Package ‘bikm1’

May 7, 2026

Type Package

Title Co-Clustering Adjusted Rand Index and Bikm1 Procedure for Contingency and Binary Data-Sets

Version 1.1.0

Maintainer Valerie Robert <valerie.robert.math@gmail.com>

Description Co-clustering of the rows and columns of a contingency or binary matrix, or double binary matrices and model selection for the number of row and column clusters. Three models are considered: the Poisson latent block model for contingency matrix, the binary latent block model for binary matrix and a new model we develop: the multiple latent block model for double binary matrices. A new procedure named bikm1 is implemented to investigate more efficiently the grid of numbers of clusters. Then, the studied model selection criteria are the integrated completed likelihood (ICL) and the Bayesian integrated likelihood (BIC). Finally, the co-clustering adjusted Rand index (CARI) to measure agreement between co-clustering partitions is implemented. Robert Valerie, Vasseur Yann, Brault Vincent (2021) <doi:10.1007/s00357-020-09379-w>.

Imports gtools, stats, graphics, grDevices, methods, parallel, ade4, pracma, ggplot2, reshape2, grid, lpSolve

License GPL-2

Encoding UTF-8

RoxygenNote 7.1.1

NeedsCompilation no

Author Valerie Robert [aut, cre]

Repository CRAN

Date/Publication 2021-07-16 07:30:20 UTC

Contents

`bikm1-package` *bikm1 package*

Description

This package is designed to co-cluster a contingency (resp. binary) matrix, or double binary matrices in blocks respectively under the (normalized or not) Poisson (resp binary) Latent Block Model and the Multiple Latent Block Model. It enables to automatically select the number of row and column clusters and to compare partition estimations with reference partitions.

Features

Package for the segmentation of the rows and columns inducing a co-clustering and automatically select the number of row and column clusters.

Model 1

`BIKM1_LBM_Poisson` . This fitting procedure produces a `BIKM1_LBM_Poisson` object.

Model 2

`BIKM1_LBM_Binary` . This fitting procedure produces a `BIKM1_LBM_Binary` object.

Model 3

`BIKM1_MLBM_Binary` . This fitting procedure produces a `BIKM1_MLBM_Binary` object.

Technical remarks

Display of the result with `plot,BIKM1_LBM_Poisson-method` and

with `show,BIKM1_LBM_Poisson-method`, with `summary,BIKM1_LBM_Poisson-method` and with `print,BIKM1_LBM_Poisson-method`.

Display of the result with `plot,BIKM1_LBM_Binary-method` and

with `show,BIKM1_LBM_Binary-method`, with `summary,BIKM1_LBM_Binary-method` and with `print,BIKM1_LBM_Binary-m`

Display of the result with `plot,BIKM1_MLBM_Binary-method` and

with `show,BIKM1_MLBM_Binary-method`, with `summary,BIKM1_MLBM_Binary-method` and with `print,BIKM1_MLBM_Binary-method`.

Author(s)

Valerie Robert <valerie.robert.math@gmail.com>

References

Keribin, Celeux and Robert, The Latent Block Model: a useful model for high dimensional data. https://hal.inria.fr/hal-01658589/document

Govaert and Nadif. Co-clustering, Wyley (2013).

Keribin, Brault and Celeux. Estimation and Selection for the Latent Block Model on Categorical Data, Statistics and Computing (2014).

Robert. Classification croisee pour l'analyse de bases de donnees de grandes dimensions de pharmacovigilance. Thesis, Paris Saclay (2017).

Robert, Vasseur and Brault. Comparing high dimensional partitions with the Co-clustering Adjusted Rand Index, Journal of Classification, 38(1), 158-186 (2021).

ARI *ARI function for agreement between two partitions*

Description

Produce a measure of agreement between two partitions. A value of 1 means a perfect match.

Usage

```
ARI(v,vprime)
```

Arguments

v	numeric vector specifying the class of observations.
vprime	numeric vector specifying another partitions of observations.

Value

a list including the arguments:

`ari`: value of the index.

`nv`: contingency table which the index is based on.

References

Hubert and Arabie. Comparing partitions. Journal of classification (1985).

Examples

```
require(bikm1)
J=200
K=120
h=3
l=2
theta=list()
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$gamma_hl=matrix(c(1, 6,4, 1, 7, 1),ncol=2)
data=PoissonBlocRnd(J,K,theta)
res=BIKM1_LBM_Poisson(data$x,4,4,4,init_choice='random')
mv=ARI(res@model_max$v, data$xrow)
mv$ari
mv$nv
mw=ARI(res@model_max$w, data$xcol)
```

BIKM1_LBM_Binary *BIKM1_LBM_Binary fitting procedure*

Description

Produce a blockwise estimation of a contingency matrix of observations.

Usage

```
BIKM1_LBM_Binary(x,Gmax,Hmax,a=4,b=1,
Gstart=2,Hstart=2,init_choice='smallVBayes',userparam=NULL,
ntry=50,criterion_choice='ICL', mc.cores=1,verbose=TRUE)
```

Arguments

`x`	binary matrix of observations.
`Gmax`	a positive integer less than number of rows.
`Hmax`	a positive integer less than number of columns.The bikm1 procedure stops while the numbers of rows is higher than Gmax or the number of columns is higher than Hmax.
`a`	hyperparameter used in the VBayes algorithm for priors on the mixing proportions. By default, a=4.
`b`	hyperparameter used in the VBayes algorithm for prior on the Bernoulli parameter. By default, b=1.
`Gstart`	a positive integer to initialize the procedure with number of row clusters. By default, Gstart=2.
`Hstart`	a positive integer to initialize the procedure with number of column clusters. By default, Hstart=2.
`init_choice`	a character string corresponding to the chosen initialization strategy used for the procedure, which can be "random" or "smallVBayes" or "user". By default, init_choice="smallVBayes".
`userparam`	in the case where init_choice is "user", a list containing partitions z and w. By default userparam=NULL.
`ntry`	a positive integer corresponding to the number of times which is launched the small VBayes or random initialization strategy. By default ntry=100.
`criterion_choice`	a character string corresponding to the chosen criterion used for model selection, which can be "ICL" as for now. By default, criterion_choice="ICL".
`mc.cores`	a positive integer corresponding to the available number of cores for parallel computing. By default, mc.cores=1.
`verbose`	logical. To display each step and the result. By default verbose=TRUE.

Value

a BIKM1_LBM_Binary object including

`model_max`: the selected model by the procedure with free energy W, theta, conditional probabilities (s_ig, r_jh), iter, empty_cluster, and the selected partitions z and w.

`criterion_choice`: the chosen criterion

`init_choice`: the chosen init choice

`criterion tab`: the matrix containing the criterion values for each selected number of row and column

`W_tab`: the matrix containing the free energy values for each selected number of row and column

`criterion_max`: the maximum of the criterion values

`gopt`: the selected number of rows

`hopt`: the selected number of columns

References

Govaert and Nadif. Co-clustering, Wyley (2013).

