Package 'ordinalClust'

January 8, 2018

Type Package

Title Ordinal Data Clustering, Co-Clustering and Classification

Version 1.2.1

Date 2017-11-16

Author Margot Selosse, Julien Jacques, Christophe Biernacki

Maintainer Margot Selosse <margot.selosse@gmail.com>

Description Ordinal data classification, clustering and co-clustering using model-based approach with the Bos distribution for ordinal data (Christophe Biernacki and Julien Jacques (2016) <doi:10.1007/s11222-015-9585-2>).

Imports Rcpp (>= 0.12.10)

LinkingTo Rcpp, RcppArmadillo

Depends stats (>= 3.3.2), progress (>= 1.1.2)

License GPL (>= 3)

Suggests knitr, rmarkdown, caret, ggplot2

VignetteBuilder knitr

LazyData true

SystemRequirements C++11

NeedsCompilation yes

Repository CRAN

Date/Publication 2018-01-08 22:21:31 UTC

R topics documented:

`bosclassif` *bosclassif*

Description

This function performs a classification on a dataset with features of the ordinal kind, and a label variable of the integer type (1,2,...,kr). The classification function proposes two classification models. The first one, (chosen by the option kc=0), is a multivariate BOS model assuming that, conditionally on the class of the observations, the feature are independent. The second model is a parsimonious version of the first model. Parcimony is introduced by grouping the features into clusters (as in co-clustering) and assuming that the features of a cluster have a common distribution.

Usage

```
bosclassif(x,y,to.predict,kr,kc=0,m,nbSEM=50,nbSEMburn=20,nbindmini=4,
  init='kmeans',disp=TRUE,iterordiEM=10)
```

Arguments

`x`	Matrix of ordinal data. The missing values should be equal to 0.
`y`	Vector of classes for each row of x. Must be labelled with integers (1,2,...,kr).
`to.predict`	Matrix of ordinal data with same number of features than x. Represents the observation the user wants to give a label (1,...,kr).
`kr`	Number of classes.
`kc`	Set to 0 to choose a classical multivariate BOS model. Otherwise, o choose a parsimonious model, set an integer that indicates the number of column clusters.
`m`	Integer that defines the ordinal data's number of levels.
`nbSEM`	Number of SEM-Gibbs iterations realized to estimate parameters.
`nbSEMburn`	Number of SEM-Gibbs burning iterations for estimating parameters. This parameter must be inferior to nbSEM.
`nbindmini`	Minimum number of cells belonging to a block.
`init`	String that indicates the kind of initialisation. Must one of these two words : "kmeans" or "random".
`disp`	Boolean that should be set to TRUE if the user wants the progress bars to be displayed in the console.
`iterordiEM`	Number of iterations for the internal EM algorithm that estimates the parameters of BOS distribution.

Value

`probas.to.predict`
: Matrix with nrow=nrow(to.predict) and ncol=kr. For each observation of argument to.predict, indicates the probability to belong to each class.

`zr.to.predict`
: Vector of length nrow(to.predict) which indicates at which class the observations of to predict has been associated.

`V.to.predict`
: Matrix with nrow=nrow(to.predict) and ncol=kr. Indicates at which class the observations of to.predict has been associated.

`xhat`
: dataset with missing values completed.

`mus`
: For each iteration, an array of dimension kr*kc that represents the mus that were estimated at each iteration.

`ps`
: For each iteration, an array of dimension kr*kc that represents the pis that were estimated at each iteration.

`gamma`
: Array of gammas (row mixing proportions) that were estimated at each iteration.

`rho`
: Array of rhos (columns mixing proportions) that were estimated at each iteration.

`W`
: For each iteration, array of dimension J*H such that W[j,h]=1 if j belongs to cluster h.

`res_mus`
: Array of dimension kr*kc that represents the resulting BOS position parameters mus.

`res_ps`
: Array of dimension kr*kc that represents the resulting BOS precision parameters pis.

`res_gamma`
: Vector with the resulting gamma.

`res_rho`
: Vector with the resulting rho.

`res_V`
: Array with the resulting V.

`res_W`
: Array with the resulting W.

`icl`
: ICL-BIC result value. Must be maximized.

`zc`
: Vector with resulting column partitions.

`probaV`
: Array representing the probability for each row to belong to each row-cluster.

`probaW`
: Array representing the probability for each column to belong to each column-cluster.

