Package ‘clusterSim’

September 14, 2024

Title Searching for Optimal Clustering Procedure for a Data Set

Version 0.51-5

Date 2024-09-14

Maintainer Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Depends R (>= 3.5.0), cluster, MASS

Imports ade4, e1071, grDevices, graphics, stats, utils

Suggests mlbench, testthat

Description Distance measures (GDM1, GDM2,Sokal-Michener, Bray-Curtis, for symbolic interval-valued data), cluster quality indices (Calinski-Harabasz, Baker-Hubert, Hubert-Levine, Silhouette, Krzanowski-Lai, Hartigan, Gap,Davies-Bouldin),data normalization formulas (metric data, interval-valued symbolic data), data generation (typical and non-typical data), HINoV method,replication analysis, linear ordering methods, spectral clustering, agreement indices between two partitions, plot functions (for categorical and symbolic interval-valued data).
(MILLIGAN, G.W., COOPER, M.C. (1985) <doi:10.1007/BF02294245>,
HUBERT, L., ARABIE, P. (1985) <doi:10.1007%2FBF01908075>,
RAND, W.M. (1971) <doi:10.1080/01621459.1971.10482356>,
JAJUGA, K., WALESIAK, M. (2000) <doi:10.1007/978-3-642-57280-7_11>,
MILLIGAN, G.W., COOPER, M.C. (1988) <doi:10.1007/BF01897163>,
JAJUGA, K., WALESIAK, M., BAK, A. (2003) <doi:10.1007/978-3-642-55721-7_12>,
DAVIES, D.L., BOULDIN, D.W. (1979) <doi:10.1109/TPAMI.1979.4766909>,
CALINSKI, T., HARABASZ, J. (1974) <doi:10.1080/03610927408827101>,
HUBERT, L. (1974) <doi:10.1080/01621459.1974.10480191>,
TIBSHIRANI, R., WALTHER, G., HASTIE, T. (2001) <doi:10.1111/1467-9868.00293>,
BRECKENRIDGE, J.N. (2000) <doi:10.1207/S15327906MBR3502_5>,
WALESIAK, M., DUDEK, A. (2008) <doi:10.1007/978-3-540-78246-9_11>).

License GPL (>= 2)

Encoding UTF-8

NeedsCompilation yes

Author Marek Walesiak [aut] (<`https://orcid.org/0000-0003-0922-2323`>),
Andrzej Dudek [aut, cre] (<`https://orcid.org/0000-0002-4943-8703`>)

Repository CRAN

Date/Publication 2024-09-14 14:50:01 UTC

Contents

`cluster.Description` *Descriptive statistics calculated separately for each cluster and variable*

Description

Descriptive statistics calculated separately for each cluster and variable: arithmetic mean and standard deviation, median and median absolute deviation, mode

Usage

```
cluster.Description(x, cl, sdType=”sample”,precission=4,modeAggregationChar=”;”)
```

Arguments

`x`	matrix or dataset
`cl`	a vector of integers indicating the cluster to which each object is allocated
`sdType`	type of standard deviation: for "sample" (n-1) or for "population" (n)
`precission`	Number of digits on the right side of decimal mark sign
`modeAggregationChar`	Character used for aggregation of mode values (if more than one value of mode appear in variable)

Value

Three-dimensional array:

First dimension contains cluster number

Second dimension contains original coordinate (variable) number from matrix or data set

Third dimension contains number from 1 to 5:

1 - arithmetic mean

2 - standard deviation

3 - median

4 - median absolute deviation (mad)

5 - mode (value of the variable which has the largest observed frequency. This formula is applicable for nominal and ordinal data only).

For example:

desc<-cluster.Description(x,cl)

desc[2,4,2] - standard deviation of fourth coordinate of second cluster

desc[3,1,5] - mode of first coordinate (variable) of third cluster

desc[1,,] - all statistics for all dimensions (variables) of first cluster

desc[,,3] - medians of all dimensions (variables) for each cluster

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

See Also

`cluster.Sim`, `mean`, `sd`, `median`, `mad`

Examples

```
library(clusterSim)
data(data_ratio)
cl <- pam(data_ratio,5)
desc <- cluster.Description(data_ratio,cl$cluster)
print(desc)
```

`cluster.Gen` *Random cluster generation with known structure of clusters*

Description

Random cluster generation with known structure of clusters (optionally with noisy variables and outliers)

Usage

```
cluster.Gen(numObjects=50, means=NULL, cov=NULL, fixedCov=TRUE,
                     model=1, dataType="m",numCategories=NULL,
                     numNoisyVar=0, numOutliers=0, rangeOutliers=
                     c(1,10), inputType="csv2", inputHeader=TRUE,
                     inputRowNames=TRUE, outputCsv="", outputCsv2="",
                     outputColNames=TRUE, outputRowNames=TRUE)
```

Arguments

`numObjects`	number of objects in each cluster - positive integer value or vector with the same size as *nrow(means)*, e.g. `numObjects=c(50,20)`
`means`	matrix of cluster means (e.g. `means=matrix(c(0,8,0,8),2,2)`). If `means = NULL` matrix should be read from *means_<modelNumber>.csv file*
`cov`	covariance matrix (the same for each cluster, e.g. `cov=matrix(c(1, 0, 0, 1), 2, 2)`). If `cov=NULL` matrix should be read from *cov_<modelNumber>.csv file*. Note: you cannot use this argument for generation of clusters with different covariance matrices. Those kind of generation should be done by setting `fixedCov` to `FALSE` and using appropriate model

`model`	model number, `model=1` - no cluster structure. Observations are simulated from uniform distribution over the unit hypercube in number of dimensions (variables) given in `numNoisyVar` argument; `model=2` - means and covariances are taken from arguments `means` and `cov` (see Example 1); `model=3,4,...,20` - see file $R_HOME\library\clusterSim\pdf\clusterGen_details.pdf; `model=21,22,...` - if `fixedCov=TRUE` `means` should be read from *means_<modelNumber>.csv* and covariance matrix for all clusters should be read from *cov_<modelNumber>.csv* and if `fixedCov=FALSE` `means` should be read from *means_<modelNumber>.csv* and covariance matrices should be read separately for each cluster from *cov_<modelNumber>_<clusterNumber>.csv*
`fixedCov`	if `fixedCov=TRUE` covariance matrix for all clusters is the same and if `fixedCov=FALSE` each cluster is generated from different covariance matrix - see `model`
`dataType`	"m" - metric (ratio, interval), "o" - ordinal, "s" - symbolic interval
`numCategories`	number of categories (for ordinal data only). Positive integer value or vector with the same size as *ncol(means)* plus number of noisy variables.
`numNoisyVar`	number of noisy variables. For `model=1` it means number of variables
`numOutliers`	number of outliers (for metric and symbolic interval data only). If a positive integer - number of outliers, if value from <0,1> - percentage of outliers in whole data set
`rangeOutliers`	range for outliers (for metric and symbolic interval data only). The default range is [1, 10].The outliers are generated independently for each variable for the whole data set from uniform distribution. The generated values are randomly added to maximum of *j*-th variable or subtracted from minimum of *j*-th variable
`inputType`	"csv" - a dot as decimal point or "csv2" - a comma as decimal point in *means_<modelNumber>.csv* and *cov_<modelNumber>.csv* files
`inputHeader`	`inputHeader=TRUE` indicates that input files (*means_<modelNumber>.csv*; *cov_<modelNumber...>.csv*) contain header row
`inputRowNames`	`inputRowNames=TRUE` indicates that input files (*means_<modelNumber>.csv*; *cov_<modelNumber...>.csv*) contain first column with row names or with number of objects (positive integer values)
`outputCsv`	optional, name of csv file with generated data (first column contains id, second - number of cluster and others - data)
`outputCsv2`	optional, name of csv (a comma as decimal point and a semicolon as field separator) file with generated data (first column contains id, second - number of cluster and others - data)
`outputColNames`	`outputColNames=TRUE` indicates that output file (given by `outputCsv` and `outputCsv2` parameters) contains first row with column names
`outputRowNames`	`outputRowNames=TRUE` indicates that output file (given by `outputCsv` and `outputCsv2` parameters) contains a vector of row names

Details

See file $R_HOME\library\clusterSim\pdf\clusterGen_details.pdf for further details

Value

`clusters`	cluster number for each object, for `model=1` each object belongs to its own cluster thus this variable contains objects numbers
`data`	generated data: for metric and ordinal data - matrix with objects in rows and variables in columns; for symbolic interval data three-dimensional structure: first dimension represents object number, second - variable number and third dimension contains lower- and upper-bounds of intervals

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Billard, L., Diday, E. (2006), *Symbolic data analysis. Conceptual statistics and data mining*, Wiley, Chichester. ISBN: 978-0-470-09016-9.

Qiu, W., Joe, H. (2006), *Generation of random clusters with specified degree of separation*, "Journal of Classification", vol. 23, 315-334. Available at: doi:10.1007/s003570060018y.

Steinley, D., Henson, R. (2005), *OCLUS: an analytic method for generating clusters with known overlap*, "Journal of Classification", vol. 22, 221-250. Available at: doi:10.1007/s0035700500156.

Walesiak, M., Dudek, A. (2008), *Identification of noisy variables for nonmetric and symbolic data in cluster analysis*, In: C. Preisach, H. Burkhardt, L. Schmidt-Thieme, R. Decker (Eds.), Data analysis, machine learning and applications, Springer-Verlag, Berlin, Heidelberg, 85-92. Available at: doi:10.1007/9783540782469_11.

Walesiak, M. (2016), *Uogólniona miara odległości GDM w statystycznej analizie wielowymiarowej z wykorzystaniem programu R. Wydanie 2 poprawione i rozszerzone [The Generalized Distance Measure GDM in multivariate statistical analysis with R]*, Wydawnictwo Uniwersytetu Ekonomicznego, Wroclaw.

Examples

```
# Example 1
library(clusterSim)
means <- matrix(c(0,7,0,7),2,2)
cov <- matrix(c(1,0,0,1),2,2)
grnd <- cluster.Gen(numObjects=60,means=means,cov=cov,model=2,
numOutliers=8)
colornames <- c("red","blue","green")
grnd$clusters[grnd$clusters==0]<-length(colornames)
plot(grnd$data,col=colornames[grnd$clusters],ask=TRUE)

# Example 2
```

```
library(clusterSim)
grnd <- cluster.Gen(50,model=4,dataType="m",numNoisyVar=2)
data <- as.matrix(grnd$data)
colornames <- c("red","blue","green")
plot(grnd$data,col=colornames[grnd$clusters],ask=TRUE)

# Example 3
library(clusterSim)
grnd<-cluster.Gen(50,model=4,dataType="o",numCategories=7, numNoisyVar=2)
plotCategorial(grnd$data,,grnd$clusters,ask=TRUE)

# Example 4 (1 nonnoisy variable and 2 noisy variables, 3 clusters)
library(clusterSim)
grnd <- cluster.Gen(c(40,60,20), model=2, means=c(2,14,25),
cov=c(1.5,1.5,1.5),numNoisyVar=2)
colornames <- c("red","blue","green")
plot(grnd$data,col=colornames[grnd$clusters],ask=TRUE)

# Example 5
library(clusterSim)
grnd <- cluster.Gen(c(20,35,20,25),model=14,dataType="m",numNoisyVar=1,
fixedCov=FALSE, numOutliers=0.1)
# or
#grnd <- cluster.Gen(c(20,35,20,25),model=14,dataType="m",numNoisyVar=1,
#fixedCov=FALSE, numOutliers=0.1, outputCsv2="data14.csv")
data <- as.matrix(grnd$data)
colornames <- c("red","blue","green","brown","black")
grnd$clusters[grnd$clusters==0]<-length(colornames)
plot(grnd$data,col=colornames[grnd$clusters],ask=TRUE)

# Example 6 (this example needs files means_24.csv)
# and cov_24.csv to be placed in working directory
# library(clusterSim)
# grnd<-cluster.Gen(c(50,80,20),model=24,dataType="m",numNoisyVar=1,
# numOutliers=10, rangeOutliers=c(1,5))
# print(grnd)
# data <- as.data.frame(grnd$data)
# colornames<-c("red","blue","green","brown")
# grnd$clusters[grnd$clusters==0]<-length(colornames)
# plot(data,col=colornames[grnd$clusters],ask=TRUE)

# Example 7 (this example needs files means_25.csv and cov_25_1.csv)
# cov_25_2.csv, cov_25_3.csv, cov_25_4.csv, cov_25_5.csv
# to be placed in working directory
# library(clusterSim)
# grnd<-cluster.Gen(c(40,30,20,35,45),model=25,numNoisyVar=3,fixedCov=F)
# data <- as.data.frame(grnd$data)
# colornames<-c("red","blue","green","magenta","brown")
# plot(data,col=colornames[grnd$clusters],ask=TRUE)
```

`cluster.Sim` *Determination of optimal clustering procedure for a data set*

Description

Determination of optimal clustering procedure for a data set by varying all combinations of normalization formulas, distance measures, and clustering methods

Usage

```
cluster.Sim (x,p,minClusterNo,maxClusterNo,icq=”S”,
outputCsv=””,outputCsv2=””,normalizations=NULL,
distances=NULL,methods=NULL)
```

Arguments

`x`	matrix or dataset
`p`	path of simulation: 1 - ratio data, 2 - interval or mixed (ratio & interval) data, 3 - ordinal data, 4 - nominal data, 5 - binary data, 6 - ratio data without normalization, 7 - interval or mixed (ratio & interval) data without normalization, 8 - ratio data with k-means, 9 - interval or mixed (ratio & interval) data with k-means
`minClusterNo`	minimal number of clusters, between 2 and no. of objects - 1 (for G3 or C: no. of objects - 2)
`maxClusterNo`	maximal number of clusters, between 2 and no. of objects - 1 (for G3 or C: no. of objects - 2; for KL: no. of objects - 3), greater or equal minClusterNo
`icq`	Internal cluster quality index, "S" - Silhouette,"G1" - Calinski & Harabasz index, "G2" - Baker & Hubert index ,"G3" - G3 index,"C" - C index, "KL" - Krzanowski & Lai index
`outputCsv`	optional, name of csv file with results
`outputCsv2`	optional, name of csv (comma as decimal point sign) file with results
`normalizations`	optional, vector of normalization formulas that should be used in procedure
`distances`	optional, vector of distance measures that should be used in procedure
`methods`	optional, vector of classification methods that should be used in procedure

Details

Parameter `normalizations` for each path may be the subset of the following values

path 1: "n6" to "n11" (if measurement scale of variables is ratio and transformed measurement scale of variables is ratio) or "n1" to "n5" (if measurement scale of variables is ratio and transformed measurement scale of variables is interval)

path 2: "n1" to "n5"

path 3 to 7 : "n0"

path 8: "n1" to "n11"

path 9: "n1" to "n5"

Parameter `distances` for each path may be the subset of the following values

path 1: "d1" to "d7" (if measurement scale of variables is ratio and transformed measurement scale of variables is ratio) or "d1" to "d5" (if measurement scale of variables is ratio and transformed measurement scale of variables is interval) path 2: "d1" to "d5"

path 3: "d8"

path 4: "d9"

path 5: "b1" to "b10"

path 6: "d1" to "d7"

path 7: "d1" to "d5"

path 8 and 9: N.A.

Parameter `methods` for each path may be the subset of the following values

path 1 to 7 : "m1" to "m8"

path 8: "m9"

path 9: "m9"

See file `../doc/clusterSim_details.pdf` for further details

Value

`result`	optimal value of icq for all classifications
`normalization`	normalization used to obtain optimal value of icq
`distance`	distance measure used to obtain optimal value of icq
`method`	clustering method used to obtain optimal value of icq
`classes`	number of clusters for optimal value of icq
`time`	time of all calculations for path

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Everitt, B.S., Landau, E., Leese, M. (2001), *Cluster analysis*, Arnold, London. ISBN 9780340761199.

