Package ‘pez’

January 15, 2016

Type Package

Title Phylogenetics for the Environmental Sciences

Version 1.1-1

Date 2016-01-15

Author William D. Pearse, Marc W. Cadotte, Jeannine Cavender-Bares, Anthony R. Ives, Caroline Tucker, Steve C. Walker, Matthew R. Helmus

Maintainer William D. Pearse <will.pearse@gmail.com>

Description Eco-phylogenetic and community phylogenetic analyses. Keeps community ecological and phylogenetic data matched up and comparable using 'comparative.comm' objects. Wrappers for common community phylogenetic indices ('pez.shape', 'pez.evenness', 'pez.dispersion', and 'pez.dissimilarity' metrics). Implementation of Cavender-Bares (2004) correlation of phylogenetic and ecological matrices ('fingerprint.regression'). Phylogenetic Generalised Linear Mixed Models (PGLMMs; 'pglmm') following Ives & Helmus (2011) and Rafferty & Ives (2013). Simulation of null assemblages, traits, and phylogenies ('scape', 'sim.meta.comm').

License GPL-3

VignetteBuilder knitr

Suggests knitr (>= 1.6), lme4 (>= 1.1-7)

Imports caper (>= 0.5-2), picante (>= 1.6-2), quantreg (>= 5.05), mvtnorm (>= 1.0-0), vegan (>= 2.0-10), ade4 (>= 1.6-2), apTreeshape (>= 1.4-5), FD (>= 1.0-12), Matrix (>= 1.1-4), methods (>= 3.1.0)

Depends ape (>= 3.1-4)

NeedsCompilation no

Repository CRAN

Date/Publication 2016-01-15 17:59:25

R topics documented:

cc.manip . . . 2
comparative.comm . . . 5
ConDivSim . . . 6
dist.xxx . . . 8
drop_tip . . . 9
eco.scape . . . 10
eco.xxx.regression . . . 13
eco.xxx.regression.list . . . 15
fingerprint.regression . . . 16
generic.metrics . . . 18
laja . . . 20
pez . . . 21
pez.dispersion . . . 21
pez.dissimilarity . . . 23
pez.evenness . . . 25
pez.metrics . . . 28
pez.shape . . . 32
pglmm . . . 34
phy.build . . . 44
phy.signal . . . 45
plot.comparative.comm . . . 47
scape . . . 48
sim.meta . . . 51
sim.phy . . . 52
trait.asm . . . 54
traitgram.cc . . . 55

Index **57**

`cc.manip` *Manipulating and examining comparative.comm objects*

Description

As described in the vignette, we recommend using these wrappers to manipulate species and site data, as it guarantees that everything will be kept consistent across all parts of the `comparative.comm` object. With them, you can drop species, sites, and work directly with each part of your data. You can also manipulate your `comparative.comm` object's `phy`, `data`, `env`, and `comm` slots directly if you wish, but altering the object directly yourself runs the risk of things getting unsynchronised.

Manipulate internals of `comparative.comm` object

Usage

```
## S3 method for class 'comparative.comm'
x[sites, spp, warn = FALSE]

trait.names(object)

env.names(object)

species(x)

species(x) <- value

sites(x)

sites(x) <- value

traits(x) <- value

traits(x)

env(x) <- value

env(x)

comm(x) <- value

comm(x)

tree(x)

phy(x)

tree(x) <- value

phy(x) <- value

assemblage.phylogenies(data)

## S3 method for class 'comparative.comm'
as.data.frame(x, row.names = NULL,
  optional = FALSE, abundance.weighted = FALSE, ...)

## S3 method for class 'comparative.comm'
within(data, expr, ...)
```

Arguments

x	comparative.comm object

sites	numbers of sites to be kept or dropped from x; must be given as numbers. For example, x[1:5,], or x[-1:-5,], but not x[c("site a", "site b"),].
spp	numbers of species to be kept or dropped from x; must be given as numbers. For example, x[,1:5], or x[,-1:-5], but not x[c("sp a", "sp b"),].
warn	whether to warn if species/sites are dropped when creating object (default: TRUE)
object	A comparative.comm object
value	when altering a comparative.comm object's internal structure, the thing that you're inserting into it!
data	A comparative.comm object
row.names	ignored
optional	ignored presence-absence dataset (default: FALSE)
abundance.weighted	whether to create to create a
...	ignored
expr	expression to be evaluated within the scope of data

Value

Names of the traits or environmental variables

Note

As described in comparative.comm, each comparative.comm object contains a phylogeny ($phy) and a site-by-species community matrix (as used in vegan). Optionally, it may contain a data.frame of trait data (each row a species, each column a trait) *called data* for compatibility with comparative.data.

See Also

comparative.comm plot.comaparative.comm

Examples

```
data(laja)
data <- comparative.comm(invert.tree, river.sites, invert.traits, river.env)
#Subset on species, then sites
data <- data[1:5,]
data <- data[,1:5]
#Site and species can be manipulated
species(data)
sites(data)[1:3] <- c("lovely", "invert", "sites")
#Other data can be viewed
trait.names(data)
env.names(data)
#Get assemblage phylogenies of all sites
assemblage.phylogenies(data)
#Add some trait/env data in
traits(data)$new.trait <- sample(letters, nrow(comm(data)), replace=TRUE)
env(data)$new.env <- sample(letters, ncol(comm(data)), replace=TRUE)
```

```
#Manipulate/check phylogeny and community matrix
phy(data) #...tree(data) works too...
comm(data)[1,3] <- 3
comm(data) <- comm(data)[-3,]
```

---

comparative.comm *Creates a community comparative ecology object, the basis of all functions in pez*

Description

Basic checking of whether the input data match up is performed; you need only supply comm and phy, nothing else is mandatory. You can manipulate the internals of comparative.comm, or use the wrappers inside pez to keep everything in order. Examples of these features are given below; they are described in detailed at cc.manip.

Usage

```
comparative.comm(phy, comm, traits = NULL, env = NULL, warn = TRUE,
  force.root = -1)

## S3 method for class 'comparative.comm'
print(x, ...)
```

Arguments

phy	phylogeny (in phylo format) of species
comm	community matrix (as used in vegan) with species as columns and rows as communities. Must contain rownames and colnames; NAs are not checked for but probably unwise.
traits	data.frame of species traits, with rownames matching comm. Saved in the data slot of the resulting comparative.comm object for compatibility with comparative.data.
env	data.frame of environmental data with rownames matching comm
warn	whether to warn if species/sites are dropped when creating object (default: TRUE)
force.root	if phy is unrooted, a root.edge of force.root will be added (default: -1, which means this will never happen). Rarely needed, rarely advisable.
x	comparative.comm object to be printed
...	ignored

Value

comparative.comm object

Note

comparative.comm is compatible with comparative.data; this means that the slot for species' trait data is called data. I appreciate this is somewhat unwieldy, but hopefully you agree it is helpful in the long-term.

Author(s)

Will Pearse

See Also

plot.comparative.comm cc.manip link[caper:comparative.data]{comparative.data}

Examples

```
data(laja)
data <- comparative.comm(invert.tree, river.sites, invert.traits, river.env)
#Subset on species, then sites
data <- data[1:5,]
data <- data[,1:5]
#Site and species can be manipulated
species(data)
sites(data)[1:3] <- c("lovely", "invert", "sites")
#Other data can be viewed
trait.names(data)
env.names(data)
#Get assemblage phylogenies of all sites
assemblage.phylogenies(data)
#Do some manual manipulation of your objects (NOTE: $data for traits)
data$data$new.trait <- sample(letters, nrow(data$comm), replace=TRUE)
```

ConDivSim *Null models for functional-phylogenetic diversity*

Description

Simulate expectations (under a null model) of mean pairwise distance for a set of communities with different species richness.

Usage

```
ConDivSim(object, type = "traits", n.sim = 100, plot = TRUE,
  disp99 = FALSE)
```

Arguments

`object`	a `comparative.comm` object, with presence-absence community data.
`type`	character string giving the type of distance matrix on which the mean pairwise distance is based. Either "trait" or "phy" to a phylogenetic or trait-based distance matrix, or an actual matrix to use (e.g., one from `funct.phylo.dist`)
`n.sim`	The number of permutations of the presence vector used to make the estimations.
`plot`	TRUE or FALSE to make the plot of the expected average mean pairwise distance, and the 5-95% confidence interval.
`disp99`	Display the 99% interval?

Details

If `plot == TRUE`, then a surface is drawn giving the null distribution. Lighter shades of gray give larger intervals with categories: 0.005-0.995 = 99%, 0.025-0.975 = 95%, 0.05-0.95 = 90%, 0.25-0.75 = 50%.

Value

`matrix` with quantiles of mean pairwise distances for all quantiles of of mean pairwise distance, with one row for the range of species richnesses in the data (see column SpRich).

Note

No serious checking of user-provided matrices is performed; this is both useful and dangerous!

Author(s)

Steve Walker, wrappers by Will Pearse

See Also

`sim.phy` `scape`

Examples

```
data(laja)
data <- comparative.comm(invert.tree, river.sites, invert.traits)
#Must have all species present in at least one community!
#...and must be presence-absence data
data <- data[,colSums(data$comm) > 0]
data$comm[data$comm>1] <- 1
sims <- ConDivSim(data)
#...without traits...
sims.phy <- ConDivSim(data, type="phy")
```

dist.xxx *Make co-existence matrices based on phylogeny (and/or) traits, and community or environemntal overlap*

Description

Make co-existence matrices based on phylogeny (and/or) traits, and community or environemntal overlap

Usage

```
comm.dist(x)

## S3 method for class 'matrix'
comm.dist(x)

## S3 method for class 'comparative.comm'
comm.dist(x)

traits.dist(x, dist.func = dist.func.default, ...)

## S3 method for class 'comparative.comm'
traits.dist(x, dist.func = dist.func.default,
  alltogether = TRUE, ...)

## Default S3 method:
traits.dist(x, dist.func = dist.func.default, ...)

## S3 method for class 'data.frame'
traits.dist(x, dist.func = dist.func.default, ...)

dist.func.default(x)

phylo.dist(x, ...)

## S3 method for class 'phylo'
phylo.dist(x, ...)

## S3 method for class 'comparative.comm'
phylo.dist(x, ...)

funct.phylo.dist(x, phyloWeight, p = 2, ...)

pianka.dist(x, ...)

## S3 method for class 'matrix'
pianka.dist(x, env = NULL, ...)
```

```
## S3 method for class 'comparative.comm'
pianka.dist(x, alltogether = TRUE, ...)
```

Arguments

x	an object
dist.func	a function for computing distances. The default, dist.func.default, returns a Euclidean distance of the scaled and centred data.
...	not used
alltogether	should one multivariate distance matrix be computed for all traits at once (DEFAULT; alltogether = TRUE) or for each trait at a time (alltogether = FALSE)?
phyloWeight	phylogenetic weighting parameter (referred to as a in Cadotte et al. (2013)
p	exponent giving the exponent to use for combining functional and phylogenetic distances (the default, p = 2, gives a Euclidean combination).
env	environmental variable to be used to calculate the distance matrix

Details

comm.dist returns the 1 - co-existence of species. Look at how this is calcualted; it incorporates abundances, and if you don't want it to do so simply call it on a presence/absensence (1/0) matrix.

traits.dist returns the functional trait distance of species

phylo.dist returns the phylogenetic (cophenetic) distance of species

funct.phylo.dist returns the combined phylogenetic and trait distances of species, based on the traitgram approach of Cadotte et al. (2013).

pianka.dist returns the environemntal tolerances distance matrices of species. Based on Pianka's distance (i.e., niche overlap based on environmental variables at co-occuring sites), as defined in Cavender-Bares et al. (2004) - likely not the original reference!

References

Cadotte M.A., Albert C.H., & Walker S.C. The ecology of differences: assessing community assembly with trait and evolutionary distances. Ecology Letters 16(10): 1234–1244.

Cavender-Bares J., Ackerly D.D., Baum D.A. & Bazzaz F.A. (2004) Phylogenetic overdispersion in Floridian oak communities. The Americant Naturalist 163(6): 823–843.

---

drop_tip *Trim a phylogeny*

Description

This is a weak wrapper around ape's drop.tip. Importantly, if asked to drop no species from a phylogeny, it will just return the phylogeny (not an empty phylogeny, as drop.tip) will.

Usage

```
drop_tip(tree, spp)
```

Arguments

tree	An phylo object
spp	A vector of species (one, many, or none) to be removed from tree

Value

phylo object

See Also

drop.tip extract.clade

---

eco.scape — *eco.space scape simulation with a macro-ecological focus*

Description

eco.scape is a modified version of the Helmus et al. method implemented in scape. It produces phylogenetically structured communities. It allows phylogenetic signals in niche optima, but unlike scape, does not include the ability to specify niche optima signal type (attraction/repulsion) or phylogenetic signal in range size. Instead, the focus is on having more control over the macroecological characteristics of the resulting landscapes. In particular, eco.scape produces landscapes with fixed mean range sizes, reasonable range size and abundance distributions, and control over whether species present on a tree must be present in the landscape.

Usage

```
eco.scape(tree, scape.size = 10, g.center = 1, wd.all = 0.2 * (scape.size
  + 1)^2, signal.center = TRUE, center.scale = 1, site.stoch.scale = 0,
  sd.center = 1, sd.range = 1, K = 100, extinction = FALSE,
  rho = NULL)
```

Arguments

tree	phylo object; must have branch lengths and be ultrametric
scape.size	edge dimension of square landscape
g.center	strength of phylogenetic signal in species range centers. See corBlomberg, 1=brownian,<1=rates of evol accelerate, >1=rates decelerate.
wd.all	niche width, larger values simulate broader range sizes
signal.center	simulate with phylosignal in range centers
center.scale	adjust strength of phylogenetic attraction in range centers independent of g.center

`site.stoch.scale`
: adjust strength of random variation in species richness across sites

`sd.center`
: sd in `rnorm`rnorm for the range centers, increase to get more variation in center values across species

`sd.range`
: sd in rnorm for the range sizes, increase to get more variation in range sizes across gradients

`K`
: carrying capacity of a site in terms of maximum individuals that can be present. Currently a constant value. Used to scale the presence-absence matrix to include abundances.

