeta.ProbaVector (without correction) are possible to avoid ambiguity. The .integer and .numeric methods accept Ps or Ns arguments instead of NorP for backward compatibility.

Value

A named number equal to the calculated entropy. The name is that of the bias correction used.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


Examples

# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Ps is the vector of probabilities
Ps <- as.ProbaVector(Paracou618.MC$Ps)
# Probability distribution of the first plot
Ps1 <- as.ProbaVector(Paracou618.MC$Psi[, 1])
# Prepare the similarity matrix
DistanceMatrix <- as.matrix(Paracou618.dist)
# Similarity can be 1 minus normalized distances between species
Z <- 1 - DistanceMatrix/max(DistanceMatrix)
# Divergence of order 2 between plot 1 and the whole forest
HqzBeta(Ps1, Ps, q=2, Z)
**Hurlbert's Index and Explicit Diversity**

**Description**

Calculates the Hurlbert entropy of order $k$ of a probability or abundance vector, and its effective number of species.

**Usage**

```r
hurlbert(NorP, k = 2, ...)  # S3 method for class 'ProbaVector'
bcHurlbert(Ns, k = 2, CheckArguments = TRUE)

hurlbertD(NorP, k = 2, ...)  # S3 method for class 'ProbaVector'
bcHurlbertD(Ns, k = 2, CheckArguments = TRUE)
```

**Arguments**

- **Ps**  
  A probability vector, summing to 1.
- **Ns**  
  A numeric vector containing species abundances.
- **NorP**  
  A numeric vector, an integer vector, an abundance vector (AbdVector) or a probability vector (ProbaVector). Contains either abundances or probabilities.
- **k**  
  A number: the order of diversity. Default is 2 for Simpson’s diversity.
Additional arguments. Unused.

CheckArguments Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

Details

Hurlbert’s index of diversity (1971) of order \( k \) is the expected number of species in a sample of size \( k \).

Bias correction requires the number of individuals. Use \( \text{bcHurlbert} \). It is limited to orders \( k \) less than or equal to the number of individuals in the community.

The effective number of species \( \text{HurlbertD} \) (explicit diversity) has been derived by Dauby & Hardy (2012). It is calculated numerically. \( \text{bcHurlbertD} \) calculates it from the bias-corrected index \( \text{bcHurlbert} \).

The functions are designed to be used as simply as possible. \( \text{Hurlbert} \) is a generic method. If its first argument is an abundance vector, an integer vector or a numeric vector which does not sum to 1, the bias corrected function \( \text{bcHurlbert} \) is called. Explicit calls to \( \text{bcHurlbert} \) (with bias correction) or to \( \text{Hurlbert} \) \( \text{ProbaVector} \) (without correction) are possible to avoid ambiguity. The \text{.integer} and \text{.numeric} methods accept \( \text{Ps} \) or \( \text{Ns} \) arguments instead of \( \text{NorP} \) for backward compatibility.

Value

A named number equal to the calculated index or diversity. The name is either "Biased" or "Unbiased", depending on the estimator used.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


Examples

```
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Ns is the total number of trees per species
Ns <- as.AbdVector(Paracou618.MC$Ns)
# Species probabilities
Ps <- as.ProbaVector(Paracou618.MC$Ns)
# Whittaker plot
plot(Ns)

# Calculate Hurlbert entropy of order 2, equal to Simpson's index of diversity
Hurlbert(Ps, 2)
```
# Calculate an unbiased estimator of Hurlbert entropy of order 2
Hurlbert(Ns, 2)

---

**KLq**

*Generalized Kullback-Leibler divergence*

**Description**

Calculates the generalized Kullback-Leibler divergence between an observed and an expected probability distribution.

**Usage**

```r
KLq(Ps, Pexp, q = 1, CheckArguments = TRUE)
```

**Arguments**

- **Ps**: The observed probability vector.
- **Pexp**: The expected probability vector.
- **q**: A number: the order of entropy. Default is 1.
- **CheckArguments**: Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

**Details**

The generalized Kullback-Leibler divergence (Borland et al., 1998) converges to the Kullback-Leibler divergence (Kullback and Leibler, 1951) when \( q \) tends to 1. It is used to calculate the generalized beta entropy (Marcon et al., 2014).

**Value**

A number equal to the generalized Kullback-Leibler divergence between the probability distributions.

**Author(s)**

Eric Marcon <Eric.Marcon@ecofog.gf>

**References**


lnq

See Also

TsallisBeta

Examples

# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Ps is the vector of probabilities
Ps <- Paracou618.MC$Ps
# Probability distribution of the first plot
Psi <- Paracou618.MC$Psi[, 1]
# Divergence of order 2 between the first plot and the whole forest
KLq(Psi, Ps, 2)

Description

Calculates the deformed logarithm of order $q$.

Usage

lnq(x, q)
lnq.CommunityProfile(Profile)

Arguments

x
A numeric vector.
Profile
A CommunityProfile.
q
A number.

Details

The deformed logarithm is defined as $\ln_q x = \frac{x^{(1-q)} - 1}{(1-q)}$.

The shape of the deformed logarithm is similar to that of the regular one. $\ln_1 x = \log x$.

For $q > 1$, $\ln_q (+\infty) = \frac{1}{(q-1)}$.

lnq.CommunityProfile calculates the deformed logarithm of a CommunityProfile. Its $x$ item (the order of diversity) is kept unchanged whilst other items are set to their logarithm of order $x$.

Thus, a diversity profile is transformed into an entropy profile.

Value

A vector of the same length as $x$ containing the transformed values or a CommunityProfile.
Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


See Also

expq

Examples

curve(log(x), 0, 1, lty=1)
curve(lnq(x, 2), 0, 1, lty=2, add=TRUE)
curve(lnq(x, 3), 0, 1, lty=3, add=TRUE)
legend("topleft", legend = c("log(x)", "ln2(x)", "ln3(x)"), lty = c(1, 2, 3), inset=0.02)

MC Utilities

Manipulation of meta-communities

Description

Tools to manipulate meta-communities. From a list of meta-communities, MergeMC creates a meta-community whose communities are each original metacommunity. MergeC creates a metacommunity whose communities are each original community. ShuffleMC randomly assigns original communities to a metacommunity, keeping original weights, and returns a list of meta-communities.

Usage

MergeMC(MClist, Weights = rep(1, length(MClist)), CheckArguments = TRUE)
MergeC(MClist, Weights = rep(1, length(MClist)), CheckArguments = TRUE)
ShuffleMC(MClist, Weights = rep(1, length(MClist)), CheckArguments = TRUE)

Arguments

MList A list of MetaCommunity objects.
Weights A vector of numbers containing the weight of each metacommunity of the list. It does not have to be normalized to sum to 1.
CheckArguments Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.
Details

MergeMC is used for hierarchical partitioning of diversity. The gamma diversity of communities of the list becomes alpha diversity of the merged meta-community.

MergeC creates a new meta-community by mixing original ones. Original communities are kept, their new weight is the product of their original weight and the weight of their original meta-community.

ShuffleMC is used for simulations of the null hypothesis that all metacommunities of the list are identical.

Value

MergeMC and MergeC return a MetaCommunity.
ShuffleMC returns a list of MetaCommunity objects.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

See Also

MetaCommunity

Examples

# First meta-community
(df <- data.frame(C1 = c(10, 10, 10, 10), C2 = c(0, 20, 35, 5),
C3 = c(25, 15, 0, 2), row.names = c("sp1", "sp2", "sp3", "sp4")))
w <- c(1, 2, 1)
MC1 <- MetaCommunity(Abundances = df, Weights = w)

# Second meta-community
(df <- data.frame(C1 = c(10, 4), C2 = c(3, 4), row.names = c("sp1", "sp5")))
w <- c(3, 2)
MC2 <- MetaCommunity(Abundances = df, Weights = w)

# Merge communities
plot(MergeC(list(MC1, MC2)), main="Merged communities")

# Merge metacommunities
plot(MergeMC(list(MC1, MC2)), main="Merged meta-communities")
smc <- ShuffleMC(list(MC1, MC2))
plot(MergeMC(smc), main="Shuffled, then Merged meta-communities")

MCdiversity

Meta-Community diversity class.

Description

Methods for objects of type "MCdiversity".
Usage

is.MCdiversity(x)
## S3 method for class 'MCdiversity'
plot(x, ...)
## S3 method for class 'MCdiversity'
autoplot(object, ...)
## S3 method for class 'MCdiversity'
summary(object, ...)

Arguments

x
An object to be tested or plotted.

object
A MCdiversity object to be summarized or plotted.

... Additional arguments to be passed to the generic methods.

Value

Meta-community diversity objects are lists containing:

MetaCommunity The name of the MetaCommunity object containing inventory data.
Type The type of diversity ("alpha", "beta" or "gamma").
Order The order of diversity q.
Correction The estimation bias correction used to calculate diversity.
Tree The phylogenetic or functional tree used to calculate phylodiversity.
Normalized Logical. Indicates whether phylodiversity is normalized or proportional to the height of the tree.
Weights A vector containing the weights of communities.
Communities A vector containing the diversity of communities.
Total The total diversity.

is.MCdiversity returns TRUE if the object is of class MCdiversity.
summary.MCdiversity returns a summary of the object's value.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>
Meta-Community entropy class.

Description

Methods for objects of type "MCentropy".

Usage

```r
is.MCentropy(x)
## S3 method for class 'MCentropy'
plot(x, ...)
## S3 method for class 'MCentropy'
autoplot(object, ...)
## S3 method for class 'MCentropy'
summary(object, ...)
```

Arguments

- `x`: An object to be tested or plotted.
- `object`: A MCentropy object to be summarized or plotted.
- `...`: Additional arguments to be passed to the generic methods.

Value

Meta-community entropy objects are lists containing:

- `MetaCommunity`: The name of the MetaCommunity object containing inventory data.
- `Method`: The method used to calculate entropy ("HCDT", "Similarity-based").
- `Type`: The type of entropy ("alpha", "beta" or "gamma").
- `Order`: The order of entropy $q$.
- `Correction`: The estimation bias correction used to calculate entropy.
- `Tree`: The phylogenetic or functional tree used to calculate phyloentropy.
- `Normalized`: Logical. Indicates whether phyloentropy is normalized or proportional to the height of the tree.
- `Z`: The matrix used to calculate similarity-based entropy.
- `Weights`: A vector containing the weights of communities.
- `Communities`: A vector containing the entropy of communities.
- `Total`: The total entropy.

`is.MCentropy` returns TRUE if the object is of class MCentropy.

`summary.MCentropy` returns a summary of the object’s value.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>
MetaCommunity

**MetaCommunity class**

**Description**
Methods for objects of type "MetaCommunity".

**Usage**

```r
MetaCommunity(Abundances, Weights = rep(1, ncol(Abundances)))
is.MetaCommunity(x)
## S3 method for class 'MetaCommunity'
summary(object, ...)
## S3 method for class 'MetaCommunity'
plot(x, ...)
```

**Arguments**

- `Abundances` A dataframe containing the number of observations (lines are species, columns are communities). The first column of the dataframe may contain the species names.
- `Weights` A vector of positive numbers equal to community weights or a dataframe containing a vector named `Weights`. It does not have to be normalized. Weights are equal by default.
- `x` An object to be tested or plotted.
- `object` A MetaCommunity object to be summarized.
- `...` Additional arguments to be passed to the generic methods.

**Details**

In the entropart package, individuals of different "species" are counted in several "communities" which are aggregated to define a "meta-community". This is a naming convention, which may correspond to plots in a forest inventory or any data organized the same way.

Alpha and beta entropies of communities are summed according to `Weights` and the probability to find a species in the metacommunity is the weighted average of probabilities in communities.

The simplest way to import data is to organize it into two text files. The first file should contain abundance data: the first column named `Species` for species names, and a column for each community.