Author(s)

Margot Selosse, Julien Jacques, Christophe Biernacki.

Examples

```
# loading the real dataset
data("dataqol.classif")

set.seed(5)

# loading the ordinal data
```

```
M <- as.matrix(dataqol.classif[,2:29])


# creating the classes values
y <- as.vector(dataqol.classif$death)


# sampling datasets for training and to predict
nb.sample <- ceiling(nrow(M)*2/3)
sample.train <- sample(1:nrow(M), nb.sample, replace=FALSE)

M.train <- M[sample.train,]
M.validation <- M[-sample.train,]
nb.missing.validation <- length(which(M.validation==0))
m <- c(4)
M.validation[which(M.validation==0)] <- sample(1:m, nb.missing.validation,replace=TRUE)


y.train <- y[sample.train]
y.validation <- y[-sample.train]



# configuration for SEM algorithm
nbSEM=50
nbSEMburn=40
nbindmini=1

# number of classes to predict
kr <- 2
# different kc to test with cross-validation
kcol <- 1


res <- bosclassif(x=M.train,y=y.train,to.predict=M.validation,kr,kc=kcol,m=m,
                  nbSEM=nbSEM,nbSEMburn=nbSEMburn,
                  nbindmini=nbindmini, disp=TRUE)
```

`bosclassifMulti` *bosclassifMulti*

Description

This function performs a classification on a dataset with features of the ordinal kind, and a label variable of the integer type (1,2,...,kr). The classification function proposes two classification models. The first one, (chosen by the option kc=0), is a multivariate BOS model assuming that, conditionally on the class of the observations, the feature are independent. The second model is a parsimonious

version of the first model. Parcimony is introduced by grouping the features into clusters (as in co-clustering) and assuming that the features of a cluster have a common distribution. Furthermore, the features can have D different numbers of levels but it has to be indicated with argument d.list.

Usage

```
bosclassifMulti(x,y,to.predict, d.list, kr,kc=0,m,nbSEM=50,nbSEMburn=20,nbindmini=4,
  init='kmeans',disp=TRUE,iterordiEM=10)
```

Arguments

x	Matrix of ordinal data. The missing values should be equal to 0.
y	Vector of classes for each row of x. Must be labelled with integers (1,2,...,kr).
to.predict	Matrix of ordinal data with same number of features than x. Represents the observation the user wants to give a label (1,...,kr).
kr	Number of classes.
kc	Set to 0 to choose a classical multivariate BOS model. Otherwise, set a vector of length D. Element d defines the number of column-cluster for the variables of group d.
m	Vector of length D. Element d defines the number of levels for the variables of group d.
d.list	List of length D. Item d is a vector with the columns of that x that have a number of levels of group d.
nbSEM	Number of SEM-Gibbs iterations realized to estimate parameters.
nbSEMburn	Number of SEM-Gibbs burning iterations for estimating parameters. This parameter must be inferior to nbSEM.
nbindmini	Minimum number of cells belonging to a block.
init	String that indicates the kind of initialisation. Must one of these two words : "kmeans" or "random".
disp	Boolean that should be set to TRUE if the user wants the progress bars to be displayed in the console.
iterordiEM	Number of iterations for the internal EM algorithm that estimates the parameters of BOS distribution.

Value

probas.to.predict	Matrix with nrow=nrow(to.predict) and ncol=kr. For each observation of argument to.predict, indicates to belong to each class.
zr.to.predict	Vector of length nrow(to.predict) which indicates at which class the obesrvations of to predict has been asscoiated.
V.to.predict	Matrix with nrow=nrow(to.predict) and ncol=kr. Indicates at which class the obesrvations of to predict has been asscoiated.
xhat	List of length D. Item d represents the group d in dataset, with missing values completed.

mus	List of length D. Item d represents the mus that were estimated at each iteration for group d.
ps	List of length D. Item d represents the pis that were estimated at each iteration for group d.
gamma	Array of gammas that were estimated at each iteration.
rho	List of length D. Item d represents the rho that were estimated at each iteration for group d.
W	List of length D. Item d represents, for each iteration, an array of dimension J*H such that W[j,h]=1 if j belongs to cluster h.
res_mus	List of length D. Item d represents the resulting BOS position parameters mus for group d.
res_ps	List of length D. Item d represents the resulting BOS precision parameters pis for group d.
res_gamma	Vector with the resulting gammas.
res_rho	List of length D. Item d represents the resulting rho for group d.
res_W	List of length D. Item d represents the resulting W for group d.
icl	ICL-BIC result value. Must be maximized.
zr	Vector with resulting row partitions.
zc	List of length D. Item d represents the resulting column partition for group d.
probaW	List of length D. Item d represents the probability for each column to belong to each column-cluster for group d.

