Package: VBLPCM (via r-universe)

August 22, 2024

Type Package

Title Variational Bayes Latent Position Cluster Model for Networks

Version 2.4.9

Date 2023-03-22

Author Michael Salter-Townshend

Maintainer Michael Salter-Townshend <mike.saltertownshend@gmail.com>

Description Fit and simulate latent position and cluster models for
network data, using a fast Variational Bayes approximation
developed in Salter-Townshend and Murphy (2013)
<doi:10.1016/j.csda.2012.08.004>.

Depends ergm, network

Imports mclust, sna

SystemRequirements Gnu Scientific Library

URL https://www.r-project.org, https://mststats.github.io/#software

License GPL (>= 2)

LazyLoad yes

NeedsCompilation yes

Repository CRAN

Date/Publication 2023-03-22 16:10:02 UTC

Contents

VBLPCM–package   *VBLPCM: Variational Bayes for the Latent Position Cluster Model for networks*

Description

A faster approximate alternative to using latentnet. Interfaces C code to fit a Variational Bayes approximation to the posterior for the Latent Position Cluster Model for networks.

Details

Package:	VBLPCM
Type:	Package
Version:	2.4.9
Date:	2023-03-22
License:	GPL (>=2)
LazyLoad:	yes

This package is designed to be used as an alternative to the latentnet package when network size computationally prohibits latentnet. It uses a Variational Bayesian Expectation Maximisation algorithm to compute a closed-form approximation to the posterior that the ergmm function in latentnet samples from. It may be thought of as an intermediary approximation that is more accurate than the two-stage MLE fit provided by latentnet but a faster approximation to the MCMC sampler provided by latentnet. In fact, the VB iterations also converge quicker than the two-stage MLE.

VBLPCM can also take advantage of the stratified sampler of Adrian Raftery, Xiaoyue Niu, Peter Hoff and Ka Yee Yeung. This approximation to the (log)likelihood allows for even larger networks

to be analysed (see tech report below). Rather than using a fixed number of "controls" per geodesic distance we set a probability of sampling each non-link at each level.

We also provide four choices of model; these are "plain" and three with random node-specific social effects. "rsender" for sender random effects, "rreceiver" for receiver random effects and "rsocial" for both. For undirected networks only "plain" or "rsocial" may be chosen.

References

Michael Salter-Townshend and Thomas Brendan Murphy (2013). "Variational Bayesian Inference for the Latent Position Cluster Model." Computational Statistics and Data Analysis, volume 57, number 1, pages 661-671. DOI=10.1016/j.csda.2012.08.004

Pavel N. Krivitsky and Mark S. Handcock (2008). "Fitting Latent Cluster Models for Social Networks with latentnet." Journal of Statistical Software, number 5, volume 24, pages 1-23.

Mark S. Handcock, Adrian E. Raftery and Jeremy Tantrum (2007). "Model-Based Clustering for Social Networks." Journal of the Royal Statistical Society: Series A (Statistics in Society), 170(2), 301-354.

Adrian Raftery, Xiaoyue Niu, Peter Hoff and Ka Yee Yeung (2012). "Fast Inference for the Latent Space Network Model Using a Case-Control Approximate Likelihood." Journal of Computational and Graphical Statistics. doi: 10.1080/10618600.2012.679240

Sucharita Gopal (2007). "The Evolving Social Geography of Blogs" Societies and Cities in the Age of Instant Access Berlin:Springer, 275–294

See Also

vblpcmstart vblpcmfit

Examples

```
### Sampson's monks with sender random effects ###
data(sampson,package="VBLPCM")
v.start<-vblpcmstart(samplike,G=3,model="rreceiver",LSTEPS=1e3)
v.fit<-vblpcmfit(v.start,STEPS=20)
### plot the mean posterior positions ###
plot(v.fit, R2=0.05,main="Sampson's Monks: VB with Receiver Random Effects")
### Who's in each group?  ###
vblpcmgroups(v.fit)

### Look at a goodness-of-fit plot ###
plot(gof(v.fit,GOF=~distance))

### create a matrix of link posterior probabilities given the fitted model ###
probs<-predict.vblpcm(v.fit)
### create a boxplot goodness-of-fit graphic ###
boxplot(split(probs,as.sociomatrix(samplike)))

### run a bigger example, using the likelihood sampler set to 0.1 ###
## Not run:
data(aids,package="VBLPCM")
v.start<-vblpcmstart(aids.net,G=7,model="rsender",d=3)
```

```
use the case-control sampler with 10 controls per case
v.fit<-vblpcmfit(v.start,NC=10)
plot the mean posterior positions ###
plot(v.fit, R2=0.1,main="Aids Blogs with Sender Random Effects")

### Use ROC / AUC to get a measure of model fit to the data ###
vblpcmroc(v.fit)

## End(Not run)
```