Gatnar, E., Walesiak, M. (Eds.) (2004), *Metody statystycznej analizy wielowymiarowej w badaniach marketingowych [Multivariate statistical analysis methods in marketing research]*, Wydawnictwo AE, Wroclaw, p. 338.

Gordon, A.D. (1999), *Classification*, Chapman & Hall/CRC, London. ISBN 9781584880134.

Milligan, G.W., Cooper, M.C. (1985), *An examination of procedures of determining the number of cluster in a data set*, "Psychometrika", vol. 50, no. 2, 159-179. Available at: doi:10.1007/BF02294245.

Milligan, G.W., Cooper, M.C. (1988), *A study of standardization of variables in cluster analysis*, "Journal of Classification", vol. 5, 181-204. Available at: doi:10.1007/BF01897163.

Walesiak, M., Dudek, A. (2006), *Symulacyjna optymalizacja wyboru procedury klasyfikacyjnej dla danego typu danych - oprogramowanie komputerowe i wyniki badan*, Prace Naukowe AE we Wroclawiu, 1126, 120-129.

Walesiak, M., Dudek, A. (2007), *Symulacyjna optymalizacja wyboru procedury klasyfikacyjnej dla danego typu danych - charakterystyka problemu*, Zeszyty Naukowe Uniwersytetu Szczecinskiego nr 450, 635-646.

See Also

`data.Normalization`, `dist.GDM`, `dist.BC`, `dist.SM`, `index.G1`, `index.G2`, `index.G3`, `index.C`, `index.S`, `index.KL`, `hclust`, `dist`,

Examples

```
#library(clusterSim)
#data(data_ratio)
#cluster.Sim(data_ratio, 1, 2, 3, "G1", outputCsv="results1")
#data(data_interval)
#cluster.Sim(data_interval, 2, 2, 4, "G1", outputCsv="results2")
#data(data_ordinal)
#cluster.Sim(data_ordinal, 3, 2, 4,"G2", outputCsv2="results3")
#data(data_nominal)
#cluster.Sim(data_nominal, p=4, 2, 4, icq="G3", outputCsv="results4", methods=c("m2","m3","m5"))
#data(data_binary)
#cluster.Sim(as.matrix(data_binary), p=5, 2, 4, icq="S",
#outputCsv="results5", distances=c("b1","b3","b6"))
#data(data_ratio)
#cluster.Sim(data_ratio, 1, 2, 4,"G1", outputCsv="results6",normalizations=c("n1","n3"),
#distances=c("d2","d5"),methods=c("m5","m3","m1"))
```

`comparing.Partitions` *Calculate agreement indices between two partitions*

Description

Calculate agreement indices between two partitions

Usage

```
comparing.Partitions(cl1,cl2,type="nowak")
```

Arguments

`cl1`	A vector of integers (or letters) indicating the cluster to which each object is allocated for first clustering
`cl2`	A vector of integers (or letters) indicating the cluster to which each object is allocated for second clustering
`type`	"rand" - for Rand index, "crand" - for adjusted Rand index or "nowak" for Nowak index

Details

See file $R_HOME\library\clusterSim\pdf\comparingPartitions_details.pdf for further details.

Rand and adjusted Rand indices uses `classAgreement` function from `e1071` library.

Value

Returns value of index.

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Hubert, L., Arabie, P. (1985), *Comparing partitions*, "Journal of Classification", no. 1, 193-218. Available at: doi:10.1007/BF01908075.

Nowak, E. (1985), *Wskaznik podobienstwa wynikow podzialow*, "Przeglad Statystyczny" ["Statistical Review"], no. 1, 41-48.

Rand, W.M. (1971), *Objective criteria for the evaluation of clustering methods*, "Journal of the American Statistical Association", no. 336, 846-850. Available at: doi:10.1080/01621459.1971.10482356.

See Also

`replication.Mod`

Examples

```
# Example 1
library(clusterSim)
dataSet<-cluster.Gen(model=5)
cl1<-dataSet$clusters
cl2<-kmeans(dataSet$data,2)$cluster
print(comparing.Partitions(cl1,cl2,type="rand"))

# Example 2
library(clusterSim)
data(data_patternGDM1)
z<-data.Normalization(data_patternGDM1,type="n1")
d<-dist.GDM(z,method="GDM1")
cl1<-pam(d,3,diss=TRUE)$clustering
cl2<-pam(d,4,diss=TRUE)$clustering
print(comparing.Partitions(cl1,cl2,type="crand"))

# Example 3
library(clusterSim)
data(data_patternGDM1)
z<-data.Normalization(data_patternGDM1,type="n9")
d<-dist.GDM(z,method="GDM1")
cl1<-pam(d,3,diss=TRUE)$clustering
```

```
hc<-hclust(d, method="complete")
cl2<-cutree(hc,k=3)
print(comparing.Partitions(cl1,cl2,type="nowak"))
```

`data.Normalization` *Types of variable (column) and object (row) normalization formulas*

Description

Types of variable (column) and object (row) normalization formulas

Usage

```
data.Normalization (x,type="n0",normalization="column",...)
```

Arguments

`x`	vector, matrix or dataset
`type`	type of normalization:

n0 - without normalization

n1 - standardization ((x-mean)/sd)

n2 - positional standardization ((x-median)/mad)

n3 - unitization ((x-mean)/range)

n3a - positional unitization ((x-median)/range)

n4 - unitization with zero minimum ((x-min)/range)

n5 - normalization in range <-1,1> ((x-mean)/max(abs(x-mean)))

n5a - positional normalization in range <-1,1> ((x-median)/max(abs(x-median)))

n6 - quotient transformation (x/sd)

n6a - positional quotient transformation (x/mad)

n7 - quotient transformation (x/range)

n8 - quotient transformation (x/max)

n9 - quotient transformation (x/mean)

n9a - positional quotient transformation (x/median)

n10 - quotient transformation (x/sum)

n11 - quotient transformation (x/sqrt(SSQ))

n12 - normalization ((x-mean)/sqrt(sum((x-mean)^2)))

n12a - positional normalization ((x-median)/sqrt(sum((x-median)^2)))

n13 - normalization with zero being the central point ((x-midrange)/(range/2))

`normalization`	"column" - normalization by variable, "row" - normalization by object
`...`	arguments passed to `sum`, `mean`, `min` `sd`, `mad` and other aggregation functions. In particular: `na.rm` - a logical value indicating whether NA values should be stripped before the computation

Details

See file `../doc/dataNormalization_details.pdf` for further details

Thanks Wolfgang Lederer (<wolfgang.lederer@gmail.com>) for reporting n4/vector error

Value

Normalized data The numeric shifts and scalings used (if any) are returned as attributes "normalized:shift" and "normalized:scale"

Author(s)

Marek Walesiak `<marek.walesiak@ue.wroc.pl>`, Andrzej Dudek `<andrzej.dudek@ue.wroc.pl>`

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Anderberg, M.R. (1973), *Cluster analysis for applications*, Academic Press, New York, San Francisco, London. ISBN 9780120576500.

Gatnar, E., Walesiak, M. (Eds.) (2004), *Metody statystycznej analizy wielowymiarowej w badaniach marketingowych [Multivariate statistical analysis methods in marketing research]*, Wydawnictwo AE, Wroclaw, 35-38.

Jajuga, K., Walesiak, M. (2000), *Standardisation of data set under different measurement scales*, In: R. Decker, W. Gaul (Eds.), Classification and information processing at the turn of the millennium, Springer-Verlag, Berlin, Heidelberg, 105-112. Available at: doi:10.1007/9783642572807_11.

Milligan, G.W., Cooper, M.C. (1988), *A study of standardization of variables in cluster analysis*, "Journal of Classification", vol. 5, 181-204. Available at: doi:10.1007/BF01897163.

Mlodak, A. (2006), *Analiza taksonomiczna w statystyce regionalnej*, Difin, Warszawa. ISBN 83-7251-605-7.

Walesiak, M. (2014), *Przeglad formul normalizacji wartosci zmiennych oraz ich wlasnosci w statystycznej analizie wielowymiarowej [Data normalization in multivariate data analysis. An overview and properties]*, "Przeglad Statystyczny" ("Statistical Review"), vol. 61, no. 4, 363-372. Available at: doi:10.5604/01.3001.0016.1740.

See Also

`cluster.Sim`

Examples

```
library(clusterSim)
data(data_ratio)
z1 <- data.Normalization(data_ratio,type="n1",normalization="column",na.rm=FALSE)
z2 <- data.Normalization(data_ratio,type="n10",normalization="row",na.rm=FALSE)
```

`data_binary` *Binary data*

Description

Binary variables for eight people

Format

data.frame: 8 objects, 10 variables

Source

Kaufman, L., Rousseeuw, P.J. (1990), *Finding groups in data: an introduction to cluster analysis*, Wiley, New York, p. 24.

Examples

```
#library(clusterSim)
#data(data_binary)
#cluster.Sim(as.matrix(data_binary), p=5, 2, 6, icq="S",
#outputCsv="results5", distances=c("b1","b3","b6"))
```

`data_interval` *Interval data*

Description

Artificially generated interval data

Format

data.frame: 75 objects, 5 variables, 5-class structure

Source

Artificially generated data

Examples

```
#library(clusterSim)
#data(data_interval)
#cluster.Sim(data_interval, 2, 2, 3, "G1", outputHtml="results2")
```

`data_mixed` *Mixed data*

Description

Artificial mixed data

Format

data.frame: 25 objects, 4 variables (1, 3 - interval variables, 2 - ordinal variable, 4, nominal variable

Source

Artificial data

Examples

```
library(clusterSim)
data(data_mixed)
r3 <- HINoV.Mod(data_mixed, type=c("m","n","m","n"), s=2, 3, distance="d1",
     method="complete", Index="cRAND")
print(r3$stopri)
plot(r3$stopri[,2], xlab="Variable number", ylab="topri", xaxt="n")
axis(1,at=c(1:max(r3$stopri[,1])),labels=r3$stopri[,1])
```

`data_nominal` *Nominal data*

Description

Artificial nominal data

Format

data.frame: 26 objects, 12 variables

Source

Artificial data

Examples

```
#library(clusterSim)
#data(data_nominal)
#cluster.Sim(data_nominal, p=4, 2, 5, icq="G3",
#outputHtml="results4", methods=c("m2","m3","m5"))
```

`data_ordinal` *Ordinal data*

Description

Artificial ordinal data

Format

data.frame: 26 objects, 12 variables

Source

Artificial data

Examples

```
#library(clusterSim)
#data(data_ordinal)
#cluster.Sim(data_ordinal, 3, 3, 12,"S",
#outputCsv2="results3")
```

`data_patternGDM1` *Metric data with 17 objects and 10 variables (8 stimulant variables, 2 destimulant variables)*

Description

Metric data with 17 objects and 10 variables (8 stimulant variables, 2 destimulant variables)

Data on the Polish voivodships, owing to the conditions of the population living in cities in 2007. The analysis includes the following variables:

x1 - dwellings in per cent fitted with water-line system,

x2 - dwellings in per cent fitted with lavatory,

x3 - dwellings in per cent fitted with bathroom,

x4 - dwellings in per cent fitted with gas-line system,

x5 - dwellings in per cent fitted with central heating,

x6 - average number of rooms per dwelling,

x7 - average number of persons per dwelling,

x8 - average number of persons per room,

x9 - usable floor space in square meter per dwelling,

x10 - usable floor space in square meter per person.

Types of performance variables:

x1 - x6, x9, x10 - stimulants,

x7, x8 - destimulants.

Format

data.frame: 17 objects, 10 variables

Source

Voivodships Statistical Yearbook, Poland 2008.

Examples

```
# Example 1
library(clusterSim)
data(data_patternGDM1)
res<-pattern.GDM1(data_patternGDM1,
performanceVariable=c("s","s","s","s","s","s","d","d","s","s"),
scaleType="r",nomOptValues=NULL,weightsType<-"equal",weights=NULL,
normalization<-"n4",patternType<-"lower",patternCoordinates<-"manual",
patternManual<-c(0,0,0,0,0,"min","max","max","min","min"),
nominalTransfMethod <-NULL)
print(res)
gdm_p<-res$distances
plot(cbind(gdm_p,gdm_p),xlim=c(max(gdm_p),min(gdm_p)),
ylim=c(min(gdm_p),max(gdm_p)),xaxt="n",
xlab="Order of objects from the best to the worst",
ylab="GDM distances from pattern object",lwd=1.6)
axis(1, at=gdm_p,labels=names(gdm_p), cex.axis=0.5)

# Example 2
library(clusterSim)
data(data_patternGDM1)
res<-pattern.GDM1(data_patternGDM1,
performanceVariable=c("s","s","s","s","s","s","d","d","s","s"),
scaleType="r",nomOptValues=NULL,weightsType<-"equal",weights=NULL,
normalization<-"n2",patternType<-"upper",
patternCoordinates<-"dataBounds",patternManual<-NULL,
nominalTransfMethod <-NULL)
print(res)
gdm_p<-res$distances
plot(cbind(gdm_p,gdm_p),xlim=c(min(gdm_p),max(gdm_p)),
ylim=c(min(gdm_p),max(gdm_p)),xaxt="n",
xlab="Order of objects from the best to the worst",
ylab="GDM distances from pattern object", lwd=1.6)
axis(1, at=gdm_p,labels=names(gdm_p), cex.axis=0.5)
```

`data_patternGDM2` *Ordinal data with 27 objects and 6 variables (3 stimulant variables, 2 destimulant variables and 1 nominant variable)*

Description

Ordinal data with 27 objects and 6 variables (3 stimulant variables, 2 destimulant variables and 1 nominant variable)

Residential housing properties were described by the following variables:

x1 - Location of environmental land, which is linked to a dwelling (1 - poor, 2 - inadequate, 3 - satisfactory, 4 - good, 5 - very good),

x2 - Standard utility of a dwelling (1 - bad, 2 - low, 3 - average, 4 - high),

x3 - Living conditions occurring on the land, which is linked to a dwelling (1 - bad, 2 - average, 3 - good),

x4 - Location of land, which is related to dwelling in the area of the city (1 - central, 2 - downtown, 3 - intermediate, 4 - peripheral),

x5 - Type of condominium (1 - low, 2 - large),

x6 - Area of land, which is related to dwelling (1 - below the contour of the building, 2 - outline of the building, 3 - the outline of the building with the environment acceptable, such as parking, playground, 4 - the outline of the building with the environment too much).

Types of performance variables:

x1, x2, x3 - stimulants,

x4, x5 - destimulants,

x6 - nominant (the nominal category: 3).