Author(s)

Margot Selosse, Julien Jacques, Christophe Biernacki.

Examples

```
## Not run:
  # loading the real dataset
  data("dataqol.classif")

  set.seed(5)

  # loading the ordinal data
  M <- as.matrix(dataqol.classif[,2:31])

  # creating the classes values
  y <- as.vector(dataqol.classif$death)


  # sampling datasets for training and to predict
  nb.sample <- ceiling(nrow(M)*2/3)
  sample.train <- sample(1:nrow(M), nb.sample, replace=FALSE)

  M.train <- M[sample.train,]
```

```
M.validation <- M[-sample.train,]
nb.missing.validation <- length(which(M.validation==0))


y.train <- y[sample.train]
y.validation <- y[-sample.train]




# defining different number of categories:
m=c(4,7)


# defining number of row and column clusters
krow = 2
kcol = c(3,1)

# configuration for the inference
nbSEM=70
nbSEMburn=50
nbindmini=1

d.list <- list(1:28,29:30)

# Co-clustering execution
object <- bosclassifMulti(x=M.validation,y=y.validation,kr=krow,kc=kcol,m=m, to.predict=M.train,
                          d.list=d.list,nbSEM=nbSEM,nbSEMburn=nbSEMburn,
                         nbindmini=nbindmini, init='kmeans',disp=TRUE)


## End(Not run)
```

---

bosclust *bosclust*

Description

This function performs a clustering on ordinal data by using the multiple latent block model (cf references for further details). It allows the user to define D groups of variables that have different number of levels. A BOS distribution is used, and the parameters inference is realized with an SEM-Gbbs algorithm.

Usage

```
bosclust(x, kr, m, nbSEM = 50, nbSEMburn = 20, nbindmini = 4,
  init = "kmeans", disp = TRUE, iterordiEM = 10)
```

Arguments

x	Matrix of ordinal data. The missing values should be equal to 0.
kr	Number of row clusters.
m	Defines the number of levels for the variables.
nbSEM	Number of SEM-Gibbs iterations realized to estimate parameters.
nbSEMburn	Number of SEM-Gibbs burning iterations for estimating parameters. This parameter must be inferior to nbSEM.
nbindmini	Minimum number of cells belonging to a block.
init	String that indicates the kind of initialisation. Must one of these two words : "kmeans" or "random".
disp	Boolean that should be set to TRUE if the user wants the progress bars to be displayed in the console.
iterordiEM	Number of iterations for the internal EM algorithm that estimates the parameters of BOS distribution.

Value

xhat	Represents the dataset, with missing values completed.
mu	Represents the BOS position parameter mus that were estimated at each iteration of the SEM algorithm.
p	Represents the BOS precision parameter pis that were estimated at each iteration of the SE algorithm.
gamma	Vector of length kr indicating the row mixing proportions that were estimated at each iteration of the SE algorithm.
V	For each iteration of the SEM algorithm, array of dimension N*G such that V[i,g]=1 if i belongs to cluster g.
res_mu	Represents the resulting position parameter mu.
res_p	Represents the resulting precision parameter pi.
res_gamma	Vector with the resulting gammas.
res_V	Array with the resulting V.
icl	ICL-BIC result value. Must be maximized.
zr	Vector with resulting row partitions.
probaV	Array representing the probability for each row to belong to each row-cluster.

Author(s)

Margot Selosse, Julien Jacques, Christophe Biernacki.

Examples

```
#\dontshow{
#  data("Msimulated")

#  nbSEM=15
#  nbSEMburn=10
#  nbindmini=1

#  kr=2
#  m=3

#  res <- bosclust(x=Msimulated,kr = kr,m=m,
#                  nbSEM=nbSEM,nbSEMburn=nbSEMburn,
#                  nbindmini=#nbindmini,
#                  disp=FALSE,iterordiEM=5)
#}

  library(ordinalClust)
  data("dataqol")
  set.seed(5)

  # loading the ordinal data
  M <- as.matrix(dataqol[,2:29])

  m=4

  krow = 4

  nbSEM=50
  nbSEMburn=40
  nbindmini=1

  object <- bosclust(x=M, kr=krow, m=m, nbSEM=nbSEM,
      nbindmini=nbindmini, nbSEMburn=nbSEMburn,
      init='kmeans',disp=TRUE)
```

`bosclustMulti` *bosclustMulti*

Description

This function performs a clustering on ordinal data by using the multiple latent block model (cf references for further details). It allows the user to define D groups of variables that have different number of levels. A BOS distribution is used, and the parameters inference is realized with an SEM-Gbbs algorithm.

