Package: diffudist (via r-universe)

November 25, 2024

Title Diffusion Distance for Complex Networks

Version 1.0.1

URL https://gbertagnolli.github.io/diffudist/

BugReports https://github.com/gbertagnolli/diffudist/issues/

Description Enables the evaluation of diffusion distances for complex single-layer networks. Given a network one can define different types of Laplacian (or transition) matrices corresponding to different continuous-time random walks dynamics on the network. This package enables the evaluation of Laplacians, stochastic matrices, and the corresponding diffusion distance matrices. The metric structure induced by the network-driven process is richer and more robust than the one given by shortest-paths and allows to study the geometry induced by different types of diffusion-like communication mechanisms taking place on complex networks. For more details see: De Domenico, M. (2017) <doi:10.1103/physrevlett.118.168301> and Bertagnolli, G. and De Domenico, M. (2021) <doi:10.1103/PhysRevE.103.042301>.

Depends R (>= 3.5.0)

Imports expm, ggdendro, ggplot2, grid, igraph, Matrix, stats, RColorBrewer, Rcpp (>= 1.0.10), reshape2, rlang, viridis

LinkingTo Rcpp, RcppEigen

Suggests knitr, cowplot, parallelDist, strex, tidyr, rmarkdown

License GPL (>= 2)

Encoding UTF-8

RoxygenNote 7.2.3

VignetteBuilder knitr

NeedsCompilation yes

Author Giulia Bertagnolli [aut, cre] (<https://orcid.org/0000-0001-8637-0632>), Manlio De Domenico [aut] (<https://orcid.org/0000-0001-5158-8594>)

Maintainer Giulia Bertagnolli <giulia.bertagnolli@gmail.com>

Date/Publication 2023-02-27 19:42:40 UTC

Additional_repositories https://cranhaven.r-universe.dev

Config/pak/sysreqs libglpk-dev libicu-dev libxml2-dev

Repository https://cranhaven.r-universe.dev

RemoteUrl https://github.com/cranhaven/cranhaven.r-universe.dev

RemoteRef package/diffudist

RemoteSha 8d61ca12b41bca288ce55d241ab3b47f062beeb0

Contents

`eigenMapMatMult` *Matrix Multiplication using RcppEigen*

Description

Matrix multiplication of the two matrices in input, without copy.

Usage

```
eigenMapMatMult(A, B)
```

Arguments

`A`	numeric matrix of dimension $m \times n$
`B`	numeric matrix of dimension $n \times l$

Value

C matrix of dimension $m \times p$ of the row-column product of A and B and C

eigenMatMult *Matrix Multiplication using RcppEigen*

Description

Matrix multiplication of the two matrices in input.

Usage

```
eigenMatMult(A, B)
```

Arguments

A	numeric matrix of dimension $m \times n$
B	numeric matrix of dimension $n \times l$

Value

C matrix of dimension $m \times p$ of the row-column product of A and B and C

get_ddm_from_eigendec *Distance Matrix from Laplacian spectral decomposition*

Description

Returns the diffusion distance matrix when the spectrum (more precisely, the eigendecomposition) of the Laplacian is provided as input (useful to speed up batch calculations).

For instance, the random walk normalised Laplacian $I - D^{-1}A$, which generates the classical continuous-time random walk over a network, can be easily and obtained from the spectral decomposition of the symmetric normalised Laplacian $\mathcal{L} = D^{-\frac{1}{2}}LD^{-\frac{1}{2}} = D^{-\frac{1}{2}}(D - A)D^{-\frac{1}{2}}$. More specifically, $\bar{L} = I - D^{-1}A = D^{-\frac{1}{2}}\mathcal{L}D^{\frac{1}{2}}$ and, since $\mathcal{L}$ is symmetric it can be decomposed into $\mathcal{L} = \sum_{l=1}^{N} \lambda_l u_l u_l^T$, hence

$$\bar{L} = \sum_{l=1}^{N} \lambda_l u_l^R u_l^L$$

where $u_l^L = u_l^T D^{\frac{1}{2}}$ and $u_l^R = u_l D^{-\frac{1}{2}}$.

