Package 'codep'

February 14, 2012

Type Package

Title Multiscale Codependence Analysis

Version 0.1-6

Date 2010-03-2

Author Guillaume Guenard, with contributions from Bertrand Pages

Maintainer Guillaume Guenard <guenard@cict.fr>

Description Computation of Multiscale Codependence Analysis and Moran's eigenvector maps (mem), as an additional feature.

Depends R (>= 2.0.0), stats

Suggests vegan

License GPL-2

LazyLoad yes

Revision 001

URL http://www.r-project.org

Repository CRAN

Date/Publication 2010-02-03 10:13:18

R topics documented:

`cthreshold` *Familywise type I error rate*

Description

Function to calculate the testwise type I error rate threshold that corresponds to a give familywise threshold.

Usage

```
cthreshold(alpha, nbtest)
```

Arguments

`alpha`	The familywise type I error threshold.
`nbtest`	The number of tests performed.

Details

Type I error rate inflation occurs when a single hypothesis is tested indirectly using inferences about two or more (*i.e.* a family of) sub-hypotheses. In such situation, the probability of type I error (*i.e.* the probability of incorrectly rejecting the null hypothesis) of the single, familywise, hypothesis is higher than the lowest, testwise, probabilities. As a consequence, the rejection of null hypothesis for one or more individual tests does not warrant that the correct decision (whether to reject the the null hypothesis on a familywise basis) was taken properly. This function allows to obtain correct, familywise, alpha thresholds in the context of multiple testing. It is base on the Sidak inegality.

Value

The threshold that have to be used for individual tests.

Author(s)

Guillaume Guenard, Laboratoire evolution et diversite biologique, CNRS / Universite Paul-Sabatier, Toulouse, France.

References

Sidak, Z. 1967. Rectangular Confidence Regions for Means of Multivariate Normal Distributions J. Am. Stat. Assoc. 62: 626-633

Wright, P. S. 1992. Adjusted p-values for simultaneous inference. Biometrics 48: 1005-1013

See Also

Legendre, P. and Legendre, L. 1998. Numerical Ecology. Elsevier Science B.V., Amsterdam, The Neatherlands. p. 18

Examples

```
# For a familywise threshold of 5% with 5 tests:
cthreshold(c(0.05),5)   # Threshold of each test is 0.01020622
```

`mca` *Multiscale Codependence Analysis*

Description

Functions and methods to handle Multiscale Codependence Analysis.

Usage

```
mca(y, x, memobj)
test.mca(mcaobj, alpha = 0.05, max.step = Inf)
permute.mca(mcaobj, permute = NA, alpha = 0.05, max.step = Inf)
## S3 method for class 'mca'
print(x, ...)
## S3 method for class 'mca'
summary(object, ...)
## S3 method for class 'mca'
plot(x, ...)
## S3 method for class 'mca'
fitted(object, which=NA, components=FALSE, ...)
## S3 method for class 'mca'
residuals(object, which=NA, ...)
## S3 method for class 'mca'
predict(object, which=NA, newdata=NA, components=FALSE, ...)
```

Arguments

`y`	A numeric vector containing the explained variable.
`x`	A numeric vector containing the explanatory variable for functions `test.mca` or `permute.mca` or a mca object for methods `print` and `plot`.
`memobj`	An object of `class` 'mem' obtained using `meigmap` or `mem`.
`object, mcaobj`	An object of `class` 'mca' obtained from `mca`, `test.mca`, or `permute.mca`
`alpha`	The type I (α) error threshold used by the testing procedure.
`max.step`	The maximum number of steps to perform when testing for statistical significance.
`permute`	The number of random permutations used for testing. Obtained through `minpermute` when let to `NA`.
`which`	A numeric vector of indices or character vector variable names to force-use. Mandatory if `mcaobj` is untested.
`components`	A boolean specifying whether the components of fitted or predicted values associated with single eigenvector in the map (see `mem`) should be returned.

`newdata`	A numeric vector containing new values of the explonatory variable.
`...`	Further parameters to be passed to other functions or methods (currently ignored).

Details

Multiscale Codependence Analysis (MCA) allows to calculate correlation-like (i.e.codependence) coefficients between two variables with respect to structuring variables (Moran's eigenvector maps). The analysis is performed using function mca. The purpose of this function is limited to parameter fitting. Test procedures are handled through test.mca (parametric testing) or permute.mca (permutation testing). Additionaly, methods are provided for printing, obtaining testing summary, plotting results, calculating fitted and residuals values, and making predictions.

