Package ‘AMOUNTAIN’

November 27, 2025

Type Package

Title Active modules for multilayer weighted gene co-expression networks: a continuous optimization approach

Version 1.36.0

Date 2016-11-12

Author Dong Li, Shan He, Zhisong Pan and Guyu Hu

Maintainer Dong Li <dxl466@cs.bham.ac.uk>

Description

A pure data-driven gene network, weighted gene co-expression network (WGCN) could be constructed only from expression profile. Different layers in such networks may represent different time points, multiple conditions or various species. AMOUNTAIN aims to search active modules in multi-layer WGCN using a continuous optimization approach.

License GPL (>= 2)

Depends R (>= 3.3.0)

Imports stats

RoxygenNote 5.0.1

SystemRequirements gsl

biocViews GeneExpression, Microarray, DifferentialExpression, Network

Suggests BiocStyle, qgraph, knitr, rmarkdown

VignetteBuilder knitr

git_url https://git.bioconductor.org/packages/AMOUNTAIN

git_branch RELEASE_3_22

git_last_commit 5e47cdb

git_last_commit_date 2025-10-29

Repository Bioconductor 3.22

Date/Publication 2025-11-27

Contents

CGPFixSS 2
CGPFixSSMultiLayer 3
CGPFixSSTwolayer 4
EuclideanProjectionENNORM 5

moduleIdentificationGPFixSS 6
moduleIdentificationGPFixSSMultilayer 7
moduleIdentificationGPFixSSTwolayer 8
multilayernetworkSimulation 10
networkSimulation 11
twolayernetworkSimulation 11

Index **13**

`CGPFixSS` *Module Identification*

Description

Call C version of moduleIdentificationGPFixSS

Usage

```
CGPFixSS(W, z, x0, a = 0.5, lambda = 1, maxiter = 50)
```

Arguments

`W`	edge score matrix of the network, n x n matrix
`z`	node score vector of the network, n-length vector
`x0`	initial solution, n-length vector
`a`	parameter in elastic net the same as in `EuclideanProjectionENNORM`
`lambda`	parameter in objective, coefficient of node score part
`maxiter`	maximal interation of whole procedure

Value

a list containing function objective vector and the solution

Author(s)

Dong Li, <dxl466@cs.bham.ac.uk>

References

AMOUNTAIN

See Also

`moduleIdentificationGPFixSS`

Examples

```
n = 100
k = 20
theta = 0.5
pp <- networkSimulation(n,k,theta)
moduleid <- pp[[3]]
## use default parameters here
x <- CGPFixSS(pp[[1]],pp[[2]],rep(1/n,n))
predictedid<-which(x[[2]]!=0)
recall <- length(intersect(predictedid,moduleid))/length(moduleid)
precise <- length(intersect(predictedid,moduleid))/length(predictedid)
Fscore <- (2*precise*recall/(precise+recall))
```

CGPFixSSMultiLayer *Module Identification for multi-layer network*

Description

Call C version of moduleIdentificationGPFixSSMultilayer

Usage

```
CGPFixSSMultiLayer(W, listzs, x0, a = 0.5, lambda = 1, maxiter = 50)
```

Arguments

W	edge score matrix of the network, n x n matrix
listzs	a list of node score vectors, each layer has a n-length vector
x0	initial solution, n-length vector
a	parameter in elastic net the same as in EuclideanProjectionENNORM
lambda	parameter in objective, coefficient of node score of other layers
maxiter	maximal interation of whole procedure

Value

a list containing solution for network 1 and network 2

Author(s)

Dong Li, <dxl466@cs.bham.ac.uk>

References

AMOUNTAIN

See Also

moduleIdentificationGPFixSSMultilayer

Examples

```
n = 100
k = 20
L = 5
theta = 0.5
cpl <- multilayernetworkSimulation(n,k,theta,L)
listz <- list()
for (i in 1:L){
listz[[i]] <- cpl[[i+2]]
}
moduleid <- cpl[[2]]
## use default parameters here
x <- CGPFixSSMultiLayer(cpl[[1]],listz,rep(1/n,n))
predictedid <- which(x[[2]]!=0)
recall <- length(intersect(predictedid,moduleid))/length(moduleid)
precise <- length(intersect(predictedid,moduleid))/length(predictedid)
Fscore <- (2*precise*recall/(precise+recall))
```

CGPFixSSTwolayer *Module Identification for two-layer network*

Description

Call C version of moduleIdentificationGPFixSSTwolayer

Usage

```
CGPFixSSTwolayer(W1, z1, x0, W2, z2, y0, interlayerA, lambda1 = 1,
  lambda2 = 1, lambda3 = 1, maxiter = 100, a1 = 0.5, a2 = 0.5)
```

