Package: randnet (via r-universe)

September 15, 2024

Type Package

Title Random Network Model Estimation, Selection and Parameter Tuning

Version 0.7

Date 2023-05-11

Description Model selection and parameter tuning procedures for a class of random network models. The model selection can be done by a general cross-validation framework called ECV from Li et. al. (2016) <arXiv:1612.04717> . Several other model-based and task-specific methods are also included, such as NCV from Chen and Lei (2016) <arXiv:1411.1715>, likelihood ratio method from Wang and Bickel (2015) <arXiv:1502.02069>, spectral methods from Le and Levina (2015) <arXiv:1507.00827>. Many network analysis methods are also implemented, such as the regularized spectral clustering (Amini et. al. 2013 <doi:10.1214/13-AOS1138>) and its degree corrected version and graphon neighborhood smoothing (Zhang et. al. 2015 <arXiv:1509.08588>). It also includes the consensus clustering of Gao et. al. (2014) <arXiv:1410.5837>, the method of moments estimation of nomination SBM of Li et. al. (2020) <arxiv:2008.03652>, and the network mixing method of Li and Le (2021) <arxiv:2106.02803>. It also includes the informative core-periphery data processing method of Miao and Li (2021) <arXiv:2101.06388>. The work to build and improve this package is partially supported by the NSF grants DMS-2015298 and DMS-2015134.

License GPL (>= 2)

Depends Matrix, entropy, AUC,sparseFLMM, mgcv

Imports methods, stats, poweRlaw, RSpectra, irlba,pracma,nnls,data.table

NeedsCompilation no

Author Tianxi Li [aut, cre], Elizeveta Levina [aut], Ji Zhu [aut], Can M. Le [aut]

Maintainer Tianxi Li <tianxili@virginia.edu>

Date/Publication 2023-05-20 07:30:02 UTC
Repository https://tianxili.r-universe.dev
RemoteUrl https://github.com/cran/randnet
RemoteRef HEAD
RemoteSha 9dcbcd5ef79d868a830a4a4be8d86507439e363e

Contents

`randnet-package` *Statistical modeling of random networks with model estimation, selection and parameter tuning*

Description

The package provides model fitting and estimation functions for some popular random network models. More importantly, it implements a general cross-validation framework for model selection and parameter tuning called ECV. Several other model selection methods are also included. The work to build and improve this package is partially supported by the NSF grants DMS-2015298 and DMS-2015134.

Details

Package:	randnet
Type:	Package
Version:	0.7
Date:	2023-05-11
License:	GPL (>= 2)

Author(s)

Tianxi Li, Elizaveta Levina, Ji Zhu, Can M. Le

Maintainer: Tianxi Li <tianxili@virginia.edu>

References

T. Li, E. Levina, and J. Zhu. Network cross-validation by edge sampling. Biometrika, 107(2), pp.257-276, 2020.

K. Chen and J. Lei. Network cross-validation for determining the number of communities in network data. Journal of the American Statistical Association, 113(521):241-251, 2018.

K. Rohe, S. Chatterjee, and B. Yu. Spectral clustering and the high-dimensional stochastic blockmodel. The Annals of Statistics, pages 1878-1915, 2011.

A. A. Amini, A. Chen, P. J. Bickel, and E. Levina. Pseudo-likelihood methods for community detection in large sparse networks. The Annals of Statistics, 41(4):2097-2122, 2013.

Qin, T. & Rohe, K. Regularized spectral clustering under the degree-corrected stochastic blockmodel Advances in Neural Information Processing Systems, 2013, 3120-3128

J. Lei and A. Rinaldo. Consistency of spectral clustering in stochastic block models. The Annals of Statistics, 43(1):215-237, 2014.

C. M. Le, E. Levina, and R. Vershynin. Concentration and regularization of random graphs. Random Structures & Algorithms, 2017.

S. J. Young and E. R. Scheinerman. Random dot product graph models for social networks. In International Workshop on Algorithms and Models for the Web-Graph, pages 138-149. Springer, 2007.

C. M. Le and E. Levina. Estimating the number of communities in networks by spectral methods. arXiv preprint arXiv:1507.00827, 2015.

Zhang, Y.; Levina, E. & Zhu, J. Estimating network edge probabilities by neighbourhood smoothing Biometrika, Oxford University Press, 2017, 104, 771-783

B. Karrer and M. E. Newman. Stochastic blockmodels and community structure in networks. Physical Review E, 83(1):016107, 2011.

Wang, Y. R. & Bickel, P. J. Likelihood-based model selection for stochastic block models The Annals of Statistics, Institute of Mathematical Statistics, 2017, 45, 500-528

Gao, C.; Ma, Z.; Zhang, A. Y. & Zhou, H. H. Achieving optimal misclassification proportion in stochastic block models The Journal of Machine Learning Research, JMLR. org, 2017, 18, 1980-2024

T. Li, E. Levina, and J. Zhu. Community models for networks observed through edge nominations. arXiv preprint arXiv:2008.03652 (2020).

T. Li and C. M. Le, Network Estimation by Mixing: Adaptivity and More. arXiv preprint arXiv:2106.02803, 2021.

R. Miao and T. Li. Informative core identification in complex networks. arXiv preprint arXiv:2101.06388, 2021

Sischka, B. and Kauermann, G., 2022. EM-based smooth graphon estimation using MCMC and spline-based approaches. Social Networks, 68, pp.279-295.

