Package ‘mobr’

February 10, 2021

Title Measurement of Biodiversity

Version 2.0.2

Date 2021-02-09

Description Functions for calculating metrics for the measurement biodiversity and its changes across scales, treatments, and gradients. The methods implemented in this package are described in:

Depends R (>= 3.5.0)

Imports plotrix, scales, dplyr, purrr, tidyr, pbapply, ggplot2, egg, tibble, vctrs, rlang, geosphere

Suggests knitr, rmarkdown, testthat

License MIT + file LICENSE

LazyData true

RoxygenNote 7.1.1

Encoding UTF-8

VignetteBuilder knitr

NeedsCompilation no

Author Daniel McGlinn [aut, cre],
Xiao Xiao [aut],
Brian McGill [aut],
Felix May [aut],
Thore Engel [aut],
Caroline Oliver [aut],
Shane Blowes [aut],
Tiffany Knight [aut],
Oliver Purschke [aut],
Nicholas Gotelli [aut],
Jon Chase [aut]

Maintainer Daniel McGlinn <danmcglinn@gmail.com>

Repository CRAN

Date/Publication 2021-02-10 06:30:02 UTC
R topics documented:

- avg_nn_dist ........................................... 2
- calc_biodiv ........................................... 3
- calc_chao1 ........................................... 5
- calc_PIE .................................................. 6
- compare_samp_rarefaction ......................... 7
- fire_comm ............................................. 8
- get_delta_stats .......................................... 8
- get_mob_stats .......................................... 11
- get_null_comm ......................................... 15
- inv_comm ............................................... 16
- kNCN_average ......................................... 17
- make_mob_in .......................................... 18
- mobr ..................................................... 20
- plot.mob_out ........................................... 20
- plot.mob_stats ......................................... 22
- plot_abu ............................................... 23
- plot_N ................................................... 24
- plot_rarefaction ......................................... 25
- rarefaction .............................................. 26
- subset.mob_in .......................................... 29
- tank_comm ............................................. 30

Index

avg_nn_dist ........................................... 2

Description

This function computes the average distance of the next nearest sample for a given set of coordinates. This method of sampling is used by the function `rarefaction` when building the spatial, sample-based rarefaction curves (sSBR).

Usage

```
avg_nn_dist(coords)
```

Arguments

- `coords` a matrix with n-dimensional coordinates

Value

a vector of average distances for each sequential number of accumulated nearest samples.
calc_biodiv

Examples

# transect spatial arrangement
transect = 1:100
avg_nn_dist(transect)
grid = expand.grid(1:10, 1:10)
avg_nn_dist(grid)
oldpar <- par(no.readonly = TRUE)
par(mfrow=c(1,2))
plot(avg_nn_dist(transect), type='o', main='transect',
     xlab='# of samples', ylab='average distance')
# 2-D grid spatial arrangement
plot(avg_nn_dist(grid), type='o', main='grid',
     xlab='# of samples', ylab='average distance')
par(oldpar)

calc_biodiv  Calculate biodiversity statistics from sites by species table.

Description

Calculate biodiversity statistics from sites by species table.

Usage

calc_biodiv(
    abund_mat,
    groups,
    index,
    effort,
    extrapolate,
    return_NA,
    rare_thres
)

Arguments

abund_mat  Sites by species table with species abundances in the respective cells
groups  Vector with group labels for the sites. The length of the vector has to correspond
to the number of rows of the sites by species table.
index  The calculated biodiversity indices. The options are

type 'N' ... Number of individuals (total abundance)
    'S' ... Number of species
    'S_n' ... Rarefied or extrapolated number of species for n individuals
    'S_asym' ... Estimated asymptotic species richness
    'f_0' ... Estimated number of undetected species
    'pct_rare' ... The percent of species with abundances below rare_thres
calc_biodiv

- PIE ... Hurlbert's PIE (Probability of Interspecific Encounter)
- S_PIE ... Effective number of species based on PIE

See the documentation of `get_mob_stats` for further details on the biodiversity indices.

effort

The standardized number of individuals used for the calculation of rarefied species richness. This can be a single value or an integer vector.

extrapolate

Boolean which specifies if richness should be extrapolated when effort is larger than the number of individuals using the chao1 method.

return_NA

Boolean in which the rarefaction function returns the observed S when effort is larger than the number of individuals. If set to TRUE then NA is returned. Note that this argument is only relevant when extrapolate = FALSE.

rare_thres

The threshold that determines how pct_rare is computed. It can range from (0, 1] and defaults to 0.05 which specifies that any species with less than or equal to 5 considered rare. It can also be specified as "N/S" which results in using average abundance as the threshold which McGill (2011) found to have the best small sample behavior.

Details

This function is primarily intended as auxiliary function used in `get_mob_stats`, but can be also used directly for data exploration.

Value

A data.frame with four columns:

- group ... Group label for sites
- index ... Name of the biodiversity index
- effort ... Sampling effort for rarefied richness (NA for the other indices)
- value ... Value of the biodiversity index

Author(s)

Felix May and Dan McGlinn

References

calc_chao1

Estimation of species richness

**Description**

calc_chao1 estimates the number of species at the asymptote ($S_{asymp}$) of the species accumulation curve based on the methods proposed in Chao (1984, 1987, 2005).

**Usage**

calc_chao1(x)

**Arguments**

- `x`: a vector of species abundances or a site-by-species matrix

**Details**

This function is a trimmed version of `iNext::ChaoRichess`. T. C. Hsieh, K. H. Ma and Anne Chao are the original authors of the `iNEXT` package.

**Value**

a vector of species richness estimates

**References**


**Examples**

data(inv_comm)
calc_chao1(inv_comm)
calc_PIE

Calculate probability of interspecific encounter (PIE)

Description

calc_PIE returns the probability of interspecific encounter (PIE) which is also known as Simpson’s evenness index and Gini-Simpson index. For ENS=TRUE, PIE will be converted to an asymptotic effective number of species (S_PIE).