The second file should contain the community weights in two columns. The first one, named `Communities` should contain their names and the second one, named `Weights`, their weights.

Files can be read and data imported by code such as:

```r
Abundances <- read.csv(file="Abundances.csv", row.names = 1)
Weights <- read.csv(file="Weights.csv")
MC <- MetaCommunity(Abundances, Weights)
```
An object of class `MetaCommunity` is a list:

- **Ns**
  A matrix containing abundance data, species in line, communities in column.

- **Ns**
  A vector containing the number of individuals of each species.

- **Ni**
  A vector containing the number of individuals of each community.

- **N**
  The total number of individuals.

- **Psi**
  A matrix whose columns are the probability vectors of communities (each of them sums to 1).

- **Wi**
  A vector containing the normalized community weights (sum to 1).

- **Ps**
  A vector containing the probability vector of the metacommunity.

- **Nspecies**
  The number of species.

- **Ncommunities**
  The number of communities.

- **SampleCoverage**
  The sample coverage of the metacommunity.

- **SampleCoverage.communities**
  A vector containing the sample coverages of each community.

`is.MetaCommunity` returns TRUE if the object is of class `MetaCommunity`.

`summary.MetaCommunity` returns a summary of the object's value.

`plot.MetaCommunity` plots it.

**Author(s)**

Eric Marcon <Eric.Marcon@ecofog.gf>

**Examples**

```r
# Use BCI data from vegan package
if (require(vegan, quietly = TRUE)) {
  # Load BCI data (number of trees per species in each 1-ha plot of a tropical forest)
  data(BCI)
  # BCI dataframe must be transposed (its lines are plots, not species)
  BCI.df <- as.data.frame(t(BCI))
  # Create a metacommunity object from a matrix of abundances and a vector of weights
  # (here, all plots have a weight equal to 1)
  MC <- MetaCommunity(BCI.df)
}
```
Optimal.Similarity

**Optimal scale parameter to transform a distance matrix into a similarity matrix**

**Description**

Calculates the scale parameter \( u \) that maximizes the variance of the similarity matrix \( \exp(-u \times \text{DistanceMatrix}) \).

**Usage**

```r
Optimal.Similarity(Distance, CheckArguments = TRUE)
```

**Arguments**

- **Distance**: A distance matrix, i.e. a square matrix with zeros on its diagonal or a `dist` object.
- **CheckArguments**: Logical; if `TRUE`, the function arguments are verified. Should be set to `FALSE` to save time when the arguments have been checked elsewhere.

**Details**

The similarity matrix used by `dqz` can be optimized following Marcon et al. (2014) such that the variance of similarities between pairs of species is maximized. See the example.

**Value**

A list:

- **u**: The optimal scale \( u \).
- **Matrix**: The optimal similarity matrix \( Z \).

**Author(s)**

Eric Marcon <Eric.Marcon@ecofog.gf>

**References**


**See Also**

`Dqz`
Examples

```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)

# Prepare the similarity matrix. The square root of Paracou618.dist is euclidean.
optimal <- Optimal.Similarity(sqrt(Paracou618.dist))

# Optimal scale
optimal$s

# Calculate diversity of order 2
bcdqz(Paracou618.MC$Ns, 2, optimal$Matrix)
```

Description

This dataset is from Paracou field station, French Guiana, managed by Cirad. Traits are detailed in Marcon and Herault (2014), the distance matrix was built following Paine et al. (2011).

Usage

```r
data(Paracou618)
```

Format

An object of class `dist`.

Source

Permanent data census of Paracou.

References


Examples

```r
data(Paracou618)
plot(density(Paracou618.dist, from=0), main="Distances between species")
```
Functional tree of species of Paracou field station plots 6 and 18, two 1-ha plots inventoried by the Bridge project.

Description

This dataset is from Paracou field station, French Guiana, managed by Cirad. Traits are detailed in Marcon and Herault (2014), the tree was built following Paine et al. (2011), based on Paracou618.dist.

Usage

data(Paracou618)

Format

An object of class hclust.

Source

Permanent data census of Paracou.

References


Examples

data(Paracou618)
plot(Paracou618.Functional)
Paracou618.MC

Paracou field station plots 6 and 18, two 1-ha plots inventoried by the Bridge project.

Description
This dataset is from Paracou field station, French Guiana, managed by Cirad.

Usage
data(Paracou618)

Format
An object of class MetaCommunity made of two communities and 425 species.

Source
Permanent data census of Paracou and Marcon et al. (2012).

References

Examples

data(Paracou618)
summary(Paracou618.MC)

Paracou618.Taxonomy
Taxonomy (Family - Genus - Species) of Paracou field station plots 6 and 18, two 1-ha plots inventoried by the Bridge project.

Description
This dataset is from Paracou field station, French Guiana, managed by Cirad.

Usage
data(Paracou618)
Format
An object of class `phylo` containing a taxonomy.

Source
Permanent data census of Paracou.

References

Examples
```r
data(Paracou618)
plot(Paracou618.Taxonomy, type="fan", show.tip.label=FALSE)
```

**PDFD**

*Phylogenetic Diversity / Functional Diversity of a Community*

Description
Calculates Faith’s PD / Petchey and Gaston’ FD of a community described by a probability vector and a phylogenetic / functional tree.

Usage
```r
PDFD(Ps, Tree, CheckArguments = TRUE)
```

Arguments
- **Ps**: A probability vector, summing to 1.
- **Tree**: An object of class `hclust`, `phylo`, `phylog` or `PPTree`. The tree must be ultrametric.
- **CheckArguments**: Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

Details
PD and FD are defined as the total length of the branches of the tree.
The probability vector is used to select branches: branches with probability 0 are eliminated.
Bias correction requires the number of individuals to estimate sample Coverage.
Use `bcPhyloDiversity(Ps, 0, Tree)` and choose the Correction.
Value

A named number equal to the calculated diversity. The name is that of the bias correction used.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


See Also

bcPhyloDiversity

Examples

```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest
# and their taxonomy)
data(Paracou618)
# Ps is the vector of probabilities
Ps <- Paracou618.MC$Ps
# Calculate the phylogenetic Shannon diversity of the plot
PDFD(Ps, Paracou618.Taxonomy)
```

---

**PhyloApply**

*Apply a Function over a Phylogenetic Tree*

**Description**

Cuts the tree into slices separated by nodes, applies the function to each slice and returns the weighted (by slice lengths) sum of the results.

**Usage**

```
PhyloApply(Tree, FUN, NorP, Normalize = TRUE, dfArgs = NULL,
           ..., CheckArguments = TRUE)
```
**Arguments**

- **Tree**
  - An object of class `hclust, phylo, phylog` or `PPTree`. The tree must be ultrametric.

- **FUN**
  - The function to be applied to each interval of the tree.

- **NorP**
  - A numeric vector or a two-column matrix. Contains either abundances or probabilities. Two-column matrices should contain the observed abundances (or probabilities) in the first column and the expected ones in the second column, to allow using beta diversity functions.

- **Normalize**
  - If TRUE (default), the Total value returned by Function is normalized by the height of the tree (it is the weighted average value of the result in each slice). If FALSE, it is the unnormalized weighted sum of the results.

- **dfArgs**
  - A dataframe. Columns are arguments for FUN: their names are those of valid arguments. Values will be passed to FUN in each slice of the tree, starting from the tips. The number of lines must equal the number of slices.

- **...**
  - Further arguments to pass to Function.

- **CheckArguments**
  - Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

**Details**

This function is generally not used directly. It is a tool to calculate *PhyloEntropy* and *PhyloDiversity*. Intervals (slices) separate two cuts (nodes) in a tree: no node is found at heights contained in an interval.

Objects of class `PPTree` are returned by `Preprocess.Tree`.

...allow passing arguments to the function but they can’t change along the tree. If necessary, `dfArgs` allow passing a different value for each slice of the tree.

**Value**

An object of class `PhyloValue`. It is a list:

- **Distribution**
  - The distribution used to calculate the value

- **Function**
  - The function used to calculate the value

- **Tree**
  - The functional or phylogenetic tree used to calculate the value

- **Normalized**
  - Logical. Indicates whether phylovalue is normalized or proportional to the height of the tree.

- **Cuts**
  - A named vector containing values along the tree. Names are cut ends, *i.e.* the ends of intervals (the first interval starts at 0 for leaves, the max value is the height of the tree).

- **Corrections**
  - A named vector containing the correction used by FUN to obtain each value of Cuts. Names are those of Cuts.

- **Total**
  - The total value, multiplied by the tree height if `Normalize` is FALSE.
**PhyloBetaEntropy**

**Author(s)**

Eric Marcon <Eric.Marcon@ecofog.gf>

**References**


**See Also**

`Preprocess.Tree`

**Examples**

```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest and their taxonomy)
data(Paracou618)
# Plot the taxonomy
plot(Paracou618$Taxonomy, type="fan", show.tip.label=FALSE)
# Calculate the mean number of trees (individuals) per species
# (cuts are 1=species, 2=genus, 3=family)
summary(phyloapply(Paracou618$Taxonomy, mean, Paracou618$MC$Ns, TRUE))
```

**Description**

Calculates the phylogenetic beta entropy of order \( q \) of a community belonging to a metacommunity.

**Usage**

```r
PhyloBetaEntropy(NorP, NorPexp = NULL, q = 1, Tree, Normalize = TRUE, ...)
cbcPhyloBetaEntropy(Ns, Nexp, q = 1, Tree, Normalize = TRUE, Correction = "Best", CheckArguments = TRUE)
## S3 method for class 'ProbaVector'
PhyloBetaEntropy(NorP, NorPexp = NULL, q = 1, Tree, Normalize = TRUE, ...
CheckArguments = TRUE, Ps = NULL, Pexp = NULL)
## S3 method for class 'AbdVector'
PhyloBetaEntropy(NorP, NorPexp = NULL, q = 1, Tree, Normalize = TRUE, ...
CheckArguments = TRUE, Ns = NULL, Nexp = NULL)
## S3 method for class 'integer'
PhyloBetaEntropy(NorP, NorPexp = NULL, q = 1, Tree, Normalize = TRUE, ...
CheckArguments = TRUE, Ns = NULL, Nexp = NULL)
## S3 method for class 'numeric'
PhyloBetaEntropy(NorP, NorPexp = NULL, q = 1, Tree, Normalize = TRUE, ...
CheckArguments = TRUE, Ps = NULL, Ns = NULL, Pexp = NULL)
```

---

**Phylogenetic Beta Entropy of a community**

**Description**

Calculates the phylogenetic beta entropy of order \( q \) of a community belonging to a metacommunity.

