Package ‘blockmodels’

October 12, 2022

Type Package
Title Latent and Stochastic Block Model Estimation by a 'V-EM' Algorithm
Version 1.1.5
Date 2021-12-01
Author Jean-Benoist Leger <jbleger@hds.utc.fr>, Pierre Barbillon <pierre.barbillon@agroparistech.fr>, Julien Chiquet <julien.chiquet@inrae.fr>
Maintainer Jean-Benoist Leger <jbleger@hds.utc.fr>
Description Latent and Stochastic Block Model estimation by a Variational EM algorithm. Various probability distribution are provided (Bernoulli, Poisson...), with or without covariates.
License LGPL-2.1
Depends R (>= 3.4.0)
LinkingTo Rcpp, RcppArmadillo
RoxygenNote 7.1.1
Imports Rcpp, parallel, methods, digest
NeedsCompilation yes
Repository CRAN
Date/Publication 2021-12-01 11:20:09 UTC

R topics documented:

BM_bernoulli .................................................. 2
BM_bernoulli_covariates .................................. 4
BM_bernoulli_covariates_fast ........................... 7
BM_bernoulli_multiplex ................................... 10
BM_gaussian .................................................. 14
BM_gaussian_covariates .................................. 16
BM_gaussian_multivariate ................................. 19
BM_gaussian_multivariate_independent ................ 22
BM_gaussian_multivariate_independent_homoscedastic .. 25
BM_poisson .................................................. 28
BM_poisson_covariates .................................... 30
BM_bernoulli

Perform estimation on blockmodels for bernoulli probability distribution

Description
With the provided network and blockmodel type, estimate number of groups, parameters and node membership

Usage

```r
## S4 method for signature 'new'
BM_bernoulli(
  membership_type,
  adj,
  verbosity=6,
  autosave=",
  plotting=character(0),
  exploration_factor=1.5,
  exploration_direction=numeric(0),
  explore_min=4,
  explore_max=Inf,
  ncores=detectCores())
```

Arguments

- `membership_type` The type of node membership, i.e. 'SBM', 'SBM_sym' or 'LBM'
- `adj` The adjacency matrix
- `verbosity` The verbosity level, 0 means quiet. Level 1 display the phase of reinitialization. Level 2 display the level 1 and the ascending and descending phase for the number of groups. Level 3 display the level 2 and the number current number of groups which is estimated. Level 4 display the level 3 and the steps inside the estimation. Level 5 display the level 4, the current status of parallel running jobs and the current sub-step. Level 6 display level 5 and informations about ICL criteria found. Default is level 6. This parameter can be changed by accessing to the field $verbosity of the object.
- `autosave` If `autosave != "`, after each estimation, the model object is writed into file `autosave`. The model object is readable by the function `readRDS`. Use-it for long computation to allow restarting the estimation on system crash. You can use it to alanyze the partial results when the estimation is running. This parameter can be changed by accessing to the field $autosave of the object.
- `plotting` Control plot of ICL values while the estimation is running. If `plotting==character(0)` (the default), plots are done on screen, if `plotting=="", no plot are done, if `plotting` is a filename, plots are done in this filename. This parameter can be changed by accessing the field $plotting of the object.
exploration_factor
Control the exploration of the number of groups. The exploration is stop when the number of groups reach exploration_factor times the current maximum. By default 1.5. This parameter can be changed by accessing the field $exploration_factor of the object.

explore_min
Explore to the explore_min number of groups even if the exploration_factor rule is satisfied. By default 4. This parameter can be changed by accessing the field $explore_min of the object.

explore_max
Stop exploration after explore_max number of group in any case. By default Inf. This parameter can be changed by accessing the field $explore_max of the object.

exploration_direction
Only for LBM membership. Control the exploration direction for groups number. When provided, the exploring strategy is made to explore the provided group number. Must be a vector of two integer value representing the row group number and the column group number.

ncores
Number of parallel jobs to launch different EM intializations. By default detectCores(). This parameter can be changed by accessing the field $ncores of the object. This parameters is used only on Linux. Parallism is disabled on other plateform. (Not working on Windows, not tested on Mac OS, not tested on *BSD.)

Examples

## Not run:

##
## SBM
##

```r
## generation of one SBM network
npc <- 30 # nodes per class
Q <- 3 # classes
n <- npc * Q # nodes
Z <- diag(Q) %x% matrix(1, npc, 1)
P <- matrix(runif(Q * Q), Q, Q)
M <- 1 * (matrix(runif(n * n), n, n) < Z %*% P %*% t(Z))  # adjacency matrix

## estimation
my_model <- BM_bernoulli("SBM", M)
my_model$estimate()
which.max(my_model$ICL)
```

## SBM symmetric

##

```r
## generation of one SBM_sym network
npc <- 30 # nodes per class
Q <- 3 # classes
```
n <- npc * Q  # nodes
Z <- diag(Q) %x% matrix(1, npc, 1)
P <- matrix(runif(Q * Q), Q, Q)
P[lower.tri(P)] <- t(P)[lower.tri(P)]
M <- 1 * (matrix(runif(n * n), n, n) < Z %*% P %*% t(Z))  # adjacency matrix
M[lower.tri(M)] <- t(M)[lower.tri(M)]

## estimation
my_model <- BM_bernoulli("SBM_sym", M)
my_model$estimate()
which.max(my_model$ICL)

##
## LBM
##

## generation of one LBM network
npc <- c(50, 40)  # nodes per class
Q <- c(2, 3)  # classes
n <- npc * Q  # nodes
Z1 <- diag(Q[1]) %x% matrix(1, npc[1], 1)
Z2 <- diag(Q[2]) %x% matrix(1, npc[2], 1)
P <- matrix(runif(Q[1] * Q[2]), Q[1], Q[2])
M <- 1 * (matrix(runif(n[1] * n[2]), n[1], n[2]) < Z1 %*% P %*% t(Z2))  # adjacency matrix

## estimation
my_model <- BM_bernoulli("LBM", M)
my_model$estimate()
which.max(my_model$ICL)

## End(Not run)

---

**BM_bernoulli_covariates**

*Perform estimation on blockmodels for bernoulli probability distribution with covariates*

**Description**

With the provided network and blockmodel type, estimate number of groups, parameters and node membership, and impact vector of covariates.