Format

data.frame: 27 objects, 6 variables

Source

data from real estate market

Examples

```
# Example 1
library(clusterSim)
data(data_patternGDM2)
res<-pattern.GDM2(data_patternGDM2,
performanceVariable=c("s","s","s","d","d","n"),
nomOptValues=c(NA,NA,NA,NA,NA,3), weightsType<-"equal", weights=NULL,
patternType="lower", patternCoordinates="manual",
patternManual=c("min","min",0,5,"max","max"),
nominalTransfMethod="symmetrical")
print(res)
gdm_p<-res$distances
plot(cbind(gdm_p,gdm_p),xlim=c(max(gdm_p),min(gdm_p)),
ylim=c(min(gdm_p),max(gdm_p)),
xaxt="n",xlab="Order of objects from the best to the worst",
ylab="GDM distances from pattern object",lwd=1.6)
axis(1, at=gdm_p,labels=names(gdm_p), cex.axis=0.5)
```

```
# Example 2
library(clusterSim)
data(data_patternGDM2)
res<-pattern.GDM2(data_patternGDM2,
performanceVariable=c("s","s","s","d","d","n"),
nomOptValues=c(NA,NA,NA,NA,NA,3), weightsType<-"equal", weights=NULL,
patternType="upper", patternCoordinates="dataBounds",
patternManual=NULL, nominalTransfMethod="database")
print(res)
gdm_p<-res$distances
plot(cbind(gdm_p,gdm_p), xlim=c(min(gdm_p),max(gdm_p)),
ylim=c(min(gdm_p),max(gdm_p)),
xaxt="n",xlab="Order of objects from the best to the worst",
ylab="GDM distances from pattern object", lwd=1.6)
axis(1, at=gdm_p,labels=names(gdm_p), cex.axis=0.5)
```

`data_ratio` *Ratio data*

Description

Artificially generated ratio data

Format

data.frame: 75 objects, 5 variables, 5-class structure

Source

Artificially generated data

Examples

```
#library(clusterSim)
#data(data_ratio)
#c <- pam(data_ratio,10)
#index.G1(data_ratio, c$clustering)
```

`data_symbolic` *Symbolic interval data*

Description

Artificially generated symbolic interval data

Format

3-dimensional array: 125 objects, 6 variables, third dimension represents begining and end of interval, 5-class structure

Source

Artificially generated data

Examples

```
library(clusterSim)
data(data_symbolic)
r<- HINoV.Symbolic(data_symbolic, u=5)
print(r$stopri)
plot(r$stopri[,2], xlab="Variable number", ylab="topri",
xaxt="n", type="b")
axis(1,at=c(1:max(r$stopri[,1])),labels=r$stopri[,1])
```

`data_symbolic_interval_polish_voivodships` *The evaluation of Polish voivodships tourism attractiveness level*

Description

The empirical study uses the statistical data referring to the attractiveness level of 18 objects (16 Polish NUTS-2 regions - voivodships, pattern and anti-pattern object).

Two-stage data collection was performed. Firstly, data on tourist attractiveness were collected for 380 counties using 9 classic metric variables (measured on a ratio scale):

x1 - beds in hotels per 1000 inhabitants of a county,

x2 - number of nights spent daily by resident tourists per 1000 inhabitants of a county,

x3 - number of nights spent daily by foreign tourists per 1000 inhabitants of a county,

x4 - dust pollution emission in tons per 10 km2 of a county area,

x5 - gas pollution emission in tons per 1 km2 of a county area,

x6 - number of criminal offences, crimes against life and health and property crimes per 1000 inhabitants of a county, x7 - forest cover of the county in

x8 - participants of mass events per 1000 inhabitants of a county,

x9 - number of tourist economy entities (sections: I, N79) registered in the system REGON per 1000 inhabitants of a county.

The three variables (x4, x5 and x6) are destimulants. Other variables are stimulants.

In the second step, the data were aggregated to the level of the voivodships (NUTS-2), giving the symbolic interval-valued data. The lower bound of the interval for each symbolic interval-valued variable in the voivodship was obtained by calculating the first quartile on the basis of data from counties. The upper bound of the interval was obtained by calculating the third quartile.

Format

Tree-dimansional array: 18 objects (16 Polish NUTS-2 regions - voivodships, pattern and anti-pattern object), 9 symbolic interval-valued variables with lower and upper values of interval in third dimension. The coordinates of an pattern object cover the most preferred preference variable values. The coordinates of an anti-pattern object cover the least preferred preference variable values.

Source

The statistical data were collected in 2016 and come from the Local Data Bank of the Central Statistical Office of Poland.

Examples

```
library(clusterSim)
data(data_symbolic_interval_polish_voivodships)
print(data_symbolic_interval_polish_voivodships)
```

`dist.BC` *Calculates Bray-Curtis distance measure for ratio data*

Description

Calculates Bray-Curtis distance measure for ratio data

Usage

```
dist.BC (x)
```

Arguments

`x`	matrix or dataset

Details

See file $R_HOME\library\clusterSim\pdf\distBC_details.pdf for further details

Value

object with calculated distance

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Cormack, R.M. (1971), *A review of classification (with discussion)*, "Journal of the Royal Statistical Society", ser. A, part 3, 321-367.

Gatnar, E., Walesiak, M. (Eds.) (2004), *Metody statystycznej analizy wielowymiarowej w badaniach marketingowych [Multivariate statistical analysis methods in marketing research]*, Wydawnictwo AE, Wroclaw, p. 41.

See Also

`dist.GDM`, `dist.SM`, `dist`

Examples

```
library(clusterSim)
sampleData <- cbind(c(2,3,5),c(4,5,6),c(5,3,4))
d <- dist.BC(sampleData)
```

`dist.GDM` *Calculates Generalized Distance Measure*

Description

Calculates Generalized Distance Measure for variables measured on metric scale (ratio & interval) or ordinal scale

Usage

```
dist.GDM(x, method="GDM1", weightsType="equal", weights=NULL)
GDM(x, method="GDM1", weightsType="equal", weights=NULL)
GDM1(x, weightsType="equal", weights=NULL)
GDM2(x, weightsType="equal", weights=NULL)
```

Arguments

`x`	matrix or data set
`method`	GDM1 or GDM2

"GDM1" - metric scale (ratio & interval)

"GDM2" - ordinal scale

`weightsType`	equal or different1 or different2

"equal" - equal weights

"different1" - vector of different weights should satisfy conditions: each weight takes value from interval [0; 1] and sum of weights equals one

"different2" - vector of different weights should satisfy conditions: each weight takes value from interval [0; m] and sum of weights equals m (m - the number of variables)

`weights`	vector of weights

Details

See file $R_HOME\library\clusterSim\pdf\distGDM_details.pdf for further details

Value

object with calculated distance

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Jajuga, K., Walesiak, M., Bak, A. (2003), *On the general distance measure*, In: M. Schwaiger, O. Opitz (Eds.), Exploratory data analysis in empirical research, Springer-Verlag, Berlin, Heidelberg, 104-109. Available at: doi:10.1007/9783642557217_12.

Walesiak, M. (1999), *Distance Measure for Ordinal Data*, "Argumenta Oeconomica", No. 2 (8), 167-173.

Walesiak, M. (2006), *Uogolniona miara odleglosci w statystycznej analizie wielowymiarowej [The Generalized Distance Measure in multivariate statistical analysis]*, Wydawnictwo AE, Wroclaw.

See Also

`dist.BC`, `dist.SM`, `dist`

Examples

```
#Example 1
library(clusterSim)
data(data_ratio)
d1 <- GDM(data_ratio, method="GDM1")
data(data_ordinal)
d2 <- GDM(data_ordinal, method="GDM2")
d3 <- GDM1(data_ratio)
d4 <- GDM2(data_ordinal)

#Example 2
library(clusterSim)
data(data_ratio)
d1w <- GDM(data_ratio, method="GDM1", weightsType="different1",
weights=c(0.4,0.1,0.3,0.15,0.05))
data(data_ordinal)
d2w <- GDM(data_ordinal, method="GDM2", weightsType="different2",
weights=c(1,3,0.5,1.5,1.8,0.2,0.4,0.6,0.2,0.4,0.9,1.5))
```

`dist.SM` *Calculates Sokal-Michener distance measure for nominal variables*

Description

Calculates Sokal-Michener distance measure for nominal variables

Usage

```
dist.SM(x)
```

Arguments

`x`	matrix or data set

Details

See file $R_HOME\library\clusterSim\pdf\distSM_details.pdf for further details

Value

object with calculated distance

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Gatnar, E., Walesiak, M. (Eds.) (2004), *Metody statystycznej analizy wielowymiarowej w badaniach marketingowych [Multivariate statistical analysis methods in marketing research]*, Wydawnictwo AE, Wroclaw, p. 43.

Kaufman, L., Rousseeuw, P.J. (1990), *Finding groups in data: an introduction to cluster analysis*, Wiley, New York, p. 28. ISBN: 978-0-471-73578-6.

See Also

`dist.GDM`, `dist.BC`, `dist`

Examples

```
library(clusterSim)
data(data_nominal)
d <- dist.SM(data_nominal)
```

`dist.Symbolic` *Calculates distance between interval-valued symbolic data*

Description

Calculates distance between interval-valued symbolic data for four distance types

Usage

```
dist.Symbolic(data,type="U_2",gamma=0.5,power=2)
```

Arguments

`data`	symbolic data
`type`	type of distance used for symbolic interval-valued data U_2 - Ichino and Yaguchi distance M - distance between points given by means of intervals (for interval-values variables), H - Hausdorff distance, S - sum of distances between all corresponding vertices of hyperrectangles given by symbolic objects with interval-valued variables
`gamma`	parameter for calculating Ichino and Yaguchi distance
`power`	parameter for calculating distance: Ichino and Yaguchi distance, Hausdorff distance

Author(s)

Marek Walesiak `<marek.walesiak@ue.wroc.pl>`, Andrzej Dudek `<andrzej.dudek@ue.wroc.pl>`

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Billard, L., Diday, E. (2006), *Symbolic data analysis. Conceptual statistics and data mining*, Wiley, Chichester. ISBN: 978-0-470-09016-9.

Ichino, M., Yaguchi, H. (1994), *Generalized Minkowski metrics for mixed feature type data analysis*, "IEEE Transactions on Systems Man and Cybernetics", Vol. 24, Issue 4, 698-708. Http://dx.doi.org/10.1109/21.286391.

See Also

`symbolicDA::dist.SDA`

Examples

```
library(clusterSim)
dataSymbolic<-cluster.Gen(numObjects=10,model=5,dataType="s")$data
print(dist.Symbolic(dataSymbolic))
```

`HINoV.Mod` *Modification of Carmone, Kara & Maxwell Heuristic Identification of Noisy Variables (HINoV) method*

Description

Modification of Heuristic Identification of Noisy Variables (HINoV) method

Usage

```
HINoV.Mod (x, type="metric", s = 2, u, distance=NULL,
method = "kmeans", Index ="cRAND")
```

Arguments

`x`	data matrix
`type`	"metric" (default) - all variables are metric (ratio, interval), "nonmetric" - all variables are nonmetric (ordinal, nominal) or vector containing for each variable value "m"(metric) or "n"(nonmetric) for mixed variables (metric and nonmetric), e.g. type=c("m", "n", "n", "m")
`s`	for metric data only: 1 - ratio data, 2 - interval or mixed (ratio & interval) data
`u`	number of clusters (for metric data only)
`distance`	NULL for kmeans method (based on data matrix) and nonmetric data for ratio data: "d1" - Manhattan, "d2" - Euclidean, "d3" - Chebychev (max), "d4" - squared Euclidean, "d5" - GDM1, "d6" - Canberra, "d7" - Bray-Curtis for interval or mixed (ratio & interval) data: "d1", "d2", "d3", "d4", "d5"
`method`	NULL for nonmetric data clustering method: "kmeans" (default) , "single", "ward.D", "ward.D2", "complete", "average", "mcquitty", "median", "centroid", "pam"
`Index`	"cRAND" - corrected Rand index (default); "RAND" - Rand index

Details

See file `../doc/HINoVMod_details.pdf` for further details

Value

`parim`	*m* x *m* symmetric matrix (*m* - number of variables). Matrix contains pairwise corrected Rand (Rand) indices for partitions formed by the *j*-th variable with partitions formed by the *l*-th variable
`topri`	sum of rows of `parim`
`stopri`	ranked values of `topri` in decreasing order

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Carmone, F.J., Kara, A., Maxwell, S. (1999), *HINoV: a new method to improve market segment definition by identifying noisy variables*, "Journal of Marketing Research", November, vol. 36, 501-509.

Hubert, L.J., Arabie, P. (1985), *Comparing partitions*, "Journal of Classification", no. 1, 193-218. Available at: doi:10.1007/BF01908075.

Rand, W.M. (1971), *Objective criteria for the evaluation of clustering methods*, "Journal of the American Statistical Association", no. 336, 846-850. Available at: doi:10.1080/01621459.1971.10482356.

Walesiak, M. (2005), *Variable selection for cluster analysis - approaches, problems, methods*, Plenary Session of the Committee on Statistics and Econometrics of the Polish Academy of Sciences, 15 March, Wroclaw.

Walesiak, M., Dudek, A. (2008), *Identification of noisy variables for nonmetric and symbolic data in cluster analysis*, In: C. Preisach, H. Burkhardt, L. Schmidt-Thieme, R. Decker (Eds.), Data analysis, machine learning and applications, Springer-Verlag, Berlin, Heidelberg, 85-92. Available at: doi:10.1007/9783540782469_11

See Also

`hclust`, `kmeans`, `dist`, `dist.GDM`, `dist.BC`, `dist.SM`, `cluster.Sim`

Examples

```
# for metric data
library(clusterSim)
data(data_ratio)
r1<- HINoV.Mod(data_ratio, type="metric", s=1, 4, method="kmeans",
     Index="cRAND")
print(r1$stopri)
plot(r1$stopri[,2],xlab="Variable number", ylab="topri",
xaxt="n", type="b")
```

```
axis(1,at=c(1:max(r1$stopri[,1])),labels=r1$stopri[,1])

# for nonmetric data
library(clusterSim)
data(data_nominal)
r2<- HINoV.Mod (data_nominal, type="nonmetric", Index = "cRAND")
print(r2$stopri)
plot(r2$stopri[,2], xlab="Variable number", ylab="topri",
xaxt="n", type="b")
axis(1,at=c(1:max(r2$stopri[,1])),labels=r2$stopri[,1])

# for mixed data
library(clusterSim)
data(data_mixed)
r3<- HINoV.Mod(data_mixed, type=c("m","n","m","n"), s=2, 3, distance="d1",
     method="complete", Index="cRAND")
print(r3$stopri)
plot(r3$stopri[,2], xlab="Variable number", ylab="topri",
xaxt="n", type="b")
axis(1,at=c(1:max(r3$stopri[,1])),labels=r3$stopri[,1])
```

`HINoV.Symbolic` *Modification of Carmone, Kara & Maxwell Heuristic Identification of Noisy Variables (HINoV) method for symbolic interval data*

Description

Modification of Heuristic Identification of Noisy Variables (HINoV) method for symbolic interval data

Usage

```
HINoV.Symbolic(x, u=NULL, distance="H", method = "pam",
Index = "cRAND")
```

Arguments

`x`	symbolic interval data: a 3-dimensional table, first dimension represents object number, second dimension - variable number, and third dimension contains lower- and upper-bounds of intervals
`u`	number of clusters
`distance`	"M" - minimal distance between all vertices of hyper-cubes defined by symbolic interval variables; "H" - Hausdorff distance; "S" - sum of squares of distance between all vertices of hyper-cubes defined by symbolic interval variables
`method`	clustering method: "single", "ward.D", "ward.D2", "complete", "average", "mcquitty", "median", "centroid", "pam" (default)
`Index`	"cRAND" - corrected Rand index (default); "RAND" - Rand index

Details

See file `../doc/HINoVSymbolic_details.pdf` for further details

Value

`parim`	m x m symmetric matrix (m - number of variables). Matrix contains pairwise corrected Rand (Rand) indices for partitions formed by the j-th variable with partitions formed by the l-th variable
`topri`	sum of rows of `parim`
`stopri`	ranked values of `topri` in decreasing order

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Carmone, F.J., Kara, A., Maxwell, S. (1999), *HINoV: a new method to improve market segment definition by identifying noisy variables*, "Journal of Marketing Research", November, vol. 36, 501-509.

Hubert, L.J., Arabie, P. (1985), *Comparing partitions*, "Journal of Classification", no. 1, 193-218. Available at: doi:10.1007/BF01908075.