Arguments

W1	edge score matrix of the network 1, n_1 x n_1 matrix
z1	node score vector of the network 1, n_1-length vector
x0	initial solution of network 1, n_1-length vector
W2	edge score matrix of the network 2, n_2 x n_2 matrix
z2	node score vector of the network 2, n_2-length vector
y0	initial solution of network 2, n_2-length vector
interlayerA	inter-layer links weight, n_1 x n_2 matrix
lambda1	parameter in objective, coefficient of node score of network 1
lambda2	parameter in objective, coefficient of node score of network 2
lambda3	parameter in objective, coefficient of inter-layer links part
maxiter	maximal interation of whole procedure
a1	parameter in elastic net the same as in EuclideanProjectionENNORM
a2	parameter in elastic net the same as in EuclideanProjectionENNORM

Value

a list containing solution for network 1 and network 2 and objective

Author(s)

Dong Li, <dxl466@cs.bham.ac.uk>

References

AMOUNTAIN

See Also

moduleIdentificationGPFixSSTwolayer

Examples

```
n1=100
k1=20
theta1 = 0.5
n2=80
k2=10
theta2 = 0.5
ppresult <- twolayernetworkSimulation(n1,k1,theta1,n2,k2,theta2)
A <- ppresult[[3]]
pp <- ppresult[[1]]
moduleid <- pp[[3]]
netid <- 1:n1
restp<- netid[-moduleid]
pp2 <- ppresult[[2]]
moduleid2 <- pp2[[3]]
## use default parameters here
modres=CGPFixSSTwolayer(pp[[1]],pp[[2]],rep(1/n1,n1),
pp2[[1]],pp2[[2]],rep(1/n2,n2),A)
predictedid<-which(modres[[1]]!=0)
recall = length(intersect(predictedid,moduleid))/length(moduleid)
precise = length(intersect(predictedid,moduleid))/length(predictedid)
F1 = 2*precise*recall/(precise+recall)
predictedid2<-which(modres[[2]]!=0)
recall2 = length(intersect(predictedid2,moduleid2))/length(moduleid2)
precise2 = length(intersect(predictedid2,moduleid2))/length(predictedid2)
F2 = 2*precise2*recall2/(precise2+recall2)
```

EuclideanProjectionENNORM *Euclidean projection on elastic net*

Description

Piecewise root finding algorithm for Euclidean projection on elastic net

Usage

```
EuclideanProjectionENNORM(y, t, alpha = 0.5)
```

Arguments

`y`	constant vector
`t`	radius of elastic net ball
`alpha`	parameter in elastic net: alpha x_1 + (1-alpha)*x_2^2=t

Value

a list containing network adjacency matrix, node score and module membership

Author(s)

Dong Li, <dxl466@cs.bham.ac.uk>

References

Gong, Pinghua, Kun Gai, and Changshui Zhang. "Efficient euclidean projections via piecewise root finding and its application in gradient projection." Neurocomputing 74.17 (2011): 2754-2766.

Examples

```
y=rnorm(100)
x=EuclideanProjectionENNORM(y,1,0.5)
sparistyx = sum(x==0)/100
```

---

moduleIdentificationGPFixSS *Module Identification*

Description

Algorithm for Module Identification on single network

Usage

```
moduleIdentificationGPFixSS(W, z, x0, a = 0.5, lambda = 1, maxiter = 1000)
```

Arguments

`W`	edge score matrix of the network, n x n matrix
`z`	node score vector of the network, n-length vector
`x0`	initial solution, n-length vector
`a`	parameter in elastic net the same as in `EuclideanProjectionENNORM`
`lambda`	parameter in objective, coefficient of node score part
`maxiter`	maximal interation of whole procedure

Value

a list containing function objective vector and the solution

Author(s)

Dong Li, <dxl466@cs.bham.ac.uk>

References

AMOUNTAIN

See Also

EuclideanProjectionENNORM

Examples

```
n = 100
k = 20
theta = 0.5
pp <- networkSimulation(n,k,theta)
moduleid <- pp[[3]]
## use default parameters here
x <- moduleIdentificationGPFixSS(pp[[1]],pp[[2]],rep(1/n,n))
predictedid<-which(x[[2]]!=0)
recall <- length(intersect(predictedid,moduleid))/length(moduleid)
precise <- length(intersect(predictedid,moduleid))/length(predictedid)
Fscore <- (2*precise*recall/(precise+recall))
```

---

moduleIdentificationGPFixSSMultilayer
Module Identification for multi-layer network

