Package ‘rasterdiv’

July 18, 2020

Version 0.2-1

Date 2020-07-15

Title Diversity Indices for Numerical Matrices

Depends R (>= 3.6.0)

Imports raster, parallel, doParallel, methods, foreach, proxy, svMisc, pbapply, pbmcapply

Suggests knitr, rmarkdown, RColorBrewer, rasterVis, gridExtra, ggplot2

LazyData TRUE

Description Providing functions to calculate indices of diversity on numerical matrices based on information theory. The rationale behind the package is described in Rocchini, Marcantonio and Ricotta (2017) <doi:10.1016/j.ecolind.2016.07.039>.

VignetteBuilder knitr

Encoding UTF-8

License GPL (>= 2)

NeedsCompilation no

Author Matteo Marcantonio [aut, cre],
Martina Iannacito [aut, ctb],
Elisa Thouverai [aut, ctb],
Daniele Da Re [aut],
Clara Tattoni [aut],
Giovanni Bacaro [aut],
Saverio Vicario [aut, ctb],
Carlo Ricotta [aut],
Duccio Rocchini [aut, ctb]

Maintainer Matteo Marcantonio <marcantoniomatteo@gmail.com>

Repository CRAN

Date/Publication 2020-07-18 05:20:02 UTC

R topics documented:

`accRao` *Accumulation function for parametric Rao's index of quadratic entropy (Q)*

Description

AccRao computes the accumulation function (integral or area udner the curve) of the parametric version of Rao's index of quadratic entropy (Q) on different classes of numeric matrices using a moving window algorithm.

Usage

```
accRao(alphas=1:5, x, dist_m="euclidean", window=9,
  method="classic", rasterAUC=TRUE, lambda=0,
  na.tolerance=1.0, rescale=FALSE, diag=TRUE,
  simplify=2, np=1, cluster.type="SOCK", debugging=FALSE)
```

Arguments

`alphas` a continuous vector of alphas in the form start:end over which integrated the parametric Rao's index. Default value is 1:5.

`x` input data may be a matrix, a Spatial Grid Data Frame, a RasterLayer or a list of these objects. In the latter case, if `method="classic"` only the first element of the list will be considered.

`dist_m` define the type of distance to be calculated between numerical categories. `dist_m` can be a character string which defines the name of the distance to derive such as "euclidean". The distance names allowed are the same as for `proxy::dist`. Alternatively, `dist_m` can be a function which calculates an user defined distance, (i.e., `function(x,y) {return(cos(y-x)-sin(y-x))}`) or a matrix of distances. If `method="multidimension"` then only "euclidean", "manhattan", "canberra", "minkowski" and "mahalanobis" can be used. Default value is "euclidean".

window	the side of the square moving window, it must be a odd numeric value greater than 1 to ensure that the target pixel is in the center of the moving window. Default value is 3. If `proxy::dist` is a matrix then the function will assume that this is the distance matrix, and therefore no distance will be derived.
method	Currently, there are two ways to calculate the parameteric version of Rao's index. If `method="classic"`, then the normal parameteric Rao's index will be calculated on a single matrix. If `method="multidimension"` (experimental!) a list of matrices must be provided as input. In the latter case, the overall distance matrix will be calculated in a multi- or hyper-dimensional system by using the distance measure defined through the function argument `dist_m`. Each pairwise distance is then multiplied by the inverse of the squared number of pixels in the considered moving window, and the Rao's Q is finally derived by applying a summation. Default value is *"classic"*
rasterAUC	Boolean, if TRUE the output will be a RasterLayer object with *x* as a raster template.
lambda	the value of the lambda of Minkowski's distance. Considered only if `dist_m = "minkowski"` and `method="multidimension"`. `Default value is 0`.
na.tolerance	Numeric value $(0.0 - 1.0)$ which indicates the proportion of NA values that will be tolerated to calculate Rao's index in each moving window over *x*. If the relative proportion of NA's in a moving window is bigger than na.tolerance, then the value of the window will be set as NA, otherwise Rao's index will be calculated considering the non-NA values. Default values is 0.0 (i.e., no tolerance for NA's). Defaut value is 1.0.
rescale	Boolean. Considered only if `method="multidimension"`. If TRUE, each element of `x` is rescaled and centered.
diag	Boolean. If TRUE then the diagonal of the distance matrix is filled with 0's, otherwise with NA's. If `diag=TRUE` and `alpha=0`, the output matrix will inexorably be composed of 0's.
simplify	Number of decimal places to be retained to calculate distances in Rao's index. Only if *x* is floats.
np	the number of processes (cores) which will be spawned. Default value is 2.
cluster.type	the type of cluster which will be created. The options are *"MPI"* (which calls "makeMPIcluster"), *"FORK"* and *"SOCK"* (which call "makeCluster"). Default type is *"SOCK"*.
debugging	a boolean variable set to FALSE by default. If TRUE, additional messages will be printed. For debugging only.