Rand, W.M. (1971), *Objective criteria for the evaluation of clustering methods*, "Journal of the American Statistical Association", no. 336, 846-850. Available at: doi:10.1080/01621459.1971.10482356.

Walesiak, M., Dudek, A. (2008), *Identification of noisy variables for nonmetric and symbolic data in cluster analysis*, In: C. Preisach, H. Burkhardt, L. Schmidt-Thieme, R. Decker (Eds.), Data analysis, machine learning and applications, Springer-Verlag, Berlin, Heidelberg, 85-92. Available at: doi:10.1007/9783540782469_11.

See Also

`hclust`, `kmeans`, `cluster.Sim`

Examples

```
library(clusterSim)
data(data_symbolic)
r<- HINoV.Symbolic(data_symbolic, u=5)
print(r$stopri)
plot(r$stopri[,2], xlab="Variable number", ylab="topri",
xaxt="n", type="b")
axis(1,at=c(1:max(r$stopri[,1])),labels=r$stopri[,1])

#symbolic data from .csv file
#library(clusterSim)
#dsym<-as.matrix(read.csv2(file="csv/symbolic.csv"))
#dim(dsym)<-c(dim(dsym)[1],dim(dsym)[2]%/%2,2)
```

```
#r<- HINoV.Symbolic(dsym, u=5)
#print(r$stopri)
#plot(r$stopri[,2], xlab="Variable number", ylab="topri",
#xaxt="n", type="b")
#axis(1,at=c(1:max(r$stopri[,1])),labels=r$stopri[,1])
```

`index.C` *Calculates Hubert & Levin C index - internal cluster quality index*

Description

Calculates Hubert & Levin C index - internal cluster quality index

Usage

```
index.C(d,cl)
```

Arguments

`d`	'dist' object
`cl`	A vector of integers indicating the cluster to which each object is allocated

Details

See file $R_HOME\library\clusterSim\pdf\indexC_details.pdf for further details

Thanks to Özge Sahin from Technical University of Munich for for pointing the difference between `index.G3` and `index.C`.

Value

calculated C index

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Hubert, L.J., Levin, J.R. (1976), A General Statistical Framework for Assessing Categorical Clustering in Free Recall, Psychological Bulletin, Vol. 83, No. 6, 1072-1080.

See Also

`index.G1`, `index.G2`, `index.G3`, `index.S`, `index.H`, `index.KL`, `index.Gap`, `index.DB`

Examples

```
# Example 1
library(clusterSim)
data(data_ratio)
d <- dist.GDM(data_ratio)
c <- pam(d, 5, diss = TRUE)
icq <- index.C(d,c$clustering)
print(icq)

# Example 2
library(clusterSim)
data(data_ordinal)
d <- dist.GDM(data_ordinal, method="GDM2")
# nc - number_of_clusters
min_nc=2
max_nc=6
res <- array(0,c(max_nc-min_nc+1, 2))
res[,1] <- min_nc:max_nc
clusters <- NULL
for (nc in min_nc:max_nc)
{
hc <- hclust(d, method="complete")
cl2 <- cutree(hc, k=nc)
res[nc-min_nc+1,2] <- C <- index.C(d,cl2)
clusters <- rbind(clusters,cl2)
}
print(paste("min C for",(min_nc:max_nc)[which.min(res[,2])],"clusters=",min(res[,2])))
print("clustering for min C-index")
print(clusters[which.min(res[,2]),])
#write.table(res,file="C_res.csv",sep=";",dec=",",row.names=TRUE,col.names=FALSE)
plot(res, type="p", pch=0, xlab="Number of clusters", ylab="C", xaxt="n")
axis(1, c(min_nc:max_nc))
```

`index.DB` *Calculates Davies-Bouldin's index*

Description

Calculates Davies-Bouldin's cluster separation measure

Usage

```
index.DB(x, cl, d=NULL, centrotypes="centroids", p=2, q=2)
```

Arguments

`x`	data
`cl`	vector of integers indicating the cluster to which each object is allocated
`d`	optional distance matrix, used for calculations if centrotypes="medoids"

`centrotypes`	"centroids" or "medoids"
`p`	the power of the Minkowski distance between centroids or medoids of clusters: p=1 - Manhattan distance; p=2 - Euclidean distance
`q`	the power of dispersion measure of a cluster: q=1 - the average distance of objects in the r-th cluster to the centroid or medoid of the r-th cluster; q=2 - the standard deviation of the distance of objects in the r-th cluster to the centroid or medoid of the r-th cluster

Details

See file `../doc/indexDB_details.pdf` for further details

Thanks to prof. Christian Hennig <`c.hennig@ucl.ac.uk`> for finding and fixing the "immutable p" error

Value

`DB`	Davies-Bouldin's index
`r`	vector of maximal R values for each cluster
`R`	R matrix $(S_r + S_s)/d_{rs}$
`d`	matrix of distances between centroids or medoids of clusters
`S`	vector of dispersion measures for each cluster
`centers`	coordinates of centroids or medoids for all clusters

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Davies, D.L., Bouldin, D.W. (1979), *A cluster separation measure*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 1, no. 2, 224-227. Available at: doi:10.1109/TPAMI.1979.4766909.

See Also

`index.G1`, `index.G2`, `index.G3`, `index.C`, `index.S`, `index.H`, `index.Gap`, `index.KL`

Examples

```
# Example 1
library(clusterSim)
data(data_ratio)
cl1 <- pam(data_ratio, 4)
d<-dist(data_ratio)
print(index.DB(data_ratio, cl1$clustering,d, centrotypes="medoids"))

# Example 2
```

```
library(clusterSim)
data(data_ratio)
cl2 <- pam(data_ratio, 5)
print(index.DB(data_ratio, cl2$clustering, centrotypes="centroids"))

# Example 3
library(clusterSim)
data(data_ratio)
md <- dist(data_ratio, method="euclidean")
# nc - number_of_clusters
min_nc=2
max_nc=8
res <- array(0, c(max_nc-min_nc+1, 2))
res[,1] <- min_nc:max_nc
clusters <- NULL
for (nc in min_nc:max_nc)
{
hc <- hclust(md, method="complete")
cl2 <- cutree(hc, k=nc)
res[nc-min_nc+1, 2] <- DB <- index.DB(data_ratio, cl2, centrotypes="centroids")$DB
clusters <- rbind(clusters, cl2)
}
print(paste("min DB for",(min_nc:max_nc)[which.min(res[,2])],"clusters=",min(res[,2])))
print("clustering for min DB")
print(clusters[which.min(res[,2]),])
#write.table(res,file="DB_res.csv",sep=";",dec=",",row.names=TRUE,col.names=FALSE)
plot(res, type="p", pch=0, xlab="Number of clusters", ylab="DB", xaxt="n")
axis(1, c(min_nc:max_nc))

# Example 4
library(clusterSim)
data(data_ordinal)
md <- dist.GDM(data_ordinal, method="GDM2")
# nc - number_of_clusters
min_nc=2
max_nc=6
res <- array(0, c(max_nc-min_nc+1, 2))
res[,1] <- min_nc:max_nc
clusters <- NULL
for (nc in min_nc:max_nc)
{
hc <- hclust(md, method="complete")
cl2 <- cutree(hc, k=nc)
res[nc-min_nc+1,2] <- DB <- index.DB(data_ordinal,cl2,d=md,centrotypes="medoids")$DB
clusters <- rbind(clusters, cl2)
}
print(paste("min DB for",(min_nc:max_nc)[which.min(res[,2])],"clusters=",min(res[,2])))
print("clustering for min DB")
print(clusters[which.min(res[,2]),])
#write.table(res,file="DB_res.csv",sep=";",dec=",",row.names=TRUE,col.names=FALSE)
plot(res, type="p", pch=0, xlab="Number of clusters", ylab="DB", xaxt="n")
axis(1, c(min_nc:max_nc))
```

`index.G1` *Calculates Calinski-Harabasz pseudo F-statistic*

Description

Calculates Calinski-Harabasz pseudo F-statistic

Usage

```
index.G1 (x,cl,d=NULL,centrotypes="centroids")
```

Arguments

`x`	data
`cl`	A vector of integers indicating the cluster to which each object is allocated
`d`	optional distance matrix, used for calculations if centrotypes="medoids"
`centrotypes`	"centroids" or "medoids"

Details

See file `../doc/indexG1_details.pdf` for further details.

thank to Nejc Ilc from University of Ljubljana for fixing error for one-element clusters.

Value

Calinski-Harabasz pseudo F-statistic

Author(s)

Marek Walesiak `<marek.walesiak@ue.wroc.pl>`, Andrzej Dudek `<andrzej.dudek@ue.wroc.pl>`

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Calinski, T., Harabasz, J. (1974), *A dendrite method for cluster analysis*, "Communications in Statistics", vol. 3, 1-27. Available at: doi:10.1080/03610927408827101.

Everitt, B.S., Landau, E., Leese, M. (2001), *Cluster analysis*, Arnold, London, p. 103. ISBN 9780340761199.

Gatnar, E., Walesiak, M. (Eds.) (2004), *Metody statystycznej analizy wielowymiarowej w badaniach marketingowych [Multivariate statistical analysis methods in marketing research]*, Wydawnictwo AE, Wroclaw, p. 338.

Gordon, A.D. (1999), *Classification*, Chapman & Hall/CRC, London, p. 62. ISBN 9781584880134.

Milligan, G.W., Cooper, M.C. (1985), *An examination of procedures of determining the number of cluster in a data set*, "Psychometrika", vol. 50, no. 2, 159-179. Available at: doi:10.1007/BF02294245.

See Also

index.G2,index.G3,index.S, index.C, index.H,index.KL,index.Gap, index.DB

Examples

```
# Example 1
library(clusterSim)
data(data_ratio)
c<- pam(data_ratio,10)
index.G1(data_ratio,c$clustering)

# Example 2
library(clusterSim)
data(data_ratio)
md <- dist(data_ratio, method="euclidean")
# nc - number_of_clusters
min_nc=2
max_nc=20
res <- array(0,c(max_nc-min_nc+1,2))
res[,1] <- min_nc:max_nc
clusters <- NULL
for (nc in min_nc:max_nc)
{
cl2 <- pam(md, nc, diss=TRUE)
res[nc-min_nc+1,2] <- G1 <- index.G1(data_ratio,cl2$cluster,centrotypes="centroids")
clusters <- rbind(clusters, cl2$cluster)
}
print(paste("max G1 for",(min_nc:max_nc)[which.max(res[,2])],"clusters=",max(res[,2])))
print("clustering for max G1")
print(clusters[which.max(res[,2]),])
#write.table(res,file="G1_res.csv",sep=";",dec=",",row.names=TRUE,col.names=FALSE)
plot(res, type="p", pch=0, xlab="Number of clusters", ylab="G1", xaxt="n")
axis(1, c(min_nc:max_nc))
```

`index.G2` *Calculates G2 internal cluster quality index*

Description

Calculates G2 internal cluster quality index - Baker & Hubert adaptation of Goodman & Kruskal's Gamma statistic

Usage

```
index.G2(d,cl)
```

Arguments

`d`	'dist' object
`cl`	A vector of integers indicating the cluster to which each object is allocated

Details

See file $R_HOME\library\clusterSim\pdf\indexG2_details.pdf for further details

Value

calculated G2 index

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Everitt, B.S., Landau, E., Leese, M. (2001), *Cluster analysis*, Arnold, London, p. 104. ISBN 9780340761199.

Gatnar, E., Walesiak, M. (Eds.) (2004), *Metody statystycznej analizy wielowymiarowej w badaniach marketingowych [Multivariate statistical analysis methods in marketing research]*, Wydawnictwo AE, Wroclaw, p. 339.

Gordon, A.D. (1999), *Classification*, Chapman & Hall/CRC, London, p. 62. ISBN 9781584880134.

Hubert, L. (1974), *Approximate evaluation technique for the single-link and complete-link hierarchical clustering procedures*, "Journal of the American Statistical Association", vol. 69, no. 347, 698-704. Available at: doi:10.1080/01621459.1974.10480191.

Milligan, G.W., Cooper, M.C. (1985), *An examination of procedures of determining the number of cluster in a data set*, "Psychometrika", vol. 50, no. 2, 159-179. Available at: doi:10.1007/BF02294245.

See Also

`index.G1`, `index.G3`, `index.S`, `index.H`, `index.KL`, `index.Gap`, `index.C`, `index.DB`

Examples

```
# Example 1
library(clusterSim)
data(data_ratio)
d <- dist.GDM(data_ratio)
c <- pam(d, 5, diss = TRUE)
icq <- index.G2(d,c$clustering)
#print(icq)

# Example 2
library(clusterSim)
data(data_ordinal)
d <- dist.GDM(data_ordinal, method="GDM2")
# nc - number_of_clusters
min_nc=2
max_nc=6
res <- array(0,c(max_nc-min_nc+1, 2))
```

```
res[,1] <- min_nc:max_nc
clusters <- NULL
for (nc in min_nc:max_nc)
{
  cl2 <- pam(d, nc, diss=TRUE)
  res[nc-min_nc+1,2] <- G2 <- index.G2(d,cl2$cluster)
  clusters <- rbind(clusters,cl2$cluster)
}
print(paste("max G2 for",(min_nc:max_nc)[which.max(res[,2])],"clusters=",max(res[,2])))
print("clustering for max G2")
print(clusters[which.max(res[,2]),])
plot(res, type="p", pch=0, xlab="Number of clusters", ylab="G2", xaxt="n")
axis(1, c(min_nc:max_nc))
```

`index.G3` *Calculates G3 internal cluster quality index*

Description

Calculates G3 internal cluster quality index

Usage

```
index.G3(d,cl)
```

Arguments

`d`	'dist' object
`cl`	A vector of integers indicating the cluster to which each object is allocated

Details

See file $R_HOME\library\clusterSim\pdf\indexG3_details.pdf for further details

Value

calculated G3 index

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Gordon, A.D. (1999), *Classification*, Chapman & Hall/CRC, London, p. 62. ISBN 9781584880134.

See Also

`index.G1`, `index.G2`, `index.S`, `index.C`, `index.H`, `index.KL`, `index.Gap`, `index.DB`

Examples

```
# Example 1
library(clusterSim)
data(data_ratio)
d <- dist.GDM(data_ratio)
c <- pam(d, 5, diss = TRUE)
icq <- index.G3(d,c$clustering)
print(icq)

# Example 2
library(clusterSim)
data(data_ordinal)
d <- dist.GDM(data_ordinal, method="GDM2")
# nc - number_of_clusters
min_nc=2
max_nc=6
res <- array(0,c(max_nc-min_nc+1, 2))
res[,1] <- min_nc:max_nc
clusters <- NULL
for (nc in min_nc:max_nc)
{
hc <- hclust(d, method="complete")
cl2 <- cutree(hc, k=nc)
res[nc-min_nc+1,2] <- G3 <- index.G3(d,cl2)
clusters <- rbind(clusters,cl2)
}
print(paste("min G3 for",(min_nc:max_nc)[which.min(res[,2])],"clusters=",min(res[,2])))
print("clustering for min G3")
print(clusters[which.min(res[,2]),])
#write.table(res,file="G3_res.csv",sep=";",dec=",",row.names=TRUE,col.names=FALSE)
plot(res, type="p", pch=0, xlab="Number of clusters", ylab="G3", xaxt="n")
axis(1, c(min_nc:max_nc))
```

`index.Gap` *Calculates Tibshirani, Walther and Hastie gap index*

Description

Calculates Tibshirani, Walther and Hastie gap index

Usage

```
index.Gap (x, clall, reference.distribution="unif", B=10,
method="pam",d=NULL,centrotypes="centroids")
```

Arguments

`x`	data
`clall`	Two vectors of integers indicating the cluster to which each object is allocated in partition of n objects into u, and u+1 clusters
`reference.distribution`	"unif" - generate each reference variable uniformly over the range of the observed values for that variable or "pc" - generate the reference variables from a uniform distribution over a box aligned with the principal components of the data. In detail, if $X = \{x_{ij}\}$ is our n x m data matrix, assume that the columns have mean 0 and compute the singular value decomposition $X=UDV^T$. We transform via $X'=XV$ and then draw uniform features Z' over the ranges of the columns of X' , as in method a) above. Finally we back-transform via $Z=Z'V^T$ to give reference data Z
`B`	the number of simulations used to compute the gap statistic
`method`	the cluster analysis method to be used. This should be one of: "ward.D", "ward.D2", "single", "complete", "average", "mcquitty", "median", "centroid", "pam", "k-means","diana"
`d`	optional distance matrix, used for calculations if centrotypes="medoids"
`centrotypes`	"centroids" or "medoids"

Details

See file `../doc/indexGap_details.pdf` for further details

Thanks to dr Michael P. Fay from National Institute of Allergy and Infectious Diseases for finding "one column error".

