Package 'vegclust'

August 25, 2022

Type Package

Title Fuzzy Clustering of Vegetation Data

Version 2.0.2

Date 2022-08-24

Description A set of functions to: (1) perform fuzzy clustering of vegetation data (De Caceres et al, 2010) <doi:10.1111/j.1654-1103.2010.01211.x>; (2) to assess ecological community similarity on the basis of structure and composition (De Caceres et al, 2013) <doi:10.1111/2041-210X.12116>.

Depends R (>= 3.4.0)

Imports vegan

License GPL (>= 2)

URL https://emf-creaf.github.io/vegclust/

LazyLoad yes

Encoding UTF-8

NeedsCompilation yes

RoxygenNote 7.1.1

Suggests knitr, rmarkdown

VignetteBuilder utils, knitr

Author Miquel De Cáceres [aut, cre]

Maintainer Miquel De Cáceres <miquelcaceres@gmail.com>

Repository CRAN

Date/Publication 2022-08-25 08:50:02 UTC

R topics documented:

`vegclust-package` *Fuzzy Clustering of Vegetation Data Functions for fuzzy and hard clustering of vegetation data*

Description

A set of functions to: (1) perform fuzzy clustering of vegetation data (De Caceres et al, 2010) <doi:10.1111/j.1654-1103.2010.01211.x>; (2) to assess ecological community similarity on the basis of structure and composition (De Caceres et al, 2013) <doi:10.1111/2041-210X.12116>. This package contains functions used to perform fuzzy and hard clustering of vegetation data under different models.

Details

The DESCRIPTION file:

Package:	vegclust
Type:	Package
Title:	Fuzzy Clustering of Vegetation Data
Version:	2.0.2

Date:	2022-08-24
Authors@R:	c(person('Miquel', 'De Cáceres', role=c('aut','cre'), email='miquelcaceres@gmail.com'))
Description:	A set of functions to: (1) perform fuzzy clustering of vegetation data (De Caceres et al, 2010) <doi:10.1
Depends:	R (>= 3.4.0)
Imports:	vegan
License:	GPL (>= 2)
URL:	https://emf-creaf.github.io/vegclust/
LazyLoad:	yes
Encoding:	UTF-8
NeedsCompilation:	yes
RoxygenNote:	7.1.1
Suggests:	knitr, rmarkdown
VignetteBuilder:	utils, knitr
Author:	Miquel De Cáceres [aut, cre]
Maintainer:	Miquel De Cáceres <miquelcaceres@gmail.com>

Index of help topics:

vegclust-package	Fuzzy Clustering of Vegetation Data Functions for fuzzy and hard clustering of vegetation data
vegclust2kmeans	Reshapes as kmeans object
vegclustIndex	Compute fuzzy evaluation statistics
vegdiststruct	Structural and compositional dissimilarity
wetland	Wetland vegetation data set

Author(s)

NA Maintainer: NA

References

De Caceres, M., Font, X, Oliva, F. (2010) The management of numerical vegetation classifications with fuzzy clustering methods. Journal of Vegetation Science 21 (6): 1138-1151.

De Cáceres, M., Legendre, P., & He, F. 2013. Dissimilarity measurements and the size structure of ecological communities (D. Faith, Ed.). Methods in Ecology and Evolution 4: 1167–1177.

Examples

```
## Loads data
data(wetland)

## This equals the chord transformation
wetland.chord = as.data.frame(sweep(as.matrix(wetland), 1,
                              sqrt(rowSums(as.matrix(wetland)^2)), "/"))

## Create noise clustering with 3 clusters. Perform 10 starts from random seeds
## and keep the best solution
wetland.nc = vegclust(wetland.chord, mobileCenters=3, m = 1.2, dnoise=0.75,
                      method="NC", nstart=10)
```

---

as.memb *Turns into membership matrix*

Description

Attempts to turn its cluster vector argument into a membership matrix

Usage

```
as.memb(cluster)
```

Arguments

cluster	A vector indicating the hard membership of each object in x to a set of groups. Can contain NA values.

Value

An matrix with as many rows as the length of `cluster` and as many columns as different cluster levels. `NA` values will have zero membership to all clusters

Author(s)

Miquel De Cáceres, CREAF.

See Also

`vegclust`, `vegclass`

Examples

```
as.memb(factor(c(1,2,NA)))
```

`as.vegclust` *Turns into vegclust objects*

Description

Attempts to turn its arguments into a `vegclust` object

Usage

```
as.vegclust(x, y, method="KM", m=1.0, dnoise=NULL, eta=NULL)
```

Arguments

`x`	A site-by-species data matrix (raw mode), or a site-by-site distance matrix (distance mode).
`y`	A vector indicating the cluster that each object in `x` belongs to. Alternatively, a fuzzy/hard site-by-group matrix of membership values.
`method`	A clustering model from which `y` was obtained (normally "KM"). Current accepted models are:

- `"KM"`: K-means or hard c-means (MacQueen 1967)
- `"KMdd"`: Hard c-medoids (Krishnapuram et al. 1999)
- `"FCM"`: Fuzzy c-means (Bezdek 1981)
- `"FCMdd"`: Fuzzy c-medoids (Krishnapuram et al. 1999)
- `"NC"`: Noise clustering (Dave and Krishnapuram 1997)
- `"NCdd"`: Noise clustering with medoids
- `"HNC"`: Hard noise clustering
- `"HNCdd"`: Hard noise clustering with medoids
- `"PCM"`: Possibilistic c-means (Krishnapuram and Keller 1993)
- `"PCMdd"`: Possibilistic c-medoids

m	The fuzziness exponent to be used, relevant for all fuzzy models (FCM, FCMdd, NC, NCdd, PCM and PCMdd)
dnoise	The distance to the noise cluster, relevant for noise clustering models (NC, HNC, NCdd and HNCdd).
eta	A vector of reference distances, relevant for possibilistic models (PCM and PCMdd).

Details

This function is used to generate vegclust objects which can then be used in vegclass to classify new data. If the input classification is hard (i.e. yes/no membership), cluster centers are calculated as multivariate means, and the method for assigning new data is assumed to be k-means ("KM"), i.e. plots will be assigned to the nearest cluster center. If community data is given as site-by-species data matrix the cluster centroids are added as mobileCenters in the vegclust object. Centroids will not be computed if community data is given as a site-by-site dissimilarity matrix. Moreover, current implementation does not allow y to be a membership matrix when x is a distance matrix.

Value

An object of class vegclust.

Author(s)

Miquel De Cáceres, CREAF.

See Also

vegclust, vegclass

Examples

```
## Loads data
data(wetland)

## This equals the chord transformation
## (see also  \code{\link{decostand}} in package vegan)
wetland.chord = as.data.frame(sweep(as.matrix(wetland), 1,
                              sqrt(rowSums(as.matrix(wetland)^2)), "/"))

## Splits wetland data into two matrices of 30x27 and 11x22
wetland.30 = wetland.chord[1:30,]
wetland.30 = wetland.30[,colSums(wetland.30)>0]
dim(wetland.30)
wetland.11 = wetland.chord[31:41,]
wetland.11 = wetland.11[,colSums(wetland.11)>0]
dim(wetland.11)

## Performs a K-means clustering of the data set with 30 sites
wetland.km = kmeans(wetland.30, centers=3, nstart=10)

## Transforms the 'external' classification of 30 sites into a 'vegclust' object
```

```
wetland.30.vc = as.vegclust(wetland.30, wetland.km$cluster)

## Assigns the second set of sites according to the (k-means) membership rule
## That is, sites are assigned to the cluster whose cluster centroids is nearest.
wetland.11.km = vegclass(wetland.30.vc, wetland.11)

## A similar 'vegclust' object is obtained when using the distance mode...
wetland.d.vc = as.vegclust(dist(wetland.30), wetland.km$cluster)

## which can be also used to produce the assignment of the second set of objects
wetland.d.11 = as.data.frame(as.matrix(dist(wetland.chord)))[31:41,1:30]
wetland.d.11.km = vegclass(wetland.d.vc,wetland.d.11)
```

CAP *Cumulative abundance profile (CAP)*

Description

Functions to calculate cumulative abundance profiles (CAPs), to build matrices from them, and to summarize several profiles.

Usage

```
CAP(x, transform=NULL, verbose=FALSE)
CAP2matrix(CAP, type="cumulative",classWeights=NULL)
CAPcenters(CAP, y=NULL)
CAPquantile(CAP, q = 0.5, y = NULL)
```

Arguments

x	A stratified vegetation data set (see function stratifyvegdata).
transform	A function or the name of a function to be applied to each cumulative abundance value.
verbose	A logical flag to indicate extra output.
CAP	An object of class 'CAP'.
type	The type of information that the resulting matrix should contain. Either "profile", "abundance" or "volume".
classWeights	A numerical vector containing the weight for size class. If NULL, then all classes are assumed to have the same weight.
y	A vector used as a factor to calculate average or quantile profiles per each level. Alternatively, an object of class vegclust for which CAP centroids or medoids are desired.
q	Probability value for which the quantile is desired. By default the median is given.