`BHMC.estimate` *Estimates the number of communities under block models by the spectral methods*

Description

Estimates the number of communities under block models by using the spectral properties of network Beth-Hessian matrix with moment correction.

Usage

```
BHMC.estimate(A, K.max = 15)
```

Arguments

`A`	adjacency matrix of the network
`K.max`	the maximum possible number of communities to check

Details

Note that the method cannot distinguish SBM and DCSBM. But it works under either model.

Value

A list of result

`K`	Estimated K
`values`	eigenvalues of the Beth-Hessian matrix

Author(s)

Tianxi Li, Elizaveta Levina, Ji Zhu
Maintainer: Tianxi Li <tianxili@virginia.edu>

References

C. M. Le and E. Levina. Estimating the number of communities in networks by spectral methods. arXiv preprint arXiv:1507.00827, 2015.

See Also

LRBIC,ECV.Block, NCV.select

Examples

```
dt <- BlockModel.Gen(30,300,K=3,beta=0.2,rho=0.9,simple=FALSE,power=TRUE)


A <- dt$A


bhmc <- BHMC.estimate(A,15)

bhmc
```

BlockModel.Gen *Generates networks from degree corrected stochastic block model*

Description

Generates networks from degree corrected stochastic block model, with various options for node degree distribution

Usage

```
BlockModel.Gen(lambda, n, beta = 0, K = 3, w = rep(1, K),
 Pi = rep(1, K)/K, rho = 0, simple = TRUE, power = TRUE,
alpha = 5, degree.seed = NULL)
```

Arguments

lambda	average node degree
n	size of network
beta	out-in ratio: the ratio of between-block edges over within-block edges
K	number of communities
w	not effective
Pi	a vector of community proportion

rho	proportion of small degrees within each community if the degrees are from two point mass disbribution. rho >0 gives degree corrected block model. If rho > 0 and simple=TRUE, then generate the degrees from two point mass distribution, with rho porition of 0.2 values and 1-rho proportion of 1 for degree parameters. If rho=0, generate from SBM.
simple	Indicator of wether two point mass degrees are used, if rho > 0. If rho=0, this is not effective
power	Whether or not use powerlaw distribution for degrees. If FALSE, generate from theta from U(0.2,1); if TRUE, generate theta from powerlaw. Only effective if rho >0, simple=FALSE.
alpha	Shape parameter for powerlaw distribution.
degree.seed	Can be a vector of a prespecified values for theta. Then the function will do sampling with replacement from the vector to generate theta. It can be used to control noise level between different configuration settings.

Value

A list of

A	the generated network adjacency matrix
g	community membership
P	probability matrix of the network
theta	node degree parameter

Author(s)

Tianxi Li, Elizaveta Levina, Ji Zhu
Maintainer: Tianxi Li <tianxili@virginia.edu>

References

B. Karrer and M. E. Newman. Stochastic blockmodels and community structure in networks. Physical Review E, 83(1):016107, 2011.

A. A. Amini, A. Chen, P. J. Bickel, and E. Levina. Pseudo-likelihood methods for community detection in large sparse networks. The Annals of Statistics, 41(4):2097-2122, 2013.

T. Li, E. Levina, and J. Zhu. Network cross-validation by edge sampling. Biometrika, 107(2), pp.257-276, 2020.

Examples

```
dt <- BlockModel.Gen(30,300,K=3,beta=0.2,rho=0.9,simple=FALSE,power=TRUE)
```

`ConsensusClust` *clusters nodes by concensus (majority voting) initialized by regularized spectral clustering*

Description

community detection by concensus (majority voting) initialized by regularized spectral clustering

Usage

```
ConsensusClust(A,K,tau=0.25,lap=TRUE)
```

Arguments

`A`	adjacency matrix
`K`	number of communities
`tau`	reguarlization parameter for regularized spectral clustering. Default value is 0.25. Typically set between 0 and 1. If tau=0, no regularization is applied.
`lap`	indicator. If TRUE, the Laplacian matrix for initializing clustering. If FALSE, the adjacency matrix will be used.

Details

Community detection algorithm by majority voting algorithm of Gao et. al. (2016). When initialized by regularized spectral clustering, it is shown that the clustering accuracy of this algorithm gives minimax rate under the SBM. However, it can slow compared with spectral clustering.

Value

cluster labels

Author(s)

Tianxi Li, Elizaveta Levina, Ji Zhu

Maintainer: Tianxi Li <tianxili@virginia.edu>

References

Gao, C.; Ma, Z.; Zhang, A. Y. & Zhou, H. H. Achieving optimal misclassification proportion in stochastic block models The Journal of Machine Learning Research, JMLR. org, 2017, 18, 1980-2024

See Also

`reg.SP`

Examples

```
dt <- BlockModel.Gen(15,50,K=2,beta=0.2,rho=0)

A <- dt$A

cc <- ConsensusClust(A,K=2,lap=TRUE)
```

DCSBM.estimate *Estimates DCSBM model*

Description

Estimates DCSBM model by given community labels

Usage

```
DCSBM.estimate(A, g)
```

Arguments

A	adjacency matrix
g	vector of community labels for the nodes

Details

Estimation is based on maximum likelihood.

Value

A list object of

Phat	estimated probability matrix
B	the B matrix with block connection probability, up to a scaling constant
Psi	vector of of degree parameter theta, up to a scaling constant

Author(s)

Tianxi Li, Elizaveta Levina, Ji Zhu
Maintainer: Tianxi Li <tianxili@virginia.edu>

References

B. Karrer and M. E. Newman. Stochastic blockmodels and community structure in networks. Physical Review E, 83(1):016107, 2011.

