Package ‘prabclus’

January 9, 2020

Title Functions for Clustering and Testing of Presence-Absence, Abundance and Multilocus Genetic Data

Version 2.3-2

Date 2020-01-06

Author Christian Hennig <christian.hennig@unibo.it>, Bernhard Hausdorf <Hausdorf@zoologie.uni-hamburg.de>

Depends R (>= 2.10), MASS, mclust

Suggests spdep, spatialreg, bootstrap, maptools, foreign, mvtnorm


Maintainer Christian Hennig <christian.hennig@unibo.it>

License GPL


NeedsCompilation no

Repository CRAN

Date/Publication 2020-01-08 23:00:39 UTC

R topics documented:

   prabclus-package .................................................. 3
   abundtest .......................................................... 5
   allele2zeroone ...................................................... 9
   alleleconvert ....................................................... 10
   allelesdist .......................................................... 12
   alleleinit ........................................................... 13
   allelepaircomp ..................................................... 16
<table>
<thead>
<tr>
<th>R topics documented:</th>
</tr>
</thead>
<tbody>
<tr>
<td>autoconst</td>
</tr>
<tr>
<td>build.charmatrix</td>
</tr>
<tr>
<td>build.ext.nblist</td>
</tr>
<tr>
<td>build.nblist</td>
</tr>
<tr>
<td>cluspop.nb</td>
</tr>
<tr>
<td>communities</td>
</tr>
<tr>
<td>communitydist</td>
</tr>
<tr>
<td>comp.test</td>
</tr>
<tr>
<td>con.comp</td>
</tr>
<tr>
<td>con.regmat</td>
</tr>
<tr>
<td>coord2dist</td>
</tr>
<tr>
<td>crmatrix</td>
</tr>
<tr>
<td>dicedist</td>
</tr>
<tr>
<td>distratio</td>
</tr>
<tr>
<td>geco</td>
</tr>
<tr>
<td>geo2neighbor</td>
</tr>
<tr>
<td>homogen.test</td>
</tr>
<tr>
<td>hprabclust</td>
</tr>
<tr>
<td>incmatrix</td>
</tr>
<tr>
<td>jaccard</td>
</tr>
<tr>
<td>kulczynski</td>
</tr>
<tr>
<td>kykladspecreg</td>
</tr>
<tr>
<td>lcomponent</td>
</tr>
<tr>
<td>lociplots</td>
</tr>
<tr>
<td>nastats</td>
</tr>
<tr>
<td>nb</td>
</tr>
<tr>
<td>nbtest</td>
</tr>
<tr>
<td>nn</td>
</tr>
<tr>
<td>NNcLean</td>
</tr>
<tr>
<td>phipt</td>
</tr>
<tr>
<td>piecewiselin</td>
</tr>
<tr>
<td>plotdistreg</td>
</tr>
<tr>
<td>pop.sim</td>
</tr>
<tr>
<td>prab.sarestimate</td>
</tr>
<tr>
<td>prabclust</td>
</tr>
<tr>
<td>prabinit</td>
</tr>
<tr>
<td>prabtest</td>
</tr>
<tr>
<td>qkulczynski</td>
</tr>
<tr>
<td>randpop.nb</td>
</tr>
<tr>
<td>reqdist</td>
</tr>
<tr>
<td>regdistbetween</td>
</tr>
<tr>
<td>regdistbetweennone</td>
</tr>
<tr>
<td>regdistdiff</td>
</tr>
<tr>
<td>regdistdiffone</td>
</tr>
<tr>
<td>reqeqdist</td>
</tr>
<tr>
<td>regpop.sar</td>
</tr>
<tr>
<td>siskiyou</td>
</tr>
<tr>
<td>specgroups</td>
</tr>
</tbody>
</table>
prabclus-package

Description

Here is a list of the main functions in package prabclus. Most other functions are auxiliary functions for these.

Initialisation

prabinit  Initialises presence/absence-, abundance- and multilocus data with dominant markers for use with most other key prabclus-functions.

alleleinit  Initialises multilocus data with codominant markers for use with key prabclus-functions.

alleleconvert  Generates the input format required by alleleinit.

Tests for clustering and nestedness

prabtest  Computes the tests introduced in Hausdorf and Hennig (2003) and Hennig and Hausdorf (2004; these tests occur in some further publications of ours but this one is the most detailed statistical reference) for presence/absence data. Allows use of the geco-dissimilarity (Hennig and Hausdorf, 2006).

abundtest  Computes the test introduced in Hausdorf and Hennig (2007) for abundance data.

homogen.test  A classical distance-based test for homogeneity going back to Erdos and Renyi (1960) and Ling (1973).

Clustering

prabclus  Species clustering for biotic element analysis (Hausdorf and Hennig, 2007, Hennig and Hausdorf, 2004 and others), clustering of individuals for species delimitation (Hausdorf and Hennig, 2010) based on Gaussian mixture model clustering with noise as implemented in R-package mclust, Fraley and Raftery (1998), on output of multidimensional scaling from distances as computed by prabinit or alleleinit. See also stressvals for help with choosing the number of MDS-dimensions.

hprabclus  An unpublished alternative to prabclus using hierarchical clustering methods.

lociplots  Visualisation of clusters of genetic markers vs. clusters of species.

NNclean  Nearest neighbor based classification of observations as noise/outliers according to Byers and Raftery (1998).
Dissimilarity matrices

- **alleledist**: Shared allele distance (see the corresponding help pages for references).
- **dicedist**: Dice distance.
- **geco**: Geco coefficient, taking geographical distance into account.
- **jaccard**: Jaccard distance.
- **kulczynski**: Kulczynski dissimilarity.
- **qkulczynski**: Quantitative Kulczynski dissimilarity for abundance data.

Communities

- **communities**: Constructs communities from geographical distances between individuals.
- **communitydist**: Chord-, phiPT- and various versions of the shared allele distance between communities.

Tests for equality of dissimilarity-based regression

- **regeqdist**: Jackknife-based test for equality of two independent regressions between distances (Hausdorff and Hennig 2019).
- **regdistbetween**: Jackknife-based test for equality of regression involving all distances and regression involving within-group distances only (Hausdorff and Hennig 2019).
- **regdistbetweennon**: Jackknife-based test for equality of regression involving within-group distances of a reference group only and regression involving between-group distances (Hausdorff and Hennig 2019).

Small conversion functions

- **coord2dist**: Computes geographical distances from geographical coordinates.
- **geo2neighbor**: Computes a neighborhood list from geographical distances.
- **alleleconvert**: A somewhat restricted function for conversion of different file formats used for genetic data with codominant markers.

Data sets

- **kykladspecreg, siskiyou, veronica, tetragonula.**

Author(s)

Christian Hennig <christian.hennig@unibo.it> [https://www.unibo.it/sitoweb/christian.hennig/en/]
abundtest

References


Usage

```r
abundtest(prabobj, teststat = "distratio", tuning = 0.25, times = 1000, p.nb = NULL, prange = c(0, 1), nperp = 4, step = 0.1, step2 = 0.01, 
```
twostep = TRUE, species.fixed=TRUE, prab01=NULL, 
groupvector=NULL, 
sareestimate=prab.sarestimate(prabobj), 
dist = prabobj$distance, 
n.species = prabobj$n.species)

Arguments

prabobj: an object of class prab (presence-absence data), as generated by prabinit.
teststat: string, indicating the test statistics. "isovertice": number of isolated vertices in the graph of tuning smallest distances between species. "lcomponent": size of largest connectivity component in this graph. "distratio": ratio between tuning smallest and largest distances. "nn": average distance of species to tuning nearest neighbor. "inclusions": number of inclusions between areas of different species (tests for nestedness structure, not for clustering, and treats abundance matrices as presence-absence-data). "mean": mean of the distances between species (this is a rough measure of species co-occurrence). "groups": this requires a specification of a vector defining different groups of species via parameter groupvector. The test statistic is then the mean of the distances between species of the same group. This is computed over all species, but also for every single group of species. It also includes the "mean"-test, so that the number of tests carried out is number of species groups with more than one element plus two.
tuning: integer or (if teststat="distratio") numerical between 0 and 1. Tuning constant for test statistics, see teststat.
times: integer. Number of simulation runs.
p.nb: numerical between 0 and 1. The probability that a new region is drawn from the non-neighborhood of the previous regions belonging to a species under generation. If NULL (the default), and prabobj$spatial, prabtest estimates this by function autoconst. Otherwise the next five parameters have no effect. If NULL, and !prabobj$spatial, spatial structure is ignored.
prange: numerical range vector, lower value not smaller than 0, larger value not larger than 1. Range where pd is to be found. Used by function autoconst.
nperp: integer. Number of simulations per pd-value. Used by function autoconst.
step: numerical between 0 and 1. Interval length between subsequent choices of pd for the first simulation. Used by function autoconst.
step2: numerical between 0 and 1. Interval length between subsequent choices of pd for the second simulation (see parameter twostep). Used by function autoconst.
twostep: logical. If TRUE, a first estimation step for pd is carried out in the whole prange, and then the final estimation is determined between the preliminary estimator -5*step2 and +5*step2. Else, the first simulation determines the final estimator. Used by function autoconst.
species.fixed: logical. Indicates if the range sizes of the species are held fixed in the test simulation (TRUE) or generated from their empirical distribution in x (FALSE) for presence-absence data. See function randpop.nb. Use always TRUE for abundance data (not necessary if teststat="inclusions").
prab01: object based on presence-absence matrix of same dimensions than
the abundance matrix of prabobj. This specifies the presences and absences on
which the presence/absence step of abundance-based tests is based (see details).
If NULL (which is usually the only reasonable choice), prab01 is computed in
order to indicate the nonzeros of prabobj$prab.

prab01

integer vector. For every species, a number indicating the species’ group mem-
bership. Needed only if teststat="groups".

sarestimate: Estimator of the parameters of a simultaneous autoregression model corresponding
to the null model for abundance data from Hausdorf and Hennig (2007) as
generated by prab.sarestimate. This requires package spdep. Note that by
explicitly specifying sarestimate=NULL simulation of 0-1 matrices can be enforced.

dist: One of "jaccard", "kulczynski", "qkulczynski" or "logkulczynski" specifying the distance measure on which the test is based. By default, this is taken from prabobj.

n.species: number of species. By default this is taken from prabobj. This should normally
not be changed.

Details

For presence-absence data, the routine is described in prabtest. For abundance data, the first step
under the null model is to simulated presence-absence patterns as in prabtest. The second step is
to fit a simultaneous autoregression (SAR) model (Ripley 1981, section 5.2) to the log-abundances,
see prab.sarestimate. The simulation from the null model is implemented in regpop.sar. For
more details see Hennig and Hausdorf (2004) for presence-absence data and Hausdorf and Hennig
(2007) for abundance data and the test statistics "mean" and "groups", which can also be applied
to binary data.

If p.nb=NA was specified, a diagnostic plot for the estimation of pd is plotted by autoconst. For
details see Hennig and Hausdorf (2004) and the help pages of the cited functions.

Value

An object of class prabtest, which is a list with components

results: vector of test statistic values for all simulated populations. For teststat="groups"
a list with components overall (means of within group-distances), mean (means
of all distances), gr (matrix with a row for every group, giving the groupwise
within-group distance means).

p.above: p-value against an alternative that generates large values of the test statistic (usu-
ally reasonable for teststat="inclusions", "groups", "mean").

p.below: p-value against an alternative that generates small values of the test statistic
(usually reasonable for "lcomponent", "nn", "distratio"; for "isovertice",
the two-sided p may make sense which is twice the smaller one of p.above and
p.below).

datac: test statistic value for the original data. (specgroups-output for teststat="groups").

tuning: see above.
distance   dist above.
teststat   see above.
pd   p.nb above.
abund   TRUE if simultaneous autoregression has been used (i.e., a sarestimate has been supplied or computed).
sarlambda   Estimator of the autocorrelation parameter lambda (see errorsarlm) defined so that the average weight of neighbors (see nb2listw) is standardized to 1.
sarestimate   the output object of prab.sarestimate.
groupinfo   list containing information from "groups" tests, with components lg (levels of groupvector), ng (number of groups), nsg (vector of group sizes), testm (value of "means" test statistic for input prabobj), pa (group-wise p.above), pb (group-wise p.below), pma (p.above of "means" test), pmb (p.below of "means" test).

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References


See Also

prabinit generates objects of class prab.
autoconst estimates pd from such objects.
prabtest (analogous function for presence-absence data).
regpop.sar generates populations from the null model.
prab.sarestimate (parameter estimators for simultaneous autoregression model). This calls errorsarlm (original estimation function from package spdep).
Some more information on the test statistics is given in homogen.test, lcomponent, distratio, nn, incmatrix.
Summary and print methods: summary.prabtest.
**Allele2zeroone**

Converts alleleobject into binary matrix

### Description

Converts `alleleobject` with codominant markers into binary matrix with a column for each marker.

### Usage

```r
allele2zeroone(alleleobject)
```

### Arguments

- **alleleobject**: object of class `alleleobject` as generated by `alleleinit`.

### Value

A 0-1-matrix with individuals as rows and markers (alleles) as columns.

### Author(s)

Christian Hennig <christian.hennig@unibo.it> [https://www.unibo.it/sitoweb/christian.hennig/en](https://www.unibo.it/sitoweb/christian.hennig/en)

### Examples

```r
data(tetragonula)
ta <- alleleconvert(strmatrix=tetragonula[21:50,])
tai <- alleleinit(allelematrix=ta)
allele2zeroone(tai)
```
Description

Codominant marker data (which here means: data with several diploid loci; two alleles per locus) can be represented in various ways. This function converts the formats "genepop" and "structure" into "structurama" and "prabclus". "genepop" is a version of the format used by the package GENEPOP (Rousset, 2008), "structure" is a version of what is used by STRUCTURE (Pritchard et al., 2000), another one is "structureb". "structurama" is a version of what is used by STRUCTURAMA (Huelsenbeck and Andolfatto, 2007) and "prabclus" is required by the function alleleinit in the present package.

Usage

```r
alleleconvert(file=NULL,strmatrix=NULL, format.in="genepop", format.out="prabclus", alength=3,orig.nachar="000",new.nachar="-", rows.are.individuals=TRUE, firstcolname=FALSE, aletters=intToUtf8(c(65:90,97:122),multiple=TRUE), outfile=NULL,skip=0)
```

Arguments

- **file** string. Filename of input file, see details. One of file and strmatrix needs to be specified.
- **strmatrix** matrix or data frame of strings, see details. One of file and strmatrix needs to be specified.
- **format.in** string. One of "genepop", "structure", or "structureb", see details.
- **format.out** string. One of "structurama" or "prabclus", see details.
- **alength** integer. If format.in="genepop", length of code for a single allele.
- **orig.nachar** string. Code for missing values in input data.
- **new.nachar** string. Code for missing values in output data.
- **rows.are.individuals** logical. If TRUE, rows are interpreted as individuals and columns (variables if strmatrix is a data frame) as loci.
- **firstcolname** logical. If TRUE, it is assumed that the first column contains row names.
- **aletters** character vector. String of default characters for alleles if format.out="prabclus" (the default is fine unless there is a locus that can have more than 62 different alleles in the dataset).
- **outfile** string. If specified, the output matrix (omitting quotes) is written to a file of this name (including row names if firstcolname==TRUE).
- **skip** number of rows to be skipped when reading data from a file (skip-argument of `read.table`).
Details

The formats are as follows (described is the format within R, i.e., for the input, the format of `strmatrix`; if `file` is specified, the file is read with `read.table(file, colClasses="character")` and should give the format explained below - note that `colClasses="character"` implies that quotes are not needed in the input file):

**genepop** Alleles are coded by strings of length `alength` and there is no space between the two alleles in a locus, so a value of "258260" means that in the corresponding locus the two alleles have codes 258 and 260.

**structure** Alleles are coded by strings of arbitrary length. Two rows correspond to each individual, the first row containing the first alleles in all loci and the second row containing the second ones.

**structureb** Alleles are coded by strings of arbitrary length. One row corresponds to each individual, containing first and second alleles in all loci (first and second allele of first locus, first and second allele of second locus etc.). This starts in the third row (first two have locus names and other information).

**structurama** Alleles are coded by strings of arbitrary length. The two alleles in each locus are written with brackets around them and a comma in between, so "258260" in "genepop" corresponds to "(258,260)" in "structurama".