Details

The accumulation function for the parametric Rao's Index (Q) is calulcated integrating numerically over a range of alphas. *accRao* is therefore equal to $(\int_a^b \frac{1}{N^4} \cdot d_{i,j}^{\alpha})^{\frac{1}{\alpha}} dx$. Where *N* is the number of pixels in a moving window, and *alpha* is a weight assigned to distances.

Value

A matrix of dimension `dim(x)`. If `rasterAUC=TRUE`, then the output is a RasterLayer withe *x* as template.

Author(s)

Matteo Marcantonio <marcantoniomatteo@gmail.com>

References

[1] Rocchini, D., M. Marcantonio, and C. Ricotta (2017). Measuring Rao's Q diversity index from remote sensing: An open source solution. Ecological Indicators. 72: 234–238.

See Also

Rao paRao

Examples

```
#Minimal example; accRao with alphas ranging from 1 to 10
a <- matrix(c(10,10,10,20,20,20,20,30,30),ncol=3,nrow=3)
out <- accRao(alphas=1:10,x=a,window=3,dist_m="euclidean",na.tolerance=1, rasterAUC=TRUE)
```

BergerParker — *Berger-Parker's diversity index*

Description

Computes Berger-Parker's diversity index on different classes of numeric matrices using a moving window algorithm.

Usage

```
BergerParker(x, window=3, rasterOut=TRUE, np=1,
na.tolerance=1, cluster.type="SOCK", debugging=FALSE)
```

Arguments

`x`	input data may be a matrix, a Spatial Grid Data Frame, a RasterLayer or a list of these objects. In the latter case, only the first element of the list will be considered.
`window`	the side of the square moving window, it must be a odd numeric value greater than 1 to ensure that the target pixel is in the center of the moving window. Default value is 3.
`rasterOut`	Boolean, if TRUE output will be in RasterLayer format with *x* as template.
`np`	the number of processes (cores) which will be spawned. Default value is 1.
`na.tolerance`	a numeric value $(0.0 - 1.0)$ which indicates the proportion of NA values that will be tolerated to calculate Rao's index in each moving window over *x*. If the relative proportion of NA's in a moving window is bigger than na.tolerance, then the value of the window will be set as NA, otherwise Rao's index will be calculated considering the non-NA values. Default values is 0.0 (i.e., no tolerance for NA's).

`cluster.type` the type of cluster which will be created. The options are "MPI" (calls "makeMPIcluster"), "FORK" and "SOCK" (call "makeCluster"). Default type is "SOCK".

`debugging` a boolean variable set to FALSE by default. If TRUE, additional messages will be printed. For de-bugging only.

Details

Berger-Parker's index is the relative abundance of the most abundant category (i.e., unique numerical values in the considered numerical matrix). Berger-Parker's index equals the logarithm of the inverse Renyi's index of order infinity, $log(1/^{\infty}H)$ or the inverse of Hill's index of order infinity, $1/^{\infty}D$.

Value

A numerical matrix with dimension as `dim(x)`.

Note

Linux users need to install libopenmpi for MPI parallel computing. Linux Ubuntu users may try: apt-get update; apt-get upgrade; apt-get install mpi; apt-get install libopenmpi-dev; apt-get install r-cran-rmpi

Microsoft Windows users may need some additional work to use "MPI", see:
https://bioinfomagician.wordpress.com/2013/11/18/installing-rmpi-mpi-for-r-on-mac-and-windows/

Author(s)

Marcantonio Matteo <marcantoniomatteo@gmail.com>
Martina Iannacito <martina.iannacito@inria.fr>
Duccio Rocchini <duccio.rocchini@unibo.it>

References

Berger, W.H., Parker, F.L. (1970). Diversity of planktonic foraminifera in deep-sea sediments". Science, 168: 1345-1347.

Examples

```
#Minimal example; compute Renyi's index with alpha 1:5
a <- matrix(c(10,10,10,20,20,20,20,30,30),ncol=3,nrow=3)
berpar <- BergerParker(x=a,window=3)
```

copNDVI *Copernicus Long Term (1999-2017) NDVI Overview (5km)*

Description

A RasterLayer (EPSG: 4326) of the global average NDVI value per pixel for the 21st of June over the period 1999-2017.

Usage

```
copNDVI
```

Format

RasterLayer:

NDVI Normalised Difference Vegetation Index value (0-255) for each 5 km pixel.

References

https://land.copernicus.eu/global/products/ndvi

CRE *Cumulative Residual Entropy (CRE)*

Description

Computes Cumulative Residual Entropy (CRE) on different classes of numeric matrices using a moving window algorithm.

