Package ‘island’
January 21, 2019

Type Package
Title Stochastic Island Biogeography Theory Made Easy
Version 0.2.2
Date 2019-01-21
Description Tools to develop stochastic models based on the Theory of Island Biogeography (TIB) of MacArthur and Wilson (1967) <DOI:10.1023/A:1016393430551> and extensions. The package implements methods to estimate colonization and extinction rates (including environmental variables) given presence-absence data, simulate community assembly, and perform model selection.

NeedsCompilation yes
Depends R (>= 3.0.0), stats, utils
License GPL-3
LazyData TRUE
RoxygenNote 6.1.1
Suggests testthat, knitr, rmarkdown, rootSolve
VignetteBuilder knitr
Author Vicente Jimenez [aut, cre],
David Alonso [aut]
Maintainer Vicente Jimenez <vicente.jimenez.ontiveros@gmail.com>
Repository CRAN
Date/Publication 2019-01-21 16:20:03 UTC

R topics documented:

akaikeic .................................................. 2
all_environmental_fit .................................. 3
alonso15 .................................................. 4
cetotrans ............................................... 5
data_generation ........................................ 6
ibd_models ............................................. 7
idaho ..................................................... 8
akaikeic calculates the Akaike Information Criterion (AIC) of a model.
akaikeicc calculates the corrected Akaike Information Criterion (AICc) for small samples.

Usage
akaikeic(NLL, k)
akaikeicc(NLL, k, n)

Arguments
NLL Negative Log-Likelihood of the model.
k Number of parameters of the model.
n Sample size.

Details

\[ AIC = 2 * k + 2 * NLL \]

\[ AICc = 2 * k - 2 * lnL + 2 * k * (k + 1)/(n - k - 1) \]

Value
A number with the AIC value for a model with k parameters and negative log-likelihood NLL, or the AICc value for a model with k parameters, negative log-likelihood NLL and sample size n.
all_environmental_fit

See Also

weight_of_evidence

Examples

akaikeic(1485.926, 3)
akaikeic(736.47, 6, 15)
akaikeic(736.47, 6, 100)

Description

all_environmental_fit estimates the best expressions for colonization and extinction rates given
their dependency on environmental variables.
greedy_environmental_fit estimates expressions for colonization and extinction rates given their
dependency on environmental variables using a greedy algorithm.
custom_environmental_fit estimates the m.l.e. of the parameters describing the relationship be-
tween colonization and extinction rates and environmental variables.
NLL_env returns the Negative Log-Likelihood of a pair of colonization and extinction rates for a
given dataset with an specific relationship with environmental variables.

Usage

all_environmental_fit(dataset, vector, env, c, e, aic, verbose = F)
custom_environmental_fit(dataset, vector, params, c_expression, 
e_expression)
NLL_env(dataset, vector, params, c_expression, e_expression)
greedy_environmental_fit(dataset, vector, env, c, e, aic, verbose = F)

Arguments

dataset A single dataset.
vector A vector indicating the columns with presence-absence data.
env The names of the environmental variables to be considered.
c Tentative colonization rate.
e Tentative extinction rate.
aic Tentative AIC to be improved by the optimizer.
verbose Logical. Do you want to get the intermediate steps looking for the best model?
params A vector with priors of the parameters in c_expression and e_expression.
c_expression Expression for colonization.
e_expression Expression for extinction.
Details

`all_environmental_fit` calculates all the combinations of parameters, that increase exponentially with the number of parameters. We advise to keep low the number of parameters.

`greedy_environmental_fit` adds sequentially environmental variables to the expressions of colonization and extinction rates and fix one at a time until termination, when only adding one variable does not improve the AIC of the last accepted model.

Value

A list with three components: a expression for colonization, a expression for extinction and the output of the optimization function, or the output of the optimization function in the custom environmental fit.

In the case of `NLL_env`, returns the NLL of an specific set or parameters describing the relationship of environmental covariates with colonization and extinction.

Note

AIC is recommended to be higher than the AIC of the most simple model (i.e. not including environmental variables).

See Also

`rates_calculator`

Examples

```r
# Not run:
all_environmental_fit(idaho[[1]],3:23,c("idaho[[2]]$TOTAL.ppt", "idaho[[2]]$ANNUAL.temp"),0.13,0.19,100000)

# End(Not run)
custom_environmental_fit(idaho[[1]], 3:23, c(-0.008497925, -0.01729602, 0.19006501, 0.93486956), expression(params[[1]] * idaho[[2]]$TOTAL.ppt[1] + params[3]), expression(params[[2]] * idaho[[2]]$ANNUAL.temp[i] + params[4]))
NLL_env(idaho[[1]], 3:23, c(-0.008497925, -0.01729602, 0.19006501, 0.93486956), expression(params[[1]] * idaho[[2]]$TOTAL.ppt[1] + params[3]), expression(params[[2]] * idaho[[2]]$ANNUAL.temp[i] + params[4]))
```

Description

A list with three datasets containing presence-absence data for the reassembly process of coral fish communities in three atolls (Agatti, Kadmat and Kavaratti) of the Lakshadweep Archipelago (India).
Format

A list with 3 dataframes, each corresponding to the survey of a different atoll. Dataframes have in columns:

Species Name of the species found
Trophic.Level A number indicating the trophic level of the surveyed species
Presence-absence data Several columns with letters (indicating the atoll surveyed) and the year in which the surveys were done
Guild Guild of the surveyed species

Details

Surveys were conducted from 2000 to 2011 in order to follow community reassembly after a coral mass mortality event in the relatively unfished Lakshadweep Archipelago. Results indicated that higher trophic groups suffer an increased extinction rate even without fishing targeting them.