Details

Function CAP replaces the abundance value of a size class by the sum of abundances in this and larger size classes (strata). Thus, upper classes contain smaller abundance values than lower classes, creating a cumulative abundance profile. Function CAP2matrix takes an object of class 'CAP' and returns a data matrix, where values differ depending on parameter type: (1) type="cumulative" simply reshapes the 'CAP' object (a list) into a matrix with as many rows as plot records and where columns are organized in blocks (there are as many blocks as species and each block has as many columns as size classes); (2) type="total" returns a plot-by-species matrix where each value is the total abundance of the species in the plot (i.e. the CAP value at the ground level); (3) type="volume" returns a plot-by-species matrix where each value is the sum of CAP values across size classes (a measure of the "volume" occupied by the species in the plot). When provided, classWeights are used to weight size classes of the cumulative abundance profiles (for (1) and (3) only). Function CAPcenters calculates the average abundance profile for a set of plot records. If y is a factor, it is used to speficy groups of samples for which average profiles are to be calculated. If y is an object of class 'vegclust' then the function returns the CAP centroids or medoids corresponding to the clustering result. Function CAPquantile calculates a quantile profile for a set of CAPs. The usage of y is the same as for CAPcenters.

Value

Function CAP returns an object of class 'CAP', similar to objects of class 'stratifiedvegdata' but where abundance values of upper size classes have beed added to those of lower size classes. Function CAP2matrix returns a matrix with species as rows (columns depend on the value of type). Functions CAPcenters and CAPquantile return an object of class 'CAP'.

Author(s)

Miquel De Cáceres, CREAF.

References

De Cáceres, M., Legendre, P. & He, F. (2013) Dissimilarity measurements and the size structure of ecological communities. Methods in Ecology and Evolution 4: 1167-1177.

De Cáceres, M., Coll, L., Martín-Alcón, S., González-Olabarria, J.R. (submitted) A general method for the classification of forest stands using structure and composition.

See Also

stratifyvegdata, plot.CAP, vegdiststruct

Examples

```
## Load stratified data
data(medreg)

## Check that 'medreg' has correct class
class(medreg)

## Look at the data for the third plot
```

```
medreg[[3]]

## Create cumulative abundance profile (CAP) for each plot
medreg.CAP = CAP(medreg)

## Look at the profile of the third plot
medreg.CAP[[3]]

## Create matrix with species abundances
medreg.X = CAP2matrix(medreg.CAP, type="total")
head(medreg.X)

## Generate and plot average profile
average.CAP = CAPcenters(medreg.CAP)
plot(average.CAP)

## Generate and plot median profile
median.CAP = CAPquantile(medreg.CAP, q = 0.5)
plot(median.CAP)
```

CAS *Cumulative abundance surface (CAS)*

Description

Functions to calculate cumulative abundance surfaces (CASs), to build matrices from them, and to summarize several CASs.

Usage

```
CAS(x, transform=NULL, verbose=FALSE)
CASmargin(CAS, margin=1, verbose=FALSE)
CAS2matrix(CAS, type="cumulative",classWeights=NULL)
CAScenters(CAS, y=NULL)
CASquantile(CAS, q = 0.5, y = NULL)
```

Arguments

x	An object of class 'doublestratifiedvegdata' (see function stratifyvegdata).
transform	A function or the name of a function to be applied to each cumulative abundance value.
verbose	A logical flag to indicate extra output.
CAS	An object of class 'CAS'.
margin	Indicates whether marginalization should be done in primary (margin = 1) or secondary (margin = 2) size classes.
type	The type of information that the resulting matrix should contain (either "cummulative" or "total").

classWeights	A numerical matrix containing the weight for each combination of size classes. If NULL, then all classes are assumed to have the same weight.
y	A vector used as a factor to calculate average or quantile surfaces per each level. Alternatively, an object of class vegclust for which CAS centroids or medoids are desired.
q	Probability value for which the quantile is desired. By default the median is given.

Details

Function CAS replaces the abundance value of each combination of size classes by the sum of abundances in this and larger size classes. This creates a cumulative abundance surface (similar to a bivariant cummulative distribution function). Function CASmargin takes an object of class 'CAS' and returns an object of class 'CAP' that corresponds marginal profile in either the primary or the secondary size classes. Function CAS2matrix takes an object of class 'CAS' and returns a data matrix, where values differ depending on parameter type: (1) type="cummulative" simply reshapes the 'CAS' object (a list) into a matrix with as many rows as plot records and where columns are organized in blocks (there are as many blocks as species and each block has as many columns as combinations of size classes); (2) type="total" returns a plot-by-species matrix where each value is the total abundance of the species in the plot (i.e. the CAS value at the ground level). When provided, classWeights are used to weight size classes of the cumulative abundance surfaces (for (1) only). Function CAScenters calculates the average abundance surface for a set of plot records. If y is a factor, it is used to speficy groups of samples for which average profiles are to be calculated. If y is an object of class 'vegclust' then the function returns the CAS centroids or medoids corresponding to the clustering result. Function CASquantile calculates a quantile surface for a set of CASs. The usage of y is the same as for CAScenters.

Value

Function CAS returns an object of class 'CAS', similar to objects of class 'doublestratifiedvegdata' but where abundance values of upper size classes have beed added to those of lower size classes. Function CAS2matrix returns a matrix with species as rows (columns depend on the value of type). Functions CAScenters and CASquantile return an object of class 'CAS'.

Author(s)

Miquel De Cáceres, CREAF.

References

De Cáceres, M., Legendre, P. & He, F. (2013) Dissimilarity measurements and the size structure of ecological communities. Methods in Ecology and Evolution 4: 1167-1177.

De Cáceres, M., Coll, L., Martín-Alcón, S., González-Olabarria, J.R. (submitted) A general method for the classification of forest stands using structure and composition.

See Also

stratifyvegdata, plot.CAS, vegdiststruct

Examples

```
## Load tree data
data(treedata)

## Define stratum thresholds (4 strata)
heights = seq(0,4, by=0.5)
diameters = seq(0,2, by=0.5)


## Stratify tree data using heights and diameters as structural variables
X = stratifyvegdata(treedata, sizes1=heights, sizes2=diameters, plotColumn="plotID",
                    speciesColumn="species", size1Column="height", size2Column="diam",
                    counts=TRUE)
X[[2]]

## Build cummulative abundance surface
Y = CAS(X)

Y[[2]]

##  Extracts the first and second marginal (i.e. CAP on heights or diameters respectively)
Y.M1 = CASmargin(Y, margin = 1)
Y.M1[[2]]

Y.M2 = CASmargin(Y, margin = 2)
Y.M2[[2]]

##  For comparison we calculate the same profiles using the stratifyvegdata and CAP functions
Y1 = CAP(stratifyvegdata(treedata, sizes1=heights, plotColumn="plotID",
                    speciesColumn="species", size1Column="height",
                    counts=TRUE))
Y1[[2]]
Y2 = CAP(stratifyvegdata(treedata, sizes1=diameters, plotColumn="plotID",
                    speciesColumn="species", size1Column="diam",
                    counts=TRUE))
Y2[[2]]

##  Compare Y.M1[[2]] with Y1[[2]] and Y.M2[[2]] with Y2[[2]]
```

`clustcentroid` *Cluster centers of a classification*

Description

Function `clustcentroid` calculates the centroid (multivariate average) coordinates of a classification. Function `clustmedoid` determines the medoid (object whose average dissimilarity to all the other objects is minimal) for each cluster in the classification.

Usage

```
clustcentroid(x, y, m = 1)
clustmedoid(x, y, m = 1)
```

Arguments

x	Community data, a site-by-species data frame. In function clustmedoid, x can alternatively be an object of class dist (otherwise, the dissimilarity measure is assumed to be the Euclidean distance).
y	It can be (a) A vector indicating the cluster that each object in x belongs to; (b) a fuzzy/hard site-by-group matrix of membership values; (c) an object of class vegclust or vegclass
m	Fuzziness exponent, only effective when y is a fuzzy membership matrix.

Value

Function clustcentroid returns a group-by-species matrix containing species average abundance values (i.e. the coordinates of each cluster centroid). Function clustmedoid returns a vector of indices (medoids).

Note

In order to assign new plot record data into a predefined set of classes, one should use functions as.vegclust and vegclass instead.