Usage

```
CRE(x, window=3, mode="classic", rasterOut=TRUE,
  rescale=FALSE, na.tolerance=1.0, simplify=2, np=1,
  cluster.type="SOCK", debugging=FALSE)
```

Arguments

`x` input data may be a matrix, a Spatial Grid Data Frame, a RasterLayer or a list of these objects. In the latter case, if *mode="classic"* only the first element of the list will be considered.

`window` the side of the square moving window, it must be a odd numeric value greater than 1 to ensure that the target pixel is in the center of the moving window. Default value is 3.

`mode`	currently, there are two modes to calculate Cumulative Residual Entropy (CRE). If mode is "classic", then the monodimension CRE will be calculated on one single matrix. If mode is "multidimension" (experimental!) a list of matrices must be provided as input. In this latter case, the multidimensional cumulative residual probability will be calculated. Default value is "classic".
`rasterOut`	Boolean, if TRUE output will be in RasterLayer format with *x* as template.
`rescale`	a boolean variable set to FALSE by default. If TRUE, *x* will be scaled and centered to standardise different matrices if mode is "multidimension". Default value is FALSE.
`na.tolerance`	a numeric value $(0.0-1.0)$ which indicates the proportion of NA values that will be tolerated to calculate CRE in each moving window over *x*. If the relative proportion of NA's in a moving window is bigger than na.tolerance, then the value of the window will be set as NA, otherwise CRE will be calculated considering the non-NA values. Default values is 1.0 (i.e., full tolerance for NA's).
`simplify`	Number of decimal places to be retained to calculate CRE. Only if *x* is floats.
`np`	the number of processes (cores) which will be spawned. Default value is 1.
`cluster.type`	the type of cluster which will be created. The options are *"MPI"* (which calls "makeMPIcluster"), *"FORK"* and *"SOCK"* (which call "makeCluster"). Default type is *"SOCK"*.
`debugging`	a boolean variable set to FALSE by default. If TRUE, additional messages will be printed. For debugging only.

Details

Unidimension Cumulative Residual Entropy (CRE) is calculated on a numerical vector as $CRE = CRE = -\sum_{i=1}^{N} P(X \geq x_i) \log P(X \geq x_i) dx$ [1] where $dx = (x_i - x_{i-1})$ and *P* is a vector of probabilities that the vector of observations is larger or equal to each value of the vector. In the "multidimension" CRE, *P* becomes an array with as many dimensions as the one of the observations. In each cell of *P* the probability estimates with the frequency of the number of observation that as the same time satisify the larger or equal requirment for the different combination of values along the dimension. *dx* becomes the products of the difference along each dimension.
The theoretical minimum of CRE is 0, when all values are identical in a window. The values of CRE increases with the range of observations, thus the more the observations are equally spread (even) across values the higher CRE will be.

Value

A matrix of dimension *dim(x)*.

Author(s)

Saverio Vicario `<saverio.vicario@cnr.it>`

References

[1] Rao M, Chen Y, Vemuri BC, Wang F (2004) Cumulative Residual Entropy: A New Measure of Information. IEEE Trans Inf Theory 50:1220–1228.

Examples

```
#Minimum example; compute CRE
a <- matrix(c(10,10,10,20,20,20,20,30,30),ncol=3,nrow=3)
out <- CRE(x=a,window=3,na.tolerance=0,mode="classic")

#Minimum example; compute CRE in paralllel
a <- matrix(c(10,10,10,20,20,20,20,30,30),ncol=3,nrow=3)
out <- CRE(x=a,window=3,na.tolerance=0,mode="classic",np=1)

#Compute multidimension Rao's index rescaling the input matrices
a <- matrix(c(10,10,10,20,20,20,20,30,30),ncol=3,nrow=3)
b <- matrix(c(0.5,0.5,0.1,0.1,0.3,0.3,0.3,0.3,0.3),ncol=3,nrow=3)
out <- CRE(x=list(a,b),window=3,na.tolerance=0,
  mode="multidimension",rescale=TRUE,debugging=TRUE)
```

Hill *Hill's index of diversity - Hill numbers (D)*

Description

Computes Hill's index of diversity (Hill numbers) on different classes of numeric matrices using a moving window algorithm.

Usage

```
Hill(x, window = 3, alpha = 1, rasterOut=TRUE,
np = 1, na.tolerance=1, cluster.type = "SOCK",
debugging = FALSE)
```

Arguments

`x`	input data may be a matrix, a Spatial Grid Data Frame, a RasterLayer or a list of these objects. In the latter case, only the first element of the list will be considered.
`window`	the side of the square moving window, it must be a odd numeric value greater than 1 to ensure that the target pixel is in the center of the moving window. Default value is 3.
`alpha`	Order of the Hill number to compute the index. If "`alpha`" is a vector with length greater than 1, then the index will be calculated over `x` for each value in the sequence.
`rasterOut`	Boolean, if TRUE output will be in RasterLayer format with *x* as template.
`np`	the number of processes (cores) which will be spawned. Default value is 1.
`na.tolerance`	a numeric value $(0.0 - 1.0)$ which indicates the proportion of NA values that will be tolerated to calculate Rao's index in each moving window over *x*. If the relative proportion of NA's in a moving window is bigger than na.tolerance, then the value of the window will be set as NA, otherwise Rao's index will be calculated considering the non-NA values. Default values is 0.0 (i.e., no tolerance for NA's).

