Package 'clusterv'

July 22, 2025

Type Package

Version 1.1.1

Title Assessment of Cluster Stability by Randomized Maps

Maintainer Jessica Gliozzo <jessica.gliozzo@unimi.it>

Description The reliability of clusters is estimated using random projections.
A set of stability measures is provided to assess the reliability of the clusters
discovered by a generic clustering algorithm.
The stability measures are taylored to high dimensional data (e.g. DNA microarray data)
(Valentini, G (2005), <doi:10.1093/bioinformatics/bti817>.

License GPL (>= 2)

URL https://valentini.di.unimi.it/SW/clusterv/

BugReports https://github.com/AnacletoLAB/clusterv/issues

Encoding UTF-8

Imports stats, MASS, cluster

Suggests R.rsp

VignetteBuilder R.rsp

NeedsCompilation no

Author Giorgio Valentini [aut],
Jessica Gliozzo [cre]

Repository CRAN

Date/Publication 2025-05-14 08:30:02 UTC

Contents

`AC.index` *Assignment Confidence (AC) index*

Description

Assignment confidence index computation. For a given clustering and similarity matrix, the set of AC indices are computed (for each cluster and each example) It assumes that the label of the examples are integers.

Usage

```
AC.index(cluster, c, Sim.M)
```

Arguments

cluster	list of the clusters whose validity indices will be computed
c	number of clusters
Sim.M	similarity matrix

Details

The *Assignment-Confidence (AC)* index estimates the confidence of the assignment of an example i to a cluster A using a similarity matrix M:

$$AC(i, A) = \frac{1}{|A| - 1} \sum_{j \in A, j \neq i} M_{ij}$$

Using a set of realizations of a given randomized projection, the *AC*-index represents the frequency by which i appears with the other elements of the cluster A.

Value

matrix with the Assignment Confidence index for each example. Each row corresponds to an example, each column to a cluster.

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

See Also

Validity.indices, Cluster.validity, Cluster.validity.from.similarity,

Do.similarity.matrix.partition, Do.similarity.matrix

Examples

```
# Computation of the AC indices of a hierarchical clustering algorithm
M <- generate.sample0(n=10, m=2, sigma=2, dim=800)
d <- dist (t(M));
tree <- hclust(d, method = "average");
plot(tree, main="");
cl.orig <- rect.hclust(tree, k = 3);
l.norm <- Multiple.Random.hclustering (M, dim=100, pmethod="Norm",
                                       c=3, hmethod="average", n=20)
Sim <- Do.similarity.matrix.partition(l.norm);
ac <- AC.index(cl.orig, c=3, Sim)
```

`Achlioptas.hclustering` *Multiple Hierarchical clusterings using Achlioptas random projections*

Description

Multiple Hierarchical clusterings using Achlioptas random projections of the data.

Usage

```
Achlioptas.hclustering(M, dim, c = 3, hmethod = "average", n = 50,
scale = TRUE, seed = 100, distance="euclidean")

Achlioptas.hclustering.tree(M, dim, hmethod = "average", n = 50, scale = TRUE,
seed = 100, distance = "euclidean")
```

Arguments

`M`	matrix of data: rows are variables and columns are examples
`dim`	subspace dimension
`c`	number of clusters
`hmethod`	the agglomeration method to be used. This should be one of "ward.D", "single", "complete", "average", "mcquitty", "median" or "centroid", according to the `hclust` method of the package `stats`.
`n`	number of random projections
`scale`	if TRUE (default) Achlioptas random projections are scaled
`seed`	numerical seed for the random generator
`distance`	it must be one of the two: "euclidean" (default) or "pearson" (that is 1 - Pearson correlation)

Value

a list with components "cluster" and "tree":

`cluster`	list of the n clusterings obtained. Each element is in turn a list of vectors that correspond to the clusters of the clustering. Each cluster is represented by a vector of integers whose values corresponds to the indices of the columns (examples) of the original data.
`tree`	list of the trees generated by the multiple clusterings

`Achlioptas.hclustering.tree` returns only the list of the trees.

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

References

D.Achlioptas, Database-friendly random projections., in: Proc. ACM Symp. on the Principles of Database Systems, Contemporary Mathematics, 2001, pp. 274-281.

See Also

Achlioptas.random.projection, Plus.Minus.One.random.projection, norm.random.projection,random.subspace

Examples

```
# 20 hierarchical clusterings on multiple Achlioptas projected data with
# subspace dimension equal to 100
M <- generate.sample0(n=10, m=2, sigma=1, dim=800)
l <- Achlioptas.hclustering(M, dim=100, hmethod = "average", n = 20, scale = TRUE)
# Equal as above, but only the trees are generated
l <- Achlioptas.hclustering.tree(M, dim=100, hmethod = "average", n = 20, scale = TRUE)
# 10 hierarchical clusterings on multiple Achlioptas projected data with
# subspace dimension equal to 200
M <- generate.sample0(n=8, m=1, sigma=2, dim=1000)
l <- Achlioptas.hclustering(M, dim=200, hmethod = "average", n = 10, scale = TRUE)
```

Achlioptas.random.projection

Achlioptas random projection

Description

Random projections to a lower dimension subspace with the Achlioptas' projection matrix. The projection is performed using a projection matrix R s.t. Prob(R[i,j]=sqrt(3))=Prob(R[i,j]=-sqrt(3)=1/6; Prob(R[i,j]=0)=2/3

Usage

```
Achlioptas.random.projection(d = 2, m, scaling = TRUE)
```

Arguments

d	subspace dimension
m	data matrix (rows are features and columns are examples)
scaling	if TRUE (default) scaling is performed

Details

Achlioptas random projections are represented by $d' \times d$ matrices $P = 1/\sqrt{d'}(r_{ij})$, where r_{ij} are chosen in $\{-\sqrt{3}, 0, \sqrt{3}\}$, such that $Prob(r_{ij} = 0) = 2/3$, $Prob(r_{ij} = \sqrt{3}) = Prob(r_{ij} = -\sqrt{3}) = 1/6$. In this case also we have $E[r_{ij}] = 0$ and $Var[r_{ij}] = 1$ and the Johnson-Lindenstrauss lemma holds.

Value

data matrix (dimension d x ncol(m)) of the examples projected in a d-dimensional random subspace

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

References

D.Achlioptas, Database-friendly random projections., in: Proc. ACM Symp. on the Principles of Database Systems, Contemporary Mathematics, 2001, pp. 274-281.

W.Johnson, J.Lindenstrauss, Extensions of Lipshitz mapping into Hilbert space, in: Conference in modern analysis and probability, Vol.~26 of Contemporary Mathematics, Amer. Math. Soc., 1984, pp. 189–206.

See Also

Plus.Minus.One.random.projection, norm.random.projection, random.subspace

Examples

```
# Achlioptas random projection from a 1000 dimensional space to a 50-dimensional subspace
m <- matrix(runif(10000), nrow=1000)
m.p <- Achlioptas.random.projection(d = 50, m, scaling = TRUE)
# Achlioptas random projection from a 10000 dimensional space to a 1000-dimensional subspace
m <- matrix(rnorm(500000), nrow=5000)
m.p <- Achlioptas.random.projection(d = 1000, m, scaling = TRUE)
# The same as above without scaling
m <- matrix(rnorm(500000), nrow=5000)
m.p <- Achlioptas.random.projection(d = 1000, m, scaling = FALSE)
```

Cluster.validity *Validity indices computation*

Description

It computes the stability indices for each individual cluster, the overall validity index of the clustering and (optionally) the Assignment Confidence (AC) index for each example. To compute the indices a set of clusterings is used. It assumes that the label of the examples are integers.

Usage

```
Cluster.validity(cluster, M.clusters, AC = FALSE)

Cluster.validity.from.similarity(cluster, Sim.M, AC = TRUE)
```

Arguments

`cluster`	list of the clustering whose validity indices will be computed
`M.clusters`	list of the n clusterings (a list of lists) used for validity index computation
`Sim.M`	similarity matrix
`AC`	if it is TRUE the Assignment Confidence index for each example is computed

Details

Using the similarity matrix M, the *stability index* s for a cluster A is:

$$s(A) = \frac{1}{|A|(|A|-1)} \sum_{(i,j)\in A\times A, i\neq j} M_{ij}$$

The index $s(A)$ estimates the stability of a cluster A by measuring how much the projections of the pairs $(i,j) \in A \times A$ occur together in the same cluster in the projected subspaces. The stability index has values between 0 and 1: low values indicate no reliable clusters, high values denote stable clusters.

The *overall validity* of the clustering is the average between the validity indices of the individual clusters.

The *Assignment-Confidence (AC)* index estimates the confidence of the assignment of an example i to a cluster A using a similarity matrix M:

$$AC(i,A) = \frac{1}{|A|-1} \sum_{j\in A, j\neq i} M_{ij}$$

Using a set of realizations of a given randomized projection, the *AC*-index represents the frequency by which i appears with the other elements of the cluster A.