**Usage**

```r
## S4 method for signature 'new'
BM_bernoulli_covariates(
  membership_type,
  adj,
  covariates,
  verbosity=6,
)```

Arguments

**membership_type**
The type of node membership, i.e. 'SBM', 'SBM_sym' or 'LBM'

**adj**
The adjacency matrix

**covariates**
Covariates matrix, or list of covariates matrices. Covariates matrix must have the same size than the adjacency matrix.

**verbosity**
The verbosity level, 0 means quiet. Level 1 display the phase of reinitialization. Level 2 display the level 1 and the ascending and descending phase for the number of groups. Level 3 display the level 2 and the number current number of groups which is estimated. Level 4 display the level 3 and the steps inside the estimation. Level 5 display the level 4, the current status of parallel running jobs and the current sub-step. Level 6 display level 5 and informations about ICL criteria found. Default is level 6. This parameter can be changed by accessing to the field $verbosity of the object.

**autosave**
If autosave != '', after each estimation, the model object is writed into file autosave. The model object is readable by the function readRDS. Use-it for long computation to allow restarting the estimation on system crash. You can use it to alanyze the partial results when the estimation is running. This parameter can be changed by accessing to the field $autosave of the object.

**plotting**
Control plot of ICL values while the estimation is running. If plotting==character(0) (the default), plots are done on screen, if plotting==", no plot are done, if plotting is a filename, plots are done in this filename. This parameter can be changed by accessing the field $plotting of the object.

**exploration_factor**
Control the exploration of the number of groups. The exploration is stop when the number of groups reach exploration factor times the current maximum. By default 1.5. This parameter can be changed by accessing the field $exploration_factor of the object.

**explore_min**
Explore to the explore_min number of groups even if the exploration_factor rule is satisfied. By default 4. This parameter can be changed by accessing the field $explore_min of the object.

**explore_max**
Stop exploration after explore_max number of group in any case. By default Inf. This parameter can be changed by accessing the field $explore_max of the object.

**exploration_direction**
Only for LBM membership. Control the exploration direction for groups number. When provided, the exploring strategy is made to explore the provided
group number. Must be a vector of two integer value representing the row group
number and the column group number.

**ncores**

Number of parallel jobs to launch different EM intializations. By default detectCores(). This parameter can be changed by accessing the field $ncores of
the object. This parameters is used only on Linux. Parallism is disabled on
other plateform. (Not working on Windows, not tested on Mac OS, not tested
on *BSD.)

**Examples**

```r
## Not run:
##
## ## SBM
##
## ## generation of one SBM network
npc <- 30 # nodes per class
Q <- 3 # classes
n <- npc * Q # nodes
sigmo <- function(x){1/(1+exp(-x))}
Z<-diag(Q)%x%matrix(1,npc,1)
Mg<-8*matrix(runif(Q*Q),Q,Q)-4
Y1 <- matrix(runif(n*n),n,n)-.5
Y2 <- matrix(runif(n*n),n,n)-.5
M_in_expectation<-sigmo(Z%*%Mg%*%t(Z) + 5*Y1-3*Y2)
M<-1*(matrix(runif(n*n),n,n)<M_in_expectation)
## estimation
my_model <- BM_bernoulli_covariates("SBM",M,list(Y1,Y2) )
my_model$estimate()
which.max(my_model$ICL)
```

```r
##
## ## SBM symmetric
##
## ## generation of one SBM_sym network
npc <- 30 # nodes per class
Q <- 3 # classes
n <- npc * Q # nodes
sigmo <- function(x){1/(1+exp(-x))}
Z<-diag(Q)%x%matrix(1,npc,1)
Mg<-8*matrix(runif(Q*Q),Q,Q)-4
Mg[lower.tri(Mg)]<-t(Mg)[lower.tri(Mg)]
Y1 <- matrix(runif(n*n),n,n)-.5
Y2 <- matrix(runif(n*n),n,n)-.5
Y1[lower.tri(Y1)]<-t(Y1)[lower.tri(Y1)]
Y2[lower.tri(Y2)]<-t(Y2)[lower.tri(Y2)]
M_in_expectation<-sigmo(Z%*%Mg%*%t(Z) + 5*Y1-3*Y2)
M<-1*(matrix(runif(n*n),n,n)<M_in_expectation)
```
BM_bernoulli_covariates_fast

Perform estimation on blockmodels for bernoulli probability distribution with covariates

Description

With the provided network and blockmodel type, estimate number of groups, parameters and node membership, and impact vector of covariates.

Usage

## S4 method for signature 'new'
BM_bernoulli_covariates_fast(
  membership_type,
  adj,
  covariates,
)
verbosity=6,
autosave=''
plotting=character(0),
exploration_factor=1.5,
exploration_direction=numeric(0),
explot_min=4,
explot_max=Inf,
ncores=detectCores()}

Arguments

membership_type
The type of node membership, i.e. 'SBM', 'SBM_sym' or 'LBM'

adj
The adjacency matrix

covariates
Covariates matrix, or list of covariates matrices. Covariates matrix must have the same size than the adjacency matrix.

verbosity
The verbosity level, 0 means quiet. Level 1 display the phase of reinitialization. Level 2 display the level 1 and the ascending and descending phase for the number of groups. Level 3 display the level 2 and the number current number of groups which is estimated. Level 4 display the level 3 and the steps inside the estimation. Level 5 display the level 4, the current status of parallel running jobs and the current sub-step. Level 6 display level 5 and informations about ICL criteria found. Default is level 6. This parameter can be changed by accessing to the field $verbosity of the object.

autosave
If autosave != '', after each estimation, the model object is writed into file autosave. The model object is readable by the function readRDS. Use-it for long computation to allow restarting the estimation on system crash. You can use it to alanyze the partial results when the estimation is running. This parameter can be changed by accessing to the field $autosave of the object.

plotting
Control plot of ICL values while the estimation is running. If plotting==character(0) (the default), plots are done on screen, if plotting== '', no plot are done, if plotting is a filename, plots are done in this filename. This parameter can be changed by accessing the field $plotting of the object.

exploration_factor
Control the exploration of the number of groups. The exploration is stop when the number of groups reach exploration factor times the current maximum. By default 1.5. This parameter can be changed by accessing the field $exploration_factor of the object.

explore_min
Explore to the explore_min number of groups even if the exploration_factor rule is satisfied. By default 4. This parameter can be changed by accessing the field $explore_min of the object.

explore_max
Stop exploration after explore_max number of group in any case. By default Inf. This parameter can be changed by accessing the field $explore_max of the object.

exploration_direction
Only for LBM membership. Control the exploration direction for groups number. When provided, the exploring strategy is made to explore the provided
group number. Must be a vector of two integer value representing the row group number and the column group number.

**ncores**  
Number of parallel jobs to launch different EM initializations. By default detectCores(). This parameter can be changed by accessing the field ncores of the object. This parameters is used only on Linux. Parallelism is disabled on other platform. (Not working on Windows, not tested on Mac OS, not tested on *BSD.)

**Examples**

```r
## Not run:

##
## SBM
##
## generation of one SBM network
npc <- 30 # nodes per class
Q <- 3 # classes
n <- npc * Q # nodes
sigmo <- function(x){1/(1+exp(-x))}
Z<-diag(Q)%x%matrix(1,npc,1)
Mg<-8*matrix(runif(Q*Q),Q,Q)-4
Y1 <- matrix(runif(n*n),n,n)-.5
Y2 <- matrix(runif(n*n),n,n)-.5
M_in_expectation<-sigmo(Z*Mg%*%t(Z) + 5*Y1-3*Y2)
M<-1*(matrix(runif(n*n),n,n)<M_in_expectation)

## estimation
my_model <- BM_bernoulli_covariates_fast("SBM",M,list(Y1,Y2) )
my_model$estimate()
which.max(my_model$ICL)

##
## SBM symmetric
##
## generation of one SBM_sym network
npc <- 30 # nodes per class
Q <- 3 # classes
n <- npc * Q # nodes
sigmo <- function(x){1/(1+exp(-x))}
Z<-diag(Q)%x%matrix(1,npc,1)
Mg<-8*matrix(runif(Q*Q),Q,Q)-4
Mg[lower.tri(Mg)]<-t(Mg)[lower.tri(Mg)]
Y1 <- matrix(runif(n*n),n,n)-.5
Y2 <- matrix(runif(n*n),n,n)-.5
Y1[lower.tri(Y1)]<-t(Y1)[lower.tri(Y1)]
Y2[lower.tri(Y2)]<-t(Y2)[lower.tri(Y2)]
M_in_expectation<-sigmo(Z*Mg%*%t(Z) + 5*Y1-3*Y2)
M<-1*(matrix(runif(n*n),n,n)<M_in_expectation)
```