Usage

```
get_ddm_from_eigendec(tau, Q, Q_inv, lambdas, verbose = FALSE)
```

Arguments

tau	diffusion time (scalar)
Q	eigenvector matrix
Q_inv	inverse of the eigenvector matrix
lambdas	eigenvalues (vector)
verbose	whether warnings have to be printed or not

Value

The diffusion distance matrix D_t, a square numeric matrix of the Euclidean distances between the rows of the stochastic matrix $P(t) = e^{-\tau L}$, where $-L$ is the Laplacian generating a continuous-time random walk (Markov chain) over the network. The matrix exponential is here computed using the given eigendecomposition of the Laplacian matrix $e^{-\tau L} = Qe^{-\tau \Lambda}Q^{-1}$.

References

Bertagnolli, G., & De Domenico, M. (2021). Diffusion geometry of multiplex and interdependent systems. Physical Review E, 103(4), 042301. doi:10.1103/PhysRevE.103.042301 arXiv:2006.13032

See Also

get_spectral_decomp

---

get_diffusion_probability_matrix
Diffusion Probability Matrix

Description

Returns a matrix where each entry encodes the diffusion probability between two nodes

Usage

```
get_diffusion_probability_matrix(
  g,
  tau,
  type = "Normalized Laplacian",
  weights = NULL,
  verbose = TRUE
)

getDiffusionProbabilityMatrix(g, tau, type = "Normalized Laplacian", weights = NULL,
                                      verbose = TRUE)

get_diffu_Pt(
```

```
  g,
  tau,
  type = "Normalized Laplacian",
  weights = NULL,
  verbose = TRUE
)
```

Arguments

g	a single-layer network
tau	diffusion time
type	default "Normalized Laplacian". The type of Laplacian (i.e. of dynamics) to consider. Other types available are: **Laplacian** for the classical combinatorial Laplacian matrix; it governs the diffusion dynamics on the network **Normalized Laplacian** for the Laplacian matrix normalized by degree matrix, the so-called classical random walk normalized Laplacian; it governs stochastic walks on the network **Quantum Laplacian** for the Laplacian matrix normalized to be symmetric; it governs quantum walks on the network **MERW normalized Laplacian** the maximal-entropy random walk (RW) normalized Laplacian; it governs stochastic walks on the network, in which the random walker moves according to a maximal-entropy RW [1]. Note that you can type abbreviations, e.g. "L", "N", "Q", "M" for the respective types (case is ignored). The argument match is done through `match_arg`.
weights	edge weights, representing the strength/intensity (not the cost!) of each link. if weights is NULL (the default) and g has an edge attribute called weight, then it will be used automatically. If this is NA then no weights are used (even if the graph has a weight attribute).
verbose	default `TRUE`. If information on the type of Laplacian or on edge weights should be printed.

Value

The matrix $exp^{-\tau L}$, exponential of a Laplacian matrix.

Functions

- `getDiffusionProbabilityMatrix()`: Old deprecated function

References

De Domenico, M. (2017). Diffusion Geometry Unravels the Emergence of Functional Clusters in Collective Phenomena. Physical Review Letters. doi:10.1103/PhysRevLett.118.168301

Bertagnolli, G., & De Domenico, M. (2021). Diffusion geometry of multiplex and interdependent systems. Physical Review E, 103(4), 042301. doi:10.1103/PhysRevE.103.042301 arXiv:2006.13032

See Also

get_laplacian, get_distance_matrix

`get_diffusion_probability_matrix_from_T`
Diffusion probability matrix from transition matrix

Description

Description here

Usage

```
get_diffusion_probability_matrix_from_T(Pi, tau)

get_diffu_Pt_from_T(Pi, tau)

get_diffu_Pt_from_Pi(Pi, tau)

get_diffusion_probability_matrix_from_Pi(Pi, tau)
```

Arguments

`Pi`	Transition matrix. We do not use `T` to avoid conflicts with the abbreviation for `TRUE`), instead we indicated the transition matrix with the capital greek letter Π in the equations and `Pi` in the code.
`tau`	diffusion time

Value

$exp^{-\tau(I-\Pi)}$, exponential of the normalized Laplacian matrix corresponding to the given transition rate matrix (or transition probability matrix of a discrete-time Markov chain).

