Package ‘NetworkDistance’

December 8, 2017

Type Package

Title Distance Measures for Networks

Version 0.1.0

Description Network is a prevalent form of data structure in many fields. As an object of analysis, many distance or metric measures have been proposed to define the concept of similarity between two networks. We provide a number of distance measures for networks. See Jurman et al (2011) <doi:10.3233/978-1-60750-692-8-227> for an overview on spectral class of inter-graph distance measures.

License GPL (>= 3)

Encoding UTF-8

LazyData true

Imports Matrix, Rcpp, Rdpack, RSpectra, doParallel, foreach, parallel, stats, igraph, network

LinkingTo Rcpp, RcppArmadillo

RdMacros Rdpack

RoxygenNote 6.0.1

NeedsCompilation yes

Author Kisung You [aut, cre] (0000-0002-8584-459X)

Maintainer Kisung You <kyou@nd.edu>

Repository CRAN

Date/Publication 2017-12-08 09:54:21 UTC

R topics documented:

`nd.centrality` *Centrality Distance*

Description

Centrality is a core concept in studying the topological structure of complex networks, which can be either defined for each node or edge. `nd.centrality` offers 3 distance measures on node-defined centralities. See this Wikipedia page for more on network/graph centrality.

Usage

```
nd.centrality(A, out.dist = TRUE, mode = c("Degree", "Close", "Between"),
  directed = FALSE)
```

Arguments

`A`	a list of length N containing $(M \times M)$ adjacency matrices.
`out.dist`	a logical; `TRUE` for computed distance matrix as a `dist` object.
`mode`	type of node centrality definitions to be used.
`directed`	a logical; `FALSE` as symmetric, undirected graph.

Value

a named list containing

D an $(N \times N)$ matrix or `dist` object containing pairwise distance measures.

features an $(N \times M)$ matrix where rows are node centralities for each graph.

References

Roy M, Schmid S and Trédan G (2014). "Modeling and Measuring Graph Similarity: The Case for Centrality Distance." In *FOMC 2014, 10th ACM International Workshop on Foundations of Mobile Computing*, pp. 53. https://hal.archives-ouvertes.fr/hal-01010901.

Examples

```
## generate two types of adjacency matrices of size (3-by-3)
rbin1 = rbinom(9,1,0.8); mat1 = matrix(rbin1,nrow=3)
rbin2 = rbinom(9,1,0.2); mat2 = matrix(rbin2,nrow=3)

mattype1 = ceiling((mat1+t(mat1))/2); diag(mattype1)=0;
mattype2 = ceiling((mat2+t(mat2))/2); diag(mattype2)=0;
```

```
A = list()
for (i in 1:3){A[[i]]=mattype1} # first 3 are type-1
for (i in 4:6){A[[i]]=mattype2} # next  3 are type-2

## use 3 types of centrality measures
out1 <- nd.centrality(A,out.dist=FALSE,mode="Degree")
out2 <- nd.centrality(A,out.dist=FALSE,mode="Close")
out3 <- nd.centrality(A,out.dist=FALSE,mode="Between")

## visualize
par(mfrow=c(1,3))
image(out1$D, main="Degree")
image(out2$D, main="Closeness")
image(out3$D, main="Betweenness")
```

`nd.csd` *L_2 Distance of Continuous Spectral Densities*

Description

The method employs spectral density of eigenvalues from Laplacian in that for each, we have corresponding spectral density $\rho(w)$ as a sum of narrow Lorentz distributions with `bandwidth` parameter.

Usage

```
nd.csd(A, out.dist = TRUE, bandwidth = 1)
```

Arguments

`A`	a list of length N containing $(M \times M)$ adjacency matrices.
`out.dist`	a logical; `TRUE` for computed distance matrix as a `dist` object.
`bandwidth`	common bandwidth of positive real number.

Value

a named list containing

D an $(N \times N)$ matrix or `dist` object containing pairwise distance measures.

spectra an $(N \times M - 1)$ matrix where each row is top-$M - 1$ vibrational spectra.