**prabclus** Alleles are coded by a single character and there is no space between the two alleles in a locus (e.g., "AC").

Value

A matrix of strings in the format specified as `format.out` with an attribute "alevels", a vector of all used allele codes if `format.out="prabclus"`, otherwise vector of allele codes of last locus.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References


See Also

`alleleinit`
Examples

```r
data(tetragonula)
# This uses example data file Heterotrigona_indoF0.dat
str(alleleconvert(strmatrix=tetragonula))
strucmatrix <-
cbind(c("I1", "I1", "I2", "I2", "I3", "I3"),
c("122", "144", "122", "122", "144", "144"),c("0", "0", "21", "33", "35", "44"))
alleleconvert(strmatrix=strucmatrix, format.in="structure",
format.out="prabclus", orig.nachar="0", firstcolname=TRUE)
alleleconvert(strmatrix=strucmatrix, format.in="structure",
format.out="structurama", orig.nachar="0", new.nachar="-9", firstcolname=TRUE)
```

---

**alleledist**

*Shared allele distance for diploid loci*

Description

Shared allele distance for codominant markers (Bowcock et al., 1994). One minus proportion of alleles shared by two individuals averaged over loci (loci with missing values for at least one individual are ignored).

Usage

```r
alleledist(allelelist, ni, np, count=FALSE)
```

Arguments

- **allelelist**
  - a list of lists. In the "outer" list, there are np lists, one for each locus. In the "inner" list, for every individual there is a vector of two codes (typically characters, see *alleleinit*) for the two alleles in that locus. Such a list can be constructed by `unbuild.charmatrix` out of the `charmatrix` component of an output object of `alleleinit`.

- **ni**
  - integer. Number of individuals.

- **np**
  - integer. Number of loci.

- **count**
  - logical. If TRUE, the number of the individual to be processed is printed.

Value

A symmetrical matrix of shared allele distances between individuals.

Author(s)

Christian Hennig <christian.hennig@unibo.it> [https://www.unibo.it/sitoweb/christian.hennig/en](https://www.unibo.it/sitoweb/christian.hennig/en)
**alleleinit**

---

**References**


**See Also**

`alleleinit`, `unbuild.charmatrix`

**Examples**

```r
data(tetragonula)
tnb <- coord2dist(coordmatrix=tetragonula.coord[1:50,],cut=50,file.format="decimal2",neighbors=TRUE)
ta <- alleleconvert(strmatrix=tetragonula[1:50,])
tai <- alleleinit(allelematrix=ta,neighborhood=tnb$nblist,distance="none")
str(alleledist((unbuild.charmatrix(tai$charmatrix,50,13)),50,13))
```

---

**Diploid loci matrix initialization**

**Description**

`alleleinit` converts genetic data with diploid loci as generated by `alleleconvert` into an object of class `alleleobject`. `print.alleleobject` is a print method for such objects.

**Usage**

```r
alleleinit(file = NULL, allelematrix=NULL, 
rows.are.individuals = TRUE, 
neighborhood = "none", distance = "alleledist", namode="variables", nachar="-", distcount=FALSE)
```

```r
## S3 method for class 'alleleobject'
print(x, ...)
```

**Arguments**

- `file` string. File name. File must be in "prabclus" format, see details. Either `file` or `allelematrix` needs to be specified.
- `allelematrix` matrix in "prabclus"-format as generated by `alleleconvert`, see details. Either `file` or `allelematrix` needs to be specified.
- `rows.are.individuals` logical. If TRUE, rows are interpreted as individuals and columns are interpreted as loci.
neighborhood A string or a list with a component for every individual. The components are vectors of integers indicating neighboring individuals. An individual without neighbors should be assigned a vector `numeric(0)`. If neighborhood is a filename, it is attempted to read such a list from a file, where every row should correspond to one region (such as example dataset nb.dat). If `neighborhood="none"`, all neighborhoods are set to `numeric(0)`. The neighborhood can be tested by `nbtest` for consistency.

distance "alleledist" or "none". The distance measure between individuals to compute by `alleleinit`.

namode one of "single", "individuals", "variables", or "none". Determines whether a single probability for the entry to be missing is computed for a single locus of an individual ("single"), a vector of individual-wise probabilities for loci to be missing ("individuals"), a vector of loci-wise probabilities for individuals to be missing ("variables") or no missingness probability at all.

nachar character denoting missing values.

distcount logical. If TRUE, during distance computation individuals are counted on the screen.

x object of class `alleleobject`.

... necessary for print method.

Details

The required input format is the output format "prabclus" of `alleleconvert`. Alleles are coded by a single character, so diploid loci need to be pairs of characters without space between the two alleles (e.g., "AC"). The input needs to be an individuals*loci matrix or data frame (or a file that produces such a data frame by `read.table(file,stringsAsFactors=FALSE)`)

Value

`alleleinit` produces an object of class `alleleobject` (note that this is similar to class `prab`; for example both can be used with `prabclust`), which is a list with components:

- `distmat` distance matrix between individuals.
- `amatrix` data frame of input data with string variables in the input format, see details. Note that in the output for an individual the whole locus is declared missing if at least one of its alleles is missing in the input.
- `charmatrix` matrix of characters in which there are two rows for every individual corresponding to the two alleles in every locus (column). Entries are allele codes but missing values are coded as NA.
- `nb` neighborhood list, see above.
- `ext.nblist` a neighborhood list in which for every row in `charmatrix` the second row number corresponding to the neighboring individuals is listed.
- `n.variables` number of loci.
- `n.individuals` number of individuals.
- `n.levels` maximum number of different alleles in a locus.
alleleinit

n.species identical to n.individuals used for compatibility with prabclust.
alevels character vector with all used allele codes not including missing values.
leveeldist matrix in which rows are loci, columns are alleles and entries are frequencies of alleles per locus.
prab useless matrix of number of factor levels corresponding to amatrix added for compatibility with objects of class prab.
regperspec vector of row-wise sums of prab added for compatibility with objects of class prab.
specperreg vector of column-wise sums of prab added for compatibility with objects of class prab.
distance string denoting the chosen distance measure, see above.
namode see above.
naprob probability of missing values, numeric or vector, see documentation of argument namode.
nasum number of missing entries (individual/loci) in amatrix.
nachar see above.
spatial logical. TRUE if a neighborhood was submitted.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

See Also

alleleconvert, alleledist, prabinit.

Examples

# Only 50 observations are used in order to have a fast example.
data(tetragonula)
tnb <- coord2dist(coordmatrix=tetragonula.coord[1:50,,cut=50,file.format="decimal2",neighbors=TRUE)
ta <- alleleconvert(strmatrix=tetragonula[1:50,,])
tai <- alleleinit(allelematrix=ta,neighborhood=tnb$nblist)
print(tai)
allelepaircomp

*Internal: compares two pairs of alleles*

**Description**

Used for computation of the genetic distances `alleledist`.

**Usage**

```r
allelepaircomp(allelepair1, allelepair2, method="sum")
```

**Arguments**

- `allelepair1`: vector of two allele codes (usually characters), or NA.
- `allelepair2`: vector of two allele codes (usually characters), or NA.
- `method`: one of "sum" or "geometrical".

**Value**

If `method="sum"`, number of shared alleles (0, 1 or 2), or NA. If `method="geometrical"`, 0, 0.5, `sqrt(0.5)` (in case that one of the allelepairs is double such as in `c("A","B"),c("A","A")`) or 1, or NA.

**Author(s)**

Christian Hennig <christian.hennig@unibo.it> [https://www.unibo.it/sitoweb/christian.hennig/en](https://www.unibo.it/sitoweb/christian.hennig/en)

**See Also**

`alleledist`

**Examples**

```r
allelepaircomp(c("A","B"),c("A","C"))
```
autoconst

Spatial autocorrelation parameter estimation

Description

Monte Carlo estimation of the disjunction/spatial autocorrelation parameter pd for the simulation model used in randpop.nb, used for tests for clustering of presence-absence data.

autoconst is the main function; autoreg performs the simulation and is executed within autoconst.

Usage

autoconst(x, prange = c(0, 1), twostep = TRUE, step1 = 0.1, step2 = 0.01, plot = TRUE, nperp = 4, ejprob = NULL, species.fixed = TRUE, pdfnb=FALSE, ignore.richness=FALSE)

autoreg(x, probs, ejprob, plot = TRUE, nperp = 4, species.fixed = TRUE, pdfnb=FALSE, ignore.richness=FALSE)

Arguments

x object of class prab as generated by prabinit. Presence-absence data to be analyzed.
prange numerical range vector, lower value not smaller than 0, larger value not larger than 1. Range where the parameter is to be found.
twostep logical. If TRUE, a first estimation step is carried out in the whole prange, and then the final estimation is determined between the preliminary estimator -5*step2 and +5*step2. Else, the first simulation determines the final estimator.
step1 numerical between 0 and 1. Interval length between subsequent choices of pd for the first simulation.
step2 numerical between 0 and 1. Interval length between subsequent choices of pd for the second simulation in case of twostep=TRUE.
plot logical. If TRUE, a scatterplot of pd-values against resulting ejprob values (see below), with regression line and data value of ejprob is shown.
nperp integer. Number of simulations per pd-value.
ejprob numerical between 0 and 1. Observed disjunction probability for data x; if not specified in advance, it will be computed by autoconst.
Species.fixed logical. If TRUE, sizes of generated species match the species sizes in x, else they are generated from the empirical distribution of species sizes in x.
probs vector of numericals between 0 and 1. pd values for the simulation.
pdfnb logical. If TRUE, the probabilities of the regions are modified according to the number of neighboring regions in randpop.nb, see Hennig and Hausdorf (2002), p. 5.
ignore.richness

logical. If TRUE, there is no assumption of species richesses to differ between regions in the null model. Regionwise probabilities don’t differ in the generation of null data.

Details

The spatial autocorrelation parameter pd of the model for the generation of presence-absence data sets used by randpop.nb can be estimated by use of the observed disjuction probability ejprob which is the sum of all species’ connectivity components minus the number of species divided by the number of “presence” entries minus the number of species. This is done by a simulation of artificial data sets with characteristics of x and different pd-values, governed by prange, step1, step2 and nperp. ejprob is then calculated for all simulated populations. A linear regression of ejprob on pd is performed and the estimator of pd is determined by computing the inverse of the regression function for the ejprob-value of x.

Value

autoconst produces the same list as autoreg with additional component ejprob. The components are

pd (eventually) estimated parameter pd.

coef (eventually) estimated regression coefficients.

ejprob see above.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References


See Also

randpop.nb, prabinit, con.comp

Examples

options(digits=4)
data(kykladspecreg)
data(nb)
set.seed(1234)
Description

For use in *alleleinitt*. Creates a matrix of characters in which there are two rows for every individual corresponding to the two alleles in every locus (column) out of a list of lists, such as required by *alleledist*.

Usage

```
build.charmatrix(allelelist,n.individuals,n.variables)
```

Arguments

- `allelelist` A list of lists. In the "outer" list, there are `n.variables` lists, one for each locus. In the "inner" list, for every individual there is a vector of two codes (typically characters, see *alleleinitt*) for the two alleles in that locus.
- `n.individuals` integer. Number of individuals.
- `n.variables` integer. Number of loci.

Value

A matrix of characters in which there are two rows for every individual corresponding to the two alleles in every locus (column).

Author(s)

Christian Hennig &lt;christian.hennig@unibo.it&gt; https://www.unibo.it/sitoweb/christian.hennig/en

See Also

*alleleinitt*, *unbuild.charmatrix*

Examples

```
alist <- list()
alist[[1]] <- list(c("A","A"),c("B","A"),c(NA,NA))
alist[[2]] <- list(c("A","C"),c("B","B"),c("A","D"))
build.charmatrix(alist,3,2)
```
Description

This is for use in `allelinit`. Given a neighborhood list of individuals, a new neighborhood list is generated in which there are two entries for each individual (entry 1 and 2 refer to individual one, 3 and 4 to individual 2 and so on). Neighborhoods are preserved and additionally the two entries belonging to the same individual are marked as neighbors.

Usage

```r
build.ext.nblist(neighbors, n.individuals=length(neighbors))
```

Arguments

- `neighbors` list of integer vectors, where each vector contains the neighbors of an individual.
- `n.individuals` integer. Number of individuals.

Value

list with 2*n.individuals vectors of integers as described above.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

See Also

- `allelinit`

Examples

```r
data(veronica)
vnb <- coord2dist(coordmatrix=veronica.coord[1:20,], cut=20,
    file.format="decimal2",neighbors=TRUE)
built.ext.nblist(vnb$nblist)
```
build.nblist

Generate spatial weights from prabclus neighborhood list

Description

This generates a listw-object as needed for estimation of a simultaneous autoregression model in package spdep from a neighborhood list of the type generated in prabinit.

Usage

build.nblist(prabobj,prab01=NULL,style="C")

Arguments

- **prabobj**: object of class prab.
- **prab01**: presence-absence matrix of same dimensions than the abundance matrix of prabobj. This specifies the presences and absences on which the presence/absence step of abundance-based tests is based (see details). If NULL (which is usually the only reasonable choice), prab01 is computed in order to indicate the nonzeros of prabobj$prab.
- **style**: can take values "W", "B", "C", "U", and "S" though tests suggest that "C" should be chosen. See nb2listw.

Value

A 'listw' object with the following members:

- **style**: see above.
- **neighbours**: the neighbours list in spdep-format.
- **weights**: the weights for the neighbours and chosen style, with attributes set to report the type of relationships (binary or general, if general the form of the glist argument), and style as above.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

See Also

nb2listw (which is called)
Examples

```r
# Not run; requires package spdep
# data(siskiyou)
# x <- prabinit(prabmatrix=siskiyou, neighborhood=siskiyou.nb,
#                distance="logkulczynski")
# build.nblist(x)
```

---

**cluspop.nb**  
*Simulation of presence-absence matrices (clustered)*

**Description**

Generates a simulated matrix where the rows are interpreted as regions and the columns as species, 1 means that a species is present in the region and 0 means that the species is absent. Species are generated in order to produce 2 clusters of species with similar ranges. Spatial autocorrelation of a species' presences is governed by the parameter `p.nb` and a list of neighbors for each region.

**Usage**

```r
cluspop.nb(neighbors, p.nb = 0.5, n.species, clus.specs, reg.group, 
grouppf = 10, n.regions = length(neighbors), 
vector.species = rep(1, n.species), pdf.regions = rep(1/n.regions, 
n.regions), count = TRUE, pdfnb = FALSE)
```

**Arguments**

- `neighbors`  
  A list with a component for every region. The components are vectors of integers indicating neighboring regions. A region without neighbors (e.g., an island) should be assigned a list `numeric(0)`.
- `p.nb`  
  Numerical between 0 and 1. The probability that a new region is drawn from the non-neighborhood of the previous regions belonging to a species under generation. Note that for a given presence-absence matrix, this parameter can be estimated by `autoconst` (called `pd` there).
- `n.species`  
  Integer. Number of species.
- `clus.specs`  
  Integer not larger than `n.species`. Number of species restricted to one of the two groups of regions defined by `reg.group` (called "clustered species" because this leads to more similar species ranges).
- `reg.group`  
  Vector of pairwise distinct integers not larger than `n.regions`. Defines a group of regions to which a part of the `clus.specs` clustered species is restricted (more or less, see `grouppf`). The other clustered species are restricted to the complementary regions.
- `grouppf`  
  Numerical. The probability of the region of a clustered species to belong to the corresponding group of regions is up-weighted by factor `grouppf` compared to the generation of "non-clustered" species.
- `n.regions`  
  Integer. Number of regions.
cluspop.nb

vector.species vector of integers. The sizes (i.e., numbers of regions) of the species are generated randomly from the empirical distribution of vector.species.

pdf.regions numerical vector of length n.species. The entries must sum up to 1 and give probabilities for the regions to be drawn during the generation of a species. These probabilities are used conditional on the new region being a neighbor or a non-neighbor of the previous regions of the species, see p.nb, modified by grouppf for the clustered species.

count logical. If TRUE, the number of the currently generated species is printed.

dfnb logical. If TRUE, the probabilities of the regions are modified according to the number of neighboring regions by dividing them relative to the others by min(1, number of neighbors).