Usage

calc_PIE(x, ENS = FALSE)

Arguments

x

can either be a: 1) mob_in object, 2) community matrix-like object in which rows represent plots and columns represent species, or 3) a vector which contains the abundance of each species.

ENS

Boolean that determines if the effective number of species should be returned or the raw PIE value. Defaults to FALSE

Details

The formula of Hurlbert (1971) is used to calculate PIE:

\[ PIE = N / (N - 1) \ast (1 - p_i^2) \]

where N is the total number of individuals and \( p_i \) is the relative abundance of species i. This formulation uses sampling without replacement and it is sometimes referred to as the bias corrected formulation of PIE.

For ENS = TRUE, S_PIE will be returned which represents the species richness of a hypothetical community with equally-abundant species and infinitely many individuals corresponding to the observed value of PIE. It is computed as \( S_{PIE} = 1 / (1 - PIE) \), which is equal to the asymptotic estimator for Hill numbers of diversity order 2 provided by Chao et al (2014). Note that S_PIE is undefined for communities with exactly one individual per species.

The code in this function borrows heavily from the function vegan::diversity() but computes a different quantity. The function vegan::diversity() computes PIE when sampling with replacement is assumed. The difference between the two formulations will decrease as N becomes large. Jari Oksanen and Bob O’Hara are the original authors of the function vegan::diversity().

Author(s)

Dan McGlinn, Thore Engel
References


Examples

data(inv_comm)
calc_PIE(inv_comm)
calc_PIE(inv_comm, ENS=TRUE)

```
compare_samp_rarefaction
  Compare all sample-based curves (random, spatially constrained-k-NN, spatially constrained-k-NCN)
```

Description
This is just plotting all curves.

Usage

compare_samp_rarefaction(x)

Arguments

x a mob_in object

Value

a plot

Examples

```
data(inv_comm)
data(inv_plot_attr)
inv_mob_in = make_mob_in(inv_comm, inv_plot_attr, coord_names = c('x', 'y'))
compare_samp_rarefaction(inv_mob_in)
```
fire_comm

Fire data set

Description
Woody plant species counts in burned and unburned forest sites in the Missouri Ozarks, USA.

Details
fire_comm is a site-by-species matrix with individual counts.
fire_plot_attr is a data frame with corresponding site variables. The column group specifies whether a site is "burned" or "unburned". This variable is considered a "treatment" in the mob framework. The columns x and y contain the spatial coordinates of the sites.

The data were adapted from Myers et al (2015).

References

Examples
data(fire_comm)
data(fire_plot_attr)
fire_mob_in = make_mob_in(fire_comm, fire_plot_attr)

get_delta_stats
Conduct the MoB tests on drivers of biodiversity across scales.

Description
There are three tests, on effects of 1. the shape of the SAD, 2. treatment/group-level density, 3. degree of aggregation. The user can specifically to conduct one or more of these tests.

Usage
get_delta_stats(
    mob_in,
    env_var,
    group_var = NULL,
    ref_level = NULL,
    tests = c("SAD", "N", "agg"),
    spat_algo = NULL,
    type = c("continuous", "discrete"),
    spat_dims = c(10, 10)
)
get_delta_stats

stats = NULL,
inds = NULL,
log_scale = FALSE,
min_plots = NULL,
density_stat = c("mean", "max", "min"),
n_perm = 1000,
overall_p = FALSE
)

Arguments

mob_in         an object of class mob_in created by make_mob_in()

env_var        a character string specifying the environmental variable in mob_in$env to be
               used for explaining the change in richness

group_var      an optional character string in mob_in$env which defines how samples are pooled.
               If not provided then each unique value of the argument env_var is used define
               the groups.

ref_level      a character string used to define the reference level of env_var to which all other
               groups are compared with. Only makes sense if env_var is a factor (i.e. type
               == 'discrete')

tests          specifies which one or more of the three tests ('SAD', 'N', 'agg') are to be per-
               formed. Default is to include all three tests.

spat_algo      character string that can be either: 'kNN' or 'kNCN' for k-nearest neighbor and
               k-nearest centroid neighbor sampling respectively. It defaults to k-nearest neigh-
               bor which is a more computationally efficient algorithm that closely approxi-
               mates the potentially more correct k-NCN algo (see Details of ?rarefaction).

type           "discrete" or "continuous". If "discrete", pair-wise comparisons are conducted
               between all other groups and the reference group. If "continuous", a correlation
               analysis is conducted between the response variables and env_var.

stats          a vector of character strings that specifies what statistics to summarize effect
               sizes with. Options include: c('betas', 'r2', 'r2adj', 'f', 'p') for the beta-
               coefficients, r-squared, adjusted r-squared, F-statistic, and p-value respectively.
               The default value of NULL will result in only betas being calculated when type
               == 'discrete' and all possible stats being computed when type == 'continuous'.
               Note that for a discrete analysis all non-betas stats are meaningless because the
               model has zero degrees of freedom in this context.

inds           effort size at which the individual-based rarefaction curves are to be evaluated,
               and to which the sample-based rarefaction curves are to be interpolated. It can
               take three types of values, a single integer, a vector of integers, and NULL. If
               inds = NULL (the default), the curves are evaluated at every possible effort size,
               from 1 to the total number of individuals within the group (slow). If inds is a
               single integer, it is taken as the number of points at which the curves are evalu-
               ated; the positions of the points are determined by the "log_scale" argument. If
               inds is a vector of integers, it is taken as the exact points at which the curves are
               evaluated.
log_scale if "inds" is given a single integer, "log_scale" determines the position of the points. If log_scale is TRUE, the points are equally spaced on logarithmic scale. If it is FALSE (default), the points are equally spaced on arithmetic scale.

min_plots minimal number of plots for test 'agg', where plots are randomized within groups as null test. If it is given a value, all groups with fewer plots than min_plot are removed for this test. If it is NULL (default), all groups are kept. Warnings are issued if 1. there is only one group left and "type" is discrete, or 2. there are less than three groups left and "type" is continuous, or 3. reference group ("ref_group") is removed and "type" is discrete. In these three scenarios, the function will terminate. A different warning is issued if any of the remaining groups have less than five plots (which have less than 120 permutations), but the test will be carried out.

density_stat reference density used in converting number of plots to numbers of individuals, a step in test "N". It can take one of the three values: "mean", "max", or "min". If it is "mean", the average plot-level abundance across plots (all plots when "type" is "continuous, all plots within the two groups for each pair-wise comparison when "type" is "discrete") are used. If it is "min" or "max", the minimum/maximum plot-level density is used.

n_perm number of iterations to run for null tests, defaults to 1000.

overall_p Boolean defaults to FALSE specifies if overall across scale p-values for the null tests. This should be interpreted with caution because the overall p-values depend on scales of measurement yet do not explicitly reflect significance at any particular scale.

