Package ‘prabclus’

January 17, 2019

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

Version 2.2-7

Date 2019-01-16

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

Depends R (&gt;= 2.10), MASS, mclust

Suggests spdep, maptools, foreign, mvtnorm

Description Distance-based parametric bootstrap tests for clustering with spatial neighborhood information. Some distance measures, Clustering of presence-absence, abundance and multilocus genetical data for species delimitation, nearest neighbor based noise detection. Try package?prabclus for an overview.

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

License GPL

URL http://www.homepages.ucl.ac.uk/~ucakche

NeedsCompilation no

Repository CRAN

Date/Publication 2019-01-17 13:40:07 UTC

R topics documented:

  prabclus-package .................................................. 3
  abundtest .......................................................... 4
  allele2zeroone ...................................................... 8
  alleleconvert ...................................................... 9
  alleledist .......................................................... 11
  alleleinit .......................................................... 12
  allelepaircomp .................................................... 15
  autoconst .......................................................... 15
  build.charmatrix .................................................. 17
<table>
<thead>
<tr>
<th>R topics documented:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>build.ext.nblist</td>
<td>18</td>
</tr>
<tr>
<td>build.nblist</td>
<td>19</td>
</tr>
<tr>
<td>cluspop.nb</td>
<td>20</td>
</tr>
<tr>
<td>comp.test</td>
<td>22</td>
</tr>
<tr>
<td>con.comp</td>
<td>23</td>
</tr>
<tr>
<td>con.regmat</td>
<td>24</td>
</tr>
<tr>
<td>coord2dist</td>
<td>25</td>
</tr>
<tr>
<td>crmatrix</td>
<td>27</td>
</tr>
<tr>
<td>dicedist</td>
<td>28</td>
</tr>
<tr>
<td>distratio</td>
<td>29</td>
</tr>
<tr>
<td>geco</td>
<td>30</td>
</tr>
<tr>
<td>geo2neighbor</td>
<td>31</td>
</tr>
<tr>
<td>homogen.test</td>
<td>32</td>
</tr>
<tr>
<td>hprabclust</td>
<td>33</td>
</tr>
<tr>
<td>incmatrix</td>
<td>35</td>
</tr>
<tr>
<td>jaccard</td>
<td>36</td>
</tr>
<tr>
<td>kulczynski</td>
<td>37</td>
</tr>
<tr>
<td>kykladspecreg</td>
<td>38</td>
</tr>
<tr>
<td>lcomponent</td>
<td>38</td>
</tr>
<tr>
<td>lociplots</td>
<td>39</td>
</tr>
<tr>
<td>nastats</td>
<td>41</td>
</tr>
<tr>
<td>nb</td>
<td>42</td>
</tr>
<tr>
<td>nbtest</td>
<td>42</td>
</tr>
<tr>
<td>nn</td>
<td>43</td>
</tr>
<tr>
<td>NNelean</td>
<td>44</td>
</tr>
<tr>
<td>piecewiselin</td>
<td>46</td>
</tr>
<tr>
<td>pop.sim</td>
<td>47</td>
</tr>
<tr>
<td>prab.sarestimate</td>
<td>48</td>
</tr>
<tr>
<td>prabclust</td>
<td>50</td>
</tr>
<tr>
<td>prabinit</td>
<td>53</td>
</tr>
<tr>
<td>prabtest</td>
<td>56</td>
</tr>
<tr>
<td>qkulczynski</td>
<td>59</td>
</tr>
<tr>
<td>randpop.nb</td>
<td>60</td>
</tr>
<tr>
<td>regpop.sar</td>
<td>62</td>
</tr>
<tr>
<td>siskiyou</td>
<td>63</td>
</tr>
<tr>
<td>specgroups</td>
<td>64</td>
</tr>
<tr>
<td>stressvals</td>
<td>65</td>
</tr>
<tr>
<td>tetragonula</td>
<td>66</td>
</tr>
<tr>
<td>toprab</td>
<td>67</td>
</tr>
<tr>
<td>unbuild.charmatrix</td>
<td>68</td>
</tr>
<tr>
<td>veronica</td>
<td>69</td>
</tr>
<tr>
<td>waterdist</td>
<td>70</td>
</tr>
</tbody>
</table>