Description

Algorithm for Module Identification on multi-layer network sharing the same set of genes

Usage

```
moduleIdentificationGPFixSSMultilayer(W, listz, x0, a = 0.5, lambda = 1,
  maxiter = 1000)
```

Arguments

W	edge score matrix of the network, n x n matrix
listz	a list of node score vectors, each layer has a n-length vector
x0	initial solution, n-length vector
a	parameter in elastic net the same as in EuclideanProjectionENNORM
lambda	parameter in objective, coefficient of node score of other layers
maxiter	maximal interation of whole procedure

Value

a list containing objective values and solution

Author(s)

Dong Li, <dxl466@cs.bham.ac.uk>

References

AMOUNTAIN

See Also

moduleIdentificationGPFixSSMultilayer

Examples

```
n = 100
k = 20
L = 5
theta = 0.5
cpl <- multilayernetworkSimulation(n,k,theta,L)
listz <- list()
for (i in 1:L){
listz[[i]] <- cpl[[i+2]]
}
moduleid <- cpl[[2]]
## use default parameters here
x <- moduleIdentificationGPFixSSMultilayer(cpl[[1]],listz,rep(1/n,n))
predictedid <- which(x[[2]]!=0)
recall <- length(intersect(predictedid,moduleid))/length(moduleid)
precise <- length(intersect(predictedid,moduleid))/length(predictedid)
Fscore <- (2*precise*recall/(precise+recall))
```

---

moduleIdentificationGPFixSSTwolayer

Module Identification for two-layer network

Description

Algorithm for Module Identification on two-layer network

Usage

```
moduleIdentificationGPFixSSTwolayer(W1, z1, x0, W2, z2, y0, A, lambda1 = 1,
  lambda2 = 1, lambda3 = 1, maxiter = 1000, a1 = 0.5, a2 = 0.5)
```

Arguments

W1	edge score matrix of the network 1, n_1 x n_1 matrix
z1	node score vector of the network 1, n_1-length vector
x0	initial solution of network 1, n_1-length vector
W2	edge score matrix of the network 2, n_2 x n_2 matrix
z2	node score vector of the network 2, n_2-length vector
y0	initial solution of network 2, n_2-length vector

A	inter-layer links weight, n_1 x n_2 matrix
lambda1	parameter in objective, coefficient of node score of network 1
lambda2	parameter in objective, coefficient of node score of network 2
lambda3	parameter in objective, coefficient of inter-layer links part
maxiter	maximal interation of whole procedure
a1	parameter in elastic net the same as in EuclideanProjectionENNORM
a2	parameter in elastic net the same as in EuclideanProjectionENNORM

Value

a list containing solution for network 1 and network 2 and objective

Author(s)

Dong Li, <dxl466@cs.bham.ac.uk>

References

AMOUNTAIN

See Also

EuclideanProjectionENNORM

Examples

```
n1=100
k1=20
theta1 = 0.5
n2=80
k2=10
theta2 = 0.5
ppresult <- twolayernetworkSimulation(n1,k1,theta1,n2,k2,theta2)
A <- ppresult[[3]]
pp <- ppresult[[1]]
moduleid <- pp[[3]]
netid <- 1:n1
restp<- netid[-moduleid]
pp2 <- ppresult[[2]]
moduleid2 <- pp2[[3]]
## use default parameters here
modres=moduleIdentificationGPFixSSTwolayer(pp[[1]],pp[[2]],rep(1/n1,n1),
pp2[[1]],pp2[[2]],rep(1/n2,n2),A)
predictedid<-which(modres[[1]]!=0)
recall = length(intersect(predictedid,moduleid))/length(moduleid)
precise = length(intersect(predictedid,moduleid))/length(predictedid)
F1 = 2*precise*recall/(precise+recall)
predictedid2<-which(modres[[2]]!=0)
recall2 = length(intersect(predictedid2,moduleid2))/length(moduleid2)
precise2 = length(intersect(predictedid2,moduleid2))/length(predictedid2)
F2 = 2*precise2*recall2/(precise2+recall2)
```

multilayernetworkSimulation
Illustration of multi-layer weighted network simulation

Description

Simulate a multi-layer weighted network with each layer sharing the same set of nodes but different nodes scores

Usage

```
multilayernetworkSimulation(n, k, theta, L)
```

Arguments

n	number of nodes in each layer of the network
k	number of nodes in the conserved module
theta	module node score follow the uniform distribution in range [theta,1]
L	number of layers

Value

a list containing all the layers, each as result object of networkSimulation

Author(s)