```
M[lower.tri(M)]<-t(M)[lower.tri(M)]

## estimation
my_model <- BM_bernoulli_covariates_fast("SBM_sym",M,list(Y1,Y2) )
my_model$estimate()
which.max(my_model$ICL)

##
## LBM
##
## generation of one LBM network
npc <- c(50,40) # nodes per class
Q <- c(2,3) # classes
n <- npc * Q # nodes
sigmo <- function(x){1/(1+exp(-x))}
Z1<-diag(Q[1]%%x%matrix(1,npc[1],1))
Z2<-diag(Q[2]%%x%matrix(1,npc[2],1))
Mg<-8*matrix(runif(Q[1]*Q[2]),Q[1],Q[2])-4
Y1 <- matrix(runif(n[1]*n[2]),n[1],n[2])-.5
Y2 <- matrix(runif(n[1]*n[2]),n[1],n[2])-.5
M_in_expectation<-sigmo(Z1%%x%Mg%%x%t(Z2) + 5*Y1-3*Y2)
M<-1*(matrix(runif(n[1]*n[2]),n[1],n[2])<M_in_expectation)

## estimation
my_model <- BM_bernoulli_covariates_fast("LBM",M,list(Y1,Y2) )
my_model$estimate()
which.max(my_model$ICL)

## End(Not run)
```

---

**BM_bernoulli_multiplex**

*Perform estimation on blockmodels for multiplex binary networks*

**Description**

With the provided network and blockmodel type, estimate number of groups, parameters and node membership

**Usage**

```r
## S4 method for signature 'new'
BM_bernoulli_multiplex(
  membership_type,
  adj,
  verbosity=6,
)```
arguments

pagination

Arguments

membership_type The type of node membership, i.e. 'SBM', 'SBM_sym' or 'LBM'
adj The list of adjacency matrices. All matrices must have the same size
verbosity The verbosity level, 0 means quiet. Level 1 display the phase of reinitialization. Level 2 display the level 1 and the ascending and descending phase for the number of groups. Level 3 display the level 2 and the number current number of groups which is estimated. Level 4 display the level 3 and the steps inside the estimation. Level 5 display the level 4, the current status of parallel running jobs and the current sub-step. Level 6 display level 5 and informations about ICL criteria found. Default is level 6. This parameter can be changed by accessing to the field $verbosity of the object.
autosave If autosave != '', after each estimation, the model object is writed into file autosave. The model object is readable by the function readRDS. Use-it for long computation to allow restarting the estimation on system crash. You can use it to alanyze the partial results when the estimation is running. This parameter can be changed by accessing to the field $autosave of the object.
plotting Control plot of ICL values while the estimation is running. If plotting==character(0) (the default), plots are done on screen, if plotting=='', no plot are done, if plotting is a filename, plots are done in this filename. This parameter can be changed by accessing to the field $plotting of the object.
exploration_factor Control the exploration of the number of groups. The exploration is stop when the number of groups reach exploration factor times the current maximum. By default 1.5. This parameter can be changed by accessing the field $exploration_factor of the object.
explore_min Explore to the explore_min number of groups even if the exploration_factor rule is satisfied. By default 4. This parameter can be changed by accessing the field $explore_min of the object.
explore_max Stop exploration after explore_max number of group in any case. By default Inf. This parameter can be changed by accessing the field $explore_max of the object.
exploration_direction Only for LBM membership. Control the exploration direction for groups number. When provided, the exploring strategy is made to explore the provided group number. Must be a vector of two integer value representing the row group number and the column group number.
ncores

Number of parallel jobs to launch different EM initializations. By default detectCores(). This parameter can be changed by accessing the field ncores of the object. This parameter is used only on Linux. Parallism is disabled on other platforms. (Not working on Windows, not tested on Mac OS, not tested on *BSD.)

Examples

```
## Not run:
##
## SBM
##
## generation of one SBM network
npc <- 30 # nodes per class
Q <- 3 # classes
n <- npc * Q # nodes
Z <- diag(Q) %x% matrix(1, npc, 1)
P00 <- matrix(runif(Q * Q), Q, Q)
P10 <- matrix(runif(Q * Q), Q, Q)
P01 <- matrix(runif(Q * Q), Q, Q)
P11 <- matrix(runif(Q * Q), Q, Q)
SumP <- P00 + P10 + P01 + P11
P00 <- P00 / SumP
P01 <- P01 / SumP
P10 <- P10 / SumP
P11 <- P11 / SumP
MU <- matrix(runif(n * n), n, n)
M1 <- 1 * (MU > Z %*% (P00 + P01) %*% t(Z))
M2 <- 1 * ((MU > Z %*% P00 %*% t(Z)) & (MU < Z %*% (P00 + P01 + P11) %*% t(Z)))
```

```
## estimation
my_model <- BM_bernoulli_multiplex("SBM", list(M1, M2))
my_model$estimate()
which.max(my_model$ICL)
```

```
## SBM symmetric
##
## generation of one SBM network
npc <- 30 # nodes per class
Q <- 3 # classes
n <- npc * Q # nodes
Z <- diag(Q) %x% matrix(1, npc, 1)
P00 <- matrix(runif(Q * Q), Q, Q)
P10 <- matrix(runif(Q * Q), Q, Q)
P01 <- matrix(runif(Q * Q), Q, Q)
P11 <- matrix(runif(Q * Q), Q, Q)
SumP <- P00 + P10 + P01 + P11
```
P00<-P00/SumP
P01<-P01/SumP
P10<-P10/SumP
P11<-P11/SumP
P00[lower.tri(P00)]<-t(P00)[lower.tri(P00)]
P01[lower.tri(P01)]<-t(P01)[lower.tri(P01)]
P10[lower.tri(P10)]<-t(P10)[lower.tri(P10)]
P11[lower.tri(P11)]<-t(P11)[lower.tri(P11)]
MU<-matrix(runif(n*n),n,n)
MU[lower.tri(MU)]<-t(MU)[lower.tri(MU)]
M1<-1*(MU>Z%*%(P00+P01)%*%t(Z))
M2<-1*((MU>Z%*%(P00+P01)%*%t(Z)) & (MU<Z%*%(P00+P01+P11)%*%t(Z))) ## adjacency matrices

## estimation
my_model <- BM_bernoulli_multiplex("SBM_sym",list(M1,M2) )
my_model$estimate()
which.max(my_model$ICL)

## LBM
##
## generation of one LBM network
npc <- c(50,40) # nodes per class
Q <- c(2,3) # classes
n <- npc * Q # nodes
Z1<-diag(Q[1])%x%matrix(1,npc[1],1)
Z2<-diag(Q[2])%x%matrix(1,npc[2],1)
P00<-matrix(runif(Q[1]*Q[2]),Q[1],Q[2])
P10<-matrix(runif(Q[1]*Q[2]),Q[1],Q[2])
P01<-matrix(runif(Q[1]*Q[2]),Q[1],Q[2])
P11<-matrix(runif(Q[1]*Q[2]),Q[1],Q[2])
SumP<-P00+P10+P01+P11
P00<-P00/SumP
P01<-P01/SumP
P10<-P10/SumP
P11<-P11/SumP
MU<-matrix(runif(n[1]*n[2]),n[1],n[2])
M1<-1*(MU>Z1%*%(P00+P01)%*%t(Z2))
M2<-1*((MU>Z1%*%P00%*%t(Z2)) & (MU<Z1%*%(P00+P01+P11)%*%t(Z2))) ## adjacency matrices