Value

`Gap`	Tibshirani, Walther and Hastie gap index for u clusters
`diffu`	necessary value for choosing correct number of clusters via gap statistic Gap(u)-[Gap(u+1)-s(u+1)]

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Tibshirani, R., Walther, G., Hastie, T. (2001), *Estimating the number of clusters in a data set via the gap statistic*, "Journal of the Royal Statistical Society", ser. B, vol. 63, part 2, 411-423. Available at: doi:10.1111/14679868.00293.

See Also

`index.G1`, `index.G2`, `index.G3`, `index.C`, `index.S`, `index.H`, `index.KL`, `index.DB`

Examples

```
# Example 1
library(clusterSim)
data(data_ratio)
cl1<-pam(data_ratio,4)
cl2<-pam(data_ratio,5)
clall<-cbind(cl1$clustering,cl2$clustering)
g<-index.Gap(data_ratio, clall, reference.distribution="unif", B=10,
   method="pam")
print(g)

# Example 2
library(clusterSim)
means <- matrix(c(0,2,4,0,3,6), 3, 2)
cov <- matrix(c(1,-0.9,-0.9,1), 2, 2)
x <- cluster.Gen(numObjects=40, means=means, cov=cov, model=2)
x <- x$data
md <- dist(x, method="euclidean")^2
# nc - number_of_clusters
min_nc=1
max_nc=5
min <- 0
clopt <- NULL
res <- array(0, c(max_nc-min_nc+1, 2))
res[,1] <- min_nc:max_nc
found <- FALSE
for (nc in min_nc:max_nc){
  cl1 <- pam(md, nc, diss=TRUE)
  cl2 <- pam(md, nc+1, diss=TRUE)
  clall <- cbind(cl1$clustering, cl2$clustering)
  gap <- index.Gap(x,clall,B=20,method="pam",centrotypes="centroids")
  res[nc-min_nc+1, 2] <- diffu <- gap$diffu
  if ((res[nc-min_nc+1, 2] >=0) && (!found)){
    nc1 <- nc
    min <- diffu
    clopt <- cl1$cluster
    found <- TRUE
  }
}
if (found){
print(paste("Minimal nc where diffu>=0 is",nc1,"for diffu=",round(min,4)),quote=FALSE)
}else{
print("I have not found clustering with diffu>=0", quote=FALSE)
}
plot(res,type="p",pch=0,xlab="Number of clusters",ylab="diffu",xaxt="n")
abline(h=0, untf=FALSE)
axis(1, c(min_nc:max_nc))

# Example 3
library(clusterSim)
means <- matrix(c(0,2,4,0,3,6), 3, 2)
cov <- matrix(c(1,-0.9,-0.9,1), 2, 2)
```

```
x <- cluster.Gen(numObjects=40, means=means, cov=cov, model=2)
x <- x$data
md <- dist(x, method="euclidean")^2
# nc - number_of_clusters
min_nc=1
max_nc=5
min <- 0
clopt <- NULL
res <- array(0, c(max_nc-min_nc+1, 2))
res[,1] <- min_nc:max_nc
found <- FALSE
for (nc in min_nc:max_nc){
  cl1 <- pam(md, nc, diss=TRUE)
  cl2 <- pam(md, nc+1, diss=TRUE)
  clall <- cbind(cl1$clustering, cl2$clustering)
  gap <- index.Gap(x,clall,B=20,method="pam",d=md,centrotypes="medoids")
  res[nc-min_nc+1, 2] <- diffu <- gap$diffu
  if ((res[nc-min_nc+1, 2] >=0) && (!found)){
    nc1 <- nc
    min <- diffu
    clopt <- cl1$cluster
    found <- TRUE
  }
}
if (found){
print(paste("Minimal nc where diffu>=0 is",nc1,"for diffu=",round(min,4)),quote=FALSE)
}else{
print("I have not found clustering with diffu>=0",quote=FALSE)
}
plot(res, type="p", pch=0, xlab="Number of clusters", ylab="diffu", xaxt="n")
abline(h=0, untf=FALSE)
axis(1, c(min_nc:max_nc))
```

`index.H` *Calculates Hartigan index*

Description

Calculates Hartigan index

Usage

```
index.H (x,clall,d=NULL,centrotypes="centroids")
```

Arguments

`x`	data
`clall`	Two vectors of integers indicating the cluster to which each object is allocated in partition of n objects into u and u+1 clusters
`d`	optional distance matrix, used for calculations if centrotypes="medoids"
`centrotypes`	"centroids" or "medoids"

Details

See file $R_HOME\library\clusterSim\pdf\indexH_details.pdf for further details

Value

Hartigan index

Author(s)

Marek Walesiak `<marek.walesiak@ue.wroc.pl>`, Andrzej Dudek `<andrzej.dudek@ue.wroc.pl>`

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Hartigan, J. (1975), *Clustering algorithms*, Wiley, New York. ISBN 047135645X.

Milligan, G.W., Cooper, M.C. (1985), *An examination of procedures of determining the number of cluster in a data set*, "Psychometrika", vol. 50, no. 2, 159-179. Available at: doi:10.1007/BF02294245.

Tibshirani, R., Walther, G., Hastie, T. (2001), *Estimating the number of clusters in a data set via the gap statistic*, "Journal of the Royal Statistical Society", ser. B, vol. 63, part 2, 411-423. Available at: doi:10.1111/14679868.00293.

See Also

`index.G1`, `index.G2`, `index.G3`, `index.C`, `index.S`, `index.KL`, `index.Gap`, `index.DB`

Examples

```
# Example 1
library(clusterSim)
data(data_ratio)
cl1<-pam(data_ratio,4)
cl2<-pam(data_ratio,5)
clall<-cbind(cl1$clustering,cl2$clustering)
index.H(data_ratio,clall)

# Example 2
library(clusterSim)
data(data_ratio)
md <- dist(data_ratio, method="euclidean")
# nc - number_of_clusters
min_nc=1
max_nc=20
min <- 0
res <- array(0, c(max_nc-min_nc+1, 2))
res[,1] <- min_nc:max_nc
found <- FALSE
clusters <- NULL
for (nc in min_nc:max_nc)
{
```

```
print(nc)
hc <- hclust(md, method="complete")
cl1 <- cutree(hc, k=nc)
cl2 <- cutree(hc, k=nc+1)
clall <- cbind(cl1,cl2)
res[nc-min_nc+1,2] <- H <- index.H(data_ratio,clall,centrotypes="centroids")
if ((res[nc-min_nc+1, 2]<10) && (!found)){
       nc1 <- nc
       min <- H
       clopt <- cl1
   found <- TRUE
}
}
if (found)
{
print(paste("minimal nc for H<=10 equals",nc1,"for H=",min))
print("clustering for minimal nc where H<=10")
print(clopt)
}else
{
print("Clustering not found with H<=10")
}
#write.table(res,file="H_res.csv",sep=";",dec=",",row.names=TRUE,col.names=FALSE)
plot(res,type="p",pch=0,xlab="Number of clusters",ylab="H",xaxt="n")
abline(h=10, untf=FALSE)
axis(1, c(min_nc:max_nc))

# Example 3
library(clusterSim)
data(data_ratio)
md <- dist(data_ratio, method="manhattan")
# nc - number_of_clusters
min_nc=1
max_nc=20
min <- 0
res <- array(0, c(max_nc-min_nc+1, 2))
res[,1] <- min_nc:max_nc
found <- FALSE
clusters <- NULL
for (nc in min_nc:max_nc)
{
print(nc)
hc <- hclust(md, method="complete")
cl1 <- cutree(hc, k=nc)
cl2 <- cutree(hc, k=nc+1)
clall <- cbind(cl1,cl2)
res[nc-min_nc+1,2] <- H <- index.H(data_ratio,clall,d=md,centrotypes="medoids")
if ((res[nc-min_nc+1, 2]<10) && (!found)){
       nc1 <- nc
       min <- H
       clopt <- cl1
   found <- TRUE
}
```

```
}
if (found)
{
print(paste("minimal nc for H<=10 equals",nc1,"for H=",min))
print("clustering for minimal nc where H<=10")
print(clopt)
}else
{
print("Clustering not found with H<=10")
}
#write.table(res,file="H_res.csv",sep=";",dec=",",row.names=TRUE,col.names=FALSE)
plot(res,type="p",pch=0,xlab="Number of clusters",ylab="H",xaxt="n")
abline(h=10, untf=FALSE)
axis(1, c(min_nc:max_nc))
```

`index.KL` *Calculates Krzanowski-Lai index*

Description

Calculates Krzanowski-Lai index

Usage

```
index.KL (x,clall,d=NULL,centrotypes="centroids")
```

Arguments

`x`	data
`clall`	Three vectors of integers indicating the cluster to which each object is allocated in partition of n objects into u-1, u, and u+1 clusters
`d`	optional distance matrix, used for calculations if centrotypes="medoids"
`centrotypes`	"centroids" or "medoids"

Details

See file `../doc/indexKL_details.pdf` for further details

Value

Krzanowski-Lai index

Author(s)

Marek Walesiak `<marek.walesiak@ue.wroc.pl>`, Andrzej Dudek `<andrzej.dudek@ue.wroc.pl>`

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Krzanowski, W.J., Lai, Y.T. (1988), *A criterion for determining the number of groups in a data set using sum of squares clustering*, "Biometrics", 44, 23-34.

Milligan, G.W., Cooper, M.C. (1985), *An examination of procedures of determining the number of cluster in a data set*, "Psychometrika", vol. 50, no. 2, 159-179. Available at: doi:10.1007/BF02294245.

Tibshirani, R., Walther, G., Hastie, T. (2001), *Estimating the number of clusters in a data set via the gap statistic*, "Journal of the Royal Statistical Society", ser. B, vol. 63, part 2, 411-423. Available at: doi:10.1111/14679868.00293.

See Also

`index.G1`, `index.G2`, `index.G3`, `index.C`, `index.S`, `index.H`, `index.Gap`, `index.DB`

Examples

```
# Example 1
library(clusterSim)
data(data_ratio)
cl1<-pam(data_ratio,4)
cl2<-pam(data_ratio,5)
cl3<-pam(data_ratio,6)
clall<-cbind(cl1$clustering,cl2$clustering,cl3$clustering)
index.KL(data_ratio,clall)

# Example 2
library(clusterSim)
data(data_ratio)
md <- dist(data_ratio, method="manhattan")
# nc - number_of_clusters
min_nc=2
max_nc=15
res <- array(0, c(max_nc-min_nc+1, 2))
res[,1] <- min_nc:max_nc
clusters <- NULL
for (nc in min_nc:max_nc)
{
  if(nc-1==1){
    clustering1<-rep(1,nrow(data_ratio))
  }
  else{
    clustering1 <- pam(md, nc-1, diss=TRUE)$clustering
  }
  clustering2 <- pam(md, nc, diss=TRUE)$clustering
  clustering3 <- pam(md, nc+1, diss=TRUE)$clustering
  clall<- cbind(clustering1, clustering2, clustering3)
  res[nc-min_nc+1,2] <- KL <- index.KL(data_ratio,clall,centrotypes="centroids")
  clusters <- rbind(clusters, clustering2)
}
print(paste("max KL for",(min_nc:max_nc)[which.max(res[,2])],"clusters=",max(res[,2])))
print("clustering for max KL")
```

```
print(clusters[which.max(res[,2]),])
#write.table(res,file="KL_res.csv",sep=";",dec=",",row.names=TRUE,col.names=FALSE)
plot(res,type="p",pch=0,xlab="Number of clusters",ylab="KL",xaxt="n")
axis(1, c(min_nc:max_nc))

# Example 3
library(clusterSim)
data(data_ratio)
md <- dist(data_ratio, method="manhattan")
# nc - number_of_clusters
min_nc=2
max_nc=15
res <- array(0, c(max_nc-min_nc+1, 2))
res[,1] <- min_nc:max_nc
clusters <- NULL
for (nc in min_nc:max_nc)
{
  if(nc-1==1){
    clustering1<-rep(1,nrow(data_ratio))
  }
  else{
    clustering1 <- pam(md, nc-1, diss=TRUE)$clustering
  }
  clustering2 <- pam(md, nc, diss=TRUE)$clustering
  clustering3 <- pam(md, nc+1, diss=TRUE)$clustering
  clall<- cbind(clustering1, clustering2, clustering3)
  res[nc-min_nc+1,2] <- KL <- index.KL(data_ratio,clall,d=md,centrotypes="medoids")
  clusters <- rbind(clusters, clustering2)
}
print(paste("max KL for",(min_nc:max_nc)[which.max(res[,2])],"clusters=",max(res[,2])))
print("clustering for max KL")
print(clusters[which.max(res[,2]),])
#write.table(res,file="KL_res.csv",sep=";",dec=",",row.names=TRUE,col.names=FALSE)
plot(res,type="p",pch=0,xlab="Number of clusters",ylab="KL",xaxt="n")
axis(1, c(min_nc:max_nc))
```

`index.S` *Calculates Rousseeuw's Silhouette internal cluster quality index*

Description

Calculates Rousseeuw's Silhouette internal cluster quality index

Usage

```
index.S(d,cl,singleObject=0)
```

Arguments

`d`	'dist' object
`cl`	A vector of integers indicating the cluster to which each object is allocated
`singleObject`	0 - s(i)=0 or 1 - s(i)=1. When cluster contains a single object, it is unclear how a(i) of Silhouette index should be defined (see Kaufman & Rousseeuw (1990), p. 85).

Details

See file $R_HOME\library\clusterSim\pdf\indexS_details.pdf for further details

Value

calculated Silhouette index

Author(s)

Marek Walesiak `<marek.walesiak@ue.wroc.pl>`, Andrzej Dudek `<andrzej.dudek@ue.wroc.pl>`

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Gatnar, E., Walesiak, M. (Eds.) (2004), *Metody statystycznej analizy wielowymiarowej w badaniach marketingowych [Multivariate statistical analysis methods in marketing research]*, Wydawnictwo AE, Wroclaw, 342-343, erratum.

Kaufman, L., Rousseeuw, P.J. (1990), *Finding groups in data: an introduction to cluster analysis*, Wiley, New York, pp. 83-88. ISBN: 978-0-471-73578-6.

