Package 'adespatial'

September 27, 2018

Title Multivariate Multiscale Spatial Analysis

Version 0.3-2

Date 2018-09-26

Author Stéphane Dray, David Bauman, Guillaume Blanchet, Daniel Borcard, Sylvie Clappe, Guillaume Guenard,
Thibaut Jombart, Guillaume Larocque, Pierre Legendre, Naima Madi, Helene H Wagner

Description Tools for the multiscale spatial analysis of multivariate data.
Several methods are based on the use of a spatial weighting matrix and its
eigenvector decomposition (Moran's Eigenvectors Maps, MEM).

Maintainer Stéphane Dray <stephane.dray@univ-lyon1.fr>

Imports ade4 (>= 1.7-13), adegraphics, adephylo, sp, spdep, lattice,
methods, grDevices, graphics, MASS, stats, utils, shiny, vegan

Suggests knitr, RANN, rgeos, maptools, ape, rmarkdown

License GPL (>= 2)

Encoding UTF-8

NeedsCompilation yes

RoxygenNote 6.1.0

VignetteBuilder knitr, rmarkdown, RANN, rgeos, maptools

Repository CRAN

Date/Publication 2018-09-27 11:10:03 UTC

R topics documented:

`aem` *Construct asymmetric eigenvector maps (AEM)*

Description

This function constructs eigenvectors of a site-by-link matrix. Weights can be applied to the links.

Usage

```
aem(aem.build.binary, binary.mat, weight, rm.link0 = FALSE,
  print.binary.mat = FALSE)
```

Arguments

`aem.build.binary`	Object created by function `aem.build.binary`.
`binary.mat`	Site (n rows) by link (k columns) matrix. The 1s in the matrix represents the presence of a link influencing a site, directly or indirectly, otherwise the values are 0s.
`weight`	Vector of weights of length k, to be applied to the links.
`rm.link0`	Logical (`TRUE`, `FALSE`) determining if the links directly connecting a site to the origin (site 0) should be removed. Default value: `FALSE`. This parameter is only used when an object of class `aem.build.binary` is provided to the function.
`print.binary.mat`	Logical (`TRUE`, `FALSE`) determining if the site-by-link matrix used in the analysis should be printed.

Details

If only an object of class `aem.build.binary` is given to this function, The argument `binary.mat` is not considered. `binary.mat` is only considered when the argument `aem.build.binary` is missing.

If weights are applied to the links, the length of vector `weight` has to take into account wether the links connecting real sites to the origin (the fictitious site 0) have been kept or removed.

Value

`value`	A vector of singular values associated with the AEM.
`vectors`	A matrix of eigenvector. Each column is an AEM eigenfunction (or variable).
`mod.binary.mat`	A site-by-link matrix modified through the function.

Note

It sometimes happens that AEM eigenfunctions have equal singular values. In that case, different sets of AEM eigenfunctions may be produced on different plateforms.

Eigenvectors associated to an eigenvalue that is smaller than 10^{-12} are considered negligeable. They have been removed from the created AEM eigenfunctions.

Author(s)

F. Guillaume Blanchet

References

Blanchet F.G., P. Legendre and Borcard D. (2008) Modelling directional spatial processes in ecological data. *Ecological Modelling*, 215, 325-336.

See Also

aem.build.binary, svd

Examples

```
### Construction of object of class nb (spdep)
if(require("spdep", quietly = TRUE)){
nb <- cell2nb(5,5,"queen")

### Create fictitious geographical coordinates
xy <- cbind(1:25,expand.grid(1:5,1:5))

### Build binary site-by-link matrix
bin.mat <- aem.build.binary(nb,xy)

### Construct AEM eigenfunctions from an object of class aem.build.binary
res <- aem(aem.build.binary=bin.mat,rm.link0=FALSE)
res$values

### Illustrate 4 AEM eigenfunctions using bubble plots
opal <- palette()
palette(c("black","white"))
par(mfrow=c(2,2))
symbols(x=xy[,2:3], circles=abs(res$vectors[,1]), inches=FALSE, asp=1,
 fg=ifelse(sign(-res$vectors[,1])+1>0,1,0),
 bg=ifelse(sign(res$vectors[,1])+1>0,1,0), xlab="x", ylab="y")
title("AEM 1")
symbols(x=xy[,2:3], circles=abs(res$vectors[,2]), inches=FALSE,
asp=1, fg=ifelse(sign(-res$vectors[,2])+1>0,1,0),
 bg=ifelse(sign(res$vectors[,2])+1>0,1,0), xlab="x", ylab="y")
title("AEM 2")
symbols(x=xy[,2:3], circles=abs(res$vectors[,3]), inches=FALSE,
asp=1, fg=ifelse(sign(-res$vectors[,3])+1>0,1,0),
bg=ifelse(sign(res$vectors[,3])+1>0,1,0), xlab="x", ylab="y")
title("AEM 3")
symbols(x=xy[,2:3], circles=abs(res$vectors[,4]), inches=FALSE, asp=1,
 fg=ifelse(sign(-res$vectors[,4])+1>0,1,0),
 bg=ifelse(sign(res$vectors[,4])+1>0,1,0), xlab="x", ylab="y")
title("AEM 4")

### Construct AEM eigenfunctions using only a site-by-link matrix
res2 <- aem(binary.mat=bin.mat[[1]])
res2$values

### Illustrate 4 AEM eigenfunctions using bubble plots
par(mfrow=c(2,2))
symbols(x=xy[,2:3], circles=abs(res2$vectors[,1]), inches=FALSE,
asp=1, fg=ifelse(sign(-res2$vectors[,1])+1>0,1,0),
bg=ifelse(sign(res2$vectors[,1])+1>0,1,0), xlab="x", ylab="y")
title("AEM 1")
symbols(x=xy[,2:3], circles=abs(res2$vectors[,2]), inches=FALSE,
asp=1, fg=ifelse(sign(-res2$vectors[,2])+1>0,1,0),
```

```
bg=ifelse(sign(res2$vectors[,2])+1>0,1,0), xlab="x", ylab="y")
title("AEM 2")
symbols(x=xy[,2:3], circles=abs(res2$vectors[,3]), inches=FALSE,
asp=1, fg=ifelse(sign(-res2$vectors[,3])+1>0,1,0),
bg=ifelse(sign(res2$vectors[,3])+1>0,1,0), xlab="x", ylab="y")
title("AEM 3")
symbols(x=xy[,2:3], circles=abs(res2$vectors[,4]), inches=FALSE,asp=1,
 fg=ifelse(sign(-res2$vectors[,4])+1>0,1,0),
 bg=ifelse(sign(res2$vectors[,4])+1>0,1,0), xlab="x", ylab="y")
title("AEM 4")

palette(opal)

### Construct AEM eigenfunctions with a function of the distance
### as weights to put on the links

### Construction of object of class nb (spdep)
nb<-cell2nb(5,5,"queen")

### Create fictitious geographical coordinates
xy <- cbind(1:25,expand.grid(1:5,1:5))

### Build binary site-by-link matrix
bin.mat <- aem.build.binary(nb,xy)

### Construct a matrix of distances
long.lien.mat<-as.matrix(dist(xy))

### Extract the edges, remove the ones directly linked to site 0
lien.b<-bin.mat$edges[-1:-5,]

### Construct a vector giving the length of each edge
long.lien<-vector(length=nrow(lien.b))

for(i in 1:nrow(lien.b)){
long.lien[i]<-long.lien.mat[lien.b[i,1],lien.b[i,2]]
}

### Construct a vector of weights based on distance
weight.vec<-1-(long.lien/max(long.lien))^2

### Construct AEM eigenfunctions from an object of class aem.build.binary
res <- aem(aem.build.binary=bin.mat,weight=weight.vec,rm.link0=TRUE)
res
}
```

`aem.build.binary` *Construct a site-by-edge binary matrix*

Description

This function construct a site-by-edge binary matrix. It uses a set of sites coordinates and a connexion diagram (object of class nb from the spdep package). The 1s in the matrix represents the presence of a link influencing a site, directly or indirectly, otherwise the values are 0s. Graphically, the function is implemented such that the directional process is considered to be going from the bottom to the top of the screen in the graphical output of R. As such, the origin is underneath the set of points representing the sites. Prepare the table of site coordinates accordingly.

Usage

```
aem.build.binary(nb.object = NULL, coords, link = NULL,
  unit.angle = "degrees", rot.angle = 0, rm.same.y = TRUE,
  plot.connexions = TRUE)
```

Arguments

nb.object	Object of class nb from library spdep.
coords	A three columns matrix or data frame. Colums 1: identifiers of the points (needs to be numeric). Columns 2 and 3: the X and Y coordinates of the points.
link	A two columns matrix. Each row define an edge. Column 1: The site from which the edge starts. Column 2: the site to which the edge ends. All values in link need to be integers.
unit.angle	Character. The measurement units in which the angle is defined: either "degrees" (default) or "radians".
rot.angle	Numeric. Angle of the vector describing the process influencing the sites. This argument generate a rotation of the site coordinates. The set of coordinates is rotated counterclockwise. Negative values will produce a clockwise rotation.
rm.same.y	Logical (TRUE, FALSE). Determines if the links perpendicular to the gradient should be removed. Default value: TRUE. If these links have already been removed this argument put to TRUE will make the function crash. See detail for more information.
plot.connexions	Logical (TRUE, FALSE). Determines if the sites and the associated connexion diagram should be plotted after rotation of the coordinates by gradient.angle.

Details

The lowest site in the gradient is the one that will connect to the fictitious site 0 to consider direction. Note that if there are multiple lowest sites, they will all be connected to the fictitious site 0 to consider direction.

The site-by-edge matrix created by this function and the list of edges include the links associated to a fictitious site upstream of all other, see Blanchet et al. (In press), for details. The decision regarding wether the origin and the edges associated with it should be kept or removed is left to the user. Removal of this site and of its associated edges can be done manually after the construction of the site-by-edge matrix and of the list edges. It can also be done when running the function aem.

If the connexion diagram was modified so that the links connecting sites that are exactly perpendicular to the gradient have been removed or if there is no sites exactly at the same level in the gradient, defining `rm.same.y` to `TRUE` will generate an error.

If all the sites have the same y coordinates after rotation, e.g. a horizontal transect perpendicular to the defined spatial asymmetry, this analysis should not be used.

The argument `plot.connexions` will plot the sites (`coords`) in black, after rotation, if any, and the connexion diagram (`nb.object`), in red. The site labels are also plotted on the graph. To show the direction of the spatial asymmetry considered by the function, a fictive site (in blue) was added upstream. This fictive site is linked (blue edges) to the site(s) that are the most upstream ones. Since this graph is generic, it might sometimes look odd, however, the information given will remain the accurate.

Value

`se.mat`	A binary (n x k) matrix of site (n rows) by link edges (k columns).
`edges`	A matrix describing the link edges. It has 2 columns (from, to) and as many rows as there are edges. The edges linked to the fictitious site of origin are found at the beginning of the list.

Author(s)

F. Guillaume Blanchet

References

Blanchet F.G., P. Legendre and Borcard D. (2008) Modelling directional spatial processes in ecological data. *Ecological Modelling*, 215, 325-336.

See Also

`aem`

Examples

```
### Create an object of class nb (spdep)
if(require("spdep", quietly = TRUE)){
nb<-cell2nb(5,5,"queen")

### Create fictitious geographical coordinates
xy <- cbind(1:25,expand.grid(1:5,1:5))

### Build a binary site-by-link matrix; remove the site which have identical Y coordinate
### (by default argument: rm.same.y = TRUE)
bin.mat <- aem.build.binary(nb,xy)
str(bin.mat)

### Build a binary site-by-link matrix using the argument link: remove the site which
### have identical Y coordinate (by default argument: rm.same.y = TRUE)
edges<-expand.grid(1,2:25)
```

```
bin.mat <- aem.build.binary(coords=xy,link=edges)
str(bin.mat)

### Build a binary site-by-link matrix, making the process affect the points at
### an angle of 45 degrees
bin.mat.45 <- aem.build.binary(nb,xy, rot.angle=45)
str(bin.mat.45)

### Build a binary site-by-link matrix, making the process affect the points at
### an angle of pi/3 radians
bin.mat.pi3 <- aem.build.binary(nb,xy,unit.angle="radians", rot.angle=pi/3)
str(bin.mat.pi3)
}
```

`aem.time` *AEM for time series*

Description

This function constructs AEM eigenfunctions for multi-scale analysis of a regular time series or spatial transect of univariate or multivariate data.

Usage

```
aem.time(n, w = NULL, moran = FALSE)
```

Arguments

`n`	Numeric. Number of points in the series.
`w`	A vector of weights to be applied to the edges (columns of matrix E). Equal weights are used if no vector `w` is provided. The length of vector `w` must be (`n`-1) where `n` is the number of points in the spatial or temporal series.
`moran`	Logical. If `TRUE`, Moran's I are computed for all AEM. If `FALSE` (default value), Moran's I are not computed.

Details

Time series represent a form of directional stochastic process. To emphasize the directional nature of the process influencing the data, AEM analysis, which was designed to take trends into account, should be applied to the non-detrended series. MEM analysis (see `scores.listw`) can be applied to data series that were detrended to remove the directional component as recommended by Blanchet et al. (2008, 2011) and Legendre & Legendre (2012, Subsection 14.1.2). Detrended palaeoecological sediment core data, for example, could be studied by MEM analysis.

No data file needs to be provided to this function. The AEM eigenvectors are constructed from a matrix E generated from the regular sequence of points along the series.

A vector of weights `w` can be provided, representing the ease of communication of matter, energy or information among the points. The most simple form would be the inverse of (d/dmax) where d

is the distance between adjacent nodes and dmax is the maximum distance between adjacent nodes in the spatial or time series. More general forms of weights may represent the inverse of landscape resistance to the movement of organisms, propagules, genes, etc.

If the calculation of Moran's I is requested, the point coordinates are generated from the point positions along the series.

Value

`E`	Nodes-by-edges matrix E.
`values`	Eigenvalues of the principal component analysis of E.
`aem`	Matrix of AEM eigenfunctions normalized to unit length.
`Moran`	Moran's I statistics tested by a bilateral test with 999 permutations
`listw`	An object of class `listw` with the associated spatial weighting matrix

Author(s)

Pierre Legendre and F. Guillaume Blanchet

References

Blanchet F.G., P. Legendre and Borcard D. (2008) Modelling directional spatial processes in ecological data. *Ecological Modelling*, 215, 325-336.

Blanchet F.G., P. Legendre, R. Maranger, D. Monti, and P. Pepin. (2011) Modelling the effect of directional spatial ecological processes at different scales. *Oecologia*, 166, 357-368.

Legendre, P. and L. Legendre (2012) *Numerical Ecology*, 3rd English edition. Elsevier Science BV, Amsterdam.

Legendre, P. and O. Gauthier (2014) Statistical methods for temporal and space-time analysis of community composition data. *Proceedings of the Royal Society B - Biological Sciences*, 281, 20132728.

See Also

`aem`, `scores.listw`

Examples

```
# Time series containing 20 equispaced observations
out <- aem.time(20, moran = TRUE)

# Time series containing 20 observations with unequal spacing
# Generate (n-1) random interpoint distances
distances <- runif(19,1,5)

# Compute weights representing the ease of communication among points
w <- 1/(distances/max(distances))

# Compute the AEM eigenfunctions
```

```
out <- aem.time(20, w = w, moran = TRUE)
```

`aem.weight.edges` *Weight edges when constructing AEM variables*

Description

These functions construct a vector of weights that can be associated to the edges of the connexion diagram used as a basis to build AEM eigenfunctions. `aem.weight.edges` is general and can be used for 1 or 2 dimensional problems. `aem.weight.time` is meant to be used only for time series. It is a wrapper for `aem.weight.edges`.

Usage

```
aem.weight.edges(nb.object, coords, distmat = NULL, alpha = 2,
  beta = NULL, max.d = NULL, unit.angle = "degrees", rot.angle = 0,
  rm.same.y = TRUE, plot.connexions = TRUE)

aem.weight.time(dates, distmat = NULL, alpha = 2, beta = NULL,
  max.d = NULL, unit.angle = "degrees", rot.angle = 0,
  rm.same.y = TRUE, plot.connexions = TRUE)
```

Arguments

`nb.object`	Object with class 'nb', computed by the spdep package, containing a list of neighbours for each sampling unit (site or time).
`coords`	A three-column matrix or data frame. Column 1: identifiers of the points (must be numeric). Columns 2 and 3: the X and Y coordinates of the points.
`distmat`	Class 'matrix' or 'dist' object containing a dissimilarity or distance matrix. (See details).
`alpha`	Numeric. Exponent of the first weighting function. (See details).
`beta`	Numeric. Exponent of the second weighting function. (See details).
`max.d`	Numeric. Maximum distance for weighting. Default value if max.d=NULL: the maximum distance among a set of sites divided by 2 or the full span of a time series divided by 2 (not recommended in most problems, see details). A warning is given if `max.d = NULL` and the default value is used.
`unit.angle`	Character. The measurement units in which the angle is defined: either "degrees" (default) or "radians".
`rot.angle`	Numeric. Angle of the vector describing the process influencing the sites. This argument generates a rotation of the site coordinates. The set of coordinates is rotated counterclockwise. Negative values will produce a clockwise rotation.
`rm.same.y`	Logical (`TRUE`, `FALSE`). Determines if the links perpendicular to the gradient should be removed. Default value: `TRUE`. If these links have already been removed, this argument put to `TRUE` will make the function crash. See details for more information.

`plot.connexions`
: Logical (`TRUE`, `FALSE`). Determines if the sites and the associated connexion diagram should be plotted after rotation of the coordinates by `gradient.angle`.

`dates`
: A vector of dates, class 'numeric' or 'Date'.

Details

These functions should be used in close relationship with `aem.build.binary`, consequently many of the arguments in this function and in `aem.build.binary` are the same.

The argument `distmat` may contain general forms of dissimilarity, for example the difficulty of transfer of individuals, matter or energy among the sampling units through space or time.

In `aem.weight.edges`, two weighting functions, described in Legendre and Legendre (2012, eqs. 114.3 and 14.4) have been implemented, where d_{ij} is the distance between sites i and j:

$$\begin{aligned} \texttt{Weighting function 1:} \quad & 1 - (d_{ij}/max(d))^{\alpha} \\ \texttt{Weighting function 2:} \quad & 1/d_{ij}^{\beta} \end{aligned}$$

Also note that if a value is provided for `beta` (that is, if it is not `NULL`), weighting function 2 is used regardless of whether `alpha` is defined or not.

In most applications, the default value of `max.d` is not optimal. A more meaningful solution in many applications is to compute a Moran's I correlogram (for univariate data) or a Mantel correlogram (for multivariate data), and provide the distance where the correlation becomes 0 as the value for max.d.

Value

A vector of weights associating a value to each edge of the graph.

Functions

- `aem.weight.time`:

Author(s)

Olivier Gauthier, Pierre Legendre and F. Guillaume Blanchet

References

Legendre, P. and L. Legendre (2012) *Numerical Ecology*, 3rd English edition. Elsevier Science BV, Amsterdam.

Legendre, P. and O. Gauthier (2014) Statistical methods for temporal and space-time analysis of community composition data. *Proceedings of the Royal Society B - Biological Sciences*, 281, 20132728.

See Also

`aem.build.binary`, `sp.correlogram`, `mantel.correlog`

Examples

```
### Time serie example
### Example - 12 dates (days from January 1st of year 1)
### in a 6-year study starting September 5, 2000
if(require("spdep", quietly = TRUE)){
dates <- as.Date(c(129,269,500,631,864,976,1228,1352,1606,1730,1957,2087),origin="2000/1/1")
autocor.limit <- 522  # Limit of autcorrelation in the correlogram

### Using aem.weight.time()
(wtime <- aem.weight.time(dates, alpha=2, max.d=autocor.limit))
### Using aem.weight.edges()
n <- length(dates)
nb <- cell2nb(1, n)
xy.dates <- cbind(1:n, rep(1, n), dates)
(wtime <- aem.weight.edges(nb, xy.dates, alpha=2, max.d=autocor.limit))

n <- length(dates)
nb <- cell2nb(1, n)
xy.dates <- cbind(1:n, dates, rep(1, n)) ## Note the inversion of 'dates' and 'rep(1,n)'
wtime <- aem.weight.edges(nb, xy.dates, alpha=2,
max.d=autocor.limit,rot.angle=90) # Note that 'rot.angle=90' was used

### Spatial example using default d.max (notice the warning)
##############################################################################
nb<-cell2nb(5,5,"queen")
xy <- cbind(1:25,expand.grid(1:5,1:5))
(wspace <- aem.weight.edges(nb,xy))
}
```

---

bacProdxy *Bacterial production data set*

Description

Longitude and latitude of 25 samples when bacterial production was sampled in Lake St. Pierre (Québec, Canada). These sampled were carried out in August 18, 2005.

Usage

```
data(bacProdxy)
```

Format

A data frame with the coordinates of the 25 sampled locations

References

Blanchet F.G., P. Legendre, R. Maranger, D. Monti, and P. Pepin. (2011) Modelling the effect of directional spatial ecological processes at different scales. *Oecologia*, 166, 357-368.

`beta.div` *Beta diversity computed as Var(Y)*

Description

Compute estimates of total beta diversity as the total variance in a community data matrix Y, as well as derived SCBD and LCBD statistics, for 19 dissimilarity coefficients or the raw data table. Computing beta diversity as Var(Y) for raw, untransformed community composition data is not recommended. Tests of significance of the LCBD indices are also produced.

Usage

```
beta.div(Y, method = "hellinger", sqrt.D = FALSE, samp = TRUE,
  nperm = 999, adj = TRUE, save.D = FALSE, clock = FALSE)
```

Arguments

`Y`	Community composition data. The object class can be either `data.frame` or `matrix`.
`method`	One of the 19 dissimilarity coefficients available in the function: `"hellinger"`, `"chord"`, `"log.chord"`, `"chisquare"`, `"profiles"`, `"percentdiff"`, `"ruzicka"`, `"divergence"`, `"canberra"`, `"whittaker"`, `"wishart"`, `"kulczynski"`, `"jaccard"`, `"sorensen"`, `"ochiai"`, `"ab.jaccard"`, `"ab.sorensen"`, `"ab.ochiai"`, `"ab.simpson"`, `"euclidean"`. See Details. Names can be abbreviated to a non-ambiguous set of first letters. Default: `method="hellinger"`.
`sqrt.D`	If `sqrt.D=TRUE`, the dissimilarities in matrix D are square-rooted before computation of SStotal, BDtotal and LCBD. This transformation may be useful for methods `"manhattan"`, `"whittaker"`, `"divergence"`, `"canberra"`, `"percentdiff"`, `"ruzicka"`, `"wishart"` since square-root transformation of the dissimilarities makes these D matrices Euclidean. • Note 1 – Euclideanarity is useful for ordination by principal coordinate analysis; lack of this property does not adversely affect SStotal, BDtotal and LCBD. • Note 2 – The logical value given to parameter `sqrt.D` has no incidence on calculations through methods `"euclidean"`, `"profiles"`, `"hellinger"`, `"log.chord"`, `"chord"`, `"chisquare"` since no D matrix is computed in those cases. • Note 3 – For methods `"jaccard"`, `"sorensen"`, `"ochiai"`, that function produces the dissimilarity matrix in the form sqrt(D), which is Euclidean.
`samp`	If `samp=TRUE`, the abundance-based distances (ab.jaccard, ab.sorensen, ab.ochiai, ab.simpson) are computed for sample data. If `samp=FALSE`, they are computed for true population data.

nperm	Number of permutations for the tests of significance of LCBD indices.
adj	Compute adjusted p-values using the Holm method. Default: adj=TRUE
save.D	If save.D=TRUE, the distance matrix will appear in the output list.
clock	If clock=TRUE, the computation time is printed. Useful when nperm is large.

