

# Package: HeteroGGM (via r-universe)

November 4, 2024

**Type** Package

**Title** Gaussian Graphical Model-Based Heterogeneity Analysis

**Version** 1.0.1

**Date** 2023-10-10

**Description** The goal of this package is to user-friendly realizing Gaussian graphical model-based heterogeneity analysis. Recently, several Gaussian graphical model-based heterogeneity analysis techniques have been developed. A common methodological limitation is that the number of subgroups is assumed to be known a priori, which is not realistic. In a very recent study (Ren et al., 2022), a novel approach based on the penalized fusion technique is developed to fully data-dependently determine the number and structure of subgroups in Gaussian graphical model-based heterogeneity analysis. It opens the door for utilizing the Gaussian graphical model technique in more practical settings. Beyond Ren et al. (2022), more estimations and functions are added, so that the package is self-contained and more comprehensive and can provide "more direct" insights to practitioners (with the visualization function). Reference: Ren, M., Zhang S., Zhang Q. and Ma S. (2022). Gaussian Graphical Model-based Heterogeneity Analysis via Penalized Fusion. *Biometrics*, 78 (2), 524-535.

**License** GPL-2

**Encoding** UTF-8

**Imports** igraph, Matrix, MASS, huge

**LazyData** true

**LazyLoad** yes

**RoxygenNote** 7.1.2

**Depends** R (>= 3.4)

**Suggests** knitr, rmarkdown

**VignetteBuilder** knitr, rmarkdown

**NeedsCompilation** no

**Author** Mingyang Ren [aut, cre]  
 (<<https://orcid.org/0000-0002-8061-9940>>), Sanguo Zhang [aut],  
 Qingzhao Zhang [aut], Shuangge Ma [aut]

**Maintainer** Mingyang Ren <renmingyang17@mails.ucas.ac.cn>

**Repository** CRAN

**Date/Publication** 2023-10-11 13:10:02 UTC

## Contents

example.data . . . . .	2
FGGM . . . . .	3
FGGM.refit . . . . .	5
genelambda.obo . . . . .	7
generate.data . . . . .	8
GGMPF . . . . .	9
linked_node_names . . . . .	11
PGGMBC . . . . .	12
plot_network . . . . .	14
Power.law.network . . . . .	15
summary_network . . . . .	16
<b>Index</b>	<b>17</b>

---

example.data	<i>Some example data</i>
--------------	--------------------------

---

## Description

Some example data

## Format

A list including: data: The 600 x 20 matrix, the design matrix. L0: The subgroup to which each sample truly belongs. Mu0: The true mean parameters of 3 subgroups. Theta0: The true precision matrices of 3 subgroups. n\_all: The total sample size. K0: The true number of subgroups 3.

## Source

Simulated data (see examples in the function tuning.lambda.FGGM)

## Examples

data(example.data)

**Description**

The base function of Gaussian graphical model-based heterogeneity analysis via penalized fusion: identifying the order of subgroups and reconstructing the network structure.

**Usage**

```
FGGM(data, K, lambda1 = 0.5, lambda2 = 0.2, lambda3 = 2, a = 3, rho = 1,
      eps = 5e-2, niter = 20, maxiter=10, maxiter.AMA=5, initialization=T,
      initialize, average=F, asymmetric=T, local_appro=T,
      penalty = "MCP", theta.fusion=T)
```

**Arguments**

data	n * p matrix, the design matrix.
K	Int, a selected upper bound of $K_0$ .
lambda1	A float value, the tuning parameter controlling the sparse of the mean parameter.
lambda2	A float value, the tuning parameter controlling the sparse of the precision matrix.
lambda3	A float value, the tuning parameter controlling the number of subgroup.
a	A float value, regularization parameter in MCP, the default setting is 3.
rho	A float value, the penalty parameter in ADMM algorithm of updating precision matrix Theta, the default setting is 1.
eps	A float value, algorithm termination threshold.
niter	Int, maximum number of cycles of the EM algorithm, the default setting is 20.
maxiter	Int, maximum number of cycles of the ADMM algorithm.
maxiter.AMA	Int, maximum number of cycles of the AMA algorithm.
initialization	The logical variable, whether to calculate the initial value, the default setting is T, if initialization = F, the initial value uses initialize.
initialize	A given initial value used if initialization = F.
average	The logical variable, whether to use averaging when integrating parameters that are identified as identical subgroups, the default setting is F, which means the estimated parameters for the subgroup with the largest sample size among the subgroups identified as identical subgroups is used as the final parameter for this subgroup.
asymmetric	The logical variable, symmetry of the precision matrices or not, the default setting is T.
local_appro	The logical variable, whether to use local approximations when updating mean parameters, the default setting is T.
penalty	The type of the penalty, which can be selected from c("MCP", "SCAD", "lasso").
theta.fusion	Whether or not the fusion penalty term contains elements of the precision matrices. The default setting is T.