Value

An object of `class` 'mca' is a list containing:

`data`	A copy of the explaned (`y`) and explanatory (`x`) variables that were given to `mca`.
`memobj`	The object of `class` 'mem' that was given to `mca`.
`Upyxcb`	A 4 columns matrix containing the vectors of cross-products of structuring variable (`U` see `mem`) with variable `y` and `x`, the codependence coefficients C and the coregression coefficients B.
`test`	Results of statistical testing. `NULL` if no testing was performed. The results of statistical testing is a list containing:
`permute`	The number of randomized permutations used by `permute.mca` for permutation testing. 0 or `FALSE` for parametric testing obtained using `test.mca`.
`significant`	The indices of codependence coefficient describing statistically significant codependence between `y` and `x`, in decreasing order of importance.
`test.table`	The testing table (a 4 columns matrix) with tau statistics, degrees-of-freedom, and testwise and familywise probabilities of type I (α)) error. It contains one line for each statistically significant coefficient (if any) and an additional line showing the last tested (i.e. non-significant) coefficient.
`details`	Details about permutation testing not shown in `test.table`. `NULL` for parametric testing.

The `fitted`, `residuals`, and `predict` methods return a single-column matrix of fitted, residuals, or predicted values, respectively. The `fitted` and `predict` methods return a list a list when the parameter `component` is `TRUE`. The list contains the `fitted` or `predicted` values as a first item and a matrix `components` as a second. This matrix has one column for each statistically significant codependence coefficient.

Author(s)

Guillaume Guenard, Laboratoire evolution et diversite biologique, CNRS / Universite Paul-Sabatier, Toulouse, France.

See Also

meigmap

Examples

```
#
### Exemple 1: Atlantic salmon in the St-Marguerite river, Quebec, Canada.
#
data(Salmon)
map <- meigmap(x=Salmon[,"Position"],truncation=c(0,20),weighting="1")
mca1 <- mca(y=log(Salmon[,"Abundance"]+1),x=Salmon[,"Substrate"],memobj=map)
mca1
plot(mca1)
summary(mca1)   # Works only with tested mca.
#
## Parametric test.
mcatest1 <- test.mca(mca1)
mcatest1
summary(mcatest1)
plot(mcatest1)
plot(Abundance~Position,data=Salmon, type="l")
lines(y=exp(fitted(mcatest1)-1),x=Salmon[,"Position"])
plot(x=Salmon[,"Abundance"],y=exp(fitted(mcatest1)-1),asp=1,xlim=c(0,14),ylim=c(0,14))
abline(0,1)
residuals(mcatest1)
#
## Permutation test. Warning: takes time to calculate.
mcapermute1 <- permute.mca(mca1)
mcapermute1
summary(mcapermute1)
plot(Abundance~Position,data=Salmon, type="l")
lines(y=exp(fitted(mcapermute1)-1),x=Salmon[,"Position"])
plot(x=Salmon[,"Abundance"],y=exp(fitted(mcapermute1)-1),asp=1,xlim=c(0,14),ylim=c(0,14),
     xlab="Observed abundance",ylab="Fitted abundance")
abline(0,1)
residuals(mcapermute1)
#
## Not run:
### Exemple 2: Oribatid mites in Lake Cromwell, Quebec, Canada.
#
## Requires package vegan
library(vegan)
data(mite.xy)
map <- meigmap(x=as.matrix(mite.xy),truncation=c(0,NA),weighting="f1")
data(mite)
data(mite.env)
mca2 <- mca(y=log(mite[,"LRUG"]+1),x=mite.env[,"WatrCont"],memobj=map)
mca2
mcapermute2 <- permute.mca(mca2)
summary(mcapermute2)
#
layout(matrix(1:2,1,2))
```

```
obs <- 0.5+log(mite[,"LRUG"]+1)/2
plot(y~x,data=mite.xy, asp=1, pch=21, cex=obs, bg="black",main="Observed")
fit <- 0.5+fitted(mcapermute2)/2
plot(y~x,data=mite.xy, asp=1, pch=21, cex=fit, bg="black",main="Fitted")
#
layout(1)
plot(x=log(mite[,"LRUG"]+1),y=fitted(mcapermute2),asp=1,xlim=c(0,7),ylim=c(0,7),
     xlab="Observed abundance (log(x+1))",ylab="Fitted abundance (log(x+1))")
abline(0,1)
#

## End(Not run)
```

---

meigmap *Moran's eigenvector maps*

Description

Functions and methods to handle Moran's eigenvector maps of a set of locations in a space with an arbitrary number of dimension.