See Also

SBM.estimate

Examples

```
dt <- BlockModel.Gen(30,300,K=3,beta=0.2,rho=0.9,simple=FALSE,power=TRUE)


A <- dt$A


ssc <- reg.SSP(A,K=3,lap=TRUE)

est <- DCSBM.estimate(A,ssc$cluster)
```

ECV.block *selecting block models by ECV*

Description

Model selection by ECV for SBM and DCSBM. It can be used to select between the two models or given on model (either SBM or DCSBM) and select K.

Usage

```
ECV.block(A, max.K, cv = NULL, B = 3, holdout.p = 0.1, tau = 0, dc.est = 2, kappa = NULL)
```

Arguments

A	adjacency matrix
max.K	largest possible K for number of communities
cv	cross validation fold. The default value is NULL. We recommend to use the argument B instead, doing indpendent sampling.
B	number of replications
holdout.p	testing set proportion
tau	constant for numerical stability only. Not useful for current version.
dc.est	estimation method for DCSBM. By defaulty (dc.est=2), the maximum likelihood is used. If dc.est=1, the method used by Chen and Lei (2016) is used, which is less stable according to our observation.
kappa	constant for numerical stability only. Not useful for current version.

Details

The current version is based on a simple matrix completion procedure, as described in the paper. The performance can be improved by better matrix completion method that will be implemented in next version. Moreover, the current implementation is better in computational time but less efficient in memory. Another memory efficient implementation will be added in next version.

Value

`impute.err`	average validaiton imputation error
`l2`	average validation L_2 loss under SBM
`dev`	average validation binomial deviance loss under SBM
`auc`	average validation AUC
`dc.l2`	average validation L_2 loss under DCSBM
`dc.dev`	average validation binomial deviance loss under DCSBM
`sse`	average validation SSE
`l2.model`	selected model by L_2 loss
`dev.model`	selected model by binomial deviance loss
`l2.mat`, `dc.l2.mat`, `...`	cross-validation loss matrix for B replications

Author(s)

Tianxi Li, Elizaveta Levina, Ji Zhu
Maintainer: Tianxi Li <tianxili@virginia.edu>

References

T. Li, E. Levina, and J. Zhu. Network cross-validation by edge sampling. Biometrika, 107(2), pp.257-276, 2020.

See Also

`NCV.select`

Examples

```
dt <- BlockModel.Gen(30,300,K=3,beta=0.2,rho=0.9,simple=FALSE,power=TRUE)


A <- dt$A

ecv <- ECV.block(A,6,B=3)

ecv$l2.model
ecv$dev.model
```

```
which.min(ecv$l2)
which.min(ecv$dev)

which.min(ecv$dc.l2)
which.min(ecv$dc.dev)

which.max(ecv$auc)
which.min(ecv$sse)
```

`ECV.nSmooth.lowrank` *selecting tuning parameter for neighborhood smoothing estimation of graphon model*

Description

selecting tuning parameter for neighborhood smoothing estimation of graphon model where the tuning parameter is to control estimation smoothness.

Usage

```
ECV.nSmooth.lowrank(A, h.seq, K, cv = NULL, B = 3, holdout.p = 0.1)
```

Arguments

`A`	adjacency matrix
`h.seq`	a sequence of h values to tune. It is suggested h should be in the order of sqrt(log(n)/n).
`K`	the optimal rank for approximation. Can be obtained by rank selection of ECV.
`cv`	cross-validation fold. Recomend to use replication number B instead.
`B`	independent replication number of random splitting
`holdout.p`	proportion of test sample

Details

The neighborhood smoothing estimation can be slow, so the ECV may take long even for moderately large networks.

Value

a list object with

`err`	average validation error for h.seq
`min.index`	index of the minimum error

Author(s)

Tianxi Li, Elizaveta Levina, Ji Zhu
Maintainer: Tianxi Li <tianxili@virginia.edu>

References

T. Li, E. Levina, and J. Zhu. Network cross-validation by edge sampling. Biometrika, 107(2), pp.257-276, 2020.

Examples

```
set.seed(500)
N <- 300

U = matrix(1:N,nrow=1) / (N+1)
V = matrix(1:N,nrow=1) / (N+1)

W = (t(U))^2
W = W/3*cos(1/(W + 1e-7)) + 0.15



upper.index <- which(upper.tri(W))

A <- matrix(0,N,N)


rand.ind <- runif(length(upper.index))

edge.index <- upper.index[rand.ind < W[upper.index]]

A[edge.index] <- 1

A <- A + t(A)
diag(A) <- 0


h.seq <- sqrt(log(N)/N)*seq(0.5,5,by=0.5)


ecv.nsmooth <- ECV.nSmooth.lowrank(A,h.seq,K=2,B=3)

h <- h.seq[ecv.nsmooth$min.index]
```

ECV.Rank *estimates optimal low rank model for a network*

Description

estimates the optimal low rank model for a network

Usage

```
ECV.Rank(A, max.K, B = 3, holdout.p = 0.1, weighted = TRUE,mode="directed")
```

Arguments

A	adjacency matrix
max.K	maximum possible rank to check
B	number of replications in ECV
holdout.p	test set proportion
weighted	whether the network is weighted. If TRUE, only sum of squared errors are computed. If FALSE, then treat the network as binary and AUC will be computed along with SSE.
mode	Selectign the mode of "directed" or "undirected" for cross-validation.