---

aids.net   *aids blogs data as a "network" object*

Description

Network of citations among blogs related to AIDS, patients, and their support networks, collected by Gopal, over a three-day period in August 2005. A directed graph representing the pattern of citation among 146 unique blogs related to AIDS, patients, and their support networks, collected by Gopal over a randomly selected three-day period in August 2005. Vertices correspond to blogs. A directed edge from one blog to another indicates that the former had a link to the latter in their web page (more specifically, the former refers to the latter in their so-called 'blogroll').

Usage

```
data(aids)
```

Source

http://math.bu.edu/people/kolaczyk/datasets/AIDSBlog.zip

References

S. Gopal, "The evolving social geography of blogs," in Societies and CIties in the Age of Instant Access, H. Miller, Ed. Berlin:Springer, 2007, pp. 275-294

See Also

network, plot.network, VBLPCM

E_to_Y *create an adjacency matrix from an edgelist.*

Description

uses a call to C to transform edgelist to adjacency matrix.

Usage

```
E_to_Y(N, NE, directed, E)
```

Arguments

N	number of nodes
NE	number of edges
directed	logical indicator of directedness (TRUE=>directed, FALSE=>undirected
E	the input edgelist

Value

NxN sociomatrix / adjacency matrix

Author(s)

Michael Salter-Townshend

See Also

sociomatrix, Y_to_E

---

fruchterman_reingold *Perform Fruchterman-Reingold layout of a network in 2 or more dimensions.*

Description

This was written and incorporated into the VBLPCM package because the Fruchterman-Reingold routine in the network package only works in two dimensions.

Usage

```
fruchterman_reingold(net, D=2, steps=1000, repulserad=N^D, m=N*(D-1),
                        volume=N*(D-1))
```

Arguments

net	network object on which to perform Fruchterman-Reingold layout.
D	Desired dimension of the space in which to lay out the network.
steps	Number of desired iterations.
repulserad	The radius at which repulsion and attraction of linked nodes are equal.
m	The maximum change in position per iteration.
volume	The volume of space within which to position the nodes.

Value

An N*D matrix of coordinates.

Author(s)

Michael Salter-Townshend

See Also

log_like_forces

Examples

```
### 2D example
### load the aids blogs dataset
data(aids)
### perform the Fruchterman-Reingold layout
X<-fruchterman_reingold(aids.net, D=2, steps=1e3)
### plot the results
plot(X)

### 3D example
### load the aids blogs dataset
data(aids)
### perform the Fruchterman-Reingold layout
X<-fruchterman_reingold(aids.net, D=3, steps=1e3)
### Not run
### plot the results in 3D
# library(rgl)
# plot3d(X)
```

gof.vblpcm   *Goodness of fit based on simulations from the fitted object.*

Description

Create a goodness of fit statistics and plots based on the degree distributions of networks simulated fitted from a fitted variational approximation.

Usage

```
## S3 method for class 'vblpcm'
gof(object, ...,
        nsim=100,
        GOF=NULL,
        verbose=FALSE)
```

Arguments

object	fitted VBLPCM object; usually output from vblpcmfit() or vblpcmstart()
...	optional arguments for lower level functions
nsim	number of networks to simulate
GOF	formula; an R formula object, of the form ~ <model terms> specifying the statistics to use to diagnosis the goodness-of-fit of the model. They do not need to be in the model formula specified in formula, and typically are not. Examples are the degree distribution ("degree"), minimum geodesic distances ("dist"), and shared partner distributions ("espartners" and "dspartners"). For the details on the possible <model terms>, see ergm-terms.
verbose	Provide verbose information on the progress of the simulation.

Details

A sample of graphs is randomly drawn from the posterior of the vblpcmfit() result.

A plot of the summary measures may then be plotted using plot().