Value

a list with four components: "validity", "overall.validity", "similarity.matrix", "AC" (optional):

`validity`	vector with the validity of each of the clusters
`overall.validity`	validity index of the overall cluster
`similarity.matrix`	pairwise similarity matrix between examples
`AC`	matrix with the Assignment Confidence index for each example. Each row corresponds to an example, each column to a cluster

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

See Also

`Validity.indices` `AC.index`, `Do.similarity.matrix`

Examples

```
# Computation of the validity indices for a hierarchical clustering
M <- generate.sample0(n=10, m=1, sigma=1, dim=1000)
d <- dist (t(M));
tree <- hclust(d, method = "average");
plot(tree, main="");
cl.orig <- rect.hclust(tree, k = 3);
l.PMO <- Multiple.Random.hclustering (M, dim=100, pmethod="PMO",
                                      c=3, hmethod="average", n=20)
list.indices <- Cluster.validity(cl.orig, l.PMO, AC = TRUE)
# Computation of the validity indices for a hierarchical clustering
# with less defined clusters
M.less <- generate.sample0(n=10, m=1, sigma=2, dim=1000)
d <- dist (t(M.less));
tree.less <- hclust(d, method = "average");
plot(tree.less, main="");
cl.orig.less <- rect.hclust(tree.less, k = 3);
l.PMO.less <- Multiple.Random.hclustering (M.less, dim=100, pmethod="PMO",
                                           c=3, hmethod="average", n=20)
list.indices.less <- Cluster.validity(cl.orig.less, l.PMO.less, AC = TRUE)
```

Do.similarity.matrix *Functions to compute a pairwise similarity matrix.*

Description

The elements of a similarity matrix represent the frequency by which each pair of examples belongs to the same cluster across multiple clusterings. These functions may also be used with clusterings with a variable number of clusters.

Usage

```
Do.similarity.matrix(l, dim.Sim.M)

Do.similarity.matrix.partition(l)
```

Arguments

`l`	list of clusterings. Each element is a list of clusters. Each cluster is a vector whose elements (integers) represent the examples
`dim.Sim.M`	dimension of the similarity matrix (number of examples)

Details

A $n \times n$ similarity matrix M to a k-clustering; the elements M_{ij} of M are defined as:

$$M_{ij} = \sum_{s=1}^{k} \chi_{A_s}[i] \cdot \chi_{A_s}[j]$$

where $i, j \in \{1, 2, \ldots, n\}$, and $\chi_{A_s} \in \{0, 1\}^n$ is the characteristic vector of $A_s \subseteq \{1, 2, \ldots, n\}$, i.e. $\chi_{A_s}[i] = 1$ if $i \in A_s$, otherwise $\chi_{A_s}[i] = 0$. If the k-clustering identifies a partition, $M_{ij} \in \{0, 1\}$: in other words, M_{ij} denotes if elements i and j belong to the same cluster. Consider also a random projection $\mu : \mathcal{R}^d \to \mathcal{R}^{d'}$. Then a similarity matrix M can be computed averaging among multiple clusterings obtained from multiple random projections. This similarity matrix represents how much pairs of projected examples belong to the same cluster averaging across the repeated random projections. `Do.similarity.matrix` can be used with clusterings that do not strictly define a partition (that is a specific example may belong to more than 1 cluster). `Do.similarity.matrix.partition` may be used only with clusterings that strictly define a partition.

Value

A pairwise similarity matrix whose elements represents how much 2 examples fall in the same cluster across multiple clusterings. Each element of the matrix is normalized so that its value is beween 0 and 1.

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

Examples

```
# Computing the similarity matrix associated to 20 hierarchical clusterings
# using Normal projections.
M <- generate.sample0(n=10, m=2, sigma=2, dim=800)
l.norm <- Multiple.Random.hclustering (M, dim=100, pmethod="Norm", c=3,
                                       hmethod="average", n=20)
Sim <- Do.similarity.matrix.partition(l.norm);
# The same as above, but with 30 hierarchical clusterings using PMO projections.
l.PMO <- Multiple.Random.hclustering (M, dim=100, pmethod="PMO", c=3,
                                      hmethod="average", n=30)
Sim.PMO <- Do.similarity.matrix.partition(l.norm);
```

Generate.clusters *Multiple clusterings generation from the corresponding trees*

Description

Multiple clusterings generation from the corresponding trees for a given cut (number of clusters).

Usage

```
Generate.clusters(tr, c = 3)
```

Arguments

`tr`	a list of trees as returned by the hclust algorithm
`c`	number of clusters

Value

A list of lists. Each list represents a clustering. Each cluster is a list of vectors, whose elements are the labels of the examples.

See Also

Achlioptas.hclustering.tree, PMO.hclustering.tree,

Norm.hclustering.tree,RS.hclustering.tree

Examples

```
# list of clusterings generated using Achlioptas random projections,
# using cuts corresponding to 3, 4 and 10 clusters
M <- generate.sample0(n=10, m=2, sigma=1, dim=800)
list.trees <- Achlioptas.hclustering.tree(M, dim=100, hmethod = "average",
                                          n = 20, scale = TRUE)
list.clusters3 <- Generate.clusters(list.trees, c = 3)
list.clusters4 <- Generate.clusters(list.trees, c = 4)
list.clusters10 <- Generate.clusters(list.trees, c = 10)
```

---

generate.sample.h1 *Two-levels hierarchical cluster generator.*

Description

A 2-dimensional two-level hierarchical cluster structure is generated. At a first level 3 distinct clusters at the vertices of an equilater triangle are generated. At a second level two other clusters at the left and right of the three "primary" clusters are generated.

Usage

```
generate.sample.h1(n = 20, l = 5, Delta.h = 1, sd = 0.1, with.I.level.examples = FALSE)
```

Arguments

n	number of examples for each cluster
l	half length of the edge of the equilater triangle
Delta.h	half of the "abscissa" distance between each pair of clusters inside the three major clusters
sd	standard deviation
with.I.level.examples	if TRUE data centered at the vertices of the triangle are generated, otherwise only the secondary clusters are generated.

Value

a matrix with dim rows (variables) and n*6 columns (examples)

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

Examples

```
generate.sample.h1()
# Generation of a data set with 120 2-dimensional examples
# data have a two-level hierarchical structure with respectively 3 and 6 clusters.
generate.sample.h1(n = 20, l = 5, Delta.h = 1, sd = 0.1, with.I.level.examples = TRUE)
```

---

generate.sample.h2 *Three-level hierarchical cluster generator.*

Description

A 2-dimensional three-level hierarchical cluster structure is generated. At a first level 3 distinct clusters at the vertices of an equilater triangle are generated. At a second level two other clusters at the left and right of the three "primary" clusters are generated (6 clusters) At a third level two other clusters above and below the secondaty clusters are generated (12 clusters)

Usage

```
generate.sample.h2(n = 20, l = 8, Delta.h = 2, Delta.v = 1, sd = 0.1,
with.I.II.level.examples = FALSE)
```

Arguments

n	number of examples for each cluster
l	half length of the edge of the equilater triangle
Delta.h	half of the "abscissa" distance between each pair of clusters inside the three major clusters
Delta.v	half of the "ordinate" distance between each pair of clusters inside the three second order clusters
sd	standard deviation
with.I.II.level.examples	if TRUE data at the first and secondary level are generated (for a total of 21 clusters), otherwise only the third level

Value

a matrix with dim rows (variables) and n*6 columns (examples)

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

Examples

```
generate.sample.h2()
# Generation of a data set with 240 2-dimensional examples
# data have a three-level hierarchical structure with respectively 3 and 6 and 12 clusters.
generate.sample.h2(n = 20, l = 10, Delta.h = 2, Delta.v = 1, sd = 0.05)
```

---

generate.sample.h3 *Two-levels hierarchical cluster generator.*

Description

A 2-dimensional 2-levels hierarchical cluster structure is generated. At a first level 4 distinct clusters are generated. At a second level two other clusters at the left and right of 2 of the 4 "primary" clusters are generated (6 clusters)

Usage

```
generate.sample.h3(n = 20, DeltaA = 1, DeltaB = 1, seed = 0)
```

Arguments

n	number of examples for each cluster
DeltaA	vertical displacement of the the secondary clusters
DeltaB	horizontal displacement of the the secondary clusters
seed	seed for the random generator

Value

a matrix with dim rows (variables) and n*6 columns (examples)

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

Examples

```
generate.sample.h3()
# Generation of a data set with 120 2-dimensional examples
# data have a two-level hierarchical structure with respectively 4 and 6 clusters.
generate.sample.h3(n = 20, DeltaA = 1, DeltaB = 1, seed = 0)
```

`generate.sample0` *Sample0 generator of synthetic data*

Description

Multivariate normally distributed data synthetic generator. Data sets with 3 clusters are randomly generated. *n* examples for each class are generated. All classes (each one of *n* examples) has *dim* components The first class (first *n* examples) has its components centered in 0 (of length *dim*) The second class (second *n* examples) has its components centered in *m* (of length *dim*) The third class (last *n* examples) has its components centered in *-m* (of length *dim*) For all classes the covariance matrix is diagonal with values *sigma*.