**Usage**

```r
PhyloBetaEntropy(NorP, NorPexp = NULL, q = 1, Tree, Normalize = TRUE, ...)
cbcPhyloBetaEntropy(Ns, Nexp, q = 1, Tree, Normalize = TRUE, Correction = "Best", CheckArguments = TRUE)
## S3 method for class 'ProbaVector'
PhyloBetaEntropy(NorP, NorPexp = NULL, q = 1, Tree, Normalize = TRUE, ...
CheckArguments = TRUE, Ps = NULL, Pexp = NULL)
## S3 method for class 'AbdVector'
PhyloBetaEntropy(NorP, NorPexp = NULL, q = 1, Tree, Normalize = TRUE, ...
CheckArguments = TRUE, Ns = NULL, Nexp = NULL)
## S3 method for class 'integer'
PhyloBetaEntropy(NorP, NorPexp = NULL, q = 1, Tree, Normalize = TRUE, ...
CheckArguments = TRUE, Ns = NULL, Nexp = NULL)
## S3 method for class 'numeric'
PhyloBetaEntropy(NorP, NorPexp = NULL, q = 1, Tree, Normalize = TRUE, ...
CheckArguments = TRUE, Ps = NULL, Ns = NULL, Pexp = NULL)
```
**Arguments**

- **Ps**  
The probability vector of species of the community.
- **Pexp**  
The probability vector of species of the metacommunity.
- **Ns**  
A numeric vector containing species abundances of the community.
- **Nexp**  
A numeric vector containing species abundances of the metacommunity.
- **NorP**  
A numeric vector, an integer vector, an abundance vector (AbdVector) or a probability vector (ProbaVector). Contains either abundances or probabilities of the community.
- **NorPexp**  
A numeric vector, an integer vector, an abundance vector (AbdVector) or a probability vector (ProbaVector). Contains either abundances or probabilities of the metacommunity.
- **q**  
A number: the order of entropy. Default is 1.
- **Tree**  
An object of class hclust, phylo, phylog or PPTree. The tree must be ultrametric.
- **Normalize**  
If TRUE (default), the entropy returned by the function is normalized by the height of the tree (it is the weighted average value of the entropy in each slice). If FALSE, it is the unnormalized weighted sum of the results.
- **Correction**  
A string containing one of the possible corrections: currently, only "ChaoShen". "Best" is the default value, it is equivalent to "ChaoShen".
- **...**  
Additional arguments. Unused.
- **CheckArguments**  
Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

**Details**

The phylogenetic entropy is the generalization of HCDT entropy to unequal species distances (Pavoine et al., 2009).

Calculation relies on TsallisBeta and PhyloApply.

Bias correction requires the number of individuals to estimate sample Coverage. Use bcPhyloBetaEntropy and choose the Correction.

Note that beta entropy value is related to alpha entropy (if q is not 1) and cannot be compared accross communities (Jost, 2007). Beta entropy of a community is not meaningful in general, do rather calculate the Phylodiversity of the metacommunity.

The functions are designed to be used as simply as possible. PhyloBetaEntropy is a generic method. If its first argument is an abundance vector, an integer vector or a numeric vector which does not sum to 1, the bias corrected function bcPhyloBetaEntropy is called. Explicit calls to bcPhyloBetaEntropy (with bias correction) or to PhyloBetaEntropy.ProbaVector (without correction) are possible to avoid ambiguity. The .integer and .numeric methods accept Ps or Ns arguments instead of NorP for backward compatibility.

**Value**

A PhyloEntropy object containing entropy values at each cut of the tree.
Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


See Also

TsallisBeta, bcPhyloBetaEntropy, PhyloDiversity

Examples

```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest
# and their taxonomy)
data(Paracou618)
# Ps is the vector of probabilities
Ps <- as.ProbaVector(Paracou618.MC$Ps)
# Probability distribution of the first plot
Ps1 <- as.ProbaVector(Paracou618.MC$Ps[, 1])
# Calculate the phylogenetic Shannon beta entropy of the plot
summary(PhyloBetaEntropy(Ps1, Ps, 1, Paracou618.Taxonomy) -> e)
plot(e)

# Ns is the vector of abundances of the metacommunity
Ns <- as.AbdVector(Paracou618.MC$Ns)
# Abundances in the first plot
Ns1 <- as.AbdVector(Paracou618.MC$Ns[, 1])
# Calculate the phylogenetic Shannon beta entropy of the plot
summary(bcPhyloBetaEntropy(Ns1, Ns, 1, Paracou618.Taxonomy, Correction = "Best") -> e)
plot(e)
```

Description

Calculates the phylogenetic diversity of order q of a probability vector.
Usage

PhyloDiversity(NorP, q = 1, Tree, Normalize = TRUE, ...)  
bcPhylodiversity(Ns, q = 1, Tree, Normalize = TRUE, Correction = "Best",  
    CheckArguments = TRUE)  
## S3 method for class 'ProbaVector'
PhyloDiversity(NorP, q = 1, Tree, Normalize = TRUE,  
    ..., CheckArguments = TRUE, Ps = NULL)  
## S3 method for class 'AbdVector'
PhyloDiversity(NorP, q = 1, Tree, Normalize = TRUE,  
    Correction = "Best", ..., CheckArguments = TRUE, Ns = NULL)  
## S3 method for class 'integer'
PhyloDiversity(NorP, q = 1, Tree, Normalize = TRUE,  
    Correction = "Best", ..., CheckArguments = TRUE, Ns = NULL)  
## S3 method for class 'numeric'
PhyloDiversity(NorP, q = 1, Tree, Normalize = TRUE,  
    Correction = "Best", ..., CheckArguments = TRUE, Ps = NULL, Ns = NULL)  
is.Phylodiversity(x)  
## S3 method for class 'Phylodiversity'
summary(object, ...)  

Arguments

Ps  A probability vector, summing to 1.
Ns  A numeric vector containing species abundances.
NorP A numeric vector, an integer vector, an abundance vector (AbdVector) or a probability vector (ProbaVector). Contains either abundances or probabilities.
q   A number: the order of diversity. Default is 1.
Tree An object of class hclust, phylo, phylog or PPTree. The tree must be ultrametric.
Normalize If TRUE (default), the Total diversity is not affected by the height of the tree. If FALSE, it is proportional to the height of the tree.
Correction A string containing one of the possible corrections: "None" (no correction), "ChaoShen", "Grassberger", "Holste", "Bonachela" or "Best", the default value.
CheckArguments Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.
x   An object to be tested or plotted
object A Phylodiversity object to be summarized.
... Additional arguments to be passed to the generic methods.

Details

The phylogenetic entropy is its generalization of HCDT entropy to unequal species distances (Pavoine et al., 2009).
Diversity is obtained by transforming generalized entropy.

Bias correction requires the number of individuals to estimate sample Coverage. Use bcPhyloDiversity and choose the Correction.

The functions are designed to be used as simply as possible. PhyloDiversity is a generic method. If its first argument is an abundance vector, an integer vector or a numeric vector which does not sum to 1, the bias corrected function bcPhyloDiversity is called. Explicit calls to bcPhyloDiversity (with bias correction) or to PhyloDiversity.ProbaVector (without correction) are possible to avoid ambiguity. The .integer and .numeric methods accept Ps or Ns arguments instead of NorP for backward compatibility.

Value

An object of class Phylodiversity is a list:

Distribution  The distribution used to calculate diversity
Function      The function used to calculate diversity
Tree          The functional or phylogenetic tree used to calculate diversity
Normalized    Logical. Indicates whether phylodiversity is normalized or proportional to the height of the tree.
Type          The type of diversity ("alpha", "beta" or "gamma").
Order         The order of diversity q.
Cuts          A named vector containing values of neutral diversity along the tree. Names are cut ends, i.e. the ends of intervals (the first interval starts at 0 for leaves, the max value is the height of the tree).
Total         A value equal the total diversity (obtained by transforming the total normalized entropy), multiplied by the tree height if Normalize is FALSE.

is.PhyloDiversity returns TRUE if the object is of class PhyloDiversity.
summary.PhyloDiversity returns a summary of the object's value.
PhyloDiversity objects can be plotted by plot.PhyloValue because PhyloDiversity objects are also of class PhyloValue.

Note

The tree must contain all species of the probability vector. If it contains extra species, computation time will just be increased.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


**See Also**

PhyloEntropy, Diversity

**Examples**

```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest
# and their taxonomy)
data(Paracou618)
# Ps is the vector of probabilities
Ps <- as.ProbaVector(Paracou618.MC$Ps)
# Calculate the phylogenetic Shannon diversity of the plot
summary(PhyloDiversity(Ps, 1, Paracou618.Taxonomy) -> d)
plot(d)

# Ns is the vector of abundances of the metacommunity
Ns <- as.AbdVector(Paracou618.MC$Ns)
# Calculate the phylogenetic Shannon diversity of the plot
summary(bcPhyloDiversity(Ns, 1, Paracou618.Taxonomy, Correction = "Best") -> d)
plot(d)
```

**PhyloEntropy**

*Phylogenetic Entropy of a community*

**Description**

Calculates the phylogenetic entropy of order $q$ of a probability vector.

**Usage**

```r
PhyloEntropy(NorP, q = 1, Tree, Normalize = TRUE, ...)
bcPhyloEntropy(Ns, q = 1, Tree, Normalize = TRUE, Correction = "Best",
   SampleCoverage = NULL, CheckArguments = TRUE)
```

**Examples**

```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest
# and their taxonomy)
data(Paracou618)
# Ps is the vector of probabilities
Ps <- as.ProbaVector(Paracou618.MC$Ps)
# Calculate the phylogenetic Shannon diversity of the plot
summary(PhyloDiversity(Ps, 1, Paracou618.Taxonomy) -> d)
plot(d)

# Ns is the vector of abundances of the metacommunity
Ns <- as.AbdVector(Paracou618.MC$Ns)
# Calculate the phylogenetic Shannon diversity of the plot
summary(bcPhyloDiversity(Ns, 1, Paracou618.Taxonomy, Correction = "Best") -> d)
plot(d)
```
PhyloEntropy(NorP, q = 1, Tree, Normalize = TRUE, Correction = "Best", ... , CheckArguments = TRUE, Ps = NULL, Ns = NULL)
is.Phyl0Entropy(x)
## S3 method for class 'Phyl0Entropy'
summary(object, ...)

### Arguments

Ps        A probability vector, summing to 1.
Ns        A numeric vector containing species abundances.
NorP      A numeric vector, an integer vector, an abundance vector (AbdVector) or a probability vector (ProbaVector). Contains either abundances or probabilities.
q         A number: the order of entropy. Default is 1.
Tree      An object of class hclust, phylo, phylog or PPTree. The tree must be ultrametric.
Normalize If TRUE (default), the total entropy returned by the function is normalized by the height of the tree (it is the weighted average value of the entropy in each slice).
If FALSE, it is the unnormalized weighted sum of the results.
Correction A string containing one of the possible corrections supported by Tsallis.
SampleCoverage The sample coverage of Ns calculated elsewhere. Used to calculate the gamma diversity of meta-communities, see details.
CheckArguments Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.
x         An object to be tested or plotted
object    A Phyl0Entropy object to be summarized.
...       Additional arguments to be passed to the generic methods.

### Details

The phylogenetic entropy is its generalization of HCDT entropy to unequal species distances (Pavoine et al., 2009).
Calculation relies on Tsallis and PhyloApply.
Intervals separate two cuts in a tree: no node is found at heights contained in an interval.
Bias correction requires the number of individuals to estimate sample Coverage. Use bcPhyl0Entropy and choose the Correction.
The functions are designed to be used as simply as possible. Phyl0Entropy is a generic method.
If its first argument is an abundance vector, an integer vector or a numeric vector which does not sum to 1, the bias corrected function bcPhyl0Entropy is called. Explicit calls to bcPhyl0Entropy (with bias correction) or to Phyl0Entropy.ProbaVector (without correction) are possible to avoid ambiguity. The .integer and .numeric methods accept Ps or Ns arguments instead of NorP for backward compatibility.
The size of a metacommunity (see MetaCommunity) is unknown so it has to be set according to a rule which does not ensure that its abundances are integer values. Then, classical bias-correction
methods do not apply. Providing the SampleCoverage argument allows applying the "ChaoShen" and "Grassberger" corrections to estimate quite well the entropy. DivPart and GammaEntropy functions use this tweak.

Value

An object of class PhyloEntropy is a list:

- **Distribution**: The distribution used to calculate entropy
- **Function**: The function used to calculate entropy
- **Tree**: The functional or phylogenetic tree used to calculate entropy
- **Normalized**: Logical. Indicates whether phyloentropy is normalized or proportional to the height of the tree.
- **Type**: The type of entropy ("alpha", "beta" or "gamma").
- **Order**: The order of entropy $q$.
- **Cuts**: A named vector containing values of neutral entropy along the tree. Names are cut ends, i.e. the ends of intervals (the first interval starts at 0 for leaves, the max value is the height of the tree).
- **Total**: A value equal the total entropy multiplied by the tree height if Normalize is FALSE.

is.PhyloEntropy returns TRUE if the object is of class PhyloEntropy.

summary.PhyloEntropy returns a summary of the object’s value.