Note

Kavaratti atoll was not surveyed in 2000 and 2010.

Source


---

cetotrans From rates to probabilities

description

cetotrans calculates transition probabilities from colonization and extinction rates for a determined interval of time, when provided.

Usage

cetotrans(c, e, dt = 1)

Arguments

c Colonization rate.
e Extinction rate.
dt Interval of time or a vector of time intervals.
data_generation

Details

Given a pair of colonization and extinction rates, we can calculate the transition probabilities with the following equations:

\[
T_{01} = \left(\frac{e}{c + e}\right) \times \left(1 - \exp\left(-\left(c + e\right) \times dt\right)\right) \\
T_{10} = \left(\frac{c}{c + e}\right) \times \left(1 - \exp\left(-\left(c + e\right) \times dt\right)\right)
\]

Value

A matrix with the transition probabilities \(T_{01}\) and \(T_{10}\) of the Markov chain associated with the specified colonization and extinction rates.

Examples

```
cetotrans(0.13, 0.19)  
cetotrans(0.2, 0.2, 2)
```

---

data_generation | Data simulation of colonization-extinction dynamics

Description

data_generation simulates species richness data according to the stochastic model of island biogeography.

PA_simulation simulates presence-absence data according to the stochastic model of island biogeography.

Usage

data_generation(x, column, transitions, iter, times)

PA_simulation(x, column, transitions, times = 1)

Arguments

- **x**: A dataframe with the vector of initial absences and presences.
- **column**: A number indicating the column with the initial presence-absence data.
- **transitions**: A matrix with the transition probabilities of the simulation, in the form \((T_{01}, T_{10})\), that can contain one single pair or multiple pairs.
- **iter**: Number of times that the specified dynamics should be repeated.
- **times**: Number of temporal steps to simulate.

Details

To simulate community assembly, we need an initial vector of presence-absence, from which the subsequent assembly process will be simulated. This initial vector is considered as \(x[,\ column]\).
## Value
A matrix with species richness representing each row consecutive samples and each column a replica of the specified dynamics or a matrix with presence-absence data for the specified dynamics, each row representing a species and each column consecutive samplings.

## Note
You can simulate not only with a colonization and extinction pair, but with the pairs obtained from the environmental fit. In this case, you still have to indicate exactly the number of temporal steps that you are going to simulate.

## See Also
- `cetotrans` to obtain the transition probabilities associated with a colonization-extinction pair.

## Examples
```r
data_generation(as.data.frame(rep(0, 100)), 1, matrix(c(0.5, 0.5), ncol = 2), 5, 25)
data_generation(alonso15[[1]], 3, matrix(c(0.5, 0.5), ncol = 2), 5, 25)
PA_simulation(as.data.frame(c(rep(0, 163), rep(1, 57))), 1, c(0.13, 0.19), 20)
```

---

### ibd_models

Inmigration, birth, death-models

#### Description
`ibd_models` simulates population dynamics under three different inmigration, birth and death models.

#### Usage
```r
ibd_models(n0, beta, delta, mu, K = NULL, time_v, type)
```

#### Arguments
- `n0`: Initial number of individuals in the population.
- `beta`: Birth rate, in time^{-1} units.
- `delta`: Death rate, in time^{-1} units.
- `mu`: Inmigration rate, in time^{-1} units.
- `K`: Carrying capacity.
- `time_v`: Vector of times to sample. Must start with 0.
- `type`: Type of inmigration, birth, death-model used to simulate the dynamics. This must be "Kendall", "Alonso" or "Haegeman".

---
Details

We have included three different stochastic models: Kendall (1948) seminal work, Alonso & McKane (2002) mainland-island model and Haegeman & Loreau (2010) basic population model with denso-dependent deaths. These models are different formulations of a population dynamics with three basic processes: birth, death and immigration of individuals. For the specifics, please refer to the original articles.

Value

A data.frame with two columns: one with the time vector and the other with the number of individuals at those times.

Note

Haegeman & Loreau model specification breaks for high values of n₀ when the birth rate is lower than the death rate.