Details

The non-clustered species are generated as explained on the help page for randpop.nb. The general principle for the clustered species is the same, but with modified probabilities for the regions. For each clustered species, one of the two groups of regions is drawn, distributed according to the sum of its regions’ probability given by pdf.regions. The first region of such a species is only drawn from the regions of this group.

Value

A 0-1-matrix, rows are regions, columns are species.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References


See Also

randpop.nb,
autoconst estimates p.nb from matrices of class prab. These are generated by prabinit.

Examples

data(nb)
set.seed(888)
cluspop.nb(nb, p.nb=0.1, n.species=10, clus.specs=9, reg.group=1:17, vector.species=c(10))
communities

Construct communities from individuals

Description

Construct communities from individuals using geographical distance and hierarchical clustering. Communities are clusters of geographically close individuals, formed by hclust with specified distance cutoff.

Usage

communities(geodist, grouping=NULL, cutoff=1e-5, method="single")

Arguments

godist  dist-object or matrix of geographical distances between individuals.
grouping something that can be coerced into a factor. Different groups indicated by grouping cannot be together in the same community. (If NULL, there is no constraint.)
cutoff  numeric; clustering distance cutoff value, passed on as parameter h to cutree. Note that if this is smaller than the smallest nonzero geographical distance, communities will be all sets of individuals that have zero geographical distance to each other.
method  method-parameter for hclust.

Value

Vector of community memberships for the individuals (integer numbers from 1 to the number of communities without interruption.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.
hennig/en

See Also

communitydist

Examples

data(veronica)
ver.geo <- coord2dist(coordmatrix=veronica.coord[1:90,], file.format="decimal2")
species <- c(rep(1,64), rep(2,17), rep(3,9))
communities(ver.geo, species)
communitydist

Distances between communities

Description


Usage

communitydist(alleleobj, comvector = "auto", distance = "chord", compute.geodist = TRUE, out.dist = FALSE, grouping = NULL, geodist = NA, diploid = TRUE, phiptna = NA, ...)

Arguments

alleleobj
   if diploid=TRUE, an object of class alleleobject as produced by function alleleinit. This has the required information on the individuals that are grouped into communities. In case diploid=FALSE, a list that needs to have components n.variables (number of loci), alevels (vector of allele names, see alleleinit) and charmatrix (matrix of characters with one row for every individual and one column for every locus giving the alleles; see examples below for how this can be constructed for a prabobject with presence-absence data).

comvector
either a vector of integers indicating to which community an individual belongs (these need to be numbered from 1 to a maximum number without interruption), or "auto", which indicates that communities are automatically generated by the communities-function.

distance
   one of "chord", "phipt", "shared.average", "shared.chakraborty", "shared.problist". See Details.

compute.geodist
   logical, indicating whether geographical distances between communities should be generated.

out.dist
   logical, indicating whether dist-objects are given out or rather distance matrices.

grouping
   something that can be coerced into a factor, for passing on to communities in case that comvector=="auto". This implies that individuals in different groups indicated by grouping cannot be together in the same community. Furthermore (also if comvector is something else), a vector of groups of communities will be computed, see output component comgroup. In any case individuals in different groups are not allowed to be in the same community.
communitydist

geodist matrix or dist-object providing geographical distances between individuals. Required if compute.geodist==TRUE or convector=="auto".
diploid logical, indicating whether loci are diploid, see alleleobj.
phiptna if distance="phipt", value to be given out as phiPT-distance in case that the original definition amounts to 0/0 (particularly if communities have just one member).
... optional arguments to be passed on to communities.

Details

All genetic distances between communities are based on the information given in alleleobj; either on the alleles directly or on a genetic distance (distmat-component, see alleleinit). The possible genetic distance measures between communities are as follows:

- "chord": chord-distance (Cavalli-Sforza and Edwards, 1967)
- "phipt": phiPT-distance implemented according to Peakall and Smouse, 2012. This also appears in the literature under the name phiST (Meirmans, 2006, although the definition there is incomplete and we are not sure whether this is identical).
- "shared.average": average of between-community genetic distances.
- "shared.chakraborty": between-community shared allele distance according to Chakraborty and Jin (1993).
- "shared.problist": this implements the shared allele distance (Bowcock et al., 1994) for individuals directly for communities (one minus proportion of alleles shared by two communities averaged over loci).

Value

list with components

comvector integer vector of length of the number of individuals, indicating their community membership.

dist genetic distances between communities. Parameter out.dist determines whether this is a dist-object or a matrix.
cgeodist if compute.geodist, geographical distance between communities defined as average distance of all pairs of individuals belonging to different ones of the two communities between which the distance is computed. Parameter out.dist determines whether this is a dist-object or a matrix.
comgroup vector of length of the number of communities. If grouping was provided, this is a vector giving the group memberships of all communities, otherwise it is a vector of 1s.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en
References


See Also

`communities`: refer to `phipt` for computation of distances between specific pairs of communities. `diploidcomlist` produces relative frequencies for all alleles of all loci in all communities (on which the chord- and the "shared.problist"-distances are based).

Examples

```r
options(digits=4)
data(tetragonula)
tnb <-
coord2dist(coordmatrix=tetragonula.coord[83:120,],cut=50,
    file.format="decimal2",neighbors=TRUE)
ta <- alleleconvert(strmatrix=tetragonula[83:120,])
tai <- alleleinit(allelematrix=ta,neighborhood=tnb$nblist)
tetraspec <- c(rep(1,11),rep(2,13),rep(3,14))
tetracoms <- c(rep(1:3,each=3),4,5,rep(6:11,each=2),12,rep(13:19,each=2))
c1 <- communitydist(tai,comvector=tetracoms,distance="chord",
    geodist=tnb$distmatrix,grouping=tetraspec)
c2 <- communitydist(tai,comvector=tetracoms,distance="phipt",
    geodist=tnb$distmatrix,grouping=tetraspec,compute.geodist=FALSE)
c3 <- communitydist(tai,comvector=tetracoms,distance="shared.average",
    geodist=tnb$distmatrix,grouping=tetraspec,compute.geodist=FALSE)
c4 <- communitydist(tai,comvector=tetracoms,distance="shared.chakraborty",
    geodist=tnb$distmatrix,grouping=tetraspec,compute.geodist=FALSE)
c5 <- communitydist(tai,comvector=tetracoms,distance="shared.problist",
    geodist=tnb$distmatrix,grouping=tetraspec,compute.geodist=FALSE)
round(c1$geodist,digits=1)
c1$comvector
c2$comvector
c3$comvector
c4$comvector
c5$comvector
round(c1$dist,digits=2)
round(c2$dist,digits=2)
```
comp.test

Description

Tests for independence between a clustering and another grouping of species. This is simply an interface to chisq.test.

Usage

comp.test(cl,spg)

Arguments

cl  a vector of integers. Clustering of species (may be taken from prabclust).
spg a vector of integers of the same length, groups of species.

Details

chisq.test with simulated p-value is used.

Value

Output of chisq.test.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References


See Also

chisq.test, prabclust.

Examples

set.seed(1234)
g1 <- c(rep(1,34),rep(2,12),rep(3,15))
g2 <- sample(3,61,replace=TRUE)
comp.test(g1,g2)
connectivity components of an undirected graph

Description
Computes the connectivity components of an undirected graph from a matrix giving the edges.

Usage
con.comp(comat)

Arguments
comat  
a symmetric logical or 0-1 matrix, where comat[i,j]=TRUE means that there is an edge between vertices i and j. The diagonal is ignored.

Details
The "depth-first search" algorithm of Cormen, Leiserson and Rivest (1990, p. 477) is used.

Value
An integer vector, giving the number of the connectivity component for each vertice.

Author(s)
Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References

See Also
hclust, cutree for cutted single linkage trees (often equivalent).

Examples
```r
set.seed(1000)
x <- rnorm(20)
m <- matrix(0,nrow=20,ncol=20)
for(i in 1:20)
  for(j in 1:20)
    m[i,j] <- abs(x[i]-x[j])
d <- m < 0.2
cc <- con.comp(d)
max(cc) # number of connectivity components
```
plot(x,cc)
# The same should be produced by
# cutree(hclust(as.dist(m),method="single"),h=0.2).

---

**con.regmat**

*Connected regions per species*

---

**Description**

Returns a vector of the numbers of connected regions per species for a presence-absence matrix.

**Usage**

```r
con.regmat(regmat, neighbors, count = FALSE)
```

**Arguments**

- `regmat` 0-1-matrix. Columns are species, rows are regions.
- `neighbors` A list with a component for every region. The components are vectors of integers indicating neighboring regions. A region without neighbors (e.g., an island) should be assigned a list `numeric(0)`.
- `count` logical. If TRUE, the number of the currently processed species is printed.

**Details**

Uses `con.comp`.

**Value**

Vector of numbers of connected regions per species.

**Note**

Designed for use in `prabtest`.

**Author(s)**

Christian Hennig <christian.hennig@unibo.it> [https://www.unibo.it/sitoweb/christian.hennig/en](https://www.unibo.it/sitoweb/christian.hennig/en)

**See Also**

`con.comp`, `prabtest`
Examples

```r
data(nb)
set.seed(888)

```

```r
cp <- cluspop.nb(nb, p.nb=0.1, n.species=10, clus.specs=9,
reg.group=1:17,vector.species=c(10))
con.regmat(cp,nb)

``` coord2dist

**Geographical coordinates to distances**

Description

Computes geographical distances from geographical coordinates

Usage

```r
coord2dist(file=NULL, coordmatrix=NULL, cut=NULL,
file.format="degminsec",
output.dist=FALSE, radius=6378.137,
fp=1/298.257223563, neighbors=FALSE)
```

Arguments

**file**

string. A filename for the coordinate file. The file should have 2, 4 or 6 numeric columns and one row for each location. See file.format. One of file and coordmatrix needs to be specified (if coordmatrix is not specified, coordinates are read from file).

**coordmatrix**

something that can be coerced into a matrix with 2, 4 or 6 columns. Matrix of coordinates, one row for each location. See file.format. One of file and coordmatrix needs to be specified.

**cut**

numeric. Only active if neighbors==TRUE; see neighbors.

**file.format**

one of "degminsec", "decimal2" or "decimal4". The format of the required file or coordmatrix consists of the following columns:

"degminsec" 6 columns; the first three give degrees, minutes and seconds for latitude, columns 4-6 the same for longitude. Values in column 1 and 4 can be positive or negative (negative means "South", "West", respectively). Values in the other columns should be non-negative.

"decimal2" 2 columns; the first one gives latitude, the second one longitude in proper decimal notation. Values can be positive or negative (negative means "South", "West", respectively).

"decimal4" 4 columns; the first two give latitude, no. 3 and 4 give longitude. Values in column 1 and 3 can be positive or negative (negative means "South", "West", respectively). The give integer degrees. Values in the other columns should be non-negative. They give percentages (<=100).
output.dist logical. If TRUE, the resulting distance matrix is given out as a dist object.

radius numeric. Radius of the earth in km used in computation (the default is the equatorial radius but this is not the uniquely possible choice).

fp flattening of the earth; the default is from WGS-84.

neighbors logical. If TRUE, a neighborhood list is also computed, listing for every location all locations with distance <=cut as neighbors.

Value

If neighbors==TRUE, a list with components

distmatrix distance matrix between locations. See output.dist above. This is in km by default; the measurement unit is determined by the value used for radius.

nblist list with a vector for every location containing the numbers of its neighbors, see neighbors.

If neighbors==FALSE, only the distance matrix.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References

German Wikipedia from 29 August 2010: http://de.wikipedia.org/wiki/Orthodrome

See Also

geo2neighbor

Examples

options(digits=4)
data(veronica)
coord2dist(coordmatrix=veronica.coord[1:20,], cut=20, file.format="decimal2",neighbors=TRUE)

crmatrix Region-wise cluster membership

Description

Produce a matrix with clusters as rows and regions as columns, indicating how many species present in a region belong to the clusters

Usage

crmatrix(x,xc,percentages=FALSE)
**dicedist**

**Arguments**

- **x** object of class `prab` as generated by `prabinit`. Presence-absence data to be analyzed.
- **xc** object of class `prabclust` or `comprabclust` as generated by `prabclust` or `hprabclust`. The clustering.
- **percentages** logical. If `TRUE`, the output matrix will give the proportion of species from a certain region in the cluster.

**Value**

A clusters time regions matrix as explained above.

**Author(s)**

Christian Hennig <christian.hennig@unibo.it> [https://www.unibo.it/sitoweb/christian.hennig/en](https://www.unibo.it/sitoweb/christian.hennig/en)

**Examples**

```r
options(digits=3)
data(kykladspecreg)
data(nb)
set.seed(1234)
x <- prabinit(prabmatrix=kykladspecreg, neighborhood=nb)
xc <- prabclust(x)

crmatrix(x,xc)
crmatrix(x,xc, percentages=TRUE)
```

---

**dicedist**  
*Dice distance matrix*

**Description**

Computes a distance derived from Dice’s coincidence index between the columns of a 0-1-matrix.

**Usage**

dicedist(regmat)

**Arguments**

- **regmat** 0-1-matrix. Columns are species, rows are regions.
Details

The Dice distance between two species is 1 minus the Coincidence Index, which is \((2 \times \text{number of regions where both species are present})/(2 \times \text{number of regions where both species are present plus number of regions where at least one species is present})\). This is S23 in Shi (1993).

Value

A symmetrical matrix of Dice distances.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References


See Also

kulczynski.jaccard

Examples

```r
options(digits=4)
data(kykladspecreg)
dicedist(t(kykladspecreg))
```

---

### distratio

**Distance ratio test statistics for distance based clustering**

**Description**

Calculates the ratio between the \(\text{prop}\) smallest and largest distances of a distance matrix.

**Usage**

```r
distratio(distmat, prop = 0.25)
```

**Arguments**

- `distmat` symmetric distance matrix.
- `prop` numerical. Proportion between 0 and 1.

**Details**

Rounding is by floor for small and ceiling for large distances.
Value

A list with components

- **dr**: ratio of prop smallest to prop largest distances.
- **lom**: mean of prop smallest distances.
- **himean**: mean of prop smallest distances.
- **prop**: see above.

Author(s)

Christian Hennig <christian.hennig@unibo.it> [https://www.unibo.it/sitoweb/christian.hennig/en](https://www.unibo.it/sitoweb/christian.hennig/en)

References


See Also

prabtest

Examples

```r
options(digits=4)
data(kykladspecreg)
j <- jaccard(t(kykladspecreg))
distratio(j)
```

---

**geco**

**geco distance matrix**

**Description**

Computes geco distances between the columns of a 0-1-matrix, based on a distance matrix between regions (usually, but not necessarily, this is a geographical distance).

**Usage**

```r
geco(regmat, geodist=as.dist(matrix(as.integer(!diag(nrow(regmat)))))),
    transform="piece",
    tf=0.1,
    countmode=ncol(regmat)+1)
```
Arguments

regmat 0-1-matrix. Columns are species, rows are regions.
geodist dist-object or symmetric non-negative matrix. Geographical distances between regions.
transform transformation applied to the distances before computation of geco coefficient, see details. "piece" means piecewise linear, namely distance/(tf*maximum distance) if distance<tf*maximum distance, and 1 otherwise, "log" means log((tf*distance)+1), "sqrt" means sqrt(tf*distance), "none" means no transformation.
tf tuning constant for transformation. See transform.
countmode optional positive integer. Every 'countmode' algorithm runs 'geco' shows a message.

Details

The geco distance between two species is 0.5*(mean distance between region where species 1 is present and closest region where species 2 is present plus mean distance between region where species 2 is present and closest region where species 1 is present). 'closest' to a region could be the regions itself. It is recommended (Hennig and Hausdorf, 2006) to transform the distances first, because the differences between large distances are usually not meaningful or at least much less meaningful than differences between small distances for dissimilarity measurement between species ranges. See parameter transform.

If the between-regions distance is 1 for all pairs of non-equal regions, the geco distance degenerates to the Kulczynski distance, see kulczynski.