`cluster.type` the type of cluster which will be created. The options are "MPI" (calls "makeMPIcluster"), "FORK" and "SOCK" (call "makeCluster"). Default type is "SOCK".

`debugging` a boolean variable set to FALSE by default. If TRUE, additional messages will be printed. For debugging only.

Details

Hill numbers (${}^qD$) are calculated on a numerical matrices as ${}^qD = (\sum_{i=1}^{R} p^q{}_i)^{1/(1-q)}$,where q is the order of the Hill number, *R* is the total number of categories (i.e., unique numerical values in a numerical matrix), *p* is the relative abundance of each category. When q=1, Shannon.R is called to calculate $exp(H^1)$ instead of the indefinite ${}^1D$. if $q > 2 * 10^9$, BerkgerParker.R is called to calculate $1/{}^\infty D$. Hill numbers of low order weight more rare categories, whereas Hill numbers of higher order weight more dominant categories.

Value

A list of matrices of dimension `dim(x)` with length equal to the length of `alpha`.

Note

Linux users need to install libopenmpi for MPI parallel computing. Linux Ubuntu users may try: apt-get update; apt-get upgrade; apt-get install mpi; apt-get install libopenmpi-dev; apt-get install r-cran-rmpi

Microsoft Windows users may need some additional work to use "MPI", see:
https://bioinfomagician.wordpress.com/2013/11/18/installing-rmpi-mpi-for-r-on-mac-and-windows/

Author(s)

Marcantonio Matteo <marcantoniomatteo@gmail.com>
Martina Iannacito <martina.iannacito@inria.fr>
Duccio Rocchini <duccio.rocchini@unibo.it>

References

Hill, M.O. (1973). Diversity and evenness: a unifying notation and its consequences. Ecology 54, 427-431.

See Also

BergerParker Shannon

Examples

```
#Minimal example; compute Hill's index with alpha 1:5
a <- matrix(c(10,10,10,20,20,20,20,30,30),ncol=3,nrow=3)
hill <- Hill(x=a,window=3,alpha=1:5)
```

paRao *Parametric Rao's index of quadratic entropy (Q)*

Description

Computes the parametric version of Rao's index of quadratic entropy (Q) on different classes of numeric matrices using a moving window algorithm.

Usage

```
paRao(x, dist_m="euclidean", window=9, alpha=1,
  method="classic", rasterOut=TRUE, lambda=0,
  na.tolerance=1.0, rescale=FALSE, diag=TRUE,
  simplify=2, np=1,cluster.type="SOCK", debugging=FALSE)
```

Arguments

`x`	input data may be a matrix, a Spatial Grid Data Frame, a RasterLayer or a list of these objects. In the latter case, if `method="classic"` only the first element of the list will be considered.
`dist_m`	define the type of distance to be calculated between numerical categories. `dist_m` can be a character string which defines the name of the distance to derive such as "euclidean". The distance names allowed are the same as for `proxy::dist`. Alternatively, `dist_m` can be a function which calculates an user defined distance, (i.e., `function(x,y) {return(cos(y-x)-sin(y-x))}`) or a matrix of distances. If `method="multidimension"` then only "euclidean", "manhattan", "canberra", "minkowski" and "mahalanobis" can be used. Default value is "euclidean".
`window`	the side of the square moving window, it must be a odd numeric value greater than 1 to ensure that the target pixel is in the center of the moving window. Default value is 3. If `proxy::dist` is a matrix then the function will assume that this is the distance matrix, and therefore no distance will be derived.
`alpha`	weight for the distance matrix. If `alpha = 0`, distances will be averaged with an geometric mean, if `alpha=1` with an arithmetic mean, if `alpha = 2` with a quadratic mean, `alpha = 3` with a cubic mean and so on and so forth. if alpha tends to infinite (i.e., higher than the maximum integer allowed in R) then the maximum of the distances will be taken. *"alpha"* can be a vector with length greater than 1, in this case Rao's index will be calculated over *x* for each value in the sequence.
`method`	Currently, there are two ways to calculate the parameteric version of Rao's index. If `method="classic"`, then the normal parameteric Rao's index will be calculated on a single matrix. If `method="multidimension"` (experimental!) a list of matrices must be provided as input. In the latter case, the overall distance matrix will be calculated in a multi- or hyper-dimensional system by using the distance measure defined through the function argument `dist_m`. Each pairwise

distance is then multiplied by the inverse of the squared number of pixels in the considered moving window, and the Rao's Q is finally derived by applying a summation. Default value is *"classic"*

`rasterOut` Boolean, if TRUE the output will be a RasterLayer object with *x* as a raster template.