**Value**

A list including all estimated parameters and the BIC value.

**Author(s)**

Mingyang Ren, Sanguo Zhang, Qingzhao Zhang, Shuangge Ma. Maintainer: Mingyang Ren <ren-mingyang17@mails.ucas.ac.cn>.

**References**

Ren, M., Zhang S., Zhang Q. and Ma S. (2020). Gaussian Graphical Model-based Heterogeneity Analysis via Penalized Fusion. *Biometrics*, Published Online.

**Examples**

```
n <- 200           # The sample size of each subgroup
p <- 20           # The dimension of the precision matrix
K0 <- 3           # The true number of subgroups
N <- rep(n,K0)    # The sample sizes of K0 subgroups
K <- 6            # The given upper bound of K0.

##### The true parameters #####
mue <- 1.5
nonnum <- 4
mu01 <- c(rep(mue,nonnum),rep(-mue,nonnum),rep(0,p-2*nonnum))
mu02 <- c(rep(mue,2*nonnum),rep(0,p-2*nonnum))
mu03 <- c(rep(-mue,2*nonnum),rep(0,p-2*nonnum))

# Power law network
set.seed(2)
A.list <- Power.law.network(p,s=5,I2=c(1),I3=c(2))
Theta01 <- A.list$A1
Theta02 <- A.list$A2
Theta03 <- A.list$A3
sigma01 <- solve(Theta01)
sigma02 <- solve(Theta02)
sigma03 <- solve(Theta03)
Mu0.list <- list(mu01,mu02,mu03)
Sigma0.list <- list(sigma01,sigma02,sigma03)
Theta0.list <- list(Theta01,Theta02,Theta03)

##### Generating simulated data #####
whole.data <- generate.data(N,Mu0.list,Theta0.list,Sigma0.list)

PP = FGGM(whole.data$data, K, lambda1 = 0.22, lambda2 = 0.12, lambda3 = 1.83)
mu_hat=PP$mu; Theta_hat=PP$Xi; L.mat = PP$L.mat0
group = PP$group; prob = PP$prob0; bic = PP$bic; member = PP$member
K0_hat = as.numeric(dim(Theta_hat)[3])
K0_hat
```

FGGM.refit

*Refitting of FGGM***Description**

Refitting when  $K_0$  is identified using FGGM().

**Usage**

```
FGGM.refit(data, K, lambda1 = 0.5, lambda2 = 0.2, lambda3 = 2, a = 3, rho = 1,
           eps = 5e-2, niter = 20, maxiter=10, maxiter.AMA=5,
           initialization=T, initialize, average=F,
           asymmetric=T, local_appro=T, penalty = "MCP", theta.fusion=T)
```

**Arguments**

data	n * p matrix, the design matrix.
K	Int, a selected upper bound of $K_0$ .
lambda1	A float value, the tuning parameter controlling the sparse of the mean parameter.
lambda2	A float value, the tuning parameter controlling the sparse of the precision matrix.
lambda3	A float value, the tuning parameter controlling the number of subgroup.
a	A float value, regularization parameter in MCP, the default setting is 3.
rho	A float value, the penalty parameter in ADMM algorithm of updating precision matrix Theta, the default setting is 1.
eps	A float value, algorithm termination threshold.
niter	Int, maximum number of cycles of the algorithm, the default setting is 20.
maxiter	Int, maximum number of cycles of the ADMM algorithm.
maxiter.AMA	Int, maximum number of cycles of the AMA algorithm.
initialization	The logical variable, whether to calculate the initial value, the default setting is T, if initialization = F, the initial value uses initialize.
initialize	A given initial value used if initialization = F.
average	The logical variable, whether to use averaging when integrating parameters that are identified as identical subgroups, the default setting is F, which means the estimated parameters for the subgroup with the largest sample size among the subgroups identified as identical subgroups is used as the final parameter for this subgroup.
asymmetric	The logical variable, symmetry of the precision matrices or not, the default setting is T.
local_appro	The logical variable, whether to use local approximations when updating mean parameters, the default setting is T.
penalty	The type of the penalty, which can be selected from c("MCP", "SCAD", "lasso").
theta.fusion	Whether or not the fusion penalty term contains elements of the precision matrices. The default setting is T.