Usage

```
generate.sample0(n = 5, m = 10, sigma = 1, dim = 2)
```

Arguments

`n`	number of examples for each class
`m`	mean value for the second class
`sigma`	value of the diagonal elements of the covariance matrix
`dim`	dimension of the examples

Value

a matrix with *dim* rows (variables) and *n**3 columns (examples)

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

Examples

```
generate.sample0()
# Generation of a data set with 60 500-dimensional examples, with the examples
# of the first class  centered in the 500-dimensional 0 vector, the second class
# is centered in the 1 vector, the third in -1. The covariance matrix is the
# matrix with all 2 values on the diagonal elements
generate.sample0(n = 20, m = 1, sigma = 2, dim = 500)
```

`generate.sample1` *Sample1 generator of synthetic data*

Description

Multivariate normally distributed data synthetic generator. Data sets with 3 clusters are randomly generated. *n* examples for each class are generated. All classes (each one of n examples) have their last dim-500 variables centered in 0. The first class (first n examples) has its first 500 features centered in 0. The second class (second n examples) has its first 500 features centered in m. The third class (last n examples) has its first 500 features centered in -m. For all classes the covariance matrix is diagonal with all values on the diagonal equal to sigma.

Usage

```
generate.sample1(n = 2, m = 6, sigma = 1, dim = 10000)
```

Arguments

`n`	number of examples for each class
`m`	center of the first 500 variables of the second class
`sigma`	value of the diagonal elements of the covariance matrix
`dim`	number of variables (features)

Value

a matrix with dim rows (variables) and n*3 columns (examples)

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

Examples

```
generate.sample1()
# Generation of a data set with 30 1000-dimensional examples, with the examples
# of the first class  centered in 0 for the first 500 variables, the second class
# is centered in 1 for the first 500 variables, the third in -1.
# The covariance matrix is the matrix with all  values different from 0 (equal to 3)
# on the diagonal elements.
generate.sample1(n = 10, m = 1, sigma = 3, dim = 1000)
```

`generate.sample2` *Sample2 generator of synthetic data*

Description

Multivariate normally distributed data synthetic generator. Data sets with 2 clusters are randomly generated. *n* examples for each class are generated. n 10000-dimensional examples for each class are generated. All classes (each one of n examples) has only no-noisy features but there is substantial overlapping between classes The first class (first n examples) has its features centered in 1 (first 5000 features) and 2 (last 5000 features) The second class (second n examples) has its features centered in -1 (first 5000 features) and -2 (last 5000 features) The diagonal of the covariance matrix of the first class has its first 2500 element equal to 0.5, the next 2500 equal to 1, the next 2500 to 0.5 and the last to 1. The diagonal of the covariance matrix of the second class has its first 5000 element equal to 1, the next 5000 equal to 2

Usage

```
generate.sample2(n = 2)
```

Arguments

`n`	number of examples for each class

Value

a real data matrix with 10000 rows (variables) and n*2 columns (examples)

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

Examples

```
generate.sample2()
generate.sample2(n = 20)
```

`generate.sample3` *Sample3 generator of synthetic data*

Description

Multivariate normally distributed data synthetic generator. Data sets with 3 clusters are randomly generated. *n* examples for each class are generated. n 1000-dimensional examples for each class are generated. All classes (each one of n examples) has 300 no-noisy features and 700 noisy features. There is a certain overlap between classes and a full covariance matrix (equal for all classes is used). The first class (first n examples) has its no-noisy features centered in 0. The second class (second n examples) has its no-noisy features centered in m The third class (last n examples) has its no-noisy features centered in -m Covariance matrix Sigma = (B, Zero; Zero', I) where B is a 300X300 matrix s.t. B[i,i]=1, B[i,i+1]=B[i,i-1]=0.5 and B[i,j]=0.1 j!=i-1,i,i+1; Zero is a 300X700 zero matrix and Zero' its transpose; I is a 700X700 identity matrix.

Usage

```
generate.sample3(n = 2, m = 2)
```

Arguments

n	number of examples for each class
m	vector center of the second class

Value

a matrix with 1000 rows (variables) and n*3 columns (examples)

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

Examples

```
generate.sample3()
# Generation of a data set with 60 1000-dimensional examples,
# with the examples of the first class  centered in the 1000-dimensional
# 0 vector, the second class is centered in the 1 vector, the third in -1.
generate.sample3(n = 20, m = 1)
```

`generate.sample4` *Sample4 generator of synthetic data*

Description

Multivariate normally distributed data synthetic generator. Data sets with 5 clusters are randomly generated. *n* 6000-dimensional examples for each class are generated. All classes (each one of n examples) have 1000 no-noisy and 5000 noisy features but there is substantial overlapping between distributions underlying classes 1 and 2 and 1 and 3, while class 4 and 5 are separated. The first class (first n examples) has its no noisy variables centered in 0. The second class (second n examples)

has its no noisy variables centered in 1. The third class (third n examples) has its no noisy variables centered in -1. The fourth class (fourth n examples) has its no noisy variables centered in 5. The fifth class (fifth n examples) has its no noisy variables centered in -5. The diagonal of the covariance matrix for all classes has its elements equal to sigma (first 1000 variables) and equal to 2*sigma (last 5000 variables).

Usage

```
generate.sample4(n = 2, sigma = 1)
```

Arguments

n	number of examples for each class
sigma	standard deviation of the first 1000 variables. The remaining variables have 2*sigma standard deviation

Value

a real data matrix with 1000 rows (variables) and n*5 columns (examples)

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

Examples

```
generate.sample4()
# Generation of a data set with 100 6000-dimensional examples
generate.sample4(n = 20, sigma = 1)
```

---

generate.sample5 *Sample5 generator of synthetic data*

Description

Multivariate normally distributed data synthetic generator. Data sets with 4 clusters are randomly generated. *n* examples for each class are generated. All classes (each one with n examples) has (1-ratio.noisy)*dim of no-noisy features and ratio.noisy*dim of noisy features. For "noisy" feature we mean features that are equally distributed in all the classes (these variables are centered in 0), while for "no-noisy" we mean features that are centered in different points in the different classes. Note that if the number on no-noisy feature is less than 2 the generation is aborted. A full covariance matrix (equal for all classes) is used. The first class (first n examples) has its no-noisy features centered in 0. The second class (second n examples) has its no-noisy features centered in m The third class (third n examples) has its no-noisy features centered in -m A fourth cluster (third n examples) has its no-noisy features centered in (m,-m) alternatively Covariance matrix Sigma = (B, Zero; Zero', I) where B is a (dim*(1-ratio.noisy))X(dim*(1-ratio.noisy)) matrix s.t. B[i,i]=1, B[i,i+1]=B[i,i-1]=0.5 and B[i,j]=0.1 if j!=i-1,i,i+1; Zero is a (dim*(1-ratio.noisy))X(dim*ratio.noisy) zero matrix and Zero' its transpose; I is a (dim*ratio.noisy)X(dim*ratio.noisy) identity matrix

Usage

```
generate.sample5(n = 10, dim = 10, ratio.noisy = 0.8, m = 2)
```

Arguments

n	number of examples for each class
dim	dimension of the examples
ratio.noisy	ratio of the noisy variables. The number of "noisy" features is ratio.noisy * dim
m	center of the II cluster (the third has center -m)

Value

a matrix with dim rows (variables) and n*4 columns (examples)

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

Examples

```
generate.sample5()
# Generation of a data set with 80 1000-dimensional examples, with the 200 no-noisy
# features of the examples of the first class  centered in 0, the 200 no-noisy features
# of the examples of the second class  centered in 2, the 200 no-noisy features of
# the examples of the third class  centered in -2, and the 200 no-noisy features of the
# examples of the fourth class  centered in alternatively in (2,-2).
generate.sample5(n = 20, m = 2, ratio.noisy = 0.8, dim = 1000)
```

---

generate.sample6 — *Sample6 generator: multivariate normally distributed data synthetic generator*

Description

n examples for each from 6 classes are generated. All classes (each one of n examples) has dim components The clusters have a hierarchical structure: 2 or 6 clusters may be detected. Anyway note that the structure of the data depends on the parameters: two main clusters are centered in m and -m. Around each main cluster three other subclusters are generated using the displacement d.

Usage

```
generate.sample6(n = 20, m = 10, dim = 2, d = 3, s = 0.2)
```

Arguments

n	number of examples for each class
m	mean basic value
dim	amount of the displacement from m
d	dimension of the examples
s	value of the diagonal elements of the covariance matrix

Value

a matrix with dim rows (variables) and n*6 columns (examples)

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

Examples

```
generate.sample6()
# Generation of a data set with 120 200-dimensional examples
# data have a two-level hierarchical structure with respectively 2 and 6 clusters
generate.sample6(n = 20, m = 10, dim = 200, d = 3, s = 1)
```

---

generate.sample7 *Sample7 generator: multivariate normally distributed data synthetic generator*

Description

n examples for each from 6 classes are generated. All classes (each one of n examples) has dim components The clusters have a hierarchical structure: 2 or 6 clusters may be detected. Anyway note that the structure of the data depends on the parameters: two main clusters are centered in m and -m. Around each main cluster two other subclusters are generated using the displacement d.