Keribin, Brault and Celeux. Estimation and Selection for the Latent Block Model on Categorical Data, Statistics and Computing (2014).

Robert. Classification crois\'ee pour l'analyse de bases de donn\'ees de grandes dimensions de pharmacovigilance. Paris Saclay (2017).

Examples

```
require(bikm1)
set.seed(42)
n=200
J=120
g=3
h=2
theta=list()
theta$pi_g=1/g *matrix(1,g,1)
theta$rho_h=1/h *matrix(1,h,1)
theta$alpha_gh=matrix(runif(6),ncol=h)
data=BinBlocRnd_LBM(n,J,theta)
res=BIKM1_LBM_Binary(data$x,3,2,Gstart=3,Hstart=2,
init_choice='user',userparam=list(z=data$xrow,v=data$xcol))
```

`BIKM1_LBM_Binary-class`

Class "BIKM1_LBM_Binary"

Description

Class of object returned by the `BIKM1_LBM_Binary` function.

Slots

`model_max`: The selected model by the procedure with free energy W, theta, conditional probabilities (s_ig, r_jh), iter, empty_cluster, and the selected partitions z and v.

`criterion_choice`: A character string corresponding to the chosen criterion used for model selection, which can be "ICL" or "BIC".

`init_choice`: A character string corresponding to the chosen initialization strategy used for the procedure, which can be "random" or "Gibbs" or "smallVBayes".

`criterion_tab`: The matrix corresponding to the values of the chosen criterion for pairs of numbers of clusters visited by the BIKM1_LBM_Binary function. The matrix rows design the numbers of row clusters. If a pair is not visited, by default, the value is -Inf.

`W_tab`: The matrix corresponding to the values of the free energy (minimizer of the loglikelihood in the algorithm) for pairs of numbers of clusters visited by the procedure. The matrix rows design the numbers of row clusters. If a pair is not visited, by default, the value is -Inf.

`criterion_max`: Numeric indicating the maximum of the criterion values, calculated on the pairs of numbers of clusters visited by the BIKM1_LBM_Binary function.

`gopt`: An integer value indicating the number of row clusters selected by the BIKM1_LBM_Binary function.

`hopt`: An integer value indicating the number of column clusters selected by the BIKM1_LBM_Binary function.

Examples

```
require(bikm1)
n=200
J=120
g=3
h=2
theta=list()
theta$pi_g=1/g *matrix(1,g,1)
theta$rho_h=1/h *matrix(1,h,1)
theta$alpha_gh=matrix(runif(6),ncol=h)
data=BinBlocRnd_LBM(n,J,theta)
res=BIKM1_LBM_Binary(data$x,3,3,a=4,init_choice='smallVBayes')
```

`BIKM1_LBM_Poisson` *BIKM1_LBM_Poisson fitting procedure*

Description

Produce a blockwise estimation of a contingency matrix of observations.

Usage

```
BIKM1_LBM_Poisson(x,Hmax,Lmax,a=4,alpha=1,beta=0.01,
Hstart=2,Lstart=2,normalization=FALSE,init_choice='smallVBayes',
userparam=NULL,ntry=50,criterion_choice='ICL', mc.cores=1,verbose=TRUE)
```

Arguments

`x`	contingency matrix of observations.
`Hmax`	a positive integer less than number of rows.
`Lmax`	a positive integer less than number of columns.The bikm1 procedure stops while the numbers of rows is higher than Hmax or the number of columns is higher than Lmax.
`a`	hyperparameter used in the VBayes algorithm for priors on the mixing proportions. By default, a=4.
`alpha`	hyperparameter used in the VBayes algorithm for prior on the Poisson parameter. By default, alpha=1.
`beta`	hyperparameter used in the VBayes algorithm for prior on the Poisson parameter. By default, beta=0.01.
`Hstart`	a positive integer to initialize the procedure with number of row clusters. By default, Hstart=2.
`Lstart`	a positive integer to initialize the procedure with number of column clusters. By default, Lstart=2.
`normalization`	logical. To use the normalized Poisson modelling in the Latent Block Model. By default normalization=FALSE.
`init_choice`	character string corresponding to the chosen initialization strategy used for the procedure, which can be "random" or "Gibbs" (higher time computation) or "smallVBayes" or "user". By default, init_choice="smallVBayes"
`userparam`	In the case where init_choice is "user", a list containing partitions v and w.
`ntry`	a positive integer corresponding to the number of times which is launched the small VBayes or random initialization strategy. By default ntry=50.
`criterion_choice`	Character string corresponding to the chosen criterion used for model selection, which can be "ICL" or "BIC". By default, criterion_choice="ICL".
`mc.cores`	a positive integer corresponding to the available number of cores for parallel computing. By default, mc.cores=1.
`verbose`	logical. To display each step and the result. By default verbose=TRUE.

Value

a BIKM1_LBM_Poisson object including

`model_max`: the selected model by the procedure with free energy W, theta, conditional probabilities (r_jh, t_kl), iter, empty_cluster, and the selected partitions v and w.

`criterion_choice`: the chosen criterion

`init_choice`: the chosen init choice

`criterion tab`: matrix containing the criterion values for each selected number of row and column

`W_tab`: matrix containing the free energy values for each selected number of row and column

`criterion_max`: maximum of the criterion values

`hopt`: the selected number of rows

`lopt`: the selected number of columns

References

Keribin, Celeux and Robert, The Latent Block Model: a useful model for high dimensional data. https://hal.inria.fr/hal-01658589/document

Govaert and Nadif. Co-clustering, Wyley (2013).

Keribin, Brault and Celeux. Estimation and Selection for the Latent Block Model on Categorical Data, Statistics and Computing (2014).

Robert. Classification crois\'ee pour l'analyse de bases de donn\'ees de grandes dimensions de pharmacovigilance. Paris Saclay (2017).

Examples

```
require(bikm1)
J=200
K=120
h=3
l=2
theta=list()
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$gamma_hl=matrix(c(1, 6,4, 1, 7, 1),ncol=2)
data=PoissonBlocRnd(J,K,theta)
res=BIKM1_LBM_Poisson(data$x,3,2,Hstart=3,Lstart=2,
init_choice='user',userparam=list(v=data$xrow,w=data$xcol))
```

BIKM1_LBM_Poisson-class

Class "BIKM1_LBM_Poisson"

Description

Class of object returned by the BIKM1_LBM_Poisson function.

Slots

`model_max`: The selected model by the procedure with free energy W, theta, conditional probabilities (r_jh, t_kl), iter, empty_cluster, and the selected partitions v and w.