PhyloEntropy objects can be plotted by plot.PhyloValue because PhyloEntropy objects are also of class PhyloValue.

Note

The tree must contain all species of the probability vector. If it contains extra species, computation time will just be increased.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


See Also

Tsallis,PhyloDiversity
Examples

```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest
# and their taxonomy)
data(Paracou618)

# Ps is the vector of probabilities
Ps <- as.ProbaVector(Paracou618.MC$Ps)
# Calculate the phylogenetic Shannon entropy of the plot
summary(PhyloEntropy(Ps, 1, Paracou618.Taxonomy) -> e)
plot(e)

# Ns is the vector of abundances of the metacommunity
Ns <- as.AbdVector(Paracou618.MC$Ns)
# Calculate the phylogenetic Shannon entropy of the plot
summary(bcPhyloEntropy(Ns, 1, Paracou618.Taxonomy, Correction = "Best") -> e)
plot(e)
```

Description

Entropy or diversity against the height of the phylogenetic or functional tree.

Usage

```r
is.Phylodiversity(x)  # S3 method for class 'Phylodiversity'
autoplot(object, xlab = expression(italic("T")), ylab = NULL, main = NULL, ...)
plot(x, xlab = expression(italic("T")), ylab = NULL, main = NULL, ...)
summary(object, ...)
```

Arguments

- `x`: An object of class Phylodiversity, including Phylodiversity and PhyloEntropy objects.
- `xlab`: The X axis label, "T" by default for Time.
- `ylab`: The Y axis label. If NULL (by default), "Entropy" or "Diversity" or nothing is chosen according to the object class.
- `main`: The main title of the plot. If NULL (by default), a default value is used.
- `object`: A Phylodiversity object to be summarized.
- `...`: Additional arguments to be passed to plot.

Details

Phylodiversity objects are the result of PhyloApply.
Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

Examples

# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest
# and their taxonomy)
data(Paracou618)
# Calculate richness along the tree
# (Cuts are 1=species, 2=genus, 3=family)
summary(r <- PhyloApply(Paracou618$Taxonomy, FUN=Richness,
                       NorP=Paracou618$MC$ns, Normalize=TRUE))
autoplot(r)

PPtree

Preprocessed Trees.

Description

Methods for objects of type "PPtree".

Usage

is.PPtree(x)
## S3 method for class 'PPtree'
plot(x, ...)

Arguments

x       An object to be tested or plotted
...

Additional arguments to be passed to the generic methods.

Value

An object of class PPtree is a list:

phyTree   A phylo tree
hTree     A hclust tree
Height    The height of the tree, that is to say the distance between root and leaves
Cuts      A vector. Cut times of the tree (the distance from nodes to leaves)
Intervals A vector. The lengths of intervals between cuts

is.PPtree returns TRUE if the object is of class PPtree.
plot.PPtree plots it.
Note

Versions up to 1.3 contained a phylog tree, now deprecated in ade4. A phylo tree is now used. See the dedicated vignette (vignette("Phylogenies", package="entropart")) for more details.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

Examples

data(Paracou618)
# Preprocess a phylog object
ppt <- Preprocess.Tree(EightSpTree)
# Is it a preprocessed tree?
is.PPtree(ppt)
# Plot it
plot(ppt, hang=-1)

RAC

Fit Distributions to Well-Known Rank Abundance Curves.

Description

Observed distributions are fitted to classical RAC's.

Usage

RAClnorm(Ns, CheckArguments = TRUE)
RACgeom(Ns, CheckArguments = TRUE)
RACseries(Ns, CheckArguments = TRUE)
RACbstick(Ns, CheckArguments = TRUE)

Arguments

Ns A numeric vector containing species abundances.

CheckArguments Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

Details

SpeciesDistribution or integer vectors can be used to fit classical rank-abundance curves (RAC) of classical distributions: "RAClnorm" for log-normal (Preston, 1948), "RACseries" for log-series (Fisher et al., 1943), "RACgeom" for geometric (Motomura, 1932) or "RACbstick" for broken stick (MacArthur, 1957). method returns the estimated parameters of the fitted distribution. The broken stick has no parameter, so the maximum abundance is returned.
Value

A list (the parameters of distributions are returned only if the distribution has been fit):

- Rank: A numeric vector. The ranks of species in the fitted RAC.
- Abundance: The abundance of species in the fitted RAC.
- mu: The expectation of the log-normal distribution
- sigma: The standard deviation of the log-normal distribution
- alpha: Fisher’s alpha in the log-series distribution
- prob: The proportion of resources taken by successive species in the geometric distribution
- max: The maximum abundance in the broken-stick distribution

Note

Fisher’s alpha is estimated to fit the log-series distribution. The estimation is done by the `fisher.alpha` function of package vegan. It may differ substantially from the estimation returned by `optimal.theta` from package untb.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>, Bruno Herault <Bruno.Herault@ecofog.gf>

References


See Also

`rgeom`, `rlnorm`, `rCommunity`, `plot.SpeciesDistribution`

Examples

```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Ns is the total number of trees per species
Ns <- as.AbdVector(Paracou618.MC$Ns)
# Fitted parameters
RACln <- RAClnorm(Ns)
RACln$mu
RACln$sigma
RACgeom(Ns)$prob
RACseries(Ns)$alpha
RACbstick(Ns)$max
```
Rao Quadratic Entropy of a Community

Description

Calculates Rao’s quadratic entropy of a community described by a probability vector and a phylo-
genetic / functional tree.

Usage

Rao(NorP, Tree, ...)  
bcrao(Ns, Tree, Correction="Lande", CheckArguments = TRUE)  
## S3 method for class 'ProbaVector'  
Rao(NorP, Tree, ..., CheckArguments = TRUE, Ps = NULL)  
## S3 method for class 'AbdVector'  
Rao(NorP, Tree, Correction = "Lande", ...,  
  CheckArguments = TRUE, Ns = NULL)  
## S3 method for class 'integer'  
Rao(NorP, Tree, Correction = "Lande", ...,  
  CheckArguments = TRUE, Ns = NULL)  
## S3 method for class 'numeric'  
Rao(NorP, Tree, Correction = "Lande", ...,  
  CheckArguments = TRUE, Ps = NULL, Ns = NULL)

Arguments

Ps  A probability vector, summing to 1.
Ns  A numeric vector containing species abundances.
NorP  A numeric vector, an integer vector, an abundance vector (AbdVector) or a prob-
       ability vector (ProbaVector). Contains either abundances or probabilities.
Tree  An object of class hclust, phylo, phylog or PPTree. The tree must be ultra-
       metric.
Correction  A string containing one of the possible corrections accepted by bcTsallis or  
       "Lande", the default value (equivalent to "Best").
...  Additional arguments. Unused.
CheckArguments  Logical; if TRUE, the function arguments are verified. Should be set to FALSE to  
                 save time when the arguments have been checked elsewhere.

Details

Bias correction requires the number of individuals. Use bcRao and choose the Correction.

The unbiased estimator of Rao’s entropy is identical to that of Simpson’s entropy because Rao’s  
entropy is a linear sum of Simson entropies, all of them calculated from the same number of indi-
viduals (Marcon and Herault, 2014). It equals the plug-in estimator multiplied by n/(n-1) where n is  
the total number of individuals.
The functions are designed to be used as simply as possible. \texttt{Tsallis} is a generic method. If its first argument is an abundance vector, an integer vector or a numeric vector which does not sum to 1, the bias corrected function \texttt{bctsallis} is called. Explicit calls to \texttt{bctsallis} (with bias correction) or to \texttt{Tsallis.ProbaVector} (without correction) are possible to avoid ambiguity. The .integer and .numeric methods accept \texttt{Ps} or \texttt{Ns} arguments instead of \texttt{NorP} for backward compatibility.

**Value**

A named number equal to the calculated entropy. The name is that of the bias correction used.

**Author(s)**

Eric Marcon <Eric.Marcon@ecofog.gf>

**References**


**See Also**

\texttt{bcPhyloDiversity}

**Examples**

```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Ns is the total number of trees per species
Ns <- as.AbdVector(Paracou618.MC$Ns)
# Species probabilities
Ps <- as.ProbaVector(Paracou618.MC$Ns)
# Calculate Rao's quadratic entropy of the plot
Rao(Ps, Paracou618.Taxonomy)
```

---

**rCommunity**

### Random Communities

**Description**

Draws random communities according to a probability distribution.

**Usage**

```r
rCommunity(n, size = sum(NorP), NorP = 1, BootstrapMethod = "Chao2015", S = 300,
Distribution = "lnorm", sd = 1, prob = 0.1, alpha = 40,
CheckArguments = TRUE)
```
Arguments

- **n**: The number of communities to draw.
- **size**: The number of individuals to draw in each community.

**BootstrapMethod**

The method used to obtain the probabilities to generate bootstrapped communities from observed abundances. If "Marcon", the probabilities are simply the abundances divided by the total number of individuals (Marcon et al., 2012). If "Chao2013" or "Chao2015" (by default), a more sophisticated approach is used (see as.ProbaVector) following Chao et al. (2013) or Chao et al. (2015).

**NorP**: A numeric vector or a two-column matrix. Contains either abundances or probabilities. Two-column matrices should contain the observed abundances (or probabilities) in the first column and the expected ones in the second column, to allow using beta diversity functions.

**S**: The number of species.

**Distribution**: The distribution of species frequencies. May be "lnorm" (log-normal), "lseries" (log-series), "geom" (geometric) or "bstick" (broken stick).

**sd**: The simulated distribution standard deviation. For the log-normal distribution, this is the standard deviation on the log scale.

**prob**: The proportion of resources taken by successive species.

**alpha**: Fisher's alpha.

**CheckArguments**: Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

Details

Communities of fixed **size** are drawn in a multinomial distribution according to the distribution of probabilities provided by **NorP**.

An abundance vector may be used instead of probabilities, then **size** is by default the total number of individuals in the vector. Random communities are built by drawing in a multinomial law following Marcon et al. (2012), or trying to estimate the distribution of the actual community with as.ProbaVector. If **BootstrapMethod** = "Chao2013", the distribution is estimated by a single parameter model and unobserved species are given equal probabilities. If **BootstrapMethod** = "Chao2015", a two-parameter model is used and unobserved species follow a geometric distribution.

Alternatively, the probabilities may be drawn following a classical distribution: either a lognormal ("lnorm") one (Preston, 1948) with given standard deviation (sd; note that the mean is actually a normalizing constant. Its values is set equal to 0 for the simulation of the normal distribution of unnormalized log-abundances), a log-series ("lseries") one (Fisher et al., 1943) with parameter alpha, a geometric ("geom") one (Motomura, 1932) with parameter prob, or a broken stick ("bstick") one (MacArthur, 1957). The number of simulated species is fixed by **S**, except for "lseries" where it is obtained from alpha and **size**: \(S = \alpha \ln(1 + \frac{\text{size}}{\alpha})\).

Log-normal, log-series and broken-stick distributions are stochastic. The geometric distribution is completely determined by its parameters.
Value

A vector of species abundances (AbdVector) if a single community has been drawn, or a MetaCommunity containing simulated communities.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


See Also

*SpeciesDistribution* and the program SimAssem (Reese et al., 2013; not an R package) for more distributions.

Examples

```r
# Generate communities made of 100000 individuals among 300 species and fit them
par(mfrow = c(2,2))
for (d in c("lnorm", "lseries", "geom", "bstick")) {
  rCommunity(n = 1, size = 1E5, S = 300, Distribution = d) -> AbdVec
  plot(AbdVec, Distribution = d, main = d)
}
```
Richness: \textit{Number of species of a community}

**Description**

Calculates the number of species from probability vector. The name is that of the estimator (the bias correction) used.