References


Examples

```
ibd_models(n0 = 0, beta = 0.4, delta = 0.3, mu = 0.2,
           time_v = 0:20, type = "Kendall")
ibd_models(n0 = 0, beta = 0.4, delta = 0.3, mu = 0.1, K = 30,
           time_v = 0:20, type = "Alonso")
```

idaho

Mapped plant community time series, Dubois, ID

Description

A list with two datasets containing presence-absence and environmental data for a plant community of sagebrush steppe in Dubois, Idaho, USA
irregular_multiple_datasets

Format

A list with 2 dataframes, one corresponding to the presence-absence data and the other to the environmental variables. The first dataframe has in columns:

- **quad** Name of the quadrat surveyed
- **species** Name of the species found

**Presence-absence data** Several columns with the year in which the surveys were conducted

The second dataframe has the following columns:

- **YEAR** Year in which surveys were conducted

**Environmental variables** Data of the recorded environmental variables in the form XXX.YYY, where XXX denotes a month (or a total) and YYY can refer to snow (in inches), temperature (fahrenheit degrees) or precipitation (in inches)

Details

A historical dataset consisting of a series of permanent 1-m² quadrats located on the sagebrush steppe in eastern Idaho, USA, between 1923 and 1973. It also contains records of monthly precipitation, mean temperature and snowfall. Total precipitation, total snowfall, and mean annual temperature have been calculated from the original data.

Note

Only quadrats Q1, Q2, Q3, Q4, Q5, Q6, Q25 and Q26 are included here. The surveys were conducted annually from 1932 to 1955 with some gaps for the quadrats included here.

Source


References


irregular_multiple_datasets

c/e rates for irregular samplings in multiple datasets

Description

irregular_multiple_datasets estimates colonization and extinction rates for data in several datasets.

NLL_imd returns the Negative Log-Likelihood of a pair of colonization and extinction rates for irregular sampling schemes in several single dataset.
irregular_multiple_datasets

Usage

irregular_multiple_datasets(list, vectorlist, c, e, column = NULL, n = NULL, step = NULL, assembly = F, jacobian = F, verbose = F, CI = F)

NLL_imd(list, vectorlist, c, e, assembly = F)

Arguments

- list: A list of dataframes.
- vectorlist: A list of vectors indicating the columns with presence-absence data.
- c: Tentative colonization rate.
- e: Tentative extinction rate.
- column: The name of the column with groups to calculate their c_e pair.
- n: Minimal number of rows for each group.
- step: Accuracy to calculate the c_e pairs with.
- assembly: Logical indicating if the assembly starts from zero species or not.
- jacobian: Logical. Use the semianalytical method to estimate colonization and extinction rates?
- verbose: Logical. If TRUE, gives the output of the optimizer or the numerical solver that finds the values of c and e.
- CI: Logical. If TRUE, gives the confidence interval of the colonization and extinction rates.

Value

irregular_multiple_datasets returns a dataframe with colonization and extinction rates and their upper and lower confidence interval, and if needed, the names of the groups to which colonization and extinction rates have been calculated. NLL_imd gives the NLL for a multiple datasets with irregular sampling schemes given a specific c and e.

Note

The columns with the presence-absence data should have the day of that sampling on the name of the column in order to calculate colonization and extinction.

See Also

regular_sampling_scheme, irregular_single_dataset

Examples

irregular_multiple_datasets(simberloff, list(3:17, 3:18, 3:17, 3:19, 3:17, 3:16), 0.001, 0.001)
## Not run:
irregular_multiple_datasets(simberloff, list(3:17, 3:18, 3:17, 3:19, 3:17,
irregular_single_dataset

---

c/e rates for irregular samplings in a dataset

Description

irregular_single_dataset estimates colonization and extinction rates in a single dataset with irregular sampling scheme.

NLL_isd returns the Negative Log-Likelihood of a pair of colonization and extinction rates for an irregular sampling scheme in a single dataset.

Usage

irregular_single_dataset(dataframe, vector, c, e, column = NULL, 
n = NULL, step = NULL, assembly = F, jacobian = F, verbose = F, 
CI = F)

NLL_isd(dataframe, vector, c, e, assembly = F)

Arguments

dataframe: A single dataframe.
vector: A vector indicating the columns with presence-absence data.
c: Tentative colonization rate.
e: Tentative extinction rate.
column: The name of the column with groups to calculate their c_e pair.
n: Minimal number of rows for each group
step: Accuracy to calculate the c_e pairs with.
assembly: Logical indicating if the assembly starts from zero species or not.
jacobian: Logical. Use the semianalytical method to estimate colonization and extinction rates?
verbose: Logical. If TRUE, gives the output of the optimizer or the numerical solver that finds the values of c and e.
CI: Logical. If TRUE, gives the confidence interval of the colonization and extinction rates.
Value
irregular_single_dataset returns a dataframe with colonization and extinction rates and their upper and lower confidence interval, and if needed, the names of the groups to which colonization and extinction rates have been calculated. NLL_isd gives the NLL for a single dataset in an irregular sampling scheme given a specific \( c \) and \( e \).

Note
The columns with the presence-absence data should have the day of that sampling on the name of the column in order to calculate colonization and extinction.