`criterion_choice`: A character string corresponding to the chosen criterion used for model selection, which can be "ICL" or "BIC".

`init_choice`: A character string corresponding to the chosen initialization strategy used for the procedure, which can be "random" or "Gibbs" or "smallVBayes".

`criterion_tab`: The matrix corresponding to the values of the chosen criterion for pairs of numbers of clusters visited by the BIKM1_LBM_Poisson function. The matrix rows design the numbers of row clusters. If a pair is not visited, by default, the value is -Inf.

`W_tab`: The matrix corresponding to the values of the free energy (minimizer of the loglikelihood in the algorithm) for pairs of numbers of clusters visited by the procedure. The matrix rows design the numbers of row clusters. If a pair is not visited, by default, the value is -Inf.

`criterion_max`: Numeric indicating the maximum of the criterion values, calculated on the pairs of numbers of clusters visited by the BIKM1_LBM_Poisson function.

`lopt`: An Integer value indicating the number of row clusters selected by the BIKM1_LBM_Poisson function.

`hopt`: An integer value indicating the number of column clusters selected by the BIKM1_LBM_Poisson function.

Examples

```
require(bikm1)
set.seed(42)
J=200
K=120
h=3
l=2
theta=list()
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$gamma_hl=matrix(floor(runif(h*l)*20+1),ncol=l)
data=PoissonBlocRnd(J,K,theta)
res=BIKM1_LBM_Poisson(data$x,3,3,4,init_choice='smallVBayes')
```

`BIKM1_MLBM_Binary` *BIKM1_MLBM_Binary fitting procedure*

Description

Produce a blockwise estimation of double matrices of observations.

Usage

```
BIKM1_MLBM_Binary(x,y,Gmax,Hmax,Lmax,a=4,b=1,
Gstart=2,Hstart=2,Lstart=2,init_choice='smallVBayes',userparam=NULL,
ntry=50,criterion_choice='ICL', mc.cores=1,verbose=TRUE)
```

Arguments

`x`	matrix of observations (1rst matrix).
`y`	matrix of observations (2nd matrix).
`Gmax`	a positive integer less than number of rows.
`Hmax`	a positive integer less than number of columns of the 1st matrix.

Lmax	a positive integer less than number of columns of the 2nd matrix. The bikm1 procedure stops while the numbers of rows is higher than Gmax or the number of columns is higher than Hmax or the numbers of columns(2nd matrix) is higher than Lmax.
a	hyperparameter used in the VBayes algorithm for priors on the mixing proportions. By default, a=4.
b	hyperparameter used in the VBayes algorithm for prior on the Bernoulli parameter. By default, b=1.
Gstart	a positive integer to initialize the procedure with number of row clusters. By default, Gstart=2.
Hstart	a positive integer to initialize the procedure with number of column clusters. By default, Hstart=2.
Lstart	a positive integer to initialize the procedure with number of column clusters. By default, Lstart=2.
init_choice	character string corresponding to the chosen initialization strategy used for the procedure, which can be "random" or "smallVBayes" or "user". By default, init_choice="smallVBayes".
userparam	In the case where init_choice is "user", a list containing partitions z,v and w.
ntry	a positive integer corresponding to the number of times which is launched the small VBayes initialization strategy. By default ntry=100.
criterion_choice	Character string corresponding to the chosen criterion used for model selection, which can be "ICL" as for now. By default, criterion_choice="ICL".
mc.cores	a positive integer corresponding to the available number of cores for parallel computing. By default, mc.cores=1.
verbose	logical. To display each step and the result. By default verbose=TRUE.

Value

a BIKM1_MLBM_Binary object including

model_max: the selected model by the procedure including free energy W, theta, conditional probabilities (s_ig, r_jh,t_kl), iter, empty_cluster, and the selected partitions z,v and w.

criterion_choice: the chosen criterion

init_choice: the chosen init_choice

criterion_tab: matrix containing the criterion values for each selected number of row and column

W_tab: matrix containing the free energy values for each selected number of row and column

criterion_max: maximum of the criterion values

gopt: the selected number of rows

hopt: the selected number of columns (1rst matrix)

lopt: the selected number of columns (2nd matrix)

References

Govaert and Nadif. Co-clustering, Wyley (2013).

Keribin, Brault and Celeux. Estimation and Selection for the Latent Block Model on Categorical Data, Statistics and Computing (2014).

Robert. Classification crois\'ee pour l'analyse de bases de donn\'ees de grandes dimensions de pharmacovigilance. Paris Saclay (2017).

Examples

```
require(bikm1)
set.seed(42)
n=200
J=120
K=120
g=3
h=2
l=2
theta=list()
theta$pi_g=1/g *matrix(1,g,1)
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$alpha_gh=matrix(runif(6),ncol=h)
theta$beta_gl=matrix(runif(6),ncol=l)
data=BinBlocRnd_MLBM(n,J,K,theta)
res=BIKM1_MLBM_Binary(data$x,data$y,3,2,2,Gstart=3,Hstart=2,Lstart=2,init_choice='user',
userparam=list(z=data$xrow,v=data$xcolx,w=data$xcoly))
```

BIKM1_MLBM_Binary-class

Class "BIKM1_MLBM_Binary"

Description

Class of object returned by the BIKM1_MLBM_Binary function.

Slots

model_max: The selected model by the procedure with free energy W, theta, conditional probabilities (s_ig, r_jh, t_kl), iter, empty_cluster, and the selected partitions z, v and w.

criterion_choice: A character string corresponding to the chosen criterion used for model selection, which can be "ICL" or "BIC".

init_choice: A character string corresponding to the chosen initialization strategy used for the procedure, which can be "random" or "Gibbs" or "smallVBayes".

`criterion_tab`: The matrix corresponding to the values of the chosen criterion for pairs of numbers of clusters visited by the BIKM1_MLBM_Binary function. The matrix rows design the numbers of row clusters. If a pair is not visited, by default, the value is -Inf.

`W_tab`: The matrix corresponding to the values of the free energy (minimizer of the loglikelihood in the algorithm) for pairs of numbers of clusters visited by the procedure. The matrix rows design the numbers of row clusters. If a pair is not visited, by default, the value is -Inf.

`criterion_max`: Numeric indicating the maximum of the criterion values, calculated on the pairs of numbers of clusters visited by the BIKM1_MLBM_Binary function.

`gopt`: An integer value indicating the number of row clusters selected by the BIKM1_MLBM_Binary function.

`hopt`: An integer value indicating the number of column clusters for the first matrix selected by the BIKM1_MLBM_Binary function.

`lopt`: An integer value indicating the number of row clusters for the second matrix selected by the BIKM1_MLBM_Binary function.