Value a "mob_out" object with attributes

Author(s)
Dan McGlinn and Xiao Xiao

See Also
rarefaction

Examples
data(inv_comm)
data(inv_plot_attr)
inv_mob_in = make_mob_in(inv_comm, inv_plot_attr, coord_names = c('x', 'y'))
inv_mob_out = get_delta_stats(inv_mob_in, 'group', ref_level='uninvaded',
                             type='discrete', log_scale=TRUE, n_perm=3)
plot(inv_mob_out)
get_mob_stats

**Calculate sample based and group based biodiversity statistics.**

**Description**

Calculate sample based and group based biodiversity statistics.

**Usage**

```r
get_mob_stats(
  mob_in,
  group_var,
  ref_level = NULL,
  index = c("N", "S", "S_n", "S_PIE"),
  effort_samples = NULL,
  effort_min = 5,
  extrapolate = TRUE,
  return_NA = FALSE,
  rare_thres = 0.05,
  n_perm = 199,
  boot_groups = FALSE,
  conf_level = 0.95,
  cl = NULL,
  ...
)
```

**Arguments**

- `mob_in` an object of class `mob_in` created by `make_mob_in()`
- `group_var` String that specifies which field in `mob_in$env` the data should be grouped by
- `ref_level` String that defines the reference level of `group_var` to which all other groups are compared with, defaults to `NULL`. If `NULL` then the default contrasts of `group_var` are used.
- `index` The calculated biodiversity indices. The options are
  - N ... Number of individuals (total abundance)
  - S ... Number of species
  - S_n ... Rarefied or extrapolated number of species for n individuals
  - S_asympt ... Estimated asymptotic species richness
  - f_0 ... Estimated number of undetected species
  - pct_rare ... The percent of rare species as defined by `rare_thres`
  - PIE ... Hurlbert's PIE (Probability of Interspecific Encounter)
  - S_PIE ... Effective number of species based on PIE

If `index` is not specified then `N, S, S_n, pct_rare,` and `S_PIE` are computed by default. See Details for additional information on the biodiversity statistics.
effort_samples: The standardized number of individuals used for the calculation of rarefied species richness at the alpha-scale. This can be a single value or an integer vector. As default the minimum number of individuals found across the samples is used, when this is not smaller than effort_min.

effort_min: The minimum number of individuals considered for the calculation of rarefied richness (Default value of 5). Samples with less individuals then effort_min are excluded from the analysis with a warning. Accordingly, when effort_samples is set by the user it has to be higher than effort_min.

extrapolate: Boolean which specifies if richness should be extrapolated when effort_samples is larger than the number of individuals using the chao1 method. Defaults to TRUE.

return_NA: Boolean defaults to FALSE in which the rarefaction function returns the observed S when effort is larger than the number of individuals. If set to TRUE then NA is returned. Note that this argument is only relevant when extrapolate = FALSE.

rare_thres: The threshold that determines how pct_rare is computed. It can range from (0, 1] and defaults to 0.05 which specifies that any species with less than or equal to 5 considered rare. It can also be specified as "N/S" which results in using average abundance as the threshold which McGill (2011) found to have the best small sample behavior.

n_perm: The number of permutations to use for testing for treatment effects. Defaults to 199.

boot_groups: Use bootstrap resampling within groups to derive gamma-scale confidence intervals for all biodiversity indices. Default is FALSE. See Details for information on the bootstrap approach.

conf_level: Confidence level used for the calculation of gamma-scale bootstrapped confidence intervals. Only used when boot_groups = TRUE.

c1: A cluster object created by makeCluster, or an integer to indicate number of child-processes (integer values are ignored on Windows) for parallel evaluations (see Details on performance).

... Optional arguments to FUN.

Details

BIODIVERSITY INDICES

S_n: Rarefied species richness is the expected number of species, given a defined number of sampled individuals (n) (Gotelli & Colwell 2001). Rarefied richness at the alpha-scale is calculated for the values provided in effort_samples as long as these values are not smaller than the user-defined minimum value effort_min. In this case the minimum value is used and samples with less individuals are discarded. When no values for effort_samples are provided the observed minimum number of individuals of the samples is used, which is the standard in rarefaction analysis (Gotelli & Colwell 2001). Because the number of individuals is expected to scale linearly with sample area or effort, at the gamma-scale the number of individuals for rarefaction is calculated as the minimum number of samples within groups multiplied by effort_samples. For example, when there are 10 samples within each group, effort_groups equals 10 * effort_samples. If n
is larger than the number of individuals in sample and extrapolate = TRUE then the Chao1 (Chao 1984, Chao 1987) method is used to extrapolate the rarefaction curve.

**pct_rare: Percent of rare species** Is the ratio of the number of rare species to the number of observed species x 100 (McGill 2011). Species are considered rare in a particular sample if they have fewer individuals than rare_thres * N where rare_thres can be set by the user and N is the total number of individuals in the sample. The default value of rare_thres of 0.05 is arbitrary and was chosen because McGill (2011) found this metric of rarity performed well and was generally less correlated with other common metrics of biodiversity. Essentially this metric attempt to estimate what proportion of the species in the same occur in the tail of the species abundance distribution and is therefore sensitive to presence of rare species.