Index 71
**prabclus-package**  

**prabclus package overview**

**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**

- **prabclust** 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.
- **hprabclust** An unpublished alternative to **prabclust** using hierarchical clustering methods.
- **lociplots** Visualisation of clusters of genetic markers vs. clusters of species.
- **NNean** 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.
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 <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche/

References


abundtest  Parametric bootstrap test for clustering in abundance matrices
**abundtest**

**Description**

Parametric bootstrap test of a null model of i.i.d., but spatially autocorrelated species against clustering of the species’ population patterns. Note that most relevant functionality of `prabtest` (except of the use of the geco distance) is also included in `abundtest`, so that `abundtest` can also be used on binary presence-absence data. 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/abundmatrixfile", neighborhood="path/neighborhoodfile")
test <- abundtest(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
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,
sarestimate=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. "isovertex": 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 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` prabinit-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 nonzeroes of `prabobj$prab`.

`groupvector` integer vector. For every species, a number indicating the species' group membership. 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 (usually 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 "isovertex", 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 `errorssarlm`) 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` (group-wise `p.above` of "means" test), `pmb` (group-wise `p.below` of "means" test).

**Author(s)**

Christian Hennig <chrish@stats.ucl.ac.uk> [http://www.homepages.ucl.ac.uk/~ucakche](http://www.homepages.ucl.ac.uk/~ucakche)

**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.sarestimte (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.

Examples

# Note: NOT RUN.
# This needs package spdep and a bunch of packages that are
# called by spdep!
# data(siskiyou)
# set.seed(1234)
# x <- prabinit(prabmatrix=siskiyou, neighborhood=siskiyou.nb,
#         distance="logkulczynski")
# a1 <- abundtest(x, times=5, p.nb=0.0465)
# a2 <- abundtest(x, times=5, p.nb=0.0465, teststat="groups",
#         groupvector=siskiyou.groups)
# These settings are chosen to make the example execution
# faster; usually you will use abundtest(x).
# summary(a1)
# summary(a2)

allele2zeroone

Converts alleleobject into binary matrix

Description

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

Usage

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.


**alleleconvert**

**Author(s)**

Christian Hennig <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

**Examples**

```r
data(tetragonula)
ta <- alleleconvert(strmatrix=tetragonula[21:50,])
tai <- alleleinit(allelematrix=ta)
alglele2zeroone(tai)
```

---

**alleleconvert**

**Format conversion for codominant marker data**

**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), "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)
```

**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" or "structure", 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.

alleletters 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 fistcolname==TRUE).

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.

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 <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

References


See Also

alleleinit
**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 <chrish@stats.ucl.ac.uk> [http://www.homepages.ucl.ac.uk/~ucakche](http://www.homepages.ucl.ac.uk/~ucakche)

**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("I22", "I44", "I22", "I22", "I44", "I44"),c("0", "0", "21", "33", "35", "44"))
alleleconvert(strmatrix=strucmatrix, format.in="structure",
format.out="prabclus", orig.nchar="0", firstcolname=TRUE)
alleleconvert(strmatrix=strucmatrix, format.in="structure",
format.out="structurama", orig.nchar="0", new.nchar="-9", firstcolname=TRUE)
```
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))
```

---

### alleleinit

**Diploid loci matrix initialization**

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)
```

#### 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.
n.species identical to n.individuals used for compatibility with prabclust.
alevels character vector with all used allele codes not including missing values.
leveldist 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.
aprob 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 <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

See Also

alleleconvert, alleledist, prabininit.

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 <- alleleininit(allelematrix=ta,neighborhood=tnb$nblist)
print(tai)
**allelepaircomp**

**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 <chrish@stats.ucl.ac.uk> [http://www.homepages.ucl.ac.uk/~ucakche](http://www.homepages.ucl.ac.uk/~ucakche)

**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 richneses 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 disjunction 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 <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

References


See Also

randpop.nb, prabinit, con.comp

Examples

```r
options(digits=4)
data(kykladspecreg)
data(nb)
set.seed(1234)
x <- prabinit(prabmatrix=kykladspecreg, neighborhood=nb)
ax <- autoconst(x, nperp=2, step1=0.3, twostep=FALSE)
```

build.charmatrix | Internal: create character matrix out of allele list

Description

For use in alleleinits. 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 alleleinit) 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 <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