Examples

```
require(bikm1)
n=200
J=120
K=120
g=3
h=2
l=2
theta=list()
theta$pi_g=1/g *matrix(1,g,1)
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$alpha_gh=matrix(runif(6),ncol=h)
theta$beta_gl=matrix(runif(6),ncol=l)
data=BinBlocRnd_MLBM(n,J,K,theta)
res=BIKM1_MLBM_Binary(data$x,data$y,3,3,3,4,init_choice='smallVBayes')
```

`BinBlocICL_LBM` *BinBlocICL_LBM function for computation of the ICL criterion in the Binary LBM*

Description

Produce a value of the ICL criterion in the Binary LBM.

Usage

```
BinBlocICL_LBM(a,b,x,z1,v1)
```

Arguments

a	an hyperparameter for priors on the mixing proportions. By default, a=4.
b	an hyperparameter for prior on the Bernoulli parameter. By default, b=1.
x	contingency matrix of observations.
z1	a numeric vector specifying the class of rows.
v1	a numeric vector specifying the class of columns.

Value

a value of the ICL criterion.

Examples

```
require(bikm1)
set.seed(42)
n=200
J=120
g=3
h=2
theta=list()
theta$pi_g=1/g *matrix(1,g,1)
theta$rho_h=1/h *matrix(1,h,1)
theta$alpha_gh=matrix(runif(6),ncol=h)
data=BinBlocRnd_LBM(n,J,theta)
BinBlocICL_LBM(a=4,b=1,data$x, data$xrow,data$xcol)
```

BinBlocICL_MLBM — *BinBlocICL_MLBM function for computation of the ICL criterion in the MLBM*

Description

Produce a plot object representing the resumed co-clustered data-sets.

Usage

```
BinBlocICL_MLBM(a,b,x,y,z1,v1,w1)
```

Arguments

a	an hyperparameter for priors on the mixing proportions. By default, a=4.
b	an hyperparameter for prior on the Bernoulli parameter. By default, b=1.
x	binary matrix of observations (1rst matrix).
y	binary matrix of observations (2nd matrix).
z1	a numeric vector specifying the class of rows.
v1	a numeric vector specifying the class of columns (1rst matrix).
w1	a numeric vector specifying the class of columns (2nd matrix).

Value

a value of the ICL criterion.

Examples

```
require(bikm1)
set.seed(42)
n=200
J=120
K=120
g=2
h=2
l=2
theta=list()
theta$pi_g=1/g *matrix(1,g,1)
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$alpha_gh=matrix(runif(4),ncol=h)
theta$beta_gl=matrix(runif(4),ncol=l)
data=BinBlocRnd_MLBM(n,J,K,theta)
res=BIKM1_MLBM_Binary(data$x,data$y,2,2,2,4,init_choice='smallVBayes')
BinBlocICL_MLBM(a=4,b=1,data$x,data$y, data$xrow,data$xcolx,data$xcoly)
```

BinBlocRnd_LBM — *BinBlocRnd_LBM function for binary data matrix simulation*

Description

Produce a data matrix generated under the Binary Latent Block Model.

Usage

```
BinBlocRnd_LBM(n,J,theta)
```

Arguments

n	a positive integer specifying the number of expected rows.
J	a positive integer specifying the number of expected columns.
theta	a list specifying the model parameters: pi_g: a vector specifying the row mixing proportions. rho_h: a vector specifying the matrix column mixing proportions. alpha_gh: a matrix specifying the distribution parameter of the matrix.

Value

a list including the arguments:

x: simulated data matrix.

xrow: numeric vector specifying row partition.

xcol: numeric vector specifying column partition.

Examples

```
require(bikm1)
set.seed(42)
n=200
J=120
g=3
h=2
theta=list()
theta$pi_g=1/g *matrix(1,g,1)
theta$rho_h=1/h *matrix(1,h,1)
theta$alpha_gh=matrix(runif(6),ncol=h)
data=BinBlocRnd_LBM(n,J,theta)
```

`BinBlocRnd_MLBM` *BinBlocRnd_MLBM function for binary double data matrix simulation*

Description

Produce two simulated data matrices generated under the Binary Multiple Latent Block Model.

Usage

```
BinBlocRnd_MLBM(n,J,K,theta)
```

Arguments

`n`	a positive integer specifying the number of expected rows.
`J`	a positive integer specifying the number of expected columns of the first matrix.
`K`	a positive integer specifying the number of expected columns of the second matrix.
`theta`	a list specifying the model parameters: `pi_g`: a vector specifying the row mixing proportions. `rho_h`: a vector specifying the first matrix column mixing proportions. `tau_l`: a vector specifying the second matrix column mixing proportions. `alpha_gh`: a matrix specifying the distribution parameter of the first matrix. `beta_gl`: a matrix specifying the distribution parameter of the second matrix.

Value

a list including the arguments:

`x`: simulated first data matrix. `y`: simulated second data matrix.

`xrow`: numeric vector specifying row partition.

`xcolx`: numeric vector specifying first matrix column partition.

`xcoly`: numeric vector specifying second matrix column partition.

Examples

```
require(bikm1)
set.seed(42)
n=200
J=120
K=120
g=3
h=2
l=2
theta=list()
theta$pi_g=1/g *matrix(1,g,1)
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$alpha_gh=matrix(runif(6),ncol=h)
theta$beta_gl=matrix(runif(6),ncol=l)
data=BinBlocRnd_MLBM(n,J,K,theta)
```

`BinBlocVisuResum_LBM` *BinBlocVisuResum_LBM function for visualization of binary matrix data-sets*

Description

Produce a plot object representing the resumed co-clustered data-sets.

Usage

```
BinBlocVisuResum_LBM(x,z,v)
```

Arguments

`x`	binary matrix of observations.
`z`	a numeric vector specifying the class of rows.
`v`	a numeric vector specifying the class of columns.

Value

a **plot** object.

Examples

```
require(bikm1)
set.seed(42)
n=200
J=120
g=3
h=2
```

```
theta=list()
theta$pi_g=1/g *matrix(1,g,1)
theta$rho_h=1/h *matrix(1,h,1)
theta$alpha_gh=matrix(runif(6),ncol=h)
data=BinBlocRnd_LBM(n,J,theta)
BinBlocVisuResum_LBM(data$x,data$xrow,data$xcol)
```

BinBlocVisuResum_MLBM *BinBlocVisuResum_MLBM function for visualization of double matrix datasets*

Description

Produce a plot object representing the resumed co-clustered data-sets.

Usage

```
BinBlocVisuResum_MLBM(x,y,z,v,w)
```

Arguments

x	binary matrix of observations.
y	binary second matrix of observations.
z	a numeric vector specifying the class of rows.
v	a numeric vector specifying the class of columns (1rst matrix).
w	a numeric vector specifying the class of columns (2nd matrix).

Value

a **plot** object.