See Also
regular_sampling_scheme, irregular_multiple_datasets

Examples
irregular_single_dataset(simberloff[[1]], 3:17, 0.001, 0.001)
irregular_single_dataset(simberloff[[1]], 3:17, column = "Tax. Unit 1", 0.001, 0.001, 3)
## Not run:
irregular_single_dataset(simberloff[[1]], 3:17, column = "Tax. Unit 1", 0.001, 0.001, 3, 0.00001)

## Not run
NLL_isd(simberloff[[1]], 3:17, 0.0038, 0.0086)
on a per-species basis. In addition, we can make these rates depend on environmental variables measured at the same time that we took our samples. For more information of the basic model, please see the references.

### Data entry

The data should be organized in dataframes with consecutive presence-absence data of each sample ordered chronologically, being the data associated with a single species in a row. Additional columns can contain the filiations of every species to a group, i. e. a phylogenetic group or a guild.

### References


---

**Lakshadweep Archipelago coral fish community reassembly data (expanded)**

**Description**

A list with three data frames containing presence-absence data for the reassembly process of coral fish communities in three atolls (Agatti, Kadmat and Kavaratti) of the Lakshadweep Archipelago in India. These data contains a number of replicates (transects) per sampling time. It is in this respect that expands Alonso community data (see island R package).

**Format**

A list with three dataframes from the 3 different atoll. The dataframe has in columns:

- **Species** Name of the species found
- **Atoll** Atoll surveyed
- **Guild** Feeding strategy of the surveyed species
- **Presence-absence data** Several columns corresponding to the year in which the surveys were done. Year repetition means repeated sampling of the same atoll at the same time. Presences are represented by 1 and true absence are undetected presences by 0.
Details

Surveys were conducted from 2000 to 2013 in order to follow community reassembly after a coral mass mortality event in the relatively unfished Lakshadweep Archipelago. For most years, transects were taken in four locations per atoll. Although there might be some underlying heterogeneity, these transects are approximately taken as true replicates.

Note

Detectability per transect results to be of about 0.5, which means that the parameter 'Detectability' per atoll goes up to almost 0.94 if four transects per sampling time are taken (1-0.5^4).

Source


---

**Description**

A list with only one data frame containing presence-absence data for the reassembly process of coral fish communities in three atolls (Agatti, Kadmat and Kavaratti) of the Lakshadweep Archipelago in India. These data contains a number of replicates per sampling time. The data matrix marks missing data with a flag. It is in this respect that differs from `lakshadweep`.

**Format**

A list of a single dataframe with data from the 3 different atoll. The dataframe has in columns:

- **Species** Name of the species found
- **Atoll** Atoll surveyed
- **Guild** Feeding strategy of the surveyed species
- **Presence-absence data** Several columns corresponding to the year in which the surveys were done. Year repetition means repeated sampling of the same atoll at the same time. Presences are represented by 1 and true absences or undetected presences by 0.

**Details**

Surveys were conducted from 2000 to 2013 in order to follow community reassembly after a coral mass mortality event in the relatively unfished Lakshadweep Archipelago. For most years, transects were taken in four locations per atoll. Although there might be some underlying heterogeneity, these transects are approximately taken here as true replicates.
Note
Detectability per transect results to be of about 0.5, which means that the parameter 'Detectability' per atoll goes up to almost 0.94 if four transects per sampling time are taken (1-0.5^4). The sampling structure differs from atoll to to atoll. Certain columns are filled with 0.1. This is the missing value flag.

Source

---

**mss_cedp**

*Likelihood approach for estimating colonization/extinction with perfect or imperfect detectability*

**Description**

mss_cedp conducts maximum likelihood estimation of colonization/extinction parameters of different data sets. This function can handle imperfect detectability and missing data defining a heterogeneous sampling structure across input data matrix rows. The function generates, as an output, either a 3-column matrix (Colonization, Extinction, Negative LogLikelihood) or 5-column matrix (Colonization, Extinction, Detectability, P_0, Negative LogLikelihood), depending on the value of the input parameter PerfectDetectability (either `true` or `false`).

**Usage**

```r
mss_cedp(data, time, factor, tags, colonization = 1, extinction = 1, detectability_value = PNUL, phi_time_P_value = PNUL, tol = 1e-08, mit = 100, c_max = 10, c_min = 0, e_max = 10, e_min = 0, d_max = 99, d_min = 0, p_max = 99, p_min = 0, i_0 = 0, i_1 = 1, i_2 = 2, i_3 = 3, z = 2, minimization = 1, verbose = 0, mv_flag = PN1, perfectdetectability = TRUE)
```

**Arguments**

- **Data**: data frame containing presence data per time (in cols) and sites (in rows)
- **Time**: an array of length n containing sampling times
- **Factor**: column number containing the 'data frame' factor used to split total data into level-based groups
- **Tags**: array of names (one short for each level of the factor analyzed)
- **Colonization**: guess value to initiate search / parameter value
- **Extinction**: guess value to initiate search / parameter value
- **Detectability_Value**: guess value to initiate search / parameter value
Phi_Time_0_Value
  
  guess value to initiate search / parameter value

Tol
  
  stopping criteria of the search algorithm.