**Usage**

\begin{verbatim}
Richness(NorP, ...)  
bcRichness(Ns, Correction = "Best", Alpha = 0.05, JackOver = FALSE,  
    CheckArguments = TRUE)  
## S3 method for class 'ProbaVector'  
Richness(NorP, ..., CheckArguments = TRUE, Ps = NULL)  
## S3 method for class 'AbdVector'  
Richness(NorP, Correction = "Best", Alpha = 0.05, JackOver = FALSE,  
    Level = NULL, PCorrection = "Chao2015", Unveiling = "geom", RCorrection = "Rarefy",  
    ..., CheckArguments = TRUE, Ns = NULL)  
## S3 method for class 'integer'  
Richness(NorP, Correction = "Best", Alpha = 0.05, JackOver = FALSE,  
    Level = NULL, PCorrection = "Chao2015", Unveiling = "geom", RCorrection = "Rarefy",  
    ..., CheckArguments = TRUE, Ns = NULL)  
## S3 method for class 'numeric'  
Richness(NorP, Correction = "Best", Alpha = 0.05, JackOver = FALSE,  
    Level = NULL, PCorrection = "Chao2015", Unveiling = "geom", RCorrection = "Rarefy",  
    ..., CheckArguments = TRUE, Ps = NULL, Ns = NULL)
\end{verbatim}

**Arguments**

- **Ps**: A probability vector, summing to 1.
- **Ns**: A numeric vector containing species abundances.
- **NorP**: A numeric vector, an integer vector, an abundance vector (AbdVector) or a probability vector (ProbaVector). Contains either abundances or probabilities.
- **Correction**: A string containing one of the possible corrections: "None" (no correction), "Jackknife", "iChao1", or "Chao1". "Best", the default value, is currently "Jackknife". Ignored by richness interpolation, and by extrapolation if PCorrection is not "None".
- **Alpha**: The risk level, 5% by default, used to optimize the jackknife order.
- **JackOver**: If TRUE, retain the jackknife order immediately superior to the optimal one, usually resulting in the overestimation of the number of species. Default is FALSE.
- **Level**: The level of interpolation or extrapolation. It may be an a chosen sample size (an integer) or a sample coverage (a number between 0 and 1). Richness extrapolation require its asymptotic estimation depending on the choice of Correction.
PCorrection: A string containing one of the possible corrections to estimate a probability distribution in `as.ProbaVector`: "Chao2015" is the default value. If "None", the asymptotic distribution is not estimated and extrapolation relies only on the asymptotic estimator of richness. Used only for extrapolation.

Unveiling: A string containing one of the possible unveiling methods to estimate the probabilities of the unobserved species in `as.ProbaVector`: "geom" (the unobserved species distribution is geometric) is the default value. Used only for extrapolation.

RCorrection: A string containing a correction recognized by `Richness` to evaluate the total number of species in `as.ProbaVector`. "Rarefy" is the default value to estimate the number of species such that the entropy of the asymptotic distribution rarefied to the observed sample size equals the observed entropy of the data. Used only for extrapolation.

...: Additional arguments. Unused.

CheckArguments: Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

Details

Bias correction requires the number of individuals. Use `bcRichness` and choose the Correction. Chao correction techniques are from Chao (1984) and Chiu et al. (2015). The Jackknife estimator is calculated by a straight adaptation of the code by Ji-Ping Wang (`jackknife` in package SPECIES). The optimal order is selected according to Burnham and Overton (1978; 1979). The argument `JackOver` allows selecting one order over the optimal one. Many other estimators are available elsewhere, the ones implemented here are necessary for other entropy estimations.

The functions are designed to be used as simply as possible. `Richness` is a generic method. If its first argument is an abundance vector, an integer vector or a numeric vector which does not sum to 1, the bias corrected function `bcRichness` is called.

Richness can be estimated at a specified level of interpolation or extrapolation, either a chosen sample size or sample coverage (Chao et al., 2014), rather than its asymptotic value. Extrapolation relies on the estimation of the asymptotic richness. If `PCorrection` is "None", then the asymptotic estimation of richness is made using the chosen Correction, else the asymptotic distribution of the community is derived and its estimated richness adjusted so that the entropy of a sample of this distribution of the size of the actual sample has the entropy of the actual sample.

Value

A named number equal to the estimated number of species. The name is the Correction, except for "SAC" (Species Accumulation Curve) for interpolation.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>
References


Examples

```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Ns is the total number of trees per species
Ns <- as.AbdVector(Paracou618.MC$Ns)
# Species probabilities
Ps <- as.ProbaVector(Paracou618.MC$Ns)
# Whittaker plot
plot(Ns)

# Number of observed species
Richness(Ps)
# Estimate the actual number of species
bcRichness(Ns, Correction = "Chao1")
bcRichness(Ns, Correction = "iChao1")
bcRichness(Ns, Correction = "Jackknife")
bcRichness(Ns, Correction = "Jackknife", JackOver=TRUE)
```

---

**Shannon**

*Shannon entropy of a community*

**Description**

Calculates the Shannon entropy of a probability vector.

**Usage**

```r
Shannon(NorP, ...)
bcShannon(Ns, Correction = "Best", CheckArguments = TRUE)
## S3 method for class 'ProbaVector'
Shannon(NorP, ..., CheckArguments = TRUE, Ps = NULL)
## S3 method for class 'AbdVector'
```
Shannon(NorP, Correction = "Best", Level = NULL, 
Pcorrection = "Chao2015", Unveiling = "geom", RCorrection = "Rarefy", ..., 
CheckArguments = TRUE, Ns = NULL)
## S3 method for class 'integer'
Shannon(NorP, Correction = "Best", Level = NULL, 
Pcorrection = "Chao2015", Unveiling = "geom", RCorrection = "Rarefy", ..., 
CheckArguments = TRUE, Ns = NULL)
## S3 method for class 'numeric'
Shannon(NorP, Correction = "Best", Level = NULL, 
Pcorrection = "Chao2015", Unveiling = "geom", RCorrection = "Rarefy", ..., 
CheckArguments = TRUE, Ps = NULL, Ns = NULL)

Arguments

Ps
A probability vector, summing to 1.

Ns
A numeric vector containing species abundances.

NorP
A numeric vector, an integer vector, an abundance vector (AbdVector) or a probability vector (ProbaVector). Contains either abundances or probabilities.

Correction
A string containing one of the possible asymptotic estimators: "None" (no correction), "ChaoShen", "GenCov", "Grassberger", "Grassberger2003", "Schurmann", "Holste", "Bonachela", "Miller", "ZhangHz", "ChaoJost", "Marcon", "UnveilC", "UnveilIC", "UnveilI" or "Best", the default value. Currently, "Best" is "UnveilI".

Level
The level of interpolation or extrapolation. It may be an a chosen sample size (an integer) or a sample coverage (a number between 0 and 1). Entropy extrapolation requires its asymptotic estimation depending on the choice of Correction. Entropy interpolation relies on the estimation of Abundance Frequence Counts: then, Correction is passed to AbdFreqCount as its estimator argument.

Pcorrection
A string containing one of the possible corrections to estimate a probability distribution in as.ProbaVector: "Chao2015" is the default value. Used only for extrapolation.

Unveiling
A string containing one of the possible unveiling methods to estimate the probabilities of the unobserved species in as.ProbaVector: "geom" (the unobserved species distribution is geometric) is the default value. If "None", the asymptotic distribution is not unveiled and only the asymptotic estimator is used. Used only for extrapolation.

RCorrection
A string containing a correction recognized by Richness to evaluate the total number of species in as.ProbaVector. "Rarefy" is the default value to estimate the number of species such that the entropy of the asymptotic distribution rarefied to the observed sample size equals the observed entropy of the data. Used only for extrapolation.

... Additional arguments. Unused.

CheckArguments Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.
Details

Bias correction requires the number of individuals to estimate sample Coverage. Correction techniques are from Miller (1955), Chao and Shen (2003), Grassberger (1988), Grassberger (2003), Schurmann (2003), Holste et al. (1998), Bonachela et al. (2008), Zhang (2012), Chao, Wang and Jost (2013). More estimators can be found in the entropy package.

Using MetaCommunity mutual information, Chao, Wang and Jost (2013) calculate reduced-bias Shannon beta entropy (see the last example below) with better results than the Chao and Shen estimator, but community weights cannot be arbitrary: they must be proportional to the number of individuals.

The functions are designed to be used as simply as possible. Shannon is a generic method. If its first argument is an abundance vector, an integer vector or a numeric vector which does not sum to 1, the bias corrected function bcShannon is called.

Entropy can be estimated at a specified level of interpolation or extrapolation, either a chosen sample size or sample coverage (Chao et al., 2014), rather than its asymptotic value. Extrapolation relies on the estimation of the asymptotic entropy. If Unveiling is "None", then the asymptotic estimation of entropy is made using the chosen Correction, else the asymptotic distribution of the community is derived and its estimated richness adjusted so that the entropy of a sample of this distribution of the size of the actual sample has the entropy of the actual sample.

Value

A named number equal to the calculated entropy. The name is that of the bias correction used.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


Shannon Beta


See Also

`bcShannon`, `Tsallis`

Examples

```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Ns is the total number of trees per species
Ns <- as.AbdVector(Paracou618.MC$Ns)
# Species probabilities
Ps <- as.ProbaVector(Paracou618.MC$Ns)
# Whittaker plot
plot(Ns)

# Calculate Shannon entropy
Shannon(Ps)

# Calculate the best estimator of Shannon entropy
Shannon(Ns)

# Use metacommunity data to calculate reduced-bias Shannon beta as mutual information
(bcShannon(Paracou618.MC$Ns) + bcShannon(colSums(Paracou618.MC$Ns)))
- bcShannon(Paracou618.MC$Ns))
```

### ShannonBeta

**Shannon beta entropy of a community**

**Description**

Calculates the Shannon beta entropy of a community belonging to a metacommunity.
Usage

```r
ShannonBeta(NorP, NorPexp = NULL, ...) 
bcShannonBeta(Ns, Nexp, Correction = "Best", CheckArguments = TRUE)
```

Arguments

Ps  The probability vector of species of the community.
Pexp The probability vector of species of the metacommunity.
Ns  A numeric vector containing species abundances of the community.
Nexp A numeric vector containing species abundances of the metacommunity.
NorP A numeric vector, an integer vector, an abundance vector (AbdVector) or a probability vector (ProbaVector). Contains either abundances or probabilities of the community.
NorPexp A numeric vector, an integer vector, an abundance vector (AbdVector) or a probability vector (ProbaVector). Contains either abundances or probabilities of the metacommunity.
Correction A string containing one of the possible corrections: currently, "ChaoShen" (Marcon et al., 2012) equivalent to "Best", and "ZhangGrabchak" (Zhang and Grabchak, 2014).
CheckArguments Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

Details

The derivation of Shannon beta entropy can be found in Marcon et al. (2012).

Bias correction requires the number of individuals to estimate sample Coverage. Use bcShannonBeta and choose the Correction.