Usage

```
meigmap(x, opt.coordinates=NA, truncation=c(0,NA),weighting=c("1","f1","f2","f3"), wpar=1, select=1e-
mem(coordinates, lambda, U)
## S3 method for class 'mem'
print(x, ...)
## S3 method for class 'mem'
plot(x, ...)
```

Arguments

- `x` — A set of coordinates defined in one (numeric vector) or many (a coordinate x dimension matrix) dimensions or, alternatively, a distance matrix provided by `dist`. Coordinates are treated as geographic and distances between them are taken to be Euclidean. For `print` and `plot` methods: an object of `class` 'mem'.
- `opt.coordinates` — Coordinates to be used when a distance matrix is provided as x. Used for plotting purposes.
- `truncation` — Threshold values (minimum and maximum) used to obtain the connectivity matrix. The minimum value is the value above which locations are to be considered as neighbour (in most case 0) whereas the maximum value is the value above which locations are to be considered as distant. If any trunctation value is `NA`, the minimum value is taken to be 0 and the maximum value is taken as the minimum value that allow every locations to be either directly or indirectly connected with one another.

`weighting`	The function used to obtain the weighting matrix: 1 refers to the identity function and implies that the spatial weighting matrix is equal to the connnectivity matrix, f1 is $d_{i,j} = 1 - (d_{i,j}/d_m ax)$ where $d_m ax$ is the distance between the two most distant locations in theset, f2 is $d_{i,j} = 1 - (d_{i,j}/d_m ax)^w par$, and f3 is $d_{i,j} = 1/d_{i,j}^w par$.
`wpar`	Parameter of the weighting function f2 or f3. Ignored when weighting is 1 or f1.
`select`	The smallest absolute eigenvalue for eigenfunctions to be considered as a suitable variables.
`coordinates`	A set of n coordinates for the mem object.
`lambda`	The p eigenvalues of the mem object.
`U`	A n x p matrix of structuring variables.
`...`	Further parameters to be passed to other functions or methods (currently ignored).

Details

Moran's eigenvector maps are sets of orthonormal functions (i.e. eigenfunction) obtained from the locations of observations within a structuring framework, e.g., space, time, along graph. It is obtained by the eigenvalue decomposition of a spatial weighting matrix; a centred square matrix itself obtained from a distance matrix. The spatial weighting matrix is the haddamar product of a connectivity matrix B and a weighting matrix A. The function described herein handles user-defined truncations parameters to calculate B and provides a default approach to estimate these parameters should they be missing. It also offers four different ways to compute A through parameters `weighting` and `wpar`.

Function `mem` defines a mem object customly, by providing coordinates, eigenvalues, and structuring variables as column vectors. Basin checks for size compatibility are made. It falls on user's responsability to further ensure the adequacy of the object.

The `print` method provides the number of the number of orthonormal variables (i.e. basis functions), the number of observations these functions are spanning, and their associated eigenvalues.

The `plot` method provides a plot of the eigenvalues and offers the possibility to plot the values of variables for 1- or 2-dimensional sets of coordinates. To use this later functionnality, a coordinate parameter must be provided with the actual plotting coordinates. `plot.mem` opens the default graphical device driver, i.e., `X11`, `windows`, or `quartz` and recurses through variable with a left mouse click on the graphical window. A right mouse click interrupts recursing on `X11` and `windows`. Recursing is stopped by hitting *Esc* on the `quartz` graphical device driver (Mac OS X users).

See Dray et al. (2006) for further details about this method.

Value

An object of `class` 'mem' is a list containing:

`coordinates`	A matrix of coordinates associated with
`lambda`	A vector of the eigenvalues obtain from the computation of the the Moran's eigenvector map.
`U`	A matrix of the eigenvectors defining the Moran's eigenvector map.

Author(s)

Guillaume Guenard, Laboratoire evolution et diversite biologique, CNRS / Universite Paul-Sabatier, Toulouse, France.