References

De Domenico, M. (2017). Diffusion Geometry Unravels the Emergence of Functional Clusters in Collective Phenomena. Physical Review Letters. doi:10.1103/PhysRevLett.118.168301

Bertagnolli, G., & De Domenico, M. (2020). Diffusion Geometry of Multiplex and Interdependent Systems. arxiv preprint arxiv:2006.13032

See Also

get_diffusion_probability_matrix

`get_distance_matrix` *Diffusion Distance Matrix*

Description

Returns a matrix where each entry encodes the diffusion distance between two nodes of a network.

The diffusion distance at time τ between nodes $i, j \in G$ is defined as

$$D_\tau(i,j) = |\mathbf{p}(t|i) - \mathbf{p}(t|j)|_2$$

with $\mathbf{p}(t|i) = (e^{-\tau L})_{i\cdot} = \mathbf{e}_i e^{-\tau L}$ indicating the i-th row of the stochastic matrix $e^{-\tau L}$ and representing the probability (row) vector of a random walk dynamics corresponding to the initial condition $\mathbf{e}_i$, i.e. the random walker is in node i at time $\tau = 0$ with probability 1.

Usage

```
get_distance_matrix(
  g,
  tau,
  type = "Normalized Laplacian",
  weights = NULL,
  as_dist = FALSE,
  verbose = TRUE
)

getDistanceMatrix(g, tau, type = "Normalized Laplacian", weights = NULL,
                          verbose = TRUE)

get_DDM(
  g,
  tau,
  type = "Normalized Laplacian",
  weights = NULL,
  as_dist = FALSE,
  verbose = TRUE
)
```

Arguments

`g`	a (single-layer) network
`tau`	diffusion time
`type`	default "Normalized Laplacian". The type of Laplacian (i.e. of dynamics) to consider. Other types available are:
	Laplacian for the classical combinatorial Laplacian matrix; it governs the diffusion dynamics on the network

Normalized Laplacian for the Laplacian matrix normalized by degree matrix, the so-called classical random walk normalized Laplacian; it governs stochastic walks on the network

Quantum Laplacian for the Laplacian matrix normalized to be symmetric; it governs quantum walks on the network

MERW normalized Laplacian the maximal-entropy random walk (RW) normalized Laplacian; it governs stochastic walks on the network, in which the random walker moves according to a maximal-entropy RW [1].

Note that you can type abbreviations, e.g. "L", "N", "Q", "M" for the respective types (case is ignored). The argument match is done through `match_arg`.

`weights` edge weights, representing the strength/intensity (not the cost!) of each link. If weights is NULL (the default) and g has an edge attribute called weight, then it will be used automatically. If this is NA then no weights are used (even if the graph has a weight attribute).

`as_dist` If the function should return a matrix or an object of class "dist" as returned from [stats::as.dist]. Default is FALSE if the number of nodes is smaller than 1000.

`verbose` default TRUE

Value

The diffusion distance matrix D_t, a square numeric matrix of the L^2-norm distances between posterior probability vectors, i.e. Euclidean distances between the rows of the stochastic matrix $P(t) = e^{-\tau L}$, where $-L = -(I - T)$ is the generator of the continuous-time random walk (Markov chain) of given `type` over network `g`.

Functions

- `getDistanceMatrix()`: Old deprecated function

References

De Domenico, M. (2017). Diffusion Geometry Unravels the Emergence of Functional Clusters in Collective Phenomena. Physical Review Letters. doi:10.1103/PhysRevLett.118.168301

Bertagnolli, G., & De Domenico, M. (2021). Diffusion geometry of multiplex and interdependent systems. Physical Review E, 103(4), 042301. doi:10.1103/PhysRevE.103.042301 arXiv:2006.13032

See Also

`get_diffusion_probability_matrix`

Examples

```
g <- igraph::sample_pa(10, directed = FALSE)
dm_crw <- get_distance_matrix(g, tau = 1)
dm_merw <- get_distance_matrix(g, tau = 1, type = "MERW")
```

`get_distance_matrix_from_T` *Diffusion distance matrix from a custom transition matrix*

Description

Returns a matrix where each entry encodes the diffusion distance between two nodes of a network, given a transition matrix on the network and a diffusion time.