Usage

```
generate.sample7(n = 20, m = 10, dim = 1000, d = 3, s = 1)
```

Arguments

n	number of examples for each class
m	mean basic value
dim	amount of the displacement from m
d	dimension of the examples
s	value of the diagonal elements of the covariance matrix

Value

a matrix with dim rows (variables) and n*6 columns (examples)

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

Examples

```
generate.sample7()
# Generation of a data set with 60 100-dimensional examples
# data have a two-level hierarchical structure with respectively 2 and 6 clusters
generate.sample7(n = 10, m = 10, dim = 100, d = 4, s = 0.4)
```

---

generate.uniform *Uniform bidimensional data generator*

Description

Data are generated according to a bidimensional grid with equispatiated data.

Usage

```
generate.uniform(n = 11, range = c(0, 1))
```

Arguments

n	square root of the number of examples
range	vector with 2 values: min and max coordinates of the bidimensional grid

Value

a data matrix with examples in columns

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

Examples

```
generate.uniform()
# Generation of a bidimensional grid with 100 examples
generate.uniform(n = 10, range = c(0, 1))
```

generate.uniform.random

Uniform bidimensional random data generator.

Description

Data are generated according to a uniform bidimensional random distribution.

Usage

```
generate.uniform.random(n = 100, range = c(0, 1))
```

Arguments

n	number of examples
range	vector with 2 values: min and max random uniform values

Value

a data matrix with examples in columns

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

Examples

```
generate.uniform.random()
# Generation of  bidimensional data randomly distributed
generate.uniform.random(n = 10, range = c(0, 1))
```

JL.predict.dim

Dimension of the subspace or the distortion predicted according to the Johnson Lindenstrauss lemma

Description

Functions to compute the dimension of the subspace or the distortion predicted by the Johnson Lindenstrauss lemma.

Usage

```
JL.predict.dim(n, epsilon = 0.5)

JL.predict.dim.multiple(n, epsilon = 0.5, t = 10)

JL.predict.distortion(n, dim = 10)
```

Arguments

n	cardinality of the data
epsilon	distortion (0 < epsilon <= 0.5)
t	number of multiple projections
dim	dimensionality of the projected subspace

Details

JL.predict.dim predicts the dimension of random projection we need to obtain a given distortion according to JL lemma:

$$d = 4 * \frac{\log n}{\epsilon^2}$$

where d is the dimension of the random projection, n the cardinality of the data and $1+\epsilon$ the theoretical distortion (maximum expansion) induced by the randomized projection into the d-dimensional subspace.

JL.predict.dim.multiple predicts the dimension of random projection we need to obtain a given distortion according to JL lemma when t multiple projections are performed:

$$d = 4 * \frac{\log n + \log t}{\epsilon^2}$$

where d is the dimension of the random projection, n the cardinality of the data and $1+\epsilon$ the theoretical distortion (maximum expansion) induced by the randomized projection into the d-dimensional subspace.

JL.predict.distortion predicts the distortion of a random projection for a given subspace dimension according to JL lemma

$$\epsilon = \sqrt{\frac{4 * \log n}{d}}$$

where d is the dimension of the random projection, n the cardinality of the data and $1+\epsilon$ the theoretical distortion (maximum expansion) induced by the randomized projection into the d-dimensional subspace.

Value

the corresponding dimension of the subspace or the ϵ value of the $1 + \epsilon$ max. expansion (distortion)

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

References

W.Johnson, J.Lindenstrauss, Extensions of Lipshitz mapping into Hilbert space, in: Conference in modern analysis and probability, Vol. 26 of Contemporary Mathematics, Amer. Math. Soc., 1984, pp. 189–206.

See Also

Plus.Minus.One.random.projection, norm.random.projection, Achlioptas.random.projection, random.subspace

Examples

```
# dimension of the projected space that we need to obtain a theoretical 1.5 distortion
# (max. expansion), when 20 data examples are available.
d <- JL.predict.dim(n=20, epsilon = 0.5)
# dimension of the projected space that we need to obtain a theoretical 1.2 distortion
#(max. expansion), when 20 data examples are available, and 10 random projections
d <- JL.predict.dim.multiple(n=20, epsilon = 0.5, t = 10)
# distortion 1+epsilon that is obtained with 30 examples and a random projection
# in a 100-dimensional subspace
epsilon <- JL.predict.distortion(n=30, dim = 100)
```

Max.Expansion *Distortion measures: Max., min, and average expansion and contraction*

Description

Measures to evaluate the distortion induced by randomized projection between euclidean spaces. They evaluate the maximum, minimum and average expansion and contraction of the distances between pairs of points embedded in euclidean spaces.

Usage

```
Max.Expansion(m, m.rid)

Min.Expansion(m, m.rid)

Max.Min.Expansion(m, m.rid)

Average.Expansion(m, m.rid)

Max.Contraction(m, m.rid)

Max.Min.Contraction(m, m.rid)

Average.Contraction(m, m.rid)
```

Arguments

m	data matrix in the original space (rows are are examples, columns are components)
m.rid	data matrix in the reduced space (rows are are examples, columns are components)

Details

If $u, v \in \mathcal{S} \subset \mathcal{R}^d$, $f : \mathcal{R}^d \to \mathcal{R}^d$ is a randomized map with $d' < d$, then we have:

$$max.expansion = max_{u,v \in S} \frac{||f(u) - f(v)||}{||u - v||}$$

$$min.expansion = min_{u,v \in S} \frac{||f(u) - f(v)||}{||u - v||}$$

$$average.expansion = \frac{1}{(|S| * (|S| - 1)} sum_{u,v \in S} \frac{||f(u) - f(v)||}{||u - v||}$$

$$max.contraction = max_{u,v \in S} \frac{||u - v||}{||f(u) - f(v)||}$$

$$min.contraction = min_{u,v \in S} \frac{||u - v||}{||f(u) - f(v)||}$$

$$average.contraction = \frac{1}{(|S| * (|S| - 1)} sum_{u,v \in S} \frac{||u - v||}{||f(u) - f(v)||}$$

Value

Max.Expansion, Min.Expansion, Average.Expansion, Max.Contraction, Average.Contraction return a single real value. Max.Min.Expansion and Max.Min.Contraction a pair (vector) of real values.

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

References

A. Bertoni and G. Valentini, Random projections for assessing gene expression cluster stability, Special Session biostatistics and bioinformatics IJCNN 2005, The IEEE-INNS International Joint Conference on Neural Networks, Montreal, 2005.

Examples

```
# PMO projection from a 1000 dimensional space to a 50-dimensional subspace
m <- matrix(runif(10000), nrow=1000)
m.rid <- Plus.Minus.One.random.projection(d = 50, m, scaling = TRUE)
# Computation of the distortion induced by the PMO projection
max.exps <- Max.Expansion(m, m.rid)
min.exps <- Min.Expansion(m, m.rid)
# the same as above with max e min expansion stored in the same vector
max.min.exps <- Max.Min.Expansion(m, m.rid)
av.exps <- Average.Expansion(m, m.rid)
max.min.contr <- Max.Min.Contraction(m, m.rid)
av.contr <- Average.Contraction(m, m.rid)
```

`Multiple.Random.fuzzy.kmeans`
Multiple Random fuzzy-k-means clustering

Description

Multiple Random fuzzy-k-means clusterings are computed using random projections of data. The crisp clustering is obtained by defuzzyfication via the nearest crisp clustering: each example is assigned to the cluster for which it has the largest membership. The base fuzzy algorithm used is `fanny` of the `cluster` package. It assumes that the label of the examples are integers starting from 1 to ncol(M). Several randomized maps may be used: RS, PMO, Normal and Achlioptas random projections

Usage

```
Multiple.Random.fuzzy.kmeans(M, dim, pmethod = "PMO", c = 3, n = 50,
                             scale = TRUE, seed = -1, distance = "euclidean")
```

Arguments

`M`	matrix of data: rows are variables and columns are examples
`dim`	subspace dimension
`pmethod`	projection method. It must be one of the following: "RS" (random subspace projection) "PMO" (Plus Minus One random projection) "Norm" (normal random projection) "Achlioptas" (Achlioptas random projection)
`c`	number of clusters
`n`	number of RS projections
`scale`	if TRUE randomized projections are scaled (default)
`seed`	numerical seed for the random generator
`distance`	it must be one of the two: "euclidean" (default) or "pearson" (that is 1 - Pearson correlation)

Value

a list of the n clusterings. Each clustering is a list of vectors, and each vector represents a single cluster. The elements of the vectors are integers that corresponds to the number of the columns (examples) of the matrix M of the data.