**Value**

A list including all estimated parameters and the BIC value after refitting.

**Author(s)**

Mingyang Ren, Sanguo Zhang, Qingzhao Zhang, Shuangge Ma. Maintainer: Mingyang Ren <ren-mingyang17@mails.ucas.ac.cn>.

**References**

Ren, M., Zhang S., Zhang Q. and Ma S. (2020). Gaussian Graphical Model-based Heterogeneity Analysis via Penalized Fusion. *Biometrics*, Published Online.

**Examples**

```
n <- 200           # The sample size of each subgroup
p <- 20           # The dimension of the precision matrix
K0 <- 3           # The true number of subgroups
N <- rep(n,K0)    # The sample sizes of K0 subgroups
K <- 6            # The given upper bound of K0.

##### The true parameters #####
mue <- 1.5
nonnum <- 4
mu01 <- c(rep(mue,nonnum),rep(-mue,nonnum),rep(0,p-2*nonnum))
mu02 <- c(rep(mue,2*nonnum),rep(0,p-2*nonnum))
mu03 <- c(rep(-mue,2*nonnum),rep(0,p-2*nonnum))
# Power law network
set.seed(2)
A.list <- Power.law.network(p,s=5,I2=c(1),I3=c(2))
Theta01 <- A.list$A1
Theta02 <- A.list$A2
Theta03 <- A.list$A3
sigma01 <- solve(Theta01)
sigma02 <- solve(Theta02)
sigma03 <- solve(Theta03)
Mu0.list <- list(mu01,mu02,mu03)
Sigma0.list <- list(sigma01,sigma02,sigma03)
Theta0.list <- list(Theta01,Theta02,Theta03)

##### Generating simulated data #####
whole.data <- generate.data(N,Mu0.list,Theta0.list,Sigma0.list)

PP = FGGM.refit(whole.data$data, K, lambda1 = 0.22, lambda2 = 0.12, lambda3 = 1.83)
mu_hat=PP$mu; Theta_hat=PP$Xi; L.mat = PP$L.mat0
group = PP$group; prob = PP$prob0; bic = PP$bic; member = PP$member
K0_hat = as.numeric(dim(Theta_hat)[3])
K0_hat
```

---

genelambda.obo      *Generate tuning parameters*

---

**Description**

Generating a sequence of the tuning parameters (lambda1, lambda2, and lambda3).

**Usage**

```
genelambda.obo(nlambda1=10,lambda1_max=1,lambda1_min=0.05,  
               nlambda2=10,lambda2_max=1,lambda2_min=0.01,  
               nlambda3=10,lambda3_max=5,lambda3_min=0.5)
```

**Arguments**

nlambda1	The numbers of lambda 1.
lambda1_max	The maximum values of lambda 1.
lambda1_min	The minimum values of lambda 1.
nlambda2	The numbers of lambda 2.
lambda2_max	The maximum values of lambda 2.
lambda2_min	The minimum values of lambda 2.
nlambda3	The numbers of lambda 3.
lambda3_max	The maximum values of lambda 3.
lambda3_min	The minimum values of lambda 3.

**Value**

A sequence of the tuning parameters (lambda1, lambda2, and lambda3).

**Author(s)**

Mingyang Ren

**Examples**

```
lambda <- genelambda.obo(nlambda1=5,lambda1_max=0.5,lambda1_min=0.1, nlambda2=15,lambda2_max=1.5,  
                        lambda2_min=0.1, nlambda3=10,lambda3_max=3.5,lambda3_min=0.5)  
lambda
```

---

 generate.data

*Data Generation*


---

## Description

Data Generation

## Usage

```
generate.data(N,Mu0.list,Theta0.list,Sigma0.list)
```

## Arguments

N	K0 * 1 vector, the sample sizes of subgroups.
Mu0.list	A list including K0 mean vectors (p * 1).
Theta0.list	A list including K0 precision matrices (p * p).
Sigma0.list	A list including K0 correlation matrices (p * p).