Examples

```
require(bikm1)
set.seed(42)
n=200
J=120
K=120
g=3
h=2
l=2
theta=list()
theta$pi_g=1/g *matrix(1,g,1)
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$alpha_gh=matrix(runif(6),ncol=h)
theta$beta_gl=matrix(runif(6),ncol=l)
data=BinBlocRnd_MLBM(n,J,K,theta)
BinBlocVisuResum_MLBM(data$x,data$y, data$xrow,data$xcolx,data$xcoly)
```

BinBlocVisu_LBM *BinBlocVisu_LBM function for visualization of binary matrix datasets*

Description

Produce a plot object representing the co-clustered data-sets.

Usage

```
BinBlocVisu_LBM(x,z,v)
```

Arguments

x	data matrix of observations.
z	a numeric vector specifying the class of rows.
v	a numeric vector specifying the class of columns.

Value

a **plot** object

Examples

```
require(bikm1)
set.seed(42)
n=200
J=120
g=3
h=2
theta=list()
theta$pi_g=1/g *matrix(1,g,1)
theta$rho_h=1/h *matrix(1,h,1)
theta$alpha_gh=matrix(runif(6),ncol=h)
data=BinBlocRnd_LBM(n,J,theta)
BinBlocVisu_LBM(data$x,data$xrow,data$xcol)
```

BinBlocVisu_MLBM *BinBlocVisu_MLBM function for visualization of double matrix datasets*

Description

Produce a plot object representing the co-clustered data-sets.

Usage

```
BinBlocVisu_MLBM(x,y,z,v,w)
```

Arguments

x	first data matrix of observations.
y	second data matrix of observations.
z	a numeric vector specifying the class of rows.
v	a numeric vector specifying the class of columns (1rst matrix).
w	a numeric vector specifying the class of columns (2nd matrix).

Value

a **plot** object

Examples

```
require(bikm1)
set.seed(42)
n=200
J=120
K=120
g=3
h=2
l=2
theta=list()
theta$pi_g=1/g *matrix(1,g,1)
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$alpha_gh=matrix(runif(6),ncol=h)
theta$beta_gl=matrix(runif(6),ncol=l)
data=BinBlocRnd_MLBM(n,J,K,theta)
BinBlocVisu_MLBM(data$x,data$y, data$xrow,data$xcolx,data$xcoly)
```

CARI — *CARI function for agreement between co-clustering partitions*

Description

Produce a measure of agreement between two pairs of partitions for co-clustering. A value of 1 means a perfect match.

Usage

```
CARI(v,w,vprime,wprime)
```

Arguments

v	numeric vector specifying the class of rows.
w	numeric vector specifying the class of columns.
vprime	numeric vector specifying another partition of rows.
wprime	numeric vector specifying another partition of columns.

Value

a list including the arguments:

cari: value of the index (between 0 and 1). A value of 1 corresponds to a perfect match.

nvw: contingency table which the index is based on.

References

Robert, Vasseur and Brault. Comparing high dimensional partitions with the Co-clustering Adjusted Rand Index, Journal of classification 38 (1), 158-186 (2021).

Examples

```
require(bikm1)
J=200
K=120
h=3
l=2
theta=list()
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$gamma_hl=matrix(c(1, 6,4, 1, 7, 1),ncol=2)
data=PoissonBlocRnd(J,K,theta)
res=BIKM1_LBM_Poisson(data$x,4,4,4,init_choice='smallVBayes')
me=CARI(res@model_max$v,res@model_max$w, data$xrow,data$xcol)
me$cari
me$nvw
```

CE_LBM *CE_LBM function for agreement between co-clustering partitions*

Description

Produce a measure of agreement between two pairs of partitions for co-clustering using CE_simple on columns and rows of a matrix. A value of 1 means a perfect match.

Usage

```
CE_LBM(v,w,vprime,wprime)
```

Arguments

v	numeric vector specifying the class of rows.
w	numeric vector specifying the class of columns.
vprime	numeric vector specifying another partition of rows.
wprime	numeric vector specifying another partition of columns.

Value

ce_vw: the value of the index (between 0 and 1). A value of 0 corresponds to a perfect match.

Examples

```
require(bikm1)
set.seed(42)
v=floor(runif(4)*2)
vprime=floor(runif(4)*2)
w=floor(runif(4)*3)
wprime=floor(runif(4)*3)
error=CE_LBM(v,w,vprime,wprime)
```

CE_MLBM *CE_MLBM function for agreement between co-clustering partitions in the MBLM*

Description

Produce a measure of agreement between two triplets of partitions for co-clustering. A value of 1 means a perfect match.

Usage

```
CE_MLBM(z,v,w,zprime,vprime,wprime)
```

Arguments

z	numeric vector specifying the class of rows.
v	numeric vector specifying the class of column partitions for the first matrix.
w	numeric vector specifying the class of column partitions for the second matrix.
zprime	numeric vector specifying another partitions of rows.
vprime	numeric vector specifying another partition of columns for the first matrix.
wprime	numeric vector specifying another partition of columns for the second matrix.

Value

the value of the index (between 0 and 1). A value of 0 corresponds to a perfect match.

Examples

```
require(bikm1)
set.seed(42)
n=200
J=120
K=120
g=2
h=2
l=2
theta=list()
theta$pi_g=1/g *matrix(1,g,1)
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$alpha_gh=matrix(runif(4),ncol=h)
theta$beta_gl=matrix(runif(4),ncol=l)
data=BinBlocRnd_MLBM(n,J,K,theta)
res=BIKM1_MLBM_Binary(data$x,data$y,2,2,2,4,init_choice='smallVBayes')
error=CE_MLBM(res@model_max$z,res@model_max$v,res@model_max$w,data$xrow,data$xcolx,data$xcoly)
```

CE_simple — *CE_simple function for agreement between clustering partitions*

Description

Produce a measure of agreement between two partitions for clustering. A value of 1 means a perfect match.

Usage

```
CE_simple(v,vprime)
```

Arguments

v	numeric vector specifying the class of rows.
vprime	numeric vector specifying the class of rows.

Value

the value of the index.

Examples

```
require(bikm1)
set.seed(42)
v=floor(runif(4)*3)
vprime=floor(runif(4)*3)
error=CE_simple(v,vprime)
error
```

CoNMI *CoNMI function for agreement between co-clustering partitions*

Description

Produce a measure of agreement between two pairs of partitions for co-clustering. A value of 1 means a perfect match.

Usage

```
CoNMI(v,w,vprime,wprime)
```

Arguments

v	numeric vector specifying the class of rows.
w	numeric vector specifying the class of columns.
vprime	numeric vector specifying another partition of rows.
wprime	numeric vector specifying another partition of columns.

Value

the value of the index.

References

Robert, Vasseur and Brault. Comparing high dimensional partitions with the Co-clustering Adjusted Rand Index, Journal of Classification (2021).