References

Ipsen M and Mikhailov AS (2002). "Evolutionary reconstruction of networks." *Physical Review E*, **66**(4). ISSN 1063-651X, 1095-3787, doi: 10.1103/PhysRevE.66.046109, `https://link.aps.org/doi/10.1103/PhysRevE.66.046109`.

Examples

```
## generate two types of adjacency matrices of size (3-by-3)
rbin1 = rbinom(9,1,0.8); mat1 = matrix(rbin1,nrow=3)
rbin2 = rbinom(9,1,0.2); mat2 = matrix(rbin2,nrow=3)

mattype1 = ceiling((mat1+t(mat1))/2); diag(mattype1)=0;
mattype2 = ceiling((mat2+t(mat2))/2); diag(mattype2)=0;

A = list()
for (i in 1:3){A[[i]]=mattype1} # first 3 are type-1
for (i in 4:6){A[[i]]=mattype2} # next  3 are type-2

## Compute Distance Matrix and Visualize
output = nd.csd(A, out.dist=FALSE, bandwidth=1.0)
image(output$D, main="two group case")
```

`nd.dsd` *Discrete Spectral Distance*

Description

Discrete Spectral Distance (DSD) is defined as the Euclidean distance between the spectra of various matrices, such as adjacency matrix A("Adj"), (unnormalized) Laplacian matrix $L = D - A$("Lap"), signless Laplacian matrix $|L| = D + A$("SLap"), or normalized Laplacian matrix $\tilde{L} = D^{-1/2}LD^{-1/2}$.

Usage

```
nd.dsd(A, out.dist = TRUE, type = c("Adj", "Lap", "SLap", "NLap"))
```

Arguments

`A`	a list of length N containing $(M \times M)$ adjacency matrices.
`out.dist`	a logical; `TRUE` for computed distance matrix as a `dist` object.
`type`	type of target structure. One of `"Adj"`,`"Lap"`,`"SLap"`,`"NLap"` as defined above.

Value

a named list containing

D an $(N \times N)$ matrix or `dist` object containing pairwise distance measures.

spectra an $(N \times M - 1)$ matrix where each row is top-$M - 1$ vibrational spectra.

References

Wilson RC and Zhu P (2008). "A study of graph spectra for comparing graphs and trees." *Pattern Recognition*, **41**(9), pp. 2833–2841. ISSN 00313203, doi: 10.1016/j.patcog.2008.03.011, `http://linkinghub.elsevier.com/retrieve/pii/S0031320308000927`.

Examples

```
## generate two types of adjacency matrices of size (3-by-3)
rbin1 = rbinom(9,1,0.8); mat1 = matrix(rbin1,nrow=3)
rbin2 = rbinom(9,1,0.2); mat2 = matrix(rbin2,nrow=3)

mattype1 = ceiling((mat1+t(mat1))/2); diag(mattype1)=0;
mattype2 = ceiling((mat2+t(mat2))/2); diag(mattype2)=0;

A = list()
for (i in 1:3){A[[i]]=mattype1} # first 3 are type-1
for (i in 4:6){A[[i]]=mattype2} # next  3 are type-2

## Compute Distance Matrix and Visualize
output = nd.dsd(A, out.dist=FALSE)
image(output$D, main="two group case")
```

---

nd.edd — *Edge Difference Distance*

Description

It is of the most simplest form that Edge Difference Distance (EDD) takestwo adjacency matrices and takes Frobenius norm of their differnces.

Usage

```
nd.edd(A, out.dist = TRUE)
```

Arguments

`A`	a list of length N containing $(M \times M)$ adjacency matrices.
`out.dist`	a logical; `TRUE` for computed distance matrix as a `dist` object.

Value

a named list containing

D an $(N \times N)$ matrix or `dist` object containing pairwise distance measures.