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

Examples

```
# Multiple (20) fuzzy-k-means clusterings using Normal projections.
M <- generate.sample0(n=10, m=2, sigma=1, dim=800)
l.norm <- Multiple.Random.fuzzy.kmeans (M, dim=100, pmethod="Norm", c=3, n=20)
# The same as above, using Random Subspace projections.
l.RS <-  Multiple.Random.fuzzy.kmeans (M, dim=100, pmethod="RS", c=3,  n=20)
# The same as above, using PMO projections, but with the number of clusters set to 5
l.RS.PMO <-  Multiple.Random.fuzzy.kmeans (M, dim=100, pmethod="PMO", c=5, n=20)
```

`Multiple.Random.hclustering`

Multiple Random hierarchical clustering

Description

Multiple Random hierarchical clusterings are computed using random projections of data. It assumes that the label of the examples are integers starting from 1 to ncol(M). Several randomized maps may be used: RS, PMO, Normal and Achlioptas random projections.

Usage

```
Multiple.Random.hclustering(M, dim, pmethod = "RS", c = 3, hmethod = "average",
                            n = 50, scale = TRUE, seed = 100, distance="euclidean")
```

Arguments

`M`	matrix of data: rows are variables and columns are examples
`dim`	subspace dimension
`pmethod`	projection method. It must be one of the following: "RS" (random subspace projection) "PMO" (Plus Minus One random projection) "Norm" (normal random projection) "Achlioptas" (Achlioptas random projection)
`c`	number of clusters
`hmethod`	the agglomeration method to be used. This should be one of "ward.D", "single", "complete", "average", "mcquitty", "median" or "centroid", according to the `hclust` method of the package `stats`.
`n`	number of random projections
`scale`	if TRUE (default) the random projections are scaled
`seed`	numerical seed for the random generator
`distance`	it must be one of the two: "euclidean" (default) or "pearson" (that is 1 - Pearson correlation)

Value

a list of the n clusterings obtained by randomized hierarchical clustering. Each clustering is a list vector, and each vector represents a single cluster. The elements of the vectors are integers that corresponds to the number of the columns (examples) of the matrix M of the data.

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

See Also

Achlioptas.random.projection, Plus.Minus.One.random.projection,

norm.random.projection,random.subspace

Examples

```
# Multiple (20) hierarchical clusterings using Normal projections.
M <- generate.sample0(n=10, m=2, sigma=2, dim=800)
l.norm <- Multiple.Random.hclustering (M, dim=100, pmethod="Norm",
                                       c=3, hmethod="average", n=20)
# The same as above, using Random Subspace projections.
l.RS <-  Multiple.Random.hclustering (M, dim=100, pmethod="RS", c=3,
                                      hmethod="average", n=20)
# The same as above, using PMO projections, but with the number of clusters set to 5
l.RS <-  Multiple.Random.hclustering (M, dim=100, pmethod="PMO", c=5,
                                      hmethod="average", n=20)
# The same as above, using the single linkage method
l.RS.single <-  Multiple.Random.hclustering (M, dim=100, pmethod="PMO",
                                             c=5, hmethod="single", n=20)
```

Multiple.Random.kmeans

Multiple Random k-means clustering

Description

Multiple Random k-means clusterings are computed using random projections of data. It assumes that the label of the examples are integers starting from 1 to ncol(M). Several randomized maps may be used: RS, PMO, Normal and Achlioptas random projections

Usage

```
Multiple.Random.kmeans(M, dim, pmethod = "PMO", c = 3, n = 50, it.max = 1000,
                       scale = TRUE, seed = 100)
```

Arguments

M	matrix of data: rows are variables and columns are examples
dim	subspace dimension
pmethod	projection method. It must be one of the following: "RS" (random subspace projection) "PMO" (Plus Minus One random projection) "Norm" (normal random projection) "Achlioptas" (Achlioptas random projection)

c	number of clusters
n	number of RS projections
it.max	maximum number of iteration of the k-means algorithm (default 1000)
scale	if TRUE randomized projections are scaled (default)
seed	numerical seed for the random generator

Value

a list of the n clusterings. Each clustering is a list of vectors, and each vector represents a single cluster. The elements of the vectors are integers that corresponds to the number of the columns (examples) of the matrix M of the data.

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

Examples

```
# Multiple (20) k-means clusterings using Normal projections.
M <- generate.sample0(n=10, m=2, sigma=2, dim=800)
l.norm <- Multiple.Random.kmeans (M, dim=100, pmethod="Norm", c=3, n=20)
# The same as above, using Random Subspace projections.
l.RS <-  Multiple.Random.kmeans (M, dim=100, pmethod="RS", c=3,  n=20)
# The same as above, using PMO projections, but with the number of clusters set to 5
l.RS.PMO <-  Multiple.Random.kmeans (M, dim=100, pmethod="PMO", c=5, n=20)
```

Multiple.Random.PAM *Multiple Random PAM clustering*

Description

Multiple Random Partition Around Medoids (PAM) clusterings are computed using random projections of data. The pam function of the package cluster is used as implementation of the base PAM algorithm. It assumes that the label of the examples are integers starting from 1 to ncol(M). Several randomized maps may be used: RS, PMO, Normal and Achlioptas random projections.

Usage

```
Multiple.Random.PAM(M, dim, pmethod = "PMO", c = 3, n = 50, scale = TRUE,
                    seed = -1, distance = "euclidean")
```

Arguments

M	matrix of data: rows are variables and columns are examples
dim	subspace dimension
pmethod	projection method. It must be one of the following: "RS" (random subspace projection) "PMO" (Plus Minus One random projection) "Norm" (normal random projection) "Achlioptas" (Achlioptas random projection)
c	number of clusters
n	number of RS projections
scale	if TRUE randomized projections are scaled (default)
seed	numerical seed for the random generator
distance	it must be one of the two: "euclidean" (default) or "pearson" (that is 1 - Pearson correlation)

Value

a list of the n clusterings obtained by the PAM algorithm clustering. Each clustering is a list of vectors, and each vector represents a single cluster. The elements of the vectors are integers that corresponds to the number of the columns (examples) of the matrix M of the data.

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

Examples

```
# Multiple (20) PAM clusterings using Normal projections.
M <- generate.sample0(n=10, m=2, sigma=2, dim=800)
l.norm <- Multiple.Random.PAM (M, dim=100, pmethod="Norm", c=3, n=20)
# The same as above, using Random Subspace projections.
l.RS <-  Multiple.Random.PAM (M, dim=100, pmethod="RS", c=3,  n=20)
# The same as above, using PMO projections, but with the number of clusters set to 7
l.RS.PMO <-  Multiple.Random.PAM (M, dim=100, pmethod="PMO", c=7, n=20)
```

Norm.hclustering *Multiple Hierarchical clusterings using Normal random projections*

Description

Multiple Hierarchical clusterings using Normal random projections of the data.

Usage

```
Norm.hclustering(M, dim, c = 3, hmethod = "average", n = 50,
                 scale = TRUE, seed = 100, distance="euclidean")

Norm.hclustering.tree(M, dim, hmethod = "average", n = 50, scale = TRUE,
                      seed = 100, distance = "euclidean")
```

Arguments

M	matrix of data: rows are variables and columns are examples
dim	subspace dimension
c	number of clusters
hmethod	the agglomeration method to be used. This should be one of "ward.D", "single", "complete", "average", "mcquitty", "median" or "centroid", according to the hclust method of the package stats.
n	number of random projections
scale	if TRUE (default) Normal random projections are scaled
seed	numerical seed for the random generator
distance	it must be one of the two: "euclidean" (default) or "pearson" (that is 1 - Pearson correlation)

Value

a list with components "cluster" and "tree":

cluster	list of the n clusterings obtained. Each element is in turn a list of vectors that correspond to the clusters of the clustering. Each cluster is represented by a vector of integers whose values corresponds to the indices of the columns (examples) of the original data.
tree	list of the trees generated by the multiple clusterings

Norm.hclustering.tree returns only the list of the trees.

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

See Also

norm.random.projection

Examples

```
# 20 hierarchical clusterings on multiple Normal projected data
# with subspace dimension equal to 100
M <- generate.sample0(n=10, m=2, sigma=1, dim=800)
l <- Norm.hclustering(M, dim=100, hmethod = "average", n = 20, scale = TRUE)
# Equal as above, but only the trees are generated
l <- Norm.hclustering.tree(M, dim=100, hmethod = "average", n = 20, scale = TRUE)
# 10 hierarchical clusterings on multiple Normal projected data
# with subspace dimension equal to 200
M <- generate.sample0(n=8, m=1, sigma=2, dim=1000)
l <- Norm.hclustering(M, dim=200, hmethod = "average", n = 10, scale = TRUE)
```

`norm.random.projection` *Normal random projections*

Description

Random projections to a lower dimension subspace with a normal distributed projection matrix The projection is performed using a normally distributed projection matrix R: its elements R[i,j] ~ N(0,1).

Usage

```
norm.random.projection(d = 2, m, scaling = TRUE)
```

Arguments

`d`	subspace dimension
`m`	data matrix (rows are features and columns are examples)
`scaling`	if TRUE (default) scaling is performed

Details

Normal random projections are randomized map represented by a $d' \times d$ matrix $R = 1/\sqrt{d'}(r_{ij})$, where r_{ij} are distributed according to a gaussian with 0 mean and unit variance, and d' is the dimension of the projected space and d the dimension of the original space.