Author(s)

Miquel De Cáceres, CREAF

See Also

as.vegclust, vegclass, vegclust, kmeans

Examples

```
## Loads stats
library(stats)

## Loads data
data(wetland)

## This equals the chord transformation
## (see also \code{\link{decostand}} in package 'vegan')
wetland.chord = as.data.frame(sweep(as.matrix(wetland), 1,
                              sqrt(rowSums(as.matrix(wetland)^2)), "/"))

## Performs a K-means clustering
wetland.km = kmeans(wetland.chord, centers=3, nstart=10)

## Gets the coordinates corresponding to the centroids of KM clusters
```

```
clustcentroid(wetland.chord, y=wetland.km$cluster)

## Gets the object indices corresponding to the medoids of KM clusters
clustmedoid(wetland.chord, y=wetland.km$cluster)
```

---

clustconst *Constancy table of a classification*

Description

Allows studying the constancy table (i.e. the frequency of species in each class) of a classification represented in the form of a membership data matrix.

Usage

```
clustconst(x, memb)
## S3 method for class 'clustconst'
summary(object, mode="all", name=NULL, sort=TRUE, minconst=0.5, digits=3, ...)
```

Arguments

x	Community data, a site by species data frame.
memb	An site-by-group matrix indicating the (hard or fuzzy) membership of each object in x to a set of groups.
object	An object of class 'clustconst'.
mode	Use mode="all" to print the constancy table, mode="cluster" to print constancy values for one cluster, and mode="species", to print constancy values for one species.
name	A string with the name of a cluster (in mode="cluster"), or the name of a species (in mode="species").
sort	A flag to indicate whether constancy table should be sorted in descending order.
minconst	A threshold used to limit the values shown.
digits	The number of digits for rounding.
...	Additional parameters for summary (actually not used).

Details

The constancy value of a species in a vegetation unit is the relative frequency of occurrence of the species in plot records that belong to the unit. In case of a fuzzy vegetation unit the constancy value is the sum of memberships of sites that contain the species divided by the sum of memberships of all sites. Use the 'summary' function to obtain information about: (1) which species are more frequent on a given vegetation unit; (2) which vegetation units have higher frequencies of a given target species. Additionally, the 'summary' function can sort a constancy table if mode="all" and sort=TRUE are indicated.

Value

Function `clustconst` returns an object of type 'clustconst', in fact a data frame with the constancy value of each species (rows) on each cluster (column).

Author(s)

Miquel De Cáceres, CREAF

See Also

vegclust, kmeans

Examples

```
## Loads stats
library(stats)

## Loads data
data(wetland)

## This equals the chord transformation
## (see also \code{\link{decostand}} in package 'vegan')
wetland.chord = as.data.frame(sweep(as.matrix(wetland), 1,
                              sqrt(rowSums(as.matrix(wetland)^2)), "/"))

## Performs a K-means clustering
wetland.km = kmeans(wetland.chord, centers=3, nstart=10)

## Gets constancy table of KM (i.e. hard) clusters
c=clustconst(wetland.chord, memb=as.memb(wetland.km$cluster))

## Prints constancy values ordered and store the result in d
d=summary(c, mode="all")

## Prints the most frequent species in the first cluster
summary(c, mode="cluster", name=names(c)[1])
```

---

clustvar *Cluster variance*

Description

Computes the variation in community composition (i.e. beta diversity) found within the sites of a set of hard or fuzzy clusters.

Usage

```
clustvar(x, cluster = NULL, defuzzify=FALSE,...)
```

Arguments

x	Community data. Either a site-by-species matrix or a site-by-site matrix of compositional distances between sites (i.e., an object of class dist). Alternatively, this can be an object of class vegclust or vegclass, and in this case it is unnecessary to provide cluster.
cluster	A vector indicating the hard membership of each object in x to a set of groups.
defuzzify	A flag indicating whether fuzzy memberships should be defuzzified (see function defuzzify). Only applies to the case where an object of class vegclust or vegclass is supplied in x.
...	Additional parameters for function defuzzify.

Details

This function can be used in two ways:

- if x is a data matrix (site by species or distances among sites) and cluster is null, the function assumes a single cluster of all points in x. When cluster is provided, the function computes cluster variance for each (hard) group and this computation implies setting the centroid of the group. Cluster variance is defined as the average squared distance to the centroid.
- If x is an object of class vegclust or vegclass, the function uses the information contained there (distances to cluster centers, memberships and exponent of fuzziness) in order to compute cluster variances. Cluster centers do not need to be recomputed, and the distances to cluster centers are used directly. For centroid-based cluster models (KM, FCM, NC, HNC and PCM) the variance is defined as the average squared distance to the centroid. For medoid-based cluster models (KMdd, FCMdd, NCdd, HNCdd and PCMdd) the variance is defined as average distance to the medoid. The variance for both mobile and fixed clusters is returned. Additionally, membership matrices may be defuzzified if defuzzify=TRUE.

Value

A double value (for one cluster) or a vector of values, one per each cluster.

Author(s)

Miquel De Cáceres, CREAF

See Also

vegclust, kmeans,defuzzify

Examples

```
## Loads data
data(wetland)

## This equals the chord transformation
## (see also \code{\link{decostand}} in package 'vegan')
wetland.chord = as.data.frame(sweep(as.matrix(wetland), 1,
                              sqrt(rowSums(as.matrix(wetland)^2)), "/"))
```

```
## Create noise clustering with 3 clusters. Perform 10 starts from random seeds
## and keep the best solution
wetland.nc = vegclust(wetland.chord, mobileCenters=3, m = 1.2, dnoise=0.75,
                      method="NC", nstart=10)

## Gets cluster variance of fuzzy clusters
clustvar(wetland.nc)

## Gets cluster variance of fuzzy clusters after defuzzification
clustvar(wetland.nc, defuzzify=TRUE)

## Similar to the previous, this gets cluster variance of defuzzified (i.e. hard) clusters
clustvar(wetland.chord, cluster=defuzzify(wetland.nc)$cluster)

## Gets cluster variance of K-means (i.e. hard) clusters
clustvar(wetland.chord, cluster=kmeans(wetland.chord, centers=3, nstart=10)$cluster)
```

`concordance` *Concordance between two classifications*

Description

Computes an index to compare two classifications.

Usage

```
concordance(x, y, method="adjustedRand",...)
```

Arguments

`x,y`	Classification vector or membership matrix. Alternatively, objects of type `vegclust` or `vegclass`.
`method`	A string vector to indicate the desired indices (see details).
`...`	Additional parameters for function `defuzzify`, which will be called if `x` or `y` are of type `matrix`, `vegclust` or `vegclass`.

Details

Several indices for comparison of partitions are available:

- `method="Rand"`: Rand (1971) index.
- `method="adjustedRand"`: Rand index adjusted for random effects (Hubert & Arabie 1985).
- `method="Wallace"`: Wallace (1983) index (for asymmetrical comparisons). This index (and its adjusted version) is useful to quantify how much `x` is nested into `y`.
- `method="adjustedWallace"`: Wallace index adjusted for random effects (Pinto et al. 2008).

Value

A numeric vector with the desired index values.

Author(s)

Miquel De Cáceres, CREAF

References

Hubert, L. & Arabie, P. (1985). Comparing partitions. Journal of Classification, 2, 193–218.

Pinto, F.R., Melo-Cristino, J. & Ramirez, M. (2008). A confidence interval for the wallace coefficient of concordance and its application to microbial typing methods. PLoS ONE, 3.

Rand, W.M. (1971). Objective Criteria for the Evaluation of Clustering Methods. Journal of the American Statistical Association, 66, 846–850.

Wallace, D.L. (1983). A method for comparing two hierarchical clusterings: Comment. Journal of the American Statistical Association, 78, 569–576.

See Also

vegclust, vegclass, defuzzify

---

conformveg *Conform two community data tables*

Description

Conforms two community data tables to have the same set of columns (species)

Usage

```
conformveg(x, y, fillvalue = 0, verbose=FALSE)
```

Arguments

x	Community data, a site-by-species matrix.
y	Community data, a site-by-species matrix.
fillvalue	The value to be used to fill new entries in inflated matrices.
verbose	Displays information about the number of species shared between x and y, as well as the number of species that are in one of the data tables but not in the other.

Details

This function adds to x as many new columns as columns of y that are not in x. The same is done for y, so the two tables have the same set of columns when they are returned.

Value

A list with the two inflated matrices x and y.

Author(s)

Miquel De Cáceres, Centre Tecnologic Forestal de Catalunya.

See Also

vegclust, vegclass

Examples

```
## Loads data (38 columns and 33 species)
data(wetland)
dim(wetland)

## Splits wetland data into two matrices of 30x27 and 11x22
wetland.30 = wetland[1:30,]
wetland.30 = wetland.30[,colSums(wetland.30)>0]
dim(wetland.30)
wetland.11 = wetland[31:41,]
wetland.11 = wetland.11[,colSums(wetland.11)>0]
dim(wetland.11)

## Conforms the two matrices so they can eventually be merged
wetland.cf = conformveg(wetland.30, wetland.11)
dim(wetland.cf$x)
dim(wetland.cf$y)
names(wetland.cf$x)==names(wetland.cf$y)
```

---

crossmemb *Cross-table of two fuzzy classifications*

Description

Calculates a cross-tabulated matrix relating two fuzzy membership matrices

Usage

```
crossmemb(x, y, relativize = TRUE)
```

Arguments

x	A site-by-group fuzzy membership matrix. Alternatively, an object of class 'vegclust' or 'vegclass'.
y	A site-by-group fuzzy membership matrix. Alternatively, an object of class 'vegclust' or 'vegclass'.
relativize	If TRUE expresses the cross-tabulated values as proportions of cluster size in x.