## Value

The simulated data and the true parameters.

## Examples

```
n <- 200           # The sample size of each subgroup
p <- 20           # The dimension of the precision matrix
K0 <- 3           # The true number of subgroups
N <- rep(n,K0)    # The sample sizes of K0 subgroups

##### The true parameters #####
mue <- 1.5
nonnum <- 4
mu01 <- c(rep(mue,nonnum),rep(-mue,nonnum),rep(0,p-2*nonnum))
mu02 <- c(rep(mue,2*nonnum),rep(0,p-2*nonnum))
mu03 <- c(rep(-mue,2*nonnum),rep(0,p-2*nonnum))

# Power law network
set.seed(2)
A.list <- Power.law.network(p,s=5,I2=c(1),I3=c(2))
Theta01 <- A.list$A1
Theta02 <- A.list$A2
Theta03 <- A.list$A3
sigma01 <- solve(Theta01)
sigma02 <- solve(Theta02)
sigma03 <- solve(Theta03)
Mu0.list <- list(mu01,mu02,mu03)
Sigma0.list <- list(sigma01,sigma02,sigma03)
Theta0.list <- list(Theta01,Theta02,Theta03)
```



```
##### Generating simulated data #####
whole.data <- generate.data(N,Mu0.list,Theta0.list,Sigma0.list)
```

GGMPF

*GGM-based heterogeneity analysis.***Description**

The main function of Gaussian graphical model-based heterogeneity analysis via penalized fusion.

**Usage**

```
GGMPF(lambda, data, K, initial.selection="K-means", initialize, average=F,
       asymmetric=T, eps = 5e-2, maxiter=10,
       maxiter.AMA=5, local_appro=T, trace = F, penalty = "MCP", theta.fusion=T)
```

**Arguments**

lambda	A list, the sequences of the tuning parameters (lambda1, lambda2, and lambda3).
data	n * p matrix, the design matrix.
K	Int, a selected upper bound of K_0.
initial.selection	The different initial values from two clustering methods, which can be selected from c("K-means","dbscan").
initialize	A given initial values, which should be given when initial.selection is not in c("K-means","dbscan").
average	The logical variable, whether to use averaging when integrating parameters that are identified as identical subgroups, the default setting is F, which means the estimated parameters for the subgroup with the largest sample size among the subgroups identified as identical subgroups is used as the final parameter for this subgroup.
asymmetric	The logical variable, symmetry of the precision matrices or not, the default setting is T.
eps	A float value, algorithm termination threshold.
maxiter	Int, maximum number of cycles of the ADMM algorithm.
maxiter.AMA	Int, maximum number of cycles of the AMA algorithm.
local_appro	The logical variable, whether to use local approximations when updating mean parameters, the default setting is T.
trace	The logical variable, whether or not to output the number of identified subgroups during the search for parameters.
penalty	The type of the penalty, which can be selected from c("MCP", "SCAD", "lasso").
theta.fusion	Whether or not the fusion penalty term contains elements of the precision matrices. The default setting is T.

**Value**

A list including all estimated parameters and the BIC values with all choices of given tuning parameters, and the selected optional parameters.

**Author(s)**

Mingyang Ren, Sanguo Zhang, Qingzhao Zhang, Shuangge Ma. Maintainer: Mingyang Ren <ren-mingyang17@mails.ucas.ac.cn>.

**References**

Ren, M., Zhang S., Zhang Q. and Ma S. (2022). Gaussian Graphical Model-based Heterogeneity Analysis via Penalized Fusion. *Biometrics*.