See Also

alleleinit, 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 alleleinit. 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

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

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

**Value**
- list with $2 \times \text{n.individuals}$ vectors of integers as described above.

**Author(s)**
- Christian Hennig <chrish@stats.ucl.ac.uk> [http://www.homepages.ucl.ac.uk/~ucakche](http://www.homepages.ucl.ac.uk/~ucakche)

**See Also**
- **alleleinit**

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

**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**
```r
build.nblist(prabobj,prab@1=NULL,style="C")
```

**Arguments**
- **prabobj**: object of class prab.
- **prab@1**: 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), prab@1 is computed in order to indicate the nonzeroes 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 <chrish@stats.ucl.ac.uk> [http://www.homepages.ucl.ac.uk/~ucakche](http://www.homepages.ucl.ac.uk/~ucakche)

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.

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.

pdfnb
  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.
comp.test

Author(s)
Christian Hennig <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

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))

comp.test  Compare species clustering and species groups

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 <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche
con.comp

References

See Also
chisq.test, prabclus.

Examples

```r
set.seed(1234)
g1 <- c(rep(1,34),rep(2,12),rep(3,15))
g2 <- sample(3,61,replace=TRUE)
comp.test(g1,g2)
```

---

### con.comp

**Connectivity components of an undirected graph**

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

**Usage**

```r
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 vertex.

**Author(s)**
Christian Hennig <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

**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).
```

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`.
coord2dist

Author(s)

Christian Hennig <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

See Also

corn.comp, prabtest

Examples

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

coord2dist(cp, nb)

coord2dist

Geographical coordinates to distances

Description

Computes geographical distances from geographical coordinates

Usage

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 "decimal14". 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 <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

References

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

See Also

g2o2neighbor

Examples

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

Description

Produces 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)
```

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 <chrish@stats.ucl.ac.uk> [http://www.homepages.ucl.ac.uk/~ucakche](http://www.homepages.ucl.ac.uk/~ucakche)

Examples

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

crmatrix(x, xc)
crmatrix(x, xc, percentages=TRUE)
```
dice.dist

**Description**

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

**Usage**

`dice.dist(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*number of regions where both species are present)/(2*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 <chrish@stats.ucl.ac.uk> [http://www.homepages.ucl.ac.uk/~ucakche](http://www.homepages.ucl.ac.uk/~ucakche)

**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 prop smallest and largest distances of a distance matrix.

Usage
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.
- lowmean: mean of prop smallest distances.
- himean: mean of prop smallest distances.
- prop: see above.

Author(s)
Christian Hennig <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

References

See Also
prabtest

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

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 <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche
References


See Also

kulczynski

Examples

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

g2n <- geo2neighbor(waterdist)

desc2n <- desc2neighbors(g2n)

geoRneighbor

Neighborhood list from geographical distance

Description

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

Usage

`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 <chrish@stats.ucl.ac.uk> [http://www.homepages.ucl.ac.uk/~ucakche](http://www.homepages.ucl.ac.uk/~ucakche)

Examples

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

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 distri-
bution as given by Erdos and Renyi (1960). If testdist="ling", the test dis-
tribution 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 <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche
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 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
hprabclust(prabobj, cutdist=0.4, cutout=1,
method="average", nnout=2, mdsplot=TRUE, mdsmethod="classical")

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

**prabobj**
object of class `prab` as generated by `prabinit`. 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 `mdsplot!="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 `mdsplot` is ignored if `mdsplot=FALSE`.

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

... necessary for print method.

Value

`hprabclust` generates an object of class `comprabclust`. 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 <chrish@stats.ucl.ac.uk> [http://www.homepages.ucl.ac.uk/~ucakche](http://www.homepages.ucl.ac.uk/~ucakche)

See Also

- `hclust`, `cutree`, `prabclust`
**incmatrix**

**Examples**
```r
data(kykladspecreg)
data(nb)
data(waterdist)
x <- prabinit(prabmatrix=kykladspecreg, neighborhood=nb,
    geodist=waterdist, distance="geco")
hprabclust(x,mdsplot=FALSE)
```

---

**Description**

Computes species*species nestedness matrix and number of nestings (inclusions) from regions*species presence-absence matrix.