Details

AUC is believed to be more accurate in many simulations for binary networks. But the computation of AUC is much slower than SSE, even slower than matrix completion steps.

Note that we do not have to assume the true model is low rank. This function simply finds a best low-rank approximation to the true model.

Value

A list of

sse.rank	rank selection by SSE loss
auc.rank	rank selection by AUC loss
auc	auc sequence for each rank candidate
sse	sse sequence for each rank candidate

Author(s)

Tianxi Li, Elizaveta Levina, Ji Zhu
Maintainer: Tianxi Li <tianxili@virginia.edu>

References

T. Li, E. Levina, and J. Zhu. Network cross-validation by edge sampling. Biometrika, 107(2), pp.257-276, 2020.

See Also

ECV.block

Examples

```
dt <- BlockModel.Gen(30,300,K=3,beta=0.2,rho=0.9,simple=FALSE,power=TRUE)


A <- dt$A


ecv.rank <- ECV.Rank(A,6,weighted=FALSE,mode="undirected")

ecv.rank
```

`InformativeCore` *identify the informative core component of a network*

Description

identify the informative core component of a network based on the spectral method of Miao and Li (2021). It can be used as a general data processing function for any network modeling purpose.

Usage

```
InformativeCore(A,r=3)
```

Arguments

`A`	adjacency matrix. It does not have to be unweighted.
`r`	the rank for low-rank denoising. The rank can be selected by ECV or any other methods availale in the package.

Details

The function can be used as a general data processing function for any network modeling purpose. It automatically identify an informative core component with interesting connection structure and a noninformative periphery component with uninterestings structures. Depending on the user's preference, the uninteresting structure can be either the Erdos-Renyi type connections or configuration type of connections, both of which are generally regarded as noninformative structures. Including these additional non-informative structures in network models can potentially lower the modeling efficiency. Therefore, it is preferable to remove them and only focus on the core structure. Details can be found in the reference.

Value

A list of

`er.score`	A n dimensional vector of informative scores for ER-type periphery. A larger score indicates the corresponding node is more likely to be in the core.

`config.score` A n dimensional vector of informative scores for configuration-type periphery. A larger score indicates the corresponding node is more likely to be in the core.

`er.theory.core` The indices of identified core structure in the ER-type model based on a theoretical threshold of the scores (for large sample size).

`config.theory.core` The indices of identified core structure in the configuration-type model based on a theoretical threshold of the scores (for large sample size).

`er.kmeans.core` The indices of identified core structure in the ER-type model based on kmeans clustering of the scores.

`config.kmeans.core` The indices of identified core structure in the configuration-type model based on kmeans clustering of the scores (for large sample size).

Author(s)

Tianxi Li, Elizaveta Levina, Ji Zhu, Can M. Le
Maintainer: Tianxi Li <tianxili@virginia.edu>

References

R. Miao and T. Li. Informative core identification in complex networks. arXiv preprint arXiv:2101.06388, 2021

See Also

ECV.Rank

Examples

```
set.seed(100)
dt <- BlockModel.Gen(60,1000,K=3,beta=0.2,rho=0.9,simple=FALSE,power=TRUE)
### this is not an interesting case -- only for demonstration of the usage.
### The network has no periphery nodes, all nodes are in the core.

A <- dt$A


core.fit <- InformativeCore(A,r=3)
length(core.fit$er.theory.core)
### essentially all nodes are selected as the core.
```

LRBIC *selecting number of communities by asymptotic likelihood ratio*

Description

selecting number of communities by asymptotic likelihood ratio based the methdo of Wang and Bickel 2015

Usage

```
LRBIC(A, Kmax, lambda = NULL, model = "both")
```

Arguments

A	adjacency matrix
Kmax	the largest possible number of communities to check
lambda	a tuning parameter. By default, will use the number recommended in the paper.
model	selecting K under which model. If set to be "SBM", the calculation will be done under SBM. If set to be "DCSBM", the calculation will be done under DCSBM. The default value is "both" so will give two selections under SBM and DCSBM respectively.

Details

Note that the method cannot distinguish SBM and DCSBM, though different calculation is done under the two models. So it is not valid to compare across models. The theoretical analysis of the method is done under maximum likelihood and variational EM. But as suggested in the paper, we use spectral clustering for community detection before fitting maximum likelihood.

Value

a list of

SBM.K	estimated number of communities under SBM
DCSBM.K	estimated number of communities under DCSBM
SBM.BIC	the BIC values for the K sequence under SBM
DCSBM.BIC	the BIC values for the K sequence under DCSBM

Author(s)

Tianxi Li, Elizaveta Levina, Ji Zhu
Maintainer: Tianxi Li <tianxili@virginia.edu>

References

Wang, Y. R. & Bickel, P. J. Likelihood-based model selection for stochastic block models The Annals of Statistics, Institute of Mathematical Statistics, 2017, 45, 500-528

See Also

BHMC.estimate, ECV.block, NCV.select

Examples

```
dt <- BlockModel.Gen(30,300,K=3,beta=0.2,rho=0.9,simple=FALSE,power=TRUE)

A <- dt$A

### test LRBIC

lrbic <- LRBIC(A,6,model="both")

lrbic$SBM.K

lrbic$DCSBM.K
```

LSM.PGD *estimates inner product latent space model by projected gradient descent*

Description

estimates inner product latent space model by projected gradient descent from the paper of Ma et al. (2020).