Details

Calculations may be carried out in two ways, depending on the selected method.

- For untransformed or transformed raw data, the total sum of squares (SStotal) is first computed, then the total beta diversity (BDtotal), which is SStotal divided by (n – 1), is calculated. This algorithm is used for methods "euclidean", "profiles", "hellinger", "chord", "log.chord", "chisquare". No transformation of the data is computed when the method is "euclidean". For methods "profiles", "hellinger", "chord", "log.chord", "chisquare", the algorithm begins with computation of the same-name transformation of the community data (Legendre and Gallagher 2001; Legendre and Legendre 2012, Section 7.7; Legendre and Borcard submitted); SStotal and BDtotal are then computed for the transformed data, followed by calculation of the SCBD and LCBD indices.
- Calculations of BDtotal can also be conducted from a dissimilarity matrix. SStotal is computed by summing the squared dissimilarities in the lower triangular dissimilarity matrix and dividing by n. Then, total beta diversity (BDtotal) is obtained by dividing SStotal by (n-1). With option sqrt.D = TRUE, the computation of SStotal is equivalent to summing the distances instead of the squared distances. Choices are: "whittaker", "divergence", "canberra", "percentdiff", "ruzicka", "wishart", "kulczynski", "ab.jaccard", "ab.sorensen", "ab.ochiai", "ab.simpson", "jaccard", "sorensen", "ochiai". Equations for these dissimilarities are presented in Table 1 of Legendre and De Cáceres (2013). The Ružička index is described in Legendre (2014); this coefficient is suitable for beta diversity studies. See Chao et al. (2006) for details about the abundance-based (ab) coefficients.

Community composition data can be log-transformed prior to analysis with the chord distance; see Legendre and Borcard (submitted). The log(y+1) transformation (log1p function of base) reduces the asymmetry of the species distributions. The chord-log distance, readily available among the methods of the beta.div function, is the chord distance computed on log(y+1)-transformed data. This combined transformation is meaningful for community composition data because the log is one of the transformations in the Box-Cox series, corresponding to exponent 0; see Legendre and Legendre (2012, Section 1.5.6). Exponent 1 (no transformation of the data) followed by the chord transformation and calculation of the Euclidean distance would simply produce the chord distance. Exponent 0.5 (square root) followed by the chord transformation and the Euclidean distance would produce the Hellinger distance. The chord, Hellinger and log-chord distances represent a series where the data are increasingly transformed to reduce the asymmetry of the distributions. Note that it is meaningless to subject log-transformed community compostion data to the "profiles", "hellinger", or "chisquare" distances available in this function.

The Jaccard, Sørensen and Ochiai coefficients are the binary forms of 10 of the 12 dissimilarity coefficients (including the Ružička index) that are suitable for beta diversity assessment. The equivalences are described in Legendre and De Cáceres (2013, Table 1). These popular coefficients can be computed directly using function beta.div without going to the trouble of applying the quantitative forms of these coefficients to data reduced to presence-absence form. beta.div produces

the dissimilarity matrix in the form sqrt(D), which is Euclidean. Hence for these three coefficients, function `beta.div` should be used with option `sqrt.D=FALSE`.

Species contributions to beta diversity (SCBD indices for the species) are computed for untransformed or transformed raw data, but they cannot be computed from dissimilarity matrices.

Local contributions to beta diversity (LCBD indices) represent the degree of uniqueness of the sites in terms of their species compositions. They can be computed in all cases: raw (not recommended) or transformed data, as well as dissimilarity matrices. See Legendre and De Cáceres (2013) for details. LCBD indices are tested for significance by random, independent permutations within the columns of Y. This permutation method tests H0 that the species are distributed at random, independently of one another, among the sites, while preserving the species abundance distributions in the observed data. See Legendre and De Cáceres (2013) for discussion.

This version of `beta.div` calls computer code written in C to speed up computation, especially for the permutation tests of the LCBD indices.

Value

A list containing the following results:

- `beta`: Total sum of squares and total beta diversity [= Var(Y)] of the data matrix. BDtotal statistics computed with the same D index are comparable among data sets having the same or different numbers of sampling units (n), provided that they are of the same size or represent the same sampling effort.
- `SCBD`: Vector of Species contributions to beta diversity (SCBD), if computed.
- `LCBD`: Vector of Local contributions to beta diversity (LCBD) for the sites.
- `p.LCBD`: P-values associated with the LCBD indices.
- `p.adj`: Corrected P-values for the LCBD indices, Holm correction.
- `method`: Method selected.
- `note`: Notes indicate whether the selected coefficient is Euclidean or not.
- `D`: The distance matrix if `save.D=TRUE`.

When all sites contain a different set of species with no species in common, the maximum value that BDtotal can take depends on the method used in the calculation.

- With methods `"hellinger"`, `"chord"`, `"profiles"`, which have maximum values of sqrt(2), BDtotal produces an index in the range [0, 1] with a maximum value of 1.
- For dissimilarity indices with maximum values of 1, BDtotal has a maximum value of 0.5.
- Dissimilarity indices that do not have maximum values of 1 or sqrt(2) produce BDtotal values that do not have an upper bound; hence they cannot be compared across taxonomic groups or among study sites. This group includes the chi-square distance.

See Legendre & De Caceres (2013, p. 957–958), Table 2 and section Maximum value of BD.
For two sites only, the LCBD results are not interesting. With all coefficients, the two LCBD indices are equal to 0.5. The two associated p-values are 1 because LCBD is 0.5 for all columnwise permutations of the data.
The calculation is aborted when Y only contains two identical rows of data. In that case, SStotal and BDtotal are 0 and the LCBD indices cannot be computed (value NaN).

Author(s)

Pierre Legendre <pierre.legendre@umontreal.ca>

References

Chao, A., R. L. Chazdon, R. K. Colwell and T. J. Shen. 2006. Abundance-based similarity indices and their estimation when there are unseen species in samples. Biometrics 62: 361–371.

Legendre, P. 2014. Interpreting the replacement and richness difference components of beta diversity. Global Ecology and Biogeography 23: 1324-1334.

Legendre, P. and D. Borcard. (Submitted). Box-Cox-chord transformations for community composition data prior to beta diversity analysis.

Legendre, P. and M. De Cáceres. 2013. Beta diversity as the variance of community data: dissimilarity coefficients and partitioning. Ecology Letters 16: 951-963.

Legendre, P. and E. D. Gallagher, E.D. 2001. Ecologically meaningful transformations for ordination of species data. Oecologia 129: 271–280.

Legendre, P. and Legendre, L. 2012. Numerical Ecology. 3rd English edition. Elsevier Science BV, Amsterdam.

Examples

```
if(require("vegan", quietly = TRUE) & require("adegraphics", quietly = TRUE)){
data(mite)
res = beta.div(mite, "hellinger", nperm=999)

# Plot a map of the LCBD indices using the Cartesian coordinates
data(mite.xy)
s.value(mite.xy, res$LCBD, symbol = "circle", col = c("white", "brown"), main="Map of mite LCBD")

### Example using the mite abundance data and the percentage difference dissimilarity
res = beta.div(mite, "percentdiff", nperm=999, clock=TRUE)

# Plot a map of the LCBD indices
signif = which(res$p.LCBD <= 0.05) # Which are the significant LCBD indices?
nonsignif = which(res$p.LCBD > 0.05) # Which are the non-significant LCBD indices?
g1 <- s.value(mite.xy[signif,], res$LCBD[signif], ppoint.alpha = 0.5, plegend.drawKey = FALSE,
 symbol = "circle", col = c("white", "red"), main="Map of mite LCBD (red = significant indices)")
g2 <- s.value(mite.xy[nonsignif,], res$LCBD[nonsignif], ppoint.alpha = 0.5,
symbol = "circle", col = c("white", "blue"))
g2+g1
}
```

`beta.div.comp` *Decompose D in replacement and richness difference components*

Description

Podani-family and Baselga-family decompositions of the Jaccard and Sørensen dissimilarity coefficients and their quantitative forms (Ruzicka and percentage difference) into replacement and richness difference components, for species presence-absence or abundance data, as described in Legendre (2014).

Usage

```
beta.div.comp(mat, coef = "J", quant = FALSE, save.abc = FALSE)
```

Arguments

`mat`	Community composition data (`data.frame` or `matrix`).
`coef`	Family of coefficients to be computed. • "S" or "Sorensen": Podani family, Sørensen-based indices. • "J" or "Jaccard": Podani family, Jaccard-based indices. • "BS" – Baselga family, Sørensen-based indices. • "BJ": Baselga family, Jaccard-based indices. • "N": Podani & Schmera (2011) relativized nestedness index. The quantitative form of the Sørensen dissimilarity is the percentage difference index. The quantitative form of the Jaccard dissimilarity is the Ruzicka index.
`quant`	If `TRUE`, compute the quantitative forms of replacement, nestedness and D. If `FALSE`, compute the presence-absence forms of the coefficients.
`save.abc`	If `TRUE`, save the matrices of parameters a, b and c used in presence-absence calculations.

Details

For species presence-absence data, the dissimilarity coefficients are Jaccard = (b+c)/(a+b+c) and Sørensen = (b+c)/(2*a+b+c) with the usual a,b,c notation. For species abundance data, the dissimilarity coefficients are the Ruzicka index = (B+C)/(A+B+C) and Odum's percentage difference = (B+C)/(2A+B+C) (aka Bray-Curtis in some packages), where

- A = sum of the intersections (or minima) of species abundances at two sites,
- B = sum of abundances at site 1 minus A,
- C = sum of abundances at site 2 minus A.

The binary (`quant=FALSE`) and quantitative (`quant=TRUE`) forms of the S and J indices return the same values when computed for presence-absence data.

Value

A list containing the following results:

- `repl`: Replacement matrix, class = dist.
- `rich`: Richness/abundance difference or nestedness matrix (class `dist`). With options "BJ", "BS" and "N", `rich` contains nestedness indices. With option "N", the repl[i,j] and rich[i,j] values do not add up to D[i,j].
- `D`: Dissimilarity matrix (class`dist`).
- `part`: Beta diversity partitioning vector:
  1. BDtotal (total beta diversity) = sum(D.ij)/(n*(n-1)) (Legendre & De Cáceres 2013). This is equal to sum(d.ij^2)/(n*(n-1)) where d.ij = sqrt(D.ij). The dissimilarities are square-rooted because the Jaccard, Sørensen, Ruzicka and percentage difference indices are not Euclidean.
  2. Repl = Total replacement diversity.
  3. RichDiff\|Nes = Total richness difference diversity (or nestedness).
  4. Repl/BDtotal = Total replacement diversity/Total beta diversity.
  5. RichDiff/BDtotal = Total richness difference diversity (or nestedness)/Total beta diversity.
- `note`: Name of the dissimilarity coefficient.

The Jaccard and Sørensen dissimilarity coefficients and their quantitative forms, the Ruzicka and percentage difference indices, all have upper bounds (Dmax) of 1. Hence, when all sites contain a different set of species with no species in common, the maximum value that BDtotal can take is 0.5. See Legendre & De Caceres (2013, p. 958), section Maximum value of BD. This differs form the values produced by function beta.div(): with methods "hellinger", "chord" and "profiles", which have maximum values of sqrt(2), BDtotal has a maximum value of 1 for these dissimilarities.

Author(s)

Pierre Legendre <pierre.legendre@umontreal.ca>

References

Baselga, A. (2010) Partitioning the turnover and nestedness components of beta diversity. Global Ecology and Biogeography, 19, 134–143.

Baselga, A. (2012) The relationship between species replacement, dissimilarity derived from nestedness, and nestedness. Global Ecology and Biogeography, 21, 1223–1232.

Baselga, A. (2013) Separating the two components of abundance-based dissimilarity: balanced changes in abundance vs. abundance gradients. Methods in Ecology and Evolution, 4, 552–557.

Carvalho, J.C., Cardoso, P., Borges, P.A.V., Schmera, D. & Podani, J. (2013) Measuring fractions of beta diversity and their relationships to nestedness: a theoretical and empirical comparison of novel approaches. Oikos, 122, 825–834.

Legendre, P. 2014. Interpreting the replacement and richness difference components of beta diversity. Global Ecology and Biogeography, 23, 1324-1334.

Legendre, P. and M. De Cáceres. 2013. Beta diversity as the variance of community data: dissimilarity coefficients and partitioning. Ecology Letters 16: 951-963.

Podani, J., Ricotta, C. & Schmera, D. (2013) A general framework for analyzing beta diversity, nestedness and related community-level phenomena based on abundance data. Ecological Complexity, 15, 52-61.

Podani, J. & Schmera, D. 2011. A new conceptual and methodological framework for exploring and explaining pattern in presence-absence data. Oikos, 120, 1625–1638.

See Also

LCBD.comp

Examples

```
if(require(ade4, quietly = TRUE)){
data(doubs)
fish.sp = doubs$fish[-8,]   # Fish data; site 8 is removed because no fish were caught

# Compute and partition a matrix of Jaccard indices (presence-absence data)
out1 = beta.div.comp(fish.sp, coef="J", quant=FALSE)
out1$part

# Compute and partition a matrix of percentage difference indices
# (quantitative form of Sorensen index)
out2 = beta.div.comp(fish.sp, coef="S", quant=TRUE)
out2$part
# In paragraph Value, see the description of the 5 elements of vector part.
# Is the fish beta diversity dominated by replacement or richness/abundance difference?
}
```

---

chooseCN *Function to choose a connection network*

Description

The function chooseCN is a simple interface to build a connection network (CN) from xy coordinates. The user chooses from 6 types of graph and one additional weighting scheme. chooseCN calls functions from appropriate packages, handles non-unique coordinates and returns a connection network either with classe nb or listw. For graph types 1-4, duplicated locations are not accepted and will issue an error.

Usage

```
chooseCN(xy, ask = TRUE, type = NULL, result.type = "nb",
  d1 = NULL, d2 = NULL, k = NULL, a = NULL, dmin = NULL,
  plot.nb = TRUE, edit.nb = FALSE)
```

Arguments

xy	an matrix or data.frame with two columns for x and y coordinates.
ask	a logical stating whether graph should be chosen interactively (TRUE,default) or not (FALSE). Set to FALSE if type is provided.
type	an integer giving the type of graph (see details).
result.type	a character giving the class of the returned object. Either "nb" (default) or "listw", both from spdep package. See details.
d1	the minimum distance between any two neighbours. Used if type=5.
d2	the maximum distance between any two neighbours. Used if type=5. Can also be a character: "dmin" for the minimum distance so that each site has at least one connection, or "dmax" to have all sites connected (despite the later has no sense).
k	the number of neighbours per point. Used if type=6.
a	the exponent of the inverse distance matrix. Used if type=7.
dmin	the minimum distance between any two distinct points. Used to avoid infinite spatial proximities (defined as the inversed spatial distances). Used if type=7.
plot.nb	a logical stating whether the resulting graph should be plotted (TRUE, default) or not (FALSE).
edit.nb	a logical stating whether the resulting graph should be edited manually for corrections (TRUE) or not (FALSE, default).

Details

There are 7 kinds of graphs proposed:
Delaunay triangulation (type 1)
Gabriel graph (type 2)
Relative neighbours (type 3)
Minimum spanning tree (type 4)
Neighbourhood by distance (type 5)
K nearests neighbours (type 6)
Inverse distances (type 7)

The last option (type=7) is not a true neighbouring graph: all sites are neighbours, but the spatial weights are directly proportional to the inversed spatial distances.
Also not that in this case, the output of the function is always a listw object, even if nb was requested.

The choice of the connection network has been discuted on the adegenet forum. Please search the archives from adegenet website (section 'contact') using 'graph' as keyword.

Value

Returns a connection network having the class nb or listw. The xy coordinates are passed as attribute to the created object.

Author(s)

Thibaut Jombart <t.jombart@imperial.ac.uk>

Examples

```
if(require("ade4", quietly = TRUE)){
data(mafragh)

par(mfrow=c(2,2))
cn1 <- chooseCN(mafragh$xy,ask=FALSE,type=1)
cn2 <- chooseCN(mafragh$xy,ask=FALSE,type=2)
cn3 <- chooseCN(mafragh$xy,ask=FALSE,type=3)
cn4 <- chooseCN(mafragh$xy,ask=FALSE,type=4)

}
```

Cperiodogram *Contingency periodogram*

Description

Function to compute a contingency periodogram for a univariate series of qualitative data

Usage

```
Cperiodogram(x, T1 = 2, T2 = NULL, nperm = NULL, alpha = 0.05,
  graph = TRUE)
```

Arguments

x	a qualitative variable (factor)
T1	first period included in the calculations (default: T1 = 2)
T2	last period included in the calculations (default: T2 = n/2)
nperm	Number of permutations for the chi-square test. For chi-square tests using the chi- square distribution, use the default nperm=NULL
alpha	significance level for computation of the confidence limits
graph	a logical indicating if a graph is requested, by default TRUE.

Details

The contingency periodogram of Legendre et al. (1981) identifies periodic components in qualitative data vectors. The vector may contain classes of a qualitative variable or the classes obtained by hierarchical clustering or partitioning of a multivariate data table. The method is also described in Legendre & Legendre (2012). The optional graph produced by the function shows the following information:

- In red: the B statistics (information in common).
- In blue: Confidence limits for B without correction.
- In green: Bonferroni-corrected limits of the confidence intervals.
- In black: Confidence limits with progressive Bonferroni correction.

Value

A table with the statistics for the selected periods:

- Wilks' chi-square statistic (Wilks.chisq)
- information in common (B),
- degrees of freedom (df),
- p-value (prob)

Confidence interval limits:

- critical value of B without correction (B.crit),
- critical value of B with Bonferroni correction based on the number of periods studied in the periodogram (B.crit.Bonf),
- critical value of B with progressive Bonferroni correction (B.prog.Bonf).

Author(s)

Pierre Legendre <pierre.legendre@umontreal.ca>

References

Legendre, L., M. Fréchette & P. Legendre. 1981. The contingency periodogram: a method of identifying rhythms in series on nonmetric ecological data. Journal of Ecology 69: 965-979.

Legendre, P. and Legendre, L. 2012. Numerical Ecology. 3rd English ed. Elsevier, Amsterdam

Examples

```
# Data from the numerical example of Subsection 12.4.2 of Legendre and Legendre (2012).
test.vec <- c(1,1,2,3,3,2,1,2,3,2,1,1,2,3,3,1)
# Periodogram with tests using the chi-square distribution
res <- Cperiodogram(test.vec)
# Periodogram with permutation tests
res <- Cperiodogram(test.vec, nperm=2000, graph=FALSE)
```

`create.dbMEM.model` *Combine dbMEM matrices corresponding to groups of sites*

Description

This function reads a file containing the Cartesian coordinates of sites forming different groups on the map, and constructs a combined staggered matrix of dbMEM spatial eigenvectors, ready for use in RDA. The method was first described and used in Declerck et al. (2011) and summarized in the Borcard et al. (2011) book, section 7.4.3.5. These publications provided preliminary versions of the present function. The present version is more completely documented. Furthermore, it uses the `dbmem` function of the `adespatial` package for computation of the eigenfunctions.

Usage

```
create.dbMEM.model(coord = NULL, D.mat = NULL, nsites)
```

Arguments

`coord`	Optional file containing the Cartesian coordinates of the sites.
`D.mat`	Optional distance matrix provided by user, class `matrix` or `dist`. If `D.mat=NULL`, the geographic distance matrix will be computed from the coordinates provided in file `coord`.
`nsites`	A vector containing the number of sites per group.

Details

The geographic positions of the sites are provided either in a file of geographic coordinates `coord` or as a geographic distance matrix `D.mat`.

The sites must, of course, be in the same order in file `coord` (or in file `D.mat`) and in the response data file used in the RDA. All sites of a group must be together in these two files, i.e. not interspersed. The numbers of sites in the groups are provided in vector `nsites`. See example.

File vector `coord`, if provided, must contain Cartesian coordinates of the sites, not coordinates in degrees. The Euclidean distance computed from the geographic coordinates is a meaningful representation of the geographic relationships only if the coordinates are Cartesian. Geodetic Cartesian coordinates can be derived from Lat-Lon data in degrees using the function `geoXY` of the `SoDA` package. Beware of UTM coordinates if the sites are not all located in the same UTM zone; UTM coordinates are Cartesian only within an UTM zone. See https://en.wikipedia.org/wiki/Universal_Transverse_Mercator_coordinate_syste

Value

A matrix with `n` rows containing a set of `k` staggered matrices of dbMEM eigenfunctions in its diagonal portion; `n` is the total number of sites in the study and `k` is the number of groups. Each small matrix contains the dbMEM functions, modelling positive spatial correlation, describing the spatial relationships among the sites of a group. The remainder of the matrix is filled with zeros. Zero is the mean value of all eigenfunctions describing within-group relationships. This means that during the calculation of RDA, the sites of a focus group will have, with each other, relationships

described by the dbMEM eigenfunctions of that group, whereas the sites outside that group will have weights of 0 in the regressions that concern these eigenfunctions.

Author(s)

Pierre Legendre <pierre.legendre@umontreal.ca>, 2010. Adaptation to adespatial: Daniel Borcard and Pierre Legendre, 2016

References

Borcard, D., F. Gillet and P. Legendre. 2011. Numerical ecology with R. Use R! series, Springer Science, New York.

Declerck, S. A. J., J. S. Coronel, P. Legendre & L. Brendonck. 2011. Scale dependency of processes structuring metacommunities of cladocerans in temporary pools of High-Andes wetlands. Ecography 34: 296-305.