## estimation
my_model <- BM_bernoulli_multiplex("LBM",list(M1,M2) )
my_model$estimate()
which.max(my_model$ICL)

## End(Not run)
**BM_gaussian**  

Perform estimation on blockmodels for gaussian probability distribution

### Description

With the provided network and blockmodel type, estimate number of groups, parameters and node membership

### Usage

```r
## S4 method for signature 'new'
BM_gaussian(
  membership_type,
  adj,
  verbosity=6,
  autosave=''
)
```

### Arguments

- **membership_type**
  - The type of node membership, i.e. 'SBM', 'SBM_sym' or 'LBM'

- **adj**
  - The adjacency matrix

- **verbosity**
  - The verbosity level, 0 means quiet. Level 1 display the phase of reinitialization. Level 2 display the level 1 and the ascending and descending phase for the number of groups. Level 3 display the level 2 and the number current number of groups which is estimated. Level 4 display the level 3 and the steps inside the estimation. Level 5 display the level 4, the current status of parallel running jobs and the current sub-step. Level 6 display level 5 and informations about ICL criteria found. Default is level 6. This parameter can be changed by accessing to the field $verbosity of the object.

- **autosave**
  - If autosave != '', after each estimation, the model object is writed into file autosave. The model object is readable by the function readRDS. Use-it for long computation to allow restarting the estimation on system crash. You can use it to alanyze the partial results when the estimation is running. This parameter can be changed by accessing to the field $autosave of the object.

- **plotting**
  - Control plot of ICL values while the estimation is running. If plotting==character(0) (the default), plots are done on screen, if plotting=='', no plot are done, if plotting is a filename, plots are done in this filename. This parameter can be changed by accessing the field $plotting of the object.
BM_gaussian

exploration_factor
Control the exploration of the number of groups. The exploration is stop when
the number of groups reach exploration factor times the current maximum. By
default 1.5. This parameter can be changed by accessing the field $exploration_factor of the object.

explore_min
Explore to the explore_min number of groups even if the exploration_factor rule
is satisfied. By default 4. This parameter can be changed by accessing the field $explore_min of the object.

explore_max
Stop exploration after explore_max number of group in any case. By default
Inf. This parameter can be changed by accessing the field $explore_max of the
object.

exploration_direction
Only for LBM membership. Control the exploration direction for groups number.
When provided, the exploring strategy is made to explore the provided
group number. Must be a vector of two integer value representing the row group
number and the column group number.

ncores
Number of parallel jobs to launch different EM intializations. By default de-
tectCores(). This parameter can be changed by accessing the field $ncores of the
object. This parameters is used only on Linux. Parallism is disabled on
other plateform. (Not working on Windows, not tested on Mac OS, not tested
on *BSD.)

Examples

## Not run:

##
## SBM

##

npc <- 30 # nodes per class
Q <- 3 # classes
n <- npc * Q # nodes
Z<diag(Q)%x%matrix(1,npc,1)
Mu<20*matrix(runif(Q*Q),Q,Q)
M<-matrix(rnorm(n*n,sd=10),n,n)+Z%*%Mu%*%t(Z) ## adjacency matrix

## estimation
my_model <- BM_gaussian("SBM",M )
my_model$estimate()
which.max(my_model$ICL)

##
## SBM symmetric

##

npc <- 30 # nodes per class
Q <- 3 # classes
Perform estimation on blockmodels for gaussian probability distribution with covariates

### Description

With the provided network and blockmodel type, estimate number of groups, parameters and node membership and impact vector of covariates

### Usage

```r
## S4 method for signature 'new'
BM_gaussian_covariates(
  membership_type,
  adj,
  covariates,
  verbosity=6,
) ```
BM_gaussian_covariates

```
autosave='',
plotting=character(0),
exploration_factor=1.5,
exploration_direction=numeric(0),
explore_min=4,
explore_max=Inf,
ncores=detectCores())
```

**Arguments**

- **membership_type**
  The type of node membership, i.e. 'SBM', 'SBM_sym' or 'LBM'

- **adj**
  The adjacency matrix

- **covariates**
  Covariates matrix, or list of covariates matrices. Covariates matrix must have
  the same size than the adjacency matrix.

- **verbosity**
  The verbosity level, 0 means quiet. Level 1 display the phase of reinitialization.
  Level 2 display the level 1 and the ascending and descending phase for the number
  of groups. Level 3 display the level 2 and the number current number of
  groups which is estimated. Level 4 display the level 3 and the steps inside the
  estimation. Level 5 display the level 4, the current status of parallel running jobs
  and the current sub-step. Level 6 display level 5 and informations about ICL cri-
  teria found. Default is level 6. This parameter can be changed by accessing to
  the field $verbosity of the object.

- **autosave**
  If autosave != '', after each estimation, the model object is writed into file auto-
  save. The model object is readable by the function readRDS. Use-it for long
  computation to allow restarting the estimation on system crash. You can use it
to alanyze the partial results when the estimation is running. This parameter can
be changed by accessing to the field $autosave of the object.

- **plotting**
  Control plot of ICL values while the estimation is running. If plotting==character(0)
  (the default), plots are done on screen, if plotting==”, no plot are done, if plot-
ing is a filename, plots are done in this filename. This parameter can be changed
by accessing the field $plotting of the object.

- **exploration_factor**
  Control the exploration of the number of groups. The exploration is stop when
the number of groups reach exploration factor times the current maximum. By
default 1.5. This parameter can be changed by accessing the field $exploration_factor of the object.

- **explore_min**
  Explore to the explore_min number of groups even if the exploration_factor rule
  is satisfied. By default 4. This parameter can be changed by accessing the field
  $explore_min of the object.

- **explore_max**
  Stop exploration after explore_max number of group in any case. By default
  Inf. This parameter can be changed by accessing the field $explore_max of the
  object.

- **exploration_direction**
  Only for LBM membership. Control the exploration direction for groups num-
  ber. When provided, the exploring strategy is made to explore the provided
group number. Must be a vector of two integer value representing the row group number and the column group number.

**ncores**

Number of parallel jobs to launch different EM initializations. By default detectCores(). This parameter can be changed by accessing the field ncores of the object. This parameters is used only on Linux. Parallism is disabled on other plateform. (Not working on Windows, not tested on Mac OS, not tested on *BSD.)

### Examples

```r
# Not run:
#
# SBM
#
#
# generation of one SBM network
npc <- 30 # nodes per class
Q <- 3 # classes
n <- npc * Q # nodes
Z <- diag(Q) %x% matrix(1, npc, 1)
Mu <- 20 * matrix(runif(Q * Q), Q, Q)
Y1 <- matrix(runif(n * n), n, n)
Y2 <- matrix(runif(n * n), n, n)
M <- matrix(rnorm(n * n, sd = 5), n, n) + Z %*% Mu %*% t(Z) + 4.2 * Y1 - 1.6 * Y2  ## adjacency matrix