MIT
  
  max number of iterations of the search algorithm.

C_MAX
  
  max value for the possible range of colonization values

C_min
  
  min value for the possible range of colonization values

E_MAX
  
  max value for the possible range of colonization values

E_min
  
  min value for the possible range of colonization values

D_MAX
  
  max value for the possible range of colonization values

D_min
  
  min value for the possible range of colonization values

P_MAX
  
  max value for the possible range of colonization values

P_min
  
  min value for the possible range of colonization values

I_0
  
  has to be 0 or 1. Defaults to 0

I_1
  
  has to be 0 or 1. Defaults to 1

I_2
  
  has to be 0, 1, 2, or 3. Defaults to 2

I_3
  
  has to be 0, 1, 2, or 3. Defaults to 3

z
  
  dimension of the parameter subspace for which the optimization process will take place. Defaults to 2

Minimization, 1/0. If Minimization is 0, then no minimization is performed.

Verbose
  
  more/less (1/0) information about the optimization algorithm will be printed out.

MV_FLAG
  
  missing Value Flag (to specify sites and times where no sample exists)

PerfectDetectability
  
  TRUE means 'Perfect Detectability'. Of course, FALSE means 'Imperfect Detectability'

Details

The input is a data frame containing presence data per time (in cols). Different factors (for instance, OTU, location, etc) can slide the initial data frame accordingly. Model parameters will be estimated for each of these groups independently that correspond to each level of the chosen factor. If Minimization is 0, then no maximum likelihood estimation is performed and only the likelihood evaluation at the input model parameter values is returned. Searches are based on the Nelder-Mead simplex method, but conducted in a bounded parameter space which means that in case a neg log-likelihood (NLL) evaluation is called out from these boundaries, the returned value for this NLL evaluation is artificially given as the maximum number the machine can hold. Each group is named by a short-length-character label (ideally, 3 or 4 characters). All labels should have the same character length to fulfill memory alignment requirements of the shared object called by .C(...) function. I_0, I_1, I_2, I_3 are model parameter keys. They are used to define a 4D-vector (Index). The search will take place on the full parameter space defined by model parameters (I_0, I_1) if PerfectDetectability is TRUE or, alternatively, defined by (I_0, I_1, I_2, I_3) if PerfectDetectability is FALSE. Model parameter keys correspond to colonization (0), extinction (1), detectability (2), and P_0 (3) model parameters. For instance, if (I_0, I_1) is (1, 0), the search will take place within the parameter space defined by extinction, as the first axis, and colonization, as the second.
rates_calculator

Colonization and extinction rates calculator for expressions.

Description

rates_calculator Calculate colonization and extinction rates depending of their expressions.

Usage

rates_calculator(params, c_expression, e_expression, t)

Arguments

- params A vector with priors of the parameters in c_expression and e_expression.
- c_expression Expression for colonization.
- e_expression Expression for extinction.
- t Number of colonization and extinction pairs required.

Value

A matrix with the colonization and extinction rates.

See Also

all_environmental_fit

Examples

rates_calculator(c(-0.00497925, -0.01729602, 0.19006501, 0.93486956), expression(params[1] * idaho[[2]]$TOTAL.ppt[i] + params[3]), expression(params[2] * idaho[[2]]$ANNUAL.temp[i] + params[4]), 21)
regular_sampling_scheme

*c/e rates for a regular sampling scheme*

Description

regular_sampling_scheme estimates colonization and extinction rates for a community or groups in a community. NLL_rss returns the Negative Log-Likelihood of a pair of colonization and extinction rates for a regular sampling scheme.

Usage

```r
regular_sampling_scheme(x, vector, level = NULL, n = NULL, step = NULL, CI = FALSE)
NLL_rss(x, vector, c, e)
```

Arguments

- **x**: A single dataset.
- **vector**: A vector indicating the columns with presence-absence data.
- **level**: The name of the column that contains groups to subset them and calculate their colonization and extinction rates.
- **n**: Minimal number of rows for each group.
- **step**: Accuracy to calculate the c_e pairs with.
- **CI**: Logical. Should confidence intervals be returned?
- **c**: A colonization rate.
- **e**: An extinction rate.

Details

The confidence intervals are calculated with a binary search seeded with the Hessian of the estimated rates.

Value

regular_sampling_scheme returns a dataframe with colonization and extinction rates along with their lower and upper confidence intervals (optional), for each group if specified, and its number of rows and NLL. NLL_rss gives the NLL for a dataframe given a specific c and e.