See Also

dbmem

Examples

```
{
 # Generate random coordinates for 35 sites forming 6 distinct groups on the map
 Easting <- runif(35)+c(rep(0,6),rep(1.5,7),rep(3,6), rep(0,5),rep(1.5,5),rep(3,6))
 Northing<- runif(35)+c(rep(2.8,6),rep(2.3,7),rep(2.8,6), rep(0,5),rep(0.5,5),rep(0,6))
 cartesian <- cbind(Easting,Northing)
 rownames(cartesian) <- paste("S",1:nrow(cartesian),sep='')
 nsites.per.group <- c(6,7,6,5,5,6)

 result <- create.dbMEM.model(coord=cartesian, nsites=nsites.per.group)

 # Draw a map to check the coding of the sites into the groups
 site.codes <- unlist(apply(cbind(1:6),1,n=nsites.per.group,function(a,n) rep(a,n[a])))

 col.vec <- c("green3","gray99","orange2","gold1","brown3","gray70")
 plot(cartesian, pch=22, col="black", bg=col.vec[site.codes], cex=2, ylim=c(0,4),asp=1)
 text(cartesian,labels=rownames(cartesian), cex=0.5, pos=3)

 # Examine the staggered matrix of dbMEM eigenfunctions
 # Not run:
 result
}
```

dbmem *dbMEM spatial eigenfunctions*

Description

Compute distance-based Moran's eigenvector maps (dbMEM, also called dbMEM spatial eigenfunctions) from a geographic distance matrix, in view of spatial eigenfunction analysis.

Usage

```
dbmem(xyORdist, thresh = NULL, MEM.autocor = c("positive", "non-null",
  "all", "negative"), store.listw = TRUE, silent = TRUE)
```

Arguments

xyORdist	Either a matrix of spatial coordinates or a distance matrix (class dist).
thresh	A threshold value for truncation of the geographic distance matrix. If thresh=NULL, the length of the longest edge of the minimum spanning tree will be used as the threshold (as returned by the function give.thresh).
MEM.autocor	A string indicating if all MEMs must be returned or only those corresponding to non-null, positive or negative autocorrelation. The difference between options all and non-null is the following: when there are several null eigenvalues, option all removes only one of the eigenvectors with null eigenvalues and returns (n-1) eigenvectors, whereas non-null does not return any of the eigenvectors with null eigenvalues. Default: MEM.autocor="positive".
store.listw	A logical indicating if the spatial weighting matrix should be stored in the attribute listw of the returned object
silent	A logical indicating if some information should be printed during computation: truncation level and time to compute the dbmem

Details

dbMEM eigenfunctions were called PCNM in early papers (Borcard and Legendre 2002, Borcard et al. 2004). There is a small difference in the computation: to construct PCNMs, the distance matrix subjected to PCoA contained zeros on the diagonal. In dbMEM, the matrix contains 4*thresh values on the diagonal. The result is that the dbMEM eigenvalues are smaller than the PCNM eigenvalues by a constant (equal to (n.sites * (4*thresh)^2)/2). The dbMEM eigenvalues are proportional to Moran's I coefficient of spatial correlation (Dray et al. 2006; Legendre and Legendre 2012). The dbMEM eigenvectors only differ from the PCNM eigenvectors by a multiplicative constant; this has no impact on the use of MEMs as explanatory variables in linear models. In this implementation, dbMEM eigenvectors have a norm equal to 1 (using the uniform weigts 1/n.sites).

If a truncation value is not provided, the largest distance in a minimum spanning tree linking all sites on the map is computed (returned by the function give.thresh). That value is used as the truncation threshold value (thresh).

A square regular grid produces multiple eigenvalues (i.e. eigenvalues that are equal) and multiple eigenvalues have an infinity of eigenvector solutions. Hence, different eigenvectors may be produced by this function on computers with different operating systems or implementations of R. In addition, the eigenvectors found by the dbmem function from the site coordinates may differ from the eigenvectors computed from the geographic distance matrix among the sites. Nonetheless, the different complete sets of eigenvectors will have the exact same explanatory power (R-square) for a given response vector or matrix, despite the fact that they are not fully correlated on a one-to-one basis. This is, however, not the case for subsets of eigenvectors selected using stepwise procedures.

Value

An object of class `orthobasisSp` , subclass `orthobasis`. The dbMEM eigenfunctions (principal coordinates of the truncated distance matrix) are stored as a `data.frame`. It contains several attributes (see `?attributes`) including:

- `values`: The dbMEM eigenvalues.
- `listw`: The associated spatial weighting matrix (if `store.listw = TRUE`).

Author(s)

Stéphane Dray <stephane.dray@univ-lyon1.fr>, Pierre Legendre, Daniel Borcard and F. Guillaume Blanchet

References

Borcard, D. and P. Legendre. 2002. All-scale spatial analysis of ecological data by means of principal coordinates of neighbour matrices. Ecological Modelling 153: 51-68.

Borcard, D., P. Legendre, C. Avois-Jacquet and H. Tuomisto. 2004. Dissecting the spatial structure of ecological data at multiple scales. Ecology 85: 1826-1832.

Dray, S., P. Legendre and P. R. Peres-Neto. 2006. Spatial modelling: a comprehensive framework for principal coordinate analysis of neighbour matrices (PCNM). Ecological Modelling 196: 483-493.

Legendre, P. and L. Legendre. 2012. Numerical ecology, 3rd English edition. Elsevier Science BV, Amsterdam.

See Also

`give.thresh`, `mem`

Examples

```
if(require("ade4", quietly = TRUE) & require("adegraphics", quietly = TRUE)){

data(oribatid)
mite <- oribatid$fau      # 70 peat cores, 35 species
mite.xy <- oribatid$xy    # Geographic coordinates of the 70 cores

# Example 1: Compute the MEMs corresponding to all non-null eigenvalues
# thresh=1.012 is the value used in Borcard and Legendre (2002)
```

```
mite.dbmem1 <- dbmem(mite.xy, thresh=1.012, MEM.autocor = "non-null", silent = FALSE)
mite.dbmem1

# Print the (n-1) non-null eigenvalues
attributes(mite.dbmem1)$values
# or:  attr(mite.dbmem1, "values")

# Plot the associated spatial weighting matrix
s.label(mite.xy, nb = attr(mite.dbmem1, "listw"))

# Plot maps of the first 3 dbMEM eigenfunctions
s.value(mite.xy, mite.dbmem1[,1:3])

# Compute and test associated Moran's I values
# Eigenvalues are proportional to Moran's I

test <- moran.randtest(mite.dbmem1, nrepet = 99)
plot(test$obs, attr(mite.dbmem1, "values"), xlab = "Moran's I", ylab = "Eigenvalues")

# Decreasing values of Moran's I for the successive MEM.
# The red line is the expected value of Moran's I under H0.

plot(test$obs, xlab="MEM rank", ylab="Moran's I")
abline(h=-1/(nrow(mite.xy) - 1), col="red")

# Example 2: Compute only the MEMs with positive eigenvalues (and positive Moran's I)
mite.dbmem2 <- dbmem(mite.xy, thresh=1.012)
# or:  mite.dbmem2 <- dbmem(dist(mite.xy), thresh=1.012, silent=FALSE)
mite.dbmem2

# Examine the eigenvalues
attributes(mite.dbmem2)$values
# or:  attr(mite.dbmem2, "values")

# Examine (any portion of) the dbmem spatial eigenvectors
tmp <- as.matrix(mite.dbmem2)
tmp[1:10,1:6]
}
```

`dist.ldc` *Dissimilarity matrices for community composition data*

Description

Compute dissimilarity indices for ecological data matrices. The dissimilarity indices computed by this function are those described in Legendre & De Cáceres (2013). In the name of the function, 'ldc' stands for the author's names. Twelve of these 21 indices are not readily available in other R package functions; four of them can, however, be computed in two computation steps in vegan.

Usage

```
dist.ldc(Y, method = "hellinger", binary = FALSE, samp = TRUE,
  silent = FALSE)
```

Arguments

Y	Community composition data. The object class can be either data.frame or matrix.
method	One of the 21 dissimilarity coefficients available in the function: "hellinger", "chord", "log.chord", "chisquare", "profiles", "percentdiff", "ruzicka", "divergence", "canberra", "whittaker", "wishart", "kulczynski", "jaccard", "sorensen", "ochiai", "ab.jaccard", "ab.sorensen", "ab.ochiai", "ab.simpson", "euclidean", "manhattan", "modmeanchardiff". See Details. Names can be abbreviated to a non-ambiguous set of first letters. Default: method="hellinger".
binary	If binary=TRUE, the data are transformed to presence-absence form before computation of the dissimilarities. Default value: binary=FALSE, except for the Jaccard, Sørensen and Ochiai indices where binary=TRUE.
samp	If samp=TRUE, the abundance-based distances (ab.jaccard, ab.sorensen, ab.ochiai, ab.simpson) are computed for sample data. If samp=FALSE, binary indices are computed for true population data.
silent	If silent=FALSE, informative messages sent to users will be printed to the R console. Use silent=TRUE is called on a numerical simulation loop, for example.

Details

The dissimilarities computed by this function are the following. Indices i and k designate two rows (sites) of matrix Y, j designates a column (species). D[ik] is the dissimilarity between rows i and k. p is the number of columns (species) in Y; pp is the number of species present in one or the other site, or in both. y[i+] is the sum of values in row i; same for y[k+]. y[+j] is the sum of values in column j. y[++] is the total sum of values in Y. The indices are computed by functions written in C for greater computation speed with large data matrices.

- Group 1 - D computed by transformation of Y followed by Euclidean distance
  - Hellinger D, D[ik] = sqrt(sum((sqrt(y[ij]/y[i+])-sqrt(y[kj]/y[k+]))^2))
  - chord D, D[ik] = sqrt(sum((y[ij]/sqrt(sum(y[ij]^2))-y[kj]/sqrt(sum(y[kj]^2)))^2))
  - log-chord D, D[ik] = chord D[ik] computed on log(y[ij]+1)-transformed data (Legendre and Borcard submitted)
  - chi-square D, D[ik] = sqrt(y[++] sum((1/j[+j])(y[ij]/y[i+]-y[kj]/y[k+])^2))
  - species profiles D, D[ik] = sqrt(sum((y[ij]/y[i+]-y[kj]/y[k+])^2))
- Group 2 - Other D functions appropriate for beta diversity studies where A = sum(min(y[ij],y[kj])), B = y[i+]-A, C = y[k+]-A
  - percentage difference D (aka Bray-Curtis), D[ik] = (sum(abs(y[ij]-y[k,j])))/(y[i+]+y[k+]) or else, D[ik] = (B+C)/(2A+B+C)
  - Ružička D, D[ik] = 1-(sum(min(y[ij],y[kj])/sum(max(y[ij],y[kj])) or else, D[ik] = (B+C)/(A+B+C)
  - coeff. of divergence D, D[ik] = sqrt((1/pp)sum(((y[ij]-y(kj])/(y[ij]+y(kj]))^2))

  - Canberra metric D, D[ik] = (1/pp)sum(abs(y[ij]-y(kj])/(y[ij]+y(kj]))
  - Whittaker D, D[ik] = 0.5*sum(abs(y[ij]/y[i+]-y(kj]/y[k+]))
  - Wishart D, D[ik] = 1-sum(y[ij]y[kj])/(sum(y[ij]^2)+sum(y[kj]^2)-sum(y[ij]y[kj]))
  - Kulczynski D, D[ik] = 1-0.5((sum(min(y[ij],y[kj])/y[i+]+sum(min(y[ij],y[kj])/y[k+]))

- Group 3 - Classical indices for binary data; they are appropriate for beta diversity studies. Value a is the number of species found in both i and k, b is the number of species in site i not found in k, and c is the number of species found in site k but not in i. The D matrices are square-root transformed, as in dist.binary of ade4; the user-oriented reason for this transformation is explained below.
  - Jaccard D, D[ik] = sqrt((b+c)/(a+b+c))
  - Sørensen D, D[ik] = sqrt((b+c)/(2a+b+c))
  - Ochiai D, D[ik] = sqrt(1 - a/sqrt((a+b)(a+c)))
- Group 4 - Abundance-based indices of Chao et al. (2006) for quantitative abundance data. These functions correct the index for species that have not been observed due to sampling errors. For the meaning of the U and V notations, see Chao et al. (2006, section 3). When `samp=TRUE`, the abundance-based distances (ab.jaccard, ab.sorensen, ab.ochiai, ab.simpson) are computed for sample data. If `samp=FALSE`, indices are computed for true population data. – Do not use indices of group 4 with `samp=TRUE` on presence-absence data; the indices are not meant to accommodate this type of data. If `samp=FALSE` is used with presence-absence data, the indices are the regular Jaccard, Sorensen, Ochiai, Simpson indices. On output, however, the D matrices are not square-rooted, contrary to the Jaccard, Sorensen, Ochiai indices in section 3 which are square-rooted.
  - abundance-based Jaccard D, D[ik] = 1-(UV/(U+V-UV))
  - abundance-based Sørensen D, D[ik] = 1-(2UV/(U+V))
  - abundance-based Ochiai D, D[ik] = 1-sqrt(UV)
  - abundance-based Simpson D, D[ik] = 1-(UV/(UV+min((U-UV),(V-UV))))
- Group 5 – General-purpose dissimilarities that do not have an upper bound (maximum D value). They are inappropriate for beta diversity studies.
  - Euclidean D, D[ik] = sqrt(sum(y[ij]-y[kj])^2)
  - Manhattan D, D[ik] = sum(abs(y[ij] - y[ik]))
  - modified mean character difference, D[ik] = (1/pp) sum(abs(y[ij] - y[ik]))

The properties of all dissimilarities available in this function (except Ružička D) were described and compared in Legendre & De Cáceres (2013), who showed that most of these dissimilarities are appropriate for beta diversity studies. Inappropriate are the Euclidean, Manhattan, modified mean character difference, species profile and chi-square distances. Most of these dissimilarities have a maximum value of either 1 or sqrt(2). Three dissimilarities (Euclidean, Manhattan, Modified mean character difference) do not have an upper bound and are thus inappropriate for beta diversity studies. The chi-square distance has an upper bound of sqrt(2*(sum(Y))).

The Euclidean, Hellinger, chord, chi-square and species profiles dissimilarities have the property of being Euclidean, meaning that they never produce negative eigenvalues in principal coordinate analysis. The Canberra, Whittaker, percentage difference, Wishart and Manhattan coefficients are Euclidean when they are square-root transformed (Legendre & De Cáceres 2013, Table 2). The distance forms (1-S) of the Jaccard, Sørensen and Ochiai similarity (S) coefficients are Euclidean

after taking the square root of (1-S) (Legendre & Legendre 2012, Table 7.2). The D matrices resulting from these three coefficients are outputted in the form sqrt(1-S), as in function `dist.binary` of ade4, because that form is Euclidean and will thus produce no negative eigenvalues in principal coordinate analysis.

The Hellinger, chord, chi-square and species profile dissimilarities are computed using the two-step procedure developed by Legendre & Gallagher (2001). The data are first transformed using either the row marginals, or the row and column marginals in the case of the chi-square distance. The dissimilarities are then computed from the transformed data using the Euclidean distance formula. As a consequence, these four dissimilarities are necessarily Euclidean. D matrices for other binary coefficients can be computed in two ways: either by using function `dist.binary` of ade4, or by choosing option `binary=TRUE`, which transforms the abundance data to binary form, and using one of the quantitative indices of the present function. Table 1 of Legendre & De Cáceres (2013) shows the incidence-based (presence-absence-based) indices computed by the various indices using binary data.

The Euclidean distance computed on untransformed presence-absence or abundance data produces non-informative and incorrect ordinations, as shown in Legendre & Legendre (2012, p. 300) and in Legendre & De Cáceres (2013). However, the Euclidean distance computed on log-transformed abundance data produces meaningful ordinations in principal coordinate analysis (PCoA). Nonetheless, it is easier to compute a PCA of log-transformed abundance data instead of a PCoA; the resulting ordination with scaling 1 will be meaningful. Messages are printed to the R console indicating the Euclidean status of the computed dissimilarity matrices. Note that for the chi-square distance, the columns that sum to zero are eliminated before calculation of the distances, thus preventing divisions by zero in the calculation of the chi-square transformation.

Value

A dissimilarity matrix, with class `dist`.

Author(s)

Pierre Legendre <pierre.legendre@umontreal.ca> and Naima Madi

References

Chao, A., R. L. Chazdon, R. K. Colwell and T. J. Shen. 2006. Abundance-based similarity indices and their estimation when there are unseen species in samples. Biometrics 62: 361–371.

Legendre, P. and D. Borcard. (Submitted). Box-Cox-chord transformations for community composition data prior to beta diversity analysis.

Legendre, P. and M. De Cáceres. 2013. Beta diversity as the variance of community data: dissimilarity coefficients and partitioning. Ecology Letters 16: 951-963.

Legendre, P. and E. D. Gallagher, E.D. 2001. Ecologically meaningful transformations for ordination of species data. Oecologia 129: 271–280.

Legendre, P. and Legendre, L. 2012. Numerical Ecology. 3rd English edition. Elsevier Science BV, Amsterdam.

Examples

```
if(require("vegan", quietly = TRUE)) {
data(mite)
mat1  = as.matrix(mite[1:10, 1:15])   # No column has a sum of 0
mat2 = as.matrix(mite[61:70, 1:15])   # 7 of the 15 columns have a sum of 0

#Example 1: compute Hellinger distance for mat1
D.out = dist.ldc(mat1,"hellinger")

#Example 2: compute chi-square distance for mat2
D.out = dist.ldc(mat2,"chisquare")

#Example 3: compute percentage difference dissimilarity for mat2
D.out = dist.ldc(mat2,"percentdiff")

}
```

`envspace.test` *Perform a test of the shared space-environment fraction of a variation partitioning using torus-translation (TT) or Moran Spectral Randomisation (MSR)*

Description

The function uses two different spatially-constrained null models to test the shared space-environment fraction (SSEF, or fraction [b]) of a variation partitioning of two explanatory components.

Usage

```
envspace.test(spe, env, coord, MEM.spe, listw.env,
  MEM.autocor = c("positive", "negative", "all"), regular = FALSE,
  nperm = 999, MSR.method = "singleton", alpha = 0.05)
```

Arguments

`spe`	Vector, matrix, or dataframe of response variable(s) (e.g. species abundances)
`env`	Vector, matrix, or dataframe of environmental variables (rows = sites, columns = variables)
`coord`	Matrix or dataframe of spatial coordinates of the sampled sites
`MEM.spe`	Matrix or dataframe of spatial predictors (MEM variables) selected for `spe`
`listw.env`	An object of class `listw` (spatial weights) created by the functions of the `spdep` package or returned by `listw.candidates`

MEM.autocor	A string indicating the type of spatial structure of interest for env ("positive", "negative", or "all", for positive, negative, or both types of spatial autocorrelations, respectively); Default is "positive"
regular	Logical argument indicating whether a torus-translation test will be performed, in addition to the MSR. Set to TRUE only if the sampling design is regular (same number of sites on each line, same number of sites on each column). Set to FALSE otherwise; Default is FALSE
nperm	Number of permutations performed; Default is 999
MSR.method	Algorithm of msr to be used to perform the MSR. The three available procedures are "singleton" (default), "pair", and "triplet" (see msr for details)
alpha	Threshold value of null hypothesis rejection for the test of a spatial structure in the environment, and for the shared environment-space fraction of the variation partitioning; Default is 0.05

Details

The function tests the SSEF (also known as fraction [b]) of a variation partitioning of a response variable or matrix (y) between an environmental and a spatial component (env, and MEM.spe, respectively). The SSEF is the explained variation of y shared by env and MEM.spe. The adjusted R-squared (Peres-Neto et al. 2006; R2adj) of the SSEF is not an actual R2, as it is computed by subtracting the adjusted R2adj of other fractions and therefore has zero degree of freedom (Legendre and Legendre 2012). The SSEF can therefore not be computed in the classical way (residuals permutation; Anderson and Legendre 1999, Legendre and Legendre 2012).

The function envspace.test provides two ways of testing this fraction, that is, spatially-constrained null models based either on a torus-translation test (TT) (for regular sampling designs only), or on Moran spectral randomizations (MSR) (for any type of sampling design). The test of the SSEF should only be performed if both the global models of y against all the environmental variables and against all spatial variables are significant (see Bauman et al. 2018c). The function first checks whether the environment displays significant spatial structures, and then proceeds to the test of the SSEF if this condition is fulfilled (details in Bauman et al. 2018c).

spe can be a vector or a multicolumn matrix or dataframe (multivariate response data). If multivariate, it is greatly advised to transform spe prior to performing the variation partitioning and testing the SSEF (e.g., Hellinger transformation; see Legendre and Gallagher 2001).

MEM.spe is a set of spatial predictors (MEM variables). It is recommended to be a well-defined subset of MEM variables selected among the complete set generated from the spatial weighting matrix (SWM) (see review about spatial eigenvector selection in Bauman et al. 2018a). Optimising the selection of a subset of forward-selected MEM variables among a set of candidate SWMs has been shown to increase statistical power as well as R2-estimation accuracy (Bauman et al. 2018b). To do so, MEM.spe can be generated using listw.candidates followed by listw.select. If a SWM has already been selected in another way, then mem.select can be used to generate the MEM variables and to select an optimal subset among them, which can then be used as MEM.spe in envspace.test (see Details of function mem.select). listw.env corresponds to the SWM that will be used to test for a spatial structure in env, and to build the MEM variables for the MSR test. The choice of the SWM for env can also be optimised with listw.select. The SWMs selected for spe and env should be optimised separately to best model the spatial structure of both spe and env (see example).

To verify that env displays a significant spatial pattern, prior to performing the test of the SSEF, a residuals permutation test is performed on the global set of MEM variables (generated internally from listw.env) associated to the type of spatial structure of interest (see argument MEM.autocor). This test is performed with mem.select. The choice of MEM.autocor should be made according to the MEM.autocor argument used to build MEM.spe.

env is a dataset of environmental variables chosen by the user. We recommend dealing with collinearity issues prior to performing the variation partitioning and the test of the SSEF (see Dormann et al. 2013 for a review of methods to cope with collinearity).

regular is a logical argument indicating whether a TT test should be performed instead of the MSR to test the SSEF. Since the TT can only be performed on regular sampling designs, regular should only be set to TRUE if the sampling design is either a transect, or a grid displaying the same number of sites for all lines and columns (although the number of sites per column can differ from the number of sites per line).

listw.env is the SWM used by the MSR to generate spatially-constrained null environmental variables. It should ideally be a SWM optimised on the basis of env using the function listw.select, with the argument method = "global" (see Details of function mem.select for an explanation). This will allow detecting the spatial structures of env as accurately as possible, hence allowing MSR to generate null environmental variables as spatially faithful to the original ones. It is also on the basis of listw.env that MEM variables will be generated to test whether env is spatially structured (i.e. global test) prior to perform the test of the SSEF.

It is worth mentioning that, although a significant SSEF may provide evidence of an induced spatial dependence (Bauman et al. 2018c), a non-significant SSEF only indicates that no induced spatial dependence could be detected in relation with the chosen environmental variables. This does not exclude that this effect may exist with respect to some unmeasured variables.

Value

If the condition of env being spatially structured is fulfilled, the test is performed and the function returns an object of class randtest containing the results of the test.