Usage

```
LSM.PGD(A, k,step.size=0.3,niter=500,trace=0)
```

Arguments

A	adjacency matrix
k	the dimension of the latent position
step.size	step size of gradient descent
niter	maximum number of iterations
trace	if trace > 0, the objective will be printed out after each iteration

Details

The method is based on the gradient descent of Ma et al (2020), with initialization of the universal singular value thresholding as discussed there. The parameter identifiability constraint is the same as in the paper.

Value

a list of

Z	latent positions
alpha	individual parameter alpha as in the paper
Phat	esitmated probability matrix
obj	the objective of the gradient method

Author(s)

Tianxi Li and Can M. Le
Maintainer: Tianxi Li <tianxili@virginia.edu>

References

Z. Ma, Z. Ma, and H. Yuan. Universal latent space model fitting for large networks with edge covariates. Journal of Machine Learning Research, 21(4):1-67, 2020.

See Also

DCSBM.estimate

Examples

```
dt <- RDPG.Gen(n=600,K=2,directed=TRUE)

A <- dt$A

fit <- LSM.PGD(A,2,niter=50)
```

NCV.select *selecting block models by NCV*

Description

selecting block models by NCV of Chen and Lei (2016)

Usage

```
NCV.select(A, max.K, cv = 3)
```

Arguments

A	adjacency matrix
max.K	largest number of communities to check
cv	fold of cross-validation

Details

Spectral clustering is used for fitting the block models

Value

a list of

dev	the binomial deviance loss under SBM for each K
l2	the L_2 loss under SBM for each K
dc.dev	the binomial deviance loss under DCSBM for each K
dc.l2	the L_2 loss under DCSBM for each K
dev.model	the selected model by deviance loss
l2.model	the selected model by L_2 loss
sbm.l2.mat, sbm.dev.mat, ...	the corresponding matrices of loss for each fold (row) and each K value (column)

Author(s)

Tianxi Li, Elizaveta Levina, Ji Zhu
Maintainer: Tianxi Li <tianxili@virginia.edu>

References

Chen, K. & Lei, J. Network cross-validation for determining the number of communities in network data Journal of the American Statistical Association, Taylor & Francis, 2018, 113, 241-251

See Also

ECV.block

Examples

```
dt <- BlockModel.Gen(30,300,K=3,beta=0.2,rho=0.9,simple=FALSE,power=TRUE)


A <- dt$A


ncv <- NCV.select(A,6,3)

ncv$l2.model
```

```
ncv$dev.model

which.min(ncv$dev)
which.min(ncv$l2)

which.min(ncv$dc.dev)
which.min(ncv$dc.l2)
```

`network.mixing` *estimates network connection probability by network mixing*

Description

estimates network connection probability by network mixing of Li and Le (2021).

Usage

```
network.mixing(A,index=NULL,rho = 0.1,max.K=15,dcsbm=TRUE, usvt=TRUE,ns=FALSE,
                            lsm=FALSE,lsm.k=4,trace=FALSE)
```

Arguments

`A`	adjacency matrix
`index`	a pre-specified hold-out set. If NULL, the set will be randomly generated according to rho.
`rho`	hold-out proportion as validation entries. Only effective when index is NULL.
`max.K`	the maximum number of blocks used for the block model approximation (see details).
`dcsbm`	whether to include the DCSBM as components, up to max.K. By default, the method will include it.
`usvt`	whether to include the USVT as a component. By default, the method will include it.
`ns`	whether to include the neighborhood smoothing as a component.
`lsm`	whether to include the gradient estimator of the latent space model as a component.
`lsm.k`	the dimension of the latent space. Only effective if lsm is TRUE.
`trace`	whether to print the model fitting progress.

Details

The basic version of the mixing estimator will include SBM and DCSBM estimates with the number of blocks from 1 to max.K. Users could also specify whether to include additional USVT, neighborhood smoothing and latent space model estimators. If NNL (non-negative linear), exponential, or ECV is used, the mixing is usually robust for a reasonable range of max.K and whether to include the other models. The linear mixing, however, is vulnerable for a large number of base estimates. The NNL is our recommended method. USVT is also recommended. the neighborhood smoothing and latent space model are slower, so are not suitable for large networks. Details can be found in Li and Le (2021).

Value

a list of

`linear.Phat`	estimated probability matrix by linear mixing
`linear.weight`	the weights of the indivdiual models in linear mixing
`nnl.Phat`	estimated probability matrix by NNL mixing
`nnl.weight`	the weights of the indivdiual models in NNL mixing
`exp.Phat`	estimated probability matrix by exponential mixing
`exp.weight`	the weights of the indivdiual models in exponential mixing
`ecv.Phat`	estimated probability matrix by ECV mixing (only one nonzero)
`ecv.weight`	the weights of the indivdiual models in ECV mixing (only one nonzero)
`model.names`	the names of all individual models, in the same order as the weights

Author(s)

Tianxi Li and Can M. Le

Maintainer: Tianxi Li <tianxili@virginia.edu>

References

T. Li and C. M. Le, Network Estimation by Mixing: Adaptivity and More. arXiv preprint arXiv:2106.02803, 2021.

Examples

```
dt <- RDPG.Gen(n=500,K=5,directed=TRUE)

A <- dt$A

fit <- network.mixing(A)
fit$model.names

fit$nnl.weight
```

`network.mixing.Bfold` *estimates network connection probability by network mixing with B-fold averaging*

Description

estimates network connection probability by network mixing of Li and Le (2021) with B-fold averaging.