The functions are designed to be used as simply as possible. ShannonBeta is a generic method. If its first argument is an abundance vector, an integer vector or a numeric vector which does not sum to 1, the bias corrected function bcShannonBeta is called. Explicit calls to bcShannonBeta (with bias correction) or to ShannonBeta.ProbaVector (without correction) are possible to avoid ambiguity. The .integer and .numeric methods accept Ps or Ns arguments instead of NorP for backward compatibility.
Simpson

Value

A number equal to the calculated entropy.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


See Also

bcShannonBeta

Examples

# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Ps is the vector of probabilities
Ps <- as.ProbaVector(Paracou618.MC$Ps)
# Probability distribution of the first plot
Ps1 <- as.ProbaVector(Paracou618.MC$Psi[, 1])
# Shannon beta entropy of the plot
ShannonBeta(Ps1, Ps)

# Ns is the vector of abundances of the metacommunity
Ns <- as.AbdVector(Paracou618.MC$Ns)
# Abundances in the first plot
Ns1 <- as.AbdVector(Paracou618.MC$Ns[, 1])
# Reduced-bias estimator of Shannon beta entropy of the plot
bcShannonBeta(Ns1, Ns)

---

Simpson

Simpson entropy of a community

Description

Calculates the Simpson entropy of a probability vector.
Usage

Simpson(NorP, ...)  
bcSimpson(Ns, Correction = "Best", CheckArguments = TRUE)  
## S3 method for class 'ProbaVector'  
Simpson(NorP, ..., CheckArguments = TRUE,  
  Ps = NULL)  
## S3 method for class 'AbdVector'  
Simpson(NorP, Correction="Best", Level = NULL, ...,  
  CheckArguments = TRUE, Ns = NULL)  
## S3 method for class 'integer'  
Simpson(NorP, Correction="Best", Level = NULL, ...,  
  CheckArguments = TRUE, Ns = NULL)  
## S3 method for class 'numeric'  
Simpson(NorP, Correction="Best", Level = NULL, ...,  
  CheckArguments = TRUE, Ps = NULL, Ns = NULL)

Arguments

Ps  
A probability vector, summing to 1.

Ns  
A numeric vector containing species abundances.

NorP  
A numeric vector, an integer vector, an abundance vector (AbdVector) or a probability vector (ProbaVector). Contains either abundances or probabilities.

Correction  
A string containing one of the possible corrections accepted by bcTsallis or "Lande". "Best", the default value, is currently "Jackknife". Ignored by interpolation and extrapolation.

Level  
The level of interpolation or extrapolation. It may be an a chosen sample size (an integer) or a sample coverage (a number between 0 and 1).

...  
Additional arguments. Unused.

CheckArguments  
Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

Details

Lande’s correction has been derived (Lande, 1996; Good, 1953) especially for Simpson entropy, while other corrections are for generalized Tsallis entropy. It is identical to the unbiased estimator proposed by Simpson, although arguments were different. It equals the plug-in estimator multiplied by \(\frac{n}{n-1}\) where \(n\) is the total number of individuals.

Bias correction requires the number of individuals to estimate sample Coverage.

The functions are designed to be used as simply as possible. Simpson is a generic method. If its first argument is an abundance vector, an integer vector or a numeric vector which does not sum to 1, the bias corrected function bcSimpson is called.

Entropy can be estimated at a specified level of interpolation or extrapolation, either a chosen sample size or sample coverage (Chao et al., 2014), rather than its asymptotic value. Simpson’s extrapolated entropy estimator does not rely on the estimation of the asymptotic distribution.
Value

A named number equal to the calculated entropy. The name is that of the bias correction used.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


See Also

*Tsallis*, *bcSimpson*

Examples

```r
# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Ns is the total number of trees per species
Ns <- as.AbdVector(Paracou618.MC$Ns)
# Whittaker plot
plot(Ns)

# Calculate an unbiased estimator of Simpson's index of diversity
Simpson(Ns)
```

SimpsonBeta  

*Simpson beta entropy of a community*

Description

Calculates the Simpson beta entropy of a community belonging to a metacommunity.
SimpsonBeta

Usage

SimpsonBeta(NorP, NorPexp = NULL, ...) 
bcSimpsonBeta(Ns, Nexp, Correction = "Best", CheckArguments = TRUE) 
# S3 method for class 'ProbaVector'
SimpsonBeta(NorP, NorPexp = NULL, ...) 
# S3 method for class 'AbdVector'
SimpsonBeta(NorP, NorPexp = NULL, Correction = "Best", ...) 
# S3 method for class 'integer'
SimpsonBeta(NorP, NorPexp = NULL, Correction = "Best", ...) 
# S3 method for class 'numeric'
SimpsonBeta(NorP, NorPexp = NULL, Correction = "Best", ...) 

Arguments

Ps The probability vector of species of the community.
Pexp The probability vector of species of the metacommunity.
Ns A numeric vector containing species abundances of the community.
Nexp A numeric vector containing species abundances of the metacommunity.
NorP A numeric vector, an integer vector, an abundance vector (AbdVector) or a probability vector (ProbaVector). Contains either abundances or probabilities of the community.
NorPexp A numeric vector, an integer vector, an abundance vector (AbdVector) or a probability vector (ProbaVector). Contains either abundances or probabilities of the metacommunity.
Correction A string containing one of the possible corrections: currently, only "ChaoShen", identical to "Best".
... Additional arguments. Unused.
CheckArguments Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

Details

The derivation of Tsallis beta entropy (Simpson is Tsallis of order 2) can be found in Marcon et al. (2014).

Bias correction requires the number of individuals to estimate sample Coverage. Use bcSimpsonBeta and choose the Correction.

Note that Simpson beta entropy value is related to Simpson alpha entropy value and cannot be compared across communities (Jost, 2007). Beta entropy of a community is not meaningful in general, do rather calculate the BetaDiversity of order 2 of the metacommunity.

The functions are designed to be used as simply as possible. SimpsonBeta is a generic method. If its first argument is an abundance vector, an integer vector or a numeric vector which does not
sum to 1, the bias corrected function bcSimpsonBeta is called. Explicit calls to bcSimpsonBeta (with bias correction) or to SimpsonBeta.ProbaVector (without correction) are possible to avoid ambiguity. The .integer and .numeric methods accept Ps or Ns arguments instead of NorP for backward compatibility.

Value

A named number equal to the calculated entropy. The name is that of the bias correction used.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

References


See Also

*Simpson, bcSimpsonBeta, BetaDiversity*

Examples

# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Ps is the vector of probabilities
Ps <- as.ProbaVector(Paracou618.MC$Ps)
# Probability distribution of the first plot
Ps1 <- as.ProbaVector(Paracou618.MC$Ps[, 1])

# Simpson beta entropy of the plot
SimpsonBeta(Ps1, Ps)
# Transform into diversity
expq(SimpsonBeta(Ps1, Ps)/(1-Simpson(Ps1)), 2)

# Ns is the vector of abundances of the metacommunity
Ns <- as.AbdVector(Paracou618.MC$Ns)
# Abundances in the first plot
Ns1 <- as.AbdVector(Paracou618.MC$Ns[, 1])
# Reduced-bias Shannon beta entropy of the plot
bcSimpsonBeta(Ns1, Ns)
**SimTest**

**SimTest class**

**Description**

Methods for objects of type "SimTest", used to test a value against its distribution under a simulated null hypothesis.

**Usage**

```r
as.SimTest(RealValue, SimulatedValues)
is.SimTest(x)
## S3 method for class 'SimTest'
autoplot(object, Quantiles = c(0.025, 0.975), ...
  colValue = "red", colQuantiles = "black", ltyQuantiles = 2,
  main = NULL, xlab = "Simulated Values", ylab = "Density")
## S3 method for class 'SimTest'
plot(x, Quantiles = c(0.025, 0.975), ...
  colValue = "red", lwdValue = 2, ltyValue = 2,
  colQuantiles = "black", lwdQuantiles = 1, ltyQuantiles = 2,
  main = NULL, xlab = "Simulated Values", ylab = "Density")
## S3 method for class 'SimTest'
summary(object, Quantiles = c(0.025, 0.975), ...)
```

**Arguments**

- **x** An object to be tested or plotted.
- **object** An object.
- **RealValue** A numeric Value (the actual one).
- **SimulatedValues** A numeric vector containing the simulated values.
- **Quantiles** A vector containing the quantiles of interest.
- **colValue** The color of the line representing the real value on the plot.
- **lwdValue** The width of the line representing the real value on the plot.
- **ltyValue** The line type of the line representing the real value on the plot.
- **colQuantiles** The color of the lines representing the quantiles on the plot.
- **lwdQuantiles** The width of the lines representing the quantiles on the plot.
- **ltyQuantiles** The line type of the lines representing the quantiles on the plot.
- **main** The main title of the plot. if NULL (by default), there is no title.
- **xlab** The X axis label.
- **ylab** The Y axis label.
- **...** Additional arguments to be passed to the generic methods.
Details

Simulated values should be obtained by simulation. The actual value is compared to simulated quantiles. SimTest objects can be plotted and summarized.

Value

SimTest objects are lists containing:

- `RealValue` The value to test.
- `SimulatedValues` A vector of simulated values, whose quantiles will be used for the test.

isSimTest returns TRUE if the object is of class SimTest.

summarySimTest returns a summary of the object, including the empirical quantile of the real value in the simulated distribution.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>

Examples

```r
# Set the value to test
Real <- 0.8
# Is it a realization of a Gaussian distribution?
Sims <- rnorm(1000)
# Make a Simtest object
st <- asSimTest(Real, Sims)
summary(st)
# Plot
plot(st)
# ggplot
autoplot(st)
```

SpeciesDistribution Species Distributions.

Description

A Species Distribution is a (preferably named) vector containing species abundances or probabilities.
Usage

as.SpeciesDistribution(x, ...)
## S3 method for class 'data.frame'
as.SpeciesDistribution(x, ...)
## S3 method for class 'integer'
as.SpeciesDistribution(x, ...)
## S3 method for class 'numeric'
as.SpeciesDistribution(x, ...)
## S3 method for class 'SpeciesDistribution'
autoplot(object, ..., Distribution = NULL,
ylog = TRUE, main = NULL, xlab = "Rank", ylab = NULL)
## S3 method for class 'SpeciesDistribution'
plot(x, ..., Distribution = NULL,
type = "b", log = "y", main = NULL, xlab = "Rank", ylab = NULL)
is.SpeciesDistribution(x)
as.ProbaVector(x, ...)
## S3 method for class 'data.frame'
as.ProbaVector(x, ...)
## S3 method for class 'integer'
as.ProbaVector(x, Correction = "None", Unveiling = "None",
RCorrection = "Jackknife", JackOver = FALSE, CEstimator = "ZhangHuang",
q = 0, ..., CheckArguments = TRUE)
## S3 method for class 'numeric'
as.ProbaVector(x, Correction = "None", Unveiling = "None",
RCorrection = "Jackknife", JackOver = FALSE, CEstimator = "ZhangHuang",
q = 0, ..., CheckArguments = TRUE)
is.ProbaVector(x)
as.AbdVector(x, ...)
## S3 method for class 'data.frame'
as.AbdVector(x, Round = TRUE, ...)
## S3 method for class 'integer'
as.AbdVector(x, ...)
## S3 method for class 'numeric'
as.AbdVector(x, Round = TRUE, ...)
is.AbdVector(x)

Arguments

x An object.
object An object.
Distribution The distribution to fit on the plot. May be "lnorm" (log-normal), "lseries" (log-series), "geom" (geometric) or "bstick" (broken stick). If NULL, no distribution is fitted. See rCommunity for the description of these distributions.
Round If TRUE (by default), values of x are set to integer to create an AbdVector. This is useful if original abundances are not integers (this is often the case for MetaCommunity abundances which are the product of probabilities by the number of individuals) and integer values are required (for example to calculate the bootstrap confidence interval of a community profile).
Correction A string containing one of the possible corrections to estimate a probability distribution: "None" (no correction, the default value), or "Chao2013", "Chao2015", "ChaoShen" to estimate the probability of the observed species in the asymptotic distribution.

Unveiling A string containing one of the possible unveiling methods to estimate the probabilities of the unobserved species: "None" (default, no species is added), "unif" (uniform: all unobserved species have the same probability) or "geom" (geometric: the unobserved species distribution is geometric).