Author(s)

Michael Salter-Townshend

See Also

latentnet::gof.ergmm

Examples

```
data(sampson,package="VBLPCM")
v.start<-vblpcmstart(samplike,G=3,model="rreceiver",LSTEPS=1e3)
v.fit<-vblpcmfit(v.start,STEPS=20)
### plot the mean posterior positions
plot(v.fit, R2=0.05,main="Sampson's Monks: VB with Receiver Effects")
### Look at gof plots
plot(gof(v.fit,GOF=~distance,nsim=50))
```

---

 hops_to_hopslist   *create a handy matrix of vectors to store the hopslist*

Description

Designed for nternal use only; store the geodesic distances in a handy format Each node gets a
vector in the hopslist matrix. Each row describes a node and for each row: The first diam entries
state the number of nodes that are that distance away by shortest path where diam is the maximum
shortest path between two nodes (the graph diameter). eg if entry 3 in row 4 is a 5 then there are
exactly 5 nodes that are 4 hops away from node 3. This vector is followed by the indices of all the
nodes, grouped by the length of the shortest paths.

Usage

```
hops_to_hopslist(hops, diam, N)
```

Arguments

hops	matrix of geodesic distances
diam	diameter of the network
N	total number of nodes in the network

Author(s)

Michael Salter-Townshend

---

 log_like_forces   *create an initial configuration for the latent positions.*

Description

This performs an iterative relaxation type algorithm to approximately find the positions of the nodes
in the latent space that maximises the log-likelihood.

Usage

```
log_like_forces(net, D, X, B, m ,steps)
```

Arguments

`net`	network object on which to perform layout.
`D`	dimension of the latent space.
`X`	the initial guess for X
`B`	the intercept term.
`m`	usually N will suffice.
`steps`	maximum number of iteration steps.

Details

Usually only used internally in vblpcmstart()

Value

Matrix of latent positions X

Author(s)

Michael Salter-Townshend

See Also

igraph::layout.fruchterman.reingold

Examples

```
data(sampson)
N=network.size(samplike)
X=matrix(runif(N*2,-2,2),ncol=2)
XX=vblpcmcovs(N,"plain",as.sociomatrix(samplike))
out<-log_like_forces(samplike, 2, X, 0, m=N, 1e3)
plot(samplike,coord=out$X)
```

---

plot.vblpcm	*plot the posterior latent positions and groupings and network*

Description

Plot the network using the estimated positions with clustering. The nodes are plotted as pie-charts to show group membership probabilities. The group means are coloured crosses and the group standard deviations are shown with coloured circles.

Usage

```
## S3 method for class 'vblpcm'
plot(x, ..., R2 = 0.2, main = "Variational-Bayes Positions",
                 alpha = 0.5, colours=1:x$G, RET=FALSE)
```

Arguments

x	The fitted values; output from vblpcmfit()
...	optional arguments to be passed to lower level functions
R2	scaling factor for the size of each node in the plot
main	main title for the plot
alpha	transparency of the links
colours	colours of the groups
RET	whether to return the 2D postions of nodes and clusters

Details

Plots the latent positions and clustering of a network fitted via vblpcmfit() or vblpcmstart()

Each node appears in the latent space as a pie chart with segments size proportional to group memberships. The clusters are represented as circles in the latent space centred on the expected position of the group mean and with size proportional to the cluster standard deviation.

If applicable, the size of the pie charts represents the expected sociality effect of the node.

Author(s)

Michael Salter-Townshend

See Also

latentnet::plot.ergmm

---

predict.vblpcm	*Find all link probabilities*

Description

generate a matrix of link probabilities based on the fitted VB model.

Usage

```
## S3 method for class 'vblpcm'
predict(object, ...)
```

Arguments

object	The fitted values; output from vblpcmfit()
...	optional additional arguments.

Value

The posterior predictive link probabilities given the fitted object

Author(s)

Michael Salter-Townshend

Examples

```
data(sampson)
v.fit<-vblpcmfit(vblpcmstart(samplike,G=3))
### create a matrix of link posterior probabilities given the fitted model
probs<-predict.vblpcm(v.fit)
# show this graphically; separation of the boxes implies a good fit to the data
boxplot(split(probs,v.fit$Y),
        ylab=expression(paste("P(",Y[i][j],"=1)")),xlab=expression(paste(Y[i][j])))
```

---

print.vblpcm   *print the fitted vblpcm object*

Description

Print a vblpcm object.