`lambda` the value of the lambda of Minkowski's distance. Considered only if `dist_m = "minkowski"` and `method="multidimension". Default value is 0.`

`na.tolerance` Numeric value $(0.0 - 1.0)$ which indicates the proportion of NA values that will be tolerated to calculate Rao's index in each moving window over *x*. If the relative proportion of NA's in a moving window is bigger than na.tolerance, then the value of the window will be set as NA, otherwise Rao's index will be calculated considering the non-NA values. Default values is 0.0 (i.e., no tolerance for NA's). Defaut value is 1.0.

`rescale` Boolean. Considered only if `method="multidimension"`. If TRUE, each element of `x` is rescaled and centered.

`diag` Boolean. If TRUE then the diagonal of the distance matrix is filled with 0's, otherwise with NA's. If `diag=TRUE` and `alpha=0`, the output matrix will inexorably be composed of 0's.

`simplify` Number of decimal places to be retained to calculate distances in Rao's index. Only if *x* is floats.

`np` the number of processes (cores) which will be spawned. Default value is 2.

`cluster.type` the type of cluster which will be created. The options are *"MPI"* (which calls "makeMPIcluster"), *"FORK"* and *"SOCK"* (which call "makeCluster"). Default type is *"SOCK"*.

`debugging` a boolean variable set to FALSE by default. If TRUE, additional messages will be printed. For debugging only.

Details

The parametric Rao's Index (Q) is an extenstion of Rao's Index which considers a generalised mean between distances. The generalised formula for the parametric Rao's index is $Q_a = (\sum_{i,j=1}^{N} 1/N^2 \times d_{j,j}^{\alpha})^{1/\alpha}$. Where *N* is the number of numerical categories, *i* and *j* are pair of numerical categories in the same moving window and *alpha* is a weight given to distances. In the "multidimension" Rao's index, first distances among categories are calculated considering more than one layer, then the pairwise distance between each pair of numerical categories is multiplied to the square of the size of the moving window (this is somewhat the same as to calculate the variance of the multidimensional distance [1].).
The theoretical minimum of Rao's Q is 0, when all categories in a window have distance 0. If the distance chosen to calculate Rao's Index ranges between 0 and 1, the maximum value of Rao's Index equals the Simpson Index of Diversity $1 - 1/S_i$ where *S* is the number of categories in window *i* (given alpha=1).

Value

A list of matrices of dimension `dim(x)` with length equal to the length of *alpha*. If `rasterOut=TRUE`, then the output is a list of RasterLayer objects.

Author(s)

Matteo Marcantonio <marcantoniomatteo@gmail.com>
Duccio Rocchini <duccio.rocchini@unibo.it>

References

[1] Rocchini, D., M. Marcantonio, and C. Ricotta (2017). Measuring Rao's Q diversity index from remote sensing: An open source solution. Ecological Indicators. 72: 234–238.

Examples

```
#Minimal example; compute classic Rao's index
a <- matrix(c(10,10,10,20,20,20,20,30,30),ncol=3,nrow=3)
out <- paRao(x=a,window=3,dist_m="euclidean",na.tolerance=0,alpha=1)

#Compute Rao's index with a vector of alpha values
out <- paRao(x=a,window=3,dist_m="euclidean",na.tolerance=0,alpha=1:5)

#Compute multidimensional Rao's index with alpha = 1
a <- matrix(c(10,10,10,20,20,20,20,30,30),ncol=3,nrow=3)
b <- matrix(c(100,100,100,200,200,200,200,300,300),ncol=3,nrow=3)

out <- paRao(x=list(a,b),window=3,dist_m="euclidean",
  na.tolerance=0,alpha=1,method="multidimensional")
```

Pielou *Pielou's evenness index (E')*

Description

Computes Pielou's evenness index on different classes of numeric matrices using a moving window algorithm.

Usage

```
Pielou(x, window=3, rasterOut=TRUE, np=1, na.tolerance=1, cluster.type="SOCK",
 debugging=FALSE)
```

Arguments

x	input data may be a matrix, a Spatial Grid Data Frame, a RasterLayer or a list of these objects. In the latter case, only the first element of the list will be considered.
window	the side of the square moving window, it must be a odd numeric value greater than 1 to ensure that the target pixel is in the center of the moving window. Default value is 3.
rasterOut	Boolean, if TRUE the output will be in RasterLayer format with *x* as template.

`np` the type of cluster which will be created. The options are `"MPI"` (calls "makeMPIcluster"), "FORK" and "SOCK" (call "makeCluster"). Default type is "SOCK".