# estimation
my_model <- BM_gaussian_covariates("SBM", M, list(Y1, Y2) )
my_model$estimate()
which.max(my_model$ICL)

# SBM symmetric
#
# generation of one SBM_sym network
npc <- 30 # nodes per class
Q <- 3 # classes
n <- npc * Q # nodes
Z <- diag(Q) %x% matrix(1, npc, 1)
Mu[lower.tri(Mu)] <- t(Mu)[lower.tri(Mu)]
Y1 <- matrix(runif(n * n), n, n)
Y2 <- matrix(runif(n * n), n, n)
Y1[lower.tri(Y1)] <- t(Y1)[lower.tri(Y1)]
Y2[lower.tri(Y2)] <- t(Y2)[lower.tri(Y2)]
M <- matrix(rnorm(n * n, sd = 5), n, n) + Z %*% Mu %*% t(Z) + 4.2 * Y1 - 1.6 * Y2  ## adjacency matrix
M[lower.tri(M)] <- t(M)[lower.tri(M)]

# estimation
my_model <- BM_gaussian_covariates("SBM_sym", M, list(Y1, Y2) )
my_model$estimate()
```
which.max(my_model$ICL)

## LBM
##

## generation of one LBM network
npc <- c(50,40) # nodes per class
Q <- c(2,3) # classes
n <- npc * Q # nodes
Z1<-diag(Q[1])%x%matrix(1,npc[1],1)
Z2<-diag(Q[2])%x%matrix(1,npc[2],1)
Mu<-20*matrix(runif(Q[1]*Q[2]),Q[1],Q[2])
Y1 <- matrix(runif(n[1]*n[2]),n[1],n[2])
Y2 <- matrix(runif(n[1]*n[2]),n[1],n[2])
M<-matrix(rnorm(n[1]*n[2],sd=5),n[1],n[2])+Z1%*%Mu%*%t(Z2)+4.2*Y1-1.6*Y2 # adjacency matrix

## estimation
my_model <- BM_gaussian_covariates("LBM",M,list(Y1,Y2))
my_model$estimate()
which.max(my_model$ICL)

## End(Not run)

---

**BM_gaussian_multivariate**

Perform estimation on blockmodels for multivariate gaussian probability distribution

---

**Description**

With the provided network and blockmodel type, estimate number of groups, parameters and node membership

**Usage**

```
## S4 method for signature 'new'
BM_gaussian_multivariate(
  membership_type,adj,
  verbosity=6,
  autosave='',
  plotting=character(0),
  exploration_factor=1.5,
  exploration_direction=numeric(0),
  explore_min=4,
  explore_max=Inf,
  ncores=detectCores())
```
Arguments

- `membership_type`: The type of node membership, i.e. 'SBM', 'SBM_sym' or 'LBM'
- `adj`: The list of adjacency matrices. All matrices must have the same size
- `verbosity`: The verbosity level, 0 means quiet. Level 1 display the phase of reinitialization. Level 2 display the level 1 and the ascending and descending phase for the number of groups. Level 3 display the level 2 and the number current number of groups which is estimated. Level 4 display the level 3 and the steps inside the estimation. Level 5 display the level 4, the current status of parallel running jobs and the current sub-step. Level 6 display level 5 and informations about ICL criteria found. Default is level 6. This parameter can be changed by accessing to the field $verbosity of the object.
- `autosave`: If `autosave != "`, after each estimation, the model object is writed into file `autosave`. The model object is readable by the function `readRDS`. Use-it for long computation to allow restarting the estimation on system crash. You can use it to alanyze the partial results when the estimation is running. This parameter can be changed by accessing to the field $autosave of the object.
- `plotting`: Control plot of ICL values while the estimation is running. If `plotting==character(0)` (the default), plots are done on screen, if `plotting=="", no plot are done, if plotting is a filename, plots are done in this filename. This parameter can be changed by accessing the field $plotting of the object.
- `exploration_factor`: Control the exploration of the number of groups. The exploration is stop when the number of groups reach exploration factor times the current maximum. By default 1.5. This parameter can be changed by accessing the field $exploration_factor of the object.
- `explore_min`: Explore to the `explore_min` number of groups even if the `exploration_factor` rule is satisfied. By default 4. This parameter can be changed by accessing the field $explore_min of the object.
- `explore_max`: Stop exploration after `explore_max` number of group in any case. By default Inf. This parameter can be changed by accessing the field $explore_max of the object.
- `exploration_direction`: Only for LBM membership. Control the exploration direction for groups number. When provided, the exploring strategy is made to explore the provided group number. Must be a vector of two integer value representing the row group number and the column group number.
- `ncores`: Number of parallel jobs to launch different EM intializations. By default `detectCores()`. This parameter can be changed by accessing the field $ncores of the object. This parameters is used only on Linux. Parallism is disabled on other plateform. (Not working on Windows, not tested on Mac OS, not tested on *BSD.)

Examples

```r
## Not run:
```
## SBM

## generation of one SBM network

```r
npc <- 30  # nodes per class
Q <- 3     # classes
n <- npc * Q  # nodes
Z <- diag(Q) %x% matrix(1, npc, 1)
Mu1 <- 4 * matrix(runif(Q * Q), Q, Q)
Mu2 <- 4 * matrix(runif(Q * Q), Q, Q)
Noise1 <- matrix(rnorm(n * n, sd = 1), n, n)
Noise2 <- matrix(rnorm(n * n, sd = 1), n, n)
M1 <- Z %*% Mu1 %*% t(Z) + Noise1
M2 <- Z %*% Mu2 %*% t(Z) + 10 * Noise1 + Noise2
```

## estimation

```r
my_model <- BM_gaussian_multivariate("SBM", list(M1, M2) )
my_model$estimate()
which.max(my_model$ICL)
```

## SBM symmetric

## generation of one SBM_sym network

```r
npc <- 30  # nodes per class
Q <- 3     # classes
n <-npc * Q  # nodes
Z <- diag(Q) %x% matrix(1, npc, 1)
Mu1 <- 4 * matrix(runif(Q * Q), Q, Q)
Mu2 <- 4 * matrix(runif(Q * Q), Q, Q)
Noise1 <- matrix(rnorm(n * n, sd = 1), n, n)
Noise2 <- matrix(rnorm(n * n, sd = 1), n, n)
M1 <- Z %*% Mu1 %*% t(Z) + Noise1
M2 <- Z %*% Mu2 %*% t(Z) + 10 * Noise1 + Noise2
M1[lower.tri(M1)] <- t(M1)[lower.tri(M1)]
M2[lower.tri(M2)] <- t(M2)[lower.tri(M2)]
```

## estimation

```r
my_model <- BM_gaussian_multivariate("SBM_sym", list(M1, M2) )
my_model$estimate()
which.max(my_model$ICL)
```

## LBM

## generation of one LBM network