Usage

```
## S3 method for class 'vblpcm'
print(x, ...)
```

Arguments

x   The fitted values; output from vblpcmfit()

...   optional arguments to be passed to lower level functions

Author(s)

Michael Salter-Townshend

See Also

latentnet::print.ergmm

sampson	*Cumulative network of positive affection within a monastery as a "network" object*

Description

Sampson (1969) recorded the social interactions among a group of monks while resident as an experimenter on vision, and collected numerous sociometric rankings. During his stay, a political "crisis in the cloister" resulted in the expulsion of four monks (Nos. 2, 3, 17, and 18) and the voluntary departure of several others - most immediately, Nos. 1, 7, 14, 15, and 16. (In the end, only 5, 6, 9, and 11 remained). Of particular interest is the data on positive affect relations ("liking"), in which each monk was asked if they had positive relations to each of the other monks.

The data were gathered at three times to capture changes in group sentiment over time. They were represent three time points in the period during which a new cohort entered the monastery near the end of the study but before the major conflict began.

Each member ranked only his top three choices on "liking". (Some subjects offered tied ranks for their top four choices). A tie from monk A to monk B exists if A nominated B as one of his three best friends at that that time point.

samplike is the time-aggregated network. It is the cumulative tie for "liking" over the three periods. For this, a tie from monk A to monk B exists if A nominated B as one of his three best friends at any of the three time points.

This data is standard in the social network analysis literature, having been modeled by Holland and Leinhardt (1981), Reitz (1982), Holland, Laskey and Leinhardt (1983), and Fienberg, Meyer, and Wasserman (1981), Hoff, Raftery, and Handcock (2002), etc. This is only a small piece of the data collected by Sampson.

Usage

```
data(sampson)
```

Source

Sampson, S.~F. (1968), *A novitiate in a period of change: An experimental and case study of relationships,* Unpublished ph.d. dissertation, Department of Sociology, Cornell University.

References

White, H.C., Boorman, S.A. and Breiger, R.L. (1976). *Social structure from multiple networks. I. Blockmodels of roles and positions.* American Journal of Sociology, 81(4), 730-780.

See Also

network, plot.network, ergmm

Examples

```
data(sampson)
plot(samplike)
```

---

simulated.network	*simulated.network*

Description

adjacency matrix simulated from the latent position cluster model with 3 well separated groups

Usage

```
data(simulated.network)
```

Source

Michael Salter-Townshend

See Also

network, plot.network, VBLPCM

---

summary.vblpcm	*summary of a fitted vblpcm object.*

Description

Summarise the output of a call to either vblpcmstart or vblpcmfit.

Usage

```
## S3 method for class 'vblpcm'
summary(object, ...)
```

Arguments

object	The fitted values; output from vblpcmstart() or vblpcmfit()
...	optional arguments to be passed to lower level functions

Author(s)

Michael Salter-Townshend

See Also

latentnet::summary.ergmm

---

vblpcmbic   *calculate the BIC for the fitted VBLPCM object*

Description

calculate the BIC for the fitted VBLPCM object

Usage

```
vblpcmbic(v.params)
```

Arguments

v.params   The fitted values; output from vblpcmfit()

Details

BIC = BIC(edges | positions) + BIC(positions | clusters) w/ BIC(edges | positions) = -2 loglikelihood + (P+1)log(number of edges) and BIC(positions | clusters) as per mclust

Value

The scalar value of the BIC

Author(s)

Michael Salter-Townshend

References

Mark S. Handcock, Adrian E. Raftery and Jeremy Tantrum (2007). "Model-Based Clustering for Social Networks." Journal of the Royal Statistical Society: Series A (Statistics in Society), 170(2), 301-354.

See Also

latentnet::summary.ergmm

Examples

```
data(sampson)
set.seed(1)
### plot the BIC for G=2,3,4 groups
gbic<-list(groups=NULL,bic=NULL)
for (g in 2:4)
  {
  v.fit<-vblpcmfit(vblpcmstart(samplike,G=g,LSTEPS=1e3),STEPS=20)
  gbic$groups[g]=v.fit$G
  gbic$bic[g]=v.fit$BIC$overall
```

```
   }
 plot(gbic$groups, gbic$bic, main="BIC results", xlab="# groups", ylab="BIC", t='b')
```

vblpcmcovs	*create the design matrix for the network analysis*

Description

Add intercept (column of ones) and degree-based covariates (if model is for sociality effects) to a user-supplied (default is NULL) edge covariates matrix of size N^2 rows and C columns where C is the number of covariates. Node covariates may be converted to difference-between-pairs for edges.