References

Hammond DK, Gur Y and Johnson CR (2013). “Graph Diffusion Distance: A Difference Measure for Weighted Graphs Based on the Graph Laplacian Exponential Kernel.” In *Proceedings of the IEEE global conference on information and signal processing (GlobalSIP'13)*, pp. 419–422. doi: 10.1109/GlobalSIP.2013.6736904, `http://www.sci.utah.edu/publications/hammond13/Hammond_GlobalSIP2013.pdf`.

Examples

```
## generate two types of adjacency matrices of size (3-by-3)
rbin1 = rbinom(9,1,0.8); mat1 = matrix(rbin1,nrow=3)
rbin2 = rbinom(9,1,0.1); mat2 = matrix(rbin2,nrow=3)

mattype1 = ceiling((mat1+t(mat1))/2); diag(mattype1)=0;
mattype2 = ceiling((mat2+t(mat2))/2); diag(mattype2)=0;

A = list()
for (i in 1:3){A[[i]]=mattype1} # first 3 are type-1
for (i in 4:6){A[[i]]=mattype2} # next  3 are type-2

## Compute Distance Matrix and Visualize
output = nd.edd(A, out.dist=FALSE)
image(output$D, main="two group case")
```

`nd.extremal` *Extremal distance with top-k eigenvalues*

Description

Extremal distance (`nd.extremal`) is a type of spectral distance measures on two graphs' graph Laplacian,

$$L := D - A$$

where A is an adjacency matrix and $D_{ii} = \sum_j A_{ij}$. It takes top-k eigenvalues from graph Laplacian matrices and take normalized sum of squared differences as metric. Note that it is *1. non-negative*, *2. separated*, *3. symmetric*, and satisfies *4. triangle inequality* in that it is indeed a metric.

Usage

```
nd.extremal(A, out.dist = TRUE, k = ceiling(nrow(A)/5))
```

Arguments

`A`	a list of length `N` containing adjacency matrices.
`out.dist`	a logical; `TRUE` for computed distance matrix as a `dist` object.
`k`	the number of largest eigenvalues to be used.

Value

a named list containing

D an $(N \times N)$ matrix or `dist` object containing pairwise distance measures.

spectra an $(N \times k)$ matrix where each row is top-k Laplacian eigenvalues.

References

Jakobson D and Rivin I (2002). "Extremal metrics on graphs I." *Forum Mathematicum*, **14**(1). ISSN 0933-7741, 1435-5337, doi: 10.1515/form.2002.002, https://www.degruyter.com/view/j/form.2002.14.issue-1/form.2002.002/form.2002.002.xml.

Examples

```
## generate two types of adjacency matrices of size (3-by-3)
rbin1 = rbinom(9,1,0.8); mat1 = matrix(rbin1,nrow=3)
rbin2 = rbinom(9,1,0.2); mat2 = matrix(rbin2,nrow=3)

mattype1 = ceiling((mat1+t(mat1))/2); diag(mattype1)=0;
mattype2 = ceiling((mat2+t(mat2))/2); diag(mattype2)=0;

A = list()
for (i in 1:3){A[[i]]=mattype1} # first 3 are type-1
for (i in 4:6){A[[i]]=mattype2} # next  3 are type-2

## Compute Distance Matrix and Visualize
output = nd.extremal(A, out.dist=FALSE, k=2)
image(output$D, main="two group case")
```

---

nd.gdd *Graph Diffusion Distance*

Description

Graph Diffusion Distance (nd.gdd) quantifies the difference between two weighted graphs of same size. It takes an idea from heat diffusion process on graphs via graph Laplacian exponential kernel matrices. For a given adjacency matrix A, the graph Laplacian is defined as

$$L := D - A$$

where $D_{ii} = \sum_j A_{ij}$. For two adjacency matrices A_1 and A_2, GDD is defined as

$$d_{gdd}(A_1, A_2) = max_t \sqrt{\| \exp(-tL_1) - \exp(-tL_2) \|_F}$$

where $\exp(\cdot)$ is matrix exponential, $\| \cdot \|_F$ a Frobenius norm, and L_1 and L_2 Laplacian matrices corresponding to A_1 and A_2, respectively.