Value

A cross-tabulated matrix comparing the two classifications. In general, each cell's value is the (fuzzy) number of objects that in `x` are assigned to the cluster corresponding to the row and in `y` are assigned to the cluster corresponding to the column. If `relativize=TRUE` then the values of each row are divided by the (fuzzy) size of the corresponding cluster in `x`.

Author(s)

Miquel De Cáceres, CREAF.

See Also

`defuzzify`, `vegclust`

Examples

```
## Loads data
data(wetland)

## This equals the chord transformation
## (see also  \code{\link{decostand}} in package vegan)
wetland.chord = as.data.frame(sweep(as.matrix(wetland), 1,
                              sqrt(rowSums(as.matrix(wetland)^2)), "/"))


## Create clustering with 3 clusters. Perform 10 starts from random seeds
## and keep the best solution. Try both FCM and NC methods:
wetland.fcm = vegclust(wetland.chord, mobileCenters=3, m = 1.2, method="FCM", nstart=10)
wetland.nc = vegclust(wetland.chord, mobileCenters=3, m = 1.2, dnoise=0.75, method="NC",
                      nstart=10)

## Compare the results
crossmemb(wetland.fcm, wetland.nc, relativize=FALSE)
```

`defuzzify` *Defuzzifies a fuzzy partition*

Description

Transforms a fuzzy classification into a crisp (hard) classification.

Usage

```
defuzzify(object, method = "max", alpha = 0.5, na.rm = FALSE)
```

Arguments

object	A site-by-group fuzzy membership matrix. Alternatively, an object of class 'vegclust' or 'vegclass'.
method	Either "max" to choose for the maximum membership value across clusters, or "cut" for an alpha-cut.
alpha	Threshold for the alpha-cut, bounded between 0 and 1.
na.rm	If TRUE removes the objects that do not belong to any cluster when using method="cut".

Details

Alpha-cut means that memberships lower than alpha are transformed into 0 while memberships higher than alpha are transformed into 1. This means that if alpha values are low (i.e. close to 0), an object may belong to more than one group after defuzzification. These will generate a concatenation of cluster names in the output cluster vector and a row with sum more than one in the memb matrix). Similarly, if alpha is high (i.e. close to 1) there are objects that may be left unclassified. These will get NA in the cluster vector and zero row in the memb matrix.

Value

A list with the following items:

memb	A data frame with the hard membership partition.
cluster	A vector (factor) with the name of the cluster for each object.

Author(s)

Miquel De Cáceres, CREAF.

References

Davé, R. N. and R. Krishnapuram (1997) Robust clustering methods: a unified view. IEEE Transactions on Fuzzy Systems 5, 270-293.

See Also

vegclust

Examples

```
## Loads data
data(wetland)

## This equals the chord transformation
## (see also  \code{\link{decostand}} in package vegan)
wetland.chord = as.data.frame(sweep(as.matrix(wetland), 1,
                              sqrt(rowSums(as.matrix(wetland)^2)), "/"))

## Create noise clustering with 3 clusters. Perform 10 starts from random seeds
## and keep the best solution
```

```
wetland.nc = vegclust(wetland.chord, mobileCenters=3, m = 1.2, dnoise=0.75,
                      method="NC", nstart=10)

## Defuzzification using an alpha-cut (alpha=0.5)
wetland.nc.df = defuzzify(wetland.nc$memb, method="cut")

## Cluster vector, with 'N' for objects that are unclassified,
## and 'NA' for objects that are intermediate
print(wetland.nc.df$cluster)

## Hard membership matrix (site 22 does not get any cluster assigned)
print(wetland.nc.df$memb)
```

---

hcr *Heterogeneity-constrained random resampling (HCR)*

Description

Returns a set of indices of the original data set that maximizes the mean and minimizes the variance of the distances between pairs of plot records.

Usage

```
hcr(d, nout, nsampl=1000)
```

Arguments

d	An object of class dist containing the distance values between pairs of sites (plot records).
nout	The number of sites (plot records) to be chosen among those available in d.
nsampl	The number of resampling trials to be compared.

Details

Many subsets of the input data are selected randomly. These subsets are sorted by decreasing mean dissimilarity between pairs of plot records, and then sorted again by increasing variance of these dissimilarities. Ranks from both sortings are summed for each subset, and the subset with the lowest summed rank is considered as the most representative.

Value

Returns a vector containing the indices of the selected sites (plot records) to be used for sub-setting the original table.

Author(s)

Miquel De Cáceres, CREAF

References

Lengyel, A., Chytry, M., Tichy, L. (2011) Heterogeneity-constrained random resampling of phytosociological databases. Journal of Vegetation Science 22: 175-183.

Examples

```
## Loads data (38 columns and 33 species)
data(wetland)
dim(wetland)

## Constructs the chord distance matrix
## (see also  \code{\link{decostand}} in package vegan)
wetland.chord =dist(as.data.frame(sweep(as.matrix(wetland), 1,
                   sqrt(rowSums(as.matrix(wetland)^2)), "/")))

## Performs HCR resampling. Returns indices of objects
sel = hcr(wetland.chord, nout=20, nsampl=1000)

## Prints the names of the plot records
print(row.names(wetland)[sel])

## Subset the original distance matrix
sel.chord = as.dist(as.matrix(wetland.chord)[sel,sel])
```

---

hier.vegclust *Clustering with several number of clusters*

Description

Performs several runs of function 'vegclust' (or 'vegclustdist') on a community data matrix (or distance matrix) using different number of clusters

Usage

```
hier.vegclust(x, hclust, cmin=2,cmax=20, min.size=NULL, verbose=TRUE, ...)
hier.vegclustdist(x, hclust, cmin=2,cmax=20, min.size=NULL, verbose=TRUE, ...)
random.vegclust(x, cmin=2, cmax=20, nstart=10, min.size=NULL, verbose=TRUE, ...)
random.vegclustdist(x, cmin=2, cmax=20, nstart=10, min.size=NULL, verbose=TRUE, ...)
```

Arguments

x	For `hier.vegclust` and random.vegclust, a site (rows) by species (columns) matrix or data frame. For `hier.vegclustdist` and random.vegclustdist, a square distance matrix.
hclust	A hierarchical clustering represented in an object of type `hclust`.
cmin	Number of minimum mobile clusters.
cmax	Number of maximum mobile clusters.

nstart	A number indicating how many random trials should be performed for each number of groups
min.size	If min.size != NULL, it specifies the minimum size of clusters. If some clusters are smaller, the algorithm will return the solutions corresponding to lower numbers of clusters.
verbose	Flag used to print which number of clusters is currently running.
...	Additional parameters for function vegclust or vegclustdist.

Details

Function hier.vegclust takes starting cluster configurations from cuts of a dendrogram given by object hclust. Function random.vegclust chooses random objects as cluster centroids and for each number of clusters performs nstart trials. Functions hier.vegclustdist and random.vegclustdist are analogous to hier.vegclust and random.vegclust but accept distance matrices as input.

Value

Returns an object of type 'mvegclust' (multiple vegclust), which contains a list vector with objects of type vegclust.

Author(s)

Miquel De Cáceres, CREAF

See Also

vegclust, vegclustdist, vegclass, defuzzify, hclust

Examples

```
## Loads data
data(wetland)

## This equals the chord transformation
## (see also  \code{\link{decostand}} in package vegan)
wetland.chord = as.data.frame(sweep(as.matrix(wetland), 1,
                              sqrt(rowSums(as.matrix(wetland)^2)), "/"))

## Create noise clustering from hierarchical clustering at different number of clusters
wetland.hc = hclust(dist(wetland.chord),method="ward")
wetland.nc1 = hier.vegclust(wetland.chord, wetland.hc, cmin=2, cmax=5,
                            m = 1.2, dnoise=0.75, method="NC")

## Create noise clustering from random seeds at different levels
wetland.nc2 = random.vegclust(wetland.chord, cmin=2, cmax=5, nstart=10,
                              m = 1.2, dnoise=0.75, method="NC")
```

`incr.vegclust` *Noise clustering with increasing number of clusters*

Description

Performs several runs of function 'vegclust' on a community data matrix using an increasing number of clusters until some conditions are met.