**S_asympt: Asymptotic species richness** is the expected number of species given complete sampling and here it is calculated using the Chao1 estimator (Chao 1984, Chao 1987) see calc_chao1. Note: this metric is typically highly correlated with S (McGill 2011).

**f_0: Undetected species richness** is the number of undetected species or the number of species observed 0 times which is an indicator of the degree of rarity in the community. If there is a greater rarity then f_0 is expected to increase. This metric is calculated as S_asympt - S. This metric is less correlated with S than the raw S_asympt metric.

**PIE: Probability of intraspecific encounter** represents the probability that two randomly drawn individuals belong to the same species. Here we use the definition of Hurlbert (1971), which considers sampling without replacement. PIE is closely related to the well-known Simpson diversity index, but the latter assumes sampling with replacement.

**S_PIE: Effective number of species for PIE** represents the effective number of species derived from the PIE. It is calculated using the asymptotic estimator for Hill numbers of diversity order 2 (Chao et al, 2014). S_PIE represents the species richness of a hypothetical community with equally-abundant species and infinitely many individuals corresponding to the same value of PIE as the real community. An intuitive interpretation of S_PIE is that it corresponds to the number of dominant (highly abundant) species in the species pool.

For species richness S, rarefied richness S_n, undetected richness f_0, and the Effective Number of Species S_PIE we also calculate beta-diversity using multiplicative partitioning (Whittaker 1972, Jost 2007). That means for these indices we estimate beta-diversity as the ratio of gamma-diversity (total diversity across all plots) divided by alpha-diversity (i.e., average plot diversity).

**PERMUTATION TESTS AND BOOTSTRAP**

For both the alpha and gamma scale analyses we summarize effect size in each biodiversity index by computing D_bar: the average absolute difference between the groups. At the alpha scale the indices are averaged first before computing D_bar.

We used permutation tests for testing differences of the biodiversity statistics among the groups (Legendre & Legendre 1998). At the alpha-scale, one-way ANOVA (i.e. F-test) is implemented by shuffling treatment group labels across samples. The test statistic for this test is the F-statistic which is a pivotal statistic (Legendre & Legendre 1998). At the gamma-scale we carried out the permutation test by shuffling the treatment group labels and using D_bar as the test statistic. We could not use the F-statistic as the test statistic at the gamma scale because at this scale there are no replicates and therefore the F-statistic is undefined.

A bootstrap approach can be used to also test differences at the gamma-scale. When boot_groups = TRUE instead of the gamma-scale permutation test, there will be resampling of samples within groups to derive gamma-scale confidence intervals for all biodiversity indices. The function output
includes lower and upper confidence bounds and the median of the bootstrap samples. Please note that for the richness indices sampling with replacement corresponds to rarefaction to ca. 2/3 of the individuals, because the same samples occur several times in the resampled data sets.

Value

A list of class mob_stats that contains alpha-scale and gamma-scale biodiversity statistics, as well as the p-values for permutation tests at both scales.

When boot_groups = TRUE there are no p-values at the gamma-scale. Instead there is lower bound, median, and upper bound for each biodiversity index derived from the bootstrap within groups.

Author(s)

Felix May and Dan McGlinn

References


Examples

# a binary grouping variable (uninvaded or invaded)
data(inv_comm)
data(inv_plot_attr)
inv_mob_in = make_mob_in(inv_comm, inv_plot_attr, c('x', 'y'))
inv_stats = getMobStats(inv_mob_in, group_var = "group", ref_level = 'uninvaded',
    n_perm = 19, effort_samples = c(5,10))
plot(inv_stats)

# parallel evaluation using the parallel package
# run in parallel
library(parallel)
c1 = makeCluster(2L)
clusterEvalQ(c1, library(mobr))
get_null_comm

clusterExport(cl, 'inv_mob_in')
inv_mob_stats = get_mob_stats(inv_mob_in, 'group', ref_level = 'uninvaded',
n_perm=999, cl=cl)

stopCluster(cl)

get_null_comm Generate a null community matrix

Description

Three null models are implemented that randomize different components of community structure
while keeping other components constant.

Usage

get_null_comm(comm, null_model, groups = NULL)

Arguments

comm community matrix of abundances with plots as rows and species columns.
null_model a string which specifies which null model to use options include: 'rand_SAD',
'rand_N', and 'rand_agg'. See Details for description of each null model.
groups optional argument that is a vector of group ids which specify which group each
site is associated with. If is NULL then all rows of the community matrix are
assumed to be members of the same group

Details

This function implements three different nested null models. They are considered nested because
at the core of each null model is the random sampling with replacement of the relative abundance
distribution (RAD) to generate a random sample of a species abundance distribution (SAD). Here
we describe each null model:

• 'rand_SAD' ... A random SAD is generated using a sample with replacement of individuals
from the species pool proportional to their observed relative abundance. This null model will
produce an SAD that is of a similar functional form to the observed SAD (Green and Plotkin
2007). The total abundance of the random SAD is the same as the observed SAD but overall
species richness will be equal to or less than the observed SAD. This algorithm ignores the
group argument. This sampling algorithm is also used in the two other null models 'rand_N'
and 'rand_agg'.

• 'rand_N' ... The total number of individuals in a plot is shuffled across all plots (within
and between groups). Then for each plot that many individuals are drawn randomly from the
group specific relative abundance distribution with replacement for each plot (i.e., using the
'rand_SAD' algorithm described above. This removes group differences in the total number
of individuals in a given plot, but maintains group level differences in their SADs.
• 'rand_agg'... This null model nullifies the spatial structure of individuals (i.e., their aggregation), but it is constrained by the observed total number of individuals in each plot (in contrast to the 'rand_N' null model), and the group specific SAD (in contrast to the 'rand_SAD' null model). The other two null models also nullify spatial structure. The 'rand_agg' null model is identical to the 'rand_N' null model except that plot abundances are not shuffled.