`extinction`
: TRUE/FALSE can species on the tree go extinct on the landscape? If the number of species present on the landscape should equal the number of tips on the tree, choose FALSE. See Details.

`rho`
: Grafen branch adjustment of phylogenetic tree see `corGrafen`

Details

Simulates a landscape with species (i.e., tree tips) distributions dependent on a supplied phylogenetic tree. The amount and type of structure is determened by the signal parameter `g.center`. Parameters are based on an Ornstein-Uhlenbeck model of evolution with stabilizing selection. Values of g=1 indicate no stabilizing selection and correspond to the Brownian motion model of evolution; 01 corresponds to disruptive selection where phylogenetic signal for the supplied tree is amplified. See corBlomberg. Communities are simulated along two gradients where the positions along those gradients, `g.center`, can exhibit phylogenetic signal.

The function returns a landscape where the average range size is equivalent to the wd.all parameter - in the scape function, this parameter is not necessarily returned in the resulting landscape. To do this, the probability of presence (th) that returns the wd.all parameter is solved for. If there is no solution that can produce the wd.all given, the error "Error in uniroot(f, lower = 0, upper = max(X.), tol = 10^-200): f() values at end points not of opposite sign" will occur. This seems to mostly arise for extreme or unlikely parameter values (small species pools, low carrying capacities). Try adjusting parameter values first.

The `extinction` parameter specifies whether all of the species on the tree should be present in the final landscape. Some species will have probabilities of presence less than those required for presence. If extinctions is `TRUE`, these species will not be present. If `FALSE`, these species will be present in 1 site, that in which they have the highest probability of presence.

Value

`cc`
: `comparative.comm` object with presence/absence results of simulations. The site names are the row.columns of the cells in the original grid cells that made up the data, and these co-ordinates are also given in the `env` slot of the object along with the environmental gradient information.

`Y`
: presence/absence matrix

`Yab`
: abundance matrix

`index`
: spatial coordinates for X and Y (stacked columns)

`X.joint`
: full probabilities of species at sites, used to construct Y

`X1`
: probabilities of species along gradient 1

X2	probabilities of species along gradient 2
gradient1, gradient2	environmental gradient values
nichewd	average niche width of the assemblage
K	carrying capacity of each cell
environ	matrix depicting environmental values over the 2D landscape
sppXs	full probabilities of each species as an array arranged in a scape.size X scape.size matr ix
V.phylo	initial phylogenetic covariance matrix from tree, output of vcv.phylo(tree, corr=T)
V.phylo.rho	phylogenetic covariance matrix from tree scaled by Grafen if rho is provided, other wise just an output of vcv.phylo(tree, corr=T)
V.center	scaled (by g.center) phylo covariance matrix used in the simulations
bspp1	species optima for gradient 1
bspp2	pecies optima for gradient 2

Author(s)

Matt Helmus, Caroline Tucker, cosmetic edits by Will Pearse

See Also

scape sim.phy sim.meta

Examples

```
# Simulations
tree <- rcoal(64)

scape1 <- eco.scape(tree, scape.size=25, g.center=1,
    signal.center=FALSE, K=100, extinction=TRUE)
scape2 <- eco.scape(tree, scape.size=16, g.center=0.2,
    signal.center=TRUE, K=100, extinction=FALSE)
scape3 <- eco.scape(tree, scape.size=16, g.center=20,
    signal.center=TRUE, K=100, extinction=TRUE)

# Plotting distributions and landscape patterns
original_landscape <- scape1
abundmax <- original_landscape$K
PA_mat <- as.matrix(original_landscape$Y)
abund_mat <- original_landscape$Yab
site.size <- nrow(PA_mat)
species <- ncol(PA_mat)
mx <- original_landscape$gradient
env <- original_landscape$environ$env.gradient
par(mfrow=c(2,2), oma=c(0,0,2,0))
heatcol <- (colorRampPalette(c("yellow","red")))

image(matrix(env,sqrt(site.size),sqrt(site.size),byrow=TRUE),
```

```
    col=heatcol(max(env)),xaxt="n",yaxt="n",main="Env gradient")

image(matrix(rowSums(PA_mat),sqrt(site.size),sqrt(site.size),byrow=TRUE),
    col=heatcol(16),xaxt="n",yaxt="n",main="Species Richness")

hist(colSums(PA_mat),ylab="Number of species",xlab="Number of sites",
    main="Species Area Relationship",col="lightgrey")

hist(colSums(abund_mat),ylab="Number of species",xlab="Number of individuals",
    main="Species Abundance Relationship",col="lightgrey")
mtext("Env random, phy.signal=0.2, 32 species", outer=TRUE, side=3, cex=1.25)
```

---

eco.xxx.regression *eco.xxx.regression*

Description

Regression species co-existence against environmental tolerance, trait similarity, or phylogenetic relatedness.

Usage

```
eco.env.regression(data, randomisation = c("taxa.labels", "richness",
  "frequency", "sample.pool", "phylogeny.pool", "independentswap", "trialswap"),
  permute = 0, method = c("quantile", "lm", "mantel"), altogether = TRUE,
  indep.swap = 1000, abundance = TRUE, ...)

eco.phy.regression(data, randomisation = c("taxa.labels", "richness",
  "frequency", "sample.pool", "phylogeny.pool", "independentswap", "trialswap"),
  permute = 0, method = c("quantile", "lm", "mantel"), indep.swap = 1000,
  abundance = TRUE, ...)

eco.trait.regression(data, randomisation = c("taxa.labels", "richness",
  "frequency", "sample.pool", "phylogeny.pool", "independentswap", "trialswap"),
  permute = 0, method = c("quantile", "lm", "mantel"), altogether = TRUE,
  indep.swap = 1000, abundance = TRUE, ...)

## S3 method for class 'eco.xxx.regression'
summary(object, ...)

## S3 method for class 'eco.xxx.regression'
print(x, ...)

## S3 method for class 'eco.xxx.regression'
plot(x, ...)
```

Arguments

data	comparative.comm for analysis
randomisation	null distribution with which to compare your community data, one of: taxa.labels (DEFAULT), richness, frequency, sample.pool, phylogeny.pool, independentswap, trialswap (as implemented in picante)
permute	the number of null permutations to perform (DEFAULT 0)
method	how to compare distance matrices (only the lower triangle;), one of: lm (linear regression), quantile (DEFAULT; quantreg::rq), mantel (mantel)
altogether	use distance matrix based on all traits (default TRUE), or perform separate regressions for each trait (returns a list, see details)
indep.swap	number of independent swap iterations to perform (if specified in randomisation; DEFAULT 1000)
abundance	whether to incorporate species' abundances (default: TRUE)
...	additional parameters to pass on to model fitting functions
object	eco.xxx.regression object
x	eco.xxx.regression object

Details

These methods are similar to those performed in Cavender-Bares et al. (2004). Each function regresses the species co-existence matrix of data (calculated using comm.dist) against either species' trait dissimilarity (eco.trait.regression), species' phylogenetic distance (eco.phy.regression), or species' shared environmental tolerances as measured by Pianka's distance (eco.env.regression).

If altogether is set to FALSE, each trait or environemntal variables in your data will have a separate eco.trait.regression or eco.env.regression applied to it. The functions will return a list of individual regressions; you can either examine/plot them as a group (see examples below), or extract an individual regression and work with that. These lists are of class eco.xxx.regression.list; a bit messy, but it does work!...

Note

Like fingerprint.regression, this is a data-hungry method. Warnings will be generated if any of the methods cannot be fitted properly (the examples below give toy examples of this). In such cases the summary and plot methods of these functions may generate errors; perhaps use traceback to examine where these are coming from, and consider whether you want to be working with the data generating these errors. I am loathe to hide these errors or gloss over them, because they represent the reality of your data!

WDP loves quantile regressions, and advises that you check different quantiles using the tau options.

Author(s)

Will Pearse, Jeannine Cavender-Bares

References

Cavender-Bares J., Ackerly D.D., Baum D.A. & Bazzaz F.A. (2004) Phylogenetic overdispersion in Floridian oak communities. The Americant Naturalist 163(6): 823–843.

Kembel, S.W., Cowan, P.D., Helmus, M.R., Cornwell, W.K., Morlon, H., Ackerly, D.D., Blomberg, S.P. & Webb, C.O. Picante: R tools for integrating phylogenies and ecology. Bioinformatics 26(11): 1463–1464.

Pagel M. Inferring the historical patterns of biological evolution. Nature 401(6756): 877–884.

See Also

`fingerprint.regression` `phy.signal`

Examples

```
data(laja)
#We wouldn't recommend only using ten permutations - this is just for speed!
data <- comparative.comm(invert.tree, river.sites, invert.traits, river.env)
eco.trait.regression(data, permute=10)
#Specify additional options
eco.trait.regression(data, tau=c(0.25,0.5,0.75), permute=10)
plot(eco.trait.regression(data, permute=10, method="lm"))
plot(eco.trait.regression(data, permute=10, method="lm", altogether=FALSE))
```

---

eco.xxx.regression.list

List of eco.xxx.regressions

Description

List of eco.xxx.regressions

Usage

```
## S3 method for class 'eco.xxx.regression.list'
summary(object, ...)

## S3 method for class 'eco.xxx.regression.list'
print(x, ...)

## S3 method for class 'eco.xxx.regression.list'
plot(x, ...)
```

Arguments

`object`	`eco.xxx.regression.list` object
`...`	additional arguments to plotting functions
`x`	`eco.xxx.regression.list` object

fingerprint.regression

Regress trait evolution against trait ecology (following Cavender-Bares et al. 2004)

Description

Calculates traits' phylogenetic inertia and regresses this against trait similarity among co-existing species (sensu Cavender-Bares et al. 2004 Figure 6)

Usage

```
fingerprint.regression(data, eco.rnd = c("taxa.labels", "richness",
  "frequency", "sample.pool", "phylogeny.pool", "independentswap", "trialswap"),
  eco.method = c("quantile", "lm", "mantel"), eco.permute = 1000,
  evo.method = c("lambda", "delta", "kappa", "blom.k"), eco.swap = 1000,
  abundance = TRUE, ...)

## S3 method for class 'fingerprint.regression'
print(x, ...)

## S3 method for class 'fingerprint.regression'
summary(object, ...)

## S3 method for class 'fingerprint.regression'
plot(x, eco = c("slope", "corrected"),
  xlab = "Community Trait Similarity", ylab = "Phylogenetic inertia", ...)
```

Arguments

data	comparative.comm for analysis
eco.rnd	null distribution with which to compare your community data, one of: taxa.labels (DEFAULT), richness, frequency, sample.pool, phylogeny.pool, independentswap, trialswap (as implemented in picante)
eco.method	how to compare distance matrices (only the lower triangle;), one of: lm (linear regression), quantile (DEFAULT; rq), mantel (mantel)
eco.permute	number of permutations for ecological null model (eco.rnd); default 1000
evo.method	how to measure phylogenetic inertia, one of: lambda (default), delta, kappa, blom.k; see phy.signal.
eco.swap	number of independent swap iterations to perform (if specified in eco.rnd; DEFAULT 1000)
abundance	whether to incorporate species' abundances (default: TRUE)
...	additional parameters to pass on to model fitting functions and plotting functions
x	fingerprint.regression object

`object`	`fingerprint.regression` object
`eco`	plot the observed slopes (DEFAULT: `slope`), or the median difference between the simulations and the observed values (`corrected`)
`xlab`	label for x-axis (default "Ecological Trait Coexistence")
`ylab`	label for y-axis (default "Phylogenetic inertia")

Details

While the term 'fingerprint regression' is new to pez, the method is very similar to that employed in Cavender-Bares et al. 2004 Figure 6. For each trait, the phylogenetic inertia of species traits is regressed against their co-occurrence in the community matrix. Note that Pagel's

$$\lambda$$

,

$$\delta$$

, and

$$\kappa$$

, and Blomberg's K, can be used, unlike the original where a mantel test was employed. Moreover, note also that Pianka's distance (as described in the manuscript) is used to measure species overlap.

Note

Like `eco.xxx.regression`, this is a data-hungry method. Warnings will be generated if any of the methods cannot be fitted properly (the examples below give toy examples of this). In such cases the summary and plot methods of these functions may generate errors; perhaps using `traceback` to examine where these are coming from, and consider whether you want to be working with the data generating these errors. I am loathe to hide these errors or gloss over them, because they represent the reality of your data!

WDP loves quantile regressions, and advises that you check different quantiles using the `tau` options.

Author(s)

Will Pearse and Jeannine Cavender-Bares

References

Cavender-Bares J., Ackerly D.D., Baum D.A. & Bazzaz F.A. (2004) Phylogenetic overdispersion in Floridian oak communities. The Americant Naturalist 163(6): 823–843.

Kembel, S.W., Cowan, P.D., Helmus, M.R., Cornwell, W.K., Morlon, H., Ackerly, D.D., Blomberg, S.P. & Webb, C.O. Picante: R tools for integrating phylogenies and ecology. Bioinformatics 26(11): 1463–1464.

Pagel M. Inferring the historical patterns of biological evolution. Nature 401(6756): 877–884.

