Package ‘rMultiNet’

June 27, 2023

Title Multi-Layer Networks Analysis

Version 0.1

Description Provides two general frameworks to generate a multi-layer network. This also provides several methods to reveal the embedding of both nodes and layers. The reference paper can be found from the URL mentioned below. Ting Li, Zhongyuan Lyu, Chenyu Ren, Dong Xia (2023) <arXiv:2302.04437>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.2.3

Imports dbscan, geigen, glmnet, graphics, Matrix, plotly, rTensor, stats

NeedsCompilation no

Author Chenyu Ren [aut, cre] (<https://orcid.org/0000-0001-5422-7903>)

Maintainer Chenyu Ren <chenyu.ren@connect.polyu.hk>

Repository CRAN

Date/Publication 2023-06-27 16:30:02 UTC

R topics documented:

Community_cluster_dbscan 2
Community_cluster_km 2
Embedding_network 3
GenerateMMLSM 4
GenerateMMSBM 5
InitializationLSM 5
InitializationMMSBM 6
PowerIteration 7
ProjectedGD 8
SpecClustering 9

Index **10**

`Community_cluster_dbscan` *Title*

Description

Title

Usage

```
Community_cluster_dbscan(embedding, type, eps_value = 0.05, pts_value = 5)
```

Arguments

`embedding`	the embedding results from different methods
`type`	node embedding ‘n’ or network embedding ‘N’
`eps_value`	parameters for DBSCAN
`pts_value`	parameters for DBSCAN

Value

the embedding results

Examples

```
tnsr = GenerateMMSBM(200, 3, 10, 2, d = NULL, r = NULL)
U_list = InitializationMMSBM(tnsr, 3, 2, rank = NULL)
embed_list = PowerIteration(tnsr,3,2,rank=NULL,type="TUCKER",U_0_list=U_list)
em = embed_list[[2]]
Community_cluster_dbscan(em,"N")
```

`Community_cluster_km` *Title*

Description

Title

Usage

```
Community_cluster_km(embedding, type, cluster_number)
```

Arguments

`embedding`	the embedding results from different methods
`type`	node embedding ‘n’ or network embedding ‘N’
`cluster_number`	the number of clusters for Kmeans

Value

the embedding results

Examples

```
tnsr = GenerateMMSBM(200, 3, 10, 2, d = NULL, r = NULL)
U_list = InitializationMMSBM(tnsr, 3, 2, rank = NULL)
embed_list = PowerIteration(tnsr,3,2,rank=NULL,type="TUCKER",U_0_list=U_list)
em = embed_list[[2]]
Community_cluster_km(em,"N",5)
```

`Embedding_network` *Title*

Description

Title

Usage

```
Embedding_network(network_membership, L, paxis = 2)
```

Arguments

`network_membership`	the number of types of the network or the number of groups of vertices
`L`	the number of layers
`paxis`	the number of eigenvectors to use in the plot

Value

a plot table If the number of eigenvectors is more than two or plot the image

Examples

```
tnsr = GenerateMMSBM(200, 3, 10, 2, d = NULL, r = NULL)
U_list = InitializationMMSBM(tnsr, 3, 2, rank = NULL)
embed_list = PowerIteration(tnsr,3,2,rank=NULL,type="TUCKER",U_0_list=U_list)
Embedding_network(embed_list[[2]],10,2)
```

GenerateMMLSM *Title*

Description

Title

Usage

```
GenerateMMLSM(
  n,
  m,
  L,
  rank,
  U_mean = 0.5,
  cmax = 1,
  d,
  int_type = "Uniform",
  kernel_fun = "logit",
  scale_par = 1
)
```

Arguments

`n`	the number of vertices
`m`	the number of types of the network
`L`	the number of layers
`rank`	the rank of latent position matrix U
`U_mean`	the mean of the normal distribution of each entry of U
`cmax`	the entrywise upper bound of core tensor C
`d`	the average degree of the network
`int_type`	represents the ways of generating tensor C (‘Uniform’ or ‘Norm’)
`kernel_fun`	the link function of generating the adjacency tensor (‘logit’ or ‘probit’)
`scale_par`	the scaling factor of the parameter tensor

Value

a list including an adjacency tensor and the generating parameters

Examples

```
GenerateMMLSM(200,3,10,2,d=NULL)
```

GenerateMMSBM *Title*

Description

Title

Usage

```
GenerateMMSBM(n, m, L, K, d = NULL, r = NULL)
```

Arguments

n	the number of vertices
m	the number of types of the network
L	the number of layers
K	the number of groups of vertices
d	the average degree of the network
r	the out-in ratio in each layer