**Usage**

```r
incmatrix(regmat)
```

**Arguments**

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

**Value**

A list with components

- `m` 0-1-matrix. \( m_{ij} = 1 \) 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 <chrish@stats.ucl.ac.uk> [http://www.homepages.ucl.ac.uk/~ucakche](http://www.homepages.ucl.ac.uk/~ucakche)

**References**


**See Also**

`prabtest`
jaccard

Examples

```r
data(kykladspecreg)
incmatrix(t(kykladspecreg))$inc

jaccard

Jaccard distance matrix

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-(number of regions where both species are present)/
(number of regions where at least one species is present). As a similarity coefficient, this is S22 in Shi (1993).
Thank you to Laurent Buffat for improving this function!

Value
A symmetrical matrix of Jaccard distances.

Author(s)
Christian Hennig <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

References

See Also
kulczynski, dicedist

Examples

options(digits=4)
data(kykladspecreg)
jaccard(t(kykladspecreg))
Kulczynski distance matrix

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 <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

References

Palaeogeography, Palaeoclimatology, Palaeoecology 105, 199-234.

See Also

jaccard, geco,qkulczynski , dicedist

Examples

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

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

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

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 <chrish@stats.ucl.ac.uk> [http://www.homepages.ucl.ac.uk/~ucakche](http://www.homepages.ucl.ac.uk/~ucakche)

References


See Also

- prabtest

Examples

```r
data(kykladspecreg)
j <- jaccard(t(kykladspecreg))
lociplots(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

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

indclust  prabclust-object. Clustering of individuals.
locclust  vector of integers. Clustering of markers/loci.
locprab   prab-object in which the markers are what the help page of prabinit 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 allele2zeroone before prabinits.)
lcluster  integer. Number of cluster in locclust for which plot and statistics are produced.
symbols  vector of plot symbols. If NULL, indclust$symbols is used.
brightest.grey numeric between 0 and 1. Brightest grey value used in plot for individuals with smallest marker percentage, see details.
darkest.grey numeric between 0 and 1. Darkest grey value used in plot for individuals with highest marker percentage, see details.
mdsdim   vector of two integers. The two MDS variables taken from 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. lcluster. In the plot, the grey value visualises the marker percentage.

Value

list with components

locfreq   vector of individual marker percentages.
locfreqmin vector of minimum individual marker percentages for each cluster in indclust-clustering (the first value refers to the "noise component", if present).
locfreqmax vector of maximum individual marker percentages for each cluster in indclust-clustering (the first value refers to the "noise component", if present).
locfreqmean vector of average individual marker percentages for each cluster in indclust-clustering (the first value refers to the "noise component", if present).

Author(s)

Christian Hennig <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

See Also

prabclust
nastats

Examples

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
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 <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

Examples
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)
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(nblst, n.regions=length(nblst))
```
**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).

nregions  
Number of regions.

**Value**

invisible{TRUE}.

**Author(s)**

Christian Hennig <chrish@stats.ucl.ac.uk> [http://www.homepages.ucl.ac.uk/~ucakche](http://www.homepages.ucl.ac.uk/~ucakche)

**See Also**

prabinit.

**Examples**

```r
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 nth nearest neighbor.

**Usage**

```r
nn(distmat, ne = 1)
```

**Arguments**

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

ne  
integer.

**Value**

numerical.

**Author(s)**

Christian Hennig <chrish@stats.ucl.ac.uk> [http://www.homepages.ucl.ac.uk/~ucakche](http://www.homepages.ucl.ac.uk/~ucakche)
References


See Also

`prabtest`

Examples

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

**nnclean**

*Nearest neighbor based clutter/noise detection*

Description

Detects if data points are noise or part of a cluster, based on a Poisson process model.