Replaces depreciated function 'permute_comm'

Value

a site-by-species matrix

References


Examples

S = 3
N = 20
nplots = 4
comm = matrix(rpois(S * nplots, 1), ncol = S, nrow = nplots)
groups = rep(1:2, each=2)
set.seed(1)
get_null_comm(comm, 'rand_SAD')
# null model 'rand_SAD' ignores groups argument
set.seed(1)
get_null_comm(comm, 'rand_SAD', groups)
set.seed(1)
get_null_comm(comm, 'rand_N')
# null model 'rand_N' does not ignore the groups argument
set.seed(1)
get_null_comm(comm, 'rand_N', groups)
# note that the 'rand_agg' null model is constrained by observed plot abundances
noagg = get_null_comm(comm, 'rand_agg', groups)
noagg
rowSums(comm)
rowSums(noagg)

inv_comm

Invasive plants dataset

Description

Herbaceous plant species counts sites invaded and uninvaded by Lonicera maackii (Amur honeysuckle) which is an invasive shrub.
Details

`inv_comm` is a site-by-species matrix with individual counts.
`inv_plot_attr` is a data frame with corresponding site variables. The column group specifies whether a site is "invaded" or "uninvaded". This variable is considered a "treatment" in the mob framework. The columns x and y contain the spatial coordinates of the sites.

The data were adapted from Powell et al (2013).

References


Examples

```r
data(inv_comm)
data(inv_plot_attr)
inv_mob_in = make_mob_in(inv_comm, inv_plot_attr)
```

```
-------------------
kNCN_average

Construct spatially constrained sample-based rarefaction (sSBR) curve using the k-Nearest-Centroid-neighbor (k-NCN) algorithm

-------------------
```

Description

This function accumulates samples according their proximity to all previously included samples (their centroid) as opposed to the proximity to the initial focal sample. This ensures that included samples mutually close to each other and not all over the place.

Usage

```r
kNCN_average(
  x,
  n = NULL,
  coords = NULL,
  repetitions = 1,
  no_pb = TRUE,
  latlong = FALSE,
  cl = NULL
)
```

Arguments

- `x` a mob_in object or a community site x species matrix
- `n` number of sites to include.
- `coords` spatial coordinates of the samples. If x is a mob_in object, the function uses its 'spat' table as coordinates.
make_mob_in

repetitions Number of times to repeat the procedure. Useful in situations where there are many ties in the distance matrix.

no_pb binary, if TRUE then a progress bar is not printed, defaults to TRUE

latlong if longitude latitudes are supplied

c1 A cluster object created by makeCluster, or an integer to indicate number of child-processes (integer values are ignored on Windows) for parallel evaluations (see Details on performance).

Details

Internally the function constructs one curve per sample whereby each sample serves as the initial sample repetition times. Finally, the average curve is returned.

Value

a numeric vector of estimated species richness

Examples

data(inv_comm)
data(inv_plot_attr)
inv_mob_in = make_mob_in(inv_comm, inv_plot_attr, coord_names = c('x', 'y'))
kNCN_average(inv_mob_in, n = 5)

# parallel evaluation using the parallel package
# run in parallel
library(parallel)
c1 = makeCluster(2L)
c1 = clusterEvalQ(cl, library(mobilr))
c1 = clusterExport(c1, 'inv_mob_in')
S_kNCN = kNCN_average(inv_mob_in, cl=c1)

stopCluster(cl)

make_mob_in Create the 'mob_in' object.

Description

The 'mob_in' object will be passed on for analyses of biodiversity across scales.

Usage

make_mob_in(  
  comm,  
  plot_attr,  
  coord_names = NULL,
Arguments

comm  
community matrix in which rows are samples (e.g., plots) and columns are species.

plot_attr  
matrix which includes the environmental attributes and spatial coordinates of the plots. Environmental attributes are mandatory, while spatial coordinates are optional.

coord_names  
character vector with the names of the columns of plot_attr that specify the coordinates of the samples. Defaults to NULL (no coordinates). When providing coordinate names, the order the names are provided matters when working with latitude-longitude coordinates (i.e., argument latlong = TRUE, and it is expected that the column specifying the x-coordinate or the longitude is provided first, y-coordinate or latitude provided second. To provide coordinate names use the following syntax: coord_names = c('longitude_col_name', 'latitude_col_name')

binary  
Boolean, defaults to FALSE. Whether the plot by species matrix "comm" is in abundances or presence/absence.

latlong  
Boolean, defaults to FALSE. Whether the coordinates are latitude-longitudes. If TRUE, distance calculations by downstream functions are based upon great circle distances rather than Euclidean distances. Note latitude-longitudes should be in decimal degree.

Value

a "mob_in" object with four attributes. "comm" is the plot by species matrix. "env" is the environmental attribute matrix, without the spatial coordinates. "spat" contains the spatial coordinates (1-D or 2-D). "tests" specifies whether each of the three tests in the biodiversity analyses is allowed by data.

Author(s)

Dan McGlinn and Xiao Xiao

Examples

data(inv_comm)
data(inv_plot_attr)
inv_mob_in = make_mob_in(inv_comm, inv_plot_attr, coord_names = c('x', 'y'))
Description

The primary aim of this package is to provide ecologist’s tools to examine changes in biodiversity across spatial scales. Additionally, the package provides a method to examine how a factor mediates species richness via its effects on different aspects of community structure: total abundance, species commonness, and spatial aggregation of conspecifics.

plot.mob_out

Plot the multiscale MoB analysis output generated by get_delta_stats.