```r
npc <- c(50, 40)  # nodes per class
Q <- c(2, 3)     # classes
n <- npc * Q  # nodes
```
Z1 <- diag(Q[1]) %x% matrix(1, npc[1], 1)
Z2 <- diag(Q[2]) %x% matrix(1, npc[2], 1)
Mu1 <- 4 * matrix(runif(Q[1] * Q[2]), Q[1], Q[2])
Mu2 <- 4 * matrix(runif(Q[1] * Q[2]), Q[1], Q[2])
Noise1 <- matrix(rnorm(n[1] * n[2], sd=1), n[1], n[2])
Noise2 <- matrix(rnorm(n[1] * n[2], sd=1), n[1], n[2])
M1 <- Z1 %x% Mu1 %x% t(Z2) + Noise1  # adjacency
M2 <- Z1 %x% Mu2 %x% t(Z2) + 10 * Noise1 + Noise2  # adjacency

## estimation
my_model <- BM_gaussian_multivariate("LBM", list(M1, M2) )
my_model$estimate()
which.max(my_model$ICL)

## End(Not run)
The verbosity level, 0 means quiet. Level 1 display the phase of reinitialization. Level 2 display the level 1 and the ascending and descending phase for the number of groups. Level 3 display the level 2 and the number current number of groups which is estimated. Level 4 display the level 3 and the steps inside the estimation. Level 5 display the level 4, the current status of parallel running jobs and the current sub-step. Level 6 display level 5 and informations about ICL criteria found. Default is level 6. This parameter can be changed by accessing to the field $verbosity of the object.

If $autosave !=", after each estimation, the model object is writed into file $autosave. The model object is readable by the function readRDS. Use-it for long computation to allow restarting the estimation on system crash. You can use it to alanyze the partial results when the estimation is running. This parameter can be changed by accessing to the field $autosave of the object.

Control plot of ICL values while the estimation is running. If $plotting==character(0) (the default), plots are done on screen, if $plotting=="", no plot are done, if plotting is a filename, plots are done in this filename. This parameter can be changed by accessing the field $plotting of the object.

Control the exploration of the number of groups. The exploration is stop when the number of groups reach exploration factor times the current maximum. By default 1.5. This parameter can be changed by accessing the field $exploration_factor of the object.

Explore to the explore_min number of groups even if the exploration_factor rule is satisfied. By default 4. This parameter can be changed by accessing the field $explore_min of the object.

Stop exploration after explore_max number of group in any case. By default Inf. This parameter can be changed by accessing the field $explore_max of the object.

Only for LBM membership. Control the exploration direction for groups number. When provided, the exploring strategy is made to explore the provided group number. Must be a vector of two integer value representing the row group number and the column group number.

Number of parallel jobs to launch different EM initializations. By default detectCores(). This parameter can be changed by accessing the field $ncores of the object. This parameters is used only on Linux. Parallism is disabled on other plateform. (Not working on Windows, not tested on Mac OS, not tested on *BSD.)

Examples

```r
## Not run:
#
## SBM
##```
## generation of one SBM network

```r
npc <- 30  # nodes per class
Q <- 3  # classes
n <- npc * Q  # nodes
Z <- diag(Q) %x% matrix(1, npc, 1)
Mu1 <- 8 * matrix(runif(Q * Q), Q, Q)
Mu2 <- 8 * matrix(runif(Q * Q), Q, Q)
M1 <- matrix(rnorm(n * n, sd = 5), n, n) + Z %*% Mu1 %*% t(Z)  # adjacency
M2 <- matrix(rnorm(n * n, sd = 10), n, n) + Z %*% Mu2 %*% t(Z)  # adjacency
```

## estimation

```r
my_model <- BM_gaussian_multivariate_independent("SBM", list(M1, M2) )
my_model$estimate()
which.max(my_model$ICL)
```

## SBM symmetric

## generation of one SBM_sym network

```r
npc <- 30  # nodes per class
Q <- 3  # classes
n <- npc * Q  # nodes
Z <- diag(Q) %x% matrix(1, npc, 1)
Mu1 <- 8 * matrix(runif(Q * Q), Q, Q)
Mu2 <- 8 * matrix(runif(Q * Q), Q, Q)
Mu1[lower.tri(Mu1)] <- t(Mu1)[lower.tri(Mu1)]
Mu2[lower.tri(Mu2)] <- t(Mu2)[lower.tri(Mu2)]
M1 <- matrix(rnorm(n * n, sd = 5), n, n) + Z %*% Mu1 %*% t(Z)  # adjacency
M2 <- matrix(rnorm(n * n, sd = 10), n, n) + Z %*% Mu2 %*% t(Z)  # adjacency
M1[lower.tri(M1)] <- t(M1)[lower.tri(M1)]
M2[lower.tri(M2)] <- t(M2)[lower.tri(M2)]
```

## estimation

```r
my_model <- BM_gaussian_multivariate_independent("SBM_sym", list(M1, M2) )
my_model$estimate()
which.max(my_model$ICL)
```

## LBM

## generation of one LBM network

```r
npc <- c(50, 40)  # nodes per class
Q <- c(2, 3)  # classes
n <- npc * Q  # nodes
Z1 <- diag(Q[1]) %x% matrix(1, npc[1], 1)
Z2 <- diag(Q[2]) %x% matrix(1, npc[2], 1)
Mu1 <- 8 * matrix(runif(Q[1] * Q[2]), Q[1], Q[2])
Mu2 <- 8 * matrix(runif(Q[1] * Q[2]), Q[1], Q[2])
M1 <- matrix(rnorm(n[1] * n[2], sd = 5), n[1], n[2]) + Z1 %*% Mu1 %*% t(Z2)  # adjacency
M2 <- matrix(rnorm(n[1] * n[2], sd = 10), n[1], n[2]) + Z1 %*% Mu2 %*% t(Z2)  # adjacency
```

## estimation

```r
my_model <- BM_gaussian_multivariate_independent("LBM", list(M1, M2) )
my_model$estimate()
which.max(my_model$ICL)
```
## estimation

```r
my_model <- BM_gaussian_multivariate_independent("LBM", list(M1, M2) )
my_model$estimate()
which.max(my_model$ICL)
```

```r
## End(Not run)
```

---

**BM_gaussian_multivariate_independent_homoscedastic**

*Perform estimation on blockmodels for multivariate independent homoscedastic gaussian probability distribution*

---

### Description

With the provided network and blockmodel type, estimate number of groups, parameters and node membership

### Usage

```r
## S4 method for signature 'new'
BM_gaussian_multivariate_independent_homoscedastic(
  membership_type,
  adj,
  verbosity=6,
  autosave='',
  plotting=character(0),
  exploration_factor=1.5,
  exploration_direction=numeric(0),
  explore_min=4,
  explore_max=Inf,
  ncores=detectCores())
```

### Arguments

- **membership_type**
  - The type of node membership, i.e. 'SBM', 'SBM_sym' or 'LBM'

- **adj**
  - The list of adjacency matrices. All matrices must have the same size

- **verbosity**
  - The verbosity level, 0 means quiet. Level 1 display the phase of reinitialization. Level 2 display the level 1 and the ascending and descending phase for the number of groups. Level 3 display the level 2 and the number current number of groups which is estimated. Level 4 display the level 3 and the steps inside the estimation. Level 5 display the level 4, the current status of parallel running jobs and the current sub-step. Level 6 display level 5 and informations about ICL criteria found. Default is level 6. This parameter can be changed by accessing to the field $verbosity of the object.
autosave If `autosave != "`, after each estimation, the model object is writed into file `autosave`. The model object is readable by the function `readRDS`. Use-it for long computation to allow restarting the estimation on system crash. You can use it to anayze the partial results when the estimation is running. This parameter can be changed by accessing to the field `$autosave` of the object.

plotting Control plot of ICL values while the estimation is running. If `plotting==character(0)` (the default), plots are done on screen, if `plotting=="`, no plot are done, if `plotting` is a filename, plots are done in this filename. This parameter can be changed by accessing the field `$plotting` of the object.

exploration_factor Control the exploration of the number of groups. The exploration is stop when the number of groups reach exploration_factor times the current maximum. By default 1.5. This parameter can be changed by accessing the field `$exploration_factor` of the object.

explore_min Explore to the `explore_min` number of groups even if the exploration_factor rule is satisfied. By default 4. This parameter can be changed by accessing the field `$explore_min` of the object.

explore_max Stop exploration after `explore_max` number of group in any case. By default Inf. This parameter can be changed by accessing the field `$explore_max` of the object.

exploration_direction Only for LBM membership. Control the exploration direction for groups number. When provided, the exploring strategy is made to explore the provided group number. Must be a vector of two integer value representing the row group number and the column group number.

ncores Number of parallel jobs to launch different EM initializations. By default detectCores(). This parameter can be changed by accessing the field `$ncores` of the object. This parameters is used only on Linux. Parallism is disabled on other plateform. (Not working on Windows, not tested on Mac OS, not tested on *BSD.)