Author(s)

David Bauman and Jason Vleminckx, <davbauman@gmail.com>, <jasvlx86@gmail.com>

References

Anderson M. and Legendre P. (1999) An empirical comparison of permutation methods for tests of partial regression coefficients in a linear model. Journal of Statistical Computation and Simulation, 62(3), 271–303

Bauman D., Drouet T., Dray S. and Vleminckx J. (2018a) Disentangling good from bad practices in the selection of spatial or phylogenetic eigenvectors. Ecography, 41, 1–12

Bauman D., Fortin M-J, Drouet T. and Dray S. (2018b) Optimizing the choice of a spatial weighting matrix in eigenvector-based methods. Ecology

Bauman D., Vleminckx J., Hardy O., Drouet T. (2018c) Testing and interpreting the shared space-environment fraction in variation partitioning analyses of ecological data. Oikos

Blanchet G., Legendre P. and Borcard D. (2008) Forward selection of explanatory variables. Ecology, 89(9), 2623–2632

Legendre P., Gallagher E.D. (2001) Ecologically meaningful transformations for ordination of species data. Oecologia, 129(2), 271–280

Legendre P. and Legendre L. (2012) Numerical Ecology, Elsevier, Amsterdam

Peres-Neto P., Legendre P., Dray S., Borcard D. (2006) Variation partitioning of species data matrices: estimation and comparison of fractions. Ecology, 87(10), 2614–2625

Peres-Neto P. and Legendre P. (2010) Estimating and controlling for spatial structure in the study of ecological communities. Global Ecology and Biogeography, 19, 174–184

See Also

varpart, listw.select, listw.candidates, mem.select

Examples

```
## Not run:
if(require(vegan)) {
# Illustration of the test of the SSEF on the oribatid mite data
# (Borcard et al. 1992, 1994 for details on the dataset):
# Community data (response matrix):
data(mite)
# Hellinger-transformation of the community data (Legendre and Gallagher 2001):
Y <- decostand(mite, method = "hellinger")
# Environmental explanatory dataset:
data(mite.env)
# We only use two numerical explanatory variables:
env <- mite.env[, 1:2]
dim(Y)
dim(env)
# Coordinates of the 70 sites:
data(mite.xy)
coord <- mite.xy

### Building a list of candidate spatial weighting matrices (SWMs) for the
### optimisation of the SWM selection, separately for 'Y' and 'env':
# We create five candidate SWMs: a connectivity matrix based on a Gabriel graphs, on
# a minimum spanning tree (i.e., two contrasted graph-based SWMs), either
# not weighted, or weighted by a linear function decreasing with the distance),
# and a distance-based SWM corresponding to the connectivity and weighting
# criteria of the original PCNM method:
candidates <- listw.candidates(coord, nb = c("gab", "mst", "pcnm"), weights = c("binary",
                                                                                "flin"))
### Optimisation of the selection of a SWM:
# SWM for 'Y' (based on the best forward-selected subset of MEM variables):
modsel.Y <- listw.select(Y, candidates, method = "FWD", MEM.autocor = "positive",
                         p.adjust = TRUE)

names(candidates)[modsel.Y$best.id]                 # Best SWM selected
modsel.Y$candidates$Pvalue[modsel.Y$best.id]       # Adjusted p-value of the global model
modsel.Y$candidates$N.var[modsel.Y$best.id]        # Nb of forward-selected MEM variables
modsel.Y$candidates$R2Adj.select[modsel.Y$best.id] # Adjusted R2 of the selected MEM var.
```

```
# SWM for 'env' (method = "global" for the optimisation, as all MEM variables are required
# to use MSR):
modsel.env <- listw.select(env, candidates, method = "global", MEM.autocor = "positive",
                           p.adjust = TRUE)

names(candidates)[modsel.env$best.id]                  # Best SWM selected
modsel.env$candidates$Pvalue[modsel.env$best.id]      # Adjusted p-value of the global model
modsel.env$candidates$N.var[modsel.env$best.id]       # Nb of forward-selected MEM variables
modsel.env$candidates$R2Adj.select[modsel.env$best.id] # Adjusted R2 of the selected MEM var.

### We perform the variation partitioning:
# Subset of selected MEM variables within the best SWM:
MEM.spe <- modsel.Y$best$MEM.select

VP <- varpart(Y, env, MEM.spe)
plot(VP)

# Test of the shared space-environment fraction (fraction [b]):
SSEF.test <- envspace.test(Y, env, coord, MEM.spe,
                           listw.env = candidates[[modsel.env$best.id]],
                           regular = FALSE, nperm = 999)
SSEF.test

# The SSEF is highly significant, indicating a potential induced spatial dependence.
}

## End(Not run)
```

`forward.sel` *Forward selection with multivariate Y using permutation under reducel model*

Description

Performs a forward selection by permutation of residuals under reduced model. Y can be multivariate.

Usage

```
forward.sel(Y, X, K = nrow(X) - 1, R2thresh = 0.99,
  adjR2thresh = 0.99, nperm = 999, R2more = 0.001, alpha = 0.05,
  Xscale = TRUE, Ycenter = TRUE, Yscale = FALSE, verbose = TRUE)
```

Arguments

Y	Response data matrix with n rows and m columns containing quantitative variables
X	Explanatory data matrix with n rows and p columns containing quantitative variables

K	Maximum number of variables to be selected. The default is one minus the number of rows
R2thresh	Stop the forward selection procedure if the R-square of the model exceeds the stated value. This parameter can vary from 0.001 to 1
adjR2thresh	Stop the forward selection procedure if the adjusted R-square of the model exceeds the stated value. This parameter can take any value (positive or negative) smaller than 1
nperm	The number of permutation to be used.The default setting is 999 permutation.
R2more	Stop the forward selection procedure if the difference in model R-square with the previous step is lower than R2more. The default setting is 0.001
alpha	Significance level. Stop the forward selection procedure if the p-value of a variable is higher than alpha. The default is 0.05 is TRUE
Xscale	Standardize the variables in table X to variance 1. The default setting is TRUE
Ycenter	Center the variables in table Y. The default setting is TRUE
Yscale	Standardize the variables in table Y to variance 1. The default setting is FALSE.
verbose	If 'TRUE' more diagnostics are printed. The default setting is TRUE

Details

The forward selection will stop when either K, R2tresh, adjR2tresh, alpha and R2more has its parameter reached.

Value

A dataframe with:

variables	The names of the variables
order	The order of the selection of the variables
R2	The R2 of the variable selected
R2Cum	The cumulative R2 of the variables selected
AdjR2Cum	The cumulative adjusted R2 of the variables selected
F	The F statistic
pval	The P-value statistic

Note

Not yet implemented for CCA (weighted regression) and with covariables.

Author(s)

Stephane Dray <stephane.dray@univ-lyon1.fr>

References

Canoco manual p.49

Examples

```
x <- matrix(rnorm(30),10,3)
y <- matrix(rnorm(50),10,5)

forward.sel(y,x,nperm=99, alpha = 0.5)
```

`forward.sel.par` *Parametric forward selection of explanatory variables in regression and RDA*

Description

If Y is univariate, this function implements FS in regression. If Y is multivariate, this function implements FS using the F-test described by Miller and Farr (1971). This test requires that (i) the Y variables be standardized, and (ii) the error in the response variables be normally distributed (to be verified by the user).

Usage

```
forward.sel.par(Y, X, alpha = 0.05, K = nrow(X) - 1, R2thresh = 0.99,
  R2more = 0.001, adjR2thresh = 0.99, Yscale = FALSE,
  verbose = TRUE)
```

Arguments

`Y`	Response data matrix with n rows and m columns containing quantitative variables
`X`	Explanatory data matrix with n rows and p columns containing quantitative variables
`alpha`	Significance level. Stop the forward selection procedure if the p-value of a variable is higher than alpha. The default is 0.05
`K`	Maximum number of variables to be selected. The default is one minus the number of rows
`R2thresh`	Stop the forward selection procedure if the R-square of the model exceeds the stated value. This parameter can vary from 0.001 to 1
`R2more`	Stop the forward selection procedure if the difference in model R-square with the previous step is lower than R2more. The default setting is 0.001
`adjR2thresh`	Stop the forward selection procedure if the adjusted R-square of the model exceeds the stated value. This parameter can take any value (positive or negative) smaller than 1
`Yscale`	Standardize the variables in table Y to variance 1. The default setting is FALSE. The setting is automatically changed to TRUE if Y contains more than one variable. This is a validity condition for the parametric test of significance (Miller and Farr 1971)
`verbose`	If 'TRUE' more diagnostics are printed. The default setting is TRUE

Details

The forward selection will stop when either K, R2tresh, adjR2tresh, alpha and R2more has its parameter reached.

Value

A dataframe with:

variables	The names of the variables
order	The order of the selection of the variables
R2	The R2 of the variable selected
R2Cum	The cumulative R2 of the variables selected
AdjR2Cum	The cumulative adjusted R2 of the variables selected
F	The F statistic
pval	The P-value statistic

Author(s)

Pierre Legendre <pierre.legendre@umontreal.ca> and Guillaume Blanchet

References

Miller, J. K. & S. D. Farr. 1971. Bimultivariate redundancy: a comprehensive measure of interbattery relationship. *Multivariate Behavioral Research*, **6**, 313–324.

Examples

```
x <- matrix(rnorm(30),10,3)
y <- matrix(rnorm(50),10,5)

forward.sel.par(y,x, alpha = 0.5)
```

---

give.thresh *Compute the maximum distance of the minimum spanning tree based on a distance matrix*

Description

It is used to select a truncation value for the dbMEM approach. It returns the minimum value that keep all samples connected.

Usage

```
give.thresh(matdist)
```

Arguments

matdist	A distance matrix (class dist or matrix)

Value

The maximum distance in the minimum spanning tree.

Author(s)

Stéphane Dray <stephane.dray@univ-lyon1.fr>

Examples

```
xy <- matrix(rnorm(60),30,2)
dxy <- dist(xy)
th <- give.thresh(dxy)
```

---

global.rtest *Global and local tests*

Description

These two Monte Carlo tests are used to assess the existence of 'global' and 'local' spatial structures, corresponding respectively to positive and negative Moran's I .

Usage

```
global.rtest(X, listw, k = 1, nperm = 499)
```

Arguments

X	a data matrix, with variables in columns
listw	a list of weights of class listw. Can be obtained easily using the function chooseCN.
k	integer: the number of highest R^2 summed to form the test statistics
nperm	integer: the number of randomisations to be performed.

Details

They rely on the decomposition of a data matrix X into global and local components using multiple regression on Moran's Eigenvector Maps (MEMs). They require a data matrix (X) and a list of weights derived from a connection network. X is regressed onto global MEMs (U+) in the global test and on local ones (U-) in the local test. One mean R^2 is obtained for each MEM, the k highest being summed to form the test statistic.

The reference distribution of these statistics are obtained by randomly permuting the rows of X.

These tests were originally part of the adegenet package for R.

Value

An object of class `randtest`.

Author(s)

Thibaut Jombart <t.jombart@imperial.ac.uk>

References

Jombart, T., Devillard, S., Dufour, A.-B. and Pontier, D. 2008. Revealing cryptic spatial patterns in genetic variability by a new multivariate method. *Heredity*, 101, 92–103. doi: 10.1038/hdy.2008.34.

Examples

```
# wait for a generic dataset
```

`LCBD.comp` *Compute LCBD from any D matrix*

Description

Compute LCBD indices (Legendre & De Cáceres 2013) from a symmetric dissimilarity matrix (D) or from a beta component matrix (Repl, RichDiff or AbDiff, or Nes) (Legendre 2014).

Usage

```
LCBD.comp(D, sqrt.D = TRUE, save.D = FALSE)
```

Arguments

`D`	A dissimilarity or beta diversity component matrix, class `dist` or `matrix`.
`sqrt.D`	Take the square root of the dissimilarities in matrix D before computing the LCBD indices.
`save.D`	If `save.D` is `TRUE`, the dissimilarity matrix will appear in the output list.

Details

Use `sqrt.D = TRUE` when computing LCBD indices for most of the replacement and richness/abundance difference indices computed by function `beta.div.comp`, as well as for the corresponding D matrices. See Table S1.4 in Appendix S1 of Legendre (2014) to identify the matrices that are Euclidean without taking the square root of the individual values. Only the RichDiffS (for presence-absence data) and AbDiff abundance data) of the Sørensen group in the Podani family have that property. In all other cases, use `sqrt.D = TRUE`.

When computing LCBD from a D matrix, use `sqrt = TRUE` if the D matrix is not Euclidean. The Euclidean property can be checked with function `is.euclid` of `ade4`.

BDtotal statistics are comparable among data sets having the same or different numbers of sampling units (n), provided that the sampling units are of the same size or represent the same sampling effort and that BDtotal is computed with the same D index.

Function `LCBD.comp` produces the same (SStotal, BDtotal, LCBD) results as function `beta.div`. Note, however, that the latter produces other interesting results (p.LCBD, SCBD). Function `LCBD.comp` should then only be used to compute LCBD indices from dissimilarity matrices that cannot be computed by function `beta.div`, e.g. genetic D matrices, or from replacement and richness difference matrices produced by function `beta.div.comp`. Significance of the LCBD indices cannot be tested when their calculation starts from a D matrix because the testing procedure involves permutation of the columns of raw data.

Value

A list containing the following results:

- `beta`: Total sum of squares and total beta diversity [= Var(Y)] of the data matrix.
- `LCBD`: Vector of Local contributions to beta diversity (LCBD) for the sites.
- `D`: The input dissimilarity matrix, class `dist`; only if `save.D=TRUE`

.

Author(s)

Pierre Legendre <pierre.legendre@umontreal.ca>

References

Legendre, P. 2014. Interpreting the replacement and richness difference components of beta diversity. Global Ecology and Biogeography 23: 1324-1334.

Legendre, P. & M. De Cáceres. 2013. Beta diversity as the variance of community data: dissimilarity coefficients and partitioning. Ecology Letters 16: 951-963.

Examples

```
### Example 1
### Compute the Hellinger distance, then the LCBD indices.
if(require("vegan", quietly = TRUE)){
data(mite)
mite.hel = decostand(mite, "hellinger")
mite.D = dist(mite.hel)
out.mite.D = LCBD.comp(mite.D, sqrt.D=FALSE)
}

### Example 2
if(require("ade4", quietly = TRUE) & require("adegraphics", quietly = TRUE)){
data(doubs)
fish.sp = doubs$fish[-8,]   # Fish data; site 8 is removed because no fish were caught
```

```
out.comp = beta.div.comp(fish.sp, coef="S", quant=TRUE)

out.fish.D = LCBD.comp(out.comp$D, sqrt.D=TRUE)   # out.comp.D is not Euclidean
out.fish.D$beta
out.fish.Repl = LCBD.comp(out.comp$repl, sqrt.D=TRUE)   # out.comp$repl is not Euclidean
out.fish.Repl$beta
out.fish.AbDiff = LCBD.comp(out.comp$rich, sqrt.D=FALSE)   # out.comp$rich is Euclidean
out.fish.AbDiff$beta

### Plot maps of the LCBD indices
fish.xy = doubs$xy[-8,]  # Geographic coordinates; site 8 removed because no fish were caught

# Map of LCBD indices for %difference dissimilarity
s.value(fish.xy, out.fish.D$LCBD, method="size", symbol = "circle",
col = c("white", "brown"), main = "Doubs fish LCBD, %difference D")

# Map of LCBD indices for replacement component of %difference dissimilarity
s.value(fish.xy, out.fish.Repl$LCBD, method="size", symbol = "circle",
col = c("white", "brown"), main = "Doubs fish replacement LCBD")

# Map of LCBD indices for abundance difference component of %difference dissimilarity
s.value(fish.xy, out.fish.AbDiff$LCBD, method="size", symbol = "circle",
col = c("white", "brown"), main = "Doubs fish abundance diff. LCBD")
}

## Not run:
### Example 3
### This example requires packages \code{"betapart"} and \code{"ade4"} for data.
### For the Baselga-family indices, the same partitioning results are obtained using
### (1) beta.div.comp or (2) beta.pair.abund() of \code{"betapart"} and LCBD.comp()

data(doubs)   # Data available in \code{"ade4"}
fish.sp = doubs$fish[-8,]
# Fish data; site 8 is removed because no fish were caught
# We use abundance data in this example, not presence-absence data

# Partition into Baselga-family replacement and nestedness components
# using \code{"beta.div.comp"} with the percentage difference index (aka Bray-Curtis)
out.comp = beta.div.comp(fish.sp, coef="BS", quant=TRUE)
out.comp$part

# Compute the D and component matrices using \code{"beta.pair.abund"}
out3 = beta.pair.abund(fish.sp, index.family = "bray")
summary(out3)

is.euclid(out3$beta.bray)    # D matrix out3$beta.bray is not Euclidean
out3.D = LCBD.comp(out3$beta.bray, sqrt.D=TRUE)
out3.D$beta
# Compare BDtotal here to BDtotal in out.comp$part (above)

out3.Repl = LCBD.comp(out3$beta.bray.bal, sqrt.D=TRUE)
out3.Repl$beta
```

```
# Compare BDtotal here to RichDiff in out.comp$part (above)

out3.AbDiff = LCBD.comp(out3$beta.bray.gra, sqrt.D=TRUE)
out3.AbDiff$beta
# Compare BDtotal here to RichDiff/Nes in out.comp$part (above)

## End(Not run)
```

`listw.candidates` *Function to create a list of spatial weighting matrices*

Description

This function is a user-friendly way to create a list of one or several spatial weighting matrices (SWM) by selecting a set of predefined connectivity and weighting matrices (B and A matrices, respectively).

Usage

```
listw.candidates(coord, style = "B", nb = c("del", "gab", "rel", "mst",
  "pcnm", "dnear"), d1 = 0, d2, weights = c("binary", "flin", "fup",
  "fdown"), y_fdown = 5, y_fup = 0.5)
```

Arguments

`coord`	Vector, matrix, or dataframe of point coordinates
`style`	Coding scheme style (see `nb2listw` of the `spdep` package). Can take values 'W', 'B', 'C', 'U', 'minmax', and 'S'; default is 'B'
`nb`	Defines how the B matrix (connectivity) is build: • `del` Delaunay triangulation • `gab` Gabriel's graph • `rel` Relative neighbourhood graph • `mst` Minimum spanning tree • `pcnm` Distance-based SWM based on the principal coordinates of neighbour matrices (PCNM) criteria (see 'Details') • `dnear` Distance-based
`d1`	Only considered if `nb = "dnear"`. A single value defining the distance beyond which two sites are connected (i.e., minimum distance between two neighbor sites). The default value is 0 (no constraint on the min distance). `d1` must be smaller than `d2`
`d2`	Only considered if `nb = "dnear"`. It defines the connectivity distance threshold below which two sites are connected (i.e., maximum distance between two neighbors. It can either be a single value or a vector of values, in which case a different SWM will be generated for each threshold value. The default value is the minimum distance keeping all points connected (i.e., the largest edge of the minimum spanning tree)

weights	Defines how the A matrix (weighths) is build: • binary without weights • flin Linear weighting function • fdown Concave-down weighting function(see Details below) • fup Concave-up weighting function (see Details below)
y_fdown	Single value or vector of values of the y parameter in the concave-down weighting function; default is 5
y_fup	Single value or vector of values of the y parameter in the concave-up weighting function; default is 0.5

Details

The function allows constructing SWMs based on any combination of B and A matrices. The B matrices are either graph-based or distance-based. The function proposes the Delaunay triangulation, Gabriel graph, relative neighbourhood graph, and the minimum spanning tree criteria to build a graph-based B matrix. Distance-based SWMs can be built with the principal coordinates of neighbour matrices (PCNM; Borcard and Legendre 2002) criteria (see details below), or using another threshold distance to define the connected site pairs. The A matrix can be based on a binary, linear, concave-down, or concave-up function. The linear, concave-down, and concave-up weighting functions are defined by $1 - (D/dmax)$, $1 - (D/dmax)^y$, and $1/D^y$, respectively, where D is the euclidean distance between the two sites considered, dmax is the maximum euclidean distance between two sites, and y is a user-defined parametre that can either be a single value or a vector of values. The choice nb = "pcnm" consists in constructing a distance-based SWM based on the largest edge of the minimum spanning tree as a connectivity distance threshold, and then by weighting the links by the function $1 - (D/(4 * t))^2$, where D is the euclidean distance between the sites, and t is the distance threshold below which two sites are considered connected (Dray et al. 2006). As optimizing the choice of a SWM has to be done with a p-value correction depending on the number of candidate SWMs tested (see function listw.select), Bauman et al. (2018) strongly encouraged plotting the concave-down and concave-up weighting functions with several parametre values in order to only choose the realistic ones to build the candidate W matrices (e.g., ranging between 0.1 and 1 for the concave-up function, as values over 1 would make no ecological sense). First visualizing the connectivity schemes with the listw.explore function may also help choosing the B matrices to select for the listw.candidates function.

Spatial eigenvectors can be generated from any candidate SWM obtained by listw.candidates using scores.listw, or can be generated and tested (recommended option for real data analysis) using mem.select. If several SWMs were created, the selection of an optimized SWM can be made using listw.select.

Value

A list of SWMs. Each element of the list was built by nb2listw (package spdep) and therefore is of class listw and nb. The name of each element of the list (SWM) is composed of the corresponding B and A matrices, followed (if any) by the y parameter value of the weighting function.