The diffusion distance at time τ between nodes $i, j \in G$ is defined as

$$D_\tau(i,j) = |\mathbf{p}(t|i) - \mathbf{p}(t|j)|_2$$

with $\mathbf{p}(t|i) = (e^{-\tau L})_{i\cdot} = \mathbf{e}_i e^{-\tau L}$ indicating the i-th row of the stochastic matrix $e^{-\tau L}$ and representing the probability (row) vector of a random walk dynamics corresponding to the initial condition $\mathbf{e}_i$, i.e. the random walker is in node i at time $\tau = 0$ with probability 1.

The Laplacian L is the normalised laplacian corresponding to the given transition matrix, i.e. $L = I - Pi$.

Usage

```
get_distance_matrix_from_T(Pi, tau, verbose = TRUE)

get_DDM_from_T(Pi, tau, verbose = TRUE)

get_distance_matrix_from_Pi(Pi, tau, verbose = TRUE)

get_DDM_from_Pi(Pi, tau, verbose = TRUE)
```

Arguments

`Pi`	a transition matrix (it should be a stochastic matrix)
`tau`	diffusion time
`verbose`	default TRUE

Value

The diffusion distance matrix D_t, a square numeric matrix of the L^2-norm distances between posterior probability vectors, i.e. Euclidean distances between the rows of the stochastic matrix $P(t) = e^{-\tau L}$, where $-L = -(I - T)$ is the generator of the continuous-time random walk (Markov chain) corresponding to the discrete-time transition matrix $T =$`Pi`.

References

De Domenico, M. (2017). Diffusion Geometry Unravels the Emergence of Functional Clusters in Collective Phenomena. Physical Review Letters. doi:10.1103/PhysRevLett.118.168301

Bertagnolli, G., & De Domenico, M. (2021). Diffusion geometry of multiplex and interdependent systems. Physical Review E, 103(4), 042301. doi:10.1103/PhysRevE.103.042301 arXiv:2006.13032

See Also

get_distance_matrix get_diffusion_probability_matrix,get_diffusion_probability_matrix_from_T

Examples

```
g <- igraph::sample_pa(10, directed = FALSE)
dm <- get_distance_matrix(g, tau = 1)
```

---

get_laplacian *Evaluate a Laplacian Matrix*

Description

Returns a specific Laplacian matrix corresponding to the chosen dynamics type and network. The available types are:

Laplacian for the classical combinatorial Laplacian matrix; it governs the diffusion dynamics on the network

Normalized Laplacian for the Laplacian matrix normalized by degree matrix, the so-called classical random walk normalized Laplacian; it governs stochastic walks on the network

Quantum Laplacian for the Laplacian matrix normalized to be symmetric; it governs quantum walks on the network

MERW normalized Laplacian the maximal-entropy random walk (RW) normalized Laplacian; it governs stochastic walks on the network, in which the random walker moves according to a maximal-entropy RW [1].

The maximum entropy random walk (MERW) chooses the stochastic matrix which maximizes $H(S)$, so that the walker can explore every walk of the same length with equal probability. Let λ_N, ϕ be the leading eigenvalue and corresponding right eigenvector of the adjacency matrix A. Then the transition matrix corresponding to the discrete-time random walk is $\Pi_{ij} = \frac{A_{ij}}{\lambda_N}\frac{\phi_j}{\phi_i}$. The MERW (normalized) Laplacian is then given by $I - \Pi$. Note that we use the notation Π and `Pi` to avoid confusion with the abbreviation `T` for the logical `TRUE`.

Usage

```
get_laplacian(g, type = "Laplacian", weights = NULL, verbose = TRUE)

getLaplacianMatrix(g, type = "Laplacian", weights = NULL, verbose = TRUE)
```

Arguments

g	a network
type	the type of Laplacian matrix. default `"Laplacian"`, the combinatorial Laplacian. Other types: `c("Normalized Laplacian", "Quantum Laplacian", "MERW Normalized Laplacian")`. Note that you can type abbreviations, e.g. "L", "N", "Q", "M" for the respective types (case is ignored). The argument match is done through match_arg.

weights	edge weights, representing the strength/intensity (not the cost!) of each link. if weights is NULL (the default) and g has an edge attribute called weight, then it will be used automatically. If this is NA then no weights are used (even if the graph has a weight attribute).
verbose	default TRUE. If information on the type of Laplacian or on edge weights should be printed.