Usage

```
bosclustMulti(x, kr, m, d.list, nbSEM = 50, nbSEMburn = 20, nbindmini = 4,
    init = "kmeans", disp = TRUE, iterordiEM = 10)
```

Arguments

x	Matrix of ordinal data. The missing values should be equal to 0.
kr	Number of row clusters.
m	Vector of length D. Element d defines the number of levels for the variables of group d.
d.list	List of length D. Item d is a vector with the column indices of the variables of group d.
nbSEM	Number of SEM-Gibbs iterations realized to estimate parameters.
nbSEMburn	Number of SEM-Gibbs burning iterations for estimating parameters. This parameter must be inferior to nbSEM.
nbindmini	Minimum number of cells belonging to a block.
init	String that indicates the kind of initialisation. Must one of these two words : "kmeans" or "random".
disp	Boolean that should be set to TRUE if the user wants the progress bars to be displayed in the console.
iterordiEM	Number of iterations for the internal EM algorithm that estimates the parameters of BOS distribution.

Value

xhat	List of length D. Item d represents the group d in dataset, with missing values completed.
mus	List of length D. Item d represents the mus that were estimated at each iteration for group d.
ps	List of length D. Item d represents the pis that were estimated at each iteration for group d.
gamma	Vector of length kr indicating the row mixing proportions that were estimated at each iteration.
V	For each iteration, array of dimension N*G such that V[i,g]=1 if i belongs to cluster g.
res_mus	List of length D. Item d represents the resulting mus for group d.
res_ps	List of length D. Item d represents the resulting pis for group d.
res_gamma	Vector with the resulting gammas.
res_V	Array with the resulting V.
icl	ICL-BIC result value. Must be maximized.
zr	Vector with resulting row partitions.
probaV	Array representing the probability for each row to belong to each row-cluster.

Author(s)

Margot Selosse, Julien Jacques, Christophe Biernacki.

Examples

```
## Not run:
  # loading the real dataset
  data("dataqol")
  set.seed(5)

  # loading the ordinal data
  M <- as.matrix(dataqol[,2:31])


  # defining different number of categories:
  m=c(4,7)


  # defining number of row and column clusters
  krow = 5

  # configuration for the inference
  nbSEM=70
  nbSEMburn=50
  nbindmini=1

  d.list <- list(1:28,29:30)

  # Co-clustering execution
  object <- bosclustMulti(x=M,kr=krow,m=m, d.list=d.list,
                      nbSEM=nbSEM,nbSEMburn=nbSEMburn,
                       nbindmini=nbindmini, init='kmeans',disp=TRUE)

## End(Not run)
```

`boscoclust` *boscoclust*

Description

This function performs a co-clustering on ordinal data by using the latent block model (cf references for further details). A BOS distribution is used, and the parameters inference is realized with an SEM-Gbbs algorithm.

Usage

```
boscoclust(x,kr,kc,m,nbSEM=50,nbSEMburn=20,nbindmini=4,
  init='kmeans',disp=TRUE,iterordiEM=10)
```

Arguments

x	Matrix of ordinal data. The missing values should be equal to 0.
kr	Number of row clusters.
kc	Number of column clusters.
m	Number of categories for ordinal variables.
nbSEM	Number of SEM-Gibbs iterations realized to estimate parameters.
nbSEMburn	Number of SEM-Gibbs burning iterations for estimating parameters. This parameter must be inferior to nbSEM.
nbindmini	Minimum number of cells belonging to a block.
init	String that indicates the kind of initialisation. Must one of these two words : "kmeans" or "random".
disp	Boolean that should be set to TRUE if the user wants the progress bars to be displayed in the console.
iterordiEM	Number of iterations for the internal EM algorithm that estimates the parameters of BOS distribution.