Usage

```
nd.gdd(A, out.dist = TRUE, vect = seq(from = 0.1, to = 1, length.out = 10))
```

Arguments

A	a list of length N containing adjacency matrices.
out.dist	a logical; TRUE for computed distance matrix as a dist object.
vect	a vector of parameters t whose values will be used.

Value

a named list containing

D an $(N \times N)$ matrix or `dist` object containing pairwise distance measures.

maxt an $(N \times N)$ matrix whose entries are maximizer of the cost function.

References

Hammond DK, Gur Y and Johnson CR (2013). "Graph Diffusion Distance: A Difference Measure for Weighted Graphs Based on the Graph Laplacian Exponential Kernel." In *Proceedings of the IEEE global conference on information and signal processing (GlobalSIP'13)*, pp. 419–422. doi: 10.1109/GlobalSIP.2013.6736904, `http://www.sci.utah.edu/publications/hammond13/Hammond_GlobalSIP2013.pdf`.

Examples

```
## Not run:
## generate two types of adjacency matrices of size (3-by-3)
rbin1 = rbinom(9,1,0.8); mat1 = matrix(rbin1,nrow=3)
rbin2 = rbinom(9,1,0.2); mat2 = matrix(rbin2,nrow=3)

mattype1 = ceiling((mat1+t(mat1))/2); diag(mattype1)=0;
mattype2 = ceiling((mat2+t(mat2))/2); diag(mattype2)=0;

A = list()
for (i in 1:3){A[[i]]=mattype1} # first 3 are type-1
for (i in 4:6){A[[i]]=mattype2} # next  3 are type-2

## Compute Distance Matrix and Visualize
output = nd.gdd(A)
image(as.matrix(output$D), main="two group case")

## End(Not run)
```

`nd.hamming` *Hamming Distance*

Description

Hamming Distance is the count of discrepancy between two binary networks for each edge. Therefore, if used with non-binary networks, it might return a warning message and distorted results. It was originally designed to compare two strings of equal length, see Wikipedia page for more detailed introduction.

Usage

```
nd.hamming(A, out.dist = TRUE)
```

Arguments

A	a list of length N containing adjacency matrices.
out.dist	a logical; TRUE for computed distance matrix as a dist object.

Value

a named list containing

D an $(N \times N)$ matrix or dist object containing pairwise distance measures.

References

Hamming RW (1950). "Error Detecting and Error Correcting Codes." *Bell System Technical Journal*, **29**(2), pp. 147–160. ISSN 00058580, doi: 10.1002/j.15387305.1950.tb00463.x, http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6772729.

Examples

```
## generate two types of adjacency matrices of size (3-by-3)
rbin1 = rbinom(9,1,0.8); mat1 = matrix(rbin1,nrow=3)
rbin2 = rbinom(9,1,0.2); mat2 = matrix(rbin2,nrow=3)

mattype1 = ceiling((mat1+t(mat1))/2); diag(mattype1)=0;
mattype2 = ceiling((mat2+t(mat2))/2); diag(mattype2)=0;

A = list()
for (i in 1:3){A[[i]]=mattype1} # first 3 are type-1
for (i in 4:6){A[[i]]=mattype2} # next  3 are type-2

## Compute Distance Matrix and Visualize
output = nd.hamming(A)
image(as.matrix(output$D), main="two group case")
```

---

nd.him *HIM Distance*

Description

Hamming-Ipsen-Mikhailov (HIM) combines the local Hamming edit distance and the global Ipsen-Mikhailov distance to merge information at each scale. For Ipsen-Mikhailove distance, it is provided as nd.csd in our package for consistency. Given a parameter ξ (xi), it is defined as

$$HIM_\xi(A,B) = \sqrt{H^2(A,B) + \xi \cdot IM^2(A,B)}/\sqrt{1+\xi}$$

where H and IM stand for Hamming and I-M distance, respectively.