See Also

`eco.xxx.regression` `phy.signal`

Examples

```
data(laja)
data <- comparative.comm(invert.tree, river.sites, invert.traits, river.env)
fingerprint.regression(data, eco.permute=10)
plot(fingerprint.regression(data, permute=10, method="lm"))
```

`generic.metrics` *Calculate any metric(s) (and compare with null distributions)*

Description

Allow the calculation of any metric within pez.

`.ses` Calculate Standard Effect Sizes of metrics

`.metric.null` Produce null randomisations and compute metrics across them

`generic.metrics` Calculate arbitrary metrics across data

Usage

```
generic.null(data, metrics, null.model = c("taxa.labels", "richness",
  "frequency", "sample.pool", "phylogeny.pool", "independentswap", "trialswap",
  "trait.asm"), permute = 1000, comp.fun = .ses, ...)

.ses(observed, null)

.metric.null(data, metrics, null.model = c("taxa.labels", "richness",
  "frequency", "sample.pool", "phylogeny.pool", "independentswap", "trialswap",
  "trait.asm"), permute = 1000, trait = -1, ...)

generic.metrics(data, metrics, ...)
```

Arguments

`data`	data `comparative.comm` object
`metrics`	vector of functions to be calculated on `data`; see pez.metrics for a list of them.
`null.model`	one of "taxa.labels", "richness", "frequency", "sample.pool", "phylogeny.pool", "independentswap", or "independentswap". These correspond to the null models available in `picante`
`permute`	number of null permutations to perform (default 1000)
`comp.fun`	comparison function to compare observed values with null values. Default is `.ses`; this is a Standard Effect Size (obs - mean)/SEmean. You may supply your own function; it should take the observed site-metric matrix as its first argument, and a site-metric-permutation array as its second. See the internals of `generic.null` for an example of its use.
`...`	additional arguments (e.g, `dist`, `abundance.weighted`) to be passed to any metric functions (see `generic.metrics` for possible arguments)

`observed`	observed metric values in site-metric matrix (*e.g.*, from `generic.metrics`)
`null`	null distributions (*e.g.*, from `.metric.null`) in a site-metric-permutation array
`trait`	if using `trait.asm` `null.model`, which trait to use (as in `trait.asm`)

Details

`generic.null` Calculate metrics and compare with null distributions. Very light wrapper around the utility functions `generic.null` and `generic.metrics` (which is, itself, a very simple function!).

Value

`generic.null` Output from `comp.fun`, by default an array (site-metric-type), where type is the observed value, the mean of the null permutations, the Standard Error of that mean, the Standard Effect Size of the metric (obs-null.mean)/SE, and then the rank of the observed value in the permutations. The rank can be considered a bootstrapped p-value of significance, but remember that this is a rank: at the 95 would be significant.

`.ses` Vector of standard effect sizes

`.metric.null` site-metric-permutation array

`generic.metrics` site-metric matrix

Note

`comp.fun` can be *anything*; much ink has been written about the use of standard effect sizes in eco-phylogenetic analyses (*e.g.*, Kembel 2009). That this function makes it easy for you to compute Standard Effect Sizes does not necessarily mean that you should (see Pearse et al. 2013).

Calculating null permutations on a dispersion metric makes little sense, since (by definition; see Pearse et al. 2014) a dispersion metric *require* the use of a null distribution to be calculated. There is nothing to stop you doing so, however! The code makes no attempt to stop you calculating null dissimilarity metrics, but I am not certain that doing so is a good idea using this code as I don't know what to do with such null models!

The `pez.shape`, `pez.evenness`, `pez.dispersion`, and `pez.dissimilarity` wrapper functions go to some trouble to stop you calculating metrics using inappropriate data (see their notes). These functions give you access to the underlying code within `pez`; there is nothing I can do to stop you calculating a metric that, in my opinion, doesn't make any sense. You have been warned :D

Author(s)

Will Pearse

References

Kembel S.W. (2009) Disentangling niche and neutral influences on community assembly: assessing the performance of community phylogenetic structure tests. Ecology letters, 12(9), 949-960.

Pearse W.D., Jones F.A. & Purvis A. (2013) Barro Colorado Island's phylogenetic assemblage structure across fine spatial scales and among clades of different ages. Ecology, 94(12), 2861-2872.

Examples

```
#Setup data
data(laja)
data <- comparative.comm(invert.tree, river.sites, invert.traits)
#Calculate some metrics
generic.metrics(data, c(.mpd, .pse))
#Compare with a trait-based null model (trait.asm)
generic.null(data, c(.mpd, .pse), "trait.asm", permute=10, trait="fish.pref")
#...be patient when running large (e.g., 1000) sets of null simulations
#You can also do this in pieces, giving even more flexibility
observed <- generic.metrics(data, c(.mpd, .pse))
#null <- .metric.null(data, c(.mpd, .pse))
#ses <- .ses(observed, null)
#...this is how everything works within generic.null
#...and, as with everything in pez, all internal functions start with a "."
```

---

laja — *Macroinvertebrate samples from the Rio Laja of Mexico*

Description

This data set includes macroinvertebrate samples from the Rio Laja, a phylogenetic tree of the taxa and traits that include mean body length and fish feeding preference as in Helmus *et al.* 2013.

Usage

```
laja
```

Format

`laja` contains a `phylo` object, a dataframe of sites-by-taxa, a dataframe of sites-by-environment, and a dataframe of traits

Author(s)

M.R. Helmus

References

Helmus M., Mercado-Silva N. & Vander Zanden M.J. (2013). Subsidies to predators, apparent competition and the phylogenetic structure of prey communities. Oecologia, 173, 997-1007.

Examples

```
data(laja)
data <- comparative.comm(invert.tree, river.sites, invert.traits, river.env)
```

pez — *Phylogenetics for the Environmental Sciences*

Description

Analysis and manipulation of eco-phylogenetic datasets containing species phylogeny, species traits, community composition, and environmental data. Provide the `comparative.comm` object to ease data manipulation, and wrappers for common community phylogenetic indices grouped according to Pearse et al. 2014: `pez.shape`, `pez.evenness`, `pez.dispersion`, and `pez.dissimilarity`. Implementation of Cavender-Bares et al. (2004) correlation of phylogenetic and ecological matrices (`fingerprint.regression`). Simulation of null assemblages, traits, and phylogenies (`scape`, `sim.meta.comm`).

References

Pearse W.D., Purvis A., Cavender-Bares J. & Helmus M.R. (2014). Metrics and Models of Community Phylogenetics. In: Modern Phylogenetic Comparative Methods and Their Application in Evolutionary Biology. Springer Berlin Heidelberg, pp. 451-464.

Cavender-Bares J., Ackerly D.D., Baum D.A. & Bazzaz F.A. (2004) Phylogenetic overdispersion in Floridian oak communities. The Americant Naturalist 163(6): 823–843.

Examples

```
require(pez)
data(laja)
data <- comparative.comm(invert.tree, river.sites, invert.traits, river.env)
pez.shape(data)
```

pez.dispersion — *Calculate (phylogenetic) dispersion: examine assemblages in the context of a source pools*

Description

As described in Pearse et al. (2014), a dispersion metric is one the examines the phylogenetic structure of species present in each assemblage in the context of a source pool of potentially present species. Unlike other metrics, the value of a dispersion metric is *contingent* on the definition of source pool, and (often) randomisations used to conduct that comparison. For completeness, options are provided to calculate these metrics using species traits.

Usage

```
pez.dispersion(data, null.model = c("taxa.labels", "richness", "frequency",
  "sample.pool", "phylogeny.pool", "independentswap", "trialswap"),
  abundance = FALSE, sqrt.phy = FALSE, traitgram = NULL,
  traitgram.p = 2, ext.dist = NULL, permute = 1000, ...)
```

Arguments

data	comparative.comm object
null.model	one of "taxa.labels", "richness", "frequency", "sample.pool", "phylogeny.pool", "independentswap", or "independentswap". These correspond to the null models available in picante; only d does not use these null models
abundance	Whether to use abundance-weighted forms of these metrics (default: FALSE). D, which is presence/absence only, and so will not be calculated when TRUE.
sqrt.phy	If TRUE (default is FALSE) your phylogenetic distance matrix will be square-rooted; specifying TRUE will force the square-root transformation on phylogenetic distance matrices (in the spirit of Leitten and Cornwell, 2014). See 'details' for details about different metric calculations when a distance matrix is used.
traitgram	If not NULL (default), a number to be passed to funct.phylo.dist (phyloWeight; the 'a' parameter), causing analysis on a distance matrix reflecting both traits and phylogeny (0 –> only phylogeny, 1 –> only traits; see funct.phylo.dist). If a vector of numbers is given, pez.dispersion iterates across them and returns a data.frame with coefficients from each iteration. See 'details' for details about different metric calculations when a distance matrix is used.
traitgram.p	A value for 'p' to be used in conjunction with traitgram when calling funct.phylo.dist.
ext.dist	Supply an external species-level distance matrix for use in calculations. See 'details' for comments on the use of distance matrices in different metric calculations.
permute	number of null permutations to perform (default 1000)
...	additional parameters to be passed to metrics (unlikely you will want to use this!)

Details

Most of these metrics do not involve comparison with some kind of evolutionary-derived expectation for phylogenetic shape. Those that do, however, such as D, make no sense unless applied to a phylogenetic distance matrix - their null expectation *requires* it. Using square-rooted distance matrices, or distance matrices that incorporate trait information, can be an excellent thing to do, but (for the above reasons), pez won't give you an answer for metrics for which WDP thinks it makes no sense. SESpd can (...up to you whether it should!...) be used with a square-rooted distance matrix, but the results *will always be wrong* if you do not have an ultrametric tree (branch lengths proportional to time) and you will be warned about this. WDP strongly feels you should only be using ultrametric phylogenies in any case, but code to fix this bug is welcome.

Value

a data.frame with metric values

Author(s)

M.R. Helmus, Will Pearse

References

Pearse W.D., Purvis A., Cavender-Bares J. & Helmus M.R. (2014). Metrics and Models of Community Phylogenetics. In: Modern Phylogenetic Comparative Methods and Their Application in Evolutionary Biology. Springer Berlin Heidelberg, pp. 451-464.

`sesmpd`,`sesmntd` Webb C.O. (2000). Exploring the phylogenetic structure of ecological communities: An example for rain forest trees. American Naturalist, 156, 145-155.

`sespd` Webb C.O., Ackerly D.D. & Kembel S.W. (2008). Phylocom: software for the analysis of phylogenetic community structure and trait evolution. Bioinformatics Applications Note, 24, 2098-2100.

`innd`,`mipd` Ness J.H., Rollinson E.J. & Whitney K.D. (2011). Phylogenetic distance can predict susceptibility to attack by natural enemies. Oikos, 120, 1327-1334.

`d` Fritz S.A. & Purvis A. (2010). Selectivity in Mammalian Extinction Risk and Threat Types: a New Measure of Phylogenetic Signal Strength in Binary Traits. Conservation Biology, 24, 1042-1051.

See Also

`pez.shape` `pez.evenness` `pez.dissimilarity`

Examples

```
data(laja)
data <- comparative.comm(invert.tree, river.sites, invert.traits)
## Not run: pez.dispersion(data)
pez.dispersion(data, permute = 100)
```

---

pez.dissimilarity — *Calculate (phylogenetic) dissimilarity: compare assemblages to one-another*

Description

As described in Pearse et al. (2014), a dissimilarity metric compares diversity between communities. WARNING: Phylosor is presented as a distance matrix here, i.e. it is *not* the fraction of shared branch length among communities, but rather '1 - shared branch length'. This means `dissimilarity` always returns a *distance* object, not a similarity object; this is a different convention from other packages.

Usage

```
pez.dissimilarity(data, metric = c("all", "unifrac", "pcd", "phylosor",
  "comdist"), abundance.weighted = FALSE, permute = 1000,
  sqrt.phy = FALSE, traitgram = NULL, traitgram.p = 2, ext.dist = NULL,
  ...)
```

Arguments

`data`	`comparative.comm` object
`metric`	default (`all`) calculates everything; individually call-able metrics are: `unifrac`, `pcd`, `phylosor`, `comdist`.
`abundance.weighted`	If TRUE (default is FALSE) metrics are calculated incorporating species abundances; only `comdist` can incorporate abundances
`permute`	Number of permutations for metric (currently only for `pcd`)
`sqrt.phy`	If TRUE (default is FALSE) your phylogenetic distance matrix will be square-rooted; specifying TRUE will force the square-root transformation on phylogenetic distance matrices (in the spirit of Leitten and Cornwell, 2014). See 'details' for details about different metric calculations when a distance matrix is used.
`traitgram`	If not NULL (default), a number to be passed to `funct.phylo.dist` (`phyloWeight`; the 'a' parameter), causing analysis on a distance matrix reflecting both traits and phylogeny (0 –> only phylogeny, 1 –> only traits; see `funct.phylo.dist`). Unlike other metric wrapper functions, `dissimilarity` does not accept a vector of traitgram values; call the function many times to get these. This is simply because it can take so long: you're probably better off looping/`apply`-ing over this function yourself.
`traitgram.p`	A value for 'p' to be used in conjunction with `traitgram` when calling `funct.phylo.dist`.
`ext.dist`	Supply an external species-level distance matrix for use in calculations. See 'details' for comments on the use of distance matrices in different metric calculations.
`...`	additional parameters to be passed to 'metric function(s) you are calling

Details

Using square-rooted distance matrices, or distance matrices that incorporate trait information, can be an excellent thing to do, but (for the above reasons), `pez` won't give you an answer for metrics for which WDP thinks it makes no sense. All results from this other than `comdist` *will always be wrong* if you do not have an ultrametric tree and square-root (branch lengths proportional to time) and you will be warned about this. WDP strongly feels you should only be using ultrametric phylogenies in any case, but code to fix this bug is welcome.