Value

xhat	dataset completed with missing values completed.
mus	For each iteration, an array of dimension kr*kc that represents the mus that were estimated at each iteration.
ps	For each iteration, an array of dimension kr*kc that represents the pis that were estimated at each iteration.
gamma	Array of gammas (row mixing proportions) that were estimated at each iteration.
rho	Array of rhos (columns mixing proportions) that were estimated at each iteration.
V	For each iteration, array of dimension N*G such that V[i,g]=1 if i belongs to cluster g
W	For each iteration, array of dimension J*H such that W[j,h]=1 if j belongs to cluster h.
res_mus	Array of dimension kr*kc that represents the resulting BOS position parameters mus.
res_ps	Array of dimension kr*kc that represents the resulting BOS precision parameters pis.
res_gamma	Vector with the resulting gamma.
res_rho	Vector with the resulting rho.
res_V	Array with the resulting V.
res_W	Array with the resulting W.
icl	ICL-BIC result value. Must be maximized.
zr	Vector with resulting row partitions.
zc	Vector with resulting column partitions.
probaV	Array representing the probability for each row to belong to each row-cluster.
probaW	Array representing the probability for each column to belong to each column-cluster.

Author(s)

Margot Selosse, Julien Jacques, Christophe Biernacki.

Examples

```
# loading the real dataset
data("dataqol")
set.seed(5)

# loading the ordinal data
M <- as.matrix(dataqol[,2:29])


# defining different number of categories:
m=4


# defining number of row and column clusters
krow = 5
kcol = 4

# configuration for the inference
nbSEM=50
nbSEMburn=40
nbindmini=2

# Co-clustering execution
object <- boscoclust(x=M,kr=krow,kc=kcol,m=m,nbSEM=nbSEM,nbSEMburn=nbSEMburn,
                     nbindmini=nbindmini, init='kmeans',disp=TRUE)
```

`boscoclustMulti` *boscoclustMulti*

Description

This function performs a co-clustering on ordinal data by using the multiple latent block model (cf references for further details). It allows the user to define D groups of variables that shouldn't be clustered together. Particulary, variables with different number of levels must be in different groups. A BOS distribution is used, and the parameters inference is realized with an SEM-Gbbs algorithm.

Usage

```
boscoclustMulti(x, kr, kc, m, d.list, nbSEM = 50, nbSEMburn = 20,
nbindmini = 4, init = "kmeans", disp = TRUE, iterordiEM = 10)
```

Arguments

x	Matrix of ordinal data. The missing values should be equal to 0.
kr	Number of row clusters.
kc	Vector of length D. Element d defines the number of column-cluster for the variables of group d.
m	Vector of length D. Element d defines the number of levels for the variables of group d.
d.list	List of length D. Item d is a vector with the column indices of the variables of group d.
nbSEM	Number of SEM-Gibbs iterations realized to estimate parameters.
nbSEMburn	Number of SEM-Gibbs burning iterations for estimating parameters. This parameter must be inferior to nbSEM.
nbindmini	Minimum number of cells belonging to a block.
init	String that indicates the kind of initialisation. Must one of these two words : "kmeans" or "random".
disp	Boolean that should be set to TRUE if the user wants the progress bars to be displayed in the console.
iterordiEM	Number of iterations for the internal EM algorithm that estimates the parameters of BOS distribution.

Value

xhat	List of length D. Item d represents the group d in dataset, with missing values completed.
mus	List of length D. Item d represents the mus that were estimated at each iteration for group d.
ps	List of length D. Item d represents the pis that were estimated at each iteration for group d.
gamma	Array of gammas that were estimated at each iteration.
rho	List of length D. Item d represents the rho that were estimated at each iteration for group d.
V	For each iteration, array of dimension N*G such that V[i,g]=1 if i belongs to cluster g
W	List of length D. Item d represents, for each iteration, an array of dimension J*H such that W[j,h]=1 if j belongs to cluster h.
res_mus	List of length D. Item d represents the resulting BOS position parameters mus for group d.
res_ps	List of length D. Item d represents the resulting BOS precision parameters pis for group d.
res_gamma	Vector with the resulting gammas.
res_rho	List of length D. Item d represents the resulting rho for group d.
res_V	Array with the resulting V.

res_W	List of length D. Item d represents the resulting W for group d.
icl	ICL-BIC result value. Must be maximized.
zr	Vector with resulting row partitions.
zc	List of length D. Item d represents the resulting column partition for group d.
probaV	Array representing the probability for each row to belong to each row-cluster.
probaW	List of length D. Item d represents the probability for each column to belong to each column-cluster for group d.