Value

a list including an adjacency tensor and the generating parameters

Examples

```
GenerateMMSBM(200, 3, 10, 2, d = NULL, r = NULL)
```

InitializationLSM *Title*

Description

Title

Usage

```
InitializationLSM(
  gen_list,
  n,
  m,
  k,
  rank = NULL,
  perturb = 0.1,
  int_type = "warm"
)
```

Arguments

`gen_list`	a list including the adjacency tensor and the parameter of the mixture multilayer network
`n`	the number of nodes
`m`	the number of network types
`k`	the number of groups of vertices
`rank`	rank of U
`perturb`	the upper bound of Uniform distribution
`int_type`	the method to initialize U and W (‘spec’, ‘rand’ or ‘warm’)

Value

a list including the adjacency tensor, U0, W0 and tuning parameters

Examples

```
gen_list = GenerateMMLSM(200,3,10,2,d=NULL)
InitializationLSM(gen_list,200,3,2)
```

`InitializationMMSBM` *Title A function for initialization*

Description

Title A function for initialization

Usage

```
InitializationMMSBM(tnsr, m, k, rank = NULL)
```

Arguments

`tnsr`	the tensor of network
`m`	the number of types of the network
`k`	the number of groups of vertices
`rank`	the rank of the core tensor calculated by the equation

Value

U_list a list including the core tensor Z, network embedding and node embedding

Examples

```
tnsr = GenerateMMSBM(200, 3, 10, 2, d = NULL, r = NULL)
U_list = InitializationMMSBM(tnsr, 3, 2, rank = NULL)
```

PowerIteration *Title*

Description

Title

Usage

```
PowerIteration(
  tnsr,
  m,
  k,
  rank = NULL,
  type = "TWIST",
  U_0_list,
  delta1 = 1000,
  delta2 = 1000,
  max_iter = 5,
  tol = 1e-05
)
```

Arguments

`tnsr`	the adjacency tensor of the network
`m`	the number of types of the network
`k`	the number of groups of vertices
`rank`	the rank of the core tensor calculated by the equation
`type`	specifies the iterative algorithm to run ‘TWIST’ or ‘Tucker’
`U_0_list`	InitializationMMSBM outputs
`delta1`	tuning parameters for regularization in mode1
`delta2`	tuning parameters for regularization in mode2
`max_iter`	the max times of iteration
`tol`	the convergence tolerance

Value

a list including the core tensor Z, network embedding and node embedding

Examples

```
tnsr = GenerateMMSBM(200, 3, 10, 2, d = NULL, r = NULL)
U_list = InitializationMMSBM(tnsr, 3, 2, rank = NULL)
embed_list = PowerIteration(tnsr,3,2,rank=NULL,type="TUCKER",U_0_list=U_list)
```

ProjectedGD *Title*

Description

Title

Usage

```
ProjectedGD(
  Ini_list,
  cmax = 1,
  eta_outer = 0.001,
  tmax_outer = 10,
  p_type = "logit",
  rd = "Non",
  show = TRUE,
  sgma = 1,
  sample_size = 500
)
```

Arguments

Ini_list	the output of function InitializationLSM
cmax	the upper limits for adding the coefficient constraint
eta_outer	the learning rate in gradient descent
tmax_outer	the number of iterations in gradient descent
p_type	the type of link function (‘logit’, ‘probit’ or ‘poisson’)
rd	whether to use stochastic sampling (‘rand’ or ‘Non’)
show	if print the ietation process
sgma	the link function parameter
sample_size	the size of sampling

Value

the embedding results of nodes and layers

Examples

```
gen_list = GenerateMMLSM(200,3,5,2,d=NULL)
Ini_list = InitializationLSM(gen_list,200,3,2)
```

`SpecClustering` *Title*

Description

Title

Usage

```
SpecClustering(tnsr, rank, embedding_type = "Layer")
```

Arguments

`tnsr`	the adjacency tensor
`rank`	the number of columns of the output matrix U
`embedding_type`	SumAdj for 'Node' and M3SC for 'Layer'

Value

The embeddding result can be applied in cluster methods like kmeans.

Examples

```
tnsr = GenerateMMSBM(200, 3, 10, 2, d = NULL, r = NULL)
emb_result = SpecClustering(tnsr,3)
```

Index

Community_cluster_dbscan, 2
Community_cluster_km, 2

Embedding_network, 3

GenerateMMLSM, 4
GenerateMMSBM, 5

InitializationLSM, 5
InitializationMMSBM, 6

PowerIteration, 7
ProjectedGD, 8

SpecClustering, 9