Usage

```r
nnclean(data, k, distances = NULL, edge.correct = FALSE, wrap = 0.1, convergence = 0.001, plot=FALSE, quiet=TRUE)
```

## S3 method for class 'nnclean'

```r
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.
- `edge.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 `edge.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.
- `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 `nnclean`.
- `...`: necessary for print methods.
NNclean

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

NNclean returns a list of class nnclean 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.
- kthNN: 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 <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche
original Splus package by S. Byers and A. E. Raftery.

References


Examples

```r
library(mclust)
data(chevron)
nc <- NNclean(chevron[,2:3],15,plot=TRUE)
plot(chevron[,2:3],col=1+nc$z)
```
**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 <chrish@stats.ucl.ac.uk> [http://www.homepages.ucl.ac.uk/~ucakche](http://www.homepages.ucl.ac.uk/~ucakche)

**See Also**

- `geco`

**Examples**

```r
options(digits=4)
data(waterdist)
piecewiselin(waterdist)
```
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 richmesses 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 <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

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)
```

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.
Usage

prab.sarestimate(abmat, prab01=NULL, sarmethod="eigen",
weightstyle="C",
quiet=TRUE, sar=TRUE,
add.lmobject=TRUE)

Arguments

abmat  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 nonzeroes of prabobj$prab.
sarmethod  this is passed on as parameter method to errorsarlm and documented there. We don’t have experience with any other choice than "eigen".
weightstyle  can take values "W", "B", "C", "U", and "S" though tests suggest that "C" should be chosen. See nb2listw.
quiet  this is passed on as parameter quiet to errorsarlm and documented there.
sar  logical. If TRUE, a simultaneous autoregression model is fitted by calling errorsarlm. If FALSE, a two-way additive unreplicated linear model for log-abundances on factors species and region is computed by lm, ignoring the spatial arrangement of the regions.
add.lmobject  logical. If TRUE, the whole output object of errorsarlm (or lm) is given out.

Value

A list with the following components:
sar  see above.
intercept  numeric. Estimator of the intercept.
sigma  numeric. Estimator of error standard deviation.
regeffects  numeric vector. Estimator for region effects.
speceffects  numeric vector. Estimator for species effects.
lambda  numeric. Governs the degree of spatial autocorrelation. See errorsarlm.
size  integer. Length of neighborhood list generated by nb2listw used by errorsarlm.
nbweight  numeric. Average weight of neighbors.
lmobject  if add.lmobject=TRUE, output object of either lm or errorsarlm.

Author(s)

Christian Hennig <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

See Also

errorsarlm, abundtest
prabclust

### Examples

```r
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)
```

### 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

```r
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 `prabininit` and `alleleinit` help pages. You may also consider the example datasets kykladspcreg.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

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

```r
## 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`.)

- **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.
mdsdim integer. Dimension of the MDS points. For mdsmethod="kruskal", stressvals can be used to see how the stress depends on mdsdim in order to choose mdsdim to get a small stress (smaller than 5%, say).

nnk integer. Number of nearest neighbors to determine the initial noise estimation by NNClean. nnk=0 fits the model without a noise component.

nclus vector of integers. Numbers of clusters to perform the mixture estimation.

modelid string. Model name for mclustBIC (see the corresponding help page; all models or combinations of models mentioned there are possible). modelid="all" compares all possible models. Additionally, "noVVV" is possible, which fits all methods except "VVV".

permutations integer. It has been found occasionally that depending on the order of observations the algorithms isoMDS and mclustBIC converge to different solutions. This is because these methods require an ordering of the distances, which, if equal distance values are involved, may depend on the order. prabclust uses a standard ordering which should give a reproducible solution in these cases as well. However, if permutations>0, which gives a number of random permutations of the observations, the algorithm is carried out for every permutation and the best solution (in terms of the BIC, based on the lowest stress MDS configuration) is given out (for many datasets this won’t change anything except increasing the computing time).

x object of class prabclust. Output of prabclust.

bic logical. If TRUE, information about the BIC criterion to choose the model is displayed.

Details

Note that if mdsmethod="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

print.prabclust does not produce output. prabclust generates an object of class prabclust. This is a list with components

clustering vector of integers indicating the cluster memberships of the species. Noise can be recognized by output component symbols.

clustsummary output object of summary.mclustBIC. A list giving the optimal (according to BIC) 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 output object of mclustBIC. 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 <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

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")
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

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)

### S3 method for class 'prab'

print(x, ...)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>file</td>
<td>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.</td>
</tr>
<tr>
<td>prabmatrix</td>
<td>matrix with non-negative entries. Either file or prabmatrix should be NA.</td>
</tr>
</tbody>
</table>
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.

nbbetweenregions

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

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.
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.
n.betweenregions   see above.