Examples

```
require(bikm1)
J=200
K=120
h=3
l=2
theta=list()
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$gamma_hl=matrix(c(1, 6,4, 1, 7, 1),ncol=2)
data=PoissonBlocRnd(J,K,theta)
res=BIKM1_LBM_Poisson(data$x,4,4,4,init_choice='smallVBayes')
me=CoNMI(res@model_max$v,res@model_max$w, data$xrow,data$xcol)
me
```

ENMI *ENMI function for agreement between co-clustering partitions*

Description

Produce a measure of agreement between two pairs of partitions for co-clustering. A value of 1 means a perfect match.

Usage

```
ENMI(v,w,vprime,wprime)
```

Arguments

v	numeric vector specifying the class of rows.
w	numeric vector specifying the class of columns.
vprime	numeric vector specifying another partition of rows.
wprime	numeric vector specifying another partition of columns.

Value

the value of the index.

References

Robert, Vasseur and Brault. Comparing high dimensional partitions with the Co-clustering Adjusted Rand Index, Journal of Classification (2021).

Examples

```
require(bikm1)
J=200
K=120
h=3
l=2
theta=list()
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$gamma_hl=matrix(c(1, 6,4, 1, 7, 1),ncol=2)
data=PoissonBlocRnd(J,K,theta)
res=BIKM1_LBM_Poisson(data$x,4,4,4,init_choice='smallVBayes')
me=ENMI(res@model_max$v,res@model_max$w, data$xrow,data$xcol)
me
```

MI_simple *MI_simple function for agreement between two partitions*

Description

Produce a measure of agreement between two partitions.(between 0 and 1). A value of 1 corresponds to a perfect match.

Usage

```
MI_simple(v,vprime)
```

Arguments

v	numeric vector specifying the class of observations.
vprime	numeric vector specifying another partitions of observations.

Value

the value of the index.

References

Robert, Vasseur and Brault. Comparing high-dimensional partitions with the Co-clustering Adjusted Rand Index. Journal of Classification (2021).

Examples

```
require(bikm1)
J=200
K=120
h=3
l=2
theta=list()
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$gamma_hl=matrix(c(1, 6,4, 1, 7, 1),ncol=2)
data=PoissonBlocRnd(J,K,theta)
res=BIKM1_LBM_Poisson(data$x,4,4,4,init_choice='random')
mi=MI_simple(res@model_max$v, data$xrow)
mi
mw=MI_simple(res@model_max$w, data$xcol)
```

`NCE_LBM` *NCE_LBM function for agreement between co-clustering partitions using NCE_simple*

Description

Produce a measure of agreement between two pairs of partitions for co-clustering. A value of 1 means a perfect match.

Usage

```
NCE_LBM(v,w,vprime,wprime)
```

Arguments

`v`	numeric vector specifying the class of rows.
`w`	numeric vector specifying the class of columns.
`vprime`	numeric vector specifying another partition of rows.
`wprime`	numeric vector specifying another partition of columns.

Value

the value of the index.

Examples

```
require(bikm1)
set.seed(42)
v=floor(runif(4)*2)
vprime=floor(runif(4)*2)
w=floor(runif(4)*3)
wprime=floor(runif(4)*3)
error=NCE_LBM(v,w,vprime,wprime)
```

`NCE_simple` *NCE_simple function for agreement between clustering partitions*

Description

Produce a measure of agreement between two partitions for clustering. A value of 1 means a perfect match. It's the normalized version of CE_simple.

Usage

```
NCE_simple(v,vprime)
```

Arguments

v	numeric vector specifying the class of rows.
vprime	numeric vector specifying the class of rows.

Value

the value of the index. A value of 0 means a perfect match.

Examples

```
require(bikm1)
set.seed(42)
v=floor(runif(4)*3)
vprime=floor(runif(4)*3)
error=NCE_simple(v,vprime)
error
```

---

plot,BIKM1_LBM_Binary-method

Plot method for a BIKM1_LBM_Binary *object*

Description

Produce respectively one plot of two-dimensional segmentation of a BIKM1_LBM_Binary fit, a plot of evolution of the chosen criterion as a function of the number of row and column clusters, and a boxplot of conditional posteriors for each row and column cluster.

Usage

```
## S4 method for signature 'BIKM1_LBM_Binary'
plot(x, y, ...)
```

Arguments

x	an object of class BIKM1_LBM_Binary.
y	binary matrix of observations.
...	in the plot method, additional parameters (ignored)

Value

One **plot** (initial and estimated partitions) and three **ggplot2** objects (conditional posterior in each cluster for each matrix and the graph of chosen criterion values.

Examples

```
require(bikm1)
g=5
h=3
theta=list()
theta$pi_g=t(1/g*rep(1,g))
theta$rho_h=t(1/h*rep(1,h))
eps=0.1
theta$alpha_gh=matrix(c(1-eps,eps,eps,eps,1-eps,eps,eps,1-eps,1-eps,
1-eps,1-eps,eps,eps,eps,eps),ncol=h,byrow=TRUE)
n=250
J=150
data=BinBlocRnd_LBM(n,J,theta)
BinBlocVisu_LBM(data$x, data$xrow,data$xcol)
res=BIKM1_LBM_Binary(data$x,8,5,4,init_choice='smallVBayes')
BinBlocVisu_LBM(data$x,res@model_max$z,res@model_max$v)
e=CARI(data$xrow,data$xcol,res@model_max$z,res@model_max$v)
plot(res,data)
```

`plot,BIKM1_LBM_Poisson-method`

Plot method for a `BIKM1_LBM_Poisson` object

Description

Produce respectively one plot of two-dimensional segmentation of a `BIKM1_LBM_Poisson` fit, an evolution of the criterion as a function of the numbers of rows and columns, and a boxplot of conditional posteriors for each row and column cluster.

Usage

```
## S4 method for signature 'BIKM1_LBM_Poisson'
plot(x, y, ...)
```

Arguments

`x`	an object of class `BIKM1_LBM_Poisson`.
`y`	a list specifying `x` : contingency matrix of observations.
`...`	in the plot method, additional parameters (ignored)

Value

Two **plot**s (initial matrix and block estimation) and two **ggplot2** objects (conditional posterior in each cluster and the graph of chosen criterion values).

Examples

```
require(bikm1)
J=200
K=120
h=3
l=2
theta=list()
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$gamma_hl=matrix(c(1, 6,4, 1, 7, 1),ncol=2)
data=PoissonBlocRnd(J,K,theta)
res=BIKM1_LBM_Poisson(data$x,3,3,4,init_choice='random')
plot(res,data)
```

`plot,BIKM1_MLBM_Binary-method`

Plot method for a `BIKM1_MLBM_Binary` *object*

Description

Produce respectively a plot of two-dimensional segmentation of a `BIKM1_MLBM_Binary` fit, and a boxplot of conditional posteriors for each row and column cluster.