Value

data matrix (dimension d x ncol(m)) of the examples projected in a d-dimensional subspace

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

References

E.Bingham, H.Mannila, Random projection in dimensionality reduction: Applications to image and text data, in: Proc. of KDD 01, ACM, San Francisco, CA, USA, 2001.

See Also

`Plus.Minus.One.random.projection`, `random.subspace`, `Achlioptas.random.projection`

Examples

```
# Normal random projection from a 1000 dimensional space to a
# 50-dimensional subspace
m <- matrix(runif(10000), nrow=1000)
m.p <- norm.random.projection(d = 50, m, scaling = TRUE)
# Normal random subspace projection from a 10000 dimensional space
# to a 1000-dimensional subspace
m <- matrix(rnorm(500000), nrow=5000)
m.p <- norm.random.projection(d = 1000, m, scaling = TRUE)
# The same as above without scaling
m <- matrix(rnorm(500000), nrow=5000)
m.p <- norm.random.projection(d = 1000, m, scaling = FALSE)
```

Plus.Minus.One.random.projection

Plus-Minus-One (PMO) random projections

Description

Random projections to a lower dimensional subspace with a random +1/-1 projection matrix The projection is performed using a projection matrix R s.t. Prob(R[i,j]=1)=Prob(R[i,j]=-1)=1/2.

Usage

```
Plus.Minus.One.random.projection(d = 2, m, scaling = TRUE)
```

Arguments

d	subspace dimension
m	data matrix (rows are features and columns are examples)
scaling	if TRUE (default) scaling is performed

Details

Plus-Minus-One (PMO) random projections are represented by $d' \times d$ matrices $R = 1/\sqrt{d'}(r_{ij})$, where r_{ij} are uniformly chosen in $\{-1, 1\}$, such that $Prob(r_{ij} = 1) = Prob(r_{ij} = -1) = 1/2$.

Value

data matrix (dimension d X ncol(m)) of the examples projected in a d-dimensional random subspace

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

See Also

random.subspace, norm.random.projection, Achlioptas.random.projection

Examples

```
# PMO projection from a 1000 dimensional space to a 50-dimensional subspace
m <- matrix(runif(10000), nrow=1000)
m.p <- Plus.Minus.One.random.projection(d = 50, m, scaling = TRUE)
# PMO projection from a 10000 dimensional space to a 1000-dimensional subspace
m <- matrix(rnorm(500000), nrow=5000)
m.p <- Plus.Minus.One.random.projection(d = 1000, m, scaling = TRUE)
# The same as above without scaling
m <- matrix(rnorm(500000), nrow=5000)
m.p <- Plus.Minus.One.random.projection(d = 1000, m, scaling = FALSE)
```

PMO.hclustering *Multiple Hierarchical clusterings using Plus Minus One (PMO) random projections*

Description

Multiple Hierarchical clusterings using Plus Minus One (PMO) random projections of the data.

Usage

```
PMO.hclustering(M, dim, c = 3, hmethod = "average", n = 50,
                scale = TRUE, seed = 100, distance="euclidean")

PMO.hclustering.tree(M, dim, hmethod = "average", n = 50,
                     scale = TRUE, seed = 100, distance = "euclidean")
```

Arguments

M	matrix of data: rows are variables and columns are examples
dim	subspace dimension
c	number of clusters
hmethod	the agglomeration method to be used. This should be one of "ward.D", "single", "complete", "average", "mcquitty", "median" or "centroid", according to the hclust method of the package stats.
n	number of random projections
scale	if TRUE (default) Achlioptas random projections are scaled
seed	numerical seed for the random generator
distance	it must be one of the two: "euclidean" (default) or "pearson" (that is 1 - Pearson correlation)

Value

a list with components "cluster" and "tree":

`cluster`	list of the n clusterings obtained. Each element is in turn a list of vectors that correspond to the clusters of the clustering. Each cluster is represented by a vector of integers whose values corresponds to the indices of the columns (examples) of the original data.
`tree`	list of the trees generated by the multiple clusterings

`PMO.hclustering.tree` returns only the list of the trees.

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

See Also

`Plus.Minus.One.random.projection`

Examples

```
# 20 hierarchical clusterings on multiple PMO projected data
# with subspace dimension equal to 100
M <- generate.sample0(n=10, m=2, sigma=1, dim=800)
l <- PMO.hclustering(M, dim=100, hmethod = "average", n = 20, scale = TRUE)
# Equal as above, but only the trees are generated
l <- PMO.hclustering.tree(M, dim=100, hmethod = "average", n = 20, scale = TRUE)
# 10 hierarchical clusterings on multiple PMO projected data
# with subspace dimension equal to 200
M <- generate.sample0(n=8, m=1, sigma=2, dim=1000)
l <- PMO.hclustering(M, dim=200, hmethod = "average", n = 10, scale = TRUE)
```

`rand.norm` *Random generation of normal distributed data*

Description

Random generation of a matrix of n columns with with diagonal covariance matrix (rand.norm.generate) or with full covariance matrix (rand.norm.generate.full). These functions are used by `generate.sampleN` functions $0 \leq N \leq 5$ to generate the data.

Usage

```
rand.norm.generate(n = 5, mean = 0, sd = 1)
rand.norm.generate.full(n = 5, mean = c(0, 0),
                        Sigma = matrix(c(0.1, 0, 0, 0.1), 2, 2))
```

Arguments

n	number of samples to be generated
mean	vector of means
sd	vector of standard deviations
Sigma	Covariance matrix

Value

a matrix of n columns with length(mean) rows. With rand.norm.generate Row[i] has mean mean[i] and standard deviation sd[i]. With rand.norm.generate.full Row[i] has mean mean[i]

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

See Also

generate.sample0, generate.sample1, generate.sample2

generate.sample3, generate.sample4, generate.sample5

Examples

```
library(MASS)
rand.norm.generate(n = 10)
rand.norm.generate(n = 10, mean = c(0,1,2), sd = c(1,1,5))
rand.norm.generate.full()
rand.norm.generate.full(n = 10, mean = c(0, 0, 2),
                        Sigma = matrix(seq(1,1.8, by=0.1), 3, 3))
```

---

random.component.selection

Function to randomly select the indices of the variables selected by the random subspace projection

---

Description

It is used by the function random.subspace to randomly select the indices of the variables used for the random subspace projections. It randomly select a subset of the indices, that is a set of positive integers that correspond to the selected variables

Usage

```
random.component.selection(d = 2, d.original = 10)
```

Arguments

d	subspace dimension
d.original	dimension of the space from which components are randomly selected

Value

vector of the selected features: it contain the indices of the components randomly selected

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

See Also

random.subspace

Examples

```
# it generates a vector of 2 elements whose components are randomly
# chosen from 1..10
random.component.selection(d = 2, d.original = 10)
# it generates a vector of 10 elements whose components are randomly
# chosen from 1..1000
random.component.selection(d = 10, d.original = 1000)
```

Random.fuzzy.kmeans.validity

Fuzzy-k-means clustering and validity indices computation using random projections of data

---

Description

This function applies the fuzzy-k-means clustering algorithm to the data and then computes stability indices for the obtained cluster using multiple random subspace projections. It computes the validity indices for each cluster found in the original space, the overall validity index for the clustering and (optionally) the set of the AC indices. Different randomized maps (e.g. PMO, Achlioptas, Normal, Random Subspace projections) may be applied. It assumes that the label of the examples are integer starting from 1 to ncol(M). Note that the fuzzy-k-means algorithm strongly depends from the initial conditions. Hence choosing different random seed we may obtain different results; setting seed=-1 (default) each time a different random seed is chosen.