Examples

```r
## Not run:
##
## ## SBM
##
## ## generation of one SBM network
npc <- 30 # nodes per class
Q <- 3 # classes
n <- npc * Q # nodes
Z <- diag(Q) %x% matrix(1, npc, 1)
Mu1 <- 4 * matrix(runif(Q * Q), Q, Q)
Mu2 <- 4 * matrix(runif(Q * Q), Q, Q)
M1 <- matrix(rnorm(n * n, sd = 5), n, n) + Z %*% Mu1 %*% t(Z) # adjacency
M2 <- matrix(rnorm(n * n, sd = 5), n, n) + Z %*% Mu2 %*% t(Z) # adjacency
```
## estimation
my_model <- BM_gaussian_multivariate_independent_homoscedastic("SBM", list(M1, M2) )
my_model$estimate()
which.max(my_model$ICL)

##
## SBM symmetric
##

## generation of one SBM_sym network
npc <- 30 # nodes per class
Q <- 3 # classes
n <- npc * Q # nodes
Z <- diag(Q) %x% matrix(1, npc, 1)
Mu1 <- 4 * matrix(runif(Q * Q), Q, Q)
Mu2 <- 4 * matrix(runif(Q * Q), Q, Q)
Mu1[lower.tri(Mu1)] <- t(Mu1)[lower.tri(Mu1)]
Mu2[lower.tri(Mu2)] <- t(Mu2)[lower.tri(Mu2)]
M1 <- matrix(rnorm(n * n, sd = 5), n, n) + Z %*% Mu1 %*% t(Z) # adjacency
M2 <- matrix(rnorm(n * n, sd = 5), n, n) + Z %*% Mu2 %*% t(Z) # adjacency
M1[lower.tri(M1)] <- t(M1)[lower.tri(M1)]
M2[lower.tri(M2)] <- t(M2)[lower.tri(M2)]

## estimation
my_model <- BM_gaussian_multivariate_independent_homoscedastic("SBM_sym", list(M1, M2) )
my_model$estimate()
which.max(my_model$ICL)

## LBM
##

## generation of one LBM network
npc <- c(50, 40) # nodes per class
Q <- c(2, 3) # classes
n <- npc * Q # nodes
Z1 <- diag(Q[1]) %x% matrix(1, npc[1], 1)
Z2 <- diag(Q[2]) %x% matrix(1, npc[2], 1)
Mu1 <- 4 * matrix(runif(Q[1] * Q[2]), Q[1], Q[2])
Mu2 <- 4 * matrix(runif(Q[1] * Q[2]), Q[1], Q[2])
M1 <- matrix(rnorm(n[1] * n[2], sd = 5), n[1], n[2]) + Z1 %*% Mu1 %*% t(Z2) # adjacency
M2 <- matrix(rnorm(n[1] * n[2], sd = 5), n[1], n[2]) + Z1 %*% Mu2 %*% t(Z2) # adjacency

## estimation
my_model <- BM_gaussian_multivariate_independent_homoscedastic("LBM", list(M1, M2) )
my_model$estimate()
which.max(my_model$ICL)

## End(Not run)
BM_poisson

Perform estimation on blockmodels for poisson probability distribution

Description

With the provided network and blockmodel type, estimate number of groups, parameters and node membership

Usage

```r
## S4 method for signature 'new'
BM_poisson(
  membership_type,
  adj,
  verbosity=6,
  autosave=''
)
```

Arguments

- `membership_type`:
  The type of node membership, i.e. 'SBM', 'SBM_sym' or 'LBM'
- `adj`:
  The adjacency matrix
- `verbosity`:
  The verbosity level, 0 means quiet. Level 1 display the phase of reinitialization. Level 2 display the level 1 and the ascending and descending phase for the number of groups. Level 3 display the level 2 and the number current number of groups which is estimated. Level 4 display the level 3 and the steps inside the estimation. Level 5 display the level 4, the current status of parallel running jobs and the current sub-step. Level 6 display level 5 and informations about ICL criteria found. Default is level 6. This parameter can be changed by accessing to the field $verbosity of the object.
- `autosave`:
  If autosave != '', after each estimation, the model object is writed into file autosave. The model object is readable by the function `readRDS`. Use-it for long computation to allow restarting the estimation on system crash. You can use it to alanyze the partial results when the estimation is running. This parameter can be changed by accessing to the field $autosave of the object.
- `plotting`:
  Control plot of ICL values while the estimation is running. If plotting==character(0) (the default), plots are done on screen, if plotting=='', no plot are done, if plotting is a filename, plots are done in this filename. This parameter can be changed by accessing the field $plotting of the object.
BM_poisson

exploration_factor
Control the exploration of the number of groups. The exploration is stop when the number of groups reach exploration factor times the current maximum. By default 1.5. This parameter can be changed by accessing the field $exploration\_factor\_of\_the\_object.$

explore_min
Explore to the explore_min number of groups even if the exploration_factor rule is satisfied. By default 4. This parameter can be changed by accessing the field $explore\_min\_of\_the\_object.$

explore_max
Stop exploration after explore_max number of group in any case. By default Inf. This parameter can be changed by accessing the field $explore\_max\_of\_the\_object.$

exploration_direction
Only for LBM membership. Control the exploration direction for groups number. When provided, the exploring strategy is made to explore the provided group number. Must be a vector of two integer value representing the row group number and the column group number.

ncores
Number of parallel jobs to launch different EM intializations. By default detectCores(). This parameter can be changed by accessing the field $n\_cores\_of\_the\_object.$ This parameters is used only on Linux. Parallism is disabled on other plateform. (Not working on Windows, not tested on Mac OS, not tested on *BSD.)

Examples

```r
## Not run:
#
# SBM
#
## generation of one SBM network
npc <- 30 # nodes per class
Q <- 3 # classes
n <- npc * Q # nodes
Z <- diag(Q) %x% matrix(1, npc, 1)
L <- 70 * matrix(runif(Q * Q), Q, Q)
M_in_expectation <- Z %*% L %*% t(Z)
M <- matrix(
  rpois(
    length(as.vector(M_in_expectation)),
    as.vector(M_in_expectation))
  , n, n)

## estimation
my_model <- BM_poisson("SBM", M )
my_model$estimate()
which.max(my_model$ICL)

##
## SBM symmetric
```
BM_poisson_covariates

```r
##
## generation of one SBM_sym network
npc <- 30 # nodes per class
Q <- 3 # classes
n <- npc * Q # nodes
Z <- diag(Q) %x% matrix(1, npc, 1)
L <- 70 * matrix(rnorm(Q * Q), Q, Q)
L[lower.tri(L)] <- t(L)[lower.tri(L)]
M_in_expectation <- Z %*% L %*% t(Z)
M <- matrix(
    rpois(
        length(as.vector(M_in_expectation)),
        as.vector(M_in_expectation))
    , n, n)
M[lower.tri(M)] <- t(M)[lower.tri(M)]

## estimation
my_model <- BM_poisson("SBM_sym", M )
my_model$estimate()
which.max(my_model$ICL)

## LBM
##
## generation of one LBM network
npc <- c(50, 40) # nodes per class
Q <- c(2, 3) # classes
n <- npc * Q # nodes
Z1 <- diag(Q[1]) %x% matrix(1, npc[1], 1)
Z2 <- diag(Q[2]) %x% matrix(1, npc[2], 1)
L <- 70 * matrix(rnorm(Q[1] * Q[2]), Q[1], Q[2])
M_in_expectation <- Z1 %*% L %*% t(Z2)
M <- matrix(
    rpois(
        length(as.vector(M_in_expectation)),
        as.vector(M_in_expectation))
    , n[1], n[2])

## estimation
my_model <- BM_poisson("LBM", M )
my_model$estimate()
which.max(my_model$ICL)

## End(Not run)
```