Usage

```
vblpcmcovs(N, model, Y, edgecovs=NULL, sendcovs=NULL, receivecovs=NULL,
                   socialcovs=NULL)
```

Arguments

N	number of nodes
model	model; may be "plain", "rreceiver", "rsender" or "rsocial". See Details.
Y	adjacency matrix
edgecovs	optional additional covariate / attribute data on the edges
sendcovs	optional additional covariate / attribute data on the nodes for links out
receivecovs	optional additional covariate / attribute data on the nodes for links in
socialcovs	optional additional covariate / attribute data on the nodes for links in and out

Details

Can be used to construct design matrices with edge covariates or node covariates and / or sociality effects. "rreceiver", "rsender" and "rsocial" model random social effects. Node covariates are differenced and treated as edge covariates.

Value

An edge design matrix that is Pe x N^2 and a node design matrix that is Pn x N where Pe is the number of edge covariates and Pn is the number of node covariates.

Author(s)

Michael Salter-Townshend

See Also

vblpcmstart

vblpcmdrawpie	*add a piechart of group memberships of a node to a network plot; taken mainly from latentnet equivalent*

Description

add a piechart of group memberships of a node to a network plot; taken mainly from latentnet equivalent

Usage

```
vblpcmdrawpie(center,radius,probs,n=50,colours=1:length(probs))
```

Arguments

center	where to postion the piechart
radius	radius of the piechart / node
probs	probability vector of cluster memberships
n	order of polygon to approximate a circle
colours	the colours used; default is from palette()

Note

Thanks to Pavel N. Krivitsky of the latentnet package as I copied this from there.

Author(s)

Michael Salter-Townshend

See Also

plot.vblpcm

vblpcmfit	*fit the variational model through EM type iterations*

Description

Perform optimisation of the variational parameters of the variational approximation to the posterior for the latent position cluster model for network data.

Usage

```
vblpcmfit(variational.start, STEPS = 50, maxiter = 100, tol=1e-6, NC=NULL,
          seed=NaN, d_vector=rep(TRUE,9))
```

Arguments

variational.start

         The starting configuration; use vblpcmstart() to generate this.

STEPS   Maximum number of iterations in the main VBEM loop.

maxiter   Maximum number of iterations for the internal univariate optimisation loops.

tol   tolerance of change in variational parameter updates below which the algorithm is deemed to have converged for that parameter.

NC   Number of non-links sampled in the case-control type sampler. Results in a speedup but loss of accuracy.

seed   Optional seed for the random number generator. Supplying NaN is equivalent to not supplying it. Supply a value so that results may be replicated.

d_vector   Optional logical vector specifying which sets of variational parameters are to be updated. See Details for more information.

Details

d_vector is a logical vector of length 9 that can be used to select which variational parameters are held fixed and which are updated. The parameters are in the following order: z (latent positions), sigma2 (variance of latent positions), lambda (membership probability matrix), eta (cluster centres), omega2 (cluster variances), alpha (cluster specific variance of nodes), nu (Dirichlet parameter for marginal cluster probabilities), xi (likelihood intercept term mean), psi2 (likelihood intercept term variance).

Value

A v.params list containing the fitted variational parameters for the latent positions, clustering membership probabilities, etc. conv indicated whether convergence was obtained within the specified number of iterations.

Author(s)

Michael Salter-Townshend

References

Michael Salter-Townshend and Thomas Brendan Murphy (2009). "Variational Bayesian Inference for the Latent Position Cluster Model." Workshop on Analyzing Networks and Learning with Graphs. Neural Information Processing Systems.

See Also

vblpcmstart, latentnet::ergmm

---

vblpcmgroups   *list the maximum VB a-posteriori group memberships.*

Description

Prints to screen the most likely a-posteriori membership of each node.

Usage

```
vblpcmgroups(v.params, colours)
```

Arguments

v.params   The fitted values; output from vblpcmfit()

colours   The colours to be used.

Value

Prints to screen of the most probable group membership for each node.

Author(s)

Michael Salter-Townshend

---

vblpcmKL   *print and returns the Kullback-Leibler divergence from the fitted*
                                    vblpcm object to the true LPCM posterior

---

Description

print and returns the Kullback-Leibler divergence from the fitted vblpcm object to the true LPCM
posterior

Usage

```
vblpcmKL(x)
```

Arguments

x   The fitted values; output from vblpcmfit() or vblpcmstart()

Details

The normalising constant of the posterior is unknown and therefore the Kullback-Leibler divergence
is missing a constant.

Author(s)

Michael Salter-Townshend

---

vblpcmroc   *ROC curve plot for vblpcmfit*

Description

Plot a Receiver Operating Curve to show model fit in terms of link prediction.