Usage

```r
## S3 method for class 'mob_out'
plot(
  x, 
  stat = "b1", 
  log2 = "", 
  scale_by = NULL, 
  display = c("S ~ effort", "effect ~ grad", "stat ~ effort"), 
  eff_sub_effort = TRUE, 
  eff_log_base = 2, 
  eff_disp_pts = TRUE, 
  eff_disp_smooth = FALSE, 
  ...
)
```

Arguments

- `x` a mob_out class object
- `stat` a character string that specifies what statistic should be used in the effect size plots. Options include: c('b0', 'b1', 'r', 'r2', 'r2adj', 'f') for the beta-coefficients, person correlation coefficient, r-squared, adjusted r-squared, and F-statistic respectively. If the explanatory variable is a factor then 'b1' is the only reasonable option. The default is set to the regression slope 'b1' because this appears to have the strongest statistical power.
log2

A character string specifying if the x- or y-axis should be rescaled by log base 2. Only applies when display == 'S ~ effort' | 'S ~ effort'. Options include: c('','x','y','xy') for no rescaling, x-axis, y-axis, and both x and y-axes respectively. Default is set to no rescaling.

scale_by

A character string specifying if sampling effort should be rescaled. Options include: NULL, 'indiv', and 'plot' for no rescaling, rescaling to number of individuals, and rescaling to number of plots respectively. The rescaling is carried out using mob_out$density_stat.

display

A string that specifies what graphical panels to display. Options include:
- 'S ~ expl' ... plot of S versus the explanatory variable
- 'S ~ effort' ... plot of S versus sampling effort (i.e., a rarefaction curve)
- 'effect ~ expl' ... plot of agg., N, and SAD effect size versus explanatory variable
- 'stat ~ effort' ... plot of summary statistic versus sampling effort

Defaults to 'S ~ effort', 'effect ~ expl', and 'stat ~ effort'.

eff_sub_effort

Boolean which determines if only a subset of efforts will be considered in the plot of effect size (i.e., when display = 'effect ~ expl'. Defaults to TRUE to declutter the plots.

eff_log_base

A positive real number that determines the base of the logarithm that efforts were be distributed across, the larger this number the fewer efforts will be displayed.

eff_disp_pts

Boolean to display the raw effect points, defaults to TRUE

eff_disp_smooth

Boolean to display the regressions used to summarize the linear effect of the explanatory variable on the effect sizes, defaults to FALSE

... parameters passed to other functions

Value

Plots the effect of the SAD, the number of individuals, and spatial aggregation on the difference in species richness.

Author(s)

Dan McGlinn and Xiao Xiao

Examples

data(inv_comm)
data(inv_plot_attr)
inv_mob_in = make.mob.in(inv_comm, inv_plot_attr, coord_names = c('x', 'y'))
inv_mob_out = get_delta_stats(inv_mob_in, 'group', ref_level='uninvaded',
                              type='discrete', log_scale=TRUE, n_perm=4)
plot(inv_mob_out, 'b1')
plot(inv_mob_out, 'b1', scale_by = 'indiv')
### Description

Plots a `mob_stats` object which is produced by the function `get_mob_stats`. The p-value for each statistic is displayed in the plot title if applicable.

### Usage

```r
## S3 method for class 'mob_stats'
plot(
x, 
index = NULL,
multi_panel = FALSE,
col = c("#FFB3B5", "#78D3EC", "#6BDABD", "#C5C0FE", "#E2C288", "#F7B0E6", "#AAD28C"),
cex.axis = 1.2,
...)
```

### Arguments

- `x`: a `mob_stats` object that has the samples and treatment level statistics
- `index`: The biodiversity statistics that should be plotted. See `get_mob_stats` for information on the indices. By default there is one figure for each index, with panels for alpha- and gamma-scale results as well as for beta-diversity when applicable.
- `multi_panel`: A logical variable. If `multi_panel = TRUE` then a multipanel plot is produced, which shows observed, rarefied, and asymptotic species richness and S_PIE at the alpha- and gamma-scale. This set of variables conveys a comprehensive picture of the underlying biodiversity changes.
- `col`: a vector of colors for the groups, set to NA if no color is preferred
- `cex.axis`: The magnification to be used for axis annotation relative to the current setting of `cex`. Defaults to 1.2.
- `...`: additional arguments to provide to `boxplot`, `points`, and confidence interval functions

### Details

The user may specify which results to plot or simply to plot all the results.

### Author(s)

Felix May, Xiao Xiao, and Dan McGlinn
Examples

```r
data(inv_comm)
data(inv_plot_attr)
inv_mob_in = make_mob_in(inv_comm, inv_plot_attr)
# without bootstrap CI for gamma-scale
inv_stats = get_mob_stats(inv_mob_in, group_var = "group", n_perm = 20)
plot(inv_stats)
# with bootstrap CI for gamma-scale
inv_stats_boot = get_mob_stats(inv_mob_in, group_var = "group", n_perm = 20,
    boot_groups=TRUE)
plot(inv_stats_boot)
```

---

**plot_abu**  
*Plot distributions of species abundance*

**Description**
Plot distributions of species abundance

**Usage**

```r
plot_abu(
    mob_in,
    group_var,
    ref_level = NULL,
    type = c("sad", "rad"),
    pooled = FALSE,
    col = NULL,
    lwd = 3,
    log = "",
    leg_loc = "topleft"
)
```

**Arguments**
- `mob_in`  
a `mob_in` class object produced by `make_mob_in`
- `group_var`  
String that specifies which field in `mob_in$env` the data should be grouped by
- `ref_level`  
String that defines the reference level of `group_var` to which all other groups are compared with, defaults to NULL. If NULL then the default contrasts of `group_var` are used.
- `type`  
either 'sad' or 'rad' for species abundance vs rank abundance distribution
- `pooled`  
Boolean defaults to FALSE which specifies that abundances should not be pooled at the group level, TRUE specifies that they should be pooled
- `col`  
optional vector of colors.
- `lwd`  
a number which specifies the width of the lines
plot_N

Plot the relationship between the number of plots and the number of individuals

Description

The MoB methods assume a linear relationship between the number of plots and the number of individuals. This function provides a means of verifying the validity of this assumption.

Usage

plot_N(comm, n_perm = 1000)

Arguments

comm 
community matrix with sites as rows and species as columns

n_perm 
number of permutations to average across, defaults to 1000

Author(s)

Dan McGlinn

Examples

data(inv_comm)
plot_N(inv_comm)
**plot_rarefaction**  

*Plot rarefaction curves for each treatment group*

**Description**

Plot rarefaction curves for each treatment group

**Usage**

```r
plot_rarefaction(
  mob_in,  
  group_var,  
  ref_level = NULL,  
  method,  
  dens_ratio = 1,  
  pooled = TRUE,  
  spat_algo = NULL,  
  col = NULL,  
  lwd = 3,  
  log = "",  
  leg_loc = "topleft",  
  ...
)
```

**Arguments**

- `mob_in`  
  an object of class mob_in created by make_mob_in()

- `group_var`  
  String that specifies which field in `mob_in$env` the data should be grouped by

- `ref_level`  
  String that defines the reference level of `group_var` to which all other groups are compared with, defaults to `NULL`. If `NULL` then the default contrasts of `group_var` are used.