Usage

```
## S4 method for signature 'BIKM1_MLBM_Binary'
plot(x, y, ...)
```

Arguments

`x`	an object of class `BIKM1_MLBM_Binary`.
`y`	a list specifying : `x`: the first matrix of observations `y`: the second matrix of observations.
`...`	in the plot method, additional parameters (ignored)

Value

Two **plot** and on **ggplot2** object.

Examples

```
require(bikm1)
n=200
J=120
K=120
g=3
h=2
```

```
l=2
theta=list()
theta$pi_g=1/g *matrix(1,g,1)
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$alpha_gh=matrix(runif(6),ncol=h)
theta$beta_gl=matrix(runif(6),ncol=l)
data=BinBlocRnd_MLBM(n,J,K,theta)
res=BIKM1_MLBM_Binary(data$x,data$y,3,3,3,4)
plot(res,data)
```

`PoissonBlocBIC` *PoissonBlocBIC function for the computation of the BIC criterion in the Poisson LBM*

Description

Produce a value of the BIC criterion for co-clustering partitions

Usage

```
PoissonBlocBIC(a,alpha,beta,v1,w1,x,res,normalization)
```

Arguments

`a`	hyperparameter used in the VBayes algorithm for priors on the mixing proportions. By default, a=4.
`alpha`	hyperparameter used in the VBayes algorithm for prior on the Poisson parameter. By default, alpha=1.
`beta`	hyperparameter used in the VBayes algorithm for prior on the Poisson parameter. By default, beta=0.01.
`v1`	a numeric vector of row partitions
`w1`	a numeric vector of column partitions
`x`	contingency matrix of observations.
`res`	a BIKM1_LBM_Poisson object `rho_h` mixing row proportions `tau_l` mixing column proportions `gamma_hl` Bernoulli parameters
`normalization`	logical. To use the normalized Poisson modelling in the Latent Block Model. By default normalization=FALSE.

Value

a value of the BIC criterion

Examples

```
require(bikm1)
J=200
K=120
h=3
l=2
theta=list()
theta$rho_h=1/h*matrix(1,h,1)
theta$tau_l=1/l*matrix(1,l,1)
theta$gamma_hl=matrix(c(1, 6,4, 1, 7, 1),ncol=2)
data=PoissonBlocRnd(J,K,theta)
res=BIKM1_LBM_Poisson(data$x,3,3,4,init_choice='smallVBayes')
bic=PoissonBlocBIC(v1=res@model_max$v,w1=res@model_max$w,x=data$x,res=res,normalization=TRUE)
```

PoissonBlocICL — *PoissonBlocICL function for the computation of the ICL criterion in the Poisson LBM*

Description

Produce a value of the ICL criterion for co-clustering partitions

Usage

```
PoissonBlocICL(a,alpha,beta,x,v1,w1,normalization)
```

Arguments

a	hyperparameter used in the VBayes algorithm for priors on the mixing proportions. By default, a=4.
alpha	hyperparameter used in the VBayes algorithm for prior on the Poisson parameter. By default, alpha=1.
beta	hyperparameter used in the VBayes algorithm for prior on the Poisson parameter. By default, beta=0.01.
x	contingency matrix of observations.
v1	a numeric vector specifying the class of rows.
w1	a numeric vector specifying the class of columns.
normalization	logical. To use the normalized Poisson modelling in the Latent Block Model. By default normalization=FALSE.

Value

a value of the ICL criterion

Examples

```
require(bikm1)
J=200
K=120
h=3
l=2
theta=list()
theta$rho_h=(1/h)*matrix(1,h,1)
theta$tau_l=(1/l)*matrix(1,l,1)
theta$gamma_hl=matrix(c(1, 6,4, 1, 7, 1),ncol=2)
data=PoissonBlocRnd(J,K,theta)
res=BIKM1_LBM_Poisson(data$x,4,4,4,init_choice='smallVBayes')
icl=PoissonBlocICL(4,1,0.01,data$x,res@model_max$v,res@model_max$w, normalization=FALSE)
```

`PoissonBlocRnd` *PoissonBlocRnd function for contingency data simulation*

Description

Produce a simulated data matrix generated under the Poisson Latent Block Model.

Usage

```
PoissonBlocRnd(J,K,theta)
```

Arguments

`J`	a positive integer specifying the number of expected rows.
`K`	a positive integer specifying the number of expected columns.
`theta`	a list specifying the model parameters: `rho_h`: a vector specifying the row mixing proportions. `tau_l`: a vector specifying the column mixing proportions. `gamma_hl`: a matrix specifying the distribution parameter.

Value

a list including the arguments:

`x`: simulated contingency data matrix.

`xrow`: numeric vector specifying row partition.

`xcol`: numeric vector specifying column partition.

Examples

```
require(bikm1)
J=200
K=120
h=3
l=2
theta=list()
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$gamma_hl=matrix(c(1, 6,4, 1, 7, 1),ncol=2)
data=PoissonBlocRnd(J,K,theta)
```

PoissonBlocVisu *PoissonBlocVisu function for visualization of contingency datasets*

Description

Produce a plot object representing the co-clustered data-sets.

Usage

```
PoissonBlocVisu(x,v,w)
```

Arguments

x	contingency matrix of observations.
v	a numeric vector specifying the class of rows.
w	a numeric vector specifying the class of columns.

Value

a **plot** object

Examples

```
require(bikm1)
J=200
K=120
h=3
l=2
theta=list()
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$gamma_hl=matrix(c(1, 6,4, 1, 7, 1),ncol=2)
data=PoissonBlocRnd(J,K,theta)
PoissonBlocVisu(data$x,data$xrow,data$xcol)
```

PoissonBlocVisuResum — *PoissonBlocVisuResum function for visualization of contingency datasets*

Description

Produce a plot object representing the resumed co-clustered data-sets.

Usage

```
PoissonBlocVisuResum(x,v,w)
```

Arguments

x	contingency matrix of observations.
v	a numeric vector specifying the class of rows.
w	a numeric vector specifying the class of columns.

Value

a **plot** object.