RCorrection A string containing a correction recognized by richness to evaluate the total number of species. "Jackknife" is the default value. An alternative is "Rarefy" to estimate the number of species such that the entropy of order q of the asymptotic distribution rarefied to the observed sample size equals the actual entropy of the data.

JackOver If TRUE, retain the jackknife order immediately superior to the optimal one, usually resulting in the overestimation of the number of species. Default is FALSE. Ignored if RCorrection is not "Jackknife".

CEstimator A string containing an estimator recognized by Coverage to evaluate the sample coverage. "ZhangHuang" is the default value.

q A number: the order of entropy. Default is 0 for richness. Used only to estimate asymptotic probability distributions with RCorrection equal to "Rarefy". Then, the number of unobserved species is fitted so that the entropy of order q of the asymptotic probability distribution at the observed sample size equals the actual entropy of the data.

type The plot type, see plot.

log The axis to plot in log scale, e.g. "xy" for both axes. Default is "y".

main The main title of the plot. If NULL (by default), there is no title.

xlab The X axis label, "Rank" by default.

ylab The Y axis label. If NULL (by default), "Probability" or "Abundance" is chosen according to the object class.

ylog Logical; if TRUE (by default), the Y-axis of the plot is log scaled.

... Additional arguments to be passed to plot. Unused elsewhere.

CheckArguments Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

Details

SpeciesDistribution objects include AbdVector and ProbaVector objects.

as.AbdVector just sets the class of the numeric or integer x so that appropriate versions of community functions (generic methods such as Diversity) are applied. Abundance values are rounded (by default) to the nearest integer.

as.ProbaVector normalizes the vector so that it sums to 1. If Correction is not "None", the observed abundance distribution is used to estimate the actual species distribution. The list of species will be changed: zero-abundance species will be cleared, and some unobserved species will be added. First, observed species probabilities are estimated following Chao and Shen (2003),
SpeciesDistribution

i.e. input probabilities are multiplied by the sample coverage, or according to more sophisticated models: Chao et al. (2013, single-parameter model), or Chao et al. (2015, two-parameter model). The total probability of observed species equals the sample coverage. Then, the distribution of unobserved species can be unveiled: their number is estimated according to RCorrection (if the Jackknife estimator is chosen, the JackOver argument allows using the order immediately over the optimal one). The coverage deficit (1 minus the sample coverage) is shared by the unobserved species equally (Unveiling = "unif", Chao et al., 2013) or according to a geometric distribution (Unveiling = "geom", Chao et al., 2015).

These functions can be applied to data frames to calculate the joint diversity (Gregorius, 2010). SpeciesDistribution objects can be plotted. The plot method returns the estimated parameters of the fitted distribution. The broken stick has no parameter, so the maximum abundance is returned.

Note

Fisher’s alpha (Fisher et al., 1943) is estimated to fit the log-series distribution. The estimation is done by the fisher.alpha function of package vegan. It may differ substantially from the estimation returned by optimal.theta from package untb.

Author(s)

Eric Marcon <Eric.Marcon@ecofog.gf>, Bruno Herault <Bruno.Herault@ecofog.gf>

References


See Also

rgeom, rlnorm, rCommunity, RA1norm

Examples

# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Ns is the total number of trees per species
Ns <- as.AbdVector(Paracou618.MCSNs)
# Whittaker plot, poorly fitted by a log-normal distribution
Description

Calculates the HCDT, also known as Tsallis entropy of order $q$ of a probability vector.

Usage

```r
Tsallis(NorP, q = 1, ...)  # S3 method for class 'ProbaVector'
bcsallis(Ns, q = 1, Correction = "Best", SampleCoverage = NULL, 
         CheckArguments = TRUE)
Tsallis(NorP, q = 1, ...,
         CheckArguments = TRUE, Ps = NULL)
## S3 method for class 'AbdVector'
Tsallis(NorP, q = 1, Correction = "Best", Level = NULL,
         PCorrection="Chao2015", Unveiling="geom", RCorrection="Rarefy", 
         CheckArguments = TRUE, Ns = NULL)
## S3 method for class 'numeric'
Tsallis(NorP, q = 1, Correction = "Best", Level = NULL,
         PCorrection="Chao2015", Unveiling="geom", RCorrection="Rarefy", 
         CheckArguments = TRUE, Ps = NULL, Ns = NULL)
```

Arguments

- **Ps**: A probability vector, summing to 1.
- **Ns**: A numeric vector containing species abundances.
- **NorP**: A numeric vector, an integer vector, an abundance vector (AbdVector) or a probability vector (ProbaVector). Contains either abundances or probabilities.
- **q**: A number: the order of entropy. Some corrections allow only a positive number. Default is 1 for Shannon entropy.
- **Correction**: A string containing one of the possible asymptotic estimators: "None" (no correction), "ChaoShen", "GenCov", "Grassberger", "Holste", "Bonachela", "ZhangGrabchak", or "ChaoJost", "Marcon", "UnveilC", "UnveiliC", "UnveilJ" or "Best", the default value. Currently, "Best" is "UnveilJ".
- **Level**: The level of interpolation or extrapolation. It may be an a chosen sample size (an integer) or a sample coverage (a number between 0 and 1).
**Details**

Tsallis (Havrda and Charvat, 1967; Daroczy, 1970; Tsallis, 1988) generalized entropy is a generalized measure of diversity (Jost, 2006).

Bias correction requires the number of individuals to estimate sample Coverage.

Correction techniques are from Chao and Shen (2003), Grassberger (1988), Holste et al. (1998), Bonachela et al. (2008), (Marcon et al., 2014), which is actually the max value of "ChaoShen" and "Grassberger", Zhang and Grabchak (2014), Chao and Jost (2015) and Marcon (2015).

The "ChaoJost" (Chao, Wang and Jost, 2013 for $q = 1$; Chao and Jost, 2015) estimator contains an unbiased part concerning observed species, equal to that of Zhang and Grabchak (2014), and a (biased) estimator of the remaining bias based on the estimation of the species-accumulation curve. It is very efficient but very slow if the number of individuals is more than a few hundreds. This estimator was named "ChaoWangJost" in previous versions of the package; its old name is still supported for backward compatibility.

The unveiled estimators rely on Chao et al. (2015), completed by Marcon (2015). The actual probabilities of observed species are estimated and completed by a geometric distribution of the probabilities of unobserved species. The number of unobserved species is estimated by the Chao1 estimator ("UnveilC"), following Chao et al. (2015), or by the iChao1 ("UnveilIC") or the jacknife ("UnveilJ"). The "UnveilJ" correction often has a lower bias but a greater variance (Marcon, 2015). It is a good first choice thanks to the versatility of the jacknife estimator of richness.

The functions are designed to be used as simply as possible. Tsallis is a generic method. If its first argument is an abundance vector, an integer vector or a numeric vector which does not sum to 1, the bias corrected function bcTsallis is called.

The size of a metacommunity (see MetaCommunity) is unknown so it has to be set according to a rule which does not ensure that its abundances are integer values. Then, classical bias-correction methods do not apply. Providing the SampleCoverage argument allows applying the "ChaoShen" and "Grassberger" corrections to estimate quite well the entropy. DivPart and GammaEntropy functions use this tweak.
Entropy can be estimated at a specified level of interpolation or extrapolation, either a chosen sample size or sample coverage (Chao et al., 2014), rather than its asymptotic value. Special cases $q = 0, 1$ or 2 are treated by Richness, Shannon and Simpson functions. For extrapolation of entropy of other values of $q$, the asymptotic distribution of the community must be estimated by `as.ProbaVector`. The default arguments allow joining smoothly the extrapolated entropy and the observed entropy by estimating the number of unobserved species so that the entropy of the observed distribution equals the entropy of the asymptotic distribution rarefied to the actual sample size.

**Value**

A named number equal to the calculated entropy. The name is that of the bias correction used.

**Author(s)**

Eric Marcon <Eric.Marcon@ecofog.gf>

**References**


Examples

# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Ns is the total number of trees per species
Ns <- as.AbdVector(Paracou618.MC$Ns)
# Species probabilities
Ps <- as.ProbaVector(Paracou618.MC$Ns)
# Whittaker plot
plot(Ns)

# Calculate entropy of order 1, i.e. Shannon's entropy
Tsallis(Ps, 1)
# Calculate it with estimation bias correction
Tsallis(Ns, 1)

Description

Calculates the Tsallis beta entropy of order q of a community belonging to a metacommunity.

Usage

TsallisBeta(NorP, NorPexp = NULL, q = 1, ...)
bcTsallisBeta(Ns, Nexp = NULL, q, Correction = "Best", CheckArguments = TRUE)
## S3 method for class 'ProbaVector'
TsallisBeta(NorP, NorPexp = NULL, q = 1, ..., CheckArguments = TRUE, Ps = NULL, Pexp = NULL)
## S3 method for class 'AbdVector'
TsallisBeta(NorP, NorPexp = NULL, q = 1, Correction = "Best", ..., CheckArguments = TRUE, Ns = NULL, Nexp = NULL)
## S3 method for class 'integer'
TsallisBeta(NorP, NorPexp = NULL, q = 1, Correction = "Best", ..., CheckArguments = TRUE, Ns = NULL, Nexp = NULL)
## S3 method for class 'numeric'
TsallisBeta(NorP, NorPexp = NULL, q = 1, Correction = "Best", ..., CheckArguments = TRUE, Ps = NULL, Ns = NULL, Nexp = NULL)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ps</td>
<td>The probability vector of species of the community.</td>
</tr>
<tr>
<td>Pexp</td>
<td>The probability vector of species of the metacommunity.</td>
</tr>
<tr>
<td>Ns</td>
<td>A numeric vector containing species abundances of the community.</td>
</tr>
<tr>
<td>Nexp</td>
<td>A numeric vector containing species abundances of the metacommunity.</td>
</tr>
</tbody>
</table>
TsallisBeta

Norp
A numeric vector, an integer vector, an abundance vector (AbdVector) or a probability vector (ProbaVector). Contains either abundances or probabilities of the community.

NorPexp
A numeric vector, an integer vector, an abundance vector (AbdVector) or a probability vector (ProbaVector). Contains either abundances or probabilities of the metacommunity.

q
A number: the order of entropy. Default is 1 for Shannon entropy.

Correction
A string containing one of the possible corrections: currently, only "ChaoShen" or "None". "Best" is the default value, it is equivalent to "ChaoShen".

... Additional arguments. Unused.

CheckArguments Logical; if TRUE, the function arguments are verified. Should be set to FALSE to save time when the arguments have been checked elsewhere.

Details
The derivation of Tsallis beta entropy can be found in Marcon et al. (2014).

Bias correction requires the number of individuals to estimate sample Coverage. Use bcTsallisBeta and choose the Correction.

Note that beta entropy value is related to alpha entropy (if q is not 1) and cannot be compared across communities (Jost, 2007). Beta entropy of a community is not meaningful in general, do rather calculate the BetaDiversity of the metacommunity.

The functions are designed to be used as simply as possible. TsallisBeta is a generic method. If its first argument is an abundance vector, an integer vector or a numeric vector which does not sum to 1, the bias corrected function bcTsallisBeta is called. Explicit calls to bcTsallisBeta (with bias correction) or to TsallisBeta.ProbaVector (without correction) are possible to avoid ambiguity. The .integer and .numeric methods accept Ps or Ns arguments instead of NorP for backward compatibility.

Value
A named number equal to the calculated entropy. The name is that of the bias correction used.