Usage

```
nd.him(A, out.dist = TRUE, xi = 1, ntest = 10)
```

Arguments

`A`	a list of length N containing $(M \times M)$ adjacency matrices.
`out.dist`	a logical; `TRUE` for computed distance matrix as a `dist` object.
`xi`	a parameter to control balance between two distances.
`ntest`	the number of searching over `nd.csd` parameter.

Value

a named list containing

D an $(N \times N)$ matrix or `dist` object containing pairwise distance measures.

References

Jurman G, Visintainer R, Filosi M, Riccadonna S and Furlanello C (2015). "The HIM glocal metric and kernel for network comparison and classification." In *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 1–10. ISBN 978-1-4673-8272-4, doi: 10.1109/DSAA.2015.7344816, `http://ieeexplore.ieee.org/document/7344816/`.

See Also

`nd.hamming`, `nd.csd`

Examples

```
## generate two types of adjacency matrices of size (3-by-3)
rbin1 = rbinom(9,1,0.8); mat1 = matrix(rbin1,nrow=3)
rbin2 = rbinom(9,1,0.2); mat2 = matrix(rbin2,nrow=3)

mattype1 = ceiling((mat1+t(mat1))/2); diag(mattype1)=0;
mattype2 = ceiling((mat2+t(mat2))/2); diag(mattype2)=0;

A = list()
for (i in 1:3){A[[i]]=mattype1} # first 3 are type-1
for (i in 4:6){A[[i]]=mattype2} # next  3 are type-2

## compute distance and visualize
output = nd.him(A, out.dist=FALSE)
image(output$D, main="two group case")
```

nd.wsd *Distance with Weighted Spectral Distribution*

Description

Normalized Laplacian matrix contains topological information of a corresponding network via its spectrum. nd.wsd adopts weighted spectral distribution of eigenvalues and brings about a metric via binning strategy.

Usage

```
nd.wsd(A, out.dist = TRUE, K = 50, wN = 4)
```

Arguments

A	a list of length N containing $(M \times M)$ adjacency matrices.
out.dist	a logical; TRUE for computed distance matrix as a dist object.
K	the number of bins for the spectrum interval $[0, 2]$.
wN	a decaying exponent; default is 4 set by authors.

Value

a named list containing

D an $(N \times N)$ matrix or dist object containing pairwise distance measures.

spectra an $(N \times M)$ matrix of rows being eigenvalues for each graph.

References

Fay D, Haddadi H, Thomason A, Moore A, Mortier R, Jamakovic A, Uhlig S and Rio M (2010). "Weighted Spectral Distribution for Internet Topology Analysis: Theory and Applications." *IEEE/ACM Transactions on Networking*, **18**(1), pp. 164–176. ISSN 1063-6692, 1558-2566, doi: 10.1109/TNET.2009.2022369, http://ieeexplore.ieee.org/document/5233839/.

Examples

```
## generate two types of adjacency matrices of size (3-by-3)
rbin1 = rbinom(9,1,0.8); mat1 = matrix(rbin1,nrow=3)
rbin2 = rbinom(9,1,0.2); mat2 = matrix(rbin2,nrow=3)

mattype1 = ceiling((mat1+t(mat1))/2)
mattype2 = ceiling((mat2+t(mat2))/2)

A = list()
for (i in 1:3){A[[i]]=mattype1} # first 3 are type-1
for (i in 4:6){A[[i]]=mattype2} # next  3 are type-2

## Compute Distance Matrix and Visualize
```

```
output = nd.wsd(A, out.dist=FALSE, K=10)
image(output$D, main="two group case")
```

`NetworkDistance` *Distance Measures for Networks*

Description

Network has gathered much attention from many disciplines, as many of real data can be well represented in the relational form. The concept of distance - or, metric - between two networks is the starting point for inference on population of networks. **NetworkDistance** package provides a not-so-comprehensive collection of distance measures for measuring dissimilarity between two network objects. Data should be supplied as *adjacency* matrices, where we support three formats of data representation; `matrix` object in **R** base, `network` class from **network** package, and `igraph` class from **igraph** package.

Index