Value

list object of metric values.

Author(s)

M.R. Helmus, Will Pearse

References

Pearse W.D., Purvis A., Cavender-Bares J. & Helmus M.R. (2014). Metrics and Models of Community Phylogenetics. In: Modern Phylogenetic Comparative Methods and Their Application in Evolutionary Biology. Springer Berlin Heidelberg, pp. 451-464.

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

pcd Ives A.R. & Helmus M.R. (2010). Phylogenetic metrics of community similarity. The American Naturalist, 176, E128-E142.

phylosor Bryant J.A., Lamanna C., Morlon H., Kerkhoff A.J., Enquist B.J. & Green J.L. (2008). Microbes on mountainsides: Contrasting elevational patterns of bacterial and plant diversity. Proceedings of the National Academy of Sciences of the United States of America, 105, 11505-11511.

comdist C.O. Webb, D.D. Ackerly, and S.W. Kembel. 2008. Phylocom: software for the analysis of phylogenetic community structure and trait evolution. Bioinformatics 18:2098-2100.

See Also

pez.shape pez.evenness pez.dispersion

Examples

```
data(laja)
data <- comparative.comm(invert.tree, river.sites, invert.traits)
## Not run:
dissim <- pez.dissimilarity(data)

## End(Not run)
```

---

pez.evenness *Calculate (phylogenetic) evenness: examine assemblage composition and abundance*

Description

As described in Pearse et al. (2014), an evenness metric is one the examines the phylogenetic structure of species present in each assemblage, taking into account their abundances. For completeness, options are provided to calculate these metrics using species traits.

Usage

```
pez.evenness(data, sqrt.phy = FALSE, traitgram = NULL, traitgram.p = 2,
  ext.dist = NULL, quick = TRUE, q = 1e-04)
```

Arguments

data	comparative.comm object
sqrt.phy	If TRUE (default is FALSE) your phylogenetic distance matrix will be square-rooted; specifying TRUE will force the square-root transformation on phylogenetic distance matrices (in the spirit of Leitten and Cornwell, 2014). See 'details' for details about different metric calculations when a distance matrix is used.

`traitgram` If not NULL (default), a number to be passed to `funct.phylo.dist` (`phyloWeight`; the 'a' parameter), causing analysis on a distance matrix reflecting both traits and phylogeny (0–>only phylogeny, 1–> only traits; see `funct.phylo.dist`). If a vector of numbers is given, `pez.eveness` iterates across them and returns a `data.frame` with coefficients from each iteration. See 'details' for details about different metric calculations when a distance matrix is used.

`traitgram.p` A value for 'p' to be used in conjunction with `traitgram` when calling `funct.phylo.dist`.

`ext.dist` Supply an external species-level distance matrix for use in calculations. See 'details' for comments on the use of distance matrices in different metric calculations.

`quick` Only calculate metrics which are quick to calculate (default: TRUE); setting to FALSE will also calculate `fd.dist` and the Pagel transformations (

$$\lambda$$

,

$$\delta$$

,

$$\kappa$$

).

`q` value for q in `scheiner` (default 0.0001)

Details

Most of these metrics do not involve comparison with some kind of evolutionary-derived expectation for phylogenetic shape. Those that do, however, such as PSE, make no sense unless applied to a phylogenetic distance matrix - their null expectation *requires* it. Using square-rooted distance matrices, or distance matrices that incorporate trait information, can be an excellent thing to do, but (for the above reasons), `pez` won't give you an answer for metrics for which WDP thinks it makes no sense. `pae`, `iac`, `haead` & `eaed` can (...up to you whether you should!...) be used with a square-rooted distance matrix, but the results *will always be wrong* if you do not have an ultrametric tree (branch lengths proportional to time) and you will be warned about this. WDP strongly feels you should only be using ultrametric phylogenies in any case, but code to fix this bug is welcome.

Value

`phy.structure` list object of metric values. Use `coefs` to extract a summary metric table, or examine each individual metric (which gives more details for each) by calling `print` on the output (i.e., type `output` in the example below).

Note

As mentioned above, `dist.fd` is calculated using a phylogenetic distance matrix if no trait data are available, or if you specify `sqrt.phy`. It is not calculated by default because it generates warning messsages (which WDP is loathe to suppress) which are related to the general tendency for a low rank of phylogenetic distance matrices. Much ink has been written about this, and in par this problem is why the `eigen.sum` measure came to be suggested.

Some of these metrics can cause (inconsequential) warnings if given assemblages with only one species/individual in them, and return NA/NaN values depending on the metric. I consider these 'features', not bugs.

Some of the metrics in this wrapper are also in `pez.shape`; such metrics can be calculated using species' abundances (making them *evenness*) metrics or simply using presence/absence of species (making them *shape* metrics).

Author(s)

M.R. Helmus, Will Pearse

References

Pearse W.D., Purvis A., Cavender-Bares J. & Helmus M.R. (2014). Metrics and Models of Community Phylogenetics. In: Modern Phylogenetic Comparative Methods and Their Application in Evolutionary Biology. Springer Berlin Heidelberg, pp. 451-464.

`pse` Helmus M.R., Bland T.J., Williams C.K. & Ives A.R. (2007). Phylogenetic measures of biodiversity. American Naturalist, 169, E68-E83.

Pearse W.D., Purvis A., Cavender-Bares J. & Helmus M.R. (2014). Metrics and Models of Community Phylogenetics. In: Modern Phylogenetic Comparative Methods and Their Application in Evolutionary Biology. Springer Berlin Heidelberg, pp. 451-464.

`pse` Helmus M.R., Bland T.J., Williams C.K. & Ives A.R. (2007). Phylogenetic measures of biodiversity. American Naturalist, 169, E68-E83.

`rao` Webb C.O. (2000). Exploring the phylogenetic structure of ecological communities: An example for rain forest trees. American Naturalist, 156, 145-155.

`taxon` Clarke K.R. & Warwick R.M. (1998). A taxonomic distinctness index and its statistical properties. J. Appl. Ecol., 35, 523-531.

`entropy` Allen B., Kon M. & Bar-Yam Y. (2009). A New Phylogenetic Diversity Measure Generalizing the Shannon Index and Its Application to Phyllostomid Bats. The American Naturalist, 174, 236-243.

`pae,iac,haed,eaed` Cadotte M.W., Davies T.J., Regetz J., Kembel S.W., Cleland E. & Oakley T.H. (2010). Phylogenetic diversity metrics for ecological communities: integrating species richness, abundance and evolutionary history. Ecology Letters, 13, 96-105.

`lambda,delta,kappa` Mark Pagel (1999) Inferring the historical patterns of biological evolution. Nature 6756(401): 877–884.

`innd,mipd` Ness J.H., Rollinson E.J. & Whitney K.D. (2011). Phylogenetic distance can predict susceptibility to attack by natural enemies. Oikos, 120, 1327-1334.

`scheiner` Scheiner, S.M. (20120). A metric of biodiversity that integrates abundance, phylogeny, and function. Oikos, 121, 1191-1202.

See Also

pez.shape pez.dispersion pez.dissimilarity

Examples

```
data(laja)
data <- comparative.comm(invert.tree, river.sites, invert.traits)
pez.evenness(data)
```

---

pez.metrics *Phylogenetic and functional trait metrics within pez*

Description

Using these functions, you can calculate any of the phylogenetic metrics within pez, using comparative.comm objects. While you can call each individually, using the pez.shape, pez.evenness, pez.dispersion, and pez.dissimilarity wrapper functions (and the more flexible generic.metrics and null model functions) are probably your best bet. Note that *all of these functions* take a common first parameter: a comparative.comm object. There are additional parameters that can be passed, which are described below.

Usage

```
.colless(x, ...)

.hed(x, ...)

.eed(x, na.rm = TRUE, ...)

.psv(x, ...)

.psr(x, ...)

.mpd(x, dist = NULL, abundance.weighted = FALSE, ...)

.pd(x, include.root = TRUE, abundance.weighted = FALSE, ...)

.mntd(x, dist = NULL, abundance.weighted = FALSE, ...)

.gamma(x, ...)

.taxon(x, dist = NULL, abundance.weighted = FALSE, ...)

.eigen.sum(x, dist = NULL, which.eigen = 1, ...)

.dist.fd(x, method = "phy", abundance.weighted = FALSE, ...)

.sqrt.phy(x)

.phylo.entropy(x, ...)
```

```
.aed(x, ...)

.haed(x, ...)

.simpson.phylogenetic(x)

.iac(x, na.rm = TRUE, ...)

.pae(x, na.rm = TRUE, ...)

.scheiner(x, q = 1e-04, abundance.weighted = FALSE, ...)

.pse(x, ...)

.rao(x, ...)

.lambda(x, ...)

.delta(x, ...)

.kappa(x, ...)

.eaed(x, ...)

.unifrac(x, ...)

.pcd(x, permute = 1000, ...)

.comdist(x, dist = NULL, abundance.weighted = FALSE, ...)

.phylosor(x, dist = NULL, abundance.weighted = FALSE, ...)

.d(x, permute = 1000, ...)

.ses.mpd(x, dist = NULL, null.model = "taxa.labels",
  abundance.weighted = FALSE, permute = 1000, ...)

.ses.mntd(x, dist = NULL, null.model = "taxa.labels",
  abundance.weighted = FALSE, permute = 1000, ...)

.ses.mipd(x, dist = NULL, null.model = "taxa.labels",
  abundance.weighted = FALSE, permute = 1000, ...)

.ses.innd(x, dist = NULL, null.model = "taxa.labels",
  abundance.weighted = FALSE, permute = 1000, ...)

.mipd(x, dist = NULL, abundance.weighted = FALSE, ...)
```

```
.innd(x, dist = NULL, abundance.weighted = FALSE, ...)
```

Arguments

x	comparative.comm object
...	ignored
na.rm	remove NAs in calculations (altering this can obscure errors that are meaningful; I would advise leaving alone)
dist	distance matrix for use with calculations; could be generated from traits, a square-root-transformed distance matrix (see .sqrt.phy for creating a comparative.comm object with a square-root transformed phylogeny). Default: NULL (–> calculate distance matrix from phylogeny)
abundance.weighted	whether to include species' abundances in metric calculation, often dictating whether you're calculating a pez.shape or pez.evenness metric. Default: FALSE
include.root	include root in PD calculations (default is TRUE, as in picante, but within pez.shape I specify FALSE
which.eigen	which phylo-eigenvector to be used for PVR metric
method	whether to calculate using phylogeny ("phy"; default) or trait data ("traits")
q	the q parameter for .scheiner; default 0.0001
permute	number of permutations of null randomisations (mostly only applies to dispersion metrics)
null.model	one of "taxa.labels", "richness", "frequency", "sample.pool", "phylogeny.pool", "independentswap", or "independentswap". These correspond to the null models available in picante; only d does not use these null models

Details

.pd returns two metrics: Faith's PD (which does not take into account abundance) and Faith's PD corrected for species richness or total abundance (depending on abundance.weighted). I am almost certain that I got the idea for this from somewhere, but I can't find the reference: if you published on this before 2012, please get in touch with me.

Note

Many (but not all) of these functions are fairly trivial wrappers around functions in other packages. In the citations for each metric, * indicates a function that's essentially written in picante. The Pagel family of measures are also fairly trivial wrapper around caper code, functional dissimilarity FD code, gamma ape code, and colless apTreeshape code. I can't demand it, but I would be grateful if you would cite these authors when using these wrappers.

The pez.shape, pez.evenness, pez.dispersion, and pez.dissimilarity wrapper functions go to some trouble to stop you calculating metrics using inappropriate data (see their notes). These functions give you access to the underlying code within pez; there is nothing I can do to stop you calculating a metric that, in my opinion, doesn't make any sense. You have been warned :D

If you're a developer hoping to make your metric(s) work in this framework, please use the argument naming convention for arguments described in this help file, and use the `...` operator in your definition. That way functions that don't need particular arguments can co-exist peacefully with those that do. The first argument to one of these functions should *always* be a `comparative.comm` object; there is no method dispatch on any of these functions and I foresee future pain without this rule.

References

`colless` Colless D.H. (1982). Review of phylogenetics: the theory and practice of phylogenetic systematics. Systematic Zoology, 31, 100-104.

`eed,hed` (i.e., *Eed, Hed*) Cadotte M.W., Davies T.J., Regetz J., Kembel S.W., Cleland E. & Oakley T.H. (2010). Phylogenetic diversity metrics for ecological communities: integrating species richness, abundance and evolutionary history. Ecology Letters, 13, 96-105.

`PSV,PSR,PSE` Helmus M.R., Bland T.J., Williams C.K. & Ives A.R. (2007). Phylogenetic measures of biodiversity. American Naturalist, 169, E68-E83.

`PD` Faith D.P. (1992). Conservation evaluation and phylogenetic diversity. Biological Conservation, 61, 1-10.

`gamma` Pybus O.G. & Harvey P.H. (2000) Testing macro-evolutionary models using incomplete molecular phylogenies. _Proceedings of the Royal Society of London. Series B. Biological Sciences 267: 2267–2272.

`taxon` Clarke K.R. & Warwick R.M. (1998). A taxonomic distinctness index and its statistical properties. J. Appl. Ecol., 35, 523-531.

`eigen.sum` Diniz-Filho J.A.F., Cianciaruso M.V., Rangel T.F. & Bini L.M. (2011). Eigenvector estimation of phylogenetic and functional diversity. Functional Ecology, 25, 735-744.

`entropy` Allen B., Kon M. & Bar-Yam Y. (2009). A New Phylogenetic Diversity Measure Generalizing the Shannon Index and Its Application to Phyllostomid Bats. The American Naturalist, 174, 236-243.

`pae,aed,iac,haed,eaed` Cadotte M.W., Davies T.J., Regetz J., Kembel S.W., Cleland E. & Oakley T.H. (2010). Phylogenetic diversity metrics for ecological communities: integrating species richness, abundance and evolutionary history. Ecology Letters, 13, 96-105.

`scheiner` Scheiner, S.M. (20120). A metric of biodiversity that integrates abundance, phylogeny, and function. Oikos, 121, 1191-1202.

`rao` Webb C.O. (2000). Exploring the phylogenetic structure of ecological communities: An example for rain forest trees. American Naturalist, 156, 145-155.

`lambda,delta,kappa` Mark Pagel (1999) Inferring the historical patterns of biological evolution. Nature 6756(401): 877–884.

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

`pcd` Ives A.R. & Helmus M.R. (2010). Phylogenetic metrics of community similarity. The American Naturalist, 176, E128-E142.

`comdist` C.O. Webb, D.D. Ackerly, and S.W. Kembel. 2008. Phylocom: software for the analysis of phylogenetic community structure and trait evolution. Bioinformatics 18:2098-2100.

phylosor Bryant J.A., Lamanna C., Morlon H., Kerkhoff A.J., Enquist B.J. & Green J.L. (2008). Microbes on mountainsides: Contrasting elevational patterns of bacterial and plant diversity. Proceedings of the National Academy of Sciences of the United States of America, 105, 11505-11511.

d Fritz S.A. & Purvis A. (2010). Selectivity in Mammalian Extinction Risk and Threat Types: a New Measure of Phylogenetic Signal Strength in Binary Traits. Conservation Biology, 24, 1042-1051.

sesmpd,sesmntd Webb C.O. (2000). Exploring the phylogenetic structure of ecological communities: An example for rain forest trees. American Naturalist, 156, 145-155.

innd,mipd Ness J.H., Rollinson E.J. & Whitney K.D. (2011). Phylogenetic distance can predict susceptibility to attack by natural enemies. Oikos, 120, 1327-1334.

Examples

```
data(laja)
data <- comparative.comm(invert.tree, river.sites)
.psv(data)
```

---

pez.shape *Calculate (phylogenetic) shape: examine assemblage composition*

Description

As described in Pearse et al. (2014), a shape metric is one the examines the phylogenetic structure of species present in each assemblage, ignoring abundances entirely. For completeness, options are provided to calculate these metrics using species traits.

Usage

```
pez.shape(data, sqrt.phy = FALSE, traitgram = NULL, traitgram.p = 2,
  ext.dist = NULL, which.eigen = 1, quick = TRUE, q = 1e-04)
```

Arguments

data	comparative.comm object
sqrt.phy	If TRUE (default is FALSE) your phylogenetic distance matrix will be square-rooted; specifying TRUE will force the square-root transformation on phylogenetic distance matrices (in the spirit of Leitten and Cornwell, 2014). See 'details' for details about different metric calculations when a distance matrix is used.
traitgram	If not NULL (default), a number to be passed to funct.phylo.dist (phyloWeight; the 'a' parameter), causing analysis on a distance matrix reflecting both traits and phylogeny (0 –> only phylogeny, 1 –> only traits; see funct.phylo.dist). If a vector of numbers is given, pez.shape iterates across them and returns a data.frame with coefficients from each iteration. See 'details' for details about different metric calculations when a distance matrix is used.
traitgram.p	A value for 'p' to be used in conjunction with traitgram when calling funct.phylo.dist.

`ext.dist`	Supply an external species-level distance matrix for use in calculations. See 'details' for comments on the use of distance matrices in different metric calculations.
`which.eigen`	The eigen vector to calculate for the PhyloEigen metric (`eigen.sum`)
`quick`	Only calculate metrics which are quick to calculate (default: TRUE); setting to FALSE will also calculate `fd.dist`.
`q`	value for *q* in `scheiner` (default 0.0001)

Details

Most of these metrics do not involve comparison with some kind of evolutionary-derived expectation for phylogenetic shape. Those that do, however, such as PSV or Colless' index, make no sense unless applied to a phylogenetic distance matrix - their null expectation *requires* it. Using square-rooted distance matrices, or distance matrices that incorporate trait information, can be an excellent thing to do, but (for the above reasons), `pez` won't give you an answer for metrics for which WDP thinks it makes no sense. `pd`, `eed` & `hed` can (...up to you whether you should!...) be used with a square-rooted distance matrix, but the results *will always be wrong* if you do not have an ultrametric tree (branch lengths proportional to time) and you will be warned about this. WDP strongly feels you should only be using ultrametric phylogenies in any case, but code to fix this bug is welcome.

Value

`phy.structure` list object of metric values. Use `coefs` to extract a summary metric table, or examine each individual metric (which gives more details for each) by calling `print` on the output (i.e., type `output` in the example below).

Some of the metrics in this wrapper are also in `pez.evenness`; such metrics can be calculated using species' abundances (making them *evenness*) metrics or simply using presence/absence of species (making them *shape* metrics).

Note

As mentioned above, `dist.fd` is calculated using a phylogenetic distance matrix if no trait data are available, or if you specify `sqrt.phy`. It is not calculated by default because it generates warning messsages (which WDP is loathe to suppress) which are related to the general tendency for a low rank of phylogenetic distance matrices. Much ink has been written about this, and in par this problem is why the `eigen.sum` measure came to be suggested.

Many of these metrics, (*e.g.*, `eed`) will cause (inconsequential) warnings if given assemblages with only one species in them, and return NA/NaN values depending on the metric. I consider these 'features', not bugs.

Author(s)

M.R. Helmus, Will Pearse

References

Pearse W.D., Purvis A., Cavender-Bares J. & Helmus M.R. (2014). Metrics and Models of Community Phylogenetics. In: Modern Phylogenetic Comparative Methods and Their Application in Evolutionary Biology. Springer Berlin Heidelberg, pp. 451-464.

`PSV,PSR` Helmus M.R., Bland T.J., Williams C.K. & Ives A.R. (2007). Phylogenetic measures of biodiversity. American Naturalist, 169, E68-E83.

`PD` Faith D.P. (1992). Conservation evaluation and phylogenetic diversity. Biological Conservation, 61, 1-10.

`colless` Colless D.H. (1982). Review of phylogenetics: the theory and practice of phylogenetic systematics. Systematic Zoology, 31, 100-104.

`gamma` Pybus O.G. & Harvey P.H. (2000) Testing macro-evolutionary models using incomplete molecular phylogenies. _Proceedings of the Royal Society of London. Series B. Biological Sciences 267: 2267–2272.

`taxon` Clarke K.R. & Warwick R.M. (1998). A taxonomic distinctness index and its statistical properties. J. Appl. Ecol., 35, 523-531.

`eigen.sum` Diniz-Filho J.A.F., Cianciaruso M.V., Rangel T.F. & Bini L.M. (2011). Eigenvector estimation of phylogenetic and functional diversity. Functional Ecology, 25, 735-744.

`eed,hed` (i.e., *Eed, Hed*) Cadotte M.W., Davies T.J., Regetz J., Kembel S.W., Cleland E. & Oakley T.H. (2010). Phylogenetic diversity metrics for ecological communities: integrating species richness, abundance and evolutionary history. Ecology Letters, 13, 96-105.

`innd,mipd` Ness J.H., Rollinson E.J. & Whitney K.D. (2011). Phylogenetic distance can predict susceptibility to attack by natural enemies. Oikos, 120, 1327-1334.

`scheiner` Scheiner, S.M. (20120). A metric of biodiversity that integrates abundance, phylogeny, and function. Oikos, 121, 1191-1202.

See Also

`pez.evenness` `pez.dispersion` `pez.dissimilarity`

Examples

```
data(laja)
data <- comparative.comm(invert.tree, river.sites, invert.traits)
(output<-pez.shape(data))
```

`pglmm` *Phylogenetic Generalised Linear Mixed Model for Community Data*

Description

This function performs Generalized Linear Mixed Models for binary and continuous phylogenetic data, estimating regression coefficients with approximate standard errors. It is modeled after `lmer` but is more general by allowing correlation structure within random effects; these correlations can be phylogenetic among species, or any other correlation structure, such as geographical correlations among sites. It is, however, much more specific than `lmer` in that it can only analyze a subset of1 the types of model designed handled by `lmer`. It is also much slower than `lmer` and requires users to specify correlation structures as covariance matrices. `communityPGLMM` can analyze models in Ives and Helmus (2011). It can also analyze bipartite phylogenetic data, such as that analyzed in Rafferty and Ives (2011), by giving sites phylogenetic correlations.

`communityPGLMM.binary` calculates the statistical significance of the random effects in the generalized linear mixed model from the marginal profile likelihood.

`communityPGLMM.binary.LRT` tests statistical significance of the phylogenetic random effect on species slopes using a likelihood ratio test

`communityPGLMM.matrix.structure` produces the entire covariance matrix structure (V) when you specify random effects.

`communityPGLMM.predicted.values` calculates the predicted values of Y; for the generalized linear mixed model (family = "binomial"), these values are in the logit-1 transformed space.

Usage

```
communityPGLMM(formula, data = list(), family = "gaussian", sp = NULL,
  site = NULL, random.effects = list(), REML = TRUE, s2.init = NULL,
  B.init = NULL, reltol = 10^-6, maxit = 500, tol.pql = 10^-6,
  maxit.pql = 200, verbose = FALSE)

communityPGLMM.gaussian(formula, data = list(), family = "gaussian",
  sp = NULL, site = NULL, random.effects = list(), REML = TRUE,
  s2.init = NULL, B.init = NULL, reltol = 10^-8, maxit = 500,
  verbose = FALSE)

communityPGLMM.binary(formula, data = list(), family = "binomial",
  sp = NULL, site = NULL, random.effects = list(), REML = TRUE,
  s2.init = 0.25, B.init = NULL, reltol = 10^-5, maxit = 40,
  tol.pql = 10^-6, maxit.pql = 200, verbose = FALSE)

communityPGLMM.binary.LRT(x, re.number = 0, ...)

communityPGLMM.matrix.structure(formula, data = list(), family = "binomial",
  sp = NULL, site = NULL, random.effects = list(), ss = 1)

## S3 method for class 'communityPGLMM'
summary(object, digits = max(3, getOption("digits") -
  3), ...)

## S3 method for class 'communityPGLMM'
plot(x, digits = max(3, getOption("digits") - 3),
```

```
  ...)

communityPGLMM.predicted.values(x, show.plot = TRUE, ...)
```

Arguments

formula	a two-sided linear formula object describing the fixed-effects of the model; for example, `Y ~ X`.
data	a `data.frame` containing the variables named in formula. The data frame should have long format with factors specifying species and sites. `communityPGLMM` will reorder rows of the data frame so that species are nested within sites. Please note that calling `as.data.frame.comparative.comm` will return your `comparative.comm` object into this format for you.
family	either `gaussian` for a Linear Mixed Model, or `binomial` for binary dependent data.
sp	a `factor` variable that identifies species
site	a `factor` variable that identifies sites
random.effects	a `list` that contains, for non-nested random effects, lists of triplets of the form `list(X, group = group, covar = V)`. This is modeled after the `lmer` formula syntax `(X | group)` where `X` is a variable and `group` is a grouping factor. Note that `group` should be either your `sp` or `site` variable specified in `sp` and `site`. The additional term `V` is a covariance matrix of rank equal to the number of levels of group that specifies the covariances among groups in the random effect `X`. For nested variable random effects, `random.effects` contains lists of quadruplets of the form `list(X, group1 = group1, covar = V, group2 = group2)` where `group1` is nested within `group2`.
REML	whether REML or ML is used for model fitting. For the generalized linear mixed model for binary data, these don't have standard interpretations, and there is no log likelihood function that can be used in likelihood ratio tests.
s2.init	an array of initial estimates of s2 for each random effect that scales the variance. If s2.init is not provided for `family="gaussian"`, these are estimated using in a clunky way using `lm` assuming no phylogenetic signal. A better approach is to run `link[lme4:lmer]{lmer}` and use the output random effects for `s2.init`. If `s2.init` is not provided for `family="binomial"`, these are set to 0.25.
B.init	initial estimates of B, a matrix containing regression coefficients in the model for the fixed effects. This matrix must have `dim(B.init)=c(p+1,1)`, where `p` is the number of predictor (independent) variables; the first element of `B` corresponds to the intercept, and the remaining elements correspond in order to the predictor (independent) variables in the formula. If `B.init` is not provided, these are estimated using in a clunky way using `lm` or `glm` assuming no phylogenetic signal. A better approach is to run `lmer` and use the output fixed effects for `B.init`.
reltol	a control parameter dictating the relative tolerance for convergence in the optimization; see `optim`.
maxit	a control parameter dictating the maximum number of iterations in the optimization; see `optim`.

tol.pql	a control parameter dictating the tolerance for convergence in the PQL estimates of the mean components of the binomial GLMM.
maxit.pql	a control parameter dictating the maximum number of iterations in the PQL estimates of the mean components of the binomial GLMM.
verbose	if TRUE, the model deviance and running estimates of s2 and B are plotted each iteration during optimization.
x	communityPGLMM object
re.number	which random.effect in x to be tested
...	additional arguments to summary and plotting functions (currently ignored)
ss	which of the random.effects to produce
object	communityPGLMM object to be summarised
digits	minimal number of significant digits for printing, as in print.default
show.plot	if TRUE (default), display plot

Details

The vignette 'pez-pglmm-overview' gives a gentle introduction to using PGLMMS. For linear mixed models (family = 'gaussian'), the function estimates parameters for the model of the form, for example,

$$Y = \beta_0 + \beta_1 x + b_0 + b_1 x$$

$$b_0 \tilde{} Gaussian(0, \sigma_0^2 I_{sp})$$

$$b_1 \tilde{} Gaussian(0, \sigma_0^2 V_{sp})$$

$$\eta \tilde{} Gaussian(0, \sigma^2)$$

where β_0 and β_1 are fixed effects, and V_{sp} is a variance-covariance matrix derived from a phylogeny (typically under the assumption of Brownian motion evolution). Here, the variation in the mean (intercept) for each species is given by the random effect b_0 that is assumed to be independent among species. Variation in species' responses to predictor variable x is given by a random effect b_0 that is assumed to depend on the phylogenetic relatedness among species given by V_{sp}; if species are closely related, their specific responses to x will be similar. This particular model would be specified as

```
re.1 <- list(1, sp = dat$sp, covar = diag(nspp)) re.2 <- list(dat$X, sp = dat$sp, covar = Vsp)
z <- communityPGLMM(Y ~ X, data = data, family = "gaussian", random.effects = list(re.1, re.2))
```

The covariance matrix covar is standardized to have its determinant equal to 1. This in effect standardizes the interpretation of the scalar σ^2. Although mathematically this is not required, it is a very good idea to standardize the predictor (independent) variables to have mean 0 and variance 1. This will make the function more robust and improve the interpretation of the regression coefficients. For categorical (factor) predictor variables, you will need to construct 0-1 dummy variables, and these should not be standardized (for obvious reasons).

For binary generalized linear mixed models (family = 'binomial'), the function estimates parameters for the model of the form, for example,

$$y = \beta_0 + \beta_1 x + b_0 + b_1 x$$

$$Y = logit^{-1}(y)$$

$$b_0 \tilde{} Gaussian(0, \sigma_0^2 I_{sp})$$

$$b_1 \tilde{} Gaussian(0, \sigma_0^2 V_{sp})$$

where β_0 and β_1 are fixed effects, and V_{sp} is a variance-covariance matrix derived from a phylogeny (typically under the assumption of Brownian motion evolution).

```
z <- communityPGLMM(Y ~ X, data = data, family = 'binomial', random.effects = list(re.1, re.2))
```

As with the linear mixed model, it is a very good idea to standardize the predictor (independent) variables to have mean 0 and variance 1. This will make the function more robust and improve the interpretation of the regression coefficients. For categorical (factor) predictor variables, you will need to construct 0-1 dummy variables, and these should not be standardized (for obvious reasons).

Value

an object of class `communityPGLMM`

`formula`	the formula for fixed effects
`data`	the dataset
`family`	either `gaussian` or `binomial` depending on the model fit
`random.effects`	the list of random effects
`B`	estimates of the regression coefficients
`B.se`	approximate standard errors of the fixed effects regression coefficients
`B.cov`	approximate covariance matrix for the fixed effects regression coefficients
`B.zscore`	approximate Z scores for the fixed effects regression coefficients
`B.pvalue`	approximate tests for the fixed effects regression coefficients being different from zero
`ss`	random effects' standard deviations for the covariance matrix $\sigma^2 V$ for each random effect in order. For the linear mixed model, the residual variance is listed last
`s2r`	random effects variances for non-nested random effects
`s2n`	random effects variances for nested random effects
`s2resid`	for linear mixed models, the residual vairance
`logLIK`	for linear mixed models, the log-likelihood for either the restricted likelihood (`REML=TRUE`) or the overall likelihood (`REML=FALSE`). This is set to NULL for generalised linear mixed models
`AIC`	for linear mixed models, the AIC for either the restricted likelihood (`REML=TRUE`) or the overall likelihood (`REML=FALSE`). This is set to NULL for generalised linear mixed models

BIC	for linear mixed models, the BIC for either the restricted likelihood (REML=TRUE) or the overall likelihood (REML=FALSE). This is set to NULL for generalised linear mixed models
REML	whether or not REML is used (TRUE or FALSE)
s2.init	the user-provided initial estimates of s2
B.init	the user-provided initial estimates of B
Y	the response (dependent) variable returned in matrix form
X	the predictor (independent) variables returned in matrix form (including 1s in the first column)
H	the residuals. For the generalized linear mixed model, these are the predicted residuals in the $logit^{-1}$ space
iV	the inverse of the covariance matrix for the entire system (of dimension (nsp*nsite) by (nsp*nsite))
mu	predicted mean values for the generalized linear mixed model. Set to NULL for linear mixed models
sp, sp	matrices used to construct the nested design matrix
Zt	the design matrix for random effects
St	diagonal matrix that maps the random effects variances onto the design matrix
convcode	the convergence code provided by optim
niter	number of iterations performed by optim

Note

These function *do not* use a comparative.comm object, but you can use as.data.frame.comparative.comm to create a data.frame for use with these functions. The power of this method comes from deciding your own parameters parameters to be determined (the data for regression, the random effects, etc.), and it is our hope that this interface gives you more flexibility in model selection/fitting.

Author(s)

Anthony R. Ives, cosmetic changes by Will Pearse

References

Ives, A. R. and M. R. Helmus. 2011. Generalized linear mixed models for phylogenetic analyses of community structure. Ecological Monographs 81:511-525.

Rafferty, N. E., and A. R. Ives. 2013. Phylogenetic trait-based analyses of ecological networks. Ecology 94:2321-2333.

Examples

```
## Structure of examples:
# First, a (brief) description of model types, and how they are specified
# - these are *not* to be run 'as-is'; they show how models should be organised
# Second, a run-through of how to simulate, and then analyse, data
```

```
# - these *are* to be run 'as-is'; they show how to format and work with data

## Not run:
#######################################################
#First section; brief summary of models and their use####
#######################################################
## Model structures from Ives & Helmus (2011)
# dat = data set for regression (note: *not* an comparative.comm object)
# nspp = number of species
# nsite = number of sites
# Vphy = phylogenetic covariance matrix for species
# Vrepul = inverse of Vphy representing phylogenetic repulsion

# Model 1 (Eq. 1)
re.site <- list(1, site = dat$site, covar = diag(nsite))
re.sp.site <- list(1, sp = dat$sp, covar = Vphy, site = dat$site) # note: nested
z <- communityPGLMM(freq ~ sp, data = dat, family = "binomial", sp
= dat$sp, site = dat$site, random.effects = list(re.site,
re.sp.site), REML = TRUE, verbose = TRUE, s2.init=.1)


# Model 2 (Eq. 2)
re.site <- list(1, site = dat$site, covar = diag(nsite))
re.slope <- list(X, sp = dat$sp, covar = diag(nspp))
re.slopephy <- list(X, sp = dat$sp, covar = Vphy)
z <- communityPGLMM(freq ~ sp + X, data = dat, family = "binomial",
sp = dat$sp, site = dat$site, random.effects = list(re.site,
re.slope, re.slopephy), REML = TRUE, verbose = TRUE, s2.init=.1)

# Model 3 (Eq. 3)
re.site <- list(1, site = dat$site, covar = diag(nsite))
re.sp.site <- list(1, sp = dat$sp, covar = Vrepul, site = dat$site) # note: nested
z <- communityPGLMM(freq ~ sp*X, data = dat, family = "binomial",
sp = dat$sp, site = dat$site, random.effects = list(re.site,
re.sp.site), REML = TRUE, verbose = TRUE, s2.init=.1)

## Model structure from Rafferty & Ives (2013) (Eq. 3)
# dat = data set
# npp = number of pollinators (sp)
# nsite = number of plants (site)
# VphyPol = phylogenetic covariance matrix for pollinators
# VphyPlt = phylogenetic covariance matrix for plants

re.a <- list(1, sp = dat$sp, covar = diag(nspp))
re.b <- list(1, sp = dat$sp, covar = VphyPol)
re.c <- list(1, sp = dat$sp, covar = VphyPol, dat$site)
re.d <- list(1, site = dat$site, covar = diag(nsite))
re.f <- list(1, site = dat$site, covar = VphyPlt)
re.g <- list(1, site = dat$site, covar = VphyPlt, dat$sp)
#term h isn't possible in this implementation, but can be done with
available matlab code

z <- communityPGLMM(freq ~ sp*X, data = dat, family = "binomial",
```

```
sp = dat$sp, site = dat$site, random.effects = list(re.a, re.b,
re.c, re.d, re.f, re.g), REML = TRUE, verbose = TRUE, s2.init=.1)

## End(Not run)

#######################################################
#Second section; detailed simulation and analysis #######
#NOTE: this section is explained and annotated in #######
#      detail in the vignette 'pez-pglmm-overview'#######
#      run 'vignette('pez-pglmm-overview') to read#######
#######################################################
# Generate simulated data for nspp species and nsite sites
nspp <- 15
nsite <- 10

# residual variance (set to zero for binary data)
sd.resid <- 0

# fixed effects
beta0 <- 0
beta1 <- 0

# magnitude of random effects
sd.B0 <- 1
sd.B1 <- 1

# whether or not to include phylogenetic signal in B0 and B1
signal.B0 <- TRUE
signal.B1 <- TRUE

# simulate a phylogenetic tree
phy <- rtree(n = nspp)
phy <- compute.brlen(phy, method = "Grafen", power = 0.5)

# standardize the phylogenetic covariance matrix to have determinant 1
Vphy <- vcv(phy)
Vphy <- Vphy/(det(Vphy)^(1/nspp))

# Generate environmental site variable
X <- matrix(1:nsite, nrow = 1, ncol = nsite)
X <- (X - mean(X))/sd(X)

# Perform a Cholesky decomposition of Vphy. This is used to
# generate phylogenetic signal: a vector of independent normal random
# variables, when multiplied by the transpose of the Cholesky
# deposition of Vphy will have covariance matrix equal to Vphy.

iD <- t(chol(Vphy))

# Set up species-specific regression coefficients as random effects
if (signal.B0 == TRUE) {
b0 <- beta0 + iD %*% rnorm(nspp, sd = sd.B0)
} else {
```

```
b0 <- beta0 + rnorm(nspp, sd = sd.B0)
}
if (signal.B1 == TRUE) {
b1 <- beta1 + iD %*% rnorm(nspp, sd = sd.B1)
} else {
b1 <- beta1 + rnorm(nspp, sd = sd.B1)
}

# Simulate species abundances among sites to give matrix Y that
# contains species in rows and sites in columns
y <- rep(b0, each=nsite)
y <- y + rep(b1, each=nsite) * rep(X, nspp)
y <- y + rnorm(nspp*nsite) #add some random 'error'
Y <- rbinom(length(y), size=1, prob=exp(y)/(1+exp(y)))
y <- matrix(outer(b0, array(1, dim = c(1, nsite))), nrow = nspp,
ncol = nsite) + matrix(outer(b1, X), nrow = nspp, ncol = nsite)
e <- rnorm(nspp * nsite, sd = sd.resid)
y <- y + matrix(e, nrow = nspp, ncol = nsite)
y <- matrix(y, nrow = nspp * nsite, ncol = 1)

Y <- rbinom(n = length(y), size = 1, prob = exp(y)/(1 + exp(y)))
Y <- matrix(Y, nrow = nspp, ncol = nsite)

# name the simulated species 1:nspp and sites 1:nsites
rownames(Y) <- 1:nspp
colnames(Y) <- 1:nsite

par(mfrow = c(3, 1), las = 1, mar = c(2, 4, 2, 2) - 0.1)
matplot(t(X), type = "l", ylab = "X", main = "X among sites")
hist(b0, xlab = "b0", main = "b0 among species")
hist(b1, xlab = "b1", main = "b1 among species")

#Plot out; you get essentially this from plot(your.pglmm.model)
image(t(Y), ylab = "species", xlab = "sites", main = "abundance",
col=c("black","white"))

# Transform data matrices into "long" form, and generate a data frame
YY <- matrix(Y, nrow = nspp * nsite, ncol = 1)

XX <- matrix(kronecker(X, matrix(1, nrow = nspp, ncol = 1)), nrow =
nspp * nsite, ncol = 1)

site <- matrix(kronecker(1:nsite, matrix(1, nrow = nspp, ncol =
1)), nrow = nspp * nsite, ncol = 1)
sp <- matrix(kronecker(matrix(1, nrow = nsite, ncol = 1), 1:nspp),
nrow = nspp * nsite, ncol = 1)

dat <- data.frame(Y = YY, X = XX, site = as.factor(site), sp = as.factor(sp))

# Format input and perform communityPGLMM()
# set up random effects

# random intercept with species independent
```

```
re.1 <- list(1, sp = dat$sp, covar = diag(nspp))

# random intercept with species showing phylogenetic covariances
re.2 <- list(1, sp = dat$sp, covar = Vphy)

# random slope with species independent
re.3 <- list(dat$X, sp = dat$sp, covar = diag(nspp))

# random slope with species showing phylogenetic covariances
re.4 <- list(dat$X, sp = dat$sp, covar = Vphy)

# random effect for site
re.site <- list(1, site = dat$site, covar = diag(nsite))

simple <- communityPGLMM(Y ~ X, data = dat, family = "binomial", sp
= dat$sp, site = dat$site, random.effects = list(re.site),
REML=TRUE, verbose=FALSE)

# The rest of these tests are not run to save CRAN server time;
# - please take a look at them because they're *very* useful!
## Not run:
z.binary <- communityPGLMM(Y ~ X, data = dat, family = "binomial",
sp = dat$sp, site = dat$site, random.effects = list(re.1, re.2,
re.3, re.4), REML = TRUE, verbose = FALSE)

# output results
z.binary
plot(z.binary)

# test statistical significance of the phylogenetic random effect
# on species slopes using a likelihood ratio test
communityPGLMM.binary.LRT(z.binary, re.number = 4)$Pr

# extract the predicted values of Y
communityPGLMM.predicted.values(z.binary, show.plot = TRUE)

# examine the structure of the overall covariance matrix
communityPGLMM.matrix.structure(Y ~ X, data = dat, family =
"binomial", sp = dat$sp, site = dat$site, random.effects =
list(re.1, re.2, re.3, re.4))

# look at the structure of re.1
communityPGLMM.matrix.structure(Y ~ X, data = dat, family =
"binomial", sp = dat$sp, site = dat$site, random.effects =
list(re.1))

# compare results to glmer() when the model contains no
# phylogenetic covariance among species; the results should be
# similar.
communityPGLMM(Y ~ X, data = dat, family = "binomial", sp = dat$sp,
site = dat$site, random.effects = list(re.1, re.3), REML = FALSE,
verbose = FALSE)
```

```
# lmer
if(require(lme4)){
summary(glmer(Y ~ X + (1 | sp) + (0 + X | sp), data=dat, family =
"binomial"))

# compare results to lmer() when the model contains no phylogenetic
# covariance among species; the results should be similar.
communityPGLMM(Y ~ X, data = dat, family = "gaussian", sp = dat$sp,
site = dat$site, random.effects = list(re.1, re.3), REML = FALSE,
verbose = FALSE)

# lmer
summary(lmer(Y ~ X + (1 | sp) + (0 + X | sp), data=dat, REML = FALSE))
}

## End(Not run)
```

---

phy.build *Build a novel phylogeny from existing data (based on phyloGenerator)*

Description

Build a novel phylogeny from existing data (based on phyloGenerator)

Usage

```
congeneric.merge(tree, species, split = "_", ...)

bind.replace(backbone, donor, replacing.tip.label, donor.length = NA)
```

Arguments

tree	phylo phylogeny to have those species inserted into it
species	vector of species names to be bound into the tree if missing from it
split	the character that splits genus and species names in your phylogeny. Default is _, i.e. Quercus_robur.
...	ignored
backbone	backbone phylogeny (phylo) into which the donor is to be bound
donor	phylogeny (phylo) to bound into the backbone phylogeny
replacing.tip.label	the species in the donor phylogeny that's being replaced by the donor phylogeny
donor.length	how deep the donor phylogeny should be cut into the backbone phylogeny. If NA (default), then the bladj algorithm is followed (or, in plain English, it's put half-way along the branch)

Details

`congeneric.merge` Binds missing species into a phylogeny by replacing all members of the clade it belongs to with a polytomy. Assumes the `tip.labels` represent Latin binomials, split by the `split` argument. This code was originally shipped with phyloGenerator - this is the `merge` method in that program.

`bind.replace` Binds a phylogeny (donor) into a bigger phylogeny ('backbone'); useful if you're building a phylogeny a la Phylomatic. A version of this R code was shipped with phyloGenerator (Pearse & Purvis 2013). This is really an internal function for `congeneric.merge`, but hopefully it's of some use to you!

Value

`phylo` phylogeny

phylogeny (`phylo`)

Author(s)

Will Pearse

Will Pearse

References

Pearse W.D. & Purvis A. phyloGenerator: an automated phylogeny generation tool for ecologists. Methods in Ecology and Evolution 4(7): 692–698.

Examples

```
tree <- read.tree(text="((a_a:1,b_b:1):1, c_c:2):1;")
tree <- congeneric.merge(tree, c("a_nother", "a_gain", "b_sharp"))
```

`phy.signal` *Calculate phylogenetic 'signal'*

Description

Calculate phylogenetic 'signal'

Usage

```
phy.signal(data, method = c("lambda", "delta", "kappa", "blom.k"))
```

Arguments

`data`	`comparative.comm` object
`method`	what kind of signal to calculate, one of Pagel's λ (default), δ, and κ, or Blomberg's K.

Details

Phylogenetic 'signal' is one of those concepts that is said a lot in community ecology, but some do not full consider its meaning. Think carefully before rushing to report a value whether: (1) it makes sense to assess phylogenetic 'signal' in your datasets, and (2) what the phrase 'phylogenetic signal' actually means. This code makes use of `caper::pgls` to get estimates of fit; alternatives that offer more flexibility exist (see below).

Value

Named numeric vector, where each element is a trait or community.

Author(s)

Will Pearse, Jeannine Cavender-Bares

References

Blomberg S.P., Garland T. & Ives A.R. Testing for phylogenetic signal in comparative data: behavioral traits are more labile. Evolution 57(4): 717–745.

R. P. Freckleton, P. H. Harvey, and M. Pagel. Phylogenetic analysis and comparative data: A test and review of evidence. American Naturalist, 160:712-726, 2002.

Mark Pagel (1999) Inferring the historical patterns of biological evolution. Nature 6756(401): 877–884.

See Also

fitContinuous fitDiscrete pgls phylosignal

Examples

```
data(laja)
data <- comparative.comm(invert.tree, river.sites, invert.traits)
phy.signal(data, "lambda")
```

plot.comparative.comm *Dot-plots of community presence/absence or abundance*

Description

Dot-plots of community presence/absence or abundance

Usage

```
## S3 method for class 'comparative.comm'
plot(x, sites = NULL, abundance = FALSE,
  pch = 20, dot.cex = NULL, site.col = "black", fraction = 3,
  x.increment = NULL, show.tip.label = FALSE, ...)
```

Arguments

x	comparative.comm object
sites	names of sites to plot (default: all); see examples
abundance	make size proportional to species abundance (default: FALSE)
pch	plotting character to be used for sites (see pch)
dot.cex	function to determine point size; see examples, this isn't as terrible-sounding as it seems.
site.col	colours to use when plotting sites; if not same length as number of sites, only the first element is used (no recycling)
fraction	fraction of plot window to be taken up with phylogeny; e.g., 3 (default) means phylogeny is 1/3 of plot
x.increment	specify exact spacing of points along plot; see examples
show.tip.label	whether to plot species names on phylogeney (default: FALSE)
...	additional arguments passed to plotting functions

Details

Take a look at the examples: this is (hopefully!) a lot more straightforward than it might seem. Getting the right spacing of dots on the phylogeny may take some playing around with the `fraction` and `x.increment` arguments. It may seem a little strange to set point size using a function, however, this gives you much more flexibility and the ability to (usefully) transform your data.

Value

List containing plot.phylo information, as well as the used x.adj values (compare with your `x.increment`)

Author(s)

Will Pearse, Matt Helmus

See Also

comparative.comm traitgram.cc

Examples

```
data(laja)
data <- comparative.comm(invert.tree, river.sites, invert.traits)
plot(data)
plot(data, sites=c("AT", "BP"), fraction=1.5)
settings <- plot(data, sites=c("AT", "BP"), site.col=rainbow(2), fraction=1.5)
plot(data, sites=c("AT", "BP"), site.col=rainbow(2),
fraction=1.2, x.increment=settings$x.increment/4)
#dot.cex isn't as scary as it sounds...
plot(data, site.col=rainbow(2), fraction=1.2, abundance=TRUE, dot.cex=sqrt)
#...or other trivial variants...
abund.sqrt <- function(x) ifelse(x>0, sqrt(x), 0)
plot(data, sites=c("AT", "BP"), site.col=rainbow(2), fraction=1.2,
x.increment=settings$x.increment/4, abundance=TRUE, dot.cex=abund.sqrt)
plot(data, sites=c("AT", "BP"), site.col=rainbow(2), fraction=1.2,
x.increment=settings$x.increment/4, abundance=TRUE, dot.cex=function(x) sqrt(x))
```

---

scape *Simulate phylogenetic community structure across a landscape*

Description

scape simulates communities that are phylogenetically structured

Usage

```
scape(tree, scape.size = 10, g.center = 1, g.range = 1, g.repulse = 1,
  wd.all = 150, signal.center = TRUE, signal.range = TRUE,
  same.range = TRUE, repulse = TRUE, center.scale = 1, range.scale = 1,
  repulse.scale = 1, site.stoch.scale = 0.5, sd.center = 1,
  sd.range = 1, rho = NULL, th = 8)
```

Arguments

tree	phylo object
scape.size	edge dimension of square landscape
g.center	strength of phylogenetic signal in species range centers
g.range	strength of phylogenetic signal in species range sizes
g.repulse	strength of phylogenetic repulsion
wd.all	niche width, larger values simulate broader range sizes
signal.center	simulate with phylosignal in range centers
signal.range	simulate with phylosignal in range size

same.range	make all range sizes equal
repulse	include phylogenetic repulsion in range centers
center.scale	adjust strength of phylogenetic attraction in range centers independent of g.center
range.scale	adjust strength of phylogenetic signal in range size independent of g.range
repulse.scale	adjust strength of phylogenetic repulsion independent of g.repulse
site.stoch.scale	adjust strength of random variation in species richness across sites
sd.center	sd in rnorm for the range centers, increase to get more variation in center values across species
sd.range	sd rnorm for the range sizes, increase to get more variation in range sizes across gradients
rho	Grafen branch adjustment of phylogenetic tree see corGrafen
th	probability threshold 10^-th above which species are considered present at a site

Details

Simulates a landscape with species (i.e., tree tips) distributions dependent on a supplied phylogenetic tree. The amount and type of structure is determened by the signal parameters g.center, g.range and g.repulse. Parameters are based on an Ornstein-Uhlenbeck model of evolution with stabilizing selection. Values of g=1 indicate no stabilizing selection and correspond to the Brownian motion model of evolution; 0<g<1 represents stabilizing selection; and g>1 corresponds to disruptive selection where phylogenetic signal for the supplied tree is amplified. See corBlomberg. Communities are simulated along two gradients where the positions along those gradients, g.center and range sizes g.range, can exhibit phylogenetic signal. Phylogenetic attraction is simulated in the g.center paramter, while repulsion in g.repulse. Both can be exhibited such that closly related species are generally found with similar range centers (phylogenetic attraction) but just not at the same site (phylogenetic repulsion). The function then returns probabilities of of each species across sites and the presence and absences of species based a supplied threshold, th, which can be increased to obtain more species at sites and thus increase average site species richness.

Value

cc	comparative.comm object with presence/absence results of simulations. The site names are the row.columns of the cells in the original grid cells that made up the data, and these co-ordinates are also given in the env slot of the object.
X.joint	full probabilities of species at sites, used to construct cc
X1	probabilities of species along gradient 1
X2	probabilities of species along gradient 2
sppXs	full probabilities of each species as an array arranged in a scape.size-by-scape.size matrix
V.phylo	initial phylogenetic covariance matrix from tree
V.phylo.rho	phylogenetic covariance matrix from tree scaled by Grafen if rho is provided
V.center	scaled by g.center phylo covariance matrix used in the simulations
V.range	scaled by g.range phylo covariance matrix used in the simulations

V.repulse	scaled by g.repulse phylo covariance matrix used in the simulations
bspp1	species optima for gradient 1
bspp2	species optima for gradient 2
u	the env gradients values for the two gradients
wd	the denominator for species ranges

Author(s)

M.R. Helmus, cosmetic changes by Will Pearse

References

Helmus M.R. & Ives A.R. (2012). Phylogenetic diversity area curves. Ecology, 93, S31-S43.

See Also

eco.scape sim.phy sim.meta

Examples

```
#Create balanced tree with equal branch-lengths (signal in centers)
tree <- stree(8,type="balanced")
tree$edge.length <- rep(1, nrow(tree$edge))
tree$root <- 1
kk <- scape(tree, scape.size=100, g.center=100, g.range=1, g.repulse=1, wd.all=150,
   signal.center=TRUE, signal.range=FALSE, same.range=FALSE, repulse=FALSE,center.scale = 1,
    range.scale = 1, repulse.scale = 1, site.stoch.scale = 0, sd.center=3, sd.range=1,
    rho=NULL, th=20)

#Make some plots
par(mfrow=c(1,Ntip(tree)),mar=c(.1,.1,.1,.1))
for(j in seq_along(tree$tip.label))
    image(t(1 - kk$sppXs[,,j]/max(kk$sppXs[,,j])), xlab = "",
              ylab = "",main = "",axes=FALSE, col=grey.colors(10))

par(mfrow=c(2,1))
matplot((kk$X1), type = "l", xlab="gradient",ylab = "probability",
main = "Gradient 1",col=rainbow(dim(kk$X1)[2]),lty=1)
matplot((kk$X2), type = "l", xlab="gradient",ylab = "probability",
main = "Gradient 2",col=rainbow(dim(kk$X2)[2]),lty=1)

plot(x=seq_along(sites(kk$cc)),y = rowSums(comm(kk$cc)), main = "SR",type = "l")
cor(kk$X1)
```

`sim.meta` *Simulate a meta-community (and its phylogeny)*

Description

`sim.meta.comm` simulates species moving through a metacommunity. At each time-step each cell's next abundance for each species is `env.quality - current.abundance + stochastic`, and a species gets as many chances to migrate in each time-step as it has cells (the same cell could migrate multiple times). I use a Poisson for everything because I don't want half-species (these are individuals), and keeping everything in Poisson makes it easier to compare the relative rates of everything.

`sim.meta.phy.comm` simulate a (sort of) meta-community

Usage

```
sim.meta.comm(size = 10, n.spp = 8, timesteps = 10, p.migrate = 0.05,
  env.lam = 10, abund.lam = 5, stoch.lam = 1)

sim.meta.phy.comm(size = 10, n.spp = 8, timesteps = 10, p.migrate = 0.3,
  env.lam = 10, abund.lam = 5, stoch.lam = 1, p.speciate = 0.05)
```

Arguments

`size`	the length and width of the meta-community in grid cells
`n.spp`	number of species
`timesteps`	number of time-steps (each discrete)
`p.migrate`	probability that a group of species in each grid cell will migrate to another grid cell each timestep (i.e., 10 cells occuped by a species –> 10*`p.migrate` chance of migration)
`env.lam`	λ value for Poisson distribution used to distribute environmental quality; essentially the carrying capacity (for each species separately) for that cell
`abund.lam`	λ value for Poisson distribution used to distribute initial abundances and abundance after migration
`stoch.lam`	λ value for Poisson distribution of noise added to the next step abundance calculation. With equal chance, this is taken as either a positive or a negative number (see details if you're confused as to why this is Poisson!)
`p.speciate`	probabilty that, at each timestep, a species will speciate. A species can only speciate, migrate, or reproduce if it has individuals!

Details

`sim.meta.phy.comm` As above, but with a simulation of phylogeny as well - there are no additional extinction parameters, since extinction happens as a natural consequence of ecological interactions.

Value

For `sim.meta.comm` a list with a species-site matrix as the first slot, and the environment as the second. Rownames of the site-species are the List with the x and y co-ordinates of the simulation grid pasted together; colnames are arbitrary species names. `sim.meta.comm`, a `comparative.comm` object (since we have now simulated a phylogeny), with the same naming convention for the site names. phylogeny.

`sim.meta.phy.comm` `comparative.comm` object that describes the data; note that the rownames of the community object refer to the `row.column` of the data in the simulated grid assemblages.

Note

`scape` is a much more sophisticated simulation of the biogeography, but requires you to supply a phylogeny. You pays your money, you makes your choice.

Author(s)

Will Pearse

Will Pearse

See Also

`sim.phy` `scape`

`sim.phy` *Simulate phylogenies*

Description

Simulate phylogenies under pure birth/death or as a function of trait evolution

Usage

```
sim.bd.phy(speciate = 0.1, extinction = 0.025, time.steps = 20)

sim.bd.tr.phy(speciate = 0.1, extinction = 0.025, time.steps = 20,
  tr.range = c(0, 1), sp.tr = 2, ext.tr = 1, tr.walk = 0.2,
  tr.wrap = TRUE)

edge2phylo(edge, s, e = numeric(0), el = NA, t = NULL)
```

Arguments

`speciate`	probability each species will speciate in each time-step (0-1)
`extinction`	probability each species will go extinct in each time-step (0-1)
`time.steps`	number of time-steps for simulation
`tr.range`	vector of length two specifying boundaries for trait values (see notes); initial two species will be at the 25th and 75th percentiles of this space. See also `tr.wrap`
`sp.tr`	speciation rate's interaction with the minimum distance between a species and the species most similar to it (see details)
`ext.tr`	extinction rate's interaction with the minimum distance between a species and the species most similar to it (see details)
`tr.walk`	at each time-step a species not undergoing speciation or extinction has its trait value drawn from a distribution centered at its current value and with a standard deviation set by this value. I.e., this is the rate of the Brownian motion trait evolution.
`tr.wrap`	whether to force species' trait values to stay within the boundary defined by `tr.range`; default TRUE.
`edge`	a two-column matrix where the first column is the start node, the second the destination, as in `phylo$edge`
`s`	which of the rows in the edge matrix represent extant species
`e`	which of the tips in the edge matrix are extinct (DEFAULT: empty vector, i.e., none)
`el`	a vector to be used to give edge.length to the phylogeny (default NA, i.e., none)
`t`	if given (default NA), a vector to be used for traits (`$traits` slot) in the phylogeny

Details

`sim.bd.tree` simulates a pure birth/death speciation model. There are two important things to note: (1) speciation is randomised before extinction, and only one thing can happen to a lineage per timestep. (2) This code works well for my purposes, but absurd parameter values can cause the function to crash.

`sim.bd.tr.tree` is an extension of `sim.bd.tree`, and all its caveats apply to it. It additionally simulated the evolution of a trait under Brownain motion (`tr.walk`). Species' speciation/extinction rates change depending on whether they have a trait value similar to other species (`sp.tr`, `ext.tr`). When a speciation event happens, the two daughters split evenly about the ancestor's trait value, taking values half-way to whatever the nearest species' value is. To be precise:

$$p(speciate)_i = speciate_i + sp.tr \times min(traitdistance)$$

,

$$p(extinct)_i = exinction_i + ext.tr \times min(traitdistance)$$

, where

$$i$$

denotes each species.

edge2phylo is an internal function for the sim.phy and sim.meta function families, which may be of use to you. Check those functions' code for examples of use.

These functions are closely related to sim.meta; the latter are extensions that simulate meta-community structure at the same time.

Value

phylo object with random tip.labels, and trait values if using sim.br.tr.tree.

Author(s)

Will Pearse

Will Pearse

Will Pearse

See Also

sim.meta scape

Examples

```
tree <- sim.bd.phy(0.1, 0, 10)
plot(tree)
```

---

trait.asm *Produces simulated communities based on species attributes*

Description

trait.asm calculates phylogenetic biodiversity metrics

Usage

```
trait.asm(a, m = 1000, meanSR = NULL, interval = c(0.001, 10),
  exponential = TRUE, Pscale = 1)
```

Arguments

a	species attributes (e.g., traits like body size)
m	number of communities to be simulated
meanSR	target mean species richness across simulated communities
interval	adjust to obtain meanSR
exponential	use the exponential distribution when simulating communities
Pscale	adjust this value when not using the exponential distribution in order to scale the species richnesses in the simulated communities

Details

Simulates a set of communties based on the supplied attribute (trait) where larger values make it more likely for species to be in the communities.

Value

`Y` presence/absence matrix

`P` probabilities

`a` the supplied trait

`exponential` if the exponential distribution was used

`meanSR` supplied `meanSR` value

`std` estimated sd

Author(s)

M.R. Helmus

References

Helmus M., Mercado-Silva N. & Vander Zanden M.J. (2013). Subsidies to predators, apparent competition and the phylogenetic structure of prey communities. Oecologia, 173, 997-1007.

Examples

```
## Not run:
 data(laja)
 trait.asm(laja$fish.pref)

## End(Not run)
```

`traitgram.cc` *Traitgram for comparative community object*

Description

`traitgram.cc` A wrapper for the `traitgram` function in the `picante` package.

`princompOne` A very soft wrapper for `princomp`

Usage

```
traitgram.cc(object, trait, moreArgs = NULL, ...)

princompOne(x, ...)
```

Arguments

object	A comparative.comm object.
trait	Which trait to plot. If missing, use the first trait. If a positive numeric vector of length one, use the as.integer(trait)th trait. If a numeric vector, use it instead of the trait data in object. If a character vector of length one, use the trait with that name. If a function pass the trait data frame through that function and use the result (princompOne is a wrapper). If an expression, evaluate that expression in the environment of the trait data and use the result. If a character vector, then convert to an expression and evaluate in the environment of the trait data and use the result.
moreArgs	List of more arguments to pass on to trait (if its a function).
...	Additional arguments to be passed on to traitgram or prcomp for traitgram.cc and princomOne respectively.
x	A matrix-like object

Value

traitgram.cc: see traitgram

princompOne: the first axis of a PCA

Index

∗Topic **datasets**
laja, 20
.aed (pez.metrics), 28
.colless (pez.metrics), 28
.comdist (pez.metrics), 28
.d (pez.metrics), 28
.delta (pez.metrics), 28
.dist.fd (pez.metrics), 28
.eaed (pez.metrics), 28
.eed (pez.metrics), 28
.eigen.sum (pez.metrics), 28
.gamma (pez.metrics), 28
.haed (pez.metrics), 28
.hed (pez.metrics), 28
.iac (pez.metrics), 28
.innd (pez.metrics), 28
.kappa (pez.metrics), 28
.lambda (pez.metrics), 28
.metric.null, *19*
.metric.null (generic.metrics), 18
.mipd (pez.metrics), 28
.mntd (pez.metrics), 28
.mpd (pez.metrics), 28
.pae (pez.metrics), 28
.pcd (pez.metrics), 28
.pd (pez.metrics), 28
.phylo.entropy (pez.metrics), 28
.phylosor (pez.metrics), 28
.pse (pez.metrics), 28
.psr (pez.metrics), 28
.psv (pez.metrics), 28
.rao (pez.metrics), 28
.scheiner (pez.metrics), 28
.ses, *18*
.ses (generic.metrics), 18
.ses.innd (pez.metrics), 28
.ses.mipd (pez.metrics), 28
.ses.mntd (pez.metrics), 28
.ses.mpd (pez.metrics), 28
.simpson.phylogenetic (pez.metrics), 28
.sqrt.phy, *30*
.sqrt.phy (pez.metrics), 28
.taxon (pez.metrics), 28
.unifrac (pez.metrics), 28
[.comparative.comm (cc.manip), 2

ape, *30*
apTreeshape, *30*
as.data.frame.comparative.comm, *36*, *39*
as.data.frame.comparative.comm (cc.manip), 2
assemblage.phylogenies (cc.manip), 2

bind.replace (phy.build), 44

caper, *30*
cc.manip, 2, *5*, *6*
character, *56*
comm (cc.manip), 2
comm.dist, *14*
comm.dist (dist.xxx), 8
comm<- (cc.manip), 2
communityPGLMM (pglmm), 34
comparative.comm, *2*, *4*, 5, *7*, *11*, *14*, *16*, *18*, *21*, *22*, *25*, *28*, *30–32*, *39*, *46–49*, *52*, *56*
comparative.data, *4–6*
ConDivSim, 6
congeneric.merge (phy.build), 44
corBlomberg, *10*, *49*
corGrafen, *11*, *49*

data, *14*
data.frame, *36*
dispersion metrics, *30*
dist.func.default (dist.xxx), 8
dist.xxx, 8
drop.tip, *9*, *10*
drop_tip, 9

eco.env.regression, *14*
eco.env.regression
(eco.xxx.regression), 13
eco.phy.regression, *14*
eco.phy.regression
(eco.xxx.regression), 13
eco.scape, 10, *50*
eco.trait.regression, *14*
eco.trait.regression
(eco.xxx.regression), 13
eco.xxx.regression, 13, *17*
eco.xxx.regression.list, 15
edge2phylo (sim.phy), 52
env (cc.manip), 2
env<- (cc.manip), 2
expression, *56*
extract.clade, *10*

factor, *36*
FD, *30*
fingerprint.regression, *14, 15*, 16, *21*
funct.phylo.dist, *7*
funct.phylo.dist (dist.xxx), 8
function, *56*

generic.metrics, *18*, 18, *19*, *28*
generic.null, *18, 19*
generic.null (generic.metrics), 18
glm, *36*

invert.traits (laja), 20
invert.tree (laja), 20

laja, 20
length, *56*
list, *36*
lm, *14*, *16*, *36*
lmer, *35, 36*

mantel, *14*, *16*
missing, *56*

numeric, *56*

optim, *36*, *39*

package-pez (pez), 21
pch, *47*
pez, 21
pez-package (pez), 21
pez.dispersion, *19*, *21*, 21, *25*, *28*, *30*, *34*
pez.dissimilarity, *19*, *21*, *23*, 23, *28*, *30*, *34*
pez.evenness, *19*, *21*, *23*, *25*, 25, *28*, *30*, *33, 34*
pez.metrics, *18*, 28
pez.shape, *19*, *21*, *23*, *25*, *27, 28*, *30*, 32
pglmm, 34
pgls, *46*
phy (cc.manip), 2
phy.build, 44
phy.signal, *15–17*, 45
phy<- (cc.manip), 2
phylo, *5*, *10*, *20*, *44, 45*, *48*, *53, 54*
phylo.dist (dist.xxx), 8
pianka.dist (dist.xxx), 8
picante, *14*, *16*, *18*, *22*, *30*
plot.communityPGLMM (pglmm), 34
plot.comparative.comm, *6*, 47
plot.eco.xxx.regression
(eco.xxx.regression), 13
plot.eco.xxx.regression.list
(eco.xxx.regression.list), 15
plot.fingerprint.regression
(fingerprint.regression), 16
princomp, *55*
princompOne, *56*
princompOne (traitgram.cc), 55
print.comparative.comm
(comparative.comm), 5
print.default, *37*
print.eco.xxx.regression
(eco.xxx.regression), 13
print.eco.xxx.regression.list
(eco.xxx.regression.list), 15
print.fingerprint.regression
(fingerprint.regression), 16

river.env (laja), 20
river.sites (laja), 20
rnorm, *11*, *49*
rq, *14*, *16*

scape, *7*, *10*, *12*, *21*, 48, *52*
sim.bd.phy (sim.phy), 52
sim.bd.tr.phy (sim.phy), 52
sim.meta, *12*, *50*, 51, *54*
sim.meta.comm, *21*
sim.phy, *7*, *12*, *50*, *52*, 52, *54*
sites (cc.manip), 2

sites<- (cc.manip), 2
species (cc.manip), 2
species<- (cc.manip), 2
summary.communityPGLMM (pglmm), 34
summary.eco.xxx.regression
(eco.xxx.regression), 13
summary.eco.xxx.regression.list
(eco.xxx.regression.list), 15
summary.fingerprint.regression
(fingerprint.regression), 16

traceback, *14*, *17*
trait.asm, *19*, 54
trait.names (cc.manip), 2
traitgram, *55*, *56*
traitgram.cc, *48*, 55
traits (cc.manip), 2
traits.dist (dist.xxx), 8
traits<- (cc.manip), 2
tree (cc.manip), 2
tree<- (cc.manip), 2

vegan, *4*, *5*

within.comparative.comm (cc.manip), 2