Dong Li, <dxl466@cs.bham.ac.uk>

See Also

networkSimulation

Examples

```
n = 100
k = 20
theta = 0.5
L = 5
cpl <- multilayernetworkSimulation(n,k,theta,L)
## No proper way to visualize it yet
```

`networkSimulation` *Illustration of weighted network simulation*

Description

Simulate a single weighted network

Usage

```
networkSimulation(n, k, theta)
```

Arguments

`n`	number of nodes in the network
`k`	number of nodes in the module, n < k
`theta`	module node score follow the uniform distribution in range [theta,1]

Value

a list containing network adjacency matrix, node score and module membership

Author(s)

Dong Li, <dxl466@cs.bham.ac.uk>

Examples

```
pp <- networkSimulation(100,20,0.5)
moduleid <- pp[[3]]
netid <- 1:100
restp<- netid[-moduleid]
groupdesign=list(moduleid,restp)
names(groupdesign)=c('module','background')
## Not run: library(qgraph)
pg<-qgraph(pp[[1]],groups=groupdesign,legend=TRUE)
## End(Not run)
```

`twolayernetworkSimulation` *Illustration of two-layer weighted network simulation*

Description

Simulate a two-layer weighted network

Usage

```
twolayernetworkSimulation(n1, k1, theta1, n2, k2, theta2)
```

Arguments

n1	number of nodes in the network1
k1	number of nodes in the module1, n1 < k1
theta1	module1 node score follow the uniform distribution in range [theta1,1]
n2	number of nodes in the network2
k2	number of nodes in the module2, n2 < k2
theta2	module2 node score follow the uniform distribution in range [theta2,1]

Value

a list containing network1, network2 and a inter-layer links matrix

Author(s)

Dong Li, <dxl466@cs.bham.ac.uk>

See Also

networkSimulation

Examples

```
n1=100
k1=20
theta1 = 0.5
n2=80
k2=10
theta2 = 0.5
ppresult <- twolayernetworkSimulation(n1,k1,theta1,n2,k2,theta2)
A <- ppresult[[3]]
pp <- ppresult[[1]]
moduleid <- pp[[3]]
netid <- 1:n1
restp<- netid[-moduleid]
pp2 <- ppresult[[2]]
moduleid2 <- pp2[[3]]
netid2 <- 1:n2
restp2<- netid2[-moduleid2]
## labelling the groups
groupdesign=list(moduleid,restp,(moduleid2+n1),(restp2+n1))
names(groupdesign)=c('module1','background1','module2','background2')
twolayernet<-matrix(0,nrow=(n1+n2),ncol=(n1+n2))
twolayernet[1:n1,1:n1]<-pp[[1]]
twolayernet[(n1+1):(n1+n2),(n1+1):(n1+n2)]<-pp2[[1]]
twolayernet[1:n1,(n1+1):(n1+n2)] = A
twolayernet[(n1+1):(n1+n2),1:n1] = t(A)
## Not run: library(qgraph)
g<-qgraph(twolayernet,groups=groupdesign,legend=TRUE)
## End(Not run)
```

Index

∗ **Euclidean**
EuclideanProjectionENNORM, 5
∗ **identification,**
CGPFixSSMultiLayer, 3
CGPFixSSTwolayer, 4
moduleIdentificationGPFixSSMultilayer, 7
moduleIdentificationGPFixSSTwolayer, 8
∗ **identification**
CGPFixSS, 2
moduleIdentificationGPFixSS, 6
∗ **module**
CGPFixSS, 2
CGPFixSSMultiLayer, 3
CGPFixSSTwolayer, 4
moduleIdentificationGPFixSS, 6
moduleIdentificationGPFixSSMultilayer, 7
moduleIdentificationGPFixSSTwolayer, 8
∗ **multi-layer**
CGPFixSSMultiLayer, 3
moduleIdentificationGPFixSSMultilayer, 7
∗ **projection**
EuclideanProjectionENNORM, 5
∗ **simulation**
multilayernetworkSimulation, 10
networkSimulation, 11
twolayernetworkSimulation, 11
∗ **two-layer**
CGPFixSSTwolayer, 4
moduleIdentificationGPFixSSTwolayer, 8

CGPFixSS, 2
CGPFixSSMultiLayer, 3
CGPFixSSTwolayer, 4

EuclideanProjectionENNORM, *2–4*, 5, *6*, *7*, *9*

moduleIdentificationGPFixSS, *2*, 6
moduleIdentificationGPFixSSMultilayer, *3*, 7, *8*
moduleIdentificationGPFixSSTwolayer, *5*, 8
multilayernetworkSimulation, 10

networkSimulation, *10*, 11, *12*

twolayernetworkSimulation, 11