`na.tolerance` a numeric value $(0.0 - 1.0)$ which indicates the proportion of NA values that will be tolerated to calculate Rao's index in each moving window over *x*. If the relative proportion of NA's in a moving window is bigger than na.tolerance, then the value of the window will be set as NA, otherwise Rao's index will be calculated considering the non-NA values. Default values is 0.0 (i.e., no tolerance for NA's).

`cluster.type` the type of cluster which will be created. The options are "MPI" (calls "makeMPIcluster"), "FORK" and "SOCK" (call "makeCluster"). Default type is `"SOCK"`.

`debugging` a boolean variable set to FALSE by default. If TRUE, additional messages will be printed. For debugging only.

Details

Pielou evenness's index (E') is calculated on a numerical matrix as $E' = \frac{\sum_{i=1}^{R} p_i \times log(p_i)}{log(R)}$, where *R* is the total number of categories (i.e., unique numerical values in the considered numerical matrix) and *p* is the relative abundance of each category. Pielou's evenness represents the ratio between the observed value of Shannon's Index and the value of Shannon's Index if all categories (R) had the same relative abundance.

Value

A numerical matrix with dimension as `dim(x)`.

Note

Linux users need to install libopenmpi for MPI parallel computing. Linux Ubuntu users may try: apt-get update; apt-get upgrade; apt-get install mpi; apt-get install libopenmpi-dev; apt-get install r-cran-rmpi

Microsoft Windows users may need some additional work to use "MPI", see:
https://bioinfomagician.wordpress.com/2013/11/18/installing-rmpi-mpi-for-r-on-mac-and-windows/

Author(s)

Marcantonio Matteo <marcantoniomatteo@gmail.com>
Martina Iannacito <martina.iannacito@inria.fr>
Duccio Rocchini <duccio.rocchini@unibo.it>

References

Pielou, E.C. (1966). The measurement of diversity in different types of biological collections. Journal of Theoretical Biology, 13: 131-144.

See Also

Shannon

Examples

```
#Minimal example; compute Shannon's index
a <- matrix(c(10,10,10,20,20,20,20,30,30),ncol=3,nrow=3)
renyi <- Pielou(x=a,window=3)
```

Rao *Rao's index of quadratic entropy (Q)*

Description

Computes Rao's index of quadratic entropy (Q) on different classes of numeric matrices using a moving window algorithm.

Usage

```
Rao(x, dist_m="euclidean", window=9, rasterOut = TRUE,
 mode="classic",lambda=0, shannon=FALSE, rescale=FALSE,
 na.tolerance=1.0, simplify=2, np=1, cluster.type="SOCK",
 debugging=FALSE)
```

Arguments

`x`	input data may be a matrix, a Spatial Grid Data Frame, a RasterLayer or a list of these objects. In the latter case, if `mode="classic"` only the first element of the list will be considered.
`dist_m`	define the type of distance to be calculated between numerical categories. `dist_m` can be a character string which defines the name of the distance to derive such as "euclidean". The distance names allowed are the same as for `proxy::dist`. Alternatively, `dist_m` can be a function which calculates an user defined distance, (i.e., `function(x,y) {return(cos(y-x)-sin(y-x))}`) or a matrix of distances. If `mode="multidimension"` then only "euclidean", "manhattan", "canberra", "minkowski" and "mahalanobis" can be used. Default value is "euclidean".
`window`	the side of the square moving window, it must be a odd numeric value greater than 1 to ensure that the target pixel is in the center of the moving window. Default value is 3. If `proxy::dist` is a matrix then the function will assume that this is the distance matrix, and therefore no distance will be derived.
`rasterOut`	Boolean, if TRUE the output will be in RasterLayer format with *x* as template.
`mode`	currently, there are two modes to calculate Rao's index. If mode is "classic", then the classic Rao's index will be calculated on one single matrix. If mode is "multidimension" (experimental!) a list of matrices must be provided as input. In this latter case, the overall distance matrix will be calculated in a multi- or hyper-dimensional system by using the measure defined through the function argument `dist_m`. Each pairwise distance is then multiplied by the inverse of the squared number of pixels in the considered moving window, and the Rao's Q is finally derived by applying a summation. Default value is "classic".