Usage

```
network.mixing.Bfold(A,B=10,rho = 0.1,max.K=15,dcsbm=TRUE,usvt=TRUE,ns=FALSE,
                     lsm=FALSE,lsm.k=4)
```

Arguments

`A`	adjacency matrix
`B`	number of random replications to average over
`rho`	hold-out proportion as validation entries. Only effective when index is NULL.
`max.K`	the maximum number of blocks used for the block model approximation (see details).
`dcsbm`	whether to include the DCSBM as components, up to max.K. By default, the method will include it.
`usvt`	whether to include the USVT as a component. By default, the method will include it.
`ns`	whether to include the neighborhood smoothing as a component.
`lsm`	whether to include the gradient estimator of the latent space model as a component.
`lsm.k`	the dimension of the latent space. Only effective if lsm is TRUE.

Details

This is essentially the same procedure as the network.mixing, but repeat it B times and return the average. Use with cautious. Though it can make the estimate more stable, the procedure would increase the computational cost by a factor of B. Based on our limited experience, this is usually not a great trade-off as the improvement might be marginal.

Value

a list of

`linear.Phat`	estimated probability matrix by linear mixing
`nnl.Phat`	estimated probability matrix by NNL mixing
`exp.Phat`	estimated probability matrix by exponential mixing
`ecv.Phat`	estimated probability matrix by ECV mixing (only one nonzero)
`model.names`	the names of all individual models, in the same order as the weights

Author(s)

Tianxi Li and Can M. Le

Maintainer: Tianxi Li <tianxili@virginia.edu>

References

T. Li and C. M. Le, Network Estimation by Mixing: Adaptivity and More. arXiv preprint arXiv:2106.02803, 2021.

See Also

network.mixing

Examples

```
dt <- RDPG.Gen(n=200,K=3,directed=TRUE)
A <- dt$A

fit <- network.mixing.Bfold(A,B=2)
```

NMI *calculates normalized mutual information*

Description

calculates normalized mutual information, a metric that is commonly used to compare clustering results

Usage

```
NMI(g1, g2)
```

Arguments

g1	a vector of cluster labels
g2	a vector of cluster labels (same length as g1)

Value

NMI value

Author(s)

Tianxi Li, Elizaveta Levina, Ji Zhu
Maintainer: Tianxi Li <tianxili@virginia.edu>

Examples

```
dt <- BlockModel.Gen(30,300,K=3,beta=0.2,rho=0.9,simple=FALSE,power=TRUE)


A <- dt$A


ssc <- reg.SSP(A,K=3,lap=TRUE)

NMI(ssc$cluster,dt$g)
```

`NSBM.estimate` *estimates nomination SBM parameters given community labels by the method of moments*

Description

estimates NSBM parameters given community labels

Usage

```
NSBM.estimate(A,K,g,reg.bound=-Inf)
```

Arguments

`A`	adjacency matrix of a directed where Aij = 1 iff i -> j
`K`	number of communities
`g`	a vector of community labels
`reg.bound`	the regularity lower bound of lambda value. By default, -Inf. That means, no constraints. When the network is sparse, using certain constaints may improve stability.

Details

The method of moments is used for estimating the edge nomination SBM, so the strategy can be used for both unweighted and weighted networks. The details can be found in Li et. al. (2020).

Value

a list of

`B`	estimated block connection probability matrix
`lambda`	estimated lambda values for nomination intensity
`theta`	estimated theta values for nomination preference
`P.tilde`	estimated composiste probability matrix after nomination
`g`	community labels

Author(s)

Tianxi Li, Elizaveta Levina, Ji Zhu
Maintainer: Tianxi Li <tianxili@virginia.edu>

References

T. Li, E. Levina, and J. Zhu. Community models for networks observed through edge nominations. arXiv preprint arXiv:2008.03652 (2020).

See Also

SBM.estimate

Examples

```
dt <- NSBM.Gen(n=200,K=2,beta=0.2,avg.d=10)


A <- dt$A


sc <- RightSC(A,K=3)
est <- NSBM.estimate(A,K=3,g=sc$cluster)
```

NSBM.Gen *Generates networks from nomination stochastic block model*

Description

Generates networks from nomination stochastic block model for community structure in edge nomination procedures, proposed in Li et. al. (2020)

Usage

```
NSBM.Gen( n, K, avg.d,beta,theta.low=0.1,
    theta.p=0.2,lambda.scale=0.2,lambda.exp=FALSE)
```

Arguments

n	size of network
K	number of communities
avg.d	expected average degree of the resuling network (after edge nomination)
beta	the out-in ratio of the original SBM
theta.low	the lower value of theta's. The theta's are generated as two-point mass at theta.low and 1.

theta.p	proportion of lower value of theta's
lambda.scale	standard deviation of the lambda (before the exponential, see lambda.exp)
lambda.exp	If TRUE, lambda is generated as exponential of uniformation random randomes. Otherwise, they are normally distributed.

Value

A list of

A	the generated network adjacency matrix
g	community membership
P	probability matrix of the orignal SBM network
P.tilde	probability matrix of the observed network after nomination
B	B parameter
lambda	lambda parameter
theta	theta parameter

Author(s)

Tianxi Li, Elizaveta Levina, Ji Zhu
Maintainer: Tianxi Li <tianxili@virginia.edu>

References

T. Li, E. Levina, and J. Zhu. Community models for networks observed through edge nominations. arXiv preprint arXiv:2008.03652 (2020).