See Also

`index.G1`, `index.G2`, `index.G3`, `index.C`, `index.KL`, `index.H`, `index.Gap`, `index.DB`

Examples

```
# Example 1
library(clusterSim)
data(data_ratio)
d <- dist.GDM(data_ratio)
c <- pam(d, 5, diss = TRUE)
icq <- index.S(d,c$clustering)
print(icq)

# Example 2
library(clusterSim)
data(data_ratio)
md <- dist(data_ratio, method="manhattan")
# nc - number_of_clusters
min_nc=2
max_nc=20
```

```
res <- array(0, c(max_nc-min_nc+1, 2))
res[,1] <- min_nc:max_nc
clusters <- NULL
for (nc in min_nc:max_nc)
{
cl2 <- pam(md, nc, diss=TRUE)
res[nc-min_nc+1, 2] <- S <- index.S(md,cl2$cluster)
clusters <- rbind(clusters, cl2$cluster)
}
print(paste("max S for",(min_nc:max_nc)[which.max(res[,2])],"clusters=",max(res[,2])))
print("clustering for max S")
print(clusters[which.max(res[,2]),])
#write.table(res,file="S_res.csv",sep=";",dec=",",row.names=TRUE,col.names=FALSE)
plot(res,type="p",pch=0,xlab="Number of clusters",ylab="S",xaxt="n")
axis(1, c(min_nc:max_nc))
```

`initial.Centers` *Calculation of initial clusters centers for k-means like alghoritms*

Description

Function calculates initial clusters centers for k-means like alghoritms with the following alghoritm (similar to SPSS QuickCluster function)

(a) if the distance between x_k and its closest cluster center is greater than the distance between the two closest centers (M_m and M_n), then x_k replaces either M_m or M_n, whichever is closer to x_k.

(b) If x_k does not replace a cluster initial center in (a), a second test is made: If that distance d_q greater than the distance between M_q and its closest M_i, then x_k replaces M_q.

where:

M_i - initial center of *i-th* cluster

x_k - vector of *k-th* observation

$d(..., ...)$ - Euclidean distance

d_{mn} = $min_{ij} d(M_i, M_j)$

$d_q = min_i d(x_k, M_i)$

Usage

```
initial.Centers(x, k)
```

Arguments

`x`	matrix or dataset
`k`	number of initial cluster centers

Value

Numbers of objects choosen as initial cluster centers

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Hartigan, J. (1975), *Clustering algorithms*, Wiley, New York. ISBN 0-471-35645-X.

See Also

`cluster.Sim`

Examples

```
#Example 1 (numbers of objects choosen as initial cluster centers)
library(clusterSim)
data(data_ratio)
ic <- initial.Centers(data_ratio, 10)
print(ic)

#Example 2 (application with kmeans algorithm)
library(clusterSim)
data(data_ratio)
kmeans(data_ratio,data_ratio[initial.Centers(data_ratio, 10),])
```

`interval_normalization` *Types of normalization formulas for interval-valued symbolic variables*

Description

Types of normalization formulas for interval-valued symbolic variables

Usage

```
interval_normalization(x,dataType="simple",type="n0",y=NULL,...)
```

Arguments

`x`	matrix dataset or symbolic table object
`dataType`	Type of symbolic data table passed to function, 'sda' - full symbolicDA format object; 'simple' - three dimensional array with lower and upper bound of intervals in third dimension; 'separate_tables' - lower bounds of intervals in `x`, upper bounds in `y`;

	'rows' - lower and upper bound of intervals in neighbouring rows; 'columns' - lower and upper bound of intervals in neighbouring columns
`type`	type of normalization:

n0 - without normalization

n1 - standardization ((x-mean)/sd)

n2 - positional standardization ((x-median)/mad)

n3 - unitization ((x-mean)/range)

n3a - positional unitization ((x-median)/range)

n4 - unitization with zero minimum ((x-min)/range)

n5 - normalization in range <-1,1> ((x-mean)/max(abs(x-mean)))

n5a - positional normalization in range <-1,1> ((x-median)/max(abs(x-median)))

n6 - quotient transformation (x/sd)

n6a - positional quotient transformation (x/mad)

n7 - quotient transformation (x/range)

n8 - quotient transformation (x/max)

n9 - quotient transformation (x/mean)

n9a - positional quotient transformation (x/median)

n10 - quotient transformation (x/sum)

n11 - quotient transformation (x/sqrt(SSQ))

n12 - normalization ((x-mean)/sqrt(sum((x-mean)^2)))

n12a - positional normalization ((x-median)/sqrt(sum((x-median)^2)))

n13 - normalization with zero being the central point ((x-midrange)/(range/2))

`y`	matrix or dataset with upper bounds of intervals if argument `dataType` is uuqual to "separate_tables"
`...`	arguments passed to `sum`, `mean`, `min` `sd`, `mad` and other aggregation functions. In particular: `na.rm` - a logical value indicating whether NA values should be stripped before the computation

Value

Normalized data

Author(s)

Marek Walesiak `<marek.walesiak@ue.wroc.pl>`, Andrzej Dudek `<andrzej.dudek@ue.wroc.pl>`

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Jajuga, K., Walesiak, M. (2000), *Standardisation of data set under different measurement scales*, In: R. Decker, W. Gaul (Eds.), Classification and information processing at the turn of the millennium, Springer-Verlag, Berlin, Heidelberg, 105-112. Available at: doi:10.1007/9783642572807_11.

Milligan, G.W., Cooper, M.C. (1988), *A study of standardization of variables in cluster analysis*, "Journal of Classification", vol. 5, 181-204. Available at: doi:10.1007/BF01897163.

Walesiak, M. (2014), *Przeglad formul normalizacji wartosci zmiennych oraz ich wlasnosci w statystycznej analizie wielowymiarowej [Data normalization in multivariate data analysis. An overview and properties]*, "Przeglad Statystyczny" ("Statistical Review"), vol. 61, no. 4, 363-372. Available at: doi:10.5604/01.3001.0016.1740.

Walesiak, M., Dudek, A. (2017), *Selecting the Optimal Multidimensional Scaling Procedure for Metric Data with R Environment*, „STATISTICS IN TRANSITION new series", September, Vol. 18, No. 3, pp. 521-540. Available at: doi:10.59170/stattrans2017027.

See Also

`data.Normalization`

Examples

```
library(clusterSim)
data(data_symbolic_interval_polish_voivodships)
n<-interval_normalization(data_symbolic_interval_polish_voivodships,dataType="simple",type="n2")
plotInterval(n$simple)
```

`ordinalToMetric` *Reinforcing measurement scale for ordinal data*

Description

Reinforcing measurement scale for ordinal data (ordinal to metric scale)

Usage

```
ordinalToMetric(data,scaleType="o",patternCoordinates)
```

Arguments

`data`	matrix or dataset
`scaleType`	"o" - variables measured on ordinal scale, "m" - variables measured on metric scale, "o/m" - vector with mixed variables - e.g. c("o","m","m","o","o","m")
`patternCoordinates`	vector containing pattern coordinates c(...) given by the reaseracher for data (for metric variables - NA, for ordinal variables - one of the categories for each ordinal variable (e.g. maximum category))

Details

See file `../doc/ordinalToMetric_details.pdf` for further details

Value

`pdata`	raw (primary) data matrix
`tdata`	data matrix after transformation of ordinal variables into metric variables
`cpattern`	vector containing pattern coordinates

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, Wroclaw University of Economics, Poland

References

Jajuga, K., Walesiak, M., Bak, A. (2003), *On the general distance measure*, In: M. Schwaiger, O. Opitz (Eds.), Exploratory data analysis in empirical research, Springer-Verlag, Berlin, Heidelberg, 104-109. Available at: doi:10.1007/9783642557217_12.

Walesiak, M. (1993), *Statystyczna analiza wielowymiarowa w badaniach marketingowych [Multivariate statistical analysis in marketing research]*. Wroclaw University of Economics, Research Papers no. 654.

Walesiak, M. (1999), *Distance Measure for Ordinal Data*, "Argumenta Oeconomica", No. 2 (8), 167-173.

Walesiak, M. (2011), *Uogólniona miara odległości GDM w statystycznej analizie wielowymiarowej z wykorzystaniem programu R [The Generalized Distance Measure GDM in multivariate statistical analysis with R]*, Wydawnictwo Uniwersytetu Ekonomicznego, Wroclaw.

Walesiak, M. (2014), *Wzmacnianie skali pomiaru dla danych porządkowych w statystycznej analizie wielowymiarowej [Reinforcing measurement scale for ordinal data in multivariate statistical analysis]*, Taksonomia 22, Prace Naukowe Uniwersytetu Ekonomicznego we Wrocławiu no. 327, 60-68.

See Also

`dist.GDM`

Examples

```
# Example 1
library(clusterSim)
data(data_patternGDM2)
res1<-ordinalToMetric(data_patternGDM2,scaleType="o",patternCoordinates=c(5,4,3,1,1,3))
print(res1)

# Example 2
library(clusterSim)
data(data_patternGDM2)
res2<-ordinalToMetric(data_patternGDM2,scaleType="o",patternCoordinates=c(5,4,3,4,2,4))
```

```
print(res2)
```

`pattern.GDM1` *An application of GDM1 distance for metric data to compute the distances of objects from the pattern object (upper or lower)*

Description

An application of GDM1 distance for metric data to compute the distances of objects from the upper (ideal point co-ordinates) or lower (anti-ideal point co-ordinates) pattern object

Usage

```
pattern.GDM1(data, performanceVariable, scaleType="i",
nomOptValues=NULL, weightsType="equal", weights=NULL,
normalization="n0", patternType="upper",
patternCoordinates="dataBounds", patternManual=NULL,
nominalTransfMethod=NULL)
```

Arguments

`data`	matrix or dataset
`performanceVariable`	vector containing three types of performance variables: `s` for stimulants where higher value means better performance `d` for destimulants where low values indicate better performance `n` for nominants where the best value is implied. Object performance is positively assessed if the measure has implied value
`scaleType`	"i" - variables measured on interval scale, "r" - variables measured on ratio scale, "r/i" - vector with mixed variables
`nomOptValues`	vector containing optimal values for nominant variables and NA values for stimulants and destimulants. If `performanceVariable` do not contain nominant variables this `nomOptValues` may be set to NULL
`weightsType`	equal or different1 or different2 "equal" - equal weights "different1" - vector of different weights should satisfy conditions: each weight takes value from interval [0; 1] and sum of weights equals one "different2" - vector of different weights should satisfy conditions: each weight takes value from interval [0; m] and sum of weights equals m (m - the number of variables)
`normalization`	normalization formulas as in `data.Normalization` function
`weights`	vector of weights
`patternType`	"upper" - ideal point co-ordinates consists of the best variables' values "lower" - anti-ideal point co-ordinates consists of the worst variables' values

`patternCoordinates`	"dataBounds" - pattern should be calculated as following: "upper" pattern (maximum for stimulants, minimum for destimulants), "lower" pattern (minimum for stimulants, maximum for destimulants) "manual" - pattern should be given in `patternManual` variable
`patternManual`	Pattern co-ordinates contain: real numbers "min" - for minimal value of variable "max" - for maximal value of variable
`nominalTransfMethod`	method of transformation of nominant to stimulant variable: "q" - quotient transformation "d" - difference transformation

Details

See file `../doc/patternGDM1_details.pdf` for further details

Value

`pdata`	raw (primary) data matrix
`tdata`	data matrix after transformation of nominant variables (with pattern in last row)
`data`	data matrix after normalization (with pattern in last row)
`distances`	GDM1 distances from pattern object
`sortedDistances`	sorted GDM1 distances from pattern object

Author(s)

Marek Walesiak `<marek.walesiak@ue.wroc.pl>`, Andrzej Dudek `<andrzej.dudek@ue.wroc.pl>`

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Jajuga, K., Walesiak, M., Bak, A. (2003), *On the general distance measure*, In: M. Schwaiger, O. Opitz (Eds.), Exploratory data analysis in empirical research, Springer-Verlag, Berlin, Heidelberg, 104-109. Available at: doi:10.1007/9783642557217_12.

Walesiak, M. (1993), *Statystyczna analiza wielowymiarowa w badaniach marketingowych [Multivariate statistical analysis in marketing research].* Wroclaw University of Economics, Research Papers no. 654.

Walesiak, M. (2006), *Uogolniona miara odleglosci w statystycznej analizie wielowymiarowej [The Generalized Distance Measure in multivariate statistical analysis]*, Wydawnictwo AE, Wroclaw.

Walesiak, M. (2011), *Uogólniona miara odległości GDM w statystycznej analizie wielowymiarowej z wykorzystaniem programu R [The Generalized Distance Measure GDM in multivariate statistical analysis with R]*, Wydawnictwo Uniwersytetu Ekonomicznego, Wroclaw.

Walesiak, M. (2016), *Uogólniona miara odległości GDM w statystycznej analizie wielowymiarowej z wykorzystaniem programu R. Wydanie 2 poprawione i rozszerzone [The Generalized Distance Measure GDM in multivariate statistical analysis with R]*, Wydawnictwo Uniwersytetu Ekonomicznego, Wroclaw.

See Also

`dist.GDM`,`data.Normalization`

Examples

```
# Example 1
library(clusterSim)
data(data_patternGDM1)
res<-pattern.GDM1(data_patternGDM1,
performanceVariable=c("s","s","s","s","s","s","d","d","s","s"),
scaleType="r",nomOptValues=NULL,weightsType<-"equal",weights=NULL,
normalization<-"n4",patternType<-"lower",patternCoordinates<-"manual",
patternManual<-c("min","min","min","min","min","min","max","max","min","min"),
nominalTransfMethod <-NULL)
print(res)
gdm_p<-res$distances
plot(cbind(gdm_p,gdm_p),xlim=c(max(gdm_p),min(gdm_p)),
ylim=c(min(gdm_p),max(gdm_p)),xaxt="n",
xlab="Order of objects from the best to the worst",
ylab="GDM distances from pattern object", lwd=1.6)
axis(1, at=gdm_p,labels=names(gdm_p), cex.axis=0.5)

# Example 2
library(clusterSim)
data(data_patternGDM1)
res<-pattern.GDM1(data_patternGDM1,
performanceVariable=c("s","s","s","s","s","s","d","d","s","s"),
scaleType="r",nomOptValues=NULL,weightsType<-"equal",weights=NULL,
normalization<-"n2",patternType<-"upper",
patternCoordinates<-"dataBounds",patternManual<-NULL,
nominalTransfMethod<-NULL)
print(res)
gdm_p<-res$distances
plot(cbind(gdm_p,gdm_p),xlim=c(min(gdm_p),max(gdm_p)),
ylim=c(min(gdm_p),max(gdm_p)),xaxt="n",
xlab="Order of objects from the best to the worst",
ylab="GDM distances from pattern object", lwd=1.6)
axis(1, at=gdm_p,labels=names(gdm_p), cex.axis=0.5)

# Example 3
library(clusterSim)
data(data_patternGDM1)
res<-pattern.GDM1(data_patternGDM1,
performanceVariable=c("s","s","s","s","s","s","d","d","s","s"),
scaleType="r",nomOptValues=NULL,weightsType<-"different2",
weights=c(1.1,1.15,1.15,1.1,1.1,0.7,0.7,1.2,0.8,1.0),
```

```
normalization<-"n6",patternType<-"upper",patternCoordinates<-"manual",
patternManual<-c(100,100,100,100,100,"max","min","min","max","max"),
nominalTransfMethod <-NULL)
print(res)
gdm_p<-res$distances
plot(cbind(gdm_p,gdm_p),xlim=c(min(gdm_p),max(gdm_p)),
ylim=c(min(gdm_p),max(gdm_p)),xaxt="n",
xlab="Order of objects from the best to the worst",
ylab="GDM distances from pattern object", lwd=1.6)
axis(1, at=gdm_p,labels=names(gdm_p), cex.axis=0.5)
```

`pattern.GDM2` *An application of GDM2 distance for ordinal data to compute the distances of objects from the pattern object (upper or lower)*

Description

An application of GDM2 distance for ordinal data to compute the distances of objects from the upper (ideal point co-ordinates) or lower (anti-ideal point co-ordinates) pattern object