Value

A symmetrical matrix of geco distances.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References


See Also

kulczynski

Examples

options(digits=4)
data(kykladspecreg)
data(waterdist)
geco(t(kykladspecreg),waterdist)
**geo2neighbor**

*Neighborhood list from geographical distance*

**Description**
Generates a neighborhood list as required by `prabinit` from a matrix of geographical distances.

**Usage**
```r
geo2neighbor(geodist, cut = 0.1 * max(geodist))
```

**Arguments**
- `geodist`: dist-object or symmetric non-negative matrix. Geographical distances between regions.
- `cut`: non-negative numerical. All pairs of regions with distance <= `cut` are treated as neighbors.

**Value**
A list of integer vectors, giving the set of neighbors for every region.

**Author(s)**
Christian Hennig `<christian.hennig@unibo.it>` [https://www.unibo.it/sitoweb/christian.hennig/en](https://www.unibo.it/sitoweb/christian.hennig/en)

**Examples**
```r
data(waterdist)
geo2neighbor(waterdist)
```

---

**homogen.test**

*Classical distance-based test for homogeneity against clustering*

**Description**
Classical distance-based test for homogeneity against clustering. Test statistics is number of isolated vertices in the graph of smallest distances. The homogeneity model is a random graph model where `ne` edges are drawn from all possible edges.

**Usage**
```r
homogen.test(distmat, ne = ncol(distmat), testdist = "erdos")
```
Arguments

distmat numeric symmetric distance matrix.
ne integer. Number of edges in the data graph, corresponding to smallest distances.
testdist string. If testdist="erdos", the test distribution is a Poisson asymptotic distribution as given by Erdos and Renyi (1960). If testdist="ling", the test distribution is exact as given by Ling (1973), which needs much more computing time.

Details

The "ling"-test is one-sided (rejection if the number of isolated vertices is too large), the "erdos"-test computes a one-sided as well as a two-sided p-value.

Value

A list with components

p p-value for one-sided test.
p.twoside p-value for two-sided test, only if testdist="erdos".
iv number of isolated vertices in the data.
lambda parameter of the Poisson test distribution, only if testdist="erdos".
distcut largest distance value for which an edge has been drawn.
ne see above.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References


See Also

prabtest
Examples

```r
options(digits=4)
data(kykladspecreg)
j <- jaccard(t(kykladspecreg))
homogen.test(j, testdist="erdos")
homogen.test(j, testdist="ling")
```

**hprabclust**

*Clustering of species ranges from presence-absence matrices (hierarchical methods)*

**Description**

Clusters a presence-absence matrix object by taking the 'h-cut'-partition of a hierarchical clustering and declaring all members of too small clusters as 'noise' (this gives a distance-based clustering method, which estimates the number of clusters and allows for noise/non-clustered points). Note that this is experimental. Often, the prabclust-solutions is more convincing due to higher flexibility of that method. However, hprabclust may be more stable sometimes.

**Note:** Data formats are described on the prabinits help page. You may also consider the example datasets kykladspecreg.dat and nb.dat. Take care of the parameter rows.are.species of prabinits.

**Usage**

```r
hprabclust(prabobj, cutdist=0.4, cutout=1,
method="average", nnout=2, mdsplot=TRUE, mdsmethod="classical")
```

```r
## S3 method for class 'comprabclust'
print(x, ...)
```

**Arguments**

- **prabobj**: object of class prab as generated by prabinits. Presence-absence data to be analyzed.
- **cutdist**: non-negative integer. Cutoff distance to determine the partition, see cutree.
- **cutout**: non-negative integer. Points that have at most nnout distances smaller or equal than cutout are treated as noise.
- **method**: string. Clustering method, see hclust.
- **nnout**: non-negative integer. Members of clusters with less or equal than nnout points or that have less or equal than nnout neighbors closer than cutout are treated as noise.
- **mdsplot**: logical. If TRUE, the cluster solution is plotted on the first two MDS dimensions, see mdsmethod.
mdsmethod "classical", "kruskal", or "sammon". The MDS method to transform the distances to data points. "classical" indicates metric MDS by function cmdscale, "kruskal" is non-metric MDS. Note that if mdsmethod!="classical" zero distances between different objects are replaced by the minimum of the nonzero distances divided by 10 (otherwise the MDS method would produce an error). Note that mdsmethod is ignored if mdsplot=FALSE.

x comprabclust-object as generated by hprabclus. ...

Value

hprabclus generates an object of class comprabclus. This is a list with components

clustering vector of integers indicating the cluster memberships of the species (cutout-outliers are noise, but small clusters are allowed). Noise is coded as 0.

rclustering vector of integers indicating the cluster memberships of the species, noise as described under nnout. Noise is coded as 0.

cutdist see above.

method see above.

cutout see above.

nnout see above.

noisen number of points minus cutout-outliers.

symbols vector of characters corresponding to rclustering, but estimated noise by "N".

points numerical matrix. MDS configuration (if mdsplot=TRUE).

call function call.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

See Also

hclust, cutree, prabclus.

Examples

data(kykladspecreg)
data(nb)
data(waterdist)
x <- prabinit(prabmatrix=kykladspecreg, neighborhood=nb, geodist=waterdist, distance="geco")
hprabclus(x, mdsplot=FALSE)
Description
Computes species*species nestedness matrix and number of nestings (inclusions) from regions*species presence-absence matrix.

Usage
incmatrix(regmat)

Arguments
regmat
0-1-matrix. Columns are species, rows are regions.

Value
A list with components
m
0-1-matrix. m[i,j]=1 means that the occupied region of species j is a subset (not equal) of the region of species i.
ninc
integer. Number of strict inclusions.
neq
integer. Number of region equalities between species (not including equality between species i and i).

Author(s)
Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References

See Also
prabtest

Examples
data(kykladspecreg)
incmatrix(t(kykladspecreg))$ninc
Description

Computes Jaccard distances between the columns of a 0-1-matrix.

Usage

`jaccard(regmat)`

Arguments

- `regmat`: 0-1-matrix. Columns are species, rows are regions.

Details

The Jaccard distance between two species is \(1 - (\text{number of regions where both species are present}) / (\text{number of regions where at least one species is present})\). As a similarity coefficient, this is \(S_{22}\) in Shi (1993). Thank you to Laurent Buffat for improving this function!

Value

A symmetrical matrix of Jaccard distances.

Author(s)

Christian Hennig <christian.hennig@unibo.it> [https://www.unibo.it/sitoweb/christian.hennig/en]

References


See Also

- kulczynski, dicedist

Examples

```r
options(digits=4)
data(kykladspecreg)
jaccard(t(kykladspecreg))
```
Description

Computes Kulczynski distances between the columns of a 0-1-matrix.

Usage

kulczynski(regmat)

Arguments

regmat 0-1-matrix. Columns are species, rows are regions.

Details

The Kulczynski distance between two species is 1-(mean of (number of regions where both species are present)/(number of regions where species 1 is present) and (number of regions where both species are present)/(number of regions where species 2 is present)). The similarity version of this is S28 in Shi (1993).

Value

A symmetrical matrix of Kulczynski distances.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References


See Also

jaccard, geco.qkulczynski, dicedist

Examples

```r
options(digits=4)
data(kykladspecreg)
kulczynski(t(kykladspecreg))
```
kykladspecreg  

*Snail presence-absence data from Aegean sea*

### Description

0-1-matrix where rows are snail species and columns are islands in the Aegean sea. An entry of 1 means that the species is present in the region.

### Usage

```r
data(kykladspecreg)
```

### Format

A 0-1 matrix with 80 rows and 34 columns.

### Details

Reads from example data file `kykladspecreg.dat`.

### Source


### See Also

`nb` provides neighborhood information about the 34 islands. `waterdist` provides a geographical distance matrix between the islands.

### Examples

```r
data(kykladspecreg)
```

---

### lcomponent

*Largest connectivity component*

### Description

Computes the size of the largest connectivity component of the graph of `ncol(distmat)` vertices with edges defined by the smallest `ne` distances.

### Usage

```r
lcomponent(distmat, ne = floor(3*ncol(distmat)/4))
```
lociplots

Arguments

distmat symmetric distance matrix.
ne integer.

Value

list with components

lc size of the largest connectivity component.
ne see above.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References


See Also

prabtest

Examples

data(kykladspecreg)
j <- jaccard(t(kykladspecreg))
lcomponent(j)

lociplots

Visualises clusters of markers vs. species

Description

Given a clustering of individuals from prabclust (as generated in species delimitation) and a clustering of markers (for example dominant markers of genetic loci), lociplots visualises the presence of markers against the clustering of individuals and computes some statistics.

Usage

lociplots(indclust,locclust,locprab,lcluster,
symbols=NULL,brightest.grey=0.8,darkest.grey=0,
mdsdim=1:2)
Arguments

indclust \hspace{1em} \texttt{prabclust}-object. Clustering of individuals.
locclust \hspace{1em} vector of integers. Clustering of markers/loci.
locprab \hspace{1em} \texttt{prab}-object in which the markers are what the help page of \texttt{prabininit} refers to as "species" (i.e., reverse of what is used for species delimitation clustering; for data sets with codominant markers, such an object can be constructed by use of \texttt{allele2zeroone} before \texttt{prabininit}.)
lcluster \hspace{1em} integer. Number of cluster in \texttt{locclust} for which plot and statistics are produced.
symbols \hspace{1em} vector of plot symbols. If NULL, \texttt{indclust$symbols} is used.
brightest.grey \hspace{1em} numeric between 0 and 1. Brightest grey value used in plot for individuals with smallest marker percentage, see details.
darkest.grey \hspace{1em} numeric between 0 and 1. Darkest grey value used in plot for individuals with highest marker percentage, see details.
mdsdim \hspace{1em} vector of two integers. The two MDS variables taken from \texttt{indclust} used for visualisation.

Details

Plot and statistics are based on the individual marker percentage, which is the percentage of markers present in an individual of the markers belonging to cluster no. \texttt{lcluster}. In the plot, the grey value visualises the marker percentage.

Value

list with components

\begin{itemize}
  \item \texttt{locfreq} \hspace{1em} vector of individual marker percentages.
  \item \texttt{locfreqmin} \hspace{1em} vector of minimum individual marker percentages for each cluster in \texttt{indclust}-clustering (the first value refers to the "noise component", if present).
  \item \texttt{locfreqmax} \hspace{1em} vector of maximum individual marker percentages for each cluster in \texttt{indclust}-clustering (the first value refers to the "noise component", if present).
  \item \texttt{locfreqmean} \hspace{1em} vector of average individual marker percentages for each cluster in \texttt{indclust}-clustering (the first value refers to the "noise component", if present).
\end{itemize}

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

See Also

\texttt{prabclust}
Examples

```r
options(digits=4)
data(veronica)
vei <- prabinit(prabmatrix=veronica[1:50,],distance="jaccard")
ppv <- prabclust(vei)
veloci <- prabinit(prabmatrix=veronica[1:50,],rows.are.species=FALSE)
velociclust <- prabclust(veloci,nnk=0)
lociplots(ppv,velociclust$clustering,veloci,lcluster=3)
```

---

nastats  

### Missing values statistics for matrix

**Description**

Computes column-wise and row-wise numbers of missing values.

**Usage**

```r
nastats(amatrix, nastr="--")
```

**Arguments**

- `amatrix`: (any) matrix.
- `nastr`: missing value indicator.

**Value**

A list with components

- `narow`: vector of row-wise numbers of missing values.
- `nacol`: vector of column-wise numbers of missing values.

**Author(s)**

Christian Hennig <christian.hennig@unibo.it> [https://www.unibo.it/sitoweb/christian.hennig/en](https://www.unibo.it/sitoweb/christian.hennig/en)

**Examples**

```r
xx <- cbind(c(1,2,3),c(0,0,1),c(5,3,1))
nastats(xx,nastr=0)
```
### nb

*Neighborhood list for Aegean islands*

#### Description

List of neighboring islands for 34 Aegean islands.

#### Usage

```r
data(nb)
```

#### Format

List with 34 components, all being vectors of integers (or `numeric(0)` in case of no neighbors) indicating the neighboring islands.

#### Details

Reads from example data file `nb.dat`.

#### Source


#### Examples

```r
data(nb)
# nb <- list()
# for (i in 1:34)
#   nb <- c(nb, list(scan(file = "(path/)nb.dat",
#                       skip = i-1, nlines = 1)))
```

---

### nbtest

*Test of neighborhood list*

#### Description

Tests a list of neighboring regions for proper format. Neighborhood is tested for being symmetrical. Causes an error if tests fail.

#### Usage

```r
nbtest(nblist, n.regions = length(nblist))
```
Arguments

nblist A list with a component for every region. The components are vectors of integers indicating neighboring regions. A region without neighbors (e.g., an island) should be assigned a vector numeric(0).

n.regions Number of regions.

Value

invisible{TRUE}.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

See Also

prabinit.

Examples

data(nb)
nbtest(nb)
nb[[1]][1] <- 1
try(nbtest(nb))

nn Mean distance to kth nearest neighbor

Description

Computes the mean of the distances from each point to its neth nearest neighbor.

Usage

nn(distmat, ne = 1)

Arguments

distmat symmetric distance matrix (not a dist-object).

ne integer.

Value

numerical.
Author(s)
Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References

See Also
prabtest

Examples
```r
data(kykladspecreg)
j <- jaccard(t(kykladspecreg))
nn(j, 4)
```

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

Arguments
data      numerical matrix or data frame.
k          integer. Number of considered nearest neighbors per point.
distances  distance matrix object of class dist. If specified, it is used instead of computing distances from the data.
edgede.correct logical. If TRUE and the data is two-dimensional, neighbors for points at the edges of the parent region of the noise Poisson process are determined after wrapping the region onto a toroid.
wrap      numerical. If edgede.correct=TRUE, points in a strip of size wrap*range along the edge for each variable are candidates for being neighbors of points from the opposite.
**convergence** numerical. Convergence criterion for EM-algorithm.

**plot** logical. If TRUE, a histogram of the distance to kth nearest neighbor and fit is plotted.

**quiet** logical. If FALSE, the likelihood is printed during the iterations.

**x** object of class nneclean.

**...** necessary for print methods.

**Details**

The assumption is that the noise is distributed as a homogeneous Poisson process on a certain region and the clusters are distributed as a homogeneous Poisson process with larger intensity on a subregion (disconnected in case of more than one cluster). The distances are then distributed according to a mixture of two transformed Gamma distributions, and this mixture is estimated via the EM-algorithm. The points are assigned to noise or cluster component by use of the estimated a posteriori probabilities.

**Value**

NNeclean returns a list of class nneclean with components

- **z** 0-1-vector of length of the number of data points. 1 means cluster, 0 means noise.
- **probs** vector of estimated a priori probabilities for each point to belong to the cluster component.
- **k** see above.
- **lambda1** intensity parameter of cluster component.
- **lambda2** intensity parameter of noise component.
- **p** estimated probability of cluster component.
- **kthNND** distance to kth nearest neighbor.

**Note**

The software can be freely used for non-commercial purposes, and can be freely distributed for non-commercial purposes only.

**Author(s)**

R-port by Christian Hennig <christian.hennig@unibo.it> [https://www.unibo.it/sitoweb/christian.hennig/en](https://www.unibo.it/sitoweb/christian.hennig/en), original Splus package by S. Byers and A. E. Raftery.

**References**

Examples

library(mclust)
data(chevron)
nnc <- NNclean(chevron[,2:3],15,plot=TRUE)
plot(chevron[,2:3],col=1+nnc$z)

Description

Auxiliary functions for communitydist. phipt computes phiPT/phiST (Peakall and Smouse, 2012, Meirmans, 2006) between two communities. cfchord computes the chord-distance (Cavalli-Sforza and Edwards, 1967) between two lists or locus-wise relative allele frequencies. shared.problist computes a straightforward generalisation of the shared allele distance (Bowcock et al., 1994) between individuals for communities, namely the ‘overlap’, i.e., sum of the minima of the allele relative frequencies. diploidcomlist constructs the input lists for cfchord and shared.problist from an alleleobject. It provides relative frequencies for all alleles of all loci in all communities.