Author(s)

Margot Selosse, Julien Jacques, Christophe Biernacki.

Examples

```
## Not run:

  # loading the real dataset
  data("dataqol")
  set.seed(5)

  # loading the ordinal data
  M <- as.matrix(dataqol[,2:31])


  # defining different number of categories:
  m=c(4,7)


  # defining number of row and column clusters
  krow = 5
  kcol = c(4,1)

  # configuration for the inference
  nbSEM=70
  nbSEMburn=50
  nbindmini=1

  d.list <- list(1:28,29:30)

  # Co-clustering execution
  object <- boscoclustMulti(x=M,kr=krow,kc=kcol,m=m, d.list=d.list,
                      nbSEM=nbSEM,nbSEMburn=nbSEMburn,
                       nbindmini=nbindmini, init='kmeans',disp=TRUE)

## End(Not run)
```

bosplot *bosplot*

Description

Plots the result of a classification, clustering or co-clustering that were performed from the following functions: bosclassif,bosclust,boscoclust,bosclassifMulti,bosclustMulti,boscoclustMulti.

Usage

```
bosplot(object)
```

Arguments

object	Result of one of the fuction listed above.

Author(s)

Margot Selosse, Julien Jacques, Christophe Biernacki.

Examples

```
data("dataqol")
set.seed(5)

# loading the ordinal data
M <- as.matrix(dataqol[,2:29])

m=4

krow = 4

nbSEM=50
nbSEMburn=40
nbindmini=1

object <- bosclust(x=M, kr=krow, m=m, nbSEM=nbSEM,
    nbindmini=nbindmini, nbSEMburn=nbSEMburn,
    init='kmeans',disp=TRUE)

bosplot(object)
```

dataqol *Questionnaires Responses Of Patients Affected By Breast Cancer*

Description

This dataset contains the responses of 121 patients to 30 questions about their quality of life.

Usage

```
dataqol
```

Format

A dataframe with 121 lines and 31 columns. A line represents a patient and a column are information about the patients

Id patient Id

q1-q28 responses to 28 questions with number of levels equals to 4

q29-q30 responses to 22 questions with number of levels equals to 7

Source

The table was determined based on data associated with the package available on: https://cran.r-project.org/package=QoLR

dataqol.classif *Questionnaires Responses Of Patients Affected By Breast Cancer*

Description

This dataset contains the responses of 40 patients to 30 questions about their quality of life. Furthermore, a variable indicates if the patient survived from the disease.

Usage

```
dataqol.classif
```

Format

A dataframe with 40 lines and 32 columns. A line represents a patient and a column are information about the patients

Id patient Id

q1-q28 responses to 28 questions with number of levels equals to 4

q29-q30 responses to 22 questions with number of levels equals to 7

death if the patient survived (1) or not (2)

Source

The table was determined based on data associated with the package available on: https://cran.r-project.org/package=QoLR

`Msimulated` *Matrix of simulated ordinal data*

Description

This is a toy dataset for running simple examples.

Usage

```
Msimulated
```

Format

An ordinal data matrix with 60 lines and 50 columns. Number of levels is equal to 3. 4 blocks are simulated with (mu,pi) parameters equal to (3,0.5), (2,0.7), (1,0.8) and (2,0.6).

`pejSim` *pejSim*

Description

This function computes the probabiltiy for a level ej to be sampled from a BOS distribution of parameters (mu,pi), with a number of levels equals to m. Can be used to generate data wth BOS distribution.

Usage

```
pejSim(ej,m,mu,p)
```

Arguments

`ej`	levels to be sampled
`m`	Number of levels.
`mu`	mu parameter for BOS distribution.
`p`	pi parameter for BOS distribution.

Value

return the probabiltiy of ej to be sampled from a BOS distribution of parameters (mu,pi), with a number of levels equals to m.

Author(s)

Margot Selosse, Julien Jacques, Christophe Biernacki.

Examples

```
library(ordinalClust)
data("dataqol")
set.seed(5)

m=7
nr=10000
mu=5
pi=0.5

probaBOS=rep(0,m)
for (im in 1:m) probaBOS[im]=pejSim(im,m,mu,pi)
M <- sample(1:m,nr,prob = probaBOS, replace=TRUE)
```

Index