BM_poisson_covariates  Perform estimation on blockmodels for poisson probability distribution with covariates
**Description**

With the provided network and blockmodel type, estimate number of groups, parameters and node membership, and impact vector of covariates.

**Usage**

```r
## S4 method for signature 'new'
BM_poisson_covariates(
  membership_type, 
  adj, 
  covariates, 
  verbosity=6, 
  autosave='', 
  plotting=character(0), 
  exploration_factor=1.5, 
  exploration_direction=numeric(0), 
  explore_min=4, 
  explore_max=Inf, 
  ncores=detectCores())
```

**Arguments**

- **membership_type**
  The type of node membership, i.e. ‘SBM’, ‘SBM_sym’ or ‘LBM’

- **adj**
  The adjacency matrix

- **covariates**
  Covariates matrix, or list of covariates matrices. Covariates matrix must have the same size than the adjacency matrix.

- **verbosity**
  The verbosity level, 0 means quiet. Level 1 display the phase of reinitialization. Level 2 display the level 1 and the ascending and descending phase for the number of groups. Level 3 display the level 2 and the number current number of groups which is estimated. Level 4 display the level 3 and the steps inside the estimation. Level 5 display the level 4, the current status of parallel running jobs and the current sub-step. Level 6 display level 5 and informations about ICL criteria found. Default is level 6. This parameter can be changed by accessing to the field $verbosity of the object.

- **autosave**
  If autosave != ”, after each estimation, the model object is writed into file autosave. The model object is readable by the function readRDS. Use-it for long computation to allow restarting the estimation on system crash. You can use it to analyze the partial results when the estimation is running. This parameter can be changed by accessing to the field $autosave of the object.

- **plotting**
  Control plot of ICL values while the estimation is running. If plotting==character(0) (the default), plots are done on screen, if plotting==”, no plot are done, if plotting is a filename, plots are done in this filename. This parameter can be changed by accessing the field $plotting of the object.

- **exploration_factor**
  Control the exploration of the number of groups. The exploration is stop when the number of groups reach exploration factor times the current maximum. By
default 1.5. This parameter can be changed by accessing the field $explo-
ration_factor of the object.

explore_min Explore to the explore_min number of groups even if the exploration_factor rule
is satisfied. By default 4. This parameter can be changed by accessing the field
$explore_min of the object.

explore_max Stop exploration after explore_max number of group in any case. By default
Inf. This parameter can be changed by accessing the field $explore_max of the
object.

exploration_direction

Only for LBM membership. Control the exploration direction for groups num-
ber. When provided, the exploring strategy is made to explore the provided
group number. Must be a vector of two integer value representing the row group
number and the column group number.

core Number of parallel jobs to launch different EM intializations. By default de-
tectCores(). This parameter can be changed by accessing the field $ncore of
the object. This parameters is used only on Linux. Parallism is disabled on
other plateform. (Not working on Windows, not tested on Mac OS, not tested
on *BSD.)

Examples

```R
## Not run:
##
## SBM
##
## generation of one SBM network
npc <- 30 # nodes per class
Q <- 3 # classes
n <- npc * Q # nodes
Z <- diag(Q) %x% matrix(1, npc, 1)
L <- 70 * matrix(runif(Q * Q), Q, Q)
M_in_expectation_without_covariates <- Z %x% L %x% t(Z)
Y1 <- matrix(runif(n * n), n, n)
Y2 <- matrix(runif(n * n), n, n)
M_in_expectation <- M_in_expectation_without_covariates * exp(4.2 * Y1 - 1.2 * Y2)
M <- matrix(
  rpois(
    length(as.vector(M_in_expectation)),
    as.vector(M_in_expectation))
  , n, n)

## estimation
my_model <- BM_poisson_covariates("SBM", M, list(Y1, Y2))
my_model$estimate()
which.max(my_model$ICL)
```

## SBM symmetric
## generation of one SBM_sym network, we re-use one produced for SBM

tpc <- 30 # nodes per class
Q <- 3 # classes
n <- tpc * Q # nodes
Z <- diag(Q) %x% matrix(1, npc, 1)
L <- 70 * matrix(runif(Q * Q), Q, Q)
L[lower.tri(L)] <- t(L)[lower.tri(L)]
M_in_expectation_without_covariates <- Z * L * t(Z)
Y1 <- matrix(runif(n * n), n, n)
Y2 <- matrix(runif(n * n), n, n)
Y1[lower.tri(Y1)] <- t(Y1)[lower.tri(Y1)]
Y2[lower.tri(Y2)] <- t(Y2)[lower.tri(Y2)]
M_in_expectation <- M_in_expectation_without_covariates * exp(4.2 * Y1 - 1.2 * Y2)
M <- matrix(
  rpois(
    length(as.vector(M_in_expectation)),
    as.vector(M_in_expectation))
  , n, n)
M[lower.tri(M)] <- t(M)[lower.tri(M)]

## estimation
my_model <- BM_poisson_covariates("SBM_sym", M, list(Y1, Y2))
my_model$estimate()
which.max(my_model$ICL)

## LBM

## generation of one LBM network

tpc <- c(50, 40) # nodes per class
Q <- c(2, 3) # classes
n <- tpc * Q # nodes
Z1 <- diag(Q[1]) %x% matrix(1, npc[1], 1)
Z2 <- diag(Q[2]) %x% matrix(1, npc[2], 1)
L <- 70 * matrix(runif(Q[1] * Q[2]), Q[1], Q[2])
M_in_expectation_without_covariates <- Z1 * L * t(Z2)
Y1 <- matrix(runif(n[1] * n[2]), n[1], n[2])
Y2 <- matrix(runif(n[1] * n[2]), n[1], n[2])
M_in_expectation <- M_in_expectation_without_covariates * exp(4.2 * Y1 - 1.2 * Y2)
M <- matrix(
  rpois(
    length(as.vector(M_in_expectation)),
    as.vector(M_in_expectation))
  , n[1], n[2])

## estimation
my_model <- BM_poisson_covariates("LBM", M, list(Y1, Y2))
my_model$estimate()
which.max(my_model$ICL)
## End (Not run)
Index

BM_bernoulli, 2
BM_bernoulli_covariates, 4
BM_bernoulli_covariates_fast, 7
BM_bernoulli_multiplex, 10
BM_gaussian, 14
BM_gaussian_covariates, 16
BM_gaussian_multivariate, 19
BM_gaussian_multivariate_independent, 22
BM_gaussian_multivariate_independent_homoscedastic, 25
BM_poisson, 28
BM_poisson_covariates, 30