References

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

See Also

mca

Examples

```
#
### Exemple 1: A linear transect.
#
data(Salmon)
map <- meigmap(x=Salmon[,"Position"],truncation=c(0,20),weighting="1")
map
plot(map)
#
### Exemple 2: A unevenly sampled surface.
#
## Require package vegan
library(vegan)
data(mite.xy)
map <- meigmap(x=as.matrix(mite.xy),truncation=c(0,NA),weighting="f1")
map
plot(map)
#
### Exemple 3: Building a mem object for other basis functions (fourier transforms).
#
U <- matrix(NA,100,0)
for (i in 1:49) {
  for (j in c(1+0i, 0+1i)) {
    spc <- complex(100)
    spc[i+1] <- j
    spc[100-i+1] <- Conj(j)
    U <- cbind(U,Re(fft(spc,inverse=TRUE))/sqrt(200))
  }
}
spc <- complex(100)
spc[51] <- 1+0i
U <- cbind(U,Re(fft(spc,inverse=TRUE))/10)
dmat <- matrix(0,100,100)
for (i in 1:99) {
  dmat[i,i+1] <- 1
  dmat[i+1,i] <- 1
}
dmat[1,100] <- dmat[100,1] <- 1
```

```
dmat <- (diag(100)-(rep(0.1,100)%*%t(rep(0.1,100))))%*%dmat%*%(diag(100)-(rep(0.1,100)%*%t(rep(0.1,100))))
lambda <- diag(t(U) %*% dmat %*% U)
map <- mem(coordinates=1:100, lambda=lambda, U=U)
map
plot(map)
#
```

---

minpermute *Number of permutations for MCA*

Description

Calculate the number of permutations suitable for Multiscale Codependence Analysis.

Usage

```
minpermute(alpha,nbtest,margin=1,ru=3)
```

Arguments

`alpha`	The familywise type I error threshold allowable for the complete analysis.
`nbtest`	The number of test performed (the number of eigenvectors in the 'mem' object in the case of `mca`).
`margin`	A margin allowed for the number of permutation.
`ru`	The magnitude of the round-up to apply to the number of permutations.

Details

This function calculate the number of permutations for use with `permute.mca`. Parameter `margin` allows to apply a safe margin to the number of permutations. The minimal suitable value for this parameter is 1. Parameter `ru` allows to round-up the number of permutations. A value of 0 implies no round-up, a value of 1 a round-up to the next ten, 2 a round-up to the next hundred, and so on. Function `minpermute` is called internally by `permute.mca` in case `permute = NA`. In that case, the margin is set to 10 (`margin = 10`) and the outcome is rounded-up the next thousand (`ru = 3`). This function is meant for users that wish to apply their own margins and round-up factors to calculate the number of permutations for use with `permute.mca`.

Value

The minimum number of permutation to be used for `permute.mca`.

Author(s)

Guillaume Guenard, Laboratoire evolution et diversite biologique, CNRS / Universite Paul-Sabatier, Toulouse, France.

See Also

permute.mca

Examples

```
# For a 5% threshold under 50 tests.
minpermute(alpha = 0.05, nbtest=50)
# Allowing more margin (implies more computation time).
minpermute(alpha = 0.05, nbtest=50, margin=10, ru=3)
```

Salmon *St-Marguerite Juvenile Atlantic salmon (parr) dataset.*

Description

A 1520m transect of the St-Marguerite River, Quebec, Canada.

Usage

```
data(Salmon)
```

Format

A 76 x 5 dataframe.

Details

Contains (1) the position on the segment upstream site 'Bardsville' (48d23m01.59s N ; 70d12m10.05s W), (2) the number of Parr (age I+ and II+) observed in the segment, (3) the mean water depth estimated across the segment (m), (4) the mean current velocity estimated across the segment (m/s), and (5) The mean substrate size (mm). Water depth, current velocity, and mean substrate size were estimates in 76 segments of 20m. Sampling took place on 7 July 2002.

Author(s)

Daniel Boisclair, Departement de sciences biologiques, Universite de Montreal, Montreal, Quebec, Canada.

References

Guenard, G., Legendre, P., Boisclair, D., and Bilodeau, M. In press. Multiscale codependence analysis: an integrated approach to analyse relationships. Ecology

See Also

Bouchard, J. and Boisclair, D. 2008. The relative importance of local, lateral, and longitudinal variables on the development of habitat quality models for a river. Can. J. Fish. Aquat. Sci. 65: 61-73

Examples

```
data(Salmon)
```

Index