Usage

```
vblpcmroc(v.params, NUM=100)
```

Arguments

v.params   The fitted values; output from vblpcmfit()

NUM   The number of intervals on the roc curve

Details

A threshold is varied between zero and one. At each point the probability of a link between all pairs of nodes is calculated on the v.params argument containing a fitted vblpcm object. If greater than the threshold the link is "predicted" present, else it is "predicted" absent. A plot of the proportion of true and false positives for each threshold value is thus obtained.

Value

The Area Under the Curve (AUC). The closer to 1 the better the fit.

Author(s)

Michael Salter-Townshend

---

vblpcmstart	*Generate sensible starting configuration for the variational parameter set.*

Description

Uses fast methods to generate sensible and coherent values for the parameters of the variational method. There are returned as a list and that list may be passed directly to vblpcmfit(). User specification of the configuration is recommended as tweaks to this list only.

Usage

```
vblpcmstart(g.network,G=1,d=2,LSTEPS=5e3,model="plain", CLUST=0, B=NULL,
            lcc=TRUE, edgecovs=NULL,sendcovs=NULL,receivecovs=NULL,
            socialcovs=NULL,START="FR", seed=NaN)
```

Arguments

g.network	a network object
G	Desired number of groups
d	Desired dimensionality of the latent space
LSTEPS	Number of steps in the log-likelihood forces algorithm
model	model specified as "plain", "rreceiver", "rsender" or "rsocial". See vblpcmcovs for details.
CLUST	degree of push to clustering at the start
B	default intercept value
lcc	logical indicator. TRUE => analyze largest connected component of g.network only FALSE => analyze the whole network.
edgecovs	optional edge covariates.
sendcovs	optional sender node covariates.
receivecovs	optional receiver node covariates.
socialcovs	optional sociality node covariates.
START	what to start the initial positions with. "FR" for Fruchterman-Reingold. "geodesic" for geodesic distances. "laplace" for using the Graph Laplacian. "random" for random.
seed	Optional seed for the random number generator in R. Equivalent to using set.seed(seed). The default NaN value does not call set.seed().

Value

A v.params list containing the latent positions, clustering membership probabilities, etc.

Author(s)

Michael Salter-Townshend

See Also

vblpcmfit, vblpcmcovs

Examples

```
data(sampson)
### plot the mean posterior positions with initial estimations for variational parameters
plot(vblpcmstart(samplike,G=3),main="Sampson's Monks: VB Initial Values")
### plot the mean posterior positions with final estimations for variational parameters
plot(vblpcmfit(vblpcmstart(samplike,G=3)),main="Sampson's Monks: VB Solution")
```

Y_to_E   *calculate the edgelist for a given adjacency matrix*

Description

calls C code to quickly transform from adjacency to edgelist

Usage

```
Y_to_E(N, NE, directed, Y)
```

Arguments

N	number of nodes
NE	number of edges
directed	logical indicator of directedness; TRUE=>directed FALSE=>undirected
Y	input adjacency matrix

Value

An edgelist matrix E of size NE x 2

Author(s)

Michael Salter-Townshend

See Also

edgelist, E_to_Y

Y_to_M *calculate the missing edges as an edgelist from an adjacency matrix with NaNs indicating missing links*

Description

uses C code to quickly find all pairs of nodes for which we do not know whether there is a link or not, given an adjacency matrix with NaNs indicating unknown / unobserved linkage

Usage

```
Y_to_M(N, NM, directed, Y)
```

Arguments

N	number of nodes
NM	number of missing edges
directed	logical indicator of directedness; TRUE=>directed FALSE=>undirected
Y	input adjacency matrix

Value

A matrix of missing edges M

Author(s)

Michael Salter-Townshend

See Also

Y_to_E, E_to_Y, Y_to_nonE

Y_to_nonE *calculate a non-edge list from an adjacency matrix*

Description

uses C code to quickly calculate all non-edges as a two column matrix given an adjacency matrix. i.e. all zeros in the adjacency matrix will correspond to a row in the non-edgelist nonE

Usage

```
Y_to_nonE(N, NnonE, directed, Y)
```

Arguments

N	number of nodes
NnonE	number of non-edges
directed	logical indicator of directedness; TRUE=>directed FALSE=>undirected
Y	input adjacency matrix

Value

A matrix of the non-edges with NnonE rows and 2 columns where NnonE is the number of non-edges.

Author(s)

Michael Salter-Townshend

See Also

Y_to_E, Y_to_M, E_to_Y

Index