Author(s)

David Bauman (<dbauman@ulb.ac.be> or <davbauman@gmail.com>) and Stéphane Dray

References

Bauman D., Fortin M-J., Drouet T. and Dray S. (2018) Optimizing the choice of a spatial weighting matrix in eigenvector-based methods. Ecology

Borcard D. and Legendre P. (2002) All-scale spatial analysis of ecological data by means of principal coordinates of neighbour matrices. Ecological Modelling, 153, 51–68

Dray S., Legendre P. and Peres-Neto P. R. (2006) Spatial modeling: a comprehensive framework for principal coordinate analysis of neighbor matrices (PCNM). Ecological Modelling, 196, 483–493

See Also

listw.explore, scores.listw, mem.select, listw.select

Examples

```
### Create 100 random sampling locations in a squared grid of 120 x 120:
xy <- matrix(nrow = 100, ncol = 2)
xy[, 1] <- sample(c(1:120), 100, replace = FALSE)
xy[, 2] <- sample(c(1:120), 100, replace = FALSE)
### The function listw.candidates is used to build the spatial weighting matrices that
### we want to test and compare (with the listw.select function). We test a Gabriel's graph,
### a minimum spanning tree, and a distance-based connectivity defined by a threshold
### distance corresponding to the smallest distance keeping all sites connected (i.e.,
### the defaut value of d2). These connectivity matrices are then either not weighted
### (binary weighting), or weighted by the linearly decreasing function:
candidates <- listw.candidates(coord = xy, nb = c("gab", "mst", "dnear"),
                               weights = c("binary", "flin"))
names(candidates)
plot(candidates[[1]], xy)
plot(candidates[[3]], xy)
### Construction of a different list of spatial weighting matrices. This time, the
### connexions are defined by a distance-based criterion based on the same threshold
### value, but the connections are weighted by the concave-down function with a y parameter
### varying between 2 and 5, and a concave-up function with a y parametre of 0.2.
candidates2 <- listw.candidates(coord = xy, nb = "dnear", weights = c("fdown", "fup"),
                                y_fdown = 1:5, y_fup = 0.2)
### Number of spatial weighting matrices generated:
length(candidates2)
### A single SWM can also easily be generated with listw.candidates:
lw <- listw.candidates(xy, nb = "gab", weights = "bin")
plot(lw[[1]], xy)

### Generating MEM variables from an object of listw.candidates with scores.listw:
MEM <- scores.listw(lw[[1]])
### See functions mem.select and listw.select for examples of how to use an object
### created by listw.candidates with these functions.
```

listw.explore — *Interactive tool to generate R code that creates a spatial weighting matrix*

Description

Interactive tool to generate R code that creates a spatial weighting matrix

Usage

```
listw.explore()
```

Author(s)

Stéphane Dray <stephane.dray@univ-lyon1.fr>

See Also

chooseCN

Examples

```
## Not run:
## a matrix or an object of class 'Spatial*' should be in the global environment
xy <- matrix(rnorm(50), 25)
listw.explore()

## End(Not run)
```

listw.select — *Function to optimize the selection of a spatial weighting matrix and select the best subset of eigenvectors (MEM, Moran's Eigenvector Maps)*

Description

`listw.select` computes MEM variables (i.e., eigenvectors of a doubly centered spatial weighting matrix) for various definitions of spatial weighting matrices (SWM) and optimizes the selection of the SWM and of a subset of MEM variables. The optimization is done by maximizing the adjusted R-squared (R2) or by minimizing the residual spatial autocorrelation. The function controls the type I error rate by accounting for the number of tests performed. This function combine calls to the functions `scores.listw` and `mem.select`. The list of candidate SWMs can easily be generated using `listw.candidates`.

Usage

```
listw.select(x, candidates, MEM.autocor = c("positive", "negative",
  "all"), method = c("FWD", "MIR", "global"), MEM.all = FALSE,
  nperm = 999, nperm.global = 9999, alpha = 0.05, p.adjust = TRUE,
  verbose = FALSE)
```

Arguments

x	Vector, matrix, or dataframe of the response variable(s)
candidates	A list of SWMs of the class listw; candidates can be created by listw.candidates
MEM.autocor	Sign of the spatial eigenvectors to generate; "positive", "negative", or "all", for positively, negatively autocorrelated eigenvectors, or both, respectively; default is "positive"
method	Criterion to select the best subset of MEM variables. Either forward (default option), "MIR" (for univariate x only), or "global" (see Details)
MEM.all	A logical indicating if the complete set of MEM variables for the best model should be returned
nperm	Number of permutations to perform the tests in the selection procedure; Default is 999
nperm.global	Number of permutations to perform the tests in the global test; Default is 9999
alpha	Significance threshold value for the tests; Default is 0.05
p.adjust	A logical indicating wheter the p-value of the global test performed on each SWM should be corrected for multiple tests (TRUE) or not (FALSE); default is TRUE
verbose	If 'TRUE' more diagnostics are printed. The default setting is FALSE

Details

While the selection of the SWM is the most critical step of the spatial eigenvector-based methods (Dray et al. 2006), Bauman et al. (2018) showed that optimizing the choice of the SWM led to inflated type I error rates if an explicit control of the number of SWMs tested was not applied. The function listw.select therefore applies a Sidak correction (Sidak 1967) for multiple tests to the p-value of the global test of each SWM (i.e., the model integrating the whole set of spatial predictors). The Sidak correction is computed as: $P_corrected = 1 - (1 - P)^n$, where n is the number of tests performed, P is the observed p-value, and $P_corrected$ is the new p-value after the correction. The p-value is first computed using nperm permutations and then corrected according to the total number of SWMs tested (if p.adjust = TRUE). Although the function can be run without this correction, using the default value is strongly recommended to avoid inflated type I error rates (Bauman et al. 2018).

As a consequence of the p-value correction, the significance threshold decreases as the number of SWMs increases, hence leading to a trade-off between the gain of accuracy and the power loss.

The optimization criterion of the SWM performed by listw.select is either based on the maximization of the significant adjusted R2 of all the generated spatial eigenvectors (also referred to as spatial predictors or MEM variables) (method = "global"), or is based on an optimized subset of eigenvectors (method = "FWD" and "MIR").

If the objective is only to optimize the selection of the SWM, without the intervention of the selection of a subset of predictors within each SWM (`method = "global"`), then the best SWM is the one maximizing the significant adjusted global R2, that is, the R2 of the model of `x` against the whole set of generated MEM variables which must be significant for the global test (`method = "global"`).

The optimization of the SWM depends on the choosen `method`. See `mem.select` for a description of the situations in which `method = "FWD"`, `"MIR"`, and `"global"` should be preferred.

If a subset of MEM variables is needed, then the optimization of the subset of spatial predictors guides the optimization of the selection of SWM (`method = "FWD"` or `"MIR"`). If `method = "FWD"`, `listw.select` performs the forward selection on the significant SWMs and selects among these the SWM for which the forward-selected subset of spatial eigenvectors yields the highest adjusted R2. If `method = "MIR"`, `listw.select` performs the MIR selection on all the significant candidate SWMs, and selects the best SWM as the one with the smallest number of MIR-selected spatial eigenvectors. If two or more SWMs present the same smallest number of predictors, then the selection is made among them on the basis of the residual Moran's I. If `MEM.autocor = "all"`, the optimization criteria described above are applied on the sum of the adjusted R2 or number of selected spatial eigenvectors, for `method = "FWD"` and `"MIR"`, respectively. If no subset of MEM variable is required, then the optimization of the SWM is based on the maximization of the adjusted R2 of all the generated MEM variables (`method = "global"`).

If `MEM.autocor = "all"`, n-1 MEM variables are generated. In this case, if `method = "global"` or `method = "FWD"`, the adjusted R2 is computed separately on the MEM associated to positive and negative eigenvalues (hereafter positive and negative MEM variables, respectively), and the SWM yielding the highest sum of the the two significant R2 values is selected. If `method = "MIR"`, the MIR selection is performed separately on the positive and negative MEM variables, and the SWM is selected based on the sum of the number of positive and negative spatial predictors.

Value

`listw.select` returns a list that contains:

candidates A data.frame that summarizes the results on all SWMs

best.id The index and name of the best SWM

best The results for the best SWM as returned by `mem.select`

Author(s)

Bauman David (<dbauman@ulb.ac.be> or <davbauman@gmail.com>) and Stéphane Dray

References

Bauman D., Fortin M-J, Drouet T. and Dray S. (2018) Optimizing the choice of a spatial weighting matrix in eigenvector-based methods. Ecology

Blanchet G., Legendre P. and Borcard D. (2008) Forward selection of explanatory variables. Ecology, 89(9), 2623–2632

Dray S., Legendre P. and Peres-Neto P. R. (2006) Spatial modeling: a comprehensive framework for principal coordinate analysis of neighbor matrices (PCNM). Ecological Modelling, 196, 483–493

Sidak Z. (1967) Rectangular confidence regions for the means of multivariate normal distributions. Journal of the American Statistical Association, 62(318), 626–633

See Also

listw.candidates, mem.select, scores.listw

Examples

```
if(require(spdep)) {
### Create a grid of 15 x 15:
grid <- expand.grid(x = seq(1, 15, 1), y = seq(1, 15, 1))
### Generate a response variable Y structured at broad scale by linear combination of
### the first three MEM variables to which a normal noise is added:
nb <- cell2nb(nrow = 15, ncol = 15, "queen")
lw <- nb2listw(nb, style = "B")
MEM <- scores.listw(lw, MEM.autocor = "positive")
# Degree of spatial autocorrelation:
intensity <- 0.8
Y_space <- scale(MEM[, 1] + MEM[, 2] + MEM[, 3]) * intensity
Y_noise <- scale(rnorm(n = nrow(MEM), mean = 0, sd = 1)) * (1 - intensity)
Y <- Y_space + Y_noise
### Y is sampled in 100 randomly-chosen sites of the grid:
idx.sample <- sample(c(1:nrow(grid)), 100, replace = FALSE)
xy <- grid[idx.sample, ]
Y_sampled <- Y[idx.sample]
### The function listw.candidates is used to build the spatial weighting matrices that
### we want to test and compare (with the listw.select function). We test a Gabriel's graph,
### a minimum spanning tree, and a distance-based connectivity defined by a threshold
### distance corresponding to the smallest distance keeping all sites connected (i.e.,
### the defaut value of d2; see help of function listw.candidates).
### These connectivity matrices are then either not weighted (binary weighting), or
### weighted by the linearly decreasing function (see help of the function listw.candidates):
candidates <- listw.candidates(coord = xy, nb = c("gab", "mst"), weights = c("binary", "flin"))
### Number of candidate W matrices generated:
nbw <- length(candidates)
### Significance threshold value after p-value correction (Sidak correction):
1 - (1 - 0.05)^(1/nbw)
### Optimization of the selection of the SWM among the candidates generated above,
### using the corrected significance threshold calculated above for the global tests:
W_sel <- listw.select(Y_sampled, candidates, MEM.autocor = "positive", method = "FWD",
                   p.adjust = TRUE, nperm = 299)
### Some characteristics of the best spatial model:
# Best SWM:
W_sel$best.id
# Selected subset of spatial predictor within the best SWM:
W_sel$best$MEM.select
nrow(W_sel$best$summary)
# Corrected p-value of the global test of the best SWM:
W_sel$best$global.test$Pvalue
# Adjusted R2 of the subset of spatial predictors selected within the chosen SWM:
max(W_sel$best$summary$R2Adj)
# p-values of all the tested W matrices:
W_sel$candidates$Pvalue
# Adjusted R2 of the subset of spatial predictors selected for all the significant
# W matrices:
```

```
W_sel$candidates$R2Adj.select

# See Appendix S3 of Bauman et al. 2018 for more extensive examples and illustrations.
}
```

`mastigouche` *Mastigouche Lake network data set*

Description

A list containing the XY coordinates of the lakes and a site-by-edge matrix describing how 42 lakes influence each other. The influence is defined by 66 edges.

Usage

```
data(mastigouche)
```

Format

A list that includes the centred coordinates of 42 lakes in the Mastigouche reserve and a site-by-edge matrix describing how the 42 lakes are influenced among each other. The influence is defined by 66 edges.

References

Magnan, P., Rodriguez, M.A., Legendre, P., Lacasse, S. (1994) Dietary variation in a freshwater fish species: relative contribution of biotic interactions, abiotic factors, and spatial structure. *Canadian Journal of Fisheries and Aquatic Sciences* 51, 2856-2865. Blanchet F.G., P. Legendre, and Borcard D. (2008) Modelling directional spatial processes in ecological data. *Ecological Modelling*, 215, 325-336.

`mem.select` *Selection of the best subset of spatial eigenvectors (MEM, Moran's Eigenvector Maps)*

Description

`mem.select` computes the spatial eigenvectors (MEM) of the spatial weighting matrix (SWM) provided (`listw`) and optimizes the selection of a subset of MEM variables relative to response variable(s) stored in `x`. The optimization is done either by maximizing the adjusted R-squared (R2) of all (`method = "global"`) or a subset (`method = "FWD"`) of MEM variables or by minimizing the residual spatial autocorrelation (`method = "MIR"`) (see details in Bauman et al. 2018a).

Usage

```
mem.select(x, listw, MEM.autocor = c("positive", "negative", "all"),
  method = c("FWD", "MIR", "global"), MEM.all = FALSE, nperm = 999,
  nperm.global = 9999, alpha = 0.05, verbose = FALSE, ...)
```

Arguments

x	A vector, matrix, or dataframe of response variable(s). The `method = "MIR"` is only implemented for a vector response. Note that x can also contain the residuals of a model when other (e.g., environmental) variables should be considered in the model.
listw	A spatial weighting matrix of class `listw`; can be created with functions of the package spdep, or with the user-friendly function `listw.candidates`. Note that, the function `listw.candidates` returns a `list` of `listw` and subselection by `[[]]` should be performed in this case (see `Example`)
MEM.autocor	Sign of the spatial eigenvectors to generate; "positive", "negative", or "all", for positively, negatively autocorrelated eigenvectors, or both, respectively; default is "positive"
method	Criterion to select the best subset of MEM variables. Either `forward` (default option), `"MIR"` (for univariate x only), or `"global"` (see `Details`)
MEM.all	A logical indicating if the complete set of MEM variables should be returned
nperm	Number of permutations to perform the tests in the selection procedure; Default is 999
nperm.global	Number of permutations to perform the tests in the global test; Default is 9999
alpha	Significance threshold value for the tests; Default is 0.05
verbose	If 'TRUE' more diagnostics are printed. The default setting is FALSE
...	Other parameters (for internal use with `listw.select`)

Details

The function provides three different methods to select a subset of MEM variables. For all methods, a global test is firstly performed. If `MEM.autocor = "all"`, two global tests are performed and p-values are corrected for multiple comparison (Sidak correction).

If the MEM variables are to be further used in a model including actual predictors (e.g. environmental), then a subset of spatial eigenvectors needs to be selected before proceeding to further analyses to avoid model overfitting and/or a loss of statistical power to detect the contribution of the environment to the variability of the response data (Griffith 2003, Dray et al. 2006, Blanchet et al. 2008, Peres-Neto and Legendre 2010, Diniz-Filho et al. 2012). Although several eigenvector selection approaches have been proposed to select a best subset of eigenvectors, Bauman et al. (2018b) showed that two main procedures should be preferred, depending on the underlying objective: the forward selection with double stopping criterion (Blanchet et al. 2008; `method = "FWD"`) or the minimization of the residual spatial autocorrelation (Griffith and Peres-Neto 2006; MIR selection in Bauman et al. 2018a,b, `method = "MIR"`). The most powerful and accurate selection method, in terms of R2 estimation, is the forward selection. This method should be preferred when the objective is to capture as accurately as possible the spatial patterns of x. If the objective is to optimize the detection of the spatial patterns in the residuals of a model of the response variable(s) against a set of

environmental predictors, for instance, then x should be the model residuals, and method = "FWD". This allows optimizing the detection of residual spatial patterns once the effect of the environmental predictors has been removed. If however the objective is only to remove the spatial autocorrelation from the residuals of a model of x against a set of actual predictors (e.g. environmental) with a small number of spatial predictors, then accuracy is not as important and one should focus mainly on the number of spatial predictors (Bauman et al. 2018b). In this case, method = "MIR" is more adapted, as it has the advantage to maintain the standard errors of the actual predictor coefficients as low as possible. Note that method = "MIR" can only be used for a univariate x, as the Moran's I is a univariate index. If x is multivariate, then the best criterion is the forward selection (see Bauman et al. 2018b). A third option is to not perform any selection of MEM variables (method = "global"). This option may be interesting when the complete set of MEM variables will be used, like in Moran spectral randomizations (Wagner and Dray 2015, Bauman et al. 2018c) or when using smoothed MEM (Munoz 2009).

For method = "MIR", the global test consists in computing the Moran's I of x (e.g. residuals of the model of the response variable against environmental variables) and tests it by permutation (results stored in global.test). If the Moran's I is significant, the function performs a selection procedure that searches among the set of generated spatial predictors the one that best minimizes the value of the Moran's I. A model of x against the selected eigenvector is built, and the significance of the Moran's I of the model residuals is tested again. The procedure goes on until the Moran's I of the model residuals is not significant anymore, hence the name of Minimization of moran's I in the Residuals (MIR).

For method = "global" and method = "FWD", the global test consists in computing the adjusted global R2, that is, the R2 of the model of x against the whole set of generated MEM variables and tests it by permutation (results stored in global.test).

For method = "global", if the adjusted global R2 is significant, the functions returns the whole set of generated MEM variables in MEM.select.

For method = "FWD", if the adjusted global R2 is significant, the function performs a forward selection with double stopping criterion that searches among the set of generated spatial predictors the one that best maximizes the R2 of the model. The procedure is repeated untill one of the two stopping criterion is reached (see Blanchet et al. 2008). Note that in a few cases, the forward selection does not select any variable even though the global model is significant. This can happen for example when a single variable has a strong relation with the response variable(s), because the integration of the variable alone yields an adjusted R2 slightly higher than the global adjusted R2. In this case, we recommend checking that this is indeed the reason why the first selected variable was rejected, and rerun the analysis with a second stopping criterion equal to the global adjusted R2 plus a small amount allowing avoiding this issue (e.g. 5 done through the argument adjR2thresh of function forward.sel, until the solution is implemented in mem.select.

For the method = "FWD" and method = "MIR", the MEM selected by the procedure are returned in MEM.select and a summary of the results is provided in summary. If no MEM are selected, then MEM.select and summary are not returned.

Value

The function returns a list with:

global.test An object of class randtest containing the result of the global test associated to all MEM (adjusted R2 and p-value). If MEM.autocor = "all", a list with two elements (positive

and negative) corresponding to the results of the global tests performed on positive and negative MEM respectively.

MEM.all An object of class orthobasisSp containing the complete set of generated MEM variables (generated by scores.listw). Only returned if MEM.all = TRUE.

summary A dataframe summarizing the results of the selection procedure

MEM.select An object of class orthobasisSp containing the subset of significant MEM variables.

Author(s)

David Bauman (<dbauman@ulb.ac.be> or <davbauman@gmail.com>) and Stéphane Dray

References

Bauman D., Fortin M-J, Drouet T. and Dray S. (2018a) Optimizing the choice of a spatial weighting matrix in eigenvector-based methods. Ecology

Bauman D., Drouet T., Dray S. and Vleminckx J. (2018b) Disentangling good from bad practices in the selection of spatial or phylogenetic eigenvectors. Ecography, 41, 1–12

Bauman D., Vleminckx J., Hardy O., Drouet T. (2018c) Testing and interpreting the shared space-environment fraction in variation partitioning analyses of ecological data. Oikos

Blanchet G., Legendre P. and Borcard D. (2008) Forward selection of explanatory variables. Ecology, 89(9), 2623–2632

Diniz-Filho J.A.F., Bini L.M., Rangel T.F., Morales-Castilla I. et al. (2012) On the selection of phylogenetic eigenvectors for ecological analyses. Ecography, 35, 239–249

Dray S., Legendre P. and Peres-Neto P. R. (2006) Spatial modeling: a comprehensive framework for principal coordinate analysis of neighbor matrices (PCNM). Ecological Modelling, 196, 483–493

Griffith D. (2003) Spatial autocorrelation and spatial filtering: gaining understanding through theory and scientific visualization. Springer, Berlin

Griffith D. and Peres-Neto P. (2006) Spatial modeling in Ecology: the flexibility of eigenfunction spatial analyses. Ecology, 87, 2603–2613

Munoz, F. 2009. Distance-based eigenvector maps (DBEM) to analyse metapopulation structure with irregular sampling. Ecological Modelling, 220, 2683—2689

Peres-Neto P. and Legendre P. (2010) Estimating and controlling for spatial structure in the study of ecological communities. Global Ecology and Biogeography, 19, 174–184

Wagner H., Dray S. (2015). Generating spatially constrained null models for irregularly spaced data using Moran spectral randomization methods. Methods in Ecology and Evolution, 6, 1169–1178

See Also

listw.candidates, listw.select, link{scores.listw}

Examples

```
if(require(vegan)){
# Illustration of the MIR selection on the oribatid mite data
# (Borcard et al. 1992, 1994 for details on the dataset):
# *******************************************************
```

```
# Community data (response matrix):
data(mite)
# We will compute the example on a single species:
spe <- mite[, 2]
# Environmental explanatory dataset:
data(mite.env)
# We only use two numerical explanatory variables:
env <- mite.env[, 1:2]
dim(env)
# Coordinates of the 70 sites:
data(mite.xy)
coord <- mite.xy
# We build the model we are interested in:
mod <- lm(spe ~ ., data = env)


# In order to avoid possible type I error rate inflation issues, we check
# whether the model residuals are independent, and if they are spatially
# autocorrelated, we select a small subset of MEM variables to add to the
# model as covariables with the MIR selection:

# 1) We build a spatial weighting matrix based on Gabriel graph with a
# weighting function decreasing linearly with the distance:
w <- listw.candidates(coord, nb = "gab", weights = "flin")


# 2) We test the spatial autocorrelation of the model residuals and, if
# necessary, select a subset of spatial predictors:
y <- residuals(mod)
MEM <- mem.select(x = y, listw = w[[1]], method = "MIR", MEM.autocor = "positive",
         nperm = 999, alpha = 0.05)
dim(MEM$MEM.select)
# The residuals of the model presented spatial autocorrelation. The selection
# of MEM variables is thus performed to remove residual autocorrelation.

# 3) We can reconstruct our model adding the selected MEM variable as covariables:
env2 <- cbind(env, MEM$MEM.select)
mod_complete <- lm(spe ~ ., data = env2)
summary(mod_complete)$coefficient[, 1]   # Coefficient estimates
summary(mod_complete)$coefficient[, 2]   # Standard errors
}
```

---

mfpa *Multi-frequential periodogram analysis*

Description

This function performs multi-frequential periodogram analysis for univariate temporal or spatial data series collected with equal intervals. Compared with the traditional periodogram used in spec-

tral analysis, this method can detect overlapping signals with fractional frequencies. Fitting a joint polynomial-trigonometric model is achieved by Ordinary Least Squares (OLS) regression. The function also performs autocorrelation analysis of OLS residuals up to a number of lags determined by the user.