Usage

phipt(alleleobj,comvector,i,j)

cfchord(p1,p2)

shared.problist(p1,p2)

diploidcomlist(alleleobj,comvector,diploid=TRUE)

Arguments

alleleobj if diploid=TRUE, an object of class alleleobject as produced by function alleleinit. This has the required information on the individuals that are grouped into communities. In case diploid=FALSE, a list that needs to have components n.variables (number of loci), alevels (vector of allele names, see alleleinit) and charmatrix (matrix of characters with one row for every individual and one column for every locus giving the alleles; see examples below for how this can be constructed for a prabobject with presence-absence data).

comvector vector of integers indicating to which community an individual belongs.

i integer. Number of community.

j integer. Number of community. The phiPT-distance is computed between the communities numbered i and j.

p1 list. Every list entry refers to a locus and is a vector of relative frequencies of the alleles present in that locus in a community.

p2 list. Every list entry refers to a locus and is a vector of relative frequencies of the alleles present in that locus in a community. The chord or shared allele distance is computed between the communities encoded by p1 and p2.

diploid logical, indicating whether loci are diploid, see alleleobj.
Value

cfchord gives out the value of the chord distance. shared.problist gives out the distance value.
diploidcomlist gives out a two-dimensional list. The list has one entry for each community,
which is itself a list. This community list has one entry for each locus, which is a vector that
gives the relative frequencies of the different alleles in phipt gives out a list with components
phipt, vap, n0, sst, ssg, msa, msw. These refer to the notation on p.2.12 and 2.15 of Peakall and
Smouse (2012).

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>phipt</td>
<td>value of phiPT.</td>
</tr>
<tr>
<td>vap</td>
<td>variance among (between) populations (communities).</td>
</tr>
<tr>
<td>n0</td>
<td>standardisation factor N0, see p.2.12 of Peakall and Smouse (2012).</td>
</tr>
<tr>
<td>sst</td>
<td>total distances sum of squares.</td>
</tr>
</tbody>
</table>
| ssg      | vector with two non-NA entriesm, within community sums of squares for com-
munities i and j. |
| msa      | mean square between communities. |
| msw      | mean square within communities. |

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian. hennig/en

References

(1994) High resolution of human evolutionary trees with polymorphic microsatellites. Nature 368,
455-457.


Meirmans, P. G. (2006) Using the AMOVA framework to estimate a standardized genetic differen-
tiation measure. Evolution 60, 2399-2402.


See Also

communitydist

Examples

options(digits=4)
data(tetragonula)
tnb <-
coord2dist(coordmatrix=tetragonula.coord[83:120,],cut=50,file.format="decimal2",neighbors=TRUE)
ta <- alleleconvert(strmatrix=tetragonula[83:120,])
tai <- alleleinit(allelematrix=ta,neighborhood=tnb$nblist)
tetracoms <-
piecewiselin

Piecewise linear transformation for distance matrices

Description
Piecewise linear transformation for distance matrices, utility function for geco.

Usage
piecewiselin(distmatrix, maxdist=0.1*max(distmatrix))

Arguments
- distmatrix: symmetric (non-negative) distance matrix.
- maxdist: non-negative numeric. Larger distances are transformed to constant 1.

Details
Transforms large distances to 1, 0 to 0 and continuously linear between 0 and maxdist.

Value
A symmetrical matrix.

Author(s)
Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

See Also
geco

Examples
options(digits=4)
data(waterdist)
piecewiselin(waterdist)
plotdistreg  

Plots for within-groups and between-groups distance regression

Description

Visualisation of various regressions on distance (or dissimilarity) data where objects are from two groups.

Usage

plotdistreg(dmx, dmy, grouping, groups = levels(as.factor(grouping))[1:2],
            cols = c(1, 2, 3, 4),
            pchs = rep(1, 3),
            lty = c(1, 2, 1, 2),
            individual = TRUE, jointwithin = TRUE, jointall = TRUE,
            oneplusjoint = TRUE, jittering = TRUE, bcenterline = TRUE,
            xlim = NULL, ylim = NULL, xlab = "geographical distance",
            ylab = "genetic distance", ...)

Arguments

dmx  dissimilarity matrix or object of class dist. Explanatory dissimilarities (often these will be proper distances, but more general dissimilarities that do not necessarily fulfill the triangle inequality can be used, same for dmy).
dmy  dissimilarity matrix or object of class dist. Response dissimilarities.
grouping  something that can be coerced into a factor, defining the grouping of objects represented by the dissimilarities dmx and dmy (i.e., if grouping has length n, dmx and dmy must be dissimilarities between n objects).
groups  Vector of two levels. The two groups defining the regressions to be compared in the test. These can be factor levels, integer numbers, or strings, depending on the entries of grouping.
cols  vector of four colors (or color numbers) to be used for plotting distances and regression lines within the first group, within the second group, distances between groups, and a line marking the center of the between-groups explanatory distances, see col-argument of par.
pchs  vector of three plot symbols (or numbers) to be used for plotting distances within the first group, within the second group, and distances between groups, see pch-argument of par.
lty  vector of line type numbers to be used for single group within-group regression, both groups combined within-group regression, regression with all distances, and regression combining within-groups distances of one group with between-groups distances, see lty-argument of par.
individual  if TRUE, within-groups distances regression lines are shown for both groups.
jointwithin  if TRUE, the within-groups distances regression line for both groups combined is shown.
jointall if TRUE, the regression line based on all distances is shown.

oneplusjoint if TRUE, the regression lines combining within-groups distances of one group with between-groups distances are shown (the colors of these are the colors of the individual groups, the first two components of the cols-argument).

erasing if TRUE, points are jittered to avoid overplotting.

bccenterline if TRUE, a line is plotted to mark the center of the between-groups distances on the explanatory variable.

xlim to be passed on to plot; the default is determined from the involved distances.

ylim to be passed on to plot; the default is determined from the involved distances.

xlab to be passed on to plot.

ylab to be passed on to plot.

... optional arguments to be passed on to plot.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References


See Also

regeqdist, regdistbetween, regdistbetweenone, regdistdiffone

Examples

options(digits=4)
data(veronica)
ver.geo <- coord2dist(coordmatrix=veronica.coord[173:207,],file.format="decimal2")
vei <- prabinit(prabmatrix=veronica[173:207,],distance="jaccard")

species <-c(rep(1,13),rep(2,22))
loggeo <- log(ver.geo+quantile(as.vector(as.dist(ver.geo)),0.25))
plotdistreg(dmx=loggeo,dmy=vei$distmat,grouping=species,
jointwithin=FALSE,jointall=FALSE,groups=c(1,2))
legend(5,0.75,c("within species 1",
"within species 2","species 1 and between","species 2 and between"),lty=c(1,1,2,2),col=c(1,2,1,2))

plotdistreg(dmx=loggeo,dmy=vei$distmat,grouping=species,
jointwithin=TRUE,jointall=TRUE,oneplusjoint=FALSE,groups=c(1,2))
legend(5,0.75,c("within species 1",
"within species 2","all distances","all within species"),lty=c(1,1,1,2),col=c(1,2,3,3))
pop.sim

p-value simulation for presence-absence matrices clustering test

Description

Parametric bootstrap simulation of the p-value of a test of a homogeneity hypothesis against clustering (or significant nestedness). Designed for use within prabtest. The null model is defined by randpop.nb.

Usage

pop.sim(regmat, neighbors, h0c = 1, times = 200, dist = "kulczynski", teststat = "isovertice", testc = NULL, geodist=NULL, gtf=0.1, n.species = ncol(regmat), specperreg = NULL, regperspec = NULL, species.fixed=FALSE, pdfnb=FALSE, ignore.richness=FALSE)

Arguments

regmat 0-1-matrix. Columns are species, rows are regions.

neighbors A list with a component for every region. The components are vectors of integers indicating neighboring regions. A region without neighbors (e.g., an island) should be assigned a list numeric(0).

h0c numerical. Parameter p.nb for use in randpop.nb.

times integer. Number of simulation runs.

dist "kulczynski", "jaccard" or "geco", see kulczynski, geco and jaccard.

teststat "isovertice", "lcomponent", "distratio", "nn" or "inclusions". See the corresponding functions, homogen.test for "isovertice", incmatrix for "inclusions".

testc numerical. Tuning constant for the test statistics.

geodist matrix of non-negative reals. Geographical distances between regions. Only used if dist="geco".

gtf tuning constant for geco-distance if dist="geco", see "geco".

n.species integer. Number of species.

specperreg vector of integers. Numbers of species per region (is calculated from the data by default).

regperspec vector of integers. Number of regions per species (is calculated from the data by default).

species.fixed logical. If TRUE, the sizes of the species are taken directly from regmat. Otherwise, they are drawn by random from the empirical distribution of the values from regmat.

pdfnb logical. Probability correction in randpop.nb.

ignore.richness logical. If TRUE, there is no assumption of species richnesses to differ between regions in the null model. Regionwise probabilities don’t differ in the generation of null data.
Value

List with components

- results: vector of test statistic values for the simulated matrices.
- p.above: p-value if large test statistic leads to rejection.
- p.below: p-value if small test statistic leads to rejection.
- datac: test statistic value for the original data.
- testc: see above.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References


See Also

- prabtest, randpop.nb, jaccard, kulczynski, homogen.test, lcomponent, distratio, nn, incmatrix.

Examples

```r
options(digits=4)
data(kykladspecreg)
data(nb)
set.seed(1234)
pop.sim(t(kykladspecreg), nb, times=5, h0c=0.35, teststat="nn", testc=3)
```

---

**prab.sarestimate** *Estimates SAR model from log-abundance matrix of prab-object.*

Description

This is either an interface for the function `errorsarlm` for abundance data stored in an object of class `prab` implemented for use in `abundtest`, or, in case that spatial information should be ignored, it estimates a two-way additive unreplicated linear model for log-abundances on factors species and region.
\textit{prab.sareestimate}

Usage

\begin{verbatim}
prab.sareestimate(abmat, prab01=NULL, sarmethod="eigen",
                   weightstyle="C",
                   quiet=TRUE, sar=TRUE,
                   add.lmobject=TRUE)
\end{verbatim}

Arguments

\begin{itemize}
  \item \texttt{abmat} \hspace{1cm} object of class \texttt{prab}.
  \item \texttt{prab01} \hspace{1cm} presence-absence matrix of same dimensions than the abundance matrix of \texttt{prabobj}.
    This specifies the presences and absences on which the presence/absence step of abundance-based tests is based (see details). If \texttt{NULL} (which is usually the only reasonable choice), \texttt{prab01} is computed in order to indicate the nonzeroses of \texttt{prabobj$prab}.
  \item \texttt{sarmethod} \hspace{1cm} this is passed on as parameter method to \texttt{errorsarlm} and documented there. We don't have experience with any other choice than "eigen".
  \item \texttt{weightstyle} \hspace{1cm} can take values "W", "B", "C", "U", and "S" though tests suggest that "C" should be chosen. See \texttt{nb2listw}.
  \item \texttt{quiet} \hspace{1cm} this is passed on as parameter \texttt{quiet} to \texttt{errorsarlm} and documented there.
  \item \texttt{sar} \hspace{1cm} logical. If \texttt{TRUE}, a simultaneous autoregression model is fitted by calling \texttt{errorsarlm}. If \texttt{FALSE}, a two-way additive unreplicated linear model for log-abundances on factors species and region is computed by \texttt{lm}, ignoring the spatial arrangement of the regions.
  \item \texttt{add.lmobject} \hspace{1cm} logical. If \texttt{TRUE}, the whole output object of \texttt{errorsarlm} (or \texttt{lm}) is given out.
\end{itemize}

Value

A list with the following components:

\begin{itemize}
  \item \texttt{sar} \hspace{1cm} see above.
  \item \texttt{intercept} \hspace{1cm} numeric. Estimator of the intercept.
  \item \texttt{sigma} \hspace{1cm} numeric. Estimator of error standard deviation.
  \item \texttt{regeffects} \hspace{1cm} numeric vector. Estimator for region effects.
  \item \texttt{speceffects} \hspace{1cm} numeric vector. Estimator for species effects.
  \item \texttt{lamda} \hspace{1cm} numeric. Governs the degree of spatial autocorrelation. See \texttt{errorsarlm}.
  \item \texttt{size} \hspace{1cm} integer. Length of neighborhood list generated by \texttt{nb2listw} used by \texttt{errorsarlm}.
  \item \texttt{nbweight} \hspace{1cm} numeric. Average weight of neighbors.
  \item \texttt{lmobject} \hspace{1cm} if \texttt{add.lmobject=TRUE}, output object of either \texttt{lm} or \texttt{errorsarlm}.
\end{itemize}

Author(s)

Christian Hennig \texttt{<christian.hennig@unibo.it>} \url{https://www.unibo.it/sitoweb/christian.hennig/en}
See Also

errorsarlm, abundtest

Examples

options(digits=4)
data(siskiyou)
x <- prabinit(prabmatrix=siskiyou, neighborhood=siskiyou.nb, distance="none")
# Not run; this needs package spdep
# prab.sarestimate(x)
prab.sarestimate(x, sar=FALSE)

prabclust Clustering for biotic elements or for species delimitation (mixture method)

Description

Clusters a presence-absence matrix object (for clustering ranges/finding biotic elements, Hennig and Hausdorf, 2004) or an object of genetic information (for species delimitation, Hausdorf and Hennig, 2010) by calculating an MDS from the distances, and applying maximum likelihood Gaussian mixtures clustering with "noise" (package mclust) to the MDS points. The solution is plotted. A standard execution (using the default distance of prabinit) will be prabmatrix <- prabinit(file="path/prabmatrixfile", neighborhood="path/neighborhoodfile") clust <- prabclust(prabmatrix) print(clust)

Examples for species delimitation are given below in the examples section. Note: Data formats are described on the prabinit and alleleinit help pages. You may also consider the example datasets kykladspecreg.dat, nb.dat, Heterotrigona_indoFO.txt or MartinezOrtega04AFLP.dat. Note: prabclust calls the function mclustBIC in package mclust. An alternative is the use of hprabclust.

Usage

prabclust(prabobj, mdsmethod = "classical", mdssdim = 4, nnk = ceiling(prabobj$n.species/40), nclus = 0:9, modelid = "all", permutations=0)

## S3 method for class 'prabclust'
print(x, bic=FALSE, ...)

Arguments

prabobj object of class prab as generated by prabinit. Presence-absence data to be analyzed. (This can be geographical information for range clustering Can also be an object of class alleleobject as generated by alleleinit.
The `prabclust` function provides several arguments for specifying how to handle different aspects of the clustering process. Each argument is described below:

- **mdsfun**: The function to be used for the MDS method, which includes options for classical, kruskal, or sammon MDS.
- **mdsdim**: An integer representing the dimension of the MDS points. It's important to select the dimension to get a small stress (less than 5%, say).
- **nclus**: A vector of integers indicating the numbers of clusters to perform the mixture estimation.
- **modelid**: A string specifying the model name for `mclustBIC`, which can be either 'all' to compare all possible models or 'noVVV' to fit all methods except 'VVV'.
- **permutations**: An integer specifying the number of random permutations of the observations or the visualization of how the stress depends on `mdsdim` to choose `mdsdim` to get a small stress.
- **bic**: A logical value indicating whether information about the BIC criterion to choose the model is displayed.

The function `print.prabclust` does not produce output and generates an object of class `prabclust`. This object contains components related to clustering and summaries of the analysis.

### Details

It's important to note that if `mdsfun!="classical"`, zero distances between non-identical objects are replaced by the smallest nonzero distance divided by 10 to prevent the MDS methods from producing an error.

### Value

- `clustering`: A vector of integers indicating the cluster memberships of the species. Noise can be recognized by the output component `symbols`.
- `clustsummary`: An output object of `summary.mclustBIC`, giving optimal parameters, conditional probabilities ‘z’, and loglikelihood, together with the associated classification and its uncertainty. Note that the numbering of clusters may differ from `clustering`, see `csreorder`.
- `bicsummary`: An output object of `mclustBIC`, providing the Bayesian Information Criterion for the specified mixture models and numbers of clusters.
points  numerical matrix. MDS configuration.

nnk  see above.

mdsdim  see above.

mdsmethod  see above.

symbols  vector of characters, similar to clustering, but indicating estimated noise and points belonging to one-point-components (which should be interpreted as some kind of noise as well) by "N".

permchange  logical. If TRUE, permutations>0 has been used and the best solution is different from the one obtained by the standard ordering. (This is just for information and has no further operational consequences.)

Note

Note that we used mdsmethod="kruskal" in our publications, but mdsmethod="classical" is now the default, because of occasional numerical instabilities of the isoMDS-implementation for Jaccard, Kulczynski or geco distance matrices.