Usage

```
Random.fuzzy.kmeans.validity(M, dim, pmethod = "PMO", c = 3, n = 50, scale = TRUE,
                             seed = -1, AC = TRUE, distance = "euclidean")
```

Arguments

M	matrix of data: rows are variables and columns are examples
dim	subspace dimension
pmethod	projection method. It must be one of the following: "RS" (random subspace projection) "PMO" (Plus Minus One random projection) "Norm" (normal random projection) "Achlioptas" (Achlioptas random projection)
c	number of clusters
n	number of random projections
scale	if TRUE (default) the random projections are scaled
seed	numerical seed for the random generator
AC	if TRUE (default) the AC indices are computed.
distance	it must be one of the two: "euclidean" (default) or "pearson" (that is 1 - Pearson correlation)

Value

a list with esixight components: "validity", "overall.validity", "similarity.matrix", "dim", "cluster", "orig.cluster":

validity	a vector with the validity of each of the c clusters
overall.validity	validity index of the overall clustering
similarity.matrix	pairwise similarity matrix between examples
dimension	random projection dimension
cluster	is the list of the n clustering obtained by multiple k-means clustering on the projected subspace
orig.cluster	list of the clusters in the original space
AC	matrix with the Assignment Confidence index for each example. Each row corresponds to an example, each column to a cluster (optional)

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

See Also

Achlioptas.random.projection, Plus.Minus.One.random.projection,

norm.random.projection,random.subspace,

Cluster.validity, Validity.indices, AC.index

Examples

```
# Assessment of the reliability of clusters discovered
# by fuzzy k-means using RS projections.
M <- generate.sample0(n=10, m=2, sigma=1, dim=800)
l<-Random.fuzzy.kmeans.validity(M, dim=30, pmethod = "RS", c = 3,  n = 20)
# The same as above, but using PMO projections.
l<-Random.fuzzy.kmeans.validity(M, dim=30, pmethod = "PMO", c = 3, n = 20)
# The same as above, but evaluating clusterings with 5 clusters
l<-Random.fuzzy.kmeans.validity(M, dim=30, pmethod = "PMO", c = 5, n = 20)
# The same as above, but evaluating clusterings with 10 clusters
l<-Random.fuzzy.kmeans.validity(M, dim=30, pmethod = "PMO", c = 10, n = 20)
# Assessment of the reliability of the clusters using projections
# with limited distortion (max.
# expansion lower than 1.3 according to the Johnson Lindenstrauss lemma)
d <- JL.predict.dim(n=30, epsilon=0.3)
l<-Random.fuzzy.kmeans.validity(M, dim=d, pmethod = "PMO", c = 3, n = 20)
```

`Random.hclustering.validity`

Random hierarchical clustering and validity index computation using random projections of data.

Description

This function applies a hierarchical clustering algorithm to the data and then computes stability indices for the obtained cluster using multiple random subspace projections. The reliability of clusters discovered by a hierarchical clustering algorithm is assessed using randomized projections. The validity indices for each individual cluster, the overall validity index of the clustering and the AC indices are computed. Different hierarchical clusterings may be used (e.g. average, complete and single linkage or the Ward's method) as well as different randomized maps (e.g. PMO, Achlioptas, Normal, Random Subspace projections). It assumes that the label of the examples are integer starting from 1 to ncol(M).

Usage

```
Random.hclustering.validity(M, dim, pmethod = "RS", c = 3, hmethod = "average",
                            n = 50, scale = TRUE, seed = 100, AC=TRUE,
                            distance="euclidean")
```

Arguments

`M`	matrix of data: rows are variables and columns are examples
`dim`	subspace dimension
`pmethod`	projection method. It must be one of the following: "RS" (random subspace projection) "PMO" (Plus Minus One random projection) "Norm" (normal random projection) "Achlioptas" (Achlioptas random projection)

c	number of clusters
hmethod	the agglomeration method to be used. This should be one of "ward.D", "single", "complete", "average", "mcquitty", "median" or "centroid", according to the hclust method of the package stats.
n	number of random projections
scale	if TRUE (default) the random projections are scaled
seed	numerical seed for the random generator
AC	if TRUE (default) the AC indices are computed.
distance	it must be one of the two: "euclidean" (default) or "pearson" (that is 1 - Pearson correlation).

Value

a list with eight components: "validity", "overall.validity", "similarity.matrix", "dim", "cluster", "tree", "orig.tree", "orig.cluster":

validity	a vector with the validity of each of the c clusters
overall.validity	validity index of the overall clustering
similarity.matrix	pairwise similarity matrix between examples
dimension	random projection dimension
cluster	list of the n clustering obtained by randomized hierarchical clustering
tree	list of the n trees obtained by the randomized hierarchical clustering
orig.tree	tree built in the original space
orig.cluster	list of the clusters in the original space
AC	matrix with the Assignment Confidence index for each example. Each row corresponds to an example, each column to a cluster (optional)

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

See Also

Achlioptas.random.projection, Plus.Minus.One.random.projection,

norm.random.projection,random.subspace,

Cluster.validity, Validity.indices, AC.index

Examples

```
# Assessment of the reliability of clusters discovered
# by hierarchical clustering using RS projections.
M <- generate.sample0(n=10, m=2, sigma=2, dim=800)
l<-Random.hclustering.validity(M, dim=30, pmethod = "RS", c = 3,
                               hmethod = "average", n = 20)
# The same as above, but using PMO projections.
l<-Random.hclustering.validity(M, dim=30, pmethod = "PMO", c = 3,
                               hmethod = "average", n = 20)
# The same as above, but evaluating clusterings with 5 clusters
l<-Random.hclustering.validity(M, dim=30, pmethod = "PMO", c = 5,
                               hmethod = "average", n = 20)
# The same as above, but evaluating clusterings with 10 clusters
l<-Random.hclustering.validity(M, dim=30, pmethod = "PMO", c = 10,
                               hmethod = "average", n = 20)
# Assessment of the reliability of the clusters using projections
# with limited distortion (max.
# expansion lower than 1.3 according to the Johnson Lindenstrauss lemma)
d <- JL.predict.dim(n=30, epsilon=0.3)
l<-Random.hclustering.validity(M, dim=d, pmethod = "PMO", c = 3,
                               hmethod = "average", n = 20)
```

`Random.kmeans.validity` *k-means clustering and validity indices computation using random projections of data*

Description

This function applies a k-means clustering algorithm to the data and then computes stability indices for the obtained cluster using multiple random subspace projections. It computes the validity indices for each cluster found in the original space, the overall validity index for the clustering and (optionally) the set of the AC indices. Different randomized maps (e.g. PMO, Achlioptas, Normal, Random Subspace projections) may be applied. It assumes that the label of the examples are integer starting from 1 to ncol(M). Note that the k-means algorithm strongly depends from the initial conditions. Hence choosing different random seed we may obtain different results; setting seed=-1 (default) each time a different random seed is chosen.

Usage

```
Random.kmeans.validity(M, dim, pmethod = "PMO", c = 3, it.max = 1000,
                       n = 50, scale = TRUE, seed = -1, AC = TRUE)
```

Arguments

`M`	matrix of data: rows are variables and columns are examples
`dim`	subspace dimension

pmethod	projection method. It must be one of the following: "RS" (random subspace projection) "PMO" (Plus Minus One random projection) "Norm" (normal random projection) "Achlioptas" (Achlioptas random projection)
c	number of clusters
it.max	maximum number of iteration of the k-means algorithm (default 1000)
n	number of random projections
scale	if TRUE (default) the random projections are scaled
seed	numerical seed for the random generator
AC	if TRUE (default) the AC indices are computed.

Value

a list with esixight components: "validity", "overall.validity", "similarity.matrix", "dim", "cluster", "orig.cluster":

validity	a vector with the validity of each of the c clusters
overall.validity	validity index of the overall clustering
similarity.matrix	pairwise similarity matrix between examples
dimension	random projection dimension
cluster	is the list of the n clustering obtained by multiple k-means clustering on the projected subspace
orig.cluster	list of the clusters in the original space
AC	matrix with the Assignment Confidence index for each example. Each row corresponds to an example, each column to a cluster (optional)

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

See Also

Achlioptas.random.projection, Plus.Minus.One.random.projection, norm.random.projection,random.subspace, Cluster.validity, Validity.indices, AC.index

Examples

```
# Assessment of the reliability of clusters discovered
# by k-means using RS projections.
M <- generate.sample0(n=10, m=2, sigma=2, dim=800)
l<-Random.kmeans.validity(M, dim=30, pmethod = "RS", c = 3,  n = 20)
# The same as above, but using PMO projections.
l<-Random.kmeans.validity(M, dim=30, pmethod = "PMO", c = 3, n = 20)
# The same as above, but evaluating clusterings with 5 clusters
```

```
l<-Random.kmeans.validity(M, dim=30, pmethod = "PMO", c = 5, n = 20)
# The same as above, but evaluating clusterings with 10 clusters
l<-Random.kmeans.validity(M, dim=30, pmethod = "PMO", c = 10, n = 20)
# Assessment of the reliability of the clusters using projections
# with limited distortion (max.
# expansion lower than 1.3 according to the Johnson Lindenstrauss lemma)
d <- JL.predict.dim(n=30, epsilon=0.3)
l<-Random.kmeans.validity(M, dim=d, pmethod = "PMO", c = 3, n = 20)
```

`Random.PAM.validity` *PAM clustering and validity indices computation using random projections of data*

Description

This function applies the Prediction Around Medoids (PAM) clustering algorithm to the data and then computes stability indices for the obtained cluster using multiple random subspace projections. It computes the validity indices for each cluster found in the original space, the overall validity index for the clustering and (optionally) the set of the AC indices. Different randomized maps (e.g. PMO, Achlioptas, Normal, Random Subspace projections) may be applied. It assumes that the label of the examples are integer starting from 1 to ncol(M). The `pam` function of the package `cluster` is used as implementation of the base PAM algorithm.