Value

the ('type') Laplacian matrix of network 'g'

References

[1] Burda, Z., et al. (2009). Phys Rev. Lett. 102 160602(April), 1–4. doi:10.1103/PhysRevLett.102.160602

`get_mean_distance_matrix` *Mean distance matrix*

Description

Given a sequence of distance matrices, expected to correspond to sequential increasing values of time, calculate the average distance between any pairs of nodes shortest-path (or geodesic) distance

Usage

```
get_mean_distance_matrix(DM_list)

getMeanDistanceMatrix(DM_list)
```

Arguments

DM_list	list of distance matrices

Value

mean-distance matrix

Functions

- `getMeanDistanceMatrix()`: Old deprecated function

References

De Domenico, M. (2017). Diffusion Geometry Unravels the Emergence of Functional Clusters in Collective Phenomena. Physical Review Letters. doi:10.1103/PhysRevLett.118.168301

Bertagnolli, G., & De Domenico, M. (2020). Diffusion Geometry of Multiplex and Interdependent Systems. arxiv preprint arxiv:2006.13032

See Also

get_diffusion_probability_matrix, get_distance_matrix

---

get_spectral_decomp *Laplacian Spectral Decomposition*

Description

Returns the eigenvalue spectrum together with eigenvectors of a Laplacian corresponding to a network. This involves computing the eigendecomposition of a (symmetric) matrix, so it is computationally intense and may take some time. The decomposition of the normalized Laplacian $L = I - D^{-1}A$ takes is computed through the decomposition of its symmetric version $L = D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$. See the package vignette for details.

Usage

```
get_spectral_decomp(g, type = "Normalized Laplacian", verbose = FALSE)
```

Arguments

g	the network in the [igraph] format
type	the Laplacian type, default "Normalized Laplacian". At the moment this is the only available option. For other types of Laplacians one should get autonomously the eigendecomposition, e.g. using eigen. See the package vignette for an example.
verbose	whether warnings have to be printed or not

Value

lambdas the eigenvalues of the Laplacian

'u_L' the matrix of left eigenvectors (rows)

'u_R' the matrix of right eigenvectors (columns)

References

Bertagnolli, G., & De Domenico, M. (2021). Diffusion geometry of multiplex and interdependent systems. Physical Review E, 103(4), 042301. doi:10.1103/PhysRevE.103.042301 arXiv:2006.13032

See Also

get_laplacian get_ddm_from_eigendec

plotHeatmap *Plot distance matrix as heatmap*

Description

‘r lifecycle::badge("deprecated")‘

Usage

```
plotHeatmap(
  DM,
  colPalette = NULL,
  log.scale = FALSE,
  cex = 1,
  showDendrogram = TRUE,
  title = ""
)
```

Arguments

DM	a distance matrix
colPalette	‘r lifecycle::badge("deprecated")‘ in the new function plot_distance_matrix it is called col_palette
log.scale	‘r lifecycle::badge("deprecated")‘ in the new function plot_distance_matrix it is called log_scale
cex	numerical value by which text should be magnified (default font size in [ggplot2:theme] is 11)
showDendrogram	‘r lifecycle::badge("deprecated")‘ in the new function plot_distance_matrix it is called show_dendro
title	Title of the plot passed to labs. No title by default

Value

a ggplot

plot_distance_matrix *Plot distance matrix*

Description

Plot a heatmap of the distance matrix using geom_tile.

Usage

```
plot_distance_matrix(
  DM,
  col_palette = viridis(n = 11),
  log_scale = FALSE,
  cex = 1,
  show_dendro = TRUE,
  title = ""
)
```

Arguments

DM	a distance matrix
col_palette	a colour palette, previously it was set to the 'spectral' palette of brewer.pal but now it is, by default, to the color-blind firendly viridis.
log_scale	logical. Default FALSE
cex	numerical value by which text should be magnified (default font size in theme is 11)
show_dendro	If the dendrogram resulting from hclust should be shown. Default TRUE
title	Title of the plot passed to labs. No title by default.

Value

plot ggplot

Index