Author(s)
Eric Marcon <Eric.Marcon@ecofog.gf>

References

Examples

# Load Paracou data (number of trees per species in two 1-ha plot of a tropical forest)
data(Paracou618)
# Ps is the vector of probabilities
Ps <- Paracou618.MC$Ps
# Probability distribution of the first plot
Ps1 <- Paracou618.MC$Psi[, 1]
# Divergence of order 2 between plot 1 and the whole forest
TsallisBeta(Ps1, Ps, 2)

# Ns is the vector of abundances of the metacommunity
Ns <- Paracou618.MC$Ns
# Abundances in the first plot
Ns1 <- Paracou618.MC$Nsi[, 1]
# Divergence of order 2 between plot 1 and the whole forest, with bias correction
bcTsallisBeta(Ns1, Ns, 2)
## Index

### Topic datasets
- EightSpAbundance, 34
- EightSpTree, 34
- Paracou618.dist, 59
- Paracou618.Functional, 60
- Paracou618.MC, 61
- Paracou618.Taxonomy, 61

### Topic package
- entropartMpackage, 3

<table>
<thead>
<tr>
<th>Package</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AbdFreqCount</td>
<td>4, 84</td>
</tr>
<tr>
<td>AbdVector</td>
<td>22, 23, 32, 42, 44, 46, 48, 66, 68, 71, 77, 80, 81, 84, 87, 89, 91, 98, 102</td>
</tr>
<tr>
<td>AlphaDiversity</td>
<td>10</td>
</tr>
<tr>
<td>AlphaEntropy</td>
<td>10, 11, 12, 27, 29, 39</td>
</tr>
<tr>
<td>as.AbdVector</td>
<td>3</td>
</tr>
<tr>
<td>as.AbdVector (SpeciesDistribution), 94</td>
<td></td>
</tr>
<tr>
<td>as.AccumCurve (Accumulation), 6</td>
<td></td>
</tr>
<tr>
<td>as.CommunityProfile (CommunityProfile), 18</td>
<td></td>
</tr>
<tr>
<td>as.ProbaVector</td>
<td>3, 5, 7, 22, 23, 79, 82, 84, 99, 100</td>
</tr>
<tr>
<td>as.ProbaVector (SpeciesDistribution), 94</td>
<td></td>
</tr>
<tr>
<td>as.SimTest (SimTest), 93</td>
<td></td>
</tr>
<tr>
<td>as.SpeciesDistribution (SpeciesDistribution), 94</td>
<td></td>
</tr>
<tr>
<td>autoplot.AccumCurve (Accumulation), 6</td>
<td></td>
</tr>
<tr>
<td>autoplot.CommunityProfile (CommunityProfile), 18</td>
<td></td>
</tr>
<tr>
<td>autoplot.DivEst (DivEst), 24</td>
<td></td>
</tr>
<tr>
<td>autoplot.DivPart (DivPart), 26</td>
<td></td>
</tr>
<tr>
<td>autoplot.DivProfile (DivProfile), 29</td>
<td></td>
</tr>
<tr>
<td>autoplot.MCdiversity (MCdiversity), 53</td>
<td></td>
</tr>
<tr>
<td>autoplot.MCentropy (MCentropy), 55</td>
<td></td>
</tr>
<tr>
<td>autoplot.PhylorValue (PhylorValue), 73</td>
<td></td>
</tr>
<tr>
<td>autoplot.SimTest (SimTest), 93</td>
<td></td>
</tr>
<tr>
<td>autoplot.SpeciesDistribution (SpeciesDistribution), 94</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bcDiversity</td>
<td>(Diversity), 22</td>
</tr>
<tr>
<td>bcDqz</td>
<td>(Dqz), 32</td>
</tr>
<tr>
<td>bcGenSimpson</td>
<td>(GenSimpson), 41</td>
</tr>
<tr>
<td>bcGenSimpsonD</td>
<td>(GenSimpson), 41</td>
</tr>
<tr>
<td>bcHzq</td>
<td>12, 40</td>
</tr>
<tr>
<td>bcHzq (Hzq), 43</td>
<td></td>
</tr>
<tr>
<td>bcHzqBeta</td>
<td>15</td>
</tr>
<tr>
<td>bcHurlbert</td>
<td>(Hurlbert), 48</td>
</tr>
<tr>
<td>bcHurlbertD</td>
<td>(Hurlbert), 48</td>
</tr>
<tr>
<td>bcPhyloBetaEntropy</td>
<td>15, 67</td>
</tr>
<tr>
<td>bcPhyloBetaEntropy (PhyloBetaEntropy), 65</td>
<td></td>
</tr>
<tr>
<td>bcPhyloDiversity</td>
<td>62, 63, 78</td>
</tr>
<tr>
<td>bcPhyloDiversity (PhyloDiversity), 67</td>
<td></td>
</tr>
<tr>
<td>bcPhyloEntropy</td>
<td>12, 36, 40, 41</td>
</tr>
<tr>
<td>bcPhyloEntropy (PhyloEntropy), 70</td>
<td></td>
</tr>
<tr>
<td>bcRao</td>
<td>(Rao), 77</td>
</tr>
<tr>
<td>bcRichness</td>
<td>(Richness), 81</td>
</tr>
<tr>
<td>bcShannon</td>
<td>36, 86</td>
</tr>
<tr>
<td>bcShannon (Shannon), 83</td>
<td></td>
</tr>
<tr>
<td>bcShannonBeta</td>
<td>88</td>
</tr>
<tr>
<td>bcShannonBeta (ShannonBeta), 86</td>
<td></td>
</tr>
<tr>
<td>bcSimpson</td>
<td>36, 90</td>
</tr>
<tr>
<td>bcSimpsonBeta</td>
<td>92</td>
</tr>
<tr>
<td>bcSimpsonBeta (SimpsonBeta), 90</td>
<td></td>
</tr>
<tr>
<td>bcTsallis</td>
<td>12, 13, 36, 40, 41, 77, 89</td>
</tr>
<tr>
<td>bcTsallis (Tsallis), 98</td>
<td></td>
</tr>
<tr>
<td>bcTsallisBeta</td>
<td>13, 15, 16</td>
</tr>
<tr>
<td>bcTsallisBeta (TsallisBeta), 101</td>
<td></td>
</tr>
<tr>
<td>beta</td>
<td>35</td>
</tr>
<tr>
<td>BetaDiversity</td>
<td>13, 15, 16, 47, 91, 92, 102</td>
</tr>
<tr>
<td>BetaEntropy</td>
<td>14, 15, 27, 29</td>
</tr>
</tbody>
</table>

CEnvelope (CommunityProfile), 18
ChaoPD, 6, 9, 10, 16, 19
CommunityProfile, 18, 37, 38, 51
Coverage2Size (Coverage), 20
dist, 58, 59
DivAC (Accumulation), 6
Diversity, 3, 8, 22, 70, 96
DivEst, 4, 24
DivPart, 4, 24–26, 26, 31, 40, 45, 72, 99
DivProfile, 4, 28, 29
Dqz, 3, 32, 33, 44, 45, 58
EightSpAbundance, 34
EightSpTree, 34
Enq, 35
EntAC (Accumulation), 6
tenropart (tenropart-package), 3
tenropart-package, 3
EntropyCI, 36
exp2, 23, 37, 38, 52
fisher.alpha, 76, 97
GammaDiversity, 38
GammaEntropy, 27, 29, 39, 40, 45, 72, 99
GenSimp.z, 43
GenSimpson, 3, 41
GenSimpsonD (GenSimpson), 41
ggarrange, 25, 30
hclust, 9, 10, 12, 13, 15, 16, 25, 27, 29, 39, 40, 60, 62, 64, 66, 68, 71, 74, 77
Hzq, 33, 43
HzqBeta, 46
Hurlbert, 3, 48
HurlbertD (Hurlbert), 48
is.AbdVector (SpeciesDistribution), 94
is.AccumCurve (Accumulation), 6
is.CommunityProfile (CommunityProfile), 18
is.DivEst (DivEst), 24
is.DivPart (DivPart), 26
is.DivProfile (DivProfile), 29
is.MCdiversity (MCdiversity), 53
is.MCentropy (MCentropy), 55
is.MetaCommunity (MetaCommunity), 56
is.Phylodiversity (Phylodiversity), 67
is.Phylentropy (Phylentropy), 70
is.PhyloValue (PhyloValue), 73
is.PPtree (PPtree), 74
is.ProbaVector (SpeciesDistribution), 94
is.SimTest (SimTest), 93
is.SpeciesDistribution
  (SpeciesDistribution), 94
jackknife, 82
KLq, 50
list, 53
lnq, 51
MC Utilities, 52
MCdiversity, 11, 14, 53
MCentropy, 12, 15, 55
MergeC (MC Utilities), 52
MergeMC (MC Utilities), 52
MetaCommunity, 4, 10, 12, 13, 15, 24, 26, 27, 29, 30, 39, 40, 45, 53–55, 56, 61, 71, 80, 85, 95, 99
Optimal.Similarity, 58
Paracou618.dist, 59, 60
Paracou618.Functional, 60
Paracou618.MC, 61
Paracou618.Taxonomy, 61
PDFD, 62
phylo, 9, 10, 12, 13, 15, 16, 25, 27, 29, 39, 40, 62, 64, 66, 68, 71, 74, 75, 77
PhyloApply, 63, 66, 71, 73
PhylBetaEntropy, 65
Phylodiversity, 3, 17, 33, 64, 66, 67, 67, 72, 73
Phylentropy, 3, 6, 9, 10, 45, 64, 66, 70, 70, 73
phylog, 9, 10, 12, 13, 15, 16, 25, 27, 29, 34, 39, 40, 62, 64, 66, 68, 71, 75, 77
PhyloValue, 40, 73
plot, 7, 73, 96
plot.AccumCurve (Accumulation), 6
plot.CommunityProfile
  (CommunityProfile), 18
plot.DivEst (DivEst), 24
plot.DivPart (DivPart), 26
plot.DivProfile (DivProfile), 29
plot.MCdiversity (MCdiversity), 53
plot.MCentropy (MCentropy), 55
plot.MCdiversity (MCdiversity), 53
plot.MetaCommunity (MetaCommunity), 56
plot.PhyloValue, 69, 72
plot.PhyloValue (PhyloValue), 73
plot.PPtree (PPtree), 74
plot.SimTest (SimTest), 93
plot.SpeciesDistribution, 76
plot.SpeciesDistribution (SpeciesDistribution), 94
PPtree, 9, 10, 12, 13, 15, 16, 25, 27, 29, 39,
40, 62, 64, 66, 68, 71, 74, 77
Preprocess.Tree, 64, 65
ProbaVector, 22, 23, 32, 42, 44, 46, 48, 66,
68, 71, 77, 81, 84, 87, 89, 91, 98, 102
ProbaVector (SpeciesDistribution), 94
RAC, 75
RACbstick (RAC), 75
RACgeom (RAC), 75
RAClnorm, 97
RAClnorm (RAC), 75
RACLseries (RAC), 75
Rao, 3, 77
rCommunity, 3, 18, 19, 36, 76, 78, 95, 97
rgeom, 76, 97
Richness, 5, 7, 23, 81, 82, 84, 96, 99, 100
rlnorm, 76, 97
Shannon, 3, 83, 100
ShannonBeta, 86
ShuffleMC (MC Utilities), 52
Simpson, 3, 88, 92, 100
SimpsonBeta, 90
SimTest, 93
SpeciesDistribution, 80, 94
summary.DivEst (DivEst), 24
summary.DivPart (DivPart), 26
summary.DivProfile (DivProfile), 29
summary.MCdiversity (MCdiversity), 53
summary.MCentropy (MCentropy), 55
summary.MCentropy (MetaCommunity), 56
summary.PhyloDiversity (PhyloDiversity), 67
summary.PhyloEntropy (PhyloEntropy), 70
summary.PhyloValue (PhyloValue), 73
summary.SimTest (SimTest), 93
Tsallis, 3, 7, 8, 12, 19, 23, 71, 72, 86, 90, 98
TsallisBeta, 51, 66, 67, 101