Usage

```
mfpa(y, MaxNFreq = 2, MinFreq = 3, MaxFreq = NA, ntrend = 0,
  nlags = 0, alpha = 0.05)

## S3 method for class 'mfpa'
plot(x, xlab = "", ylab = "Values", ...)

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

Arguments

`y`	Vector of *n* observations (vector of integer or real numbers, or one-column matrix).
`MaxNFreq`	Maximum number of frequencies to be estimated in the stepwise procedure (e.g. 2).
`MinFreq`	Minimum value for frequency estimates (e.g. 3.0).
`MaxFreq`	Maximum value for frequency estimates (e.g. 10). Must be larger than MinFreq and smaller than half of the number of observations in the series. If unspecified by the user (`MaxFreq=NA`), `MaxFreq` is set to *n*/4 by the function.
`ntrend`	Number (0 to 3) of orthogonal polynomial components estimating the broad-scale trend, to be included in the joint polynomial-trigonometric model. Use 0 to estimate no trend component, only an intercept.
`nlags`	Number of lags to be used for autocorrelation analysis of OLS residuals. Use 0 to bypass this analysis.
`alpha`	Significance threshold for including frequencies.
`x`	An object of class `mfpa`
`xlab, ylab`	Labels for x and y axes
`...`	Further arguments passed to or from other methods

Details

The fitting of a joint polynomial-trigonometric model is limited to ordinary least squares (OLS), with autocorrelation analysis of OLS residuals up to a certain lag. Orthogonal polynomials are used to model broad-scale trends, whereas cosines and sines model the periodic structures at intermediate scales. See Dutilleul (2011, section 6.5) and Legendre & Legendre (2012, section 12.4.4) for details. OLS regression could be replaced by an *estimated generalized least squares* (EGLS) procedure, as described in Dutilleul (2011).

In spectral analysis in general and in mfpa in particular, the cosines and sines are considered jointly in the search for the dominant frequency components since they are both required to fully account

for a frequency component in a linear model. So, when either the cosine or the sine is significant, this is sufficient indication that a significant frequency component has been found. But see the first paragraph of the 'Recommendations to users' below.

The periodic phenomenon corresponding to each identified frequency is modelled by a cosine and a sine. The first pair ('cos 1', 'sin 1') corresponds to the first frequency, the second pair to the second frequency, and so on. An intercept is also computed, as well as a polynomial broad-scale trend if argument ntrend > 0. The coefficients shown for each periodic component ('cos' and 'sin') are the OLS regression coefficients. The tests of significance producing the p-values (called 'prob' in the output file) are 2-tailed parametric t-tests, as in standard OLS regression.

A global R-square statistic for the periodogram is computed as the variance of the fitted values divided by the variance of the data series. An R-squared corresponding to each frequency is also returned.

In the Dutilleul periodogram, the time unit is the length of the data series (in time units: seconds, hours, days, etc.). Hence, the *frequency* identified by a Dutilleul periodogram is the number of cycles of the periodic signal (how many full or partial cycles) along the time series. That number is an integer when the series contains an integer number of cycles; it may also be a real number when the number of cycles is fractional. The periodogram can identify several periodic phenomena with different frequencies. The estimated frequencies could be divided by an appropriate constant to produce numbers of cycles per second or day, or per meter or km, depending on the study.

To find the *period* (number of days, hours, etc.) of the process generating a periodic signal in the data, divide the length of the series (in days, hours, etc.) by the frequency identified by Dutilleul's periodogram. Recommendations to users The mfpa code estimates the periodic frequencies to be included in the model through a combination of a stepwise procedure and non-linear optimisation. Following that, the contributions of the 'cos' and 'sin' components of all frequencies in the model are estimated by multiple linear regression in the presence of the intercept and trends (if any). Because the mfpa method estimates fractional frequencies, the cos-sin combinations are not orthogonal among the identified frequencies, and unnecessary frequencies may be selected as 'significant'.

1. It is important that users of this periodogram have hypotheses in mind about the frequencies of the processes that may be operating on the system under study and the number of periodic components they are expecting to find. If one asks for more components than the number of periodic phenomena at work on the system, the 'real' frequency usually has a strong or fairly strong R-squared and it is followed by other components with very small R-squared. Selection of frequencies of interest should thus be based more upon examination of the R-squares of the components rather than on the p-values. For short series in particular, the adjusted R-squared is an unbiased estimate of the variance of the data explained by the model. Even series of random numbers can produce 'significant' frequencies for periodic components; the associated (adjusted) R-squares will, however, be very small.

2. Function mfpa cannot detect frequencies < 1 (smaller than one cycle in the series) or larger than (n–1) where n is the number of observations in the series, the latter case corresponding to periods smaller than the interval between successive observations. When a periodic component with such a period is present in the data, Dutilleul's periodogram can detect harmonics of that frequency. Recommendation: when a frequency is detected that does not seem to correspond to a hypothesized process, one could check, using simulated data, if it could be produced by a process operating at a temporal scale (period) smaller than the interval between successive observations. An example is shown in Example 2.

3. When analysing a time series with unknown periodic structure, it is recommended to try first with more than one frequency, say 2 or 3, and also with a trend. Eliminate the non-significant

components, step by step, in successive runs, starting with the trend(s), then eliminate the weakly significant periodic components, until there are only highly significant components left in the model.

Value

A list containing the following elements:

- `frequencies`: Vector of estimated frequencies of the model periodic components and associated R-squared. The frequencies are numbers of cycles in the whole (temporal or spatial) series under study.
- `coefficients`: Data frame containing OLS slope estimates, starting with the intercept, then the orthogonal polynomials modelling trend in increasing order, followed by the cosine and sine coefficients (alternating) in the order of the estimated frequencies. Columns: (1) `coefficient`: the OLS intercept or slope estimates; (2) `prob`: the associated probabilities.
- `predicted`: A vector (length n) of predicted response values (fitted values), including the trend if any. The data and predicted values can be plotted together using function `plot.mfpa`; type plot(name.of.output.object). The data values are represented by red circles and the fitted values by a black line.
- `auto_coeff`: If nlags > 0: data frame containing the following columns. (1) `lag`: lags at which autocorrelation analysis of the OLS residuals is performed; (2) `auto_r`: vector of sample autocorrelation coefficients calculated from OLS residuals for each lag; (3) `prob`: vector of probabilities associated with the tests of significance of the sample autocorrelation coefficients.
- `y`: the original data series (one-column matrix).
- `X`: the matrix of explanatory variables; it contains a column of "1" to estimate the intercept, a column for each of the trend components (if any), and two columns for each frequency component, each frequency being represented by a cosine and a sine.
- `r.squared.global`: The global R-squared of the model and the adjusted R-squared.

Author(s)

Guillaume Larocque <glaroc@gmail.com> and Pierre Legendre.

References

Dutilleul, P. 1990. Apport en analyse spectrale d'un périodogramme modifié et modélisation des séries chronologiques avec répétitions en vue de leur comparaison en fréquence. Doctoral Dissertation, Université Catholique de Louvain, Louvain-la-Neuve, Belgium.

Dutilleul, P. 1998. Incorporating scale in ecological experiments: data analysis. Pp. 387-425 in: D. L. Peterson & V. T. Parker [eds.] Ecological scale – Theory and applications. Columbia University Press, New York.

Dutilleul, P. 2001. Multi-frequential periodogram analysis and the detection of periodic components in time series. Commun. Stat. - Theory Methods 30, 1063–1098.

Dutilleul, P. R. L. 2011. Spatio-temporal heterogeneity — Concepts and analyses. Cambridge University Press, Cambridge.

Dutilleul, P. and C. Till. 1992. Evidence of periodicities related to climate and planetary behaviors in ring-width chronologies of Atlas cedar (Cedrus atlantica) in Morocco. Can. J. For. Res. 22: 1469-1482.

Legendre, P. and P. Dutilleul. 1992. Introduction to the analysis of periodic phenomena. 11-25 in: M. A. Ali [ed.] Rhythms in fishes. NATO ASI Series, Vol. A-236. Plenum, New York.

Legendre, P. and L. Legendre. 2012. Numerical Ecology. 3rd English edition. Elsevier, Amsterdam.

Examples

```
### Example 1

# Simulate data with frequencies 2.3 and 6.1 and a random component, n = 100.
# No trend, no autocorrelated residuals.

y <- as.matrix(0.4*(sin(2.3*2*pi*(1:100)/100)) +
0.4*(sin(6.1*2*pi*(1:100)/100)) + 0.2*rnorm(100))

res <- mfpa(y, MaxNFreq = 2, MinFreq = 2, ntrend = 0, nlags = 0)

# Compute the periods associated with the two periodic components. Each
# frequency in element $frequencies is a number of cycles in the whole series.
# The periods are expressed in numbers of time intervals of the data series. In
# this example, if the data are measured every min, the periods are in min.

periods <- 100/res$frequencies$frequency

# Draw the data series and the fitted (or predicted) values

plot(res)

### Example 2

# Generate hourly periodic data with tide signal (tide period T = 12.42 h)
# during 1 year, hence 24*365 = 8760 hourly data. See
# https://en.wikipedia.org/wiki/Tide.

# In this simulation, constant (c = 0) puts the maximum value of the cosine at
# midnight on the first day of the series.

periodic.component <- function(x, T, c) cos((2*pi/T)*(x+c))

tide.h <- periodic.component(1:8760, 12.42, 0)

# The number of tides in the series is: 8760/12.42 = 705.314 tidal cycles
# during one year.

# Sample the hourly data series once a day at 12:00 noon every day. The
# periodic signal to be detected has a period smaller then the interval between
# consecutive observations and its frequency is larger than (n1). The sequence
# of sampling hours for the tide.h data is:
```

```
h.noon <- seq(12, 8760, 24)
tide.data <- tide.h[h.noon]
length(tide.data)

# The series contains 365 sampling units

# Compute Dutilleul's multi-frequential periodogram

res.noon <- mfpa(tide.data, MaxNFreq = 1, MinFreq = 2, ntrend = 1, nlags = 2)

# Examine the frequency detected by the periodogram, element
# res.noon$frequencies. This is a harmonic of the tide signal in the original
# data series tide.h.

# Compute the period of the signal in the data series sampled hourly:

period <- 365/res.noon$frequencies$frequency

# Draw the data series and the adjusted values

plot(res.noon)

# Repeat this analysis after addition of random noise to the tide data

tide.noise <- tide.data + rnorm(365, 0, 0.25)

res.noise <- mfpa(tide.noise, MaxNFreq = 1, MinFreq = 2, ntrend = 1, nlags = 2)

plot(res.noise)
```

`moran.bounds` *Function to compute extreme values of Moran's I*

Description

This function computes the upper and lower bounds of Moran's I for a given spatial weighting matrix (stored in a `listw` object). These values are obtained by the eigendecomposition of the spatial weighting matrix.

Usage

```
moran.bounds(listw)
```

Arguments

`listw`	an object of class `listw`

Value

A vector containing the maximum and minimum of Moran's I for a given spatial weighting matrix value returned

Author(s)

Stéphane Dray <stephane.dray@univ-lyon1.fr>

References

de Jong, P., Sprenger, C., & van Veen, F. (1984). On extreme values of Moran's I and Geary's C. Geographical Analysis, 16(1), 17–24.

See Also

mem nb2listw

Examples

```
if(require("ade4", quietly = TRUE)){
 if(require("spdep", quietly = TRUE)){
     data(oribatid)
     nbtri <- tri2nb(as.matrix(oribatid$xy))
     lwB <- nb2listw(nbtri, style = "B")
     lwW <- nb2listw(nbtri, style = "W")
     scB <- mem(lwB)
     scW <- mem(lwW)
     moran.bounds(lwB)
     moran.mc(scB[,1], lwB, 9)
     moran.mc(scB[,69], lwB, 9)
     moran.bounds(lwW)
     moran.mc(scW[,1], lwW, 9)
     moran.mc(scW[,69], lwW, 9)
 }
}
```

---

moran.randtest *Function to compute Moran's index of spatial autocorrelation*

Description

This function computes Moran's I statistic and provide a testing procedure using random permutations. It is based on the moran.mc function of the spdep package. The moran.randtest is slightly different as it allows to consider several variables (x can have more than one columns) and its ouputs are objects of class randtest (one variable) or krandtest (several variables).

Usage

```
moran.randtest(x, listw, nrepet = 999, ...)
```

Arguments

x	a vector, matrix or data.frame with numeric data
listw	an object of class listw created for example by nb2listw
nrepet	an integer indicating the number of permutations used in the randomization procedure
...	other arguments to be passed to the as.randtest or codeas.krandtest functions.

Value

An object of class randtest (one variable) or krandtest (several variables)

Author(s)

Stéphane Dray <stephane.dray@univ-lyon1.fr>

References

Moran, P. A. P. (1950). Notes on continuous stochastic phenomena. Biometrika, 37, 17–23.

See Also

moran.mc

Examples

```
if(require("ade4", quietly = TRUE)  & require("spdep", quiet = TRUE)){
data(mafragh)
tests <- moran.randtest(mafragh$env, nb2listw(mafragh$nb))
tests
plot(tests)

}
```

`moranNP.randtest` *Function to compute positive and negative parts of Moran's index of spatial autocorrelation*

Description

This function computes positive and negative parts of Moran's I statistic and provide a testing procedure using random permutations. The functions compute the Moran's eigenvector maps (MEM) and eigenvalues for the `listw` object. If `alter = "greater"`, the statistic 'I+' is computed as the sum of the products between positive eigenvalues and squared correlations between `x` and associated MEMs. If `alter = "less"`, the statistic 'I-' is computed as the sum of the products between negative eigenvalues and squared correlations between `x` and associated MEMs. If `alter = "two-sided"`, both statistics are computed.

Usage

```
moranNP.randtest(x, listw, nrepet = 999, alter = c("greater", "less",
  "two-sided"), ...)
```

Arguments

`x`	a `vector` with numeric data
`listw`	an object of class `listw` created for example by `nb2listw`
`nrepet`	an integer indicating the number of permutations used in the randomization procedure
`alter`	a character string specifying the alternative hypothesis, must be one of "greater" (default), "less" or "two-sided"
`...`	other arguments (e.g., `p.adjust.method`) to be passed to the codeas.krandtest function.

Value

An object of class `randtest` (for unilateral test) or `krandtest` (for bilateral test)

Author(s)

Stéphane Dray `<stephane.dray@univ-lyon1.fr>`

References

Dray, S. (2011). A new perspective about Moran's coefficient: spatial autocorrelation as a linear regression problem. Geographical Analysis, 43, 127–141.

See Also

`moran.randtest`

Examples

```
if(require("ade4", quietly = TRUE)  & require("spdep", quiet = TRUE)){
data(mafragh)
tests <- moranNP.randtest(mafragh$env[,1], nb2listw(mafragh$nb),
 alter = "two-sided", p.adjust.method = "holm")
tests
moran.randtest(mafragh$env[,1], nb2listw(mafragh$nb))$obs
sum(tests$obs)
}
```

---

mspa *Multi-Scale Pattern Analysis*

Description

The multi-scale pattern analysis (MSPA, Jombart et al 2009) investigates the main scales of spatial variation in a multivariate dataset. This implementation allows one to perform a MSPA using any multivariate analysis (stored as a dudi object), and a list of spatial weights (class listw) or an object of class orthobasisSp.

Usage

```
mspa(dudi, lwORorthobasisSp, nblocks, scannf = TRUE, nf = 2,
  centring = c("param", "sim"), nperm = 999)

## S3 method for class 'mspa'
scatter(x, xax = 1, yax = 2, posieig = "topleft",
  bary = TRUE, plot = TRUE, storeData = TRUE, pos = -1, ...)

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

Arguments

dudi	a duality diagram (i.e. a reduced space ordination) obtained by a dudi function (for instance dudi.pca).
lwORorthobasisSp	either a list of weights (class listw) that san be obtained easily using the function chooseCN or an object of class orthobasisSp
nblocks	an integer indicating the number of blocks to divide MEMs.
scannf	logical, indicating whether the screeplot should be displayed to choose the number or retained factors.
nf	the number of retained factors
centring	a character string indicating if parametric ("param") or non-parametric ("sim") centring should be used

nperm	an integer giving the number of permutations used to compute the theoretical coefficients of determination (999 by default); used if centring="sim".
x	a mspa object.
xax	an integer indicating the x axis to be displayed.
yax	an integer indicating the y axis to be displayed.
posieig	a character indicating the position of the screeplot (any of the four combination between "top", "bottom", "left" and "right").
bary	a logical indicating whether the barycenter of the variables should be displayed.
plot	a logical indicating if the graphics is displayed
storeData	a logical indicating if the data should be stored in the returned object. If FALSE, only the names of the data arguments are stored
pos	an integer indicating the position of the environment where the data are stored, relative to the environment where the function is called. Useful only if storeData is FALSE
...	additional graphical parameters (see 'adegpar' and 'trellis.par.get')

Details

The scatter method is used for plotting the results. Compared to the original version of the method, this new implementation allows to specify a number of blocks (nblocks). In this case, the multiscale decomposition is performed by dividing MEMs into several blocks and summing R2 values. This could facilitate the interpretation of results.

Value

An object having the classes mspa and dudi: mspa objects are dudi objects with the following extra slots:
- ls: principal components of the MSPA. These are the coordinates of variables onto principal axes, to be used for plotting. Correspond to matrix **B** in Appendix A of Jombart et al (2009).
- R2: matrix of R2 between variables and MEMs. Corresponds to **S** in Jombart et al (2009).
- meanPoint: coordinates of the 'mean variable' onto principal axes. The 'mean variable' is an hypothetic variable whose scale profile is the average of those of all variables of the analysed dataset.
- varweights: the weights of variables. Corresponds to **d** in Jombart et al. (2009).

Author(s)

Thibaut Jombart <t.jombart@imperial.ac.uk>

References

Jombart T, Dray S, and Dufour, A-B. (2009) Finding essential scales of spatial variation in ecological data: a multivariate approach. *Ecography* **32**: 161-168.

See Also

chooseCN to obtain a list of spatial weights.

Examples

```
####################################
### using oribatib mites dataset ###
####################################

if(require("ade4", quietly = TRUE)){
## load data
data(oribatid)

## get the list of spatial weights
cn <- chooseCN(oribatid$xy, res = "listw", ask = FALSE, type = 1)

## Hellinger transformation
hellTrans <- function(X){
  if (!( is.matrix(X) | is.data.frame(X) )) stop("Object is not a matrix.")
  if (any(is.na(X))) stop("na entries in table.")

  sumRow <- apply(X,1,sum)
  Y <- X/sumRow
  Y <- sqrt(Y)

  return(Y)
}


## ENVIRONMENTAL VARIABLES ##
## Hill and Smith analysis for environmental variables
## (for a mixture of quantitative / qualitative variables)
hsEnv <- dudi.hillsmith(oribatid$envir,scannf=FALSE)

## detrending of the analysis (residuals of regression onto xy coordinates)
hsEnv.detr <- pcaivortho(hsEnv,oribatid$xy,scannf=FALSE)

## MSPA of the detrended analysis
mspaEnv <- mspa(hsEnv.detr,cn,scannf=FALSE,nf=2)
scatter(mspaEnv)



## SPECIES DATA ##
## PCA of species abundances, after Hellinger transformation
pcaFau <- dudi.pca(hellTrans(oribatid$fau),scale=FALSE,scannf=FALSE)

## detrending of this PCA
pcaFau.detr <- pcaivortho(pcaFau,oribatid$xy,scannf=FALSE)

# MSPA of the detrended analysis
mspaFau <- mspa(pcaFau.detr,cn,scannf=FALSE,nf=2)
scatter(mspaFau)
```

```
## CANONICAL MSPA ##
## RDA species ~ envir
## (species abundances predicted by environment)
## note: RDA = 'PCAIV' (PCA with Instrumental Variables)
rda1 <- pcaiv(dudi=pcaFau.detr, df=oribatid$envir,scannf=FALSE,nf=2)

## canonical MSPA (species predicted by environment)
mspaCan1 <- mspa(dudi=rda1, lw=cn, scannf=FALSE, nf=2)
scatter(mspaCan1)

## same analysis, using a non-parametric centring
mspaCan1NP <- mspa(dudi=rda1, lw=cn, scannf=FALSE, nf=2,cent="sim",nper=999)
scatter(mspaCan1NP) # basically no change



## PARTIAL CANONICAL MSPA ##
## partial RDA species ~ envir
## (species abundances not predicted by environment)
rda2 <- pcaivortho(dudi=pcaFau.detr,df=oribatid$envir,scannf=FALSE,nf=2)

## partial canonical MSPA
mspaCan2 <- mspa(dudi=rda2, lw=cn, scannf=FALSE, nf=2)
scatter(mspaCan2) # nothing left
}
```

`msr` *Moran spectral randomization*

Description

This function allows to generate spatially-constrained random variables preserving the global autocorrelation (Moran's I) and the spatial structures at multiple scales. Multiscale property is defined by the power spectrum (i.e. decomposition of the variance of the original variables) on a basis of orthonormal eigenvectors (Moran's Eigenvector Maps, MEM). The function provides methods for univariate randomization, joint randomization of a group of variables while keeping within-group correlations fixed and univariate randomization with a fixed correlation between original data and randomized replicates.

Usage

```
msr(x, ...)

## Default S3 method:
msr(x, listwORorthobasis, nrepet = 99,
  method = c("pair", "triplet", "singleton"), cor.fixed, nmax = 100,
  simplify = TRUE, ...)
```

Arguments

x	For msr.default, a vector, a matrix or a data.frame with the original variables. If NCOL(x) > 1, then the joint randomization procedure that preserves the correlations among variables is used.
...	further arguments passed to or from other methods
listwORorthobasis	an object of the class listw (spatial weights) created by the functions of the **spdep** package or an object of class orthobasis
nrepet	an integer indicating the number of replicates
method	an character specifying which algorithm should be used to produce spatial replicates (see Details).
cor.fixed	if not missing, the level of correlation between the original variable and its randomized replicates
nmax	the number of trials used in the "triplet" procedure.
simplify	A logical value. If TRUE, the outputs for univariate procedures are returned in a matrix where each column corresponds to a replicate. If FALSE a list is returned.

Details

Three procedures are implemented in the function. The "pair" procedure is the more general as it can be applied in the three cases (univariate, univariate with fixed correlation and multivariate). This procedure preserves the power spectrum by pair of MEMs but not strictly the global autocorrelation level (Moran's I). The "singleton" procedure can be used for univariate and multivariate cases. It preserves strictly the global level of autocorrelation and the power spectrum. The "triplet" procedure can only be applied in the univariate case. It preserves the power spectrum by triplet of MEMs and strictly the global autocorrelation level.

Value

Either a matrix (if simplify is TRUE) or a list with randomized replicates.