Author(s)

Christian Hennig <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

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(kykkladspecreg)
data(nb)
prabinit(prabmatrix=kykkladspecreg, 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$distmat)/4),
  isovertex = ncol(prabobject$distmat), n = 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,...)
```

## 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. "isovertex": 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 tuningh 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 richnesses 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 "1component", "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 "isovertex". 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$teststat="groups", components rrangeg (matrix of group-wise ranges of test statistic value), rmean (vector of group-wise means of test statistic value), rrangem (range over simulations of overall mean of within-group dissimilarities), rmeanm (mean over simulations of overall mean of within-group dissimilarities) are added to the list object$groupinfo, and this is given out.

Author(s)

Christian Hennig <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

References


See Also

prabinit 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 <- prabinit(prabmatrix=kykladspecreg, neighborhood=nb)
# If you want to use your own ASCII data files, use
# x <- prabinit(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 <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

References

See Also
kulczynski

Examples
options(digits=4)
data(kykladspecreg)
qkulczynski(t(kykladspecreg))
n.species integer. Number of species.
n.regions integer. Number of regions.
vector.species 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.
species.fixed logical. See 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.
count logical. If TRUE, the number of the currently generated species is printed.
pdfnb 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 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.

Value

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

Author(s)

Christian Hennig <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

References


See Also

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

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 sarestimate). Spatial autocorrelation of a species’ presences is governed by the parameter p.nb, sarestimate and a list of neighbors for each region.

Usage

regpop.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 nonzeros 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 <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

**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
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


Examples

data(siskiyou)

---

**specgroups**

*Average within-group distances for given groups*

Description

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

Usage

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).
stressvals

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 <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

Examples

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)
ngsv <- numeric(0)
for (i in 1:ng) ngs[i] <- sum(groupvector==lg[i])
groupinfo <- list(lg=lg,ng=ng,ngsv=ngsv)
especgroups(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 prabininit.
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 <chrish@stats.ucl.ac.uk> [http://www.homepages.ucl.ac.uk/~ucakche](http://www.homepages.ucl.ac.uk/~ucakche)

Examples

```r
options(digits=4)
set.seed(112233)
taiselect <- sample(236,40)
# Use data subset to make execution faster.

# Generate 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.

# Load the 'tetragonula' data

data(tetragonula)

tb <- coord2dist(coordmatrix=tetragonula$coord[taiselect,],
cut=50,file.format="decimal",neighbors=TRUE)
ta <- alleleconvert(strmatrix=tetragonula[taiselect,])
tai <- alleleinit(allelematrix=ta,neighborhood=tnb$neighbors)
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**

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
data(tetragonula)

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

**Usage**

toprab(prabobj)

**Arguments**

prabobj  object of class prab.

**Value**

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

**Author(s)**

Christian Hennig <chrish@stats.ucl.ac.uk> http://www.homepages.ucl.ac.uk/~ucakche

**Examples**
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 `alleleinit`) for the two alleles in that locus.

**Author(s)**

Christian Hennig <chrish@stats.ucl.ac.uk> [http://www.homepages.ucl.ac.uk/~ucakche](http://www.homepages.ucl.ac.uk/~ucakche)

**See Also**

- `alleleinit`, `build.charmatrix`

**Examples**

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

data(veronica)
Description

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

Usage

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, 23
  incmatrix, 35

*Topic cluster
  abundtest, 4
  alleledist, 11
  alleleinit, 12
  allelepaircomp, 15
  build.ext.nblist, 18
  con.comp, 23
  con.regmat, 24
  crmatrix, 27
  dicedist, 28
  distratio, 29
  geco, 30
  geo2neighbor, 31
  homogen.test, 32
  hprabclust, 33
  jaccard, 36
  kulczynski, 37
  lcomponent, 38
  lociplots, 39
  nn, 43
  NNclean, 44
  piecewiselin, 46
  pop.sim, 47
  prabclust, 50
  prabinit, 53
  prabtest, 56
  qkulczynski, 59
  specgroups, 64