See Also

irregular_single_dataset, irregular_multiple_datasets
Examples

regular_sampling_scheme(alonso15[[1]], 3:6)
regular_sampling_scheme(alonso15[[1]], 3:6, "Guild", n = 5)
regular_sampling_scheme(alonso15[[1]], 3:6, "Guild", n = 5, CI = TRUE)
NLL_rss(alonso15[[1]], 3:6, 0.52, 0.39)

Description

r_squared evaluates $R^2$ for our simulated dynamics.

simulated_model Error of the stochastic model.
null_model Error of the null model.

Usage

r_squared(observed, simulated, sp)
null_model(observed, sp)
simulated_model(observed, simulated)

Arguments

observed A vector with the actual observed species richness.
simulated A vector with the simulated species richness.
sp Number of species in the species pool.

Details

The importance of assessing how well a model predicts new data is paramount. The most used metric to assess this model error is $R^2$. $R^2$ is always referred to a null model and is defined as follows:

$$R^2 = 1 - \epsilon^2/\epsilon_0^2$$

where $\epsilon^2$ is the prediction error defined as the mean squared deviation of model predictions from actual observations, and $\epsilon_0^2$ is a null model error, in example, an average of squared deviations evaluated with a null model.

Our null model corresponds with a random species model with no time correlations, in which we draw randomly from a uniform distribution a number of species between 0 and number of species observed in the species pool. The expectation of the sum of squared errors under the null model is evaluated analytically in Alonso et al. (2015).
Value

$r_{\text{sq}}$ gives the value of $R^2$ for the predictions of the model.

null_model gives the average of squared deviations of the null model predictions from actual observations, $r_0^2$.

simulated_model gives the average of squared deviations of the model predictions from the actual observations, $\epsilon^2$.

Note

The value of $R^2$ depends critically on the definition of the null model. Note that different definitions of the null model will lead to different values of $R^2$.

References


Examples

```r
idaho.sim <- data_generation(as.data.frame(c(rep(0, 163), rep(1, 57))), 1, matrix(c(0.162599, 0.111252), ncol = 2), 250, 20)
idaho.me <- c(57, apply(idaho.sim, 1, quantile, 0.5))
r_squared(colSums(idaho[[1]][,3:23]), idaho.me, 220)

null_model(colSums(idaho[[1]][,3:23]), 220)
simulated_model(colSums(idaho[[1]][,3:23]), idaho.me)
```

Description

A list of datasets containing the presence-absence data gathered originally by Simberloff and Wilson in their defaunation experiment of six mangrove islands in the Florida Keys.

Format

A list with 6 dataframes, each corresponding to the survey of a different island. Dataframes have in columns:

**Taxa** Taxa considered

**PRE** Presence-absence before the defaunation process
Integers (e.g. 21, 40, 58...) Several columns with presence-absence data for the day specified

Tax. Unit 1 Highest taxonomic unit considered

Tax. Unit 2 Second highest taxonomic unit considered

Genera Genera of the identified taxon

Island Island of identification of the taxon

Details

The defaunation experiment of Simberloff and Wilson was aimed to test experimentally the Theory of Island Biogeography. The approach sought was eliminating the fauna of several islands and following the recolonization process.

After some trials, six red mangrove islets of Florida Bay were chosen for the task. These islets had to be stripped of all arthropoda fauna without harming the vegetation and then all the colonists were identified. The result of these defaunation experiments supported the existence of species equilibria and were consistent with the basic MacArthur-Wilson equilibrium model.

Note

The shaded entries in the original dataset, for taxa inferred to be present from other evidence rather than direct observation, are considered as present in these datasets.

Source


References


Description

`sss_cedp` conducts a maximum likelihood estimation of model parameters (Colonization, Extinction, Detectability, and Phi_Time_0) of MacKenzie et al (2003) colonization-extinction model. This function is an alternative to `mss_cedp` that takes a different input (a 2D array), and requires the same sampling structure for all input data matrix rows, this is, no missing data defining a heterogeneous sampling structure across rows are allowed. As an advantage, it may run faster than `mss_cedp`.

Usage

```r
sss_cedp(Data, Time, Transects, Colonization = 0.1, Extinction = 0.1, Detectability = 0.99, Phi_Time_0 = 0.5, Tol = 1e-06, MIT = 100, C_MAX = 2, C_min = 0, E_MAX = 2, E_min = 0, D_MAX = 0.999, D_MIN = 0.001, P_MAX = 0.999, P_min = 0.001, I_0 = 0, I_1 = 1, I_2 = 2, I_3 = 3, z = 4, Verbose = 0, Minimization = TRUE)
```

Arguments

- **Data**: S x N matrix containing presence data per transect (in cols):
- **Time**: an array of length n containing sampling times (without repetitions)
- **Transects**: an integer array of length n containing the number of transects per sampling time
- **Colonization**: guess value to initiate search / parameter value
- **Extinction**: guess value to initiate search / parameter value
- **Detectability**: guess value to initiate search / parameter value
- **Phi_Time_0**: guess value to initiate search / parameter value
- **Tol**: Stopping criteria of the search algorithm
- **MIT**: max number of iterations of the search algorithm
- **C_MAX**: max value of colonization values
- **C_min**: min value of colonization values
- **E_MAX**: max value of extinction values
- **E_min**: min value of extinction values
- **D_MAX**: max value of detectability values
- **D_MIN**: min value of detectability values
- **P_MAX**: max value for the initial presence probability on the site
- **P_min**: min value for the initial presence probability on the site
- **I_0**: parameter index of 1st parameter
- **I_1**: parameter index of 2nd parameter
Details