Examples

```
require(bikm1)
J=200
K=120
h=3
l=2
theta=list()
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$gamma_hl=matrix(c(1, 6,4, 1, 7, 1),ncol=2)
data=PoissonBlocRnd(J,K,theta)
PoissonBlocVisuResum(data$x,data$xrow,data$xcol)
```

print,BIKM1_LBM_Binary-method — *Print method for a BIKM1_LBM_Binary object*

Description

Print method for a BIKM1_LBM_Binary object

Usage

```
## S4 method for signature 'BIKM1_LBM_Binary'
print(x, ...)
```

Arguments

x	in the print method, a BIKM1_LBM_Binary object
...	in the print method, additional parameters (ignored)

Examples

```
require(bikm1)
n=200
J=120
g=3
h=2
theta=list()
theta$pi_g=1/g *matrix(1,g,1)
theta$rho_h=1/h *matrix(1,h,1)
theta$alpha_gh=matrix(runif(6),ncol=h)
data=BinBlocRnd_LBM(n,J,theta)
res=BIKM1_LBM_Binary(data$x,3,2,4,init_choice='random')
print(res)
```

---

print,BIKM1_LBM_Poisson-method

Print method for a BIKM1_LBM_Poisson object

Description

Print method for a BIKM1_LBM_Poisson object

Usage

```
## S4 method for signature 'BIKM1_LBM_Poisson'
print(x, ...)
```

Arguments

x	in the print method, a BIKM1_LBM_Poisson object
...	in the print method, additional parameters (ignored)

Examples

```
require(bikm1)
J=200
K=120
h=3
l=2
theta=list()
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$gamma_hl=matrix(c(1, 6,4, 1, 7, 1),ncol=2)
data=PoissonBlocRnd(J,K,theta)
res=BIKM1_LBM_Poisson(data$x,3,2,4,init_choice='random')
print(res)
```

---

print,BIKM1_MLBM_Binary-method

Print method for a BIKM1_MLBM_Binary object

Description

Print method for a BIKM1_MLBM_Binary object

Usage

```
## S4 method for signature 'BIKM1_MLBM_Binary'
print(x, ...)
```

Arguments

x	in the print method, a BIKM1_MLBM_Binary object
...	in the print method, additional parameters (ignored)

Examples

```
require(bikm1)
n=200
J=120
K=120
g=3
h=2
l=2
theta=list()
theta$pi_g=1/g *matrix(1,g,1)
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$alpha_gh=matrix(runif(6),ncol=h)
theta$beta_gl=matrix(runif(6),ncol=l)
data=BinBlocRnd_MLBM(n,J,K,theta)
```

```
res=BIKM1_MLBM_Binary(data$x,data$y,3,3,3,4)
print(res)
```

---

show,BIKM1_LBM_Binary-method
Show method for a BIKM1_LBM_Binary object

Description

show method for a BIKM1_LBM_Binary object

Usage

```
## S4 method for signature 'BIKM1_LBM_Binary'
show(object)
```

Arguments

object	a BIKM1_LBM_Binary object

Examples

```
require(bikm1)
n=200
J=120
g=3
h=2
theta=list()
theta$pi_g=1/g *matrix(1,g,1)
theta$rho_h=1/h *matrix(1,h,1)
theta$alpha_gh=matrix(runif(6),ncol=h)
data=BinBlocRnd_LBM(n,J,theta)
res=BIKM1_LBM_Binary(data$x,4,4,4,init_choice='random')
show(res)
```

---

show,BIKM1_LBM_Poisson-method
Show method for a BIKM1_LBM_Poisson object

Description

show method for a BIKM1_LBM_Poisson object

Usage

```
## S4 method for signature 'BIKM1_LBM_Poisson'
show(object)
```

Arguments

object	a BIKM1_LBM_Poisson object

Examples

```
require(bikm1)
J=200
K=120
h=3
l=2
theta=list()
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$gamma_hl=matrix(c(1, 6,4, 1, 7, 1),ncol=2)
data=PoissonBlocRnd(J,K,theta)
res=BIKM1_LBM_Poisson(data$x,4,4,4,init_choice='random')
show(res)
```

---

show,BIKM1_MLBM_Binary-method
Show method for a BIKM1_MLBM_Binary object

Description

show method for a BIKM1_MLBM_Binary object

Usage

```
## S4 method for signature 'BIKM1_MLBM_Binary'
show(object)
```

Arguments

object	a BIKM1_MLBM_Binary object

Examples

```
require(bikm1)
n=200
J=120
K=120
g=3
h=2
```

```
l=2
theta=list()
theta$pi_g=1/g *matrix(1,g,1)
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$alpha_gh=matrix(runif(6),ncol=h)
theta$beta_gl=matrix(runif(6),ncol=l)
data=BinBlocRnd_MLBM(n,J,K,theta)
res=BIKM1_MLBM_Binary(data$x,data$y,3,3,3,4)
show(res)
```

---

summary,BIKM1_LBM_Binary-method
Summary method for a BIKM1_LBM_Binary object

Description

Produce a summary of informations of a BIKM1_LBM_Binary object

Usage

```
## S4 method for signature 'BIKM1_LBM_Binary'
summary(object, ...)
```

Arguments

object	in the summary method, a BIKM1_LBM_Binary object
...	in the summary method, additional parameters (ignored)

Examples

```
require(bikm1)
n=200
J=120
g=3
h=2
theta=list()
theta$pi_g=1/g *matrix(1,g,1)
theta$rho_h=1/h *matrix(1,h,1)
theta$alpha_gh=matrix(runif(6),ncol=h)
data=BinBlocRnd_LBM(n,J,theta)
res=BIKM1_LBM_Binary(data$x,3,2,4,init_choice='random')
summary(res)
```

`summary,BIKM1_LBM_Poisson-method`
Summary method for a BIKM1_LBM_Poisson object

Description

Produce a summary of informations of a `BIKM1_LBM_Poisson` object

Usage

```
## S4 method for signature 'BIKM1_LBM_Poisson'
summary(object, ...)
```

Arguments

`object`	in the summary method, a BIKM1_LBM_Poisson object
`...`	in the summary method, additional parameters (ignored)

Examples

```
require(bikm1)
J=200
K=120
h=3
l=2
theta=list()
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$gamma_hl=matrix(c(1, 6,4, 1, 7, 1),ncol=2)
data=PoissonBlocRnd(J,K,theta)
res=BIKM1_LBM_Poisson(data$x,4,4,4,init_choice='random')
summary(res)
```

`summary,BIKM1_MLBM_Binary-method`
Summary method for a BIKM1_MLBM_Binary object

Description

Produce a summary of informations of a `BIKM1_MLBM_Binary` object

Usage

```
## S4 method for signature 'BIKM1_MLBM_Binary'
summary(object, ...)
```

Arguments

`object`	in the summary method, a BIKM1_MLBM_Binary object
`...`	in the summary method, additional parameters (ignored)

Examples

```
require(bikm1)
n=200
J=120
K=120
g=3
h=2
l=2
theta=list()
theta$pi_g=1/g *matrix(1,g,1)
theta$rho_h=1/h *matrix(1,h,1)
theta$tau_l=1/l *matrix(1,l,1)
theta$alpha_gh=matrix(runif(6),ncol=h)
theta$beta_gl=matrix(runif(6),ncol=l)
data=BinBlocRnd_MLBM(n,J,K,theta)
res=BIKM1_MLBM_Binary(data$x,data$y,3,3,3,4)
summary(res)
```

Index