Examples

```
dt <- NSBM.Gen(n=200,K=2,beta=0.2,avg.d=10)
```

---

nSmooth *estimates probabilty matrix by neighborhood smoothing*

Description

estimates probabilty matrix by neighborhood smoothing of Zhang et. al. (2017)

Usage

```
nSmooth(A, h = NULL)
```

Arguments

A	adjacency matrix
h	quantile value used for smoothing. Recommended to be in the scale of sqrt(log(n)/n) where n is the size of the network. The default value is sqrt(log(n)/n) from the paper.

Details

The method assumes a graphon model where the underlying graphon function is piecewise Lipchitz. However, it may be slow for moderately large networks, though it is one of the fastest methods for graphon models.

Value

the probability matrix

Author(s)

Tianxi Li, Elizaveta Levina, Ji Zhu

Maintainer: Tianxi Li <tianxili@virginia.edu>

References

Zhang, Y.; Levina, E. & Zhu, J. Estimating network edge probabilities by neighbourhood smoothing Biometrika, Oxford University Press, 2017, 104, 771-783

Examples

```
N <- 100

U = matrix(1:N,nrow=1) / (N+1)
V = matrix(1:N,nrow=1) / (N+1)

W = (t(U))^2
W = W/3*cos(1/(W + 1e-7)) + 0.15

upper.index <- which(upper.tri(W))

A <- matrix(0,N,N)


rand.ind <- runif(length(upper.index))

edge.index <- upper.index[rand.ind < W[upper.index]]

A[edge.index] <- 1

A <- A + t(A)
```

```
diag(A) <- 0

What <- nSmooth(A)
```

RDPG.Gen *generates random networks from random dot product graph model*

Description

generates random networks from random dot product graph model

Usage

```
RDPG.Gen(n, K, directed = TRUE, avg.d = NULL)
```

Arguments

n	size of the network
K	dimension of latent space
directed	whether the network is directed or not
avg.d	average node degree of the network (in expectation)

Details

The network is generated according to special formulation mentioned in ECV paper.

Value

a list of

A	the adjacency matrix
P	the probability matrix

Author(s)

Tianxi Li, Elizaveta Levina, Ji Zhu
Maintainer: Tianxi Li <tianxili@virginia.edu>

References

S. J. Young and E. R. Scheinerman. Random dot product graph models for social networks. In International Workshop on Algorithms and Models for the Web-Graph, pages 138-149. Springer, 2007. T. Li, E. Levina, and J. Zhu. Network cross-validation by edge sampling. Biometrika, 107(2), pp.257-276, 2020.

Examples

```
dt <- RDPG.Gen(n=600,K=2,directed=TRUE)

A <- dt$A
```

`reg.SP` *clusters nodes by regularized spectral clustering*

Description

community detection by regularized spectral clustering

Usage

```
reg.SP(A, K, tau = 1, lap = FALSE,nstart=30,iter.max=100)
```

Arguments

`A`	adjacency matrix
`K`	number of communities
`tau`	reguarlization parameter. Default value is one. Typically set between 0 and 1. If tau=0, no regularization is applied.
`lap`	indicator. If TRUE, the Laplacian matrix for clustering. If FALSE, the adjacency matrix will be used.
`nstart`	number of random initializations for K-means
`iter.max`	maximum number of iterations for K-means

Details

The regularlization is done by adding a small constant to each element of the adjacency matrix. It is shown by such perturbation helps concentration in sparse networks. It is shown to give consistent clustering under SBM.

Value

a list of

`cluster`	cluster labels
`loss`	the loss of Kmeans algorithm

Author(s)

Tianxi Li, Elizaveta Levina, Ji Zhu

Maintainer: Tianxi Li <tianxili@virginia.edu>

References

K. Rohe, S. Chatterjee, and B. Yu. Spectral clustering and the high-dimensional stochastic blockmodel. The Annals of Statistics, pages 1878-1915, 2011.

A. A. Amini, A. Chen, P. J. Bickel, and E. Levina. Pseudo-likelihood methods for community detection in large sparse networks. The Annals of Statistics, 41(4):2097-2122, 2013.

J. Lei and A. Rinaldo. Consistency of spectral clustering in stochastic block models. The Annals of Statistics, 43(1):215-237, 2014.

C. M. Le, E. Levina, and R. Vershynin. Concentration and regularization of random graphs. Random Structures & Algorithms, 2017.

See Also

reg.SP

Examples

```
dt <- BlockModel.Gen(30,300,K=3,beta=0.2,rho=0)

A <- dt$A

sc <- reg.SP(A,K=3,lap=TRUE)

NMI(sc$cluster,dt$g)
```

---

reg.SSP *detects communities by regularized spherical spectral clustering*

Description

community detection by regularized spherical spectral clustering

Usage

```
reg.SSP(A, K, tau = 1, lap = FALSE,nstart=30,iter.max=100)
```

Arguments

A	adjacency matrix
K	number of communities
tau	reguarlization parameter. Default value is one. Typically set between 0 and 1. If tau=0, no regularization is applied.

lap	indicator. If TRUE, the Laplacian matrix for clustering. If FALSE, the adjacency matrix will be used.
nstart	number of random initializations for K-means
iter.max	maximum number of iterations for K-means

Details

The regularlization is done by adding a small constant to each element of the adjacency matrix. It is shown by such perturbation helps concentration in sparse networks. The difference from spectral clustering (reg.SP) comes from its extra step to normalize the rows of spectral vectors. It is proved that it gives consistent clustering under DCSBM.