Usage

```
incr.vegclust(x, method="NC", ini.fixed.centers = NULL,
              min.size = 10, max.var=NULL, alpha = 0.5,
              nstart=100, fix.previous = TRUE, dnoise=0.75, m=1.0,...)
```

Arguments

`x`	Community data table. A site (rows) by species (columns) matrix or data frame.
`method`	A clustering model. Current accepted models are of the noise clustering family: • `"NC"`: Noise clustering (Dave and Krishnapuram 1997) • `"NCdd"`: Noise clustering with medoids • `"HNC"`: Hard noise clustering • `"HNCdd"`: Hard noise clustering with medoids
`ini.fixed.centers`	The coordinates of initial fixed cluster centers. These will be used as `fixedCenters` in all calls to `vegclust`. If `method="NCdd"` or `method="HNCdd"` then `ini.fixed.centers` can be specified as a vector of indices for medoids.
`min.size`	The minimum size (cardinality) of clusters. If any of the current k clusters does not have enough members the algorithm will stop and return the solution with k-1 clusters.
`max.var`	The maximum variance allowed for clusters (see function `clustvar`). If specified, the algorithm will stop when any of the clusters is at the same time small and has large variance. If `max.var = NULL` then this criterion is not used.
`alpha`	Criterion to choose cluster seeds from the noise class. Specifically, an object is considered as cluster seed if the membership to the noise class is larger than `alpha`.
`nstart`	A number indicating how many random trials should be performed for number of groups. Each random trial uses the k-1 cluster centers plus the coordinates of the current cluster seed as initial solution for `vegclust`. Thus, if there are less cluster seed candidates than `nstart`, then not all runs are conducted.
`fix.previous`	Flag used to indicate that the cluster centers found when determining k-1 clusters are fixed when determining k clusters.
`m`	The fuzziness exponent.
`dnoise`	The distance to the noise cluster.
`...`	Additional parameters for function `vegclust`.

Details

Function `hier.vegclust` takes starting cluster configurations from cuts of a dendrogram given by object `hclust`. Function `random.vegclust` chooses random objects as cluster centroids and for each number of clusters performs `nstart` trials.

Value

Returns an object of class `vegclust`; or `NULL` if the initial cluster does not contain enough members.

Author(s)

Miquel De Cáceres, CREAF

References

Davé, R. N. and R. Krishnapuram (1997) Robust clustering methods: a unified view. IEEE Transactions on Fuzzy Systems 5, 270-293.

See Also

`vegclust`,`hier.vegclust`

Examples

```
## Loads data
data(wetland)

## This equals the chord transformation
## (see also \code{\link{decostand}} in package 'vegan')
wetland.chord = as.data.frame(sweep(as.matrix(wetland), 1,
                              sqrt(rowSums(as.matrix(wetland)^2)), "/"))

## Call incremental noise clustering
wetland.nc = incr.vegclust(wetland.chord, method="NC", m = 1.2, dnoise=0.75,
                           min.size=5)

## Inspect cluster sizes
print(wetland.nc$size)
```

`interclustdist` *Calculates the distance between pairs of cluster centroids*

Description

Calculates the distance between pairs of cluster centroids, given a distance matrix and a cluster vector.

Usage

```
interclustdist(x, cluster)
```

Arguments

x	A site-by-site data matrix or an object of class dist containing the distance values between pairs of sites (plot records).
cluster	A vector indicating the hard membership of each object in x to a set of groups. Can contain NA values.

Value

An object of class dist containing the distances between cluster centers.

Author(s)

Miquel De Cáceres, CREAF

Examples

```
##TO BE DONE##
```

---

medreg *Regeneration of Mediterranean vegetation data set*

Description

A stratified vegetation data set containing with several plot records laid to assess vegetation recovery three years after a wildfire. Collected in 2012 by Miquel De Caceres and Albert Petit in Horta de Sant Joan (Catalonia, Spain).

Usage

```
data(medreg)
```

Format

An object of class 'stratifiedvegdata' with 96 elements (plots), each of them consisting of a data.frame where rows correspond to species groups and columns correspond to vegetation strata. Abundance values are percentage cover.

See Also

CAP, plot.CAP

Examples

```
data(medreg)
```

plot.CAP *Draws cummulative abundance profiles*

Description

Create plots used to inspect one or more cumulative abundance profiles.

Usage

```
## S3 method for class 'CAP'
plot(x, sizes=NULL, species=NULL, plots=NULL, switchAxes=FALSE,
                  add=FALSE, drawAxes = TRUE, xlab="", ylab="", type="s",...)
## S3 method for class 'stratifiedvegdata'
plot(x, sizes=NULL, species=NULL, plots=NULL, switchAxes=FALSE,
                  add=FALSE, drawAxes = TRUE, xlab="", ylab="", type="s",...)
```

Arguments

x	An object returned from function CAP or an object of class stratifiedvegdata (see documentation for function stratifyvegdata).
sizes	A vector containing the size values associated to each size class. If NULL the y-axis will be defined using the size class order in x.
species	A vector of strings indicating the species whose profile is to be drawn. If NULL all species are plotted.
plots	A vector indicating the plot records whose profile is to be drawn. Can be a character vector (for plot names), a numeric vector (for plot indices) or a logical vector (for TRUE/FALSE selection). If NULL all plot records are plotted.
switchAxes	A flag indicating whether ordinate and abscissa axes should be interchanged.
add	A flag indicating whether profiles should be drawn on top of current drawing area. If add=FALSE a new plot is created.
drawAxes	A flag indicating whether axes should be drawn.
xlab	String label for the x axis.
ylab	String label for the y axis.
type	Type of plot to be drawn ("p" for points, "l" for lines, "s" for steps, ...).
...	Additional plotting parameters.

Author(s)

Miquel De Cáceres, CREAF

References

De Cáceres, M., Legendre, P. & He, F. (2013) Dissimilarity measurements and the size structure of ecological communities. Methods in Ecology and Evolution 4: 1167-1177.

See Also

CAP

Examples

```
## Load stratified data
data(medreg)

## Check that 'medreg' has correct class
class(medreg)

## Create cumulative abundance profile (CAP) for each plot
medreg.CAP = CAP(medreg)

## Draw the stratified data and profile corresponding to the third plot
plot(medreg, plots="3")
plot(medreg.CAP, plots="3")

## Look at the plot and CAP of the same plot
medreg[["3"]]
medreg.CAP[["3"]]
```

---

plot.CAS *Draws a cummulative abundance surface*

Description

Create plots used to inspect one or more cumulative abundance profiles.

Usage

```
## S3 method for class 'CAS'
plot(x, plot=NULL, species=NULL, sizes1=NULL, sizes2 = NULL,
               palette = colorRampPalette(c( "light blue","light green","white",
                   "yellow","orange","red")), zlim=NULL,...)
```

Arguments

x	An object of class CAS.
plot	A string indicating the plot record whose surface is to be drawn.
species	A string indicating the species whose profile is to be drawn.
sizes1	A vector containing the size values associated to each primary size class. If NULL the x-axis will be defined using the primary size class order in x.
sizes2	A vector containing the size values associated to each secondary size class. If NULL the y-axis will be defined using the secondary size class order in x.
palette	Color palette for z values.
zlim	The limits for the z-axis.
...	Additional plotting parameters for function persp.

Author(s)

Miquel De Cáceres, CREAF

References

De Cáceres, M., Legendre, P. & He, F. (2013) Dissimilarity measurements and the size structure of ecological communities. Methods in Ecology and Evolution 4: 1167-1177.

See Also

CAS, persp

Examples

```
## Create synthetic tree data
pl = rep(1,100) # All trees in the same plot
sp = ifelse(runif(100)>0.5,1,2) # Random species identity (species 1 or 2)
h=rgamma(100,10,2) # Heights (m)
d = rpois(100, lambda=h^2) # Diameters (cm)
m = data.frame(plot=pl,species=sp, height=h,diameter=d)
m$ba = pi*(m$diameter/200)^2
print(head(m))

## Size classes
heights = seq(0,4, by=.25)^2 # Quadratic classes
diams = seq(0,130, by=5) # Linear classes

## Stratify tree data
X<-stratifyvegdata(m, sizes1=heights, sizes2=diams,
                   plotColumn = "plot", speciesColumn = "species",
                   size1Column = "height", size2Column = "diameter",
                   abundanceColumn = "ba")


## Build cummulative abundance surface
Y = CAS(X)

## Plot the surface of species '1' in plot '1' using heights and diameters
plot(Y, species=1, sizes1=heights[-1], xlab="height (m)",
     ylab="diameter (cm)", sizes2=diams[-1], zlab="Basal area (m2)",
     zlim = c(0,6), main="Species 1")
```

---

plot.mvegclust *Plots clustering results*

Description

Create plots used to study vegclust clustering results for an increasing number of clusters

Usage

```
## S3 method for class 'mvegclust'
plot(x, type="hnc", excludeFixed=TRUE, verbose=FALSE, ylim=NULL,
                          xlab=NULL, ylab=NULL, maxvar=0.6, minsize=20,...)
```

Arguments

x	An object returned from functions hier.vegclust or random.vegclust.
type	A string indicating the type of plot desired. Current accepted values are "hnc","hmemb","var","hcs" and "valid".
excludeFixed	A flag to indicate whether clusters with fixed centroids should be excluded from plots.
verbose	A flag to print extra information.
ylim	A vector with the limits for the y axis.
xlab	String label for the x axis.
ylab	String label for the y axis.
maxvar	Maximum cluster variance allowed for the type="valid" plot.
minsize	Minimum cluster size allowed for the type="valid" plot.
...	Additional plotting parameters.

Value

Different information is returned depending on the type of plot chosen.

Author(s)

Miquel De Cáceres, CREAF

Examples

```
## Loads data
data(wetland)

## This equals the chord transformation
## (see also \code{\link{decostand}} in package 'vegan')
wetland.chord = as.data.frame(sweep(as.matrix(wetland), 1,
                               sqrt(rowSums(as.matrix(wetland)^2)), "/"))

## Create noise clustering from hierarchical clustering at different number of clusters
wetland.hc = hclust(dist(wetland.chord),method="ward")
wetland.nc = hier.vegclust(wetland.chord, wetland.hc, cmin=2, cmax=5, m = 1.2,
                           dnoise=0.75, method="NC")

## Plot changes in the number of objects falling into the noise cluster
plot(wetland.nc, type="hnc")

## Plots the number of objects falling into "true" clusters,
```

```
## the number of objects considered intermediate,
## and the number of objects falling into the noise
plot(wetland.nc, type="hmemb")

## Plot minimum, maximum and average cluster size
plot(wetland.nc, type="hcs")

## Plot minimum, maximum and average cluster variance
plot(wetland.nc, type="var")

## Plot number of groups with high variance, low membership or both
plot(wetland.nc, type="valid")
```

`relate.levels` *Relates two clustering level results.*

Description

Analyzes how lower level clusters are assigned into upper level ones. The analysis is made for several number of clusters.

Usage

```
relate.levels(lower, upper, defuzzify = FALSE, excludeFixed = FALSE, verbose=FALSE, ...)
```

Arguments

`lower`	A list of objects of type `vegclust` or `vegclass` that represent classifications at a finer level of resolution.
`upper`	A list of objects of type `vegclust` or `vegclass` that represent classifications at an broader level of resolution.
`defuzzify`	A logical flag used to indicate whether the result of calling `crossmemb` should be deffuzified.
`excludeFixed`	A logical used to indicate whether fixed clusters should be excluded from the comparison of levels.
`verbose`	A flag used to ask for extra screen output.
`...`	Additional parameters for function `defuzzify`.

Details

For each pair of `vegclust` (or `vegclass`) objects in `upper` and `lower`, the function calls function `crossmemb` and then, if asked, deffuzifies the resulting memberships (by calling function `defuzzify`) and several quantities are calculated (see 'value' section).

Value

A list with several data frames (see below). In each of them, the rows are items of `upper` and columns are items of `lower`. The names of rows and columns are the number of clusters of each `vegclust` (or `vegclass`) object.

`nnoise`	The number of low level clusters that are assigned to the Noise class (for `upper` objects using Noise clustering).
`maxnoise`	The maximum membership value of low level clusters to the Noise class (for `upper` objects using Noise clustering).
`minmaxall`	The minimum value (across upper level clusters) of the maximum membership value observed among the lower level clusters.
`minallsize`	The minimum value (across upper level clusters) of the sum of membership values across lower level clusters.
`empty`	The number of upper level clusters (mobile or fixed) that do not have any member among the lower level clusters.

Author(s)

Miquel De Cáceres, CREAF

See Also

`vegclust`, `vegclass`, `defuzzify`

Examples

```
## Loads data
data(wetland)

## This equals the chord transformation
## (see also  \code{\link{decostand}} in package vegan)
wetland.chord = as.data.frame(sweep(as.matrix(wetland), 1,
                              sqrt(rowSums(as.matrix(wetland)^2)), "/"))

## Create noise clustering from hierarchical clustering at different number of cluster
wetland.hc = hclust(dist(wetland.chord),method="ward")
wetland.nc1 = hier.vegclust(wetland.chord, wetland.hc, cmin=2, cmax=6, m = 1.2,
                            dnoise=0.75, method="NC")
wetland.nc2 = hier.vegclust(wetland.chord, wetland.hc, cmin=2, cmax=4, m = 1.2,
                            dnoise=0.85, method="NC")

## Studies the assignment of levels
relate.levels(wetland.nc1, wetland.nc2, method="cut")
```

stratifyvegdata *Reshapes community data from individual into stratified form*

Description

Function stratifyvegdata reshapes individual abundance values into species abundance values per size class or combination of size classes. Function as.stratifiedvegdata checks if the input list has appropriate properties and turns it into an object of class 'stratifiedvegdata'.

Usage

```
stratifyvegdata(x, sizes1, sizes2 = NULL, treeSel=NULL, spcodes = NULL,
                plotColumn="plot", speciesColumn = "species",
           abundanceColumn="abundance", size1Column = "size", size2Column = NULL,
              cumulative=FALSE, counts=FALSE, mergeSpecies=FALSE, verbose=FALSE)
as.stratifiedvegdata(X)
```

Arguments

x	A data frame containing individual plant data. Individuals are in rows, while measurements are in columns.
sizes1	A numerical vector containing the breaks for primary size classes in ascending order.
sizes2	A numerical vector containing the breaks for secondary size classes in ascending order.
treeSel	A logical vector specifying which rows in x to be used. By default (treeSel = NULL) all rows are taken.
spcodes	A character vector indicating the codes of species to be used for stratification (species codes beyond those appearing in x are possible). If spcodes = NULL then all species in x are used.
plotColumn	The name of the column in x that contains plot identifiers.
speciesColumn	The name of the column in x that contains species names.
abundanceColumn	The name of the column in x that contains abundance values.
size1Column	The name of the column in x that contains values for primary size classes.
size2Column	The name of the column in x that contains values for secondary size classes.
cumulative	A flag to indicate that cumulative abundance profiles or surfaces are desired.
counts	A flag to indicate that the output should be individual counts instead of added abundance values.
mergeSpecies	A flag to indicate that species identity should be ignored. This leads to analyzing the structure of biomass disregarding species identity.
verbose	A logical flag to indicate extra output.

X — A list with as many elements as plot records. Each element should be of class 'matrix' or 'data.frame' with species in rows and strata in columns. Furthermore, the number of rows (species) and columns (strata) should be the same for all elements.

Details

For each individual (row) in x, stratifyvegdata assigns it to the size class (stratum) containing its size. The corresponding abundance value (e.g. crown cover) of the individual is added to the abundance of the corresponding species at the size class (stratum). If sizes2 and size2Column are supplied, the function assigns each individual (row) in x to the combination of size classes (e.g. tree height and diameter).

Value

Both functions return an object of class 'stratifiedvegdata', which is a list of matrices, one for each plot record. Each element (matrix) has as many rows as species and as many columns as size classes (i.e., as many as elements in vector sizes1). Columns are named starting with 'S' and continuing with the size class (stratum) number. If mergeSpecies=TRUE then all matrices have a single row (whose name is "all"). If sizes2 and size2Column are supplied to stratifyvegdata, the function returns an object of class 'doublestratifiedvegdata', which is a list of arrays, one for each plot record. Each element (array) has three dimensions corresponding to species, primary sizes (number of elements in in vector sizes1) and secondary sizes (number of elements in in vector sizes2). If cumulative=TRUE then the function returns cumulative abundances (see CAP and CAS).

Author(s)

Miquel De Cáceres, CREAF.

References

De Cáceres, M., Legendre, P. & He, F. (2013) Dissimilarity measurements and the size structure of ecological communities. Methods in Ecology and Evolution 4: 1167-1177.

See Also

reshape, CAP, CAS

Examples

```
## Load tree data
data(treedata)

## Inspect tree data
head(treedata)

## Define stratum thresholds (4 strata)
heights = seq(0,4, by=0.5)
diameters = seq(0,2, by=0.5)

## Stratify tree data using heights as structural variable
```

```
X = stratifyvegdata(treedata, sizes1=heights, plotColumn="plotID",
                    speciesColumn="species", size1Column="height", counts=TRUE)

## Inspect the second plot record
X[[2]]

## Stratify tree data using heights as structural variable and cover as abundance
Y = stratifyvegdata(treedata, sizes1=heights, plotColumn="plotID",
                    speciesColumn="species", size1Column="height",
                    abundanceColumn="cover")
Y[[2]]

## Stratify tree data using heights and diameters as structural variables
Z = stratifyvegdata(treedata, sizes1=heights, sizes2=diameters, plotColumn="plotID",
                    speciesColumn="species", size1Column="height", size2Column="diam",
                    counts=TRUE)
Z[[2]]
```

`treedata` *Synthetic vegetation data set with tree data*

Description

A synthetic data set used to illustrate the stratification of data originally collected on an individual basis (e.g. forest inventory).

Usage

```
data(treedata)
```

Format

A data frame where each row corresponds to a different tree. Columns are plot code, species identity, tree height, tree diameter and cover value.

See Also

`stratifyvegdata`

vegclass *Classifies vegetation communities*

Description

Classifies vegetation communities into a previous fuzzy or hard classification.

Usage

```
vegclass(y, x)
```

Arguments

y	An object of class vegclust that represents a previous knowledge.
x	Community data to be classified, in form of a site by species matrix (if the vegclust object is in raw mode) or a data frame containing the distances between the new sites in rows and the old sites in columns (if the vegclust object is in distance mode).

Details

This function uses the classification model specified in y to classify the communities (rows) in x. When vegclust is in raw mode, the function calls first to conformveg in order to cope with different sets of species. See the help of as.vegclust to see an example of vegclass with distance matrices.

Value

Returns an object of type vegclass with the following items:

method	The clustering model used in y
m	The fuzziness exponent in y
dnoise	The distance to the noise cluster used for noise clustering (models NC, NCdd, HNC, HNCdd). This is set to NULL for other models.
eta	The reference distance vector used for possibilistic clustering (models PCM and PCMdd). This is set to NULL for other models.
memb	The fuzzy membership matrix.
dist2clusters	The matrix of object distances to cluster centers.

Author(s)

Miquel De Cáceres, CREAF.

References

Davé, R. N. and R. Krishnapuram (1997) Robust clustering methods: a unified view. IEEE Transactions on Fuzzy Systems 5, 270-293.

Bezdek, J. C. (1981) Pattern recognition with fuzzy objective functions. Plenum Press, New York.

Krishnapuram, R. and J. M. Keller. (1993) A possibilistic approach to clustering. IEEE transactions on fuzzy systems 1, 98-110.

De Cáceres, M., Font, X, Oliva, F. (2010) The management of numerical vegetation classifications with fuzzy clustering methods [Related software]. Journal of Vegetation Science 21 (6): 1138-1151.

See Also

vegclust, as.vegclust, kmeans, conformveg

Examples

```
## Loads data (38 columns and 33 species)
data(wetland)
dim(wetland)

## This equals the chord transformation
## (see also  \code{\link{decostand}} in package vegan)
wetland.chord = as.data.frame(sweep(as.matrix(wetland), 1,
                              sqrt(rowSums(as.matrix(wetland)^2)), "/"))

## Splits wetland data into two matrices of 30x27 and 11x22
wetland.30 = wetland.chord[1:30,]
wetland.30 = wetland.30[,colSums(wetland.30)>0]
dim(wetland.30)
wetland.11 = wetland.chord[31:41,]
wetland.11 = wetland.11[,colSums(wetland.11)>0]
dim(wetland.11)

## Create noise clustering with 3 clusters from the data set with 30 sites.
wetland.30.nc = vegclust(wetland.30, mobileCenters=3, m = 1.2, dnoise=0.75,
                         method="NC", nstart=10)

## Cardinality of fuzzy clusters (i.e., the number of objects belonging to)
wetland.30.nc$size

## Classifies the second set of sites according to the clustering of the first set
wetland.11.nc = vegclass(wetland.30.nc, wetland.11)

## Fuzzy membership matrix
wetland.11.nc$memb

## Obtains hard membership vector, with 'N' for objects that are unclassified
defuzzify(wetland.11.nc$memb)$cluster
```

vegclust *Vegetation clustering methods*

Description

Performs hard or fuzzy clustering of vegetation data

Usage

```
vegclust(x, mobileCenters, fixedCenters = NULL, method="NC", m = 2, dnoise = NULL,
       eta = NULL, alpha=0.001, iter.max=100, nstart=1, maxminJ = 10, seeds=NULL,
        verbose=FALSE)
vegclustdist(x, mobileMemb, fixedDistToCenters = NULL, method="NC", m = 2, dnoise = NULL,
        eta = NULL, alpha=0.001, iter.max=100, nstart=1, seeds=NULL, verbose=FALSE)
```

Arguments

x	Community data. A site-by-species matrix or data frame (for vegclust) or a site-by-site dissimilarity matrix or dist object (for vegclustdist).
mobileCenters	A number, a vector of seeds, or coordinates for mobile clusters.
fixedCenters	A matrix or data frame with coordinates for fixed (non-mobile) clusters.
mobileMemb	A number, a vector of seeds, or starting memberships for mobile clusters.
fixedDistToCenters	A matrix or data frame with the distances to fixed cluster centers.
method	A clustering model. Current accepted models are: • "KM": K-means or hard c-means (MacQueen 1967) • "KMdd": Hard c-medoids (Krishnapuram et al. 1999) • "FCM": Fuzzy c-means (Bezdek 1981) • "FCMdd": Fuzzy c-medoids (Krishnapuram et al. 1999) • "NC": Noise clustering (Dave and Krishnapuram 1997) • "NCdd": Noise clustering with medoids • "HNC": Hard noise clustering • "HNCdd": Hard noise clustering with medoids • "PCM": Possibilistic c-means (Krishnapuram and Keller 1993) • "PCMdd": Possibilistic c-medoids
m	The fuzziness exponent to be used (this is relevant for all models except for kmeans)
dnoise	The distance to the noise cluster, relevant for noise clustering (NC).
eta	A vector of reference distances, relevant for possibilistic C-means (PCM).
alpha	Threshold used to stop iterations. The maximum difference in the membership matrix of the current vs. the previous iteration will be compared to this value.
iter.max	The maximum number of iterations allowed.

nstart	If mobileCenters or mobileMemb is a number, how many random sets should be chosen?
maxminJ	When random starts are used, these will stop if at least maxminJ runs ended up in the same functional value.
seeds	If mobileCenters or mobileMemb is a number, a vector indicating which objects are potential initial centers. If NULL all objects are valid seeds.
verbose	Flag to print extra output.

Details

Functions vegclust and vegclustdist try to generalize the kmeans function in stats in three ways.

Firstly, they allows different clustering models. Clustering models can be divided in (a) fuzzy or hard; (b) centroid-based or medoid-based; (c) Partitioning (KM and FCM family), noise clustering (NC family), and possibilistic clustering (PCM and PCMdd). The reader should refer to the original publications to better understand the differences between models.

Secondly, users can specify fixed clusters (that is, centroids that do not change their positions during iterations). Fixed clusters are intended to be used when some clusters were previously defined and new data has been collected. One may allow some of these new data points to form new clusters, while some other points will be assigned to the original clusters. In the case of models with cluster repulsion (such as KM, FCM or NC) the new (mobile) clusters are not allowed to 'push' the fixed ones. As a result, mobile clusters will occupy new regions of the reference space.

Thirdly, vegclustdist implements the distance-based equivalent of vegclust. The results of vegclust and vegclustdist will be the same (if seeds are equal) if the distance matrix is calculated using the Euclidean distance (see function dist). Otherwise, the equivalence holds by resorting on principal coordinates analysis.

Note that all data frames or matrices used as input of vegclust should be defined on the same space of species (see conformveg). Unlike kmeans, which allows different specific algorithms, here updates of prototypes (centroids or medoids) are done after all objects have been reassigned (Forgy 1965). In order to obtain hard cluster definitions, users can apply the function defuzzify to the vegclust object.

Value

Returns an object of type vegclust with the following items:

mode	raw for function vegclust and dist for function vegclustdist.
method	The clustering model used
m	The fuzziness exponent used (m=1 in case of kmeans)
dnoise	The distance to the noise cluster used for noise clustering (NC, HNC, NCdd or HNCdd). This is set to NULL for other models.
eta	The reference distance vector used for possibilistic clustering (PCM or PCMdd). This is set to NULL for other models.
memb	The fuzzy membership matrix. Columns starting with "M" indicate mobile clusters, whereas columns starting with "F" indicate fixed clusters.

mobileCenters	If vegclust is used, this contains a data frame with the coordinates of the mobile centers (centroids or medoids). If vegclustdist is used, it will contain the indices of mobile medoids for models KMdd, FCMdd, HNCdd, NCdd and PCMdd; or NULL otherwise.
fixedCenters	If vegclust is used, this contains a data frame with the coordinates of the fixed centers (centroids or medoids). If vegclustdist is used, it will contain the indices of fixed medoids for models KMdd, FCMdd, HNCdd, NCdd and PCMdd; or NULL otherwise.
dist2clusters	The matrix of object distances to cluster centers. Columns starting with "M" indicate mobile clusters, whereas columns starting with "F" indicate fixed clusters.
withinss	In the case of methods KM, FCM, NC, PCM and HNC it contains the within-cluster sum of squares for each cluster (squared distances to cluster center weighted by membership). In the case of methods KMdd, FCMdd, NCdd, HNCdd and PCMdd it contains the sum of distances to each cluster (weighted by membership).
size	The number of objects belonging to each cluster. In case of fuzzy clusters the sum of memberships is given.
functional	The objective function value (the minimum value attained after all iterations).

Author(s)

Miquel De Cáceres, CREAF

References

Forgy, E. W. (1965) Cluster analysis of multivariate data: efficiency vs interpretability of classifications. Biometrics 21, 768-769.

MacQueen, J. (1967) Some methods for classification and analysis of multivariate observations. In Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, eds L. M. Le Cam and J. Neyman, 1, pp. 281-297. Berkeley, CA: University of California Press.

Davé, R. N. and R. Krishnapuram (1997) Robust clustering methods: a unified view. IEEE Transactions on Fuzzy Systems 5, 270-293.

Bezdek, J. C. (1981) Pattern recognition with fuzzy objective functions. Plenum Press, New York.

Krishnapuram, R., Joshi, A., & Yi, L. (1999). A Fuzzy relative of the k-medoids algorithm with application to web document and snippet clustering. IEEE International Fuzzy Systems (pp. 1281–1286).

Krishnapuram, R. and J. M. Keller. (1993) A possibilistic approach to clustering. IEEE transactions on fuzzy systems 1, 98-110.

De Cáceres, M., Font, X, Oliva, F. (2010) The management of numerical vegetation classifications with fuzzy clustering methods. Journal of Vegetation Science 21 (6): 1138-1151.

See Also

hier.vegclust,incr.vegclust,kmeans,vegclass,defuzzify,clustvar

Examples

```
## Loads data
data(wetland)

## This equals the chord transformation
## (see also 'decostand' in package vegan)
wetland.chord = as.data.frame(sweep(as.matrix(wetland), 1,
                              sqrt(rowSums(as.matrix(wetland)^2)), "/"))

## Create noise clustering with 3 clusters. Perform 10 starts from random seeds
## and keep the best solution
wetland.nc = vegclust(wetland.chord, mobileCenters=3, m = 1.2, dnoise=0.75,
                      method="NC", nstart=10)

## Fuzzy membership matrix
wetland.nc$memb

## Cardinality of fuzzy clusters (i.e., the number of objects belonging to each cluster)
wetland.nc$size

## Obtains hard membership vector, with 'N' for objects that are unclassified
defuzzify(wetland.nc$memb)$cluster

## The same result is obtained with a matrix of chord distances
wetland.d = dist(wetland.chord)
wetland.d.nc = vegclustdist(wetland.d, mobileMemb=3, m = 1.2, dnoise=0.75,
                            method="NC", nstart=10)
```

---

vegclust2kmeans *Reshapes as kmeans object*

Description

This function casts an object of class `vegclust` into an object of class `kmeans`.

Usage

```
vegclust2kmeans(x)
```

Arguments

x	An object of class `vegclust` to be casted, where `method="KM"` and `mode="raw"`.

Author(s)

Miquel De Cáceres, CREAF

See Also

`vegclust`, `kmeans`

Examples

```
## Loads data
data(wetland)

## This equals the chord transformation
## (see also  \code{\link{decostand}} in package vegan)
wetland.chord = as.data.frame(sweep(as.matrix(wetland), 1,
                              sqrt(rowSums(as.matrix(wetland)^2)), "/"))

## Create noise clustering with 3 clusters. Perform 10 starts from random seeds
wetland.vc = vegclust(wetland.chord, mobileCenters=3,
                      method="KM", nstart=10)

## Reshapes as kmeans object
wetland.km = vegclust2kmeans(wetland.vc)
wetland.km
```

`vegclustIndex` *Compute fuzzy evaluation statistics*

Description

Computes several evaluation statistics on the fuzzy clustering results on objects of class `vegclust`.

Usage

```
vegclustIndex(y)
```

Arguments

y	An object of class `vegclust` or a membership matrix.

Details

These statistics were conceived to be computed on fuzzy partitions, such as the ones coming from Fuzzy C-means (Bezdek 1981). Maximum values of PCN or minimum values of PEN can be used as criteria to choose the number of clusters.

Value

Returns an vector of four values: partition coefficient (PC), normalized partition coefficient (PCN), partition entropy (PE) and normalized partition entropy (PEN).

Author(s)

Miquel De Cáceres, CREAF.

References

Bezdek, J. C. (1981) Pattern recognition with fuzzy objective functions. Plenum Press, New York.

See Also

vegclust

Examples

```
## Loads data
data(wetland)

## This equals the chord transformation
## (see also  \code{\link{decostand}} in package vegan)
wetland.chord = as.data.frame(sweep(as.matrix(wetland), 1,
                              sqrt(rowSums(as.matrix(wetland)^2)), "/"))

## Create noise clustering with 2, 3 and 4 clusters. Perform 10 starts from random seeds
## and keep the best solutions
wetland.fcm2 = vegclust(wetland.chord, mobileCenters=2, m = 1.2, method="FCM", nstart=10)
wetland.fcm3 = vegclust(wetland.chord, mobileCenters=3, m = 1.2, method="FCM", nstart=10)
wetland.fcm4 = vegclust(wetland.chord, mobileCenters=4, m = 1.2, method="FCM", nstart=10)

## Compute statistics. Both PCN and PEN indicate that three groups are more advisable
## than 2 or 4.
print(vegclustIndex(wetland.fcm2))
print(vegclustIndex(wetland.fcm3))
print(vegclustIndex(wetland.fcm4))
```

---

vegdiststruct *Structural and compositional dissimilarity*

Description

Function to calculate the dissimilarity between ecological communities taking into account both their composition and the size of organisms.

Usage

```
vegdiststruct(x, y=NULL, paired=FALSE, type="cumulative", method="bray",
              transform=NULL, classWeights=NULL)
```

Arguments

x	A stratified vegetation data set (see function stratifyvegdata), a set of cummulative abundance profiles (see function CAP) or a set of cummulative abundance surfaces (see function CAS).
y	A second stratified vegetation data set (see function stratifyvegdata), a second set of cummulative abundance profiles (see function CAP) or a second set of cummulative abundance surfaces (see function CAS) against which object x should be compared.
paired	Only relevant when y != NULL. If paired = TRUE pairwise comparisons are calculated between elements in x and y (and x and y need to be of the same length). If paired = FALSE then all objects in x are compared to all objects in y.
type	Whether dissimilarities between pairs of sites should be calculated from differences in cummulative abundance ("cumulative"), in total abundance ("total") or in volumes of cumulative abundance profiles ("volume").
method	The dissimilarity coefficient to calculate (see details).
transform	A function or the name of a function to be applied to each cumulative abundance value.
classWeights	A numerical vector or a matrix containing the weight of each size class or combination of size classes (see functions CAP2matrix and CAS2matrix). If NULL, then the function assumes classes of equal weight.

Details

The six different coefficients available are described in De Caceres et al. (2013): (1) method="bray" for percentage difference (alias Bray-Curtis dissimilarity); (2) method="ruzicka" for Ruzicka index (a generalization of Jaccard); (3) method="kulczynski" for the Kulczynski dissimilarity index; (4) method="ochiai" for the complement of a quantitative generalization of Ochiai index of similarity; (5) method="canberra" for the Canberra index (Adkins form); (6) method="relman" for the relativized Manhattan coefficient (Whittaker's index of association). Currently, the function also supports (7) method="manhattan" for the city block metric.

Value

Returns an object of class 'dist'.

Author(s)

Miquel De Cáceres, CREAF.

References

De Cáceres, M., Legendre, P. & He, F. (2013) Dissimilarity measurements and the size structure of ecological communities. Methods in Ecology and Evolution 4: 1167-1177.

See Also

stratifyvegdata, vegdist

Examples

```
## Load stratified data
data(medreg)

## Check that 'medreg' has correct class
class(medreg)

## Create cumulative abundance profile (CAP) for each plot
medreg.CAP = CAP(medreg)

## Create dissimilarity (percentage difference) matrix using profiles
medreg.D = vegdiststruct(medreg, method="bray")

## Create dissimilarity (percentage difference) matrix using abundances
medreg.D2 = vegdiststruct(medreg, method="bray", type="total")

## Calculate correlation
cor(as.vector(medreg.D), as.vector(medreg.D2))
```

---

wetland *Wetland vegetation data set*

Description

Vegetation of the Adelaide river alluvial plain (Australia). This data set was published by Bowman & Wilson (1987) and used in Dale (1988) to compare fuzzy classification approaches.

Usage

```
data(wetland)
```

Format

A data frame with 41 sites (rows) and 33 species (columns). Abundance values are represented in abundance classes.

Source

Bowman, D. M. J. S. and B. A. Wilson. 1986. Wetland vegetation pattern on the Adelaide River flood plain, Northern Territory, Australia. Proceedings of the Royal Society of Queensland 97:69-77.

References

Dale, M. B. 1988. Some fuzzy approaches to phytosociology. Ideals and instances. Folia geobotanica et phytotaxonomica 23:239-274.

Examples

```
data(wetland)
```

Index