Package ‘ScorePlus’

October 12, 2022

Title Implementation of SCORE, SCORE+ and Mixed-SCORE
Version 0.1
Maintainer Shengming Luo <shengmil@andrew.cmu.edu>
License GPL-2
Encoding UTF-8
LazyData true
RoxygenNote 6.1.1
Imports utils, combinat, limSolve, RSpectra, igraph, igraphdata, stats
NeedsCompilation no
Author Jiashun Jin [aut],
Zheng Tracy Ke [aut],
Shengming Luo [aut, cre]
Repository CRAN
Date/Publication 2019-06-14 08:40:03 UTC

R topics documented:

getMaxDist ................................................................. 2
getMembership ............................................................. 2
mixedSCORE ................................................................. 3
SCORE ....................................................................... 4
SCOREplus ................................................................. 4
vertexHunting ............................................................... 5
vertexSearch ............................................................... 6

Index 7
getMaxDist

find the maximum distance from the convex hull formed by the chosen K vertices

Description
find the maximum distance from the convex hull formed by the chosen K vertices

Usage
getMaxDist(centers, vertex.ind)

Arguments

- centers: L-by-(K-1) center matrix
- vertex.ind: index of the K centers that forms the convex hull

Value
the maximum distance

getMembership

calculated the membership of each node given ratio matrix and community centers

Description
calculated the membership of each node given ratio matrix and community centers

Usage
getMembership(R, vertices, K, eig.values, eig.vectors)

Arguments

- R: n-by-(K-1) ratio matrix
- vertices: K-by-(K-1) community centers
- K: number of communities.
- eig.values: eigenvalues of adjacency matrix.
- eig.vectors: eigenvectors of adjacency matrix.

Value
n-by-K membership matrix
mixedSCORE

Membership estimation algorithm called mixedSCORE

Description

Membership estimation algorithm called mixedSCORE

Usage

mixedSCORE(A, K, verbose = F)

Arguments

A n-by-n binary symmetric adjacency matrix.
K number of communities.
verbose whether generate message

Value

A list containing

R n-by-(K-1) ratio matrix.
L Selected tuning parameter used for vertex hunting algorithm.
theta A vector of the estimated degree heterogeneity parameters
vertices K-by-(K-1) K vertices of the found convex hull
centers L-by-(K-1) L centers by kmeans
memberships n-by-K membership matrix.
purity A vector of maximum membership of each node
hard.cluster.labels A vector of integers indicating hard clustering labels, by assigning the node to the cluster with max membership

Examples

library(igraphdata)
library(igraph)
data('karate')
A = get.adjacency(karate)
kartae.mixed.out = mixedSCORE(A, 2)
kartae.mixed.out$memberships
**SCORE**

community detection method called **SCORE Spectral Clustering On Ratios-of-Eigenvectors (SCORE)**

**Description**

community detection method called SCORE Spectral Clustering On Ratios-of-Eigenvectors (SCORE)

**Usage**

```r
SCORE(A, K, threshold = NULL)
```

**Arguments**

- **A**: n-by-n binary symmetric adjacency matrix.
- **K**: number of communities.
- **threshold**: (optional) the threshold of ratio matrix. By default is \( \log(n) \).

**Value**

A list containing

- **R**: n-by-(K-1) ratio matrix.
- **labels**: A vector of integer indicating the cluster to which each point allocated.

**Examples**

```r
library(igraphdata)
library(igraph)
data('karate')
A = get.adjacency(karate)
karate.out = SCORE(A, 2)
karate.out$labels
```

---

**SCOREplus**

community detection method called **SCORE+**

**Description**

community detection method called SCORE+

**Usage**

```r
SCOREplus(A, k, c = 0.1, r = NULL)
```
vertexHunting

Arguments

A  n-by-n binary symmetric adjacency matrix.

k  number of communities (>1).

c  (optional) tuning parameter for Graph Laplacian, default is 0.1.

r  (optional) latent dimension (>1), if not given, chosen between k and k+1 determined by eigen gap

Value

A list containing

- label  Predicted community labels
- ratios  n-by-(K-1) or n-by-r ratio matrix.
- delta  calculated delta parameter
- eig.vec  Top r eigen vectors
- eig.val  Top r eigen values

Examples

```r
library(igraphdata)
library(igraph)
data('karate')
A = get.adjacency(karate)
karate.plus.out = SCOREplus(A, 2)
karate.plus.out$labels
```

vertexHunting  Vertex hunting algorithm to find the cluster centers

Description

Vertex hunting algorithm to find the cluster centers

Usage

```
vertexHunting(R, K, verbose = F)
```

Arguments

R  n-by-(K-1) ratio matrix

K  number of communities.

verbose  whether or not to show a progress bar
**vertexSearch**

*select the K vertices from given L centers*

**Description**

select the K vertices from given L centers

**Usage**

vertexSearch(centers, K)

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>centers</td>
<td>L-by-(K-1) center matrix</td>
</tr>
<tr>
<td>K</td>
<td>number of communities.</td>
</tr>
</tbody>
</table>

**Value**

A list containing

- **ind** a vector of K integers indicating the index of selected K vertices out of L centers.
- **dist** The maximum distance from centers to the convex hull formed by the K selected vertices.
Index

getMaxDist, 2
getMembership, 2

mixedSCORE, 3

SCORE, 4
SCOREplus, 4

vertexHunting, 5
vertexSearch, 6