Sometimes, prabclust produces an error because mclustBIC cannot handle all models properly. In this case we recommend to change the modelid parameter. "noVVV" and "VVV" are reasonable alternative choices (one of these is expected to reproduce the error, but the other one might work).

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References


See Also

mclustBIC, summary.mclustBIC, NNclean, cmdscale, isoMDS, sammon, prabinit, hprabclust, alleleinit, stressvals.

Examples

# Biotic element/range clustering:
data(kykladspecreg)
data(nb)
set.seed(1234)
x <- prabinit(prabmatrix=kykladspecreg, neighborhood=nb)
# If you want to use your own ASCII data files, use
# x <- prabinit(file="path/prabmatrixfile",
# neighborhood="path/neighborhoodfile")
print(prabclust(x))

# Here is an example for species delimitation with codominant markers;
# only 50 individuals were used in order to have a fast example.
data(tetragonula)
ta <- alleleconvert(strmatrix=tetragonula[1:50,])
tai <- alleleinit(allelematrix=ta)
print(prabclust(tai))

# Here is an example for species delimitation with dominant markers;
# only 50 individuals were used in order to have a fast example.
# You may want to use stressvals to choose mdsdim.
data(veronica)
vei <- prabinit(prabmatrix=veronica[1:50,],distance="jaccard")
print(prabclust(vei,mdsmethod="kruskal",mdsdim=3))

---

prabinit

**Presence-absence/abundance matrix initialization**

**Description**

prabinit converts a matrix into an object of class prab (presence-absence). The matrix may be read from a file or an R-object. It may be a 0-1 matrix or a matrix with non-negative entries (usually abundances). print.prab is a print method for such objects.

Documentation here is in terms of biotic elements analysis (species are to be clustered). For species delimitation with dominant markers, see Hausdorf and Hennig (2010), individuals take the role of species and loci take the role of regions.

**Usage**

```r
prabinit(file = NULL, prabmatrix = NULL, rows.are.species = TRUE,
neighborhood = "none", nbbetweenregions=TRUE, geodist=NULL, gtf=0.1,
distance = "kulczynski", toprab = FALSE, toprabp = 0.05, outc = 5.2)
```

```r
# S3 method for class 'prab'
print(x, ...)
```

**Arguments**

- `file` string. non-negative matrix ASCII file (such as example dataset kykladspecreg.dat) from which the matrix is read by read.table. The usual interpretation is that it is a species-by-regions matrix of species presences/absences (0-1 matrix) or abundances.
prabmatrix matrix with non-negative entries. Either file or prabmatrix should be NA.
rows.are.species logical. If TRUE, rows are interpreted as species and columns are interpreted as regions. In this case, rows and columns are interchanged by prabinit.
neighborhood A string or a list with a component for every region. The components are vectors of integers indicating neighboring regions. A region without neighbors (e.g., an island) should be assigned a vector numeric(0). If neighborhood is a filename, it is attempted to read such a list from a file, where every row should correspond to one region (such as example dataset nb.dat). If neighborhood="none", all neighborhoods are set to numeric(0). The neighborhood can be tested by nbtest for consistency.
nb.betweenregions logical. If TRUE, the neighborhood is defined between regions as explained above. Otherwise it is defined between species (or individuals, if this is used for species delimitation).
geodist matrix of non-negative reals. Geographical distances between regions. Only used if distance="geco".
gtf tuning constant for geco-distance if distance="geco", see geco.
distance "kulczynski", "jaccard", "geco", "qkulczynski", "logkulczynski" (this calls function qkulczynski with log.distance=TRUE), "dice", or "none". The distance measure between species to compute by prabinit.
toprab logical. If TRUE, a presence-absence matrix is computed from the non-negative input matrix. "Absence", i.e., the entry 0, is chosen if the original entry is 0, or the original entry is smaller than or equal to toprabp times the sum of entries in the corresponding region, and log(original entry) is considered to be a lower outlier compared with the other entries of the corresponding species (see outc). "Presence", i.e., the entry 1, thus means that the original entry is non-negligible w.r.t. the species or w.r.t. the region.
toprabp numerical between 0 and 1, see toprab.
outc numerical. Tuning constant for the outlier identification associated with toprab=TRUE. An entry smaller than or equal to outc*mad times the median is considered as a lower outlier.
x object of class prab.
... necessary for print method.

Details
Species that are absent in all regions are omitted.

Value
prabinit produces an object of class prab, which is a list with components
distmat distance matrix between species.
prab abundance or presence/absence matrix (if presence/absence, the entries are logical). Rows are regions, columns are species.
prabinit

nb     neighborhood list, see above.
regperspec vector of the number of regions occupied by a species.
specperreg vector of the number of species present in a region.
n.species number of species (in the prab-object, see nonzero).
n.regions number of regions.
distance string denoting the chosen distance measure.
geodist non-negative matrix. see above.
gtf numeric. see above.
spatial TRUE, if there is a specified neighborhood structure.
nonempty.species logical vector. The length is the number of species in the original file/matrix. If FALSE, the corresponding species had only zero entries and was therefore absent. Note that these species are not included in any other component of a prab object, i.e., n.species is the number of TRUE-entries in nonzero.
nbbetweenregions see above.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References


See Also

read.table, jaccard, kulczynski, geco, qkulczynski, nbtest, alleleinit

Examples

# If you want to use your own ASCII data files, use
# x <- prabinit(file="path/prabmatrixfile",
# neighborhood="path/neighborhoodfile")
data(kylladspecreg)
data(nb)
prabinit(prabmatrix=kykladspecreg, neighborhood=nb)
prabtest

Parametric bootstrap test for clustering in presence-absence matrices

Description

Parametric bootstrap test of a null model of i.i.d., but spatially autocorrelated species against clustering of the species’ occupied areas (or alternatively nestedness). In spite of the lots of parameters, a standard execution (for the default test statistics, see parameter teststat below) will be

```r
prabmatrix <- prabinit(file="path/prabmatrixfile", neighborhood="path/neighborhoodfile")
test <- prabtest(prabmatrix)
summary(test)
```

Note: Data formats are described on the prabinit help page. You may also consider the example datasets kykladspecreg.dat and nb.dat. Take care of the parameter rows.are.species of prabinit.

Usage

```r
prabtest(prabobject, teststat = "distratio", tuning = switch(teststat,
    distratio = 0.25, lcomponent = floor(3 * ncol(prabobject$distmat)/4),
    isovertice = ncol(prabobject$distmat), nn = 4, NA), times = 1000,
    pd = NULL, prange = c(0, 1), nperp = 4, step = 0.1, step2=0.01,
    twostep = TRUE,
    sf.sim = FALSE, sf.const = sf.sim, pdfnb = FALSE, ignore.richness=FALSE)
```

```r
## S3 method for class 'prabtest'
summary(object, above.p=object$teststat %in%
    c("groups","inclusions","mean"),
    group.outmean=FALSE,...)
```

```r
## S3 method for class 'summary.prabtest'
print(x, ...)
```

Arguments

- **prabobject**: an object of class prab (presence-absence data), as generated by prabinit.
- **teststat**: string, indicating the test statistics. "isovertice": number of isolated vertices in the graph of tuning smallest distances between species. "lcomponent": size of largest connectivity component in this graph. "distratio": ratio between tuning smallest and largest distances. "nn": average distance of species to tuning nearest neighbor. "inclusions": number of inclusions between areas of different species (tests for nestedness structure, not for clustering).
- **tuning**: integer or (if teststat="distratio") numerical between 0 and 1. Tuning constant for test statistics, see teststat.
- **times**: integer. Number of simulation runs.
prabtest

`pd` numerical between 0 and 1. The probability that a new region is drawn from the non-neighborhood of the previous regions belonging to a species under generation. If `NA` (the default), `prabtest` estimates this by function `autoconst`. Otherwise the next five parameters have no effect.

`prange` numerical range vector, lower value not smaller than 0, larger value not larger than 1. Range where `pd` is to be found. Used by function `autoconst`.

`nperp` integer. Number of simulations per `pd`-value. Used by function `autoconst`.

`step` numerical between 0 and 1. Interval length between subsequent choices of `pd` for the first simulation. Used by function `autoconst`.

`step2` numerical between 0 and 1. Interval length between subsequent choices of `pd` for the second simulation (see parameter `twostep`). Used by function `autoconst`.

`twostep` logical. If `TRUE`, a first estimation step for `pd` is carried out in the whole `prange`, and then the final estimation is determined between the preliminary estimator `-5*step2` and `+5*step2`. Else, the first simulation determines the final estimator. Used by function `autoconst`.

`sf.sim` logical. Indicates if the range sizes of the species are held fixed in the test simulation (`TRUE`) or generated from their empirical distribution in `x` (`FALSE`). See function `randpop.nb`.

`sf.const` logical. Same as `sf.sim`, but for estimation of `pd` by `autoconst`.

`pdfnb` logical. If `TRUE`, the probabilities of the regions are modified according to the number of neighboring regions in `randpop.nb`, see Hennig and Hausdorf (2002), p. 5. This is usually no improvement.

`ignore.richness` logical. If `TRUE`, there is no assumption of species richesses to differ between regions in the null model. Regionwise probabilities don’t differ in the generation of null data.

`object` object of class `prabtest`.

`above.p` logical. `TRUE` means that for output from `abundtest` the p-value is `p.above`, otherwise `p.below`.

`group.outmean` logical. If `TRUE` and `object$teststat="groups"`, statistics concerning the mean of all dissimilarities are given out by `print.summary.prabtest`.

`x` object of class `summary.prabtest`.

`...` no meaning, necessary for print and summary methods.

Details

From the original data, the distribution of the range sizes of the species, the autocorrelation parameter `pd` (estimated by `autoconst`) and the distribution on the regions induced by the relative species numbers are taken. With these parameters, times populations according to the null model implemented in `randpop.nb` are generated and the test statistic is evaluated. The resulting p-value is number of simulated statistic values more extreme than than the value of the original data+1 divided by times+1. "More extreme" means smaller for "lcomponent", "distratio", "nn", larger for "inclusions", and twice the smaller number between the original statistic value and the "border", i.e., a two-sided test for "isovertice". If `pd=NA` was specified, a diagnostic plot for the estimation of `pd` is plotted by `autoconst`. For details see Hennig and Hausdorf (2004) and the help pages of the cited functions.
Value

`prabtest` produces an object of class `prabtest`, which is a list with components

- `results` vector of test statistic values for all simulated populations.
- `datac` test statistic value for the original data.
- `p.value` the p-value.
- `tuning` see above.
- `pd` see above.
- `reg` regression coefficients from `autoconst`.
- `teststat` see above.
- `distance` the distance measure chosen, see `prabinit`.
- `gtf` the geco-distance tuning parameter (only informative if `distance="geco"`), see `prabinit`.
- `times` see above.
- `pdfnb` see above.
- `ignore.richness` see above.

`summary.prabtest` produces an object of class `summary.prabtest`, which is a list with components

- `rrange` range of the simulation results (test statistic values) of object.
- `rmean` mean of the simulation results (test statistic values) of object.
- `datac, p.value, pd, tuning, teststat, distance, times, pdfnb, abund, sarlambda` directly taken from object, see `prabtest` and `abundtest`.
- `groupinfo` if `object$teststat="groups"`, components `rrangeg` (matrix of group-wise ranges of test statistic value), `rmeang` (vector of group-wise means of test statistic value), `rrangem` (range over simulations of overall mean of within-group dissimilarities), `rmeanm` (mean over simulations of overall mean of within-group dissimilarities) are added to the list `object$groupinfo`, and this is given out.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References


See Also

prabininit generates objects of class prab.
autoconst estimates pd from such objects.
randpop.nb generates populations from the null model. An alternative model is given by cluspop.nb.

Some more information on the test statistics is given in homogen.test, lcomponent, distratio, nn, incmatrix.

The simulations are computed by pop.sim.

Examples

options(digits=4)
data(kykladspecreg)
data(nb)
set.seed(1234)
x <- prabininit(prabmatrix=kykladspecreg, neighborhood=nb)
# If you want to use your own ASCII data files, use
# x <- prabininit(file="path/prabmatrixfile",
# neighborhood="path/neighborhoodfile")
kpt <- prabtest(x, times=5, pd=0.35)
# These settings are chosen to make the example execution
# a bit faster; usually you will use prabtest(kprab).
summary(kpt)

qkulczynski

Quantitative Kulczynski distance matrix

Description

Computes quantitative Kulczynski distances between the columns of an abundance matrix.

Usage

qkulczynski(regmat, log.distance=FALSE)

Arguments

regmat (non-negative) abundance matrix. Columns are species, rows are regions.
log.distance logical. If TRUE, 1 is added to the abundance matrix and then the logs of the values are taken in order to compute the distance.

Details

The quantitative Kulczynski distance between two species is 1-(mean of (mean of over regions minimum abundance of both species)/(sum of abundances of species 1) and (mean of over regions minimum abundance of both species)/(sum of abundances of species 2)). If the abundance matrix is a 0-1-matrix, this gives the standard Kulczynski distance.
Value

A symmetrical matrix of quantitative Kulczynski distances.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian. hennig/en

References


See Also

kulczynski

Examples

```r
options(digits=4)
data(kykladspecreg)
qkulczynski(t(kykladspecreg))
```

Description

Generates a simulated matrix where the rows are interpreted as regions and the columns as species, 1 means that a species is present in the region and 0 means that the species is absent. Species are generated i.i.d.. Spatial autocorrelation of a species’ presences is governed by the parameter p.nb and a list of neighbors for each region.

Usage

```r
randpop.nb(neighbors, p.nb = 0.5, n.species, n.regions = length(neighbors), vector.species = rep(1, n.species),
species.fixed = FALSE, pdf.regions = rep(1/n.regions, n.regions),
count = TRUE, pdfnb = FALSE)
```

Arguments

- `neighbors` A list with a component for every region. The components are vectors of integers indicating neighboring regions. A region without neighbors (e.g., an island) should be assigned a list numeric(0).
The probability that a new region is drawn from the non-neighborhood of the previous regions belonging to a species under generation. Note that for a given presence-absence matrix, this parameter can be estimated by autoconst (called pd there).

integer. Number of species.

integer. Number of regions.

vector of integers. If species.fixed=TRUE, vector.species must have length n.species and gives the sizes (i.e., numbers of regions) of the species to generate. Else, the sizes are generated randomly from the empirical distribution of vector.species.

logical. See vector.species.

numerical vector of length n.species. The entries must sum up to 1 and give probabilities for the regions to be drawn during the generation of a species. These probabilities are used conditional on the new region being a neighbor or a non-neighbor of the previous regions of the species, see p.nb.

logical. If TRUE, the number of the currently generated species is printed.

logical. If TRUE, the probabilities of the regions are modified according to the number of neighboring regions by dividing them relative to the others by min(1,number of neighbors).

The principle is that a single species with given size is generated one-by-one region. The first region is drawn according to pdf.regions. For all following regions, a neighbor or non-neighbor of the previous configuration is added (if possible), as explained in pdf.regions, p.nb.

A 0-1-matrix, rows are regions, columns are species.

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en


See Also

autoconst estimates p.nb from matrices of class prab. These are generated by prabinit.

prabtest uses randpop nb as a null model for tests of clustering. An alternative model is given by cluspop.nb.

Examples

data(nb)
set.seed(2346)
randpop.nb(nb, p.nb=0.1, n.species=5, vector.species=c(1,10,20,30,34))

```
regdist                  Regression between subsets of dissimilarity matrices
```

Description

Given two dissimilarity matrices dmx and dmy and an indicator vector x, this computes a standard least squares regression between the dissimilarity between objects indicated in x.

Usage

```
regdist(x, dmx, dmy, xcenter=0, param)
```

Arguments

- **x**: vector of logicals of length of the number of objects on which dissimilarities dmx and dmy are based.
- **dmx**: dissimilarity matrix or object of class dist. Explanatory dissimilarities.
- **dmy**: dissimilarity matrix or object of class dist. Response dissimilarities.
- **xcenter**: numeric. Dissimilarities dmx are centered by this, i.e., this value is subtracted from the dissimilarities before regression.
- **param**: 1 or 2 or NULL. If 1 or 2, only the first or second parameter (intercept or slope) of the regression is given out.