Author(s)

Stephane Dray <stephane.dray@univ-lyon1.fr> and Helene H Wagner <helene.wagner@utoronto.ca>

References

Wagner, H.H. and Dray S. (2015) Generating spatially-constrained null models for irregularly spaced data using Moran spectral randomization methods. Methods in Ecology and Evolution, 6: 1169–1178. doi:10.1111/2041-210X.12407

See Also

scores.listw, nb2listw

Examples

```
library(spdep)
x1 <- matrix(rnorm(81*5), nrow = 81)
lw1 <- nb2listw(cell2nb(9, 9))

moran.mc(x1[,1], lw1, 2)$statistic

## singleton
x1.1 <- msr(x1[,1], lw1, nrepet = 9, method = "singleton")
apply(x1.1, 2, function(x) moran.mc(x, listw = lw1, nsim = 2)$statistic)

## triplet
x1.2 <- msr(x1[,1], lw1, nrepet = 9, method = "triplet")
apply(x1.2, 2, function(x) moran.mc(x, listw = lw1, nsim = 2)$statistic)

## pair
x1.3 <- msr(x1[,1], lw1, nrepet = 9, method = "pair")
apply(x1.3, 2, function(x) moran.mc(x, listw = lw1, nsim = 2)$statistic)

## pair with cor.fixed
x1.4 <- msr(x1[,1], lw1, nrepet = 9, cor.fixed = 0.5)
apply(x1.4, 2, function(x) moran.mc(x, listw = lw1, nsim = 2)$statistic)
cor(x1[,1], x1.4)

## pair preserving correlations for multivariate data
x1.5 <- msr(x1, lw1, nrepet = 9, cor.fixed = 0.5)
cor(x1)
lapply(x1.5, cor)

apply(x1, 2, function(x) moran.mc(x, listw = lw1, nsim = 2)$statistic)
apply(x1.5[[1]], 2, function(x) moran.mc(x, listw = lw1, nsim = 2)$statistic)

## singleton preserving correlations for multivariate data
x1.6 <- msr(x1, lw1, nrepet = 9, method = "singleton")
cor(x1)
lapply(x1.6, cor)

apply(x1, 2, function(x) moran.mc(x, listw = lw1, nsim = 2)$statistic)
apply(x1.6[[1]], 2, function(x) moran.mc(x, listw = lw1, nsim = 2)$statistic)
```

`msr.4thcorner` *Moran spectral randomization for fourth-corner analysis*

Description

This function allows to test fourth-corner statistics using constrained null models (for traits and/or environmental variables). If the argument `phyloORorthobasis` is specified, random traits are

phylogenetically-constrained to preserve the global autocorrelation (Moran's I) and the phylogenetic structures at multiple scales. If not, standard permutations are used. If the argument `listwORorthobasis` is specified, random environmental variables are spatially-constrained to preserve the global autocorrelation (Moran's I) and the spatial structures at multiple scales. If not, standard permutations are used. Multiscale property is defined by the power spectrum (i.e. decomposition of the variance of the original variables) on a basis of orthonormal eigenvectors (Moran's Eigenvector Maps, MEM).

Usage

```
## S3 method for class '4thcorner'
msr(x, listwORorthobasis, phyloORorthobasis,
  nrepet = x$npermut, method = c("pair", "triplet", "singleton"), ...)
```

Arguments

`x`	An object generated by the `fourthcorner` function.
`listwORorthobasis`	an object of the class `listw` (spatial weights) created by the functions of the **spdep** package or an object of class `orthobasis`
`phyloORorthobasis`	an object of the class `phylo` (phylogeny) created by the functions of the **ape** package or an object of class `orthobasis` generated by functions of **adephylo** (`me.phylo`)
`nrepet`	an `integer` indicating the number of replicates
`method`	an character specifying which algorithm should be used to produce spatial replicates (see codemsr.default).
`...`	further arguments of the codemsr.default function.

Value

An object of class `4thcorner` randomized replicates.

Author(s)

Stephane Dray <stephane.dray@univ-lyon1.fr>

References

Braga, J., Thuiller, W., ter Braak, C.J.F. and Dray, S. (submitted) A novel approach to consider phylogenetic and spatial autocorrelations when testing for trait-environment relationships.

See Also

`msr.default`, `me.phylo`

Examples

```
if(require("ade4", quietly = TRUE) & require("adephylo", quietly = TRUE)
& require("spdep", quietly = TRUE) & require("ape", quietly = TRUE)){
data(mafragh, package = "ade4")
fr1 <- fourthcorner(mafragh$env, mafragh$flo, mafragh$traits$tabQuantitative, nrepet = 49)
phy <- read.tree(text = mafragh$tre)
lw <- nb2listw(mafragh$nb)
fr1.msr <- msr(fr1, listwORorthobasis = lw, phyloORorthobasis = phy)

fr1
fr1.msr
}
```

`msr.mantelrtest` *Moran spectral randomization for Mantel test*

Description

This function allows to test the Mantel statistic using constrained null models in the presence of spatial autocorrelation. Random replicates of the second distance matrix are produced. They are spatially-constrained to preserve the global autocorrelation (Moran's I) and the spatial structures at multiple scales. Multiscale property is defined by the power spectrum (i.e. decomposition of the variance of the original variables) on a basis of orthonormal eigenvectors (Moran's Eigenvector Maps, MEM).

Usage

```
## S3 method for class 'mantelrtest'
msr(x, listwORorthobasis, nrepet = x$rep,
  method = c("pair", "triplet", "singleton"), ...)
```

Arguments

`x`	An object generated by the `mantel.randtest` function.
`listwORorthobasis`	an object of the class `listw` (spatial weights) created by the functions of the **spdep** package or an object of class `orthobasis`
`nrepet`	an `integer` indicating the number of replicates
`method`	an character specifying which algorithm should be used to produce spatial replicates (see `msr.default`).
`...`	further arguments of the `msr.default` function.

Value

An object of class `randtest`.

Author(s)

Sylvie Clappe, Stephane Dray <stephane.dray@univ-lyon1.fr>

References

Crabot, J., Clappe, S., Dray, S. and Datry, T. (submitted) Restoring the Mantel tests.

See Also

msr.default, mantel.randtest

Examples

```
if(require("ade4", quietly = TRUE)
& require("spdep", quietly = TRUE)){
data(mafragh, package = "ade4")

d1 <- dist(mafragh$env[,1:3])
d2 <- dist(mafragh$env[,7])
t1 <- mantel.randtest(d1,d2)
t1

lw <- nb2listw(mafragh$nb)
t2 <- msr(t1, listwORorthobasis = lw)
t2

}
```

---

msr.varipart *Moran spectral randomization for variation partitioning*

Description

The functions allows to evaluate the significance and estimate parts in variation partitioning using Moran Spectral Randomization (MSR) as a spatially-constrained null model to account for spatial autocorrelation in table X. Hence, this function provides a variation partioning adujsted for spurious correlation due to spatial autocorrelation in both the response and one explanatory matrix.

Usage

```
## S3 method for class 'varipart'
msr(x, listwORorthobasis, nrepet = x$test$rep[1],
  method = c("pair", "triplet", "singleton"), ...)
```

Arguments

x	An object generated by the varipart function.
listwORorthobasis	an object of the class listw (spatial weights) created by the functions of the **spdep** package or an object of class orthobasis
nrepet	an integer indicating the number of replicates
method	an character specifying which algorithm should be used to produce spatial replicates (see codemsr.default).
...	further arguments of the codemsr.default function.

Details

The function corrects the biases due to spatial autocorrelation by using MSR procedure to produce environmental predictors that preserve the spatial autocorrelation and the correlation structures of the original environmental variables while being generated independently of species distribution.

Value

An object of class varipart randomized replicates.

Author(s)

(s) Stephane Dray <stephane.dray@univ-lyon1.fr> and Sylvie Clappe <sylvie.clappe@univ-lyon1.fr>

References

Sylvie Clappe, Stephane Dray and Pedro R. Peres-Neto (in preparation) Beyond neutrality: using a null model to disentangle the effects of niche dynamics and spurious correlations in variation partitioning.

Wagner, H. H., and S. Dray, 2015. Generating spatially constrained null models for irregularly spaced data using Moran spectral randomization methods. Methods in Ecology and Evolution 6:1169–1178.

See Also

msr.default, varipart

Examples

```
library(ade4)
library(spdep)
data(mafragh)
## Performing standard variation partitioning
dudiY <- dudi.pca(mafragh$flo, scannf = FALSE, scale = FALSE)
mafragh.lw <- nb2listw(mafragh$nb)
me <- mem(mafragh.lw, MEM.autocor = "positive")
vprda <- varipart(dudiY, mafragh$env, me, type = "parametric")

## Adjust estimation and compute p-value by msr methods
```

```
vprda.msr <- msr(vprda, mafragh.lw, nrepet=99)
vprda.msr
```

`mst.nb` *Function to compute neighborhood based on the minimum spanning tree*

Description

Compute mst and returns as a nb object

Usage

```
mst.nb(dxy)
```

Arguments

dxy	A distance matrix based on spatial coordinates of samples

Value

An object of class nb

Author(s)

Stéphane Dray <stephane.dray@univ-lyon1.fr>

See Also

graph2nb, give.thresh

Examples

```
xy <- matrix(rnorm(60),30,2)
dxy <- dist(xy)
th <- give.thresh(dxy)
nb1 <- mst.nb(dxy)
nb1
wh1 <- which(as.matrix(dxy)==th,arr.ind=TRUE)
plot(nb1,xy,pch=20,cex=2,lty=3)
lines(xy[wh1[1,],1],xy[wh1[1,],2],lwd=2)
title(main="Maximum distance of the minimum spanning tree in bold")
```

multispati *Multivariate spatial analysis*

Description

This function provides a multivariate extension of the univariate method of spatial autocorrelation analysis. It provides a spatial ordination by maximizing the product of variance by spatial autocorrelation.

Usage

```
multispati(dudi, listw, scannf = TRUE, nfposi = 2, nfnega = 0)

## S3 method for class 'multispati'
summary(object, ...)

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

## S3 method for class 'multispati'
plot(x, xax = 1, yax = 2, pos = -1,
  storeData = TRUE, plot = TRUE, ...)
```

Arguments

dudi	an object of class dudi obtained by the simple analysis of a data table
listw	an object of class listw created for example by nb2listw
scannf	a logical value indicating whether the eigenvalues barplot should be displayed
nfposi	an integer indicating the number of axes with positive autocorrelation
nfnega	an integer indicating the number of axes with negative autocorrelation
...	further arguments passed to or from other methods
x, object	an object of class multispati
xax, yax	the numbers of the x-axis and the y-axis
pos	an integer indicating the position of the environment where the data are stored, relative to the environment where the function is called. Useful only if storeData is FALSE
storeData	a logical indicating if the data should be stored in the returned object. If FALSE, only the names of the data arguments are stored
plot	a logical indicating if the graphics is displayed

Details

This analysis generalizes the Wartenberg's multivariate spatial correlation analysis to various duality diagrams created by the functions (`dudi.pca`, `dudi.coa`, `dudi.acm`, `dudi.mix`...) If *dudi* is a duality diagram created by the function `dudi.pca` and *listw* gives spatial weights created by a row normalized coding scheme, the analysis is equivalent to Wartenberg's analysis.

We note X the data frame with the variables, Q the column weights matrix and D the row weights matrix associated to the duality diagram *dudi*. We note L the neighbouring weights matrix associated to *listw*. Then, the `'multispati'` analysis gives principal axes v that maximize the product of spatial autocorrelation and inertia of row scores :

$$I(XQv) * \|XQv\|^2 = v^t Q^t X^t DLXQv$$

Value

Returns an object of class `multispati`, which contains the following elements :

`eig`	a numeric vector containing the eigenvalues
`nfposi`	integer, number of kept axes associated to positive eigenvalues
`nfnega`	integer, number of kept axes associated to negative eigenvalues
`c1`	principle axes (v), data frame with p rows and (nfposi + nfnega) columns
`li`	principal components (XQv), data frame with n rows and (nfposi + nfnega) columns
`ls`	lag vector onto the principal axes (LXQv), data frame with n rows and (nfposi + nfnega) columns
`as`	principal axes of the dudi analysis (u) onto principal axes of multispati (t(u)Qv), data frame with dudi\$nf rows and (nfposi + nfnega) columns

Author(s)

Stéphane Dray <stephane.dray@univ-lyon1.fr> with contributions by Daniel Chessel, Sebastien Ollier and Thibaut Jombart

References

Dray, S., Said, S. and Debias, F. (2008) Spatial ordination of vegetation data using a generalization of Wartenberg's multivariate spatial correlation. *Journal of vegetation science*, **19**, 45–56.

Grunsky, E. C. and Agterberg, F. P. (1988) Spatial and multivariate analysis of geochemical data from metavolcanic rocks in the Ben Nevis area, Ontario. *Mathematical Geology*, **20**, 825–861.

Switzer, P. and Green, A.A. (1984) Min/max autocorrelation factors for multivariate spatial imagery. Tech. rep. 6, Stanford University.

Thioulouse, J., Chessel, D. and Champely, S. (1995) Multivariate analysis of spatial patterns: a unified approach to local and global structures. *Environmental and Ecological Statistics*, **2**, 1–14.

Wartenberg, D. E. (1985) Multivariate spatial correlation: a method for exploratory geographical analysis. *Geographical Analysis*, **17**, 263–283.

Jombart, T., Devillard, S., Dufour, A.-B. and Pontier, D. A spatially explicit multivariate method to disentangle global and local patterns of genetic variability. Submitted to *Genetics*.

See Also

dudi,mat2listw

Examples

```
if (require(spdep, quiet = TRUE) & require(ade4, quiet = TRUE)) {
    data(mafragh)
    maf.xy <- mafragh$xy
    maf.flo <- mafragh$flo
    maf.listw <- nb2listw(mafragh$nb)
    if(adegraphicsLoaded()) {
      g1 <- s.label(maf.xy, nb = mafragh$nb, plab.cex = 0.75)
    } else {
      s.label(maf.xy, neig = mafragh$neig, clab = 0.75)
    }
    maf.coa <- dudi.coa(maf.flo,scannf = FALSE)
    maf.coa.ms <- multispati(maf.coa, maf.listw, scannf = FALSE, nfposi = 2, nfnega = 2)
    maf.coa.ms

    ### detail eigenvalues components
    fgraph <- function(obj){
      # use multispati summary
      sum.obj <- summary(obj)
      # compute Imin and Imax
      Ibounds <- moran.bounds(eval(as.list(obj$call)$listw))
      Imin <- Ibounds[1]
      Imax <- Ibounds[2]
      I0 <- -1/(nrow(obj$li)-1)
      # create labels
      labels <- lapply(1:length(obj$eig),function(i) bquote(lambda[.(i)]))
      # draw the plot
      xmax <- eval(as.list(obj$call)$dudi)$eig[1]*1.1
      par(las=1)
      var <- sum.obj[,2]
      moran <- sum.obj[,3]
      plot(x=var,y=moran,type='n',xlab='Inertia',ylab="Spatial autocorrelation (I)",
           xlim=c(0,xmax),ylim=c(Imin*1.1,Imax*1.1),yaxt='n')
      text(x=var,y=moran,do.call(expression,labels))
      ytick <- c(I0,round(seq(Imin,Imax,le=5),1))
      ytlab <- as.character(round(seq(Imin,Imax,le=5),1))
      ytlab <- c(as.character(round(I0,1)),as.character(round(Imin,1)),
           ytlab[2:4],as.character(round(Imax,1)))
      axis(side=2,at=ytick,labels=ytlab)
      rect(0,Imin,xmax,Imax,lty=2)
      segments(0,I0,xmax,I0,lty=2)
      abline(v=0)
      title("Spatial and inertia components of the eigenvalues")
    }
    fgraph(maf.coa.ms)
    ## end eigenvalues details
```

```
  if(adegraphicsLoaded()) {
    g2 <- s1d.barchart(maf.coa$eig, p1d.hori = FALSE, plot = FALSE)
    g3 <- s1d.barchart(maf.coa.ms$eig, p1d.hori = FALSE, plot = FALSE)
    g4 <- s.corcircle(maf.coa.ms$as, plot = FALSE)
    G1 <- ADEgS(list(g2, g3, g4), layout = c(1, 3))
  } else {
    par(mfrow = c(1, 3))
    barplot(maf.coa$eig)
    barplot(maf.coa.ms$eig)
    s.corcircle(maf.coa.ms$as)
    par(mfrow = c(1, 1))
  }


  if(adegraphicsLoaded()) {
    g5 <- s.value(maf.xy, -maf.coa$li[, 1], plot = FALSE)
    g6 <- s.value(maf.xy, -maf.coa$li[, 2], plot = FALSE)
    g7 <- s.value(maf.xy, maf.coa.ms$li[, 1], plot = FALSE)
    g8 <- s.value(maf.xy, maf.coa.ms$li[, 2], plot = FALSE)
    G2 <- ADEgS(list(g5, g6, g7, g8), layout = c(2, 2))
  } else {
    par(mfrow = c(2, 2))
    s.value(maf.xy, -maf.coa$li[, 1])
    s.value(maf.xy, -maf.coa$li[, 2])
    s.value(maf.xy, maf.coa.ms$li[, 1])
    s.value(maf.xy, maf.coa.ms$li[, 2])
    par(mfrow = c(1, 1))
  }


  w1 <- -maf.coa$li[, 1:2]
  w1m <- apply(w1, 2, lag.listw, x = maf.listw)
  w1.ms <- maf.coa.ms$li[, 1:2]
  w1.msm <- apply(w1.ms, 2, lag.listw, x = maf.listw)
  if(adegraphicsLoaded()) {
    g9 <- s.match(w1, w1m, plab.cex = 0.75, plot = FALSE)
    g10 <- s.match(w1.ms, w1.msm, plab.cex = 0.75, plot = FALSE)
    G3 <- cbindADEg(g9, g10, plot = TRUE)
  } else {
    par(mfrow = c(1,2))
    s.match(w1, w1m, clab = 0.75)
    s.match(w1.ms, w1.msm, clab = 0.75)
    par(mfrow = c(1, 1))
  }

  maf.pca <- dudi.pca(mafragh$env, scannf = FALSE)
  multispati.randtest(maf.pca, maf.listw)
  maf.pca.ms <- multispati(maf.pca, maf.listw, scannf=FALSE)
  plot(maf.pca.ms)
}
```

ortho.AIC *Compute AIC for models with orthonormal explanatory variables*

Description

This function is now deprecated. Please try the new `mem.select` function.

Usage

```
ortho.AIC(Y, X, ord.var = FALSE)
```

Arguments

Y	A matrix with response variables (univariate or multivariate response)
X	A set of orthonormal and centered vectors
ord.var	A logical value indicating if the order of variables and cumulative R2 must be returned

Details

This function compute corrected AIC for models with orthonormal and centered explanatory variables such as MEM spatial eigenfunctions. Variables are sorted by their contribution to R2.

It ensures that a model with k variables is the best one that can be obtained. By default, response variables are centered (model with intercept).

Value

A vector with corrected AIC if `ord.var=FALSE`. A list if `ord.var=TRUE` with:

AICc	Values of corrected AIC.
AICc0	Values of corrected AIC for the null model (only intercept).
ord	Order of variables to be enter in the model
R2	Cumulative R2

Author(s)

Stéphane Dray <stephane.dray@univ-lyon1.fr>

References

Godinez-Dominguez E. and Freire J. (2003) Information-theoretic approach for selection of spatial and temporal models of community organization. Marine Ecology - Progress Series. 253, 17–24

Examples

```
y <- matrix(rnorm(50),50,1)
x <- svd(scale(y \%*\% c(0.1,0.5,2,0,0.7)+matrix(rnorm(250),50,5)))$u
res <- ortho.AIC(y,x,ord.var=TRUE)
minAIC <- which.min(res$AICc)
nvar <- length(1:minAIC)+1 # number of orthogonal vectors + 1 for intercept
lm1 <- lm(y~x[,res$ord[1:minAIC]])
summary(lm1)$r.squared # R2
res$R2[minAIC] # the same
min(res$AICc) # corrected AIC
extractAIC(lm1) # classical AIC
min(res$AICc)-2*(nvar*(nvar+1))/(nrow(x)-nvar-1) # the same

lm2 <- lm(y~1)

res$AICc0 # corrected AIC for the null model
extractAIC(lm2) # classical AIC
res$AICc0-2*(1*(1+1))/(nrow(x)-1-1) # the same
```

`orthobasis.poly` *Function to compute polynomial of geographical coordinates*

Description

This function computes orthgonal polynomials of geographical coordinates. Polynomials functions are orthogonal and centred for the weights defined in `wt` (i.e., orthogonal to `wt`). It is the classical approach to perform trend surface analysis.

Usage

```
orthobasis.poly(coords, degree = 2, wt = rep(1/nrow(coords),
  nrow(coords)))
```

Arguments

`coords`	either a `Spatial*` object or a `matrix` with geographic coordinates
`degree`	the degree of the polynomial
`wt`	a vector of weights. It is used to orthogonalize the polynomial functions

Value

an object of class `orthobasisSp` , subclass `orthobasis`

Author(s)

Stéphane Dray <stephane.dray@univ-lyon1.fr>

References

Dray S., Pélissier R., Couteron P., Fortin M.J., Legendre P., Peres-Neto P.R., Bellier E., Bivand R., Blanchet F.G., De Caceres M., Dufour A.B., Heegaard E., Jombart T., Munoz F., Oksanen J., Thioulouse J., Wagner H.H. (2012). Community ecology in the age of multivariate multiscale spatial analysis. *Ecological Monographs* **82**, 257–275.

See Also

`mem` `orthobasis`

Examples

```
if(require("ade4", quietly = TRUE)){
data(mafragh, package = "ade4")
pol2 <- orthobasis.poly(mafragh$Spatial)

if(require("adegraphics", quietly = TRUE)){
plot(pol2, mafragh$Spatial)
}
}
```

`plot.orthobasisSp` *Function to display Moran's Eigenvector Maps (MEM) and other spatial orthogonal bases*

Description

This function allow to plot or map orthogonal bases

Usage

```
## S3 method for class 'orthobasisSp'
plot(x, SpORcoords, pos = -1, plot = TRUE,
  match.ID = FALSE, ...)
```

Arguments

`x`	an object of class `orthobasisSp`
`SpORcoords`	either a `Spatial*` object or a `matrix` with geographic coordinates
`pos`	an integer indicating the position of the environment where the data are stored, relative to the environment where the function is called. Useful only if 'storeData' is 'FALSE'
`plot`	a logical indicating if the graphics is displayed

match.ID	a logical indicating if names of geographic entities match rownames of the orthobasisSp object
...	additional graphical parameters (see 'adegpar' and 'trellis.par.get')

Value

an object of class ADEgS, generated by the s.Spatial function of the adegraphics package

Author(s)

Stéphane Dray <stephane.dray@univ-lyon1.fr>

See Also

s.Spatial

Examples

```
if(require("ade4", quietly = TRUE) & require("spdep", quietly = TRUE)){
data(mafragh)
me <- mem(nb2listw(mafragh$nb))

if(require("adegraphics", quietly = TRUE)){
plot(me[,1:6], mafragh$xy)
plot(me[,1:6], mafragh$Spatial)
}
}
```

---

rotation *Rotate a set of point by a certain angle*

Description

Rotate a set of XY coordinates by an angle (in radians)

Usage

```
rotation(xy, angle)
```

Arguments

xy	A 2-columns matrix or data frame containing a set of X and Y coordinates.
angle	Numeric. A scalar giving the angle at which the points should be rotated. The angle is in radians.

Value

A 2-columns matrix of the same size as xy giving the rotated coordinates.

Author(s)

F. Guillaume Blanchet

Examples

```
### Create a set of coordinates
coords<-cbind(runif(20),runif(20))

### Create a series of angles
rad<-seq(0,pi,l=20)

for(i in rad){
coords.rot<-rotation(coords,i)
plot(coords.rot)
}

### Rotate the coordinates by an angle of 90 degrees
coords.90<-rotation(coords,90*pi/180)
coords.90

plot(coords,xlim=range(rbind(coords.90,coords)[,1]),ylim=range(rbind(coords.90,coords)[,2]),asp=1)
points(coords.90,pch=19)
```

---

scalogram *Function to compute a scalogram*

Description

The function decomposes the variance of a variable x on a basis of orthogonal vectors. The significance of the associated R-squared values is tested by a randomization procedure. A smoothed scalogram is obtained by summing the R-squared values into nblocks.