Usage

```
Random.PAM.validity(M, dim, pmethod = "PMO", c = 3, n = 50, scale = TRUE,
                    seed = -1, AC = TRUE, distance = "euclidean")
```

Arguments

`M`	matrix of data: rows are variables and columns are examples
`dim`	subspace dimension
`pmethod`	projection method. It must be one of the following: "RS" (random subspace projection) "PMO" (Plus Minus One random projection) "Norm" (normal random projection) "Achlioptas" (Achlioptas random projection)
`c`	number of clusters
`n`	number of random projections
`scale`	if TRUE (default) the random projections are scaled
`seed`	numerical seed for the random generator
`AC`	if TRUE (default) the AC indices are computed.
`distance`	it must be one of the two: "euclidean" (default) or "pearson" (that is 1 - Pearson correlation)

Value

a list with esixight components: "validity", "overall.validity", "similarity.matrix", "dim", "cluster", "orig.cluster":

`validity`	a vector with the validity of each of the c clusters
`overall.validity`	validity index of the overall clustering
`similarity.matrix`	pairwise similarity matrix between examples
`dimension`	random projection dimension
`cluster`	is the list of the n clustering obtained by multiple PAM clustering on the projected subspace
`orig.cluster`	list of the clusters in the original space
`AC`	matrix with the Assignment Confidence index for each example. Each row corresponds to an example, each column to a cluster (optional)

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

See Also

`Achlioptas.random.projection`, `Plus.Minus.One.random.projection`,

`norm.random.projection`,`random.subspace`,

`Cluster.validity`, `Validity.indices`, `AC.index`

Examples

```
# Assessment of the reliability of clusters discovered
# by fuzzy k-means using RS projections.
M <- generate.sample0(n=10, m=2, sigma=1, dim=800)
l<-Random.PAM.validity(M, dim=30, pmethod = "RS", c = 3,  n = 20)
# The same as above, but using PMO projections.
l<-Random.PAM.validity(M, dim=30, pmethod = "PMO", c = 3, n = 20)
# The same as above, but evaluating clusterings with 5 clusters
l<-Random.PAM.validity(M, dim=30, pmethod = "PMO", c = 5, n = 20)
# The same as above, but evaluating clusterings with 10 clusters
l<-Random.PAM.validity(M, dim=30, pmethod = "PMO", c = 10, n = 20)
# Assessment of the reliability of the clusters
# using projections with limited distortion (max.
# expansion lower than 1.3 according to the Johnson Lindenstrauss lemma)
d <- JL.predict.dim(n=30, epsilon=0.3)
l<-Random.PAM.validity(M, dim=d, pmethod = "PMO", c = 3, n = 20)
```

`random.subspace` *Random Subspace (RS) projections*

Description

Random projections to a lower dimension subspace (random subspace method) The projection is performed randomly selecting a subset of variables (components) and then projecting the data onto the selected components. It is the projection used by Ho for her Random subspace ensemble algorithm.

Usage

```
random.subspace(d = 2, m, scaling = TRUE)
```

Arguments

`d`	subspace dimension
`m`	data matrix (rows are features and columns are examples)
`scaling`	if TRUE (default) scaling is performed

Details

Random Subspace (RS) are randomized maps represented by $d' \times d$ matrices $R = \sqrt{d/d'}(r_{ij})$, where r_{ij} are uniformly chosen with entries in $\{0, 1\}$, and with exactly one 1 per row and at most one 1 per column (d' is the dimension of the projected space and d the dimension of the original space). It is worth noting that, in this case, the "compressed" data set $D_R = RD$ can be quickly computed in time $\mathcal{O}(nd')$, independently from d.

Value

data matrix (dimension d X ncol(m)) of the examples projected in a d-dimensional random subspace

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

References

T.Ho, The random subspace method for constructing decision forests, IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (8) (1998) 832-844.

See Also

`Plus.Minus.One.random.projection`, `norm.random.projection`,

`Achlioptas.random.projection`

Examples

```
# Random subspace projection from a 1000 dimensional space
# to a 50-dimensional subspace
m <- matrix(runif(10000), nrow=1000)
m.p <- random.subspace(d = 50, m, scaling = TRUE)
# Random subspace projection from a 10000 dimensional space
# to a 1000-dimensional subspace
m <- matrix(rnorm(500000), nrow=5000)
m.p <- random.subspace(d = 1000, m, scaling = TRUE)
# The same as above without scaling
m <- matrix(rnorm(500000), nrow=5000)
m.p <- random.subspace(d = 1000, m, scaling = FALSE)
```

RS.hclustering *Multiple Hierarchical clusterings using RS random projections*

Description

Multiple Hierarchical clusterings using RS random projections of the data.

Usage

```
RS.hclustering(M, dim, c = 3, hmethod = "average", n = 50, scale = TRUE,
               seed = 100, distance="euclidean")

RS.hclustering.tree(M, dim, hmethod = "average", n = 50, scale = TRUE,
                    seed = 100, distance = "euclidean")
```

Arguments

M	matrix of data: rows are variables and columns are examples
dim	subspace dimension
c	number of clusters
hmethod	the agglomeration method to be used. This should be one of "ward.D", "single", "complete", "average", "mcquitty", "median" or "centroid", according to the hclust method of the package stats.
n	number of random projections
scale	if TRUE (default) RS random projections are scaled
seed	numerical seed for the random generator
distance	it must be one of the two: "euclidean" (default) or "pearson" (that is 1 - Pearson correlation)

Value

a list with components "cluster" and "tree":

`cluster`	list of the n clusterings obtained. Each element is in turn a list of vectors that correspond to the clusters of the clustering. Each cluster is represented by a vector of integers whose values corresponds to the indices of the columns (examples) of the original data.
`tree`	list of the trees generated by the multiple clusterings

`RS.hclustering.tree` returns only the list of the trees.

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

See Also

`random.subspace`

Examples

```
# 20 hierarchical clusterings on multiple RS projected data
# with subspace dimension equal to 100
M <- generate.sample0(n=10, m=2, sigma=1, dim=800)
l <- RS.hclustering(M, dim=100, hmethod = "average", n = 20, scale = TRUE)
# Equal as above, but only the trees are generated
l <- RS.hclustering.tree(M, dim=100, hmethod = "average", n = 20, scale = TRUE)
# 10 hierarchical clusterings on multiple RS projected data
# with subspace dimension equal to 200
M <- generate.sample0(n=8, m=1, sigma=2, dim=1000)
l <- RS.hclustering(M, dim=200, hmethod = "average", n = 10, scale = TRUE)
```

`Transform.vector.to.list` *Vector to list transformation of cluster representation*

Description

It transforms a clustering from a vector representation to a list representation. It accepts as input a vector that represents a clustering; the indices of the vectors refer to the examples and their integer content to the number of the cluster they belong. The function returns a list that represents the same clustering; each element of the list is a vector representing the cluster. The elements of the vectors are the indices of the examples (that is they correspond to the indices of the vector representation of the clustering). This list representation of the cluster may be used to compute the validity indices of the clustering.

Usage

```
Transform.vector.to.list(v)
```

Arguments

v	vector representing the clustering

Value

a list that represents the clustering; each element is a vector representing a single cluster.

Examples

```
library(cluster);
# transforming a clustering vector obtained with PAM to a clustering list
M <- generate.sample0(n=10, m=2, sigma=1, dim=500)
clustering.vector <- pam (t(M),3,cluster.only=TRUE);
clustering.list <- Transform.vector.to.list(clustering.vector);
# transforming a clustering vector obtained with kmeans to a clustering list
r<-kmeans(t(M), 3, 100);
clustering.list.kmeans <- Transform.vector.to.list(r$cluster);
```

Validity.indices *Function to compute the validity index of each cluster.*

Description

It computes the validity index (e.g. the stability index) for each individual cluster. This function is called by `Cluster.validity` and `Cluster.validity.from.similarity`

Usage

```
Validity.indices(cluster, c, Sim.M)
```

Arguments

cluster	list of clusters representing a clustering in the original space. Each element of the list is a vector whose elements are the examples belonging to the cluster.
c	number of clusters
Sim.M	the pairwise similarity matrix

Details

Using the similarity matrix M, the *stability index* s for a cluster A is:

$$s(A) = \frac{1}{|A|(|A|-1)} \sum_{(i,j)\in A\times A, i\neq j} M_{ij}$$

The index $s(A)$ estimates the stability of a cluster A by measuring how much the projections of the pairs $(i, j) \in A \times A$ occur together in the same cluster in the projected subspaces. The stability index has values between 0 and 1: low values indicate no reliable clusters, high values denote stable clusters.

Value

vector of the validity indices. Each element corresponds to validity index of each cluster.

Author(s)

Giorgio Valentini <valentini@di.unimi.it>

See Also

Cluster.validity, Cluster.validity.from.similarity, Do.similarity.matrix.partition, Do.similarity.matrix

Examples

```
# Computation of the stability indices found out by a hierarchical clustering algorithm
M <- generate.sample0(n=10, m=2, sigma=2, dim=800)
d <- dist (t(M));
tree <- hclust(d, method = "average");
plot(tree, main="");
cl.orig <- rect.hclust(tree, k = 3);
l.norm <- Multiple.Random.hclustering (M, dim=100, pmethod="Norm",
                                       c=3, hmethod="average", n=20)
Sim <- Do.similarity.matrix.partition(l.norm);
val.indices <- Validity.indices(cl.orig, c=3, Sim)
```

Index