*Topic datasets
  kykladspecreg, 38
  nb, 42
  siskiyou, 63
  tetragonula, 66
  veronica, 69
  waterdist, 70

*Topic htest
  comp.test, 22
  homogen.test, 32
  pop.sim, 47

*Topic manip
  allele2zeroone, 8
  alleleconvert, 9
  alleleinit, 12
  build.charmatrix, 17
  nastats, 41
  toprab, 67
  unbuidl.charmatrix, 68

*Topic math
  coord2dist, 25

*Topic multivariate
  NNclean, 44
  stressvals, 65

*Topic spatial
  abundtest, 4
  alleleinit, 12
  autoconst, 15
  build.nblist, 19
  cluspop.nb, 20
  con.regmat, 24
  crmatrix, 27
  dicedist, 28
  geco, 30
  geo2neighbor, 31
  hprabclust, 33
  incmatrix, 35
  jaccard, 36
  kulczynski, 37
  nbtest, 42
  piecewiselin, 46
  prab.sarestimate, 48
  prabclus, 50
  prabinit, 53
  prabtest, 56
  qkulczynski, 59
  randpop.nb, 60
regpop.sar, 62
abundtest, 4, 48, 49, 63
allele2zeroone, 8, 40
allelconvert, 9, 12–14
alleledist, 11, 14, 15, 17, 68
alleleinit, 3, 8–12, 12, 17–19, 50, 52, 55, 65, 68
alleleobject, 8, 65, 68
alleleobject(alleleinit), 12
allelepaircomp, 15
autoconst, 8, 15, 22, 59, 61, 63
autoreg(autoconst), 15
build.charmatrix, 17, 68
build.ext.nblist, 18
build.nblist, 19
chisq.test, 23
cluspop.nb, 20, 59, 61
cmdscale, 52
comp.test, 22
con.comp, 17, 23, 25
con.regmat, 24
coord2dist, 25
crmatrix, 27
cutree, 24, 34
dicedist, 28, 36, 37
dist, 26
distratio, 8, 29, 48, 59
errorsarlm, 7, 8, 48, 49, 62
geco, 30, 37, 46, 55
geo2neighbor, 26, 31
hclust, 24, 34
homogen.test, 8, 32, 48, 59
hprabclust, 33, 50, 52
incmatrix, 8, 35, 48, 59
isoMDS, 52, 65, 66
jaccard, 28, 36, 37, 48, 55
kulczynski, 28, 31, 36, 37, 48, 55, 60
kykladspecreg, 4, 38
lcomponent, 8, 38, 48, 59
lm, 49
lociplots, 39
mclustBIC, 50, 52
nastats, 41
nb, 38, 42
nb2listw, 7, 19, 20, 49
nbtest, 13, 42, 54, 55
nn, 8, 43, 48, 59
NNclean, 44, 52
piecewising, 46
pop.sim, 47, 59
prab, 13, 40, 48, 65
prab(prabinit), 53
prab.sarestimate, 6, 8, 48
prabclus-package, 3
prabclus, 3, 13, 23, 34, 39, 40, 50
prabinit, 3, 8, 14, 17, 22, 40, 43, 50, 52, 53, 59, 61, 63, 65
prabtest, 6, 8, 25, 29, 33, 35, 39, 44, 47, 48, 56, 61
print.alleleobject(alleleinit), 12
print.comprabclust(hprabclust), 33
print.nnclen (NNclean), 44
print.prab(prabinit), 53
print.prabclust(prabclus), 50
print.summary.prabtest(prabtest), 56
qkulczynski, 37, 55, 59
randpop.nb, 17, 22, 47, 48, 59, 60, 63
read.table, 55
regpop.sar, 8, 62
sammon, 52
siskiyou, 4, 63
specgroups, 64
stressvals, 3, 51, 52, 65
summary.mclustBIC, 52
summary.prabtest, 8
summary.prabtest(prabtest), 56
tetragonula, 4, 66
toprab, 67
unbuild.charmatrix, 11, 12, 18, 68
veronica, 4, 69
waterdist, 38, 70