Usage

```
scalogram(x, orthobasisSp, nblocks = ncol(orthobasisSp), nrepet = 999,
  p.adjust.method = "none")

## S3 method for class 'scalogram'
plot(x, pos = -1, plot = TRUE, ...)
```

Arguments

x	a numeric vector for univariate data or an object of class dudi for multivariate data (for scalogram) or an object of class scalogram (for plot.scalogram)
orthobasisSp	an object of class orthobasisSp
nblocks	an integer indicating the number of blocks in the smoothed scalogram

nrepet	an integer indicating the number of permutations used in the randomization procedure
p.adjust.method	a string indicating a method for multiple adjustment, see p.adjust.methods for possible choices.
pos	an integer indicating the position of the environment where the data are stored, relative to the environment where the function is called. Useful only if 'storeData' is 'FALSE'
plot	a logical indicating if the graphics is displayed
...	additional graphical parameters (see 'adegpar' and 'trellis.par.get')

Details

On the plot, oberved R-squared values are represent by bars. A black line indicate the 0.95 quantile of the values obtained by permutations. Significant values are indicated by a '*'

Value

The function scalogram returns an object of class scalogram, subclass krandtest. The plot function returns an object of class ADEgS, generated by the functions of the adegraphics package

Author(s)

Stéphane Dray <stephane.dray@univ-lyon1.fr>

References

Dray S., Pélissier R., Couteron P., Fortin M.J., Legendre P., Peres-Neto P.R., Bellier E., Bivand R., Blanchet F.G., De Caceres M., Dufour A.B., Heegaard E., Jombart T., Munoz F., Oksanen J., Thioulouse J., Wagner H.H. (2012). Community ecology in the age of multivariate multiscale spatial analysis. *Ecological Monographs* **82**, 257–275.

See Also

mem orthobasis

Examples

```
if(require("ade4", quietly = TRUE) & require("spdep", quietly = TRUE)){
data(mafragh)
me <- mem(nb2listw(mafragh$nb))

if(require("adegraphics", quietly = TRUE)){
sc1 <- scalogram(mafragh$env$Conduc, me, nblocks = 10)
plot(sc1)
}
}
```

scores.listw *Function to compute and manage Moran's Eigenvector Maps (MEM) of a listw object*

Description

These functions compute MEM (i.e., eigenvectors of a doubly centered spatial weighting matrix). Corresponding eigenvalues are linearly related to Moran's index of spatial autocorrelation.

Usage

```
scores.listw(listw, wt = rep(1, length(listw$neighbours)),
  MEM.autocor = c("non-null", "all", "positive", "negative"),
  store.listw = FALSE)

mem(listw, wt = rep(1, length(listw$neighbours)),
  MEM.autocor = c("non-null", "all", "positive", "negative"),
  store.listw = FALSE)

orthobasis.listw(listw, wt = rep(1, length(listw$neighbours)),
  MEM.autocor = c("non-null", "all", "positive", "negative"),
  store.listw = FALSE)

## S3 method for class 'orthobasisSp'
x[i, j, drop = TRUE]
```

Arguments

listw	An object of the class listw created by functions of the spdep package
wt	A vector of weights. It is used to orthogonalize the eigenvectors. It could be useful if MEM are used in weighted regression or canonical correspondence analysis
MEM.autocor	A string indicating if all MEMs must be returned or only those corresponding to non-null, positive or negative autocorrelation. The difference between options all and non-null is the following: when there are several null eigenvalues, option all removes only one of the eigenvectors with null eigenvalues and returns (n-1) eigenvectors, whereas non-null does not return any of the eigenvectors with null eigenvalues.
store.listw	A logical indicating if the spatial weighting matrix should be stored in the attribute listw of the returned object
x	An object of class orthobasisSp.
i, j	Elements to extract (integer or empty): index of rows (i) and columns (j).
drop	A logical. If TRUE, object containing only one colum is converted in vector

Details

Testing the nullity of eigenvalues is based on E(i)/E(1) where E(i) is i-th eigenvalue and E(1) is the maximum absolute value of eigenvalues

Value

An object of class `orthobasisSp` , subclass `orthobasis`. The MEMs are stored as a `data.frame`. It contains several attributes (see `?attributes`) including:

- `values`: The associated eigenvalues.
- `listw`: The associated spatial weighting matrix (if `store.listw = TRUE`).

Author(s)

Stéphane Dray <stephane.dray@univ-lyon1.fr>

References

Dray, S., Legendre, P., and Peres-Neto, P. R. (2006). Spatial modeling: a comprehensive framework for principal coordinate analysis of neighbor matrices (PCNM). *Ecological Modelling* **196**, 483–493.

Griffith D. A. (1996) Spatial autocorrelation and eigenfunctions of the geographic weights matrix accompanying geo-referenced data. *Canadian Geographer* **40**, 351–367.

See Also

`nb2listw` `orthobasis`

Examples

```
if(require("ade4", quietly = TRUE) & require("spdep", quietly = TRUE)){
data(oribatid)
nbtri <- tri2nb(as.matrix(oribatid$xy))
sc.tri <- scores.listw(nb2listw(nbtri, style = "B"))
summary(sc.tri)
}
if(require("adegraphics", quietly = TRUE)){
s.value(oribatid$xy,sc.tri[,1:9])
plot(sc.tri[,1:6], oribatid$xy, pSp.cex = 5, pSp.alpha = 0.5, pbackground.col = 'lightblue')
}
```

`stimodels` *Space-time interaction in ANOVA without replication*

Description

Function `stimodels` performs two-way ANOVA to test space-time interaction without replicates using one among a set of possible models. Function `quicksti` allows performing space-time ANOVA in a simplified way. In many models, degrees of freedom are saved by coding space and/or time parsimoniously using distance-based Moran Eigenvector Maps (dbMEM).

Usage

```
stimodels(Y, S, Ti, model = "5", nperm = 999, nS = -1, nT = -1,
  Sfixed = TRUE, Tfixed = TRUE, COD.S = NULL, COD.T = NULL,
  print.res = TRUE)

quicksti(Y, S, Ti, nperm = 999, alpha = 0.05, COD.S = NULL,
  COD.T = NULL, print.res = TRUE)
```

Arguments

`Y`	Site-by-species response data table. Assumes row blocks corresponding to times, i.e. within each block all sites are provided (in the same order).
`S`	Number of spatial points (when they are aligned on a transect or a time series and equispaced) or a matrix of spatial coordinates (when the sites are on a two-dimensional surface or on a line but very irregularly spaced).
`Ti`	Number of time campaigns (when equispaced) or a matrix (a vector) of temporal coordinates (when the time campaigns are very irregularly spaced).
`model`	Linear space-time model to be used (can be either "2", "3a", "3b", "4", "5", "6a", "6b", or "7").
`nperm`	Number of permutations in the significance tests.
`nS`	Number of space dbMEMs to use (by default, -1, all dbMEMs with positive autocorrelation are used).
`nT`	Number of time dbMEMs to use (by default, -1, all dbMEMs with positive autocorrelation are used).
`Sfixed`	Logical: is factor Space fixed, or not (if FALSE, it is considered a random factor).
`Tfixed`	Logical: is factor Time fixed, or not (if FALSE, it is considered a random factor).
`COD.S`	Spatial coding functions to be used instead of dbMEM. The number of columns must be lower than `S` and the number of rows equal to the number of rows in `Y`.
`COD.T`	Temporal coding functions to be used instead of dbMEM. The number of columns must be lower than `Ti` and the number of rows equal to the number of rows in `Y`.
`print.res`	If TRUE displays the results and additional information onscreen (recommended).
`alpha`	In `quicksti`, confidence level for the interaction test. Depending on the decision for the interaction test, the main factors are tested differently.

Details

In stimodels tests for space-time interaction and space or time main effects are conducted using one of the different models. With Models 2, 6a and 6b the interaction test is not available.

Model 2 - Space and Time are coded using Helmert contrasts for the main effects. No interaction is tested. Model 3a - Space is coded using dbMEM variables whereas Time is coded using Helmert contrasts. Model 3b - Space is coded using Helmert contrasts whereas Time is coded using dbMEM variables. Model 4 - Both Space and Time are coded using dbMEM variables for all tests. Model 5 - Space and Time are coded using Helmert contrasts for the main factor effects, but they are coded using dbMEM variables for the interaction term. Model 6a - Nested model. Testing for the existence of spatial structure (common or separate) using dbMEM variables to code for Space. Model 6b - Nested model. Testing for the existence of temporal structure (common or separate) using dbMEM variables to code for Time. Model 7 - Space and Time are coded using dbMEM variables for the main factor effects, but they are coded using Helmert contrasts for the interaction term (not recommended).

When using quicksti, space-time interaction is first tested using Model 5. Depending on the outcome of this test, the main factors are tested using different strategies. If the interaction is not significant then the test of main factors is also done following Model 5. If the interaction is significant, then a nested model (6a) is used to know whether separate spatial structures exist and another (6b) to know whether separate temporal structures exist. In quicksti function space and time are always considered fixed factors (F ratios are constructed using residual MS in the denominator).

For the interaction the permutations are unrestricted, whereas for the main factors the permutations are restricted within time blocks (for the test of factor Space) or space blocks (for the test of factor Time). By default, the function computes dbMEM for space and time coding, but other space and/or time descriptors can be provided by the user, through COD.S and COD.T.

Value

testS	An object with the result of the space effect test, including the mean squares for the F numerator (MS.num), the mean squares for the F denominator (MS.den), the proportion of explained variance (R2), the adjusted proportion of explained variance (R2.adj), the F statistics (F) and its p-value computed from a permutation test (Prob).
testT	An object with the result of the time effect test, like testS.
teststi	An object with the result of the space-time interaction test, like testS.

Author(s)

Pierre Legendre <pierre.legendre@umontreal.ca>, Miquel De Caceres and Daniel Borcard

References

Borcard, D. and P. Legendre. 2002. All-scale spatial analysis of ecological data by means of principal coordinates of neighbour matrices. Ecological Modelling 153: 51-68.

Dray, S., P. Legendre and P. R. Peres-Neto. 2006. Spatial modelling: a comprehensive framework for principal coordinate analysis of neighbour matrices (PCNM). Ecological Modelling 196: 483-493.

Legendre, P., M. De Caceres and D. Borcard. 2010. Community surveys through space and time to assess environmental changes: testing space-time interaction in the absence of replication. Ecology 91: 262-272.

See Also

trichoptera

Examples

```
data(trichoptera)

# log-transform species data (excluding site and time colums)
trich.log <- log1p(trichoptera[,-c(1,2)])


# Run space-time interaction test using model "5"
stimodels(trich.log, S=22, Ti=10, nperm=99, model="5")

# Run space-time analysis with tests for main effects after testing
# interaction (which is significant)
quicksti(trich.log, S=22, Ti=10, nperm=99)

# Run space-time analysis for time blocks number 6 and 7.
# Interaction is then not significant and tests of main effects are done
# following model 5
quicksti(trich.log[111:154,], S=22, Ti=2, nperm=99)
```

---

test.W *Function to compute and test eigenvectors of spatial weighting matrices*

Description

This function is now deprecated. Please try the new listw.candidates and listw.select functions.

Usage

```
test.W(Y, nb, xy, MEM.autocor = c("all", "positive", "negative"),
  f = NULL, ...)
```

Arguments

Y	A matrix with response variables (univariate or multivariate response)
nb	An object of the class nb created by functions of the spdep package

xy	Coordinates of the samples, this argument is optional and is required only if the argument f is not null.
MEM.autocor	A string indicating if all MEM must be returned or only those corresponding to positive or negative autocorrelation
f	A function of the distance that can be used as a weighting spatial function. This argument is optional
...	Others arguments for the function f. It defines the range of parameters which will be tested

Details

This function is a user-friendly way to compute and test eigenvectors for various definitions of spatial weighting matrices. It combines calls to the functions scores.listw and ortho.AIC. It allows to test various definitions of the spatial weighting matrix and return results of scores.listw for the best one.

This functions allows to test one binary spatial weighting matrix (if only Y and nb are provided). It allows also to test a weighting function based on distances (if f is provided) and a weighting function with different values of parameters if other arguments of f are provided.

Value

A list with the following elements:

all	A data.frame where each row correspond to one spatial weighting matrix tested. It contains value of parameteres tested and corrected AIC and number of orthogonal vectors for the best model.
best	A list containing results of scores.listw and ortho.AIC of the best spatial weighting matrix according to corrected AIC.

Author(s)

Stéphane Dray <stephane.dray@univ-lyon1.fr>

References

Dray S., Legendre P. and Peres-Neto P. R. (2006) Spatial modeling: a comprehensive framework for principal coordinate analysis of neighbor matrices (PCNM). Ecological Modelling, 196, 483–493

See Also

ortho.AIC, scores.listw

Examples

```
if(require(ade4) & require(spdep)){

data(oribatid)
# Hellinger transformation
```

```
fau <- sqrt(oribatid$fau / outer(apply(oribatid$fau, 1, sum), rep(1, ncol(oribatid$fau)), "*"))
# remove gradient effect
faudt <- resid(lm(as.matrix(fau) ~ as.matrix(oribatid$xy)))

# test a binary spatial weighting matrix
nbtri <- tri2nb(as.matrix(oribatid$xy))
tri.res <- test.W(faudt, nbtri)

maxi <- max(unlist(nbdists(nbtri, as.matrix(oribatid$xy))))

# test a simple spatial weighting function of the distance
f1 <- function(x) {1-(x)/(maxi)}
tri.f1 <- test.W(faudt, nbtri, f = f1, xy = as.matrix(oribatid$xy))

# test a spatial weighting function with various values of parameters
f2 <- function(x,dmax,y) {1-(x^y)/(dmax)^y}
tri.f2 <- test.W(faudt,nbtri, f = f2, y = 2:10, dmax = maxi, xy = as.matrix(oribatid$xy))
}
```

`trichoptera` *Trichoptera data set*

Description

A dataset containing the abundances of 56 species in 220 sites

Usage

```
data(trichoptera)
```

Format

A data frame with 220 rows and 58 variables (Site, Date, and 56 species)

`variogmultiv` *Function to compute multivariate empirical variogram*

Description

Compute a multivariate empirical variogram. It is strictly equivalent to summing univariate variograms

Usage

```
variogmultiv(Y, xy, dmin = 0, dmax = max(dist(xy)), nclass = 20)
```

Arguments

Y	A matrix with numeric data
xy	A matrix with coordinates of samples
dmin	The minimum distance value at which the variogram is computed (i.e. lower bound of the first class)
dmax	The maximum distance value at which the variogram is computed (i.e. higher bound of the last class)
nclass	Number of classes of distances

Value

A list:

d	Distances (i.e. centers of distance classes).
var	Empirical semi-variances.
n.w	Number of connections between samples for a given distance.
n.c	Number of samples used for the computation of the variogram.
dclass	Character vector with the names of the distance classes.

Author(s)

Stéphane Dray <stephane.dray@univ-lyon1.fr>

References

Wagner H. H. (2003) Spatial covariance in plant communities: integrating ordination, geostatistics, and variance testing. Ecology, 84, 1045–1057

Examples

```
if(require(ade4)){
data(oribatid)
# Hellinger transformation
fau <- sqrt(oribatid$fau / outer(apply(oribatid$fau, 1, sum), rep(1, ncol(oribatid$fau)), "*"))
# Removing linear effect
faudt <- resid(lm(as.matrix(fau) ~ as.matrix(oribatid$xy)))
mvspec <- variogmultiv(faudt, oribatid$xy, nclass = 20)
mvspec
plot(mvspec$d, mvspec$var,type = 'b', pch = 20, xlab = "Distance", ylab = "C(distance)")
}
```

`WRperiodogram` *Whittaker-Robinson periodogram*

Description

Whittaker-Robinson periodogram for univariate series of quantitative data.

Usage

```
WRperiodogram(x, T1 = 2, T2, nperm = 499, nopermute,
  mult = c("sidak", "bonferroni"), print.time = FALSE)

## S3 method for class 'WRperio'
plot(x, prog = 1, alpha = 0.05, line.col = "red",
  main = NULL, ...)
```

Arguments

`x`	A vector of quantitative values, with class `numeric`, for function `WRperiodogram`, or an output object of WRperiodogram for function `plot`.
`T1`	First period included in the calculation (default: `T1=2`).
`T2`	Last period included in the calculation (default: `T2=n/2`).
`nperm`	Number of permutations for the tests of significance.
`nopermute`	List of item numbers that should not be permuted; see Details (default: no items should be excluded from the permutations).
`mult`	Correction method for multiple testing. Choices are "bonferroni" and "sidak" (default: `mult="bonferroni"`).
`print.time`	Print the computation time. Useful when planning the analysis of a long data series with a high number of permutations. Default: `print.time=FALSE`.
`prog`	`prog=1` (default): use the original p-values in the plot. `prog=2`: use the p-values corrected for multiple testing. `prog=3`: progressive correction of the multiple tests.
`alpha`	Significance level for the plot; p-values smaller than or equal to alpha are represented by black symbols. Default: `alpha=0.05`.
`line.col`	Colour of the lines between symbols in the graph (default: `line.col="red"`).
`main`	Main title of the plot. Users can write a custom title, in quotes (default: `main="WR Periodogram"`).
`...`	Other graphical arguments passed to this function.

Details

The Whittaker-Robinson periodogram (Whittaker and Robinson, 1924) identifies periodic components in a vector of quantitative data. The data series must contain equally-spaced observations (i.e. constant lag) along a transect in space or through time. The vector may contain missing observations, represented by NA, in reasonable amount, e.g. up to a few percent of the total number of

observations. The periodogram statistic used in this function is the standard deviation of the means of the columns of the Buys-Ballot table (Enright, 1965). The method is also described in Legendre & Legendre (2012, Section 12.4.1). Missing values (NA) are handled by skipping the NA values when computing the column means of the Buys-Ballot table.

The data must be stationary before computation of the periodogram. Stationarity is violated when there is a trend in the data or when they were obtained under contrasting environmental or experimental conditions. Users should at least test for the presence of a significant linear trend in the data (using linear regression); if a significant trend is identified, it can be removed by computing regression residuals.

The `nopermute` option allows users to include a list of items numbers that should not be permuted, whether the observations are NA or zero values. This option should not be used in routine work. It is intended for special situations where observations could not be made at some points along the space or time series because that was impossible. For example, in a spatial data series along a river, if points fall on emerging rocks or on islands, no observation of phytoplankton could have been made at those points. For the permutation test, values at these positions (NA or 0) should not be permuted with values at points where observations were possible.

The graph produced by the `plot` function shows the periodogram statistics and their significance following a permutation test, with periods in the abscissa. The p-values may be corrected for multiple testing using either the Bonferroni or the Sidak correction, which can be applied to all values in the correlogram uniformly, or following a progressive correction.

A progressive correction means that for the first periodogram statistic, the p-value is tested against the alpha significance level without any correction; for the second statistic, the p-value is corrected for 2 simultaneous tests; and so forth until the k-th statistic, where the p-value is corrected for k simultaneous tests. This approach solves the problem of "where to stop interpreting a periodogram"; one goes on as long as significant values emerge, considering the fact that the tests become progressively more conservative.

In the Whittaker-Robinson periodogram, harmonics of a basic period are often found to be also significant.

The permutation tests, which can take a bit of time in very large jobs, can be interrupted by issuing an `Escape` command. One can also click the `STOP` button at the top of the R console.

Value

The function produces an object of class `WRperio` containing a table with the following columns:

`Period`	period number;
`WR.stat`	periodogram statistic;
`p-value`	p-value after permutation test;
`p.corrected`	p-value corrected for multiple tests, using the Bonferroni or Sidak method;
`p.corr.prog`	p-value after progressive correction.

When the p-values cannot be computed because of a very high proportion of missing values in the data, values of 99 are posted in the last three columns of the output table.

Author(s)

Pierre Legendre <pierre.legendre@umontreal.ca> and Guillaume Guenard

References

Enright, J. T. 1965. The search for rhythmicity in biological time-series. Journal of Theoretical Biology 8: 426-468.

Legendre, P. and L. Legendre. 2012. Numerical ecology, 3rd English edition. Elsevier Science BV, Amsterdam.

Sarrazin, J., D. Cuvelier, L. Peton, P. Legendre and P. M. Sarradin. 2014. High-resolution dynamics of a deep-sea hydrothermal mussel assemblage monitored by the EMSO-Açores MoMAR observatory. Deep-Sea Research I 90: 62-75. (Recent application in oceanography)

Whittaker, E. T. and G. Robinson. 1924. The calculus of observations – A treatise on numerical mathematics. Blackie & Son, London.

Examples

```
###
### 1. Numerical example of Subsection 12.4.1 of Legendre and Legendre (2012)

test.vec <- c(2,2,4,7,10,5,2,5,8,4,1,2,5,9,6,3)

# Periodogram with permutation tests of significance
res <- WRperiodogram(test.vec)
plot(res)

### 2. Simulated data

periodic.component <- function(x,T,c) cos((2*pi/T)*(x+c))

n <- 500   # corresponding to 125 days, 4 observations per day
# Generate a lunar cycle, 29.5 days (T=118)
moon <- periodic.component(1:n, 118, 59)
# Generate a circadian cycle (T=4)
daily <- periodic.component(1:n, 4, 0)
# Generate a tidal cycle (T=2)
tide <- periodic.component(1:n, 2, 0)

# Periodogram of the lunar component only
res.moon <- WRperiodogram(moon, nperm=0)
res.moon <- WRperiodogram(moon, T2=130, nperm=99)
par(mfrow=c(1,2))
plot(moon)
plot(res.moon, prog=1)

# Add the three components, plus a random normal error term
var <- 5*moon + daily + tide + rnorm(n, 0, 0.5)
# Draw a graph of a portion of the data series
par(mfrow=c(1,2))
plot(var[1:150], pch=".", cex=1)
lines(var[1:150])

# Periodogram of 'var'
```

```
res.var <- WRperiodogram(var, T2=130, nperm=99)
plot(res.var, prog=1, line.col="blue")
# Find position of the maximum value of this periodogram
which(res.var[,2] == max(res.var[,2]))

# Replace 10% of the 500 data by NA
select <- sort(sample(1:500)[1:50])
var.na <- var
var.na[select] <- NA
res.var.na <- WRperiodogram(var.na, T2=130, nperm=99)
plot(res.var.na, prog=1)

### 3. Data used in the examples of the documentation file of function afc() of {stats}
# Data file "ldeaths"; time series, 6 years x 12 months of deaths in UK hospitals
ld.res.perio <- WRperiodogram(ldeaths, nperm=499)
par(mfrow=c(1,2))
plot(ld.res.perio, prog=1) # Graph with no correction for multiple testing
plot(ld.res.perio, prog=3) # Graph with progressive correction
acf(ldeaths)   # acf() results, for comparison
```

Index