Maximum likelihood parameter estimation is conducted through bounded searches. This is the reason why the minimum and maximum values for each axis should be given as input arguments. The optimization procedure is the simplex method. A bounded parameter space implies that in case a neg loglikelihood (NLL) evaluation is required outside from these boundaries, the returned value for this NLL evaluation is artificially given as the maximum number the machine can hold.

The array Parameters \(I_0, I_1, I_2, I_3\) has to be a permutation of \((0, 1, 3, 4)\). This parameter indeces along with the imput parameter ’z’ are used to define a subparameter space where the search will be conducted. If \(z = 2\), then the search will take place on the plane defined by model parameters \((I_0, I_1)\). These indeces are model parameter keys: colonization \((0)\), extinction \((1)\), detectability \((2)\), and \(\Phi_{\text{Time}_0}\) \((3)\). For instance, if \((I_0, I_1, I_2, I_3) = (2, 3, 1, 0)\), and \(z = 2\), then the search will take place within the subparameter space defined by the detection probability (Detectability) and the probability of presence at time 0 (\(\Phi_{\text{Time}_0}\)). If Minimization is TRUE (default value), then the whole mle is conducted. If FALSE, the function only return the NLL value at the input model parameter values. Likelihood evaluations are exact provided the number of ’absences’ corresponding to either true absences or undetected presences in the input data matrix is not to high.

Examples

```r
Data1 <- lacshadweep[[1]]
Name_of_Factors <- c("Species","Atoll","Guild")
Factors <- Filter(is.factor, Data1)
No_of_Factors <- length(Factors[[1]])
n <- No_of_Factors + 1
D1 <- as.matrix(Data1[1:nrow(Data1),n:ncol(Data1)]
Time <- as.double(D1[1,])
P1 <- as.matrix(D1[2:nrow(D1),1:ncol(D1)])
# Dealing with time.
Time_Vector <- as.numeric(names(table(Time)))
Transsects <- as.numeric(table(Time))
R1 <- sss_cedp(P1, Time_Vector, Transsects,
Colonization=0.5, Extinction=0.5, Detectability=0.5,
\Phi_{\text{Time}_0}=0.5,
Tol=1.0e-8, Verbose = 1)
```
upgma_model_selection

Description

upgma_model_selection function conducts a model selection procedure intended to find an optimal partition that minimize AIC values. Maximum likelihood estimation of model parameters (Colonization, Extinction) or (Colonization, Extinction, Detectability, P_0) is performed assuming either perfect detectability or imperfect detectability, respectively. In the latter case, the input data frame should contain multiple transects per sampling time. This function can handle missing data defining a heterogeneous sampling structure across the rows of the input data matrix. The function generates, as an output, a Sx6 matrix with the following 6 columns (for the S different partitions): (No of Model Parameters, NLL, AIC, AIC_c, AIC_d, AIC_w) which compares all upgma-generated partitions.

Usage

upgma_model_selection(Data, Time, Factor, Tags, Colonization = 1, Extinction = 1, Detectability_Value = 0.5, Phi_Time_0_Value = 0.5, Tol = 1e-08, MIT = 100, C_MAX = 10, C_min = 0, E_MAX = 10, E_min = 0, D_MAX = 0.99, D_min = 0, P_MAX = 0.99, P_min = 0.01, I_0 = 0, I_1 = 1, I_2 = 2, I_3 = 3, z = 2, Verbose = 0, MV_FLAG = 0.1, PerfectDetectability = TRUE)

Arguments

Data                   data frame containing presence data per time (in cols) and sites (in rows)
Time                   an array of length n containing sampling times
Factor                 column number containing the ‘data frame’ factor used to split total data into level-based groups
Tags                   array of factor level names: name[i] is the level tag (short name) for the i-th level.
Colonization           guess value to initiate search / parameter value
Extinction             guess value to initiate search / parameter value
Detectability_Value    guess value to initiate search / parameter value
Phi_Time_0_Value       guess value to initiate search / parameter value
Tol                    stopping criteria of the search algorithm.
MIT                    max number of iterations of the search algorithm.
C_MAX                  max value for the possible range of colonization values
C_min                  min value for the possible range of colonization values
E_MAX                  max value for the possible range of colonization values
E_min                  min value for the possible range of colonization values
D_MAX                  max value for the possible range of colonization values
D_min                  min value for the possible range of colonization values
P_MAX                  max value for the possible range of colonization values
**P_min**  
min value for the possible range of colonization values