- `method`  
  a character string that specifies the method of rarefaction curve construction it can be one of the following:
  - 'IBR' ... individual-based rarefaction in which species are accumulated by randomly sampling individuals
  - 'SBR' ... sample-based rarefaction in which species are accumulated by randomly sampling samples (i.e., plots). Note that within plot spatial aggregation is maintained with this approach. Although this curve is implemented here, it is not used in the current version of the MoB framework
  - 'nsSBR' ... non-spatial, sampled-based rarefaction in which species are accumulated by randomly sampling samples that represent a spatially random sample of individuals (i.e., no with-in plot spatial aggregation). The argument `dens_ratio` must also be set otherwise this sampling results in a curve identical to the IBR (see Details).
  - 'sSBR' ... spatial sample-based rarefaction in which species are accumulated by including spatially proximate samples first.
rarefaction

dens_ratio  the ratio of individual density between a reference group and the community data (i.e., x) under consideration. This argument is used to rescale the rarefaction curve when estimating the effect of individual density on group differences in richness.

pooled  Boolean specifying if samples should be pooled at the group level or not. Defaults to TRUE. This argument only applies when the individual based rarefaction is used (i.e., method = 'indiv')

spat_algo  character string that can be either: 'kNN' or 'kNCN' for k-nearest neighbor and k-nearest centroid neighbor sampling respectively. It defaults to k-nearest neighbor which is a more computationally efficient algorithm that closely approximates the potentially more correct k-NCN algo (see Details).

col  optional vector of colors.

lwd  a number which specifies the width of the lines

log  a string that specifies if any axes are to be log transformed, options include 'x', 'y' or 'xy' in which either the x-axis, y-axis, or both axes are log transformed respectively

leg_loc  a string that specifies the location of the legend, options include: 'lowerleft', 'topleft', 'loweright', 'topright'

...  other arguments to provide to rarefaction

Examples

data(inv_comm)
data(inv_plot_attr)
inv.mob.in = make.mob.in(inv_comm, inv_plot_attr, coord_names = c('x', 'y'))
# random individual based rarefaction curves
plot_rarefaction(inv.mob.in, 'group', 'uninvaded', 'IBR', pooled=TRUE, leg_loc='bottomright')
plot_rarefaction(inv.mob.in, 'group', 'uninvaded', 'IBR', pooled=FALSE, log='x')
# random sample based rarefaction curves
plot_rarefaction(inv.mob.in, 'group', 'uninvaded', 'SBR', log='xy')
# spatial sample based rarefaction curves
plot_rarefaction(inv.mob.in, 'group', 'uninvaded', 'sSBR', log='xy')

rarefaction  Rarefied Species Richness

Description

The expected number of species given a particular number of individuals or samples under random and spatially explicit nearest neighbor sampling
rarefaction

Usage

rarefaction(
  x,
  method,
  effort = NULL,
  coords = NULL,
  latlong = NULL,
  dens_ratio = 1,
  extrapolate = FALSE,
  return_NA = FALSE,
  quiet_mode = FALSE,
  spat_algo = NULL
)

Arguments

x can either be a: 1) mob_in object, 2) community matrix-like object in which rows represent plots and columns represent species, or 3) a vector which contains the abundance of each species.

method a character string that specifies the method of rarefaction curve construction it can be one of the following:

- 'IBR' ... individual-based rarefaction in which species are accumulated by randomly sampling individuals
- 'SBR' ... sample-based rarefaction in which species are accumulated by randomly sampling samples (i.e., plots). Note that within plot spatial aggregation is maintained with this approach. Although this curve is implemented here, it is not used in the current version of the MoB framework
- 'nsSBR' ... non-spatial, sampled-based rarefaction in which species are accumulated by randomly sampling samples that represent a spatially random sample of individuals (i.e., no within plot spatial aggregation). The argument dens_ratio must also be set otherwise this sampling results in a curve identical to the IBR (see Details).
- 'sSBR' ... spatial sample-based rarefaction in which species are accumulated by including spatially proximate samples first.

effort optional argument to specify what number of individuals or number of samples depending on method to compute rarefied richness as. If not specified all possible values from 1 to the maximum sampling effort are used.

coords an optional matrix of geographic coordinates of the samples. Only required when using the spatial rarefaction method and this information is not already supplied by x. The first column should specify the x-coordinate (e.g., longitude) and the second coordinate should specify the y-coordinate (e.g., latitude).

latlong Boolean if coordinates are latitude-longitude decimal degrees

dens_ratio the ratio of individual density between a reference group and the community data (i.e., x) under consideration. This argument is used to rescale the rarefaction curve when estimating the effect of individual density on group differences in richness.
extrapolate  Boolean which specifies if richness should be extrapolated when effort is larger than the number of individuals using the chao1 method. Defaults to FALSE in which case it returns observed richness. Extrapolation is only implemented for individual-based rarefaction (i.e., method = 'indiv').

return_NA  Boolean defaults to FALSE in which the function returns the observed S when effort is larger than the number of individuals or number of samples (depending on the method of rarefaction). If set to TRUE then NA is returned. Note that this argument is only relevant when extrapolate = FALSE.

quiet_mode  Boolean defaults to FALSE, if TRUE then warnings and other non-error messages are suppressed.

spat_algo  character string that can be either: ‘kNN’ or ‘kNCN’ for k-nearest neighbor and k-nearest centroid neighbor sampling respectively. It defaults to k-nearest neighbor which is a more computationally efficient algorithm that closely approximates the potentially more correct k-NCN algo (see Details).