`lambda`	the value of the lambda of Minkowski's distance. Considered only if `dist_m = "minkowski"` and `mode="multidimension"`. Default value is `0`.
`shannon`	a boolean variable set to FALSE by default. If TRUE, a matrix with Shannon index will be also calculated. Default value is FALSE.
`rescale`	a boolean variable set to FALSE by default. If TRUE, *x* will be scaled and centered to standardise different matrices if mode is "multidimension". Default value is FALSE.
`na.tolerance`	a numeric value $(0.0 - 1.0)$ which indicates the proportion of NA values that will be tolerated to calculate Rao's index in each moving window over *x*. If the relative proportion of NA's in a moving window is bigger than na.tolerance, then the value of the window will be set as NA, otherwise Rao's index will be calculated considering the non-NA values. Default values is 1.0 (i.e., full tolerance for NA's).
`simplify`	Number of decimal places to be retained to calculate distances in Rao's index. Only if *x* is floats.
`np`	the number of processes (cores) which will be spawned. Default value is 1.
`cluster.type`	the type of cluster which will be created. The options are `"MPI"` (which calls "makeMPIcluster"), `"FORK"` and `"SOCK"` (which call "makeCluster"). Default type is `"SOCK"`.
`debugging`	a boolean variable set to FALSE by default. If TRUE, additional messages will be printed. For debugging only.

Details

Classical Rao's Index (Q) is calculated on a numerical matrix as $Q = \sum_{i=1}^{R} \sum_{j=1}^{R} d_{i,j} \times p_i \times p_j$ [1]. Where R is the number of categories, whereas *i* and *j* are pair of numerical categories in the same moving window. In the "multidimension" Rao's index, distances among categories are calculated considering more than one layer, then the pairwise distance between each pair of numerical categories is multiplied to the square of the size of the moving window (which is somewhat the same as to calculate the variance of the multidimensional distance [2].).
The theoretical minimum of Rao's Q is 0, when all categories in a window have distance 0. If the distance chosen to calculate Rao's Index ranges between 0 and 1, the maximum value of Rao's Index equals the Simpson Index of Diversity $1 - 1/S_i$ where *S* is the number of categories in window *i*.

Value

If shannon=TRUE, a list of matrices of length two, otherwise a matrix of dimension `dim(x)`.

Author(s)

Matteo Marcantonio <marcantoniomatteo@gmail.com>
Duccio Rocchini <duccio.rocchini@unibo.it>

References

[1] Rao, C.R. (1982). Diversity and dissimilarity coefficients: a unified approach. Theoretical Population Biology, 21: 2443. [2] Rocchini, D., M. Marcantonio, and C. Ricotta (2017). Measuring Rao's Q diversity index from remote sensing: An open source solution. Ecological Indicators. 72: 234–238.

Examples

```
#Minimal example; compute Rao's index
a <- matrix(c(10,10,10,20,20,20,20,30,30),ncol=3,nrow=3)
out <- Rao(x=a,window=3,dist_m="euclidean",na.tolerance=0,shannon=FALSE,mode="classic")

#Compute both Rao and Shannon index
a <- matrix(c(10,10,10,20,20,20,20,30,30),ncol=3,nrow=3)
out <- Rao(x=a,window=3,dist_m="euclidean",na.tolerance=0,shannon=TRUE,mode="classic")

#Compute multidimension Rao's index rescaling the input matrices
a <- matrix(c(10,10,10,20,20,20,20,30,30),ncol=3,nrow=3)
b <- matrix(c(0.5,0.5,0.1,0.1,0.3,0.3,0.3,0.3,0.3),ncol=3,nrow=3)
out <- Rao(x=list(a,b),window=3,dist_m="euclidean",na.tolerance=0,
  mode="multidimension",rescale=TRUE,debugging=TRUE)
```

Renyi *Renyi's entropy (H)*

Description

Computes Renyi's entropy (${}^qH$) on different classes of numeric matrices using a moving window algorithm.

Usage

```
Renyi(x, window=3, alpha=1, base=exp(1), rasterOut=TRUE,
np=1, na.tolerance=, cluster.type="SOCK", debugging=FALSE)
```

Arguments

`x`	input data may be a matrix, a Spatial Grid Data Frame, a RasterLayer or a list of these objects. In the latter case, only the first element of the list will be considered.
`window`	the side of the square moving window, it must be a odd numeric value greater than 1 to ensure that the target pixel is in the center of the moving window. Default value is 3.
`alpha`	Order of diversity to compute the index. If `alpha` is a vector with length greater than 1, then the index will be calculated over `x` for each value in the sequence.
`base`	a numerical value which defines the base of the logarithm in Renyi's entropy formula. Defalut value is exp(1).

`rasterOut`	Boolean, if TRUE output will be in RasterLayer format with *x* as template.
`np`	the number of processes (cores) which will be spawned. Default value is 1.
`na.tolerance`	a numeric value $(0.0 - 1.0)$ which indicates the proportion of NA values that will be tolerated to calculate Rao's index in each moving window over *x*. If the relative proportion of NA's in a moving window is bigger than na.tolerance, then the value of the window will be set as NA, otherwise Rao's index will be calculated considering the non-NA values. Default values is 0.0 (i.e., no tolerance for NA's).
`cluster.type`	the type of cluster which will be created. The options are "MPI" (calls "makeMPIcluster"), "FORK" and "SOCK" (call "makeCluster"). Default type is "SOCK".
`debugging`	a boolean variable set to FALSE by default. If TRUE, additional messages will be printed. For debugging only.