**Examples**

```
##### Example 1: Generate simulation data and apply this method to analysis #####
n <- 200          # The sample size of each subgroup
p <- 20          # The dimension of the precision matrix
K0 <- 3          # The true number of subgroups
N <- rep(n,K0)   # The sample sizes of K0 subgroups
K <- 6          # The given upper bound of K0.

##### The true parameters #####
mue <- 1.5
nonnum <- 4
mu01 <- c(rep(mue,nonnum),rep(-mue,nonnum),rep(0,p-2*nonnum))
mu02 <- c(rep(mue,2*nonnum),rep(0,p-2*nonnum))
mu03 <- c(rep(-mue,2*nonnum),rep(0,p-2*nonnum))

# Power law network
set.seed(2)
A.list <- Power.law.network(p,s=5,I2=c(1),I3=c(2))
Theta01 <- A.list$A1
Theta02 <- A.list$A2
Theta03 <- A.list$A3
sigma01 <- solve(Theta01)
sigma02 <- solve(Theta02)
sigma03 <- solve(Theta03)
Mu0.list <- list(mu01,mu02,mu03)
Sigma0.list <- list(sigma01,sigma02,sigma03)
Theta0.list <- list(Theta01,Theta02,Theta03)

##### Generating simulated data #####
whole.data <- generate.data(N,Mu0.list,Theta0.list,Sigma0.list)

##### The implementation and evaluation #####
lambda <- genelambda.obo(nlambda1=5,lambda1_max=0.5,lambda1_min=0.1,
                        nlambda2=15,lambda2_max=1.5,lambda2_min=0.1,
                        nlambda3=10,lambda3_max=3.5,lambda3_min=0.5)
res <- GGMPF(lambda, whole.data$data, K, initial.selection="K-means")
Theta_hat.list <- res$Theta_hat.list
```

```

Mu_hat.list <- res$Mu_hat.list
prob.list <- res$prob.list
L.mat.list <- res$L.mat.list
opt_num <- res$Opt_num
opt_Theta_hat <- Theta_hat.list[[opt_num]]
opt_Mu_hat <- Mu_hat.list[[opt_num]]
opt_L.mat <- L.mat.list[[opt_num]]
opt_prob <- prob.list[[opt_num]]
K_hat <- dim(opt_Theta_hat)[3]
K_hat

##### Example 2: Call the built-in simulation data set and analyze #####
data(example.data)
K <- 6
lambda <- genelambda.obo(nlambda1=5,lambda1_max=0.5,lambda1_min=0.1,
                        nlambda2=15,lambda2_max=1.5,lambda2_min=0.1,
                        nlambda3=10,lambda3_max=3.5,lambda3_min=0.5)
res <- GGMPF(lambda, example.data$data, K, initial.selection="K-means")
Theta_hat.list <- res$Theta_hat.list
opt_num <- res$Opt_num
opt_Theta_hat <- Theta_hat.list[[opt_num]]
K_hat <- dim(opt_Theta_hat)[3]
K_hat

```

---

linked_node_names	<i>Indexes the names of all nodes connected to some particular nodes in a subgroup.</i>
-------------------	---

---

### Description

Indexes the names of all nodes connected to some particular nodes in a subgroup.

### Usage

```
linked_node_names(summ, va_names, num_subgroup=1)
```

### Arguments

summ	A list, the summary of the resulting network structures.
va_names	A vector, the names of nodes of interest.
num_subgroup	Int, the subgroup numbering.

### Value

A list including the names of connected nodes to the nodes of interest in a subgroup.

PGGMBC

*Penalized GGM-based clustering.***Description**

The main function of penalized Gaussian graphical model-based clustering with unconstrained covariance matrices.

**Usage**

```
PGGMBC(lambda, data, K, initial.selection="K-means", initialize, average=F,
        asymmetric=T, eps = 5e-2, maxiter=10,
        maxiter.AMA=5, local_appro=T, trace = F, penalty = "MCP")
```

**Arguments**

lambda	A list, the sequences of the tuning parameters (lambda1 and lambda2).
data	n * p matrix, the design matrix.
K	Int, a given number of subgroups.
initial.selection	The different initial values from two clustering methods, which can be selected from c("K-means","dbscan").
initialize	A given initial values, which should be given when initial.selection is not in c("K-means","dbscan").
average	The logical variable, whether to use averaging when integrating parameters that are identified as identical subgroups, the default setting is F, which means the estimated parameters for the subgroup with the largest sample size among the subgroups identified as identical subgroups is used as the final parameter for this subgroup.
asymmetric	The logical variable, symmetry of the precision matrices or not, the default setting is T.
eps	A float value, algorithm termination threshold.
maxiter	Int, maximum number of cycles of the ADMM algorithm.
maxiter.AMA	Int, maximum number of cycles of the AMA algorithm.
local_appro	The logical variable, whether to use local approximations when updating mean parameters, the default setting is T.
trace	The logical variable, whether or not to output the number of identified subgroups during the search for parