Package ‘ecolottery’

April 14, 2025

Type Package

Title Coalescent-Based Simulation of Ecological Communities

Version 1.0.1

URL `https://github.com/frmunoz/ecolottery`

BugReports `https://github.com/frmunoz/ecolottery/issues`

Depends R (>= 3.0.2)

Imports abc, stats, graphics, ggplot2, grDevices, parallel

Suggests ape, knitr, picante, rmarkdown, testthat, vegan

Description Coalescent-Based Simulation of Ecological Communities as proposed by Munoz et al. (2018) <doi:10.1111/2041-210X.12918>. The package includes a tool for estimating parameters of community assembly by using Approximate Bayesian Computation.

License GPL (>= 2)

Encoding UTF-8

VignetteBuilder knitr

RoxygenNote 6.0.1

NeedsCompilation no

Author François Munoz [aut, cre],
Matthias Grenié [aut],
Pierre Denelle [aut],
Adrien Taudière [ctb],
Fabien Laroche [ctb],
Caroline Tucker [ctb],
Cyrille Violle [ctb]

Maintainer François Munoz <`francois.munoz@hotmail.fr`>

Repository CRAN

Date/Publication 2025-04-14 13:00:01 UTC

Contents

`ecolottery-package` *Coalescent-Based Simulation of Ecological Communities*

Description

Coalescent-Based Simulation of Ecological Communities as proposed by Munoz et al. (2018) <doi:10.1111/2041-210X.12918>. The package includes a tool for estimating parameters of community assembly by using Approximate Bayesian Computation.

Details

The DESCRIPTION file:

Package:	ecolottery
Type:	Package
Title:	Coalescent-Based Simulation of Ecological Communities
Version:	1.0.1
Authors@R:	c(person("François", "Munoz", role = c("aut", "cre"), email = "francois.munoz@hotmail.fr"), person("M
URL:	https://github.com/frmunoz/ecolottery
BugReports:	https://github.com/frmunoz/ecolottery/issues
Depends:	R (>= 3.0.2)
Imports:	abc, stats, graphics, ggplot2, grDevices, parallel
Suggests:	ape, knitr, picante, rmarkdown, testthat, vegan
Description:	Coalescent-Based Simulation of Ecological Communities as proposed by Munoz et al. (2018) <doi:10.1
License:	GPL (>= 2)
Encoding:	UTF-8
VignetteBuilder:	knitr
RoxygenNote:	6.0.1
NeedsCompilation:	no
Packaged:	2025-04-13 10:24:20 UTC; Munoz
Author:	François Munoz [aut, cre], Matthias Grenié [aut], Pierre Denelle [aut], Adrien Taudière [ctb], Fabien La
Maintainer:	François Munoz <francois.munoz@hotmail.fr>
Repository:	CRAN
Date/Publication:	2017-07-03 11:01:29 UTC

Index of help topics:

`abund`	`Compute absolute and relative abundances in the local community and the reference pool`
`coalesc`	`Coalescent-based simulation of ecological communities undergoing both neutral and niche-base dynamics`
`coalesc_abc`	`Estimation of neutral and non-neutral parameters of community assembly using Approximate Bayesian Computation (ABC)`
`ecolottery-package`	`Coalescent-Based Simulation of Ecological Communities`
`forward`	`Simulation of neutral and niche-based community dynamics forward in time`
`plot_comm`	`Regional vs. Local trait distributions of abundances`
`tcor`	`Generates Correlated Traits`

Further information is available in the following vignettes:

`Barro_Colorado`	Example of coalesc_abc() use with Barro-Colorado dataset (source, pdf)
`coalesc_vignette`	Introductory vignette for use of ‘ecolottery‘ (source, pdf)

Two basic functions: `coalesc` for coalescent-based simulation, and `forward` for forward-in-time simulation

Author(s)

François Munoz [aut, cre], Matthias Grenié [aut], Pierre Denelle [aut], Adrien Taudière [ctb], Fabien Laroche [ctb], Caroline Tucker [ctb], Cyrille Violle [ctb]

Maintainer: François Munoz <francois.munoz@hotmail.fr>

References

Hurtt, G. C. and S. W. Pacala (1995). "The consequences of recruitment limitation: reconciling chance, history and competitive differences between plants." Journal of Theoretical Biology 176(1): 1-12.

Hubbell, S. P. (2001). "The Unified Neutral Theory of Biodiversity". Princeton University Press.

Gravel, D., C. D. Canham, M. Beaudet and C. Messier (2006). "Reconciling niche and neutrality: the continuum hypothesis." Ecology Letters 9(4): 399-409.

Munoz, F., P. Couteron, B. R. Ramesh and R. S. Etienne (2007). "Estimating parameters of neutral communities: from one Single Large to Several Small samples." Ecology 88(10): 2482-2488.

Munoz, F., B. R. Ramesh and P. Couteron (2014). "How do habitat filtering and niche conservatism affect community composition at different taxonomic resolutions?" Ecology 95(8): 2179-2191.

Examples

```
## Coalescent-based simulation of stabilizing habitat filtering around
## t = 0.5
J <- 100; theta <- 50; m <- 0.5;
comm <- coalesc(J, m, theta, filt = function(x) 0.5 - abs(0.5 - x))
plot_comm(comm)

## Forward-in-time simulation of stabilizing habitat filtering around
## t = 0.5, over 100 time steps

# A regional pool including 100 species each including 10 individuals
pool <- sort(rep(as.character(1:100), 10))

# Initial community composed of 10 species each including 10 individuals,
# with trait information for niche-based dynamics
initial <- data.frame(sp = sort(rep(as.character(1:10), 10)),
                      trait = runif(100))
final <- forward(initial = initial, prob = 0.5, gens = 100, pool = pool,
                 filt = function(x) 0.5 - abs(0.5 - x))
plot_comm(final)
```

---

abund — *Compute absolute and relative abundances in the local community and the reference pool*

Description

Compute the abundances and relative abundances of species in simulated communities and in the corresponding species pools. The input must be an output of either `coalesc` or the `forward` functions.

Usage

```
abund(x)
```

Arguments

`x`	a list including the species pool composition (`x$pool`) and the local community composition (`x$com`)

Value

`pool`	species abundances and relative abundances in the reference pool
`com`	species abundances and relative abundances in the local community

Author(s)

F. Munoz, P. Denelle and M. Grenie

Examples

```
# Simulation of a neutral community including 500 individuals
J <- 500; theta <- 50; m <- 0.05;
comm1a <- coalesc(J, m, theta)
abund1a <- abund(comm1a)

# Log-series distribution of regional abundances
fit <- vegan::fisherfit(abund1a$pool$ab)
freq <- as.numeric(names(fit$fisher))
plot(log(freq), fit$fisher,
     xlab = "Frequency (log)",
     ylab = "Species", type = "n")
rect(log(freq - 0.5), 0, log(freq + 0.5), fit$fisher, col="skyblue")

alpha <- fit$estimate
k <- fit$nuisance

curve(alpha * k^exp(x) / exp(x), log(0.5), max(log(freq)),
      col = "red", lwd = 2, add = TRUE)

# Relationship between local and regional abundances
par(mfrow=c(1, 2))
plot(abund1a$pool[rownames(abund1a$com), "relab"],
  abund1a$com$relab,
  main = "m = 0.05",
  xlab = "Regional abundance",
  ylab = "Local abundance",
  log = "xy")
abline(0,1)

# With higher immigration rate
m <- 0.95
comm1b <- coalesc(J, m, theta)
abund1b <- abund(comm1b)
plot(abund1b$pool[rownames(abund1b$com),"relab"],
  abund1b$com$relab,
  main = "m = 0.95",
  xlab = "Regional abundance",
  ylab = "Local abundance",
  log = "xy")
abline(0,1)
```

`coalesc` *Coalescent-based simulation of ecological communities undergoing both neutral and niche-base dynamics*

Description

Simulates the composition of a community based on immigration from a regional pool, habitat filtering depending on local environment and species traits, and local birth-death stochastic dynamics.

Usage

```
coalesc(J, m = 1, theta = NULL, filt = NULL, pool = NULL, traits = NULL,
        Jpool = 50 * J, verbose = FALSE)
```

Arguments

`J`	number of individuals in the local community.
`m`	migration rate (if `m = 1` the community is a subsample of the regional pool).
`theta`	parameter of neutral dynamics in the regional pool (used only if `pool=NULL`), it is the "fundamental biodiversity number" (θ).
`filt`	a function representing the effect of local habitat filtering. For a given trait value `t`, `filt(t)` represents the probability that an individual with trait `t` enters the local community. If `filt = NULL`, `coalesc()` provides a neutral community.
`pool`	the regional pool of species providing immigrants to the local community. It should include the label of individual on first column, and of its species on second column. If `pool = NULL`, the pool is simulated as a metacommunity at speciation-drift equilibrium, based on parameter `theta`. The provided `pool` can contain trait values for each individuals in a third column.
`traits`	a matrix or data.frame including one or several traits on columns. A unique trait value is assigned to each species in the regional pool. If `traits = NULL`, a random trait value is given to species of the regional pool, from a uniform distribution between 0 and 1.
`Jpool`	if `pool = NULL`, it is the number of individuals to be simulated in the regional pool.
`verbose`	if `verbose = TRUE`, functions returns a lot of outputs about parameters, species pool and environmental filter.

Details

Coalescent-based simulation of a community of size `J`. This generic function can simulate a neutral community (if `filt = NULL`) or a community undergoing both neutral and niche-based dynamics. In the latter case, `filt(t)` represents the relative ability of immigrants with trait values `t` in the regional pool to enter the community.

Value

`com`	a data.frame of simulated individuals, with the label of ancestor individual in the regional pool on first column (as in first column of pool), species label on second column (as in second column of pool), and species trait (as in third column of pool). **Not provided if `m = 1` and `filt = NULL`: in this case the function provides a sample of the regional pool.**
`pool`	a data.frame of the individuals of the regional source pool, with the label of ancestor individual in the regional pool on first column (as in first column of input `pool`), species label on second column (as in second column of input `pool`), and species trait (as in third column of input `pool`).

Author(s)

F. Munoz

References

Hurtt, G. C. and S. W. Pacala (1995). "The consequences of recruitment limitation: reconciling chance, history and competitive differences between plants." Journal of Theoretical Biology 176(1): 1-12.

Gravel, D., C. D. Canham, M. Beaudet and C. Messier (2006). "Reconciling niche and neutrality: the continuum hypothesis." Ecology Letters 9(4): 399-409.

Munoz, F., P. Couteron, B. R. Ramesh and R. S. Etienne (2007). "Estimating parameters of neutral communities: from one Single Large to Several Small samples." Ecology 88(10): 2482-2488.

Munoz, F., B. R. Ramesh and P. Couteron (2014). "How do habitat filtering and niche conservatism affect community composition at different taxonomic resolutions?" Ecology 95(8): 2179-2191.

Examples

```
# Simulation of a neutral community including 100 individuals
J <- 500; theta <- 50; m <- 0.1
comm1 <- coalesc(J, m, theta)
# Regional and local trait distributions
plot_comm(comm1)

# Define a regional pool of species with equal abundances
pool <- cbind(1:10000, rep(1:500, 20), rep(NA, 10000))
# Uniform distribution of trait values
t.sp <- runif(500)
# No intraspecific variation
pool[,3] <- t.sp[pool[,2]]
# Generate a neutral community drawn from the pool
comm2<- coalesc(J, m, pool = pool)
plot_comm(comm2)

# Directional habitat filtering toward t = 0
comm3 <- coalesc(J, m, filt = function(x) 1 - x, pool = pool)
# Regional and local trait distributions
plot_comm(comm3)

# Function for environmental filtering
sigma <- 0.1
filt_gaussian <- function(t, x) exp(-(x - t)^2/(2*sigma^2))

# Stabilizing habitat filtering around t = 0.1
comm4a <- coalesc(J, m, filt = function(x) filt_gaussian(0.1, x), pool = pool)
plot_comm(comm4a)
# Stabilizing habitat filtering around t = 0.5
comm4b <- coalesc(J, m, theta, filt = function(x) filt_gaussian(0.5, x),
                  pool = pool)
plot_comm(comm4b)
# Stabilizing habitat filtering around t = 0.9
```

```
comm4c <- coalesc(J, m, theta, filt = function(x) filt_gaussian(0.9, x),
                  pool = pool)
plot_comm(comm4c)

# Mean trait values in communities reflect the influence of habitat filtering
mean(comm4a$com[, 3])
mean(comm4b$com[, 3])
mean(comm4c$com[, 3])

# Disruptive habitat filtering around t = 0.5
comm5 <- coalesc(J, m, filt = function(x) abs(0.5 - x), pool = pool)
plot_comm(comm5)

# Multi-modal habitat filtering
t.sp <- rnorm(500)
pool[, 3] <- t.sp[pool[,2]]
comm6 <- coalesc(J, m, filt = function(x) sin(3*x) + 1, pool = pool)
plot_comm(comm6)
```

`coalesc_abc` *Estimation of neutral and non-neutral parameters of community assembly using Approximate Bayesian Computation (ABC)*

Description

Estimates parameters of neutral migration-drift dynamics (through migration rate m and parameters of environmental filtering (through a filtering function `filt.abc()`) from the composition of a local community and the related regional pool.

Usage

```
coalesc_abc(comm.obs, pool = NULL, multi = "single", traits = NULL,
            f.sumstats, filt.abc = NULL, params = NULL,
            theta.max = NULL, nb.samp = 10^6, parallel = TRUE,
            tol = NULL, pkg = NULL, method="rejection")
do.simul(J, pool = NULL, multi = "single", nb.com = NULL,
            traits = NULL, f.sumstats = NULL, filt.abc = NULL,
            params, theta.max = NULL, nb.samp = 10^6,
            parallel = TRUE, tol = NULL, pkg = NULL,
            method = "rejection")
```

Arguments

`comm.obs`	the observed community composition. If `multi = FALSE` (default), should be a matrix or data.frame of individuals on rows with their individual id (first column), and species id (second column).

`pool`	composition of the regional pool to which the local community is hypothesized to be related through migration dynamics with possible environmental filtering. Should be a matrix of individuals on rows with their individual id (first column), species id (second column), and (optionally) the trait values of the individuals.
`multi`	structure of the community inputs: • if `multi = "single"`, `comm.obs` contains a single community • if `multi = "tab"`, the user provides a site-species matrix (**sites in rows and species in columns**) • if `multi = "seqcom"`, `comm.obs` contains a list of communities
`traits`	the trait values of species in the regional pool. It is used if trait information is not provided in `pool`. In this case, intraspecific trait variation is assumed to be null.
`f.sumstats`	a function allowing to calculate the summary statistics of local community composition. Will be used to compare observed and simulated community composition in the ABC estimation. It should take a community as input and output a list of summary statistics.
`filt.abc`	the hypothesized environmental filtering function. It is a function of individual trait values and additional parameters to be estimated.
`params`	a matrix of the bounds of the parameters used in `filt.abc`. The row names of `params` provide the parameter names used in ABC calculation and output. First column contains minimum values and second column contains maximum values.
`theta.max`	if `pool = NULL`, regional abundances will be simulated following a log-series distribution. The function will estimate the `theta` parameter of this distribution. `theta.max` then provides the upper bound for this estimation.
`nb.samp`	the number of parameter values to be sampled in ABC calculation. Random values of parameters of environmental filtering (see `filt.abc` and `params`) and of migration (denoted as `m`) are drawn from a uniform distribution between minimum and maximum values provided in `params` (and between 0 and 1 for m).
`parallel`	boolean. If `parallel = TRUE`, the function will perform parallel processing using the `parLapply()` function of package `parallel`.
`tol`	the tolerance value used in ABC estimation (see help in `abc()` function of package `abc` for further information).
`pkg`	packages needed for calculation of `filt.abc` and/or `f.sumstats`.
`method`	the method to be used in ABC estimation (see help on `abc()` function of package `abc` for further information).
`J`	local community size.
`nb.com`	number of communities.

Details

`coalesc_abc()` performs ABC estimation for one (if `multi = FALSE`, default) or several communities (if `multi = TRUE`) related to the same regional pool.

`do.simul()` provides the simulated communities used in ABC estimation, and is not intended to be used directly.

Value

`par`	parameter values used in simulations.
`obs`	observed summary statistics.
`obs.scaled`	observed summary statistics standardized according to the mean and standard deviation of simulated values.
`ss`	standardized summary statistics of the communities simulated with parameter values listed in `par`.
`abc`	a single (if `multi` = `FALSE`, default) or a list of abc objects including ABC estimation information for each community provided in input (`comm.obs`).

Author(s)

F. Munoz

References

Jabot, F., and J. Chave. 2009. Inferring the parameters of the neutral theory of biodiversity using phylogenetic information and implications for tropical forests. Ecology Letters 12:239-248.

Csillery, K., M. G. B. Blum, O. E. Gaggiotti, and O. Francois. 2010. Approximate Bayesian computation (ABC) in practice. Trends in Ecology & Evolution 25:410-418.

Csillery, K., O. Francois, and M. G. Blum. 2012. abc: an R package for Approximate Bayesian Computation (ABC). Methods in Ecology and Evolution 3:475-479.

See Also

`abc()` in `abc` package, `parLapply()` in `parallel` package.

Examples

```
# Trait-dependent filtering function
filt_gaussian <- function(t, params) exp(-(t-params[1])^2/(2*params[2]^2))

# Definition of parameters and their range
params <- data.frame(rbind(c(0, 1), c(0.05, 1)))
row.names(params) <- c("topt", "sigmaopt")
# Number of values to sample in prior distributions
nb.samp <- 10^6 # Should be large

## Not run:
# Basic summary statistics
f.sumstats <- function(com) array(dimnames=list(c("cwm", "cwv", "cws",
                                                  "cwk", "S", "Es")),
                                  c(mean(com[,3]), var(com[,3]),
                                    e1071::skewness(com[,3]),
                                    e1071::kurtosis(com[,3]),
                                    vegan::specnumber(table(com[,2])),
                                    vegan::diversity(table(com[,2]))))

# An observed community is here simulated (known parameters)
```

```
comm <- coalesc(J = 400, m = 0.5, theta = 50,
                filt = function(x) filt_gaussian(x, c(0.2, 0.1)))

# ABC estimation of the parameters based on observed community composition
## Warning: this function may take a while
res <- coalesc_abc(comm$com, comm$pool, f.sumstats = f.sumstats,
                   filt.abc = filt_gaussian, params = params,
                   nb.samp = nb.samp, parallel = TRUE,
                   pkg = c("e1071","vegan"), method = "neuralnet")
plot(res$abc, param = res$par)
hist(res$abc)

# Cross validation
## Warning: this function is slow
res$cv <- abc::cv4abc(param = res$par, sumstat = res$ss, nval = 1000,
                      tols = c(0.01, 0.1, 1), method = "neuralnet")
plot(res$cv)

# Multiple community option
# When the input is a site-species matrix, use argument multi="tab"
# See vignette Barro_Colorado for more details

# When the input is a list of communities, use argument multi="seqcom"
comm.obs <- list()

comm.obs[[1]] <- cbind(rep(1,400), coalesc(J = 400, m = 0.5, filt = function(x)
                                           filt_gaussian(x, c(0.2, 0.1)),
                                           pool = comm$pool)$com))
comm.obs[[2]] <- cbind(rep(2,400), coalesc(J = 400, m = 0.5, filt = function(x)
                                           filt_gaussian(x, c(0.5, 0.1)),
                                           pool = comm$pool)$com))
comm.obs[[3]] <- cbind(rep(3,400), coalesc(J = 400, m = 0.5, filt = function(x)
                                           filt_gaussian(x, c(0.8, 0.1)),
                                           pool = comm$pool)$com))

comm.obs <- lapply(comm.obs, as.matrix)

res <- coalesc_abc(comm.obs, comm$pool, multi="seqcom", f.sumstats=f.sumstats,
                   filt.abc = filt_gaussian, params = params, nb.samp = nb.samp,
                   parallel = TRUE, pkg = c("e1071","vegan"), tol = 0.1,
                   method = "neuralnet")

lapply(res$abc, summary)


## End(Not run)
```

`forward` *Simulation of neutral and niche-based community dynamics forward in time*

Description

Simulates niche-based (habitat filtering and/or limiting similarity) and neutral community dynamics from a given initial composition, over a given number of generations.

Usage

```
forward(initial, prob = 0, d = 1, gens = 150, keep = FALSE,
        pool = NULL, limit.sim = FALSE, coeff.lim.sim = 1,
        sigm = 0.1, filt = NULL, prob.death = NULL,
        method.dist = "euclidean", plot_gens = FALSE)
get_number_of_gens(given_size, pool, nbrep = 5, prob = 1, d = 1,
                   gens = NULL, limit.sim = FALSE,
                   coeff.lim.sim = 1, sigm = 0.1, filt = NULL,
                   prob.death = NULL, method.dist = "euclidean",
                   plot_gens = FALSE)
pick(com, d = 1, prob = 0, pool = NULL, prob.death = prob.death,
     limit.sim = NULL, coeff.lim.sim = 1, sigm = 0.1, filt = NULL,
     new.index = new.index, method.dist = "euclidean")
pick.mutate(com, d = 1, prob.of.mutate = 0, new.index = 0)
pick.immigrate(com, d = 1, prob.of.immigrate = 0, pool,
               prob.death = NULL, limit.sim = NULL, coeff.lim.sim = 1,
               sigm = 0.1, filt = NULL, method.dist = "euclidean")
```

Arguments

`com`, `initial`	starting community. It is in principle a three (or more) column matrix or data.frame including individual ID, species names and trait values. For strictly neutral dynamics, it can be a vector of individual species names.
`prob`, `prob.of.immigrate`, `prob.of.mutate`	probability of an individual establishing in the community not being a descendant of an existing individual. If descendant from a new ancestor, can be either through immigration (in `pick.immigrate()`) or through mutation (in `pick.mutate()`).
`d`	number of individuals that die in each time-step.
`gens`	number of generations to simulate.
`keep`	boolean value. If `FALSE` (default) the function output only the community composition at the end of the simulation. If `TRUE` the function output a list of community composition at successive time steps (see Value section).
`pool`	the regional pool of species providing immigrants to the local community. It is in principle a three-column matrix or data frame including individual ID, species names and trait values. If trait information is missing, a random trait value is given to individuals, from a uniform distribution between 0 and 1. If `NULL`, the pool is simulated as a metacommunity at speciation-drift equilibrium, based on `prob` for speciation rate.
`given_size`	size of the community you want to have an estimate of the number of generations needed to reach stationarity in species richness.
`nbrep`	number of replicates from which you want to estimate the number of generations needed to reach stationarity in species richness.

`limit.sim`	if `TRUE`, limiting similarity will be simulated, based on species trait distances (computed with the method given by `method.dist`) and a Gaussian overlapping function.
`coeff.lim.sim`	adjust the intensity of limiting similarity.
`sigm`	adjust the variance of the overlap function used to calculate limiting similarity.
`filt`	the function used to represent habitat filtering. For a given trait value `t`, `filt(t)` represents the probability that an individual with trait `t` enters the local community.
`prob.death`	provides a baseline probability of death that is homogeneous across species. It is used in niche-based dynamics to represent the balance of baseline and niche-dependent mortality.
`method.dist`	provide the method to compute trait distances between individuals (syntax of function `dist`, can be in the list `c("euclidean","maximum", "manhattan","canberra", "binary", "minkowski")`).
`new.index`	prefix used to give a new species name when speciation occurs.
`plot_gens`	plot the number of unique individuals and species over generations.

Details

It is a zero-sum game, so that the number of individuals of the community is fixed to the number of individuals in initial community.

When niche-based dynamics are simulated, the niche-based constraints influence both immigration and mortality.

Function `get_number_of_gen()` allows determining the number of generations needed to reach stationary richness for given parameterization of `forward()`. The target number of generation is based on assessing the change point in species richness change over time for replicate simulated communities with random initial composition. A conservative measure is proposed as the maximum time to reach stationary richness over the replicate simulated communities.

Functions `pick.immigrate()` and `pick.mutate()` are used to simulate immigration and speciation events within a time step. They are embedded in forward and are not really intended for the end user.

Value

`com`	if `keep = FALSE`, a data.frame of simulated individuals, with the label of ancestor individual in the regional pool on first column (as in the first column of the pool), species label on second column (as in the second column of the pool), and species trait (as in the third column of the pool).
`pool`	a data.frame of the individuals of the regional source pool, with the label of ancestor individual in the regional pool on first column (as in first column of input `pool`), species label on second column (as in second column of input `pool`), and species trait (as in third column of input `pool`).
`sp_t`	a vector of species richness at each time step.
`com_t`	if `keep = TRUE`, a list of community composition for each time step (a data.frame as in `com`).

`dist.t`	if `limit.sim` = `TRUE`, the average value of the limiting similarity function over time.
`new.index`	for `pick.mutate()`, return the new index to be used for species name at a next speciation event.

Author(s)

F. Munoz, derived from the `untb` function of R. Hankin.

References

For neutral dynamics, S. P. Hubbell 2001. "The Unified Neutral Theory of Biodiversity". Princeton University Press.

Examples

```
## Not run:
# Initial community composed of 10 species each including 10 individuals
initial1 <- rep(as.character(1:10), each = 10)

# Simulation of speciation and drift dynamics over 100 time steps
final1 <- forward(initial = initial1, prob = 0.1, gens = 1000)
# The final community includes new species (by default names begins with "new")
final1$com$sp # includes new species generated by speciation events

# A regional pool including 100 species each including 10 individuals
pool <- rep(as.character(1:100), each = 10)

# Simulation of migration and drift dynamics over 1000 time steps
final2 <- forward(initial = initial1, prob = 0.1, gens = 1000, pool = pool)
# The final community includes species that have immigrated from the pool
final2$com$sp # includes new species that immigrated from the pool

# Initial community composed of 10 species each including 10 individuals,
# with trait information for niche-based dynamics
initial2 <- data.frame(sp = rep(as.character(1:10), each = 10),
                      trait = runif(100))

# Simulation of stabilizing hab. filtering around t = 0.5, over 1000 time steps
sigm <- 0.1
filt_gaussian <- function(t,x) exp(-(x - t)^2/(2*sigm^2))
final3 <- forward(initial = initial2, prob = 0.1, gens = 1000, pool = pool,
                 filt = function(x) filt_gaussian(0.5,x))
plot_comm(final3) # trait distribution in final community

# With higher immigration
final4 <- forward(initial = initial2, prob = 0.8, gens = 1000, pool = pool,
                 filt = function(x) filt_gaussian(0.5,x))
plot_comm(final4) # should be closer to 0.5

# Simulation of limiting similarity, over 1000 time steps
final5 <- forward(initial = initial2, prob = 0.1, gens = 1000, pool = pool,
                 limit.sim = TRUE)
plot_comm(final5)

# Stronger limiting similarity
final6 <- forward(initial = initial2, prob = 0.1, gens = 1000, pool = pool,
                 limit.sim = TRUE, coeff.lim.sim = 20)
plot_comm(final6) # the distribution will be more even

# Variation of community richness with time
final7 <- forward(initial = initial2, prob = 0.1, gens = 1000, pool = pool,
                 limit.sim = TRUE, keep = TRUE, plot_gens = TRUE)

# Check stationarity
plot(unlist(lapply(final7$com_t, function(x) length(unique(x[, 2])))),
     xlab = "Time step", ylab = "Community richness")

# Index of limiting similarity over time
plot(final7$dist.t, xlab = "Time step", ylab = "Limiting similarity")

## End(Not run)
```