Details

Renyi's entropy (${}^qH$) is calculated on a numerical matrix as ${}^qH = \frac{1}{(1-q)} \ln(\sum_{i=1}^{R} p^q{}_i)$, where q is the considered order of diversity (`alpha`), *R* is the total number of categories (i.e., unique numerical values in the considered numerical matrix) and *p* is the relative abundance of each category. If q=1, Shannon.R is called to calculate H' instead of the indefinite ${}^1D$, if $p > 2 * 10^9$, then BerkgerParker.R is called to calculate $log(1/{}^\infty H)$. Renyi's entropy of low order weight more rare numerical categories, whereas values of higher order weight more dominant categories.

Value

A list of matrices with length equal to the length of "alpha". If length of "alpha" is 1, then a matrix of dimension `dim(x)`.

Note

Linux users need to install libopenmpi for MPI parallel computing. Linux Ubuntu users may try: apt-get update; apt-get upgrade; apt-get install mpi; apt-get install libopenmpi-dev; apt-get install r-cran-rmpi

Microsoft Windows users may need some additional work to use "MPI", see:
https://bioinfomagician.wordpress.com/2013/11/18/installing-rmpi-mpi-for-r-on-mac-and-windows/

Author(s)

Matteo Marcantonio <marcantoniomatteo@gmail.com>
Martina Iannacito <martina.iannacito@inria.fr>
Duccio Rocchini <duccio.rocchini@unibo.it>

References

Rényi, A., 1970. Probability Theory. North Holland Publishing Company, Amsterdam.

See Also

`Shannon`, `BergerParker`

Examples

```
#Minimal example; compute Renyi's index with alpha 1:5
a <- matrix(c(10,10,10,20,20,20,20,30,30),ncol=3,nrow=3)
renyi <- Renyi(x=a,window=3,alpha=1:5)
```

Shannon *Shannon's diversity index (H')*

Description

Computes Shannon's diversity index (H') on different classes of numeric matrices using a moving window algorithm.

Usage

```
Shannon(x, window=3, rasterOut=TRUE, np=1, na.tolerance=1,
cluster.type="SOCK", debugging=FALSE)
```

Arguments

`x`	input data may be a matrix, a Spatial Grid Data Frame, a RasterLayer or a list of these objects. In the latter case, only the first element of the list will be considered.
`window`	the side of the square moving window, it must be a odd numeric value greater than 1 to ensure that the target pixel is in the center of the moving window. Default value is 3.
`rasterOut`	Boolean, if TRUE the output will be in RasterLayer format with *x* as template.
`np`	the number of processes (cores) which will be spawned. Default value is 1.
`na.tolerance`	a numeric value $(0.0 - 1.0)$ which indicates the proportion of NA values that will be tolerated to calculate Rao's index in each moving window over *x*. If the relative proportion of NA's in a moving window is bigger than na.tolerance, then the value of the window will be set as NA, otherwise Rao's index will be calculated considering the non-NA values. Default values is 0.0 (i.e., no tolerance for NA's).
`cluster.type`	the type of cluster which will be created. The options are `"MPI"` (calls "makeMPIcluster"), "FORK" and "SOCK" (call "makeCluster"). Default type is `"SOCK"`.
`debugging`	a boolean variable set to FALSE by default. If TRUE, additional messages will be printed. For debugging only.

Details

Shannon's index (H') is calculated on a numerical matrix as $H' = \sum_{i=1}^{R} p_i \times log(p_i)$, where *R* is the total number of categories (i.e., unique numerical values in the considered numerical matrix) and *p* is the relative abundance of each category.

Value

A numerical matrix with dimension {dim(x)}.

Note

Linux users need to install libopenmpi for MPI parallel computing. Linux Ubuntu users may try: apt-get update; apt-get upgrade; apt-get install mpi; apt-get install libopenmpi-dev; apt-get install r-cran-rmpi

Microsoft Windows users may need some additional work to use "MPI", see:
https://bioinfomagician.wordpress.com/2013/11/18/installing-rmpi-mpi-for-r-on-mac-and-windows/

Author(s)

Marcantonio Matteo <marcantoniomatteo@gmail.com>
Martina Iannacito <martina.iannacito@inria.fr>
Duccio Rocchini <duccio.rocchini@unibo.it>

References

Shannon, C.E. (1948). A mathematical theory of communication. Bell System Technical Journal, 27: 379-423, 623-656.

Examples

```
#Minimal example; compute Shannon's index
a <- matrix(c(10,10,10,20,20,20,20,30,30),ncol=3,nrow=3)
shannon <- Shannon(x=a,window=3)
```

Index