Usage

```
pattern.GDM2(data, performanceVariable, nomOptValues=NULL,
weightsType="equal", weights=NULL, patternType="upper",
patternCoordinates="dataBounds", patternManual=NULL,
nominalTransfMethod=NULL)
```

Arguments

`data`	matrix or dataset
`performanceVariable`	vector containing three types of performance variables: `s` for stimulants where higher value means better performance `d` for destimulants where low values indicate better performance `n` for nominants where the best value is implied. Object performance is positively assessed if the measure has implied value
`nomOptValues`	vector containing optimal values for nominant variables and NA values for stimulants and destimulants. If `performanceVariable` do not contain nominant variables this `nomOptValues` may be set to NULL
`weightsType`	equal or different1 or different2 "equal" - equal weights "different1" - vector of different weights should satisfy conditions: each weight takes value from interval [0; 1] and sum of weights equals one "different2" - vector of different weights should satisfy conditions: each weight takes value from interval [0; m] and sum of weights equals m (m - the number of variables)

`weights`	vector of weights
`patternType`	"upper" - ideal point co-ordinates consists of the best variables' values "lower" - anti-ideal point co-ordinates consists of the worst variables' values
`patternCoordinates`	"dataBounds" - pattern should be calculated as following: "upper" pattern (maximum for stimulants, minimum for destimulants, nominal value for nominants), "lower" pattern (minimum for stimulants, maximum for destimulants) "manual" - pattern should be given in `patternManual` variable
`patternManual`	Pattern co-ordinates contain: real numbers "min" - for minimal value of variable "max" - for maximal value of variable "nom" - for nominal value of variable (for upper pattern only - given in `nomOptValues` vector)
`nominalTransfMethod`	method of transformation of nominant to destimulant variable for patternType="lower": "database" - for each nominant separately GDM2 distance is calculated between each nominant observation (with repetitions - all variable values are used in calculation) and nominal value. Next the variable observations are replaced by those distances "symmetrical" - for each nominant separately GDM2 distance is calculated between each nominant observation (without repetition - each observation is used once) and nominal value. Next the variable observations are replaced by those distances

Details

See file `../doc/patternGDM2_details.pdf` for further details

Value

`pdata`	raw (primary) data matrix
`data`	data matrix after transformation of nominant variables (with pattern in last row)
`distances`	GDM2 distances from pattern object
`sortedDistances`	sorted GDM2 distances from pattern object

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

epartment of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Jajuga, K., Walesiak, M., Bak, A. (2003), *On the general distance measure*, In: M. Schwaiger, O. Opitz (Eds.), Exploratory data analysis in empirical research, Springer-Verlag, Berlin, Heidelberg, 104-109. Available at: doi:10.1007/9783642557217_12.

Walesiak, M. (1993), *Statystyczna analiza wielowymiarowa w badaniach marketingowych [Multivariate statistical analysis in marketing research].* Wroclaw University of Economics, Research Papers no. 654.

Walesiak, M. (1999), *Distance Measure for Ordinal Data*, "Argumenta Oeconomica", No. 2 (8), 167-173.

Walesiak, M. (2006), *Uogolniona miara odleglosci w statystycznej analizie wielowymiarowej [The Generalized Distance Measure in multivariate statistical analysis]*, Wydawnictwo AE, Wroclaw.

Walesiak, M. (2011), *Uogólniona miara odległości GDM w statystycznej analizie wielowymiarowej z wykorzystaniem programu R [The Generalized Distance Measure GDM in multivariate statistical analysis with R]*, Wydawnictwo Uniwersytetu Ekonomicznego, Wroclaw.

Walesiak, M. (2016), *Uogólniona miara odległości GDM w statystycznej analizie wielowymiarowej z wykorzystaniem programu R. Wydanie 2 poprawione i rozszerzone [The Generalized Distance Measure GDM in multivariate statistical analysis with R]*, Wydawnictwo Uniwersytetu Ekonomicznego, Wroclaw.

See Also

`dist.GDM`

Examples

```
# Example 1
library(clusterSim)
data(data_patternGDM2)
res<-pattern.GDM2(data_patternGDM2,
performanceVariable=c("s","s","s","d","d","n"),
nomOptValues=c(NA,NA,NA,NA,NA,3), weightsType<-"equal", weights=NULL,
patternType="lower", patternCoordinates="manual",
patternManual=c("min","min",0,5,"max","max"),
nominalTransfMethod="symmetrical")
print(res)
gdm_p<-res$distances
plot(cbind(gdm_p,gdm_p),xlim=c(max(gdm_p),min(gdm_p)),
ylim=c(min(gdm_p),max(gdm_p)),
xaxt="n",xlab="Order of objects from the best to the worst",
ylab="GDM distances from pattern object", lwd=1.6)
axis(1, at=gdm_p,labels=names(gdm_p), cex.axis=0.5)

# Example 2
library(clusterSim)
data(data_patternGDM2)
res<-pattern.GDM2(data_patternGDM2,
performanceVariable=c("s","s","s","d","d","n"),
nomOptValues=c(NA,NA,NA,NA,NA,3), weightsType<-"equal", weights=NULL,
patternType="upper", patternCoordinates="dataBounds",
```

```
patternManual=NULL, nominalTransfMethod="database")
print(res)
gdm_p<-res$distances
plot(cbind(gdm_p,gdm_p), xlim=c(min(gdm_p),max(gdm_p)),
ylim=c(min(gdm_p),max(gdm_p)),
xaxt="n",xlab="Order of objects from the best to the worst",
ylab="GDM distances from pattern object", lwd=1.6)
axis(1, at=gdm_p,labels=names(gdm_p), cex.axis=0.5)
```

`plotCategorial` *Plot categorial data on a scatterplot matrix*

Description

Plot categorial data on a scatterplot matrix (optionally with clusters)

Usage

```
plotCategorial(x, pairsofVar=NULL, cl=NULL, clColors=NULL,...)
```

Arguments

`x`	data matrix (rows correspond to observations and columns correspond to variables)
`pairsofVar`	pairs of variables - all variables (`pairsofVar=NULL`) or selected variables, e.g. `pairsofVar=c(1,3,4)`
`cl`	cluster membership vector
`clColors`	The colors of clusters. The colors are given arbitrary (`clColors=TRUE`) or by hand, e.g. `clColors=c("red","blue","green")`. The number of colors equals the number of clusters
`...`	Arguments to be passed to methods, such as graphical parameters (see `par`).

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

See Also

`colors`, `pairs`

Examples

```
# Example 1
library(clusterSim)
data(data_ordinal)
plotCategorial(data_ordinal, pairsofVar=c(1,3,4,9), cl=NULL,
clColors = NULL)

# Example 2
library(clusterSim)
grnd <- cluster.Gen(50,model=5,dataType="o",numCategories=5)
plotCategorial(grnd$data, pairsofVar=NULL, cl=grnd$clusters,
clColors=TRUE)

# Example 3
library(clusterSim)
grnd<-cluster.Gen(50,model=4,dataType="o",numCategories=7, numNoisyVar=2)
plotCategorial(grnd$data, pairsofVar=NULL, cl=grnd$clusters,
clColors = c("red","blue","green"))
```

`plotInterval` *Plot symbolic interval-valued data on a scatterplot matrix*

Description

Plot symbolic interval-valued data on a scatterplot matrix (optionally with clusters)

Usage

```
plotInterval(x, pairsofsVar=NULL, cl=NULL, clColors=NULL,...)
```

Arguments

`x`	symbolic interval-valued data
`pairsofsVar`	pairs of symbolic interval variables - all variables (`pairsofsVar=NULL`) or selected variables, e.g. `pairsofsVar=c(1,3,4)`
`cl`	cluster membership vector
`clColors`	The colors of clusters. The colors are given arbitrary (`clColors=TRUE`) or by hand, e.g. `clColors=c("red","blue","green")`. The number of colors equals the number of clusters
`...`	Arguments to be passed to methods, such as graphical parameters (see `par`).

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

See Also

plotCategorial, colors, pairs

Examples

```
# Example 1
library(clusterSim)
data(data_symbolic)
plotInterval(data_symbolic, pairsofsVar=c(1,3,4,6), cl=NULL,
clColors=NULL)

# Example 2
library(clusterSim)
grnd <- cluster.Gen(60, model=5, dataType="s", numNoisyVar=1,
numOutliers=10, rangeOutliers=c(1,5))
grnd$clusters[grnd$clusters==0] <- max(grnd$clusters)+1
# To colour outliers
plotInterval(grnd$data, pairsofsVar=NULL, cl=grnd$clusters,
clColors=TRUE)

# Example 3
library(clusterSim)
grnd <- cluster.Gen(50, model=4, dataType="s", numNoisyVar=2,
numOutliers=10, rangeOutliers=c(1,4))
grnd$clusters[grnd$clusters==0] <- max(grnd$clusters)+1
# To colour outliers
plotInterval(grnd$data, pairsofsVar=NULL, cl=grnd$clusters,
clColors=c("red","blue","green","yellow"))
```

`replication.Mod` *Modification of replication analysis for cluster validation*

Description

Modification of replication analysis for cluster validation

Usage

```
replication.Mod(x, v="m", u=2, centrotypes="centroids",
normalization=NULL, distance=NULL, method="kmeans",
S=10, fixedAsample=NULL)
```

Arguments

x	data matrix
v	type of data: metric ("r" - ratio, "i" - interval, "m" - mixed), nonmetric ("o" - ordinal, "n" - multi-state nominal, "b" - binary)
u	number of clusters given arbitrary

`centrotypes`	"centroids" or "medoids"
`normalization`	optional, normalization formulas for metric data (normalization by variable): for ratio data: "n0" - without normalization, "n6" - (x/sd), "n6a" - (x/mad), "n7" - (x/range), "n8" - (x/max), "n9" - (x/mean), "n9a" - (x/median), "n10" - (x/sum), "n11" - x/sqrt(SSQ) for interval or mixed data: "n0" - without normalization, "n1" - (x-mean)/sd, "n2" - (x-median)/mad, "n3" - (x-mean)/range, "n3a" - positional unitization (x-median)/range, "n4" - (x-min)/range, "n5" - (x-mean)/max[abs(x-mean)], "n5a" - (x-median)/max[abs(x-median)], "n12" - normalization (x - mean)/(sum(x - mean)^2)^0.5, "n12a" - positional normalization (x - median)/(sum(x - median)^2)^0.5, "n13" - normalization with zero being the central point ((x-midrange)/(range/2))
`distance`	distance measures NULL for "kmeans" method (based on data matrix), for ratio data: "d1" - Manhattan, "d2" - Euclidean, "d3" - Chebychev (max), "d4" - squared Euclidean, "d5" - GDM1, "d6" - Canberra, "d7" - Bray-Curtis for interval or mixed (ratio & interval) data: "d1", "d2", "d3", "d4", "d5" for ordinal data: "d8" - GDM2 for multi-state nominal: "d9" - Sokal & Michener for binary data: "b1" = Jaccard; "b2" = Sokal & Michener; "b3" = Sokal & Sneath (1); "b4" = Rogers & Tanimoto; "b5" = Czekanowski; "b6" = Gower & Legendre (1); "b7" = Ochiai; "b8" = Sokal & Sneath (2); "b9" = Phi of Pearson; "b10" = Gower & Legendre (2)
`method`	clustering method: "kmeans" (default), "single", "complete", "average", "mcquitty", "median", "centroid", "ward.D", "ward.D2", "pam", "diana"
`S`	the number of simulations used to compute mean corrected Rand index
`fixedAsample`	if NULL *A* sample is generated randomly, otherwise this parameter contains object numbers arbitrarily assigned to *A* sample

Details

See file `../doc/replication.Mod_details.pdf` for further details

Value

`A`	3-dimensional array containing data matrices for A sample of objects in each simulation (first dimension represents simulation number, second - object number, third - variable number)
`B`	3-dimensional array containing data matrices for B sample of objects in each simulation (first dimension represents simulation number, second - object number, third - variable number)
`centroid`	3-dimensional array containing centroids of u clusters for A sample of objects in each simulation (first dimension represents simulation number, second - cluster number, third - variable number)
`medoid`	3-dimensional array containing matrices of observations on u representative objects (medoids) for A sample of objects in each simulation (first dimension represents simulation number, second - cluster number, third - variable number)

`clusteringA`	2-dimensional array containing cluster numbers for A sample of objects in each simulation (first dimension represents simulation number, second - object number)
`clusteringB`	2-dimensional array containing cluster numbers for B sample of objects in each simulation (first dimension represents simulation number, second - object number)
`clusteringBB`	2-dimensional array containing cluster numbers for B sample of objects in each simulation according to 4 step of replication analysis procedure (first dimension represents simulation number, second - object number)
`cRand`	value of mean corrected Rand index for S simulations

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

References

Breckenridge, J.N. (2000), *Validating cluster analysis: consistent replication and symmetry*, "Multivariate Behavioral Research", 35 (2), 261-285. Available at: doi:10.1207/S15327906MBR3502_5.

Gordon, A.D. (1999), *Classification*, Chapman and Hall/CRC, London. ISBN 9781584880134.

Hubert, L., Arabie, P. (1985), *Comparing partitions*, "Journal of Classification", no. 1, 193-218. Available at: doi:10.1007/BF01908075.

Milligan, G.W. (1996), *Clustering validation: results and implications for applied analyses*, In P. Arabie, L.J. Hubert, G. de Soete (Eds.), *Clustering and classification*, World Scientific, Singapore, 341-375. ISBN 9789810212872.

Walesiak, M. (2008), *Ocena stabilnosci wynikow klasyfikacji z wykorzystaniem analizy replikacji*, In: J. Pociecha (Ed.), *Modelowanie i prognozowanie zjawisk spoleczno-gospodarczych*, Wydawnictwo AE, Krakow, 67-72.