Details

The analytical formulas of Cayuela et al. (2015) are used to compute the random sampling expectation for the individual and sampled based rarefaction methods. The spatially constrained rarefaction curve (Chiarucci et al. 2009) also known as the sample-based accumulation curve (Gotelli and Colwell 2001) can be computed in one of two ways which is determined by the argument spat_algo. In the kNN approach each plot is accumulated by the order of their spatial proximity to the original focal cell. If plots have the same distance from the focal plot then one is chosen randomly to be sampled first. In the kNCN approach, a new centroid is computed after each plot is accumulated, then distances are recomputed from that new centroid to all other plots and the next nearest is sampled. The kNN is faster because the distance matrix only needs to be computed once, but the sampling of kNCN which simultaneously minimizes spatial distance and extent is more similar to an actual person searching a field for species. For both kNN and kNCN, each plot in the community matrix is treated as a starting point and then the mean of these n possible accumulation curves is computed. For individual-based rarefaction if effort is greater than the number of individuals and extrapolate = TRUE then the Chao1 method is used (Chao 1984, 1987). The code used to perform the extrapolation was ported from iNext::D0.hat found at https://github.com/JohnsonHsieh/iNEXT. T. C. Hsieh, K. H. Ma and Anne Chao are the original authors of the iNEXT package.

If effort is greater than sample size and extrapolate = FALSE then the observed number of species is returned.

Value

A vector of rarefied species richness values

Author(s)

Dan McGlinn and Xiao Xiao

References


Examples

data(inv_comm)
data(inv_plot_attr)
sad = colSums(inv_comm)
inv_mob_in = make_mob_in(inv_comm, inv_plot_attr, coord_names = c('x', 'y'))
# rarefaction can be performed on different data inputs
# all three give same answer
# 1) the raw community site-by-species matrix
rarefaction(inv_comm, method='IBR', effort=1:10)
# 2) the SAD of the community
rarefaction(inv_comm, method='IBR', effort=1:10)
# 3) a mob_in class object
# rescaling of individual based rarefaction
# when the density ratio is 1 the richness values are
# identical to unscale rarefaction
rarefaction(inv_comm, method='IBR', effort=1:10, dens_ratio=1)
# however the curve is either shrunk when density is higher than
# the reference value (i.e., dens_ratio < 1)
rarefaction(inv_comm, method='IBR', effort=1:10, dens_ratio=0.5)
# the curve is stretched when density is lower than the
# reference value (i.e., dens_ratio > 1)
rarefaction(inv_comm, method='IBR', effort=1:10, dens_ratio=1.5)
# sample based rarefaction under random sampling
rarefaction(inv_comm, method='SBR')

# sampled based rarefaction under spatially explicit nearest neighbor sampling
rarefaction(inv_comm, method='sSBR', coords=inv_plot_attr[, c('x', 'y')], latlong=FALSE)
# the syntax is simpler if supplying a mob_in object
rarefaction(inv_mob_in, method='sSBR', spat_algo = 'kNCN')
rarefaction(inv_mob_in, method='sSBR', spat_algo = 'kNN')

subset.mob_in

Subset the rows of the mob data input object

Description

This function subsets the rows of comm, env, and spat attributes of the mob_in object
Usage

## S3 method for class 'mob_in'
subset(x, subset, type = "string", drop_levels = FALSE, ...)

Arguments

x                  an object of class mob_in created by \texttt{make.mob.in}
subset             expression indicating elements or rows to keep: missing values are taken as false.
type               specifies the type of object the argument \texttt{subset} specifies, may be: \texttt{string}, \texttt{integer}, or \texttt{logical}, defaults to \texttt{string}
drop_levels        Boolean if TRUE unused levels are removed from factors in \texttt{mob.in$env}
...                parameters passed to other functions

Examples

data(inv_comm)
data(inv_plot_attr)
inv.mob.in = make.mob.in(inv.comm, inv.plot.attr, coord.names = c('x', 'y'))
subset(inv.mob.in, group == 'invaded')
subset(inv.mob.in, 1:4, type='integer')
subset(inv.mob.in, 1:4, type='integer', drop.levels=TRUE)
sub_log = c(TRUE, FALSE, TRUE, rep(FALSE, nrow(inv.mob.in$comm) - 3))
subset(inv.mob.in, sub_log, type='logical')

tank_comm            \textit{Cattle tank data set}

Description

Species counts of aquatic macro-invertebrates from experimental freshwater ponds ("cattle tanks") with two different nutrient treatments.

Details

tank_comm is a site-by-species matrix with individual counts.
tank_plot_attr is a data frame with corresponding site variables. The column group specifies whether a pond has received a "high" or "low" nutrient treatment. The columns \texttt{x} and \texttt{y} contain the spatial coordinates of the sites.
The data were adapted from Chase (2010).

References

Examples

data(tank_comm)
data(tank_plot_attr)
tank_mob_in = make_mob_in(tank_comm, tank_plot_attr)
Index

* burned
  fire_comm, 8
* cattle
  tank_comm, 30
* data
  fire_comm, 8
  inv_comm, 16
  tank_comm, 30
* fire
  fire_comm, 8
* invaded
  inv_comm, 16
* invasion
  inv_comm, 16
* ponds
  tank_comm, 30
* tanks
  tank_comm, 30

avg_nn_dist, 2

calc_biodiv, 3
calc_chao1, 5, 13
calc_PIE, 6
compare_samp_rarefaction, 7

fire_comm, 8
fire_plot_attr (fire_comm), 8

get_delta_stats, 8
get_mob_stats, 4, 11, 22
get_null_comm, 15

inv_comm, 16
inv_plot_attr (inv_comm), 16

kNCN_average, 17

make_mob_in, 18, 30
makeCluster, 12, 18
mobr, 20

plot.mob_out, 20
plot.mob_stats, 22
plot_abu, 23
plot_N, 24
plot_rarefaction, 25
rarefaction, 10, 26, 26
subset.mob_in, 29
tank_comm, 30
tank_plot_attr (tank_comm), 30