`plot_comm` *Regional vs. Local trait distributions of abundances*

Description

Graphical function to used on the output of `coalesc()` or `forward()` functions.It aims at plotting links between regional and local trait/abundance distributions.

Usage

```
plot_comm(x, type = "trait", seltrait = 1, main = NULL)
```

Arguments

`x`	a list including the species pool composition (`x$pool`) and the local community composition (`x$com`). For example, x may be the output of `coalesc()` or `forward()` functions.
`type`	• if `type = "trait"`, the function displays density plots of trait distributions. • if `type = "abund"`, it displays the relationship between local and regional abundances.
`seltrait`	index of the trait to be plotted following community data.frame (if multiple traits used in simulation).
`main`	an overall title for the plot.

Details

If type = "trait", the function provides density plots of the trait or abundance distributions in the regional pool and in a local community. If type = "abund", the function displays the relationship between regional and local species relative abundances. By default type = "trait". To be used on the output of coalesc() or forward() functions.

Value

Return two stacked ggplot2 density plots if type = "trait" and a biplot if type = "abund".

Author(s)

F. Munoz; P. Denelle

Examples

```
# Simulation of a neutral community including 100 individuals
J <- 500; theta <- 50; m <- 0.1;
comm1 <- coalesc(J, m, theta)
plot_comm(comm1)
plot_comm(comm1, type = "abund")

# Stabilizing habitat filtering around t = 0.5
comm2 <- coalesc(J, m, theta, filt = function(x) 0.5 - abs(0.5 - x))
plot_comm(comm2)
plot_comm(comm2, type = "abund")
```

---

tcor *Generates Correlated Traits*

Description

Create two random vectors of traits correlated between each other or a vector of traits correlated to an existing one. The linear correlation is defined by the parameter rho.

Usage

```
tcor(n, rho = 0.5, mar.fun = rnorm, x = NULL, ...)
```

Arguments

n	the integer number of values to be generated.
rho	a numeric parameter defining the linear correlation between the two traits (default is 0.5). It must belong to the interval [-1, 1].
x	an vector of numeric values. Default is NULL.
mar.fun	a function defining the random generation for the trait distribution. Default is rnorm.
...	other arguments for the mar.fun() function.

Details

rho parameter is set to 0.5 by default. x = NULL by default. Code adapted from: http://stats.stackexchange.com/questions/15011/generate-a-random-variable-with-a-defined-correlation-to-an-exis

Value

Return a data.frame with two numeric columns, each column defining a trait.

Author(s)

P. Denelle F. Munoz

Examples

```
# With no predefined trait
traits <- tcor(n = 10000, rho = 0.8)
plot(traits[, 1], traits[, 2])
cor(traits[, 1], traits[, 2])

# With existing trait
existing_trait <- rnorm(10000, 10, 1)
traits <- tcor(n = 10000, rho = 0.8, x = existing_trait)
plot(traits[, 1], traits[, 2])
cor(traits[, 1], traits[, 2])
```

Index