Value

If param=NULL, the output object of lm. If param=1 the intercept. If param=2 the slope.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en


References


Examples

```r
options(digits=4)
data(veronica)
ver.geo <- coord2dist(coordmatrix=veronica.coord[1:20,],file.format="decimal2")
vei <- prabinit(prabmatrix=veronica[1:20,],distance="jaccard")
regdist(c(rep(TRUE,10),rep(FALSE,10)),ver.geo,vei$distmat,param=1)
```

\[
\text{regdistbetween}\quad \text{Testing equality of within-groups and between-groups distances regression}
\]

Description

Jackknife-based test for equality of two regressions between distances. Given two groups of objects, this tests whether the regression involving all distances is compatible with the regression involving within-group distances only.

Usage

```r
regdistbetween(dmx,dmy,grouping,groups=levels(as.factor(grouping))[1:2])
```

## S3 method for class 'regdistbetween'

```r
print(x,...)
```

Arguments

- **dmx**: dissimilarity matrix or object of class dist. Explanatory dissimilarities (often these will be proper distances, but more general dissimilarities that do not necessarily fulfill the triangle inequality can be used, same for dmy).
- **dmy**: dissimilarity matrix or object of class dist. Response dissimilarities.
- **grouping**: something that can be coerced into a factor, defining the grouping of objects represented by the dissimilarities dmx and dmy (i.e., if grouping has length n, dmx and dmy must be dissimilarities between n objects).
- **groups**: Vector of two levels. The two groups defining the regressions to be compared in the test. These can be factor levels, integer numbers, or strings, depending on the entries of grouping.
- **x**: object of class "regdistbetween".
- **...**: optional arguments for print method.
Details

The null hypothesis that the regressions based on all distances and based on within-group distances only are equal is tested using jackknife pseudovalues. This assumes that a single regression is appropriate at least for the within-group distances alone. The test statistic is the difference between fitted values with x (explanatory variable) fixed at the center of the between-group distances. The test is run one-sided, i.e., the null hypothesis is only rejected if the between-group distances are larger than expected under the null hypothesis, see below.

The test cannot be run in case that within-group regressions or jackknifed within-group regressions are ill-conditioned.

This was implemented having in mind an application in which the explanatory distances represent geographical distances, the response distances are genetic distances, and groups represent species or species-candidates. In this application, for testing whether the regression patterns are compatible with the two groups behaving like a single species, one would first use `regeqdist` to test whether a joint regression for the within-group distances of both groups makes sense. If this is not rejected, `regdistbetween` is run to see whether the between-group distances are compatible with the within-group distances. This is only rejected if the between-group distances are larger than expected under equality of regressions, because if they are smaller, this is not an indication against the groups belonging together genetically.

If a joint regression on within-group distances is rejected by `regeqdist`, `regdistbetweenone` can be used to test whether the between-group distances are at least compatible with the within-group distances of one of the groups, which can still be the case within a single species, see Hausdorf and Hennig (2019).

Value

list of class "regdistbetween" with components

- `pval`: p-value.
- `coeffdiff`: difference between regression fits (all distances minus within-group distances only) at `xcenterbetween`, see below.
- `condition`: condition numbers of regressions, see `kappa`.
- `lmfit`: list. Output objects of `lm` within the two groups.
- `jr`: output object of `jackknife` for difference between regression fitted values at `xcenterbetween`.
- `xcenter`: mean of within-groups distances of explanatory variable, used for centering.
- `xcenterbetween`: mean of between-groups distances of explanatory variable (after centering by `xcenter`); at this point regression fitted values are computed.
- `tstat`: t-statistic.
- `tdf`: degrees of freedom of t-statistic.
- `jackest`: jackknife-estimator of difference between regression fitted values at `xcenterbetween`.
- `jackse`: jackknife-standard error for `jackest`.
- `jackpseudo`: vector of jacknife pseudovalues on which the test is based.
- `testname`: title to be printed out when using `print.regdistbetween`.
- `groups`: see above.
Jackknife-based test for equality of two regressions between distances. Given two groups of objects, this tests whether the regression involving the distances within one of the groups is compatible with the regression involving the same within-group distances together with the between group distances.

**Usage**

```r
regdistbetweenone(dmx, dmy, grouping, groups = levels(as.factor(grouping))[1:2], rgroup)
```

**Arguments**

- `dmx` dissimilarity matrix or object of class `dist`. Explanatory dissimilarities (often these will be proper distances, but more general dissimilarities that do not necessarily fulfill the triangle inequality can be used, same for `dmy`).
- `dmy` dissimilarity matrix or object of class `dist`. Response dissimilarities.
regdistbetweenone

grouping
something that can be coerced into a factor, defining the grouping of objects represented by the
dissimilarities dmx and dmy (i.e., if grouping has length n, dmx and dmy must be dissimilarities
between n objects).

groups
vector of two levels. The two groups defining the regressions to be compared in
the test. These can be factor levels, integer numbers, or strings, depending on
the entries of grouping.

rgroup
one of the levels in groups, denoting the group of which within-group dissimi-
larities are considered.

Details

The null hypothesis that the regressions based on the distances within group species and based
on these distances together with the between-groups distances are equal is tested using jackknife
pseudovalues. The test statistic is the difference between fitted values with x (explanatory variable)
fixed at the center of the between-group distances. The test is run one-sided, i.e., the null hypothesis
is only rejected if the between-group distances are larger than expected under the null hypothesis,
see below. For the jackknife, observations from both groups are left out one at a time. However,
the roles of the two groups are different (observations from group species are used in both regres-
sions whereas observations from the other group are only used in one of them), and therefore the
corresponding jackknife pseudovalues can have different variances. To take this into account, vari-
ances are pooled, and the degrees of freedom of the t-test are computed by the Welch-Sattertwaite
approximation for aggregation of different variances.

The test cannot be run and many components will be NA in case that within-group regressions or
jackknifed within-group regressions are ill-conditioned.

This was implemented having in mind an application in which the explanatory distances represent
geographical distances, the response distances are genetic distances, and groups represent species
or species-candidates. In this application, for testing whether the regression patterns are compatible
with the two groups behaving like a single species, one would first use regeqdist to test whether
a joint regression for the within-group distances of both groups makes sense. If this is not re-
jected, regdistbetween is run to see whether the between-group distances are compatible with the
within-group distances. If a joint regression on within-group distances is rejected by regeqdist,
regdistbetweenone can be used to test whether the between-group distances are at least compati-
bles with the within-group distances of one of the groups, which can still be the case within a single
species, see Hausdorf and Hennig (2019). This is only rejected if the between-group distances are
larger than expected under equality of regressions, because if they are smaller, this is not an indica-
tion against the groups belonging together genetically. To this end, regdistbetweenone needs to
be run twice using both groups as species. This will produce two p-values. The null hypothesis
that the regressions are compatible for at least one group can be rejected if the maximum of the two
p-values is smaller than the chosen significance level.

Value

list of class "regdistbetween" with components

pval        p-value.
coefficient difference between regression fits (within-group together with between-groups
distances minus within-group distances only) at xcenterbetween, see below.
condition   condition numbers of regressions, see kappa.
lmfit  list. Output objects of lm within the two groups.
jr  output object of jackknife for difference between regression fitted values at xcenterbetween.
xcenter  mean of within-group distances for group species of explanatory variable, used for centering.
xcenterbetween  mean of between-groups distances of explanatory variable (after centering by xcenter); at this point regression fitted values are computed.
tstat  t-statistic.
tdf  degrees of freedom of t-statistic according to Welch-Sattertwaith approximation.
jackest  jackknife-estimator of difference between regression fitted values at xcenterbetween.
jackse  jackknife-standard error for jackest.
jackpseudo  vector of jacknife pseudovalues on which the test is based.
groups  see above.
species  see above.
testname  title to be printed out when using print.regdistbetween.

Author(s)
Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References

See Also
regeqdist, regdistbetweenone

Examples
options(digits=4)
data(veronica)
ver.geo <- coord2dist(coordmatrix=veronica.coord[173:207,],file.format="decimal2")
vei <- prabinit(prabmatrix=veronica[173:207,],distance="jaccard")

species <-c(rep(1,13),rep(2,22))
loggeo <- log(ver.geo+quantile(as.vector(as.dist(ver.geo)),0.25))
rtest3 <- regdistbetweenone(dmx=loggeo,dmy=vei$distmat,grouping=species,groups=c(1,2),rgroup=1)
print(rtest3)
**regdistdiff**  
*Regression difference between within-group dissimilarities*

**Description**

Given two dissimilarity matrices \(d_{mx}\) and \(d_{my}\), an indicator vector \(x\) and a grouping, this computes the difference between standard least squares regression predictions at point \(x_{centerbetween}\). The regressions are based on the dissimilarities in \(d_{mx}\) vs. \(d_{my}\) for objects indicated in \(x\). Grouping indicates the two groups, and the difference is computed between regressions based on the within-group distances of the two groups.

**Usage**

```r
regdistdiff(x, dmx, dmy, grouping, xcenter=0, xcenterbetween=0)
```

**Arguments**

- **x**  
  vector of logicals of length of the number of objects on which dissimilarities \(d_{mx}\) and \(d_{my}\) are based.

- **dmx**  
  dissimilarity matrix or object of class `dist`. Explanatory dissimilarities.

- **dmy**  
  dissimilarity matrix or object of class `dist`. Response dissimilarities.

- **grouping**  
  vector of length of the number of objects on which dissimilarities \(d_{mx}\) and \(d_{my}\) are based. Grouping vector. Regressions will be based on the first two values that appear in `unique(grouping[x])` (note that objects that are not assigned to one of these groups will be ignored); normally grouping should indicate only two groups on the objects with \(x=TRUE\), and then these are used.

- **xcenter**  
  numeric. Dissimilarities \(d_{mx}\) are centered by this, i.e., this value is subtracted from the dissimilarities before regression.

- **xcenterbetween**  
  numeric. This specifies the x- (dissimilarity) value at which predictions from the two regressions are compared. Note that this is interpreted as after centering by \(x_{center}\).

**Value**

Difference between standard least squares regression predictions for the two groups at point \(x_{centerbetween}\).

**Author(s)**

Christian Hennig  
<christian.hennig@unibo.it>  
https://www.unibo.it/sitoweb/christian.hennig/en

**References**

See Also
regdistbetween

Examples
options(digits=4)
data(veronica)
ver.geo <- coord2dist(coordmatrix=veronica.coord[173:207,],file.format="decimal2")
vei <- prabinit(prabmatrix=veronica[173:207,],distance="jaccard")
species <- c(rep(1,13),rep(2,22))
regdistdiff(rep(TRUE,35),ver.geo,vei$distmat,grouping=species,xcenter=0,xcenterbetween=100)

regdistdiffone

Regression difference within reference group and between-group dissimilarities

Description
Given two dissimilarity matrices \( \text{dmx} \) and \( \text{dmy} \), an indicator vector \( \text{x} \) and a grouping, this computes the difference between standard least squares regression predictions at point \( \text{xcenterbetween} \). The regressions are based on the dissimilarities in \( \text{dmx} \) vs. \( \text{dmy} \) for objects indicated in \( \text{x} \). groupindication the two groups, and the difference is computed between regressions based on (a) the within-group distances of the reference group \( \text{rgroup} \) and (b) these together with the between-group distances.

Usage

regdistdiffone(x, dmx, dmy, grouping, xcenter=0, xcenterbetween=0, rgroup)

Arguments

\( x \) vector of logicals of length of the number of objects on which dissimilarities \( \text{dmx} \) and \( \text{dmy} \) are based.
\( \text{dmx} \) dissimilarity matrix or object of class \text{dist}. Explanatory dissimilarities.
\( \text{dmy} \) dissimilarity matrix or object of class \text{dist}. Response dissimilarities.
\( \text{grouping} \) vector of length of the number of objects on which dissimilarities \( \text{dmx} \) and \( \text{dmy} \) are based. Grouping vector. Regressions will be based on the first two values that appear in \text{unique(grouping[x])} (note that objects that are not assigned to one of these groups will be ignored); normally grouping should indicate only two groups on the objects with \( x=\text{TRUE} \), and then these are used.
\( \text{xcenter} \) numeric. Dissimilarities \( \text{dmx} \) are centered by this, i.e., this value is subtracted from the dissimilarities before regression.
xcenterbetween numeric. This specifies the x- (dissimilarity) value at which predictions from the two regressions are compared. Note that this is interpreted as after centering by xcenter.

rgroup one of the values of grouping, specifying the reference group.

Value
Difference between standard least squares regression predictions for the two regressions at point xcenterbetween.

Author(s)
Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References

See Also
regdistbetweenone

Examples
options(digits=4)
data(veronica)
ver.geo <- coord2dist(coordmatrix=veronica.coord[173:207,],
  file.format="decimal2")
vei <- prabinit(prabmatrix=veronica[173:207,],distance="jaccard")

species <-c(rep(1,13),rep(2,22))
regdistdiffone(rep(TRUE,35),ver.geo,vei$distmat,grouping=species,
xcenter=0,xcenterbetween=100,rgroup=2)

regeqdist Testing equality of two distance-regressions

Description
Jackknife-based test for equality of two regressions between distance matrices.

Usage
regeqdist(dmx,dmy,grouping,groups=levels(as.factor(grouping))[1:2])

# S3 method for class 'regeqdist'
print(x,...)
Arguments

- **dmx**: dissimilarity matrix or object of class `dist`. Explanatory dissimilarities (often these will be proper distances, but more general dissimilarities that do not necessarily fulfill the triangle inequality can be used, same for `dmy`).
- **dmy**: dissimilarity matrix or object of class `dist`. Response dissimilarities.
- **grouping**: something that can be coerced into a factor, defining the grouping of objects represented by the dissimilarities `dmx` and `dmy` (i.e., if `grouping` has length `n`, `dmx` and `dmy` must be dissimilarities between `n` objects).
- **groups**: Vector of two, indicating the two groups defining the regressions to be compared in the test. These can be factor levels, integer numbers, or strings, depending on the entries of `grouping`.
- **x**: object of class "regeqdist".
- **...**: optional arguments for print method.

Details

The null hypothesis that the regressions within the two groups are equal is tested using jackknife pseudovalues independently in both groups allowing for potentially different variances of the pseudovalues, and aggregating as in Welch’s t-test. Tests are run separately for intercept and slope and aggregated by Bonferroni’s rule.

The test cannot be run and many components will be `NA` in case that within-group regressions or jackknifed within-group regressions are ill-conditioned.

This was implemented having in mind an application in which the explanatory distances represent geographical distances, the response distances are genetic distances, and groups represent species or species-candidates. In this application, for testing whether the regression patterns are compatible with the two groups behaving like a single species, one would first use `regeqdist` to test whether a joint regression for the within-group distances of both groups makes sense. If this is not rejected, `regdistbetween` is run to see whether the between-group distances are compatible with the within-group distances. On the other hand, if a joint regression on within-group distances is rejected, `regdistbetweenone` can be used to test whether the between-group distances are at least compatible with the within-group distances of one of the groups, which can still be the case within a single species, see Hausdorf and Hennig (2019).