Value

a list of

cluster	cluster labels
loss	the loss of Kmeans algorithm

Author(s)

Tianxi Li, Elizaveta Levina, Ji Zhu

Maintainer: Tianxi Li <tianxili@virginia.edu>

References

T. Qin and K. Rohe. Regularized spectral clustering under the degree-corrected stochastic block-model. In Advances in Neural Information Processing Systems, pages 3120-3128, 2013.

J. Lei and A. Rinaldo. Consistency of spectral clustering in stochastic block models. The Annals of Statistics, 43(1):215-237, 2014.

See Also

reg.SP

Examples

```
dt <- BlockModel.Gen(30,300,K=3,beta=0.2,rho=0.9,simple=FALSE,power=TRUE)


A <- dt$A

ssc <- reg.SSP(A,K=3,lap=TRUE)

NMI(ssc$cluster,dt$g)
```

RightSC *clusters nodes in a directed network by regularized spectral clustering on right singular vectors*

Description

community detection by regularized spectral clustering on right singular vectors

Usage

```
RightSC(A, K, normal = FALSE)
```

Arguments

A	adjacency matrix of a directed adjacecy matrix
K	number of communities
normal	indicator. If TRUE, normalization of singular vector rows will be applied, similar to the spectral spherical clustering.

Details

This is essentially the spectral clustering applied on right singular vectors. It can be used to handle directed networks where Aij = 1 if and only if i -> j, and the edges tend to have a missing issue specifically depending on the sender i. More details can be found in Li et. al. (2020).

Value

a list of

cluster	cluster labels
loss	the loss of Kmeans algorithm

Author(s)

Tianxi Li, Elizaveta Levina, Ji Zhu

Maintainer: Tianxi Li <tianxili@virginia.edu>

References

T. Li, E. Levina, and J. Zhu. Community models for networks observed through edge nominations. arXiv preprint arXiv:2008.03652 (2020).

See Also

reg.SP

Examples

```
dt <- BlockModel.Gen(30,300,K=3,beta=0.2,rho=0)

A <- dt$A

sc <- RightSC(A,K=2)
```

SBM.estimate *estimates SBM parameters given community labels*

Description

estimates SBM parameters given community labels

Usage

```
SBM.estimate(A, g)
```

Arguments

A	adjacency matrix
g	a vector of community labels

Details

maximum likelihood is used

Value

a list of

B	estimated block connection probability matrix
Phat	estimated probability matrix
g	community labels

Author(s)

Tianxi Li, Elizaveta Levina, Ji Zhu
Maintainer: Tianxi Li <tianxili@virginia.edu>

References

B. Karrer and M. E. Newman. Stochastic blockmodels and community structure in networks. Physical Review E, 83(1):016107, 2011.

See Also

DCSBM.estimate

Examples

```
dt <- BlockModel.Gen(30,300,K=3,beta=0.2,rho=0)


A <- dt$A


sc <- reg.SP(A,K=3,lap=TRUE)
sbm <- SBM.estimate(A,sc$cluster)
sbm$B
```

---

smooth.oracle *oracle smooth graphon estimation*

Description

oracle smooth graphon estimation according to given latent positions, based on smooth estimation.

Usage

```
smooth.oracle(Us,A)
```

Arguments

Us	a vector whose elements are the latent positions of the network nodes under the graphon model. The dimension of the vector equals the number of nodes in the network.
A	adjacency matrix. It does not have to be unweighted.

Details

Note that the latenet positions are unknown in practice, so this estimation is an oracle estimation mainly for evaluation purpose. However, if the latenet positions can be approximated estimated, this function can also be used for estimating the edge probability matrix. This procedure is the M-step of the algorithm used in Sischka & Kauermann (2022). Our implementation is modified from the contribution of an anonymous reviewer, leveraging the tools of the sparseFLMM package.

Value

The estimated probability matrix.

Author(s)

Tianxi Li, Elizaveta Levina, Ji Zhu, Can M. Le
Maintainer: Tianxi Li <tianxili@virginia.edu>

References

Sischka, B. and Kauermann, G., 2022. EM-based smooth graphon estimation using MCMC and spline-based approaches. Social Networks, 68, pp.279-295.

Examples

```
set.seed(100)
dt <- BlockModel.Gen(10,50,K=2,beta=0.2)

## oracle order
oracle.index <- sort(dt$g,index.return=TRUE)$ix
A <- dt$A[oracle.index,oracle.index]

oracle.est <- smooth.oracle(seq(0,1,length.out=50),A)
```

USVT *estimates the network probability matrix by the improved universal singular value thresholding*

Description

estimates the network probability matrix by the universal singular value thresholding of Chatterjee (2015), with the improvement mentioned in Zhang et. al. (2017).

Usage

```
USVT(A)
```

Arguments

A	adjacency matrix

Details

Instead of using the original threshold in Chatterjee (2015), the estimate is generated by taking the n^(1/3) leading spectral components. The method was mentioned in Zhang et. al. (2017) and suggested by an anonymous reviewer.

Value

The estimated probability matrix.

Author(s)

Tianxi Li and Can M. Le
Maintainer: Tianxi Li <tianxili@virginia.edu>

References

S. Chatterjee. Matrix estimation by universal singular value thresholding. The Annals of Statistics, 43(1):177-214, 2015. Y. Zhang, E. Levina, and J. Zhu. Estimating network edge probabilities by neighbourhood smoothing. Biometrika, 104(4):771-783, 2017.

See Also

LSM.PGD

Examples

```
dt <- RDPG.Gen(n=600,K=2,directed=TRUE)


A <- dt$A


fit <- USVT(A)
```

Index