**I_0**  
has to be 0, 1, 2, or 3. Defaults to 0

**I_1**  
has to be 0, 1, 2, or 3. Defaults to 1

**I_2**  
has to be 0, 1, 2, or 3. Defaults to 2

**I_3**  
has to be 0, 1, 2, or 3. Defaults to 3

**z**  
dimension of the parameter subspace for which the optimization process will take place. Default is 2

**Verbose**  
more/less (1/0) information about the optimization algorithm will be printed out.

**MV_FLAG**  
missing value flag (to specify sites and times where no sample exists)

**PerfectDetectability**

TRUE means 'Perfect Detectability'. Of course, FALSE means 'Imperfect Detectability'

**Details**

The output matrix contains a row for the S different binary partitions of the set of S groups. Searches are conducted using Nelder-Mead simplex method in a bounded parameter space which means that in case a neg loglikelihood (NLL) evaluation is called out from these boundaries, the returned value for this NLL evaluation is artificially given as the maximum number the machine can hold. The input is a data frame containing presence data per time (in cols) and sites (in rows). Different factors (for instance, OTU, location, etc) can slide the initial data frame in their different levels, accordingly. Each initial group (usually, species, OUTs, factors, ...) is named by a short-length-character label (ideally, 3 or 4 characters). The length of Tags array should match the number of levels in which the given factor is subdivided. All labels should have the same character length to fulfill memory alignment requirement of the shared object called by .C(...) function. I_0, I_1, I_2, and I_3 are model parameter keys. They are used to define a 4D-vector (Index). The model parameter keys correspond to the colonization (0), extinction (1), detectability (2), and Phi_0 (3) model parameters in case detectability is imperfect or, alternatively, only colonization (0) and extinction (1) in case detectability is perfect. For instance, if (I_0, I_1) is (1, 0), searches will take place within the parameter space defined by extinction, as the first axis, and colonization, as the second.

**Examples**

```r
## Not run:
Data <- lakshadweepPLUS[[1]]
Guild_Tag = c("Alg", "Cor", "Mac", "Mic", "Omn", "Pis", "Zoo")
R <- upgma_model_selection(Data, Time, Factor = 3, Tags = Guild_Tag,
PerfectDetectability = FALSE, z = 4)
Guild_Tag = c("Agt", "Kad", "Kvt")
R <- upgma_model_selection(Data, Time, Factor = 2, Tags = Guild_Tag,
PerfectDetectability = FALSE, z = 4)
```

## End(Not run)
Description
weight_of_evidence calculates the weight of evidence of a set of nested models.

Usage
weight_of_evidence(data)

Arguments
data A dataframe with the names of the models in the first column and their AIC values in the second column.

Details
Calculates the weight of evidence in favor of model i being the actual Kullback-Leibler best model given a set of models R for your data.

$$w_i = \frac{\exp(-1/2 \cdot \Delta_i)}{\sum \exp(-1/2 \cdot \Delta_r)}$$

$$r = 1, R$$

Value
A dataframe with the names of the analysed models, their AIC differences with respect to the best model and the $w_i$ of each one.

References

See Also
akaikeic

Examples
models <- c("Best_3k", "Best_4k", "Best_5k", "Best_6k", "Best_7k", "Best_8k", "Best_9k")

aks <- c(2977.852, 2968.568, 2957.384, 2952.618, 2949.128, 2947.038, 2943.480)

weight_of_evidence(cbind(models, aks))
Index

*Topic Akaike, upgma_model_selection, 23
*Topic MacKenzie, sss_cedp, 22
*Topic colonization, mss_cedp, 15
  sss_cedp, 22
  upgma_model_selection, 23
*Topic detectability, upgma_model_selection, 23
*Topic extinction, upgma_model_selection, 23
*Topic extinction
  mss_cedp, 15
  sss_cedp, 22
*Topic likelihood, mss_cedp, 15
*Topic maximum
  mss_cedp, 15
*Topic mle, sss_cedp, 22
*Topic upgma, upgma_model_selection, 23
akaikeic, 2, 26
akaikeicc (akaikeic), 2
all_environmental_fit, 3, 17
alonso15, 4
cetotrans, 5, 7
custom_environmental_fit
  (all_environmental_fit), 3
data_generation, 6
greedy_environmental_fit
  (all_environmental_fit), 3
ibd_models, 7
idaho, 8
irregular_multiple_datasets, 9, 12, 18
irregular_single_dataset, 10, 11, 18
island, 12
island-package (island), 12
lakshadweep, 13
lakshadweepPLUS, 14
mss_cedp, 15
NLL_env (all_environmental_fit), 3
NLL_imd (irregular_multiple_datasets), 9
NLL_isd (irregular_single_dataset), 11
NLL_rss (regular_sampling_scheme), 18
null_model (r_squared), 19
PA_simulation (data_generation), 6
r_squared, 19
rates_calculator, 4, 17
regular_sampling_scheme, 10, 12, 18
simberloff, 20
simulated_model (r_squared), 19
sss_cedp, 22
upgma_model_selection, 23
weight_of_evidence, 3, 26