Value

- list of class "regeqdist" with components
  - **pval**: p-values for intercept and slope.
  - **coeffdiff**: vector of differences between groups (first minus second) for intercept and slope.
  - **condition**: condition numbers of regressions, see `kappa`.
  - **lmfit**: list. Output objects of `lm` within the two groups.
  - **jr**: list of two lists of two; output object of `jackknife` within the two groups for intercept and slope.
  - **xcenter**: mean of `dmx` within the two groups used for centering.
  - **tstat**: t-statistic.
regpop.sar Simulation of abundance matrices (non-clustered)

Description

Generates a simulated matrix where the rows are interpreted as regions and the columns as species, and the entries are abundances. Species are generated i.i.d. in two steps. In the first step, a presence-absence matrix is generated as in randpop.nb. In the second step, conditionally on presence in the first step, abundance values are generated according to a simultaneous autoregression (SAR) model for the log-abundances (see errorsarlm for the model; estimates are provided by the parameter sareestimate). Spatial autocorrelation of a species’ presences is governed by the parameter p.nb, sareestimate and a list of neighbors for each region.

tdf vector of degrees of freedom of t-statistic according to Welch-Sattertwaite approximation for intercept and slope.

jackest jackknife-estimator of difference between regressions; vector with intercept and slope difference.

jackse vector with jackknife-standard errors for jackest, intercept and slope.

jackpseudo list of two lists of vectors; jacknife pseudovalues within both groups for intercept and slope estimators.

groups see above.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References


See Also

regdistbetween, regdistbetweenone

Examples

options(digits=4)
data(veronica)
ver.geo <- coord2dist(coordmatrix=veronica.coord[173:207,],file.format="decimal2")
vei <- prabininit(prabmatrix=veronica[173:207,],distance="jaccard")
loggeo <- log(ver.geo+quantile(as.vector(as.dist(ver.geo)),0.25))

species <-c(rep(1,13),rep(2,22))
rtest <- regeqdist(dmx=loggeo,dmy=vei$distmat,grouping=species,groups=c(1,2))
print(rtest)
Usage

dpop.sar(abmat, prab01=NULL, sarestimate=prab.sarestimate(abmat),
    p.nb=NULL,
    vector.species=prab01$regperspec,
    pdf.regions=prab01$specperreg/(sum(prab01$specperreg)),
    count=FALSE)

Arguments

abmat object of class prab, containing the abundance or presence/absence data.

prab01 presence-absence matrix of same dimensions than the abundance matrix of prabobj. This specifies the presences and absences on which the presence/absence step of abundance-based tests is based (see details). If NULL (which is usually the only reasonable choice), prab01 is computed in order to indicate the nonzeroes of prabobj$prab.

sarestimate Estimator of the parameters of a simultaneous autoregression model corresponding to the null model for abundance data from Hausdorf and Hennig (2007) as generated by prab.sarestimate. This requires package spdep. If sarestimate$sar=FALSE, spatial structure is ignored for generating the abundance values.

p.nb numeric between 0 and 1. The probability that a new region is drawn from the non-neighborhood of the previous regions belonging to a species under generation. If NULL, the spatial structure of the regions is ignored. Note that for a given presence-absence matrix, this parameter can be estimated by autoconst (called pd there).

vector.species vector of integers. vector.species gives the sizes (i.e., numbers of regions) of the species to generate.

pdf.regions numerical vector of length n.species. The entries must sum up to 1 and give probabilities for the regions to be drawn during the generation of a species. These probabilities are used conditional on the new region being a neighbor or a non-neighbor of the previous regions of the species, see p.nb.

count logical. If TRUE, the number of the currently generated species is printed.

Value

A matrix of abundance values, rows are regions, columns are species.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

References

See Also

`autoconst` estimates `p.nb` from matrices of class `prab`. These are generated by `prabinit`.

`abundtest` uses `regpop.sar` as a null model for tests of clustering.

`randpop.nb` (analogous function for simulating presence-absence data)

Examples

```r
options(digits=4)
data(siskiyou)
set.seed(1234)
x <- prabinit(prabmatrix=siskiyou, neighborhood=siskiyou.nb,
  distance="none")
# Not run; this needs package spdep.
# regpop.sar(x, p.nb=0.046)
regpop.sar(x, p.nb=0.046, sarestimate=prab.sarestimate(x,sar=FALSE))
```

---

**siskiyou**

*Herbs of the Siskiyou Mountains*

**Description**

Distributions of species of herbs in relation to elevation on quartz diorite in the central Siskiyou Mountains. All values are per mille frequencies in transects (The number of 1 m2 quadrats, among 1000 such quadrats, in which a species was observed, based on 1250 1m2 quadrats in the first 5 transects, and 400 1m2 quadrats in 6. transect). Observed presences in the transect, outside the sampling plots, were coded as 0.2. Rows correspond to species, columns correspond to regions.

**Usage**

```r
data(siskiyou)
```

**Format**

Three objects are generated:

- **siskiyou** numeric matrix giving the 144*6 abundance values.
- **siskiyou.nb** neighborhood list for the 6 regions.
- **siskiyou.groups** integer vector of length 144, giving group memberships for the 144 species.

**Details**

Reads from example data files LeiMik1.dat, LeiMik1NB.dat, LeiMik1G.dat.

**Source**

**specgroups**

**Examples**

```r
data(siskiyou)
```

---

**Description**

Generates average within-group distances (overall and group-wise) from a dissimilarity matrix and a given grouping.

**Usage**

```r
specgroups(distmat, groupvector, groupinfo)
```

**Arguments**

- `distmat`: dissimilarity matrix or `dist`-object.
- `groupvector`: integer vector. For every row of `distmat`, a number indicating the group membership.
- `groupinfo`: list with components `lg` (levels of `groupvector`), `ng` (number of groups), `nsg` (vector of group sizes).

**Value**

A list with parameters

- `overall`: overall average within-groups dissimilarity.
- `gr`: vector of group-wise average within-group dissimilarities (this will be NaN if the group size is only 1).

**Author(s)**

Christian Hennig &lt;christian.hennig@unibo.it&gt; [https://www.unibo.it/sitoweb/christian.hennig/en](https://www.unibo.it/sitoweb/christian.hennig/en)

**Examples**

```r
options(digits=4)
data(siskiyou)
x <- prabinit(prabmatrix=siskiyou, neighborhood=siskiyou.nb, 
distance="logkulczynski")
groupvector <- as.factor(siskiyou.groups)
ng <- length(levels(groupvector))
lg <- levels(groupvector)
nsg <- numeric(0)
for (i in 1:ng) nsg[i] <- sum(groupvector==lg[i])
groupinfo <- list(lg=lg, ng=ng, nsg=nsg)
specgroups(x$distmat, groupvector, groupinfo)
```
stressvals  Stress values for different dimensions of Kruskal’s MDS

Description

Computes Kruskal’s nonmetric multidimensional scaling isoMDS on alleleobject or prab-objects for different output dimensions in order to compare stress values.

Usage

stressvals(x, mdsdim=1:12, trace=FALSE)

Arguments

x  object of class alleleobject or link(prab). generated by alleleinit or prabin.
mdsdim  integer vector of MDS numbers of dimensions to be tried.
trace  logical. trace-argument for isoMDS (should trace information be printed during execution?).

Details

Note that zero distances between non-identical objects are replaced by the smallest nonzero distance divided by 10 to prevent isoMDS from producing an error.

Value

A list with components

MDSstress  vector of stress values.
mdsout  list of full outputs of isoMDS.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

Examples

options(digits=4)
data(tetragonula)
set.seed(112233)
taiselect <- sample(236,40)
# Use data subset to make execution faster.
tnb <- coord2dist(coordmatrix=tetragonula.coord[taiselect,],
cut=50, file.format="decimal2", neighbors=TRUE)
ta <- alleleconvert(strmatrix=tetragonula[taiselect,])
tetragonula

```r
tai <- alleleinit(allelematrix=ta,neighborhood=tnb$nblist)
stressvals(tai, mdsdim=1:3)$MDSstress
```

tetragonula  
Microsatellite genetic data of Tetragonula bees

### Description

Genetic data for 236 Tetragonula (Apidae) bees from Australia and Southeast Asia, see Franck et al. (2004). The data give pairs of alleles (codominant markers) for 13 microsatellite loci.

### Usage

```r
data(tetragonula)
```

### Format

Two objects are generated:

- **tetragonula**: A data frame with 236 observations and 13 string variables. Strings consist of six digits each. The format is derived from the data format used by the software GENEPOP (Rousset 2008). Alleles have a three digit code, so a value of "258260" on variable V10 means that on locus 10 the two alleles have codes 258 and 260. "000" refers to missing values.

- **tetragonula.coord**: a 236*2 matrix. Coordinates of locations of individuals in decimal format, i.e. the first number is latitude (negative values are South), with minutes and seconds converted to fractions. The second number is longitude (negative values are West).

### Details

Reads from example data file Heterotrigona_indoF0.dat.

### Source


### Examples

```r
data(tetragonula)
```
### toprab

**Convert abundance matrix into presence/absence matrix**

**Description**

Converts abundance matrix into binary (logical) presence/absence matrix (TRUE if abundance>0).

**Usage**

```r
toprab(prabobj)
```

**Arguments**

- `prabobj`: object of class `prab`.

**Value**

Logical matrix with same dimensions as `prabobj$prab` as described above.

**Author(s)**

Christian Hennig &lt;christian.hennig@unibo.it&gt; [https://www.unibo.it/sitoweb/christian.hennig/en](https://www.unibo.it/sitoweb/christian.hennig/en)

**Examples**

```r
data(siskiyou)
x <- prabinit(prabmatrix=siskiyou, neighborhood=siskiyou.nb, distance="none")
toprab(x)
```

---

### unbuild.charmatrix

**Internal: create allele list out of character matrix**

**Description**

Creates a list of lists, such as required by `alleledist`, from the charmatrix component of an `alleleobject`.

**Usage**

```r
unbuild.charmatrix(charmatrix, n.individuals, n.variables)
```
Arguments

charmatrix matrix of characters in which there are two rows for every individual corresponding to the two alleles in every locus (column). Entries are allele codes but missing values are coded as NA.
n.individuals integer. Number of individuals.
n.variables integer. Number of loci.

Value

A list of lists. In the "outer" list, there are n.variables lists, one for each locus. In the "inner" list, for every individual there is a vector of two codes (typically characters, see alleleininit) for the two alleles in that locus.

Author(s)

Christian Hennig <christian.hennig@unibo.it> https://www.unibo.it/sitoweb/christian.hennig/en

See Also

alleleininit, build.charmatrix

Examples

data(tetragonula)
tnb <-
coord2dist(coordmatrix=tetragonula.coord[1:50,],cut=50,file.format="decimal2",neighbors=TRUE)
ta <- alleleinconvert(strmatrix=tetragonula[1:50,])
tai <- alleleininit(allelematrix=ta,neighborhood=tnb$nblist,distance="none")
str(unbuild.charmatrix(tai$charmatrix,50,13))

veronica Genetic AFLP data of Veronica plants

Description

0-1 data indicating whether dominant markers are present for 583 different AFLP bands ranging from 61 to 454 bp of 207 plant individuals of Veronica (Pentasepalae) from the Iberian Peninsula and Morocco (Martinez-Ortega et al., 2004).

Usage

data(veronica)
Format

Two objects are generated:

**veronica** 0-1 matrix with 207 individuals (rows) and 583 AFLP bands (columns).

**veronica.coord** a 207*2 matrix. Coordinates of locations of individuals in decimal format, i.e. the first number is latitude (negative values are South), with minutes and seconds converted to fractions. The second number is longitude (negative values are West).

Details

 Reads from example data files *MartinezOrtega04AFLP.dat*, *MartinezKoord.dat*.

Source


Examples

```r
data(veronica)
```

---

**waterdist**

*Overwater distances between islands in the Aegean sea*

Description

Distance matrix of overwater distances in km between 34 islands in the Aegean sea.

Usage

```r
data(waterdist)
```

Format

A symmetric 34*34 distance matrix.

Details

Reads from example data file *Waterdist.dat*, in which there is a 35th column and line with distances to Turkey mainland.

Source

Examples

data(waterdist)
Index

*Topic array
   con.comp, 29
   incmatrix, 41

*Topic cluster
   abundtest, 5
   alleledist, 12
   alleleinit, 13
   allelepairoccomponent, 16
   build.ext.nblist, 20
   con.comp, 29
   con.regmat, 30
   crmatrix, 32
   dcedist, 33
   distratio, 34
   geco, 35
   geo2neighbor, 37
   homogen.test, 37
   hprabclust, 39
   jaccard, 42
   kulczynski, 43
   lcomponent, 44
   lociplots, 45
   nn, 49
   NNClean, 50
   piecelin, 54
   pop.sim, 57
   prabclust, 60
   prabinit, 63
   prabtest, 66
   qkulczynski, 69
   specgroups, 85

*Topic datasets
   kykladspcreg, 44
   nb, 48
   siskiyou, 84
   tetragonula, 87
   veronica, 89
   waterdist, 90

*Topic htest
   comp.test, 28
   homogen.test, 37
   plotdistreg, 55
   pop.sim, 57
   regdistbetween, 73
   regdistbetweenone, 75
   regeqdist, 80

*Topic manip
   allelepzeroone, 9
   alleleverconvert, 10
   alleleinit, 13
   build.charmatrix, 19
   nastats, 47
   toprob, 88
   unbuild.charmatrix, 88

*Topic math
   coord2dist, 31

*Topic multivariate
   communitydist, 25
   NNClean, 50
   phipt, 52
   stressvals, 86

*Topic regression
   plotdistreg, 55
   regdist, 72
   regdistbetween, 73
   regdistbetweenone, 75
   regdistdiff, 78
   regdistdiffone, 79
   regeqdist, 80

*Topic spatial
   abundtest, 5
   alleleinit, 13
   autoconst, 17
   build.nblist, 21
   cluspop.nb, 22
   communities, 24
   communitydist, 25
   con.regmat, 30
INDEX

abundtest, 5, 58, 60, 84
allele2zeroone, 9, 46
alleleconvert, 10, 13–15
alleledist, 12, 15, 16, 19, 88
alleleinit, 3, 9–13, 13, 19, 20, 25, 26, 52, 60, 62, 65, 86, 89
alleleobject, 9, 86, 88
alleleobject (alleleinit), 13
allelepaircomp, 16
autoconst, 8, 17, 23, 69, 72, 84
autoreg (autoconst), 17
build.charmatrix, 19, 89
build.ext.nblist, 20
build.nblist, 21
cfchord (phipt), 52
chisq.test, 28
cluspop.nb, 22, 69, 72
cmdscale, 62
communities, 24, 25–27
communitydist, 24, 25, 52, 53
comp.test, 28
con.comp, 18, 29, 30
con.regmat, 30
coord2dist, 31
crmatrix, 32
cutree, 29, 40
dicedist, 33, 42, 43
diploidcomlist, 27
diploidcomlist (phipt), 52
dist, 32, 72, 78, 79
distratio, 8, 34, 58, 69
errorsarlm, 8, 58–60, 82
geco, 35, 43, 54, 65
g2neighor, 32, 37
hclust, 24, 29, 40
homogen.test, 8, 37, 58, 69
hprabclust, 39, 60, 62
incmatrix, 8, 41, 58, 69
isoMDS, 62, 86
jaccard, 34, 42, 43, 58, 65
jackknife, 74, 77, 81
kappa, 74, 76, 81
kulczynski, 34, 36, 42, 43, 58, 65, 70
kykladspecreg, 4, 44
lcomponent, 8, 44, 58, 69
lm, 59, 79, 74, 77, 81
lociplots, 45
mclustBIC, 60, 62
nastats, 47
nb, 44, 48
nb2listw, 8, 21, 59
nbtest, 14, 48, 64, 65
nn, 8, 49, 58, 69
NNclean, 50, 62
par, 55
phipt, 27, 52
piecewiselin, 54
plot, 56
plotdistreg, 55
pop.sim, 57, 69
prab, 14, 46, 58, 86
prab (prabinit), 63
prab.sarestimate, 7, 8, 58
prabclus-package, 3
prabclus, 3, 14, 28, 40, 45, 46, 60
prabinit, 3, 8, 15, 18, 23, 46, 49, 60, 62, 63, 69, 72, 84, 86
prabtest, 7, 8, 30, 35, 38, 41, 45, 50, 57, 58, 66, 72
print.alleleobject (alleleinit), 13
print.comprabclust (hprabclusl), 39
print.nnclen (NNclean), 50
print.prab (prabinit), 63
print.prabclus (prabclusl), 60
print.regdistbetween (regdistbetween), 73
print.regeqdist (regeqdist), 80
print.summary.prabtest (prabtest), 66
qkulczynski, 43, 65, 69
randpop.nb, 18, 23, 57, 58, 69, 70, 84
read.table, 10, 65
regdist, 72
regdistbetween, 56, 73, 79, 82
regdistbetweenone, 56, 75, 75, 77, 80, 82
regdistdiff, 78
regdistdiffone, 56, 79
regeqdist, 56, 75, 77, 80
regpop.sar, 8, 82
sammon, 62
shared.problist (phipt), 52
siskiyou, 4, 84
specgroups, 85
stressvals, 3, 61, 62, 86
summary.mclustBIC, 62
summary.prabtest, 8
summary.prabtest (prabtest), 66
tetragonula, 4, 87
toprab, 88
unbuild.charmatrix, 12, 13, 19, 88
veronica, 4, 89
waterdist, 44, 90