See Also

`cluster.Sim`, `hclust`, `kmeans`, `dist`, `dist.BC`, `dist.SM`, `dist.GDM`,

`data.Normalization`

Examples

```
library(clusterSim)
data(data_ratio)
w <- replication.Mod(data_ratio, u=5, S=10)
print(w)

library(clusterSim)
data(data_binary)
replication.Mod(data_binary,"b", u=2, "medoids", NULL,"b1", "pam", fixedAsample=c(1,3,6,7))
```

`shapes.blocks3d` *Generation of data set containing two clusters with untypical shapes (cube divided into two parts by main diagonal plane)*

Description

Generation of data set containing two clusters with untypical shapes (cube starting at point (0,0,0) divided into two parts by main diagonal plane)

Usage

```
shapes.blocks3d(numObjects=180,shapesUnitSize=0.5, shape2coordinateX=1.2,
shape2coordinateY=1.2,shape2coordinateZ=1.2, outputCsv="", outputCsv2="",
outputColNames=TRUE, outputRowNames=TRUE)
```

Arguments

`numObjects`	number of objects in each cluster - positive integer value or vector with length=2
`shapesUnitSize`	length of one unit for shape (maximal heigth, width and depth of shape is 2*shapesUnitSize)
`shape2coordinateX`	maximal value for second shape in first (`X`) dimension
`shape2coordinateY`	maximal value for second shape in second (`Y`) dimension
`shape2coordinateZ`	maximal value for second shape in third (`Z`) dimension
`outputCsv`	optional, name of csv file with generated data (first column contains id, second - number of cluster and others - data)
`outputCsv2`	optional, name of csv (a comma as decimal point and a semicolon as field separator) file with generated data (first column contains id, second - number of cluster and others - data)
`outputColNames`	`outputColNames=TRUE` indicates that output file (given by `outputCsv` and `outputCsv2` parameters) contains first row with column names
`outputRowNames`	`outputRowNames=TRUE` indicates that output file (given by `outputCsv` and `outputCsv2` parameters) contains a vector of row names

Value

`clusters`	cluster number for each object
`data`	generated data - matrix with objects in rows and variables in columns

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

See Also

`shapes.worms`,`shapes.circles2`,`shapes.circles3`,`shapes.bulls.eye`,`shapes.two.moon`

Examples

```
library(clusterSim)
#library(rgl)
sb3d<-shapes.blocks3d(300,1,3,3,3)
#plot3d(sb3d$data,col=rainbow(2)[sb3d$clusters])
```

`shapes.circles2` *Generation of data set containing two clusters with untypical ring shapes (circles)*

Description

Generation of data set containing two clusters with untypical ring shapes. For each point first random radius `r` from given interval is generated then random angle `alpha` and finally the coordinates of point are calculated as (`r*cos(alpha)`,`r*sin(alpha)`). For bull's eye data set second shape is filled circle (`r` starts from 0)

Usage

```
shapes.circles2(numObjects=180, shape1rFrom=0.75,shape1rTo=0.9,shape2rFrom=0.35,
shape2rTo=0.5,outputCsv="", outputCsv2="", outputColNames=TRUE, outputRowNames=TRUE)
shapes.bulls.eye(numObjects=180, shape1rFrom=0.75,shape1rTo=0.95,shape2rTo=0.45,
outputCsv="", outputCsv2="", outputColNames=TRUE, outputRowNames=TRUE)
```

Arguments

`numObjects`	number of objects in each cluster - positive integer value or vector with length=2
,	
`shape1rFrom`	minimal value of radius for first ring
`shape1rTo`	maximal value of radius for first ring
`shape2rFrom`	minimal value of radius for second ring
`shape2rTo`	maximal value of radius for second ring
`outputCsv`	optional, name of csv file with generated data (first column contains id, second - number of cluster and others - data)
`outputCsv2`	optional, name of csv (a comma as decimal point and a semicolon as field separator) file with generated data (first column contains id, second - number of cluster and others - data)
`outputColNames`	`outputColNames=TRUE` indicates that output file (given by `outputCsv` and `outputCsv2` parameters) contains first row with column names
`outputRowNames`	`outputRowNames=TRUE` indicates that output file (given by `outputCsv` and `outputCsv2` parameters) contains a vector of row names

Value

`clusters`	cluster number for each object
`data`	generated data - matrix with objects in rows and variables in columns

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

See Also

`shapes.worms`,`shapes.circles3`,`shapes.bulls.eye`,`shapes.two.moon`,`shapes.blocks3d`

Examples

```
#Example1
library(clusterSim)
sc2<-shapes.circles2(180)
plot(sc2$data,col=rainbow(2)[sc2$clusters])

#Example2
library(clusterSim)
sbe<-shapes.bulls.eye(numObjects=c(120,60))
plot(sbe$data,col=rainbow(2)[sbe$clusters])
```

`shapes.circles3` *Generation of data set containing three clusters with untypical ring shapes (circles)*

Description

Generation of data set containing three clusters with untypical ring shapes. For each point first random radius `r` from given interval is generated then random angle `alpha` and finally the coordinates of point are calculated as (`r*cos(alpha)`,`r*sin(alpha)`)

Usage

```
shapes.circles3(numObjects=180,shape1rFrom=0.15,shape1rTo=0.3,
shape2rFrom=0.55,shape2rTo=0.7,shape3rFrom=1.15,shape3rTo=1.3,
outputCsv="", outputCsv2="", outputColNames=TRUE, outputRowNames=TRUE)
```

Arguments

`numObjects`	number of objects in each cluster - positive integer value or vector with length=3
,	
`shape1rFrom`	minimal value of radius for first ring

`shape1rTo`	maximal value of radius for first ring
`shape2rFrom`	minimal value of radius for second ring
`shape2rTo`	maximal value of radius for second ring
`shape3rFrom`	minimal value of radius for third ring
`shape3rTo`	maximal value of radius for third ring
`outputCsv`	optional, name of csv file with generated data (first column contains id, second - number of cluster and others - data)
`outputCsv2`	optional, name of csv (a comma as decimal point and a semicolon as field separator) file with generated data (first column contains id, second - number of cluster and others - data)
`outputColNames`	`outputColNames=TRUE` indicates that output file (given by `outputCsv` and `outputCsv2` parameters) contains first row with column names
`outputRowNames`	`outputRowNames=TRUE` indicates that output file (given by `outputCsv` and `outputCsv2` parameters) contains a vector of row names

Value

`clusters`	cluster number for each object
`data`	generated data - matrix with objects in rows and variables in columns

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

See Also

`shapes.worms`,`shapes.circles2`,`shapes.bulls.eye`,`shapes.two.moon`,`shapes.blocks3d`

Examples

```
#Example1
library(clusterSim)
sc3a<-shapes.circles3(180)
plot(sc3a$data,col=rainbow(3)[sc3a$clusters])

#Example2
library(clusterSim)
sc3b<-shapes.circles3(numObjects=c(120,180,240))
plot(sc3b$data,col=rainbow(3)[sc3b$clusters])
```

`shapes.two.moon` *Generation of data set containing two clusters with untypical shapes (similar to waxing and waning crescent moon)*

Description

Generation of data set containing two clusters with untypical shapes (similar to waxing and waning crescent moon). For each point first random radius `r` from given interval is generated then random angle `alpha` and finally the coordinates of point are calculated as (`a+abs(r*cos(alpha))`,`r*sin(alpha)`) for first shape and (`-abs(r*cos(alpha))`,`r*sin(alpha)-b` for second shape

Usage

```
shapes.two.moon(numObjects=180,shape1a=-0.4,shape2b=1,shape1rFrom=0.8,
shape1rTo=1.2,shape2rFrom=0.8, shape2rTo=1.2, outputCsv="", outputCsv2="",
outputColNames=TRUE, outputRowNames=TRUE)
```

Arguments

`numObjects`	number of objects in each cluster - positive integer value or vector with length=2
,	
`shape1a`	parameter `a` for first shape
`shape2b`	parameter `b` for first shape
`shape1rFrom`	minimal value of radius for first shape
`shape1rTo`	maximal value of radius for first shape
`shape2rFrom`	minimal value of radius for second shape
`shape2rTo`	maximal value of radius for second shape
`outputCsv`	optional, name of csv file with generated data (first column contains id, second - number of cluster and others - data)
`outputCsv2`	optional, name of csv (a comma as decimal point and a semicolon as field separator) file with generated data (first column contains id, second - number of cluster and others - data)
`outputColNames`	`outputColNames=TRUE` indicates that output file (given by `outputCsv` and `outputCsv2` parameters) contains first row with column names
`outputRowNames`	`outputRowNames=TRUE` indicates that output file (given by `outputCsv` and `outputCsv2` parameters) contains a vector of row names

Value

`clusters`	cluster number for each object
`data`	generated data - matrix with objects in rows and variables in columns

Author(s)

Marek Walesiak <marek.walesiak@ue.wroc.pl>, Andrzej Dudek <andrzej.dudek@ue.wroc.pl>

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

See Also

shapes.worms,shapes.circles2,shapes.circles3,shapes.bulls.eye,shapes.blocks3d

Examples

```
library(clusterSim)
stm<-shapes.two.moon(180)
plot(stm$data,col=rainbow(2)[stm$clusters])
```

---

shapes.worms *Generation of data set containing two clusters with untypical parabolic shapes (worms)*

Description

Generation of data set containing two clusters with untypical parabolic shapes (first is given by y=x^2, second by y=-(x-a)^2+b with distortion from <-tol,+tol>)

Usage

```
shapes.worms(numObjects=180,shape1x1=-2,shape1x2=2,shape2x1=-0.5,
shape2x2=2.5,shape2a=1.5,shape2b=5.5,tol=0.1,outputCsv="", outputCsv2="",
outputColNames=TRUE, outputRowNames=TRUE)
```

Arguments

`numObjects`	number of objects in each cluster - positive integer value or vector with length=2
`shape1x1`	starting value on abscissa axis for shape 1
`shape1x2`	end value on abscissa axis for shape 1
`shape2x1`	starting value on abscissa axis for shape 2
`shape2x2`	end value on abscissa axis for shape 2
`shape2a`	parameter a of shape 2
`shape2b`	parameter b of shape 2
`tol`	tolerance - each generated point is randomized by adding runif(1,0,tol)
`outputCsv`	optional, name of csv file with generated data (first column contains id, second - number of cluster and others - data)
`outputCsv2`	optional, name of csv (a comma as decimal point and a semicolon as field separator) file with generated data (first column contains id, second - number of cluster and others - data)

`outputColNames`	`outputColNames=TRUE` indicates that output file (given by `outputCsv` and `outputCsv2` parameters) contains first row with column names
`outputRowNames`	`outputRowNames=TRUE` indicates that output file (given by `outputCsv` and `outputCsv2` parameters) contains a vector of row names

Value

`clusters`	cluster number for each object
`data`	generated data - matrix with objects in rows and variables in columns

Author(s)

Marek Walesiak `<marek.walesiak@ue.wroc.pl>`, Andrzej Dudek `<andrzej.dudek@ue.wroc.pl>`

Department of Econometrics and Computer Science, University of Economics, Wroclaw, Poland

See Also

`shapes.worms`,`shapes.circles2`,`shapes.circles3`,`shapes.bulls.eye`,`shapes.two.moon`,`shapes.blocks3d`

Examples

```
library(clusterSim)
sw<-shapes.worms(180)
plot(sw$data,col=rainbow(2)[sw$clusters])
```

`speccl` *A spectral clustering algorithm*

Description

A spectral clustering algorithm. Cluster analysis is performed by embedding the data into the subspace of the eigenvectors of an affinity matrix

Usage

```
speccl(data,nc,distance="GDM1",sigma="automatic",sigma.interval="default",
mod.sample=0.75,R=10,iterations=3,na.action=na.omit,...)
```

Arguments

`data`	matrix or dataset
`nc`	the number of clusters
`distance`	distance function used to calculate affinity matrix: "sEuclidean" - squared Euclidean distance, "euclidean" - Euclidean distance, "manhattan" - city block distance, "maximum" - Chebyshev distance, "canberra" - Lance and Williams Canberra distance, "BC" - Bray-Curtis distance measure for ratio data, "GDM1" - GDM distance for metric data, "GDM2" - GDM distance for ordinal data, "SM" - Sokal-Michener distance measure for nominal variables

`sigma`	scale parameter used to calculate affinity matrix: sigma="automatic" - an algorithm for searching optimal value of sigma parameter; sigma=200 - value of sigma parameter given by researcher, e.g. 200
`sigma.interval`	sigma.interval="default" - from zero to square root of sum of all distances in lower triangle of distance matrix for "sEuclidean" and from zero to sum of all distances in lower triangle of distance matrix for other distances; sigma.interval=1000 - from zero to value given by researcher, e.g. 1000
`mod.sample`	proportion of data to use when estimating sigma (default: 0.75)
`R`	the number of intervals examined in each step of searching optimal value of sigma parameter algorithm (See `../doc/speccl_details.pdf`)
`iterations`	the maximum number of iterations (rounds) allowed in algorithm of searching optimal value of sigma parameter
`na.action`	the action to perform on NA
`...`	arguments passed to kmeans procedure

Details

See file `../doc/speccl_details.pdf` for further details

Value

`scdist`	returns the lower triangle of the distance matrix
`clusters`	a vector of integers indicating the cluster to which each object is allocated
`size`	the number of objects in each cluster
`withinss`	the within-cluster sum of squared distances for each cluster
`Ematrix`	data matrix n x u (n - the number of objects, u - the number of eigenvectors)
`Ymatrix`	normalized data matrix n x u (n - the number of objects, u - the number of eigenvectors)
`sigma`	the value of scale parameter given by searching algorithm

Author(s)

Marek Walesiak <`marek.walesiak@ue.wroc.pl`>, Andrzej Dudek <`andrzej.dudek@ue.wroc.pl`>

Department of Econometrics and Computer Science, Wroclaw University of Economics, Poland

References

Karatzoglou, A. (2006), *Kernel methods. Software, algorithms and applications*, Dissertation, Wien, Technical University.

Ng, A., Jordan, M., Weiss, Y. (2002), *On spectral clustering: analysis and an algorithm*, In: T. Dietterich, S. Becker, Z. Ghahramani (Eds.), Advances in Neural Information Processing Systems 14. MIT Press, 849-856. Available at:

`https://papers.nips.cc/paper/2092-on-spectral-clustering-analysis-and-an-algorithm.pdf`.

Walesiak, M. (2011), *Uogólniona miara odległości GDM w statystycznej analizie wielowymiarowej z wykorzystaniem programu R [The Generalized Distance Measure GDM in multivariate statistical analysis with R]*, Wydawnictwo Uniwersytetu Ekonomicznego, Wroclaw.

Walesiak, M. (2012), *Klasyfikacja spektralna a skale pomiaru zmiennych [Spectral clustering and measurement scales of variables]*, Przeglad Statystyczny (Statistical Review), no. 1, 13-31. Spectral Clustering and Measurement Scales of Variables Marek Walesiak Przegląd Statystyczny. Statistical Review, vol. 59, 2012, 1, pages: 13-31. Available at: doi:10.59139/ps.2012.01.2.

Walesiak, M. (2016), *Uogólniona miara odległości GDM w statystycznej analizie wielowymiarowej z wykorzystaniem programu R. Wydanie 2 poprawione i rozszerzone [The Generalized Distance Measure GDM in multivariate statistical analysis with R]*, Wydawnictwo Uniwersytetu Ekonomicznego, Wroclaw.

See Also

`dist.GDM`,`kmeans`,`dist`,`dist.binary`,`dist.SM`,`dist.BC`

Examples

```
# Commented due to long execution time
# Example 1
#library(clusterSim)
#library(mlbench)
#data<-mlbench.spirals(100,1,0.03)
#plot(data)
#x<-data$x
#res1<-speccl(x,nc=2,distance="GDM1",sigma="automatic",
#sigma.interval="default",mod.sample=0.75,R=10,iterations=3)
#clas1<-res1$cluster
#print(data$classes)
#print(clas1)
#cRand<-classAgreement(table(as.numeric(as.vector(data$classes)),
#res1$clusters))$crand
#print(res1$sigma)
#print(cRand)

# Example 2
#library(clusterSim)
#grnd2<-cluster.Gen(50,model=4,dataType="m",numNoisyVar=1)
#data<-as.matrix(grnd2$data)
#colornames<-c("red","blue","green")
#grnd2$clusters[grnd2$clusters==0]<-length(colornames)
#plot(grnd2$data,col=colornames[grnd2$clusters])
#us<-nrow(data)*nrow(data)/2
#res2<-speccl(data,nc=3,distance="sEuclidean",sigma="automatic",
#sigma.interval=us,mod.sample=0.75,R=10,iterations=3)
#cRand<-comparing.Partitions(grnd2$clusters,res2$clusters,type="crand")
#print(res2$sigma)
#print(cRand)

# Example 3
#library(clusterSim)
```

```
#grnd3<-cluster.Gen(40,model=4,dataType="o",numCategories=7)
#data<-as.matrix(grnd3$data)
#plotCategorial(grnd3$data,pairsofVar=NULL,cl=grnd3$clusters,
#clColors=c("red","blue","green"))
#res3<-speccl(data,nc=3,distance="GDM2",sigma="automatic",
#sigma.interval="default",mod.sample=0.75,R=10,iterations=3)
#cRand<-comparing.Partitions(grnd3$clusters,res3$clusters,type="crand")
#print(res3$sigma)
#print(cRand)

# Example 4
library(clusterSim)
data(data_nominal)
res4<-speccl(data_nominal,nc=4,distance="SM",sigma="automatic",
sigma.interval="default",mod.sample=0.75,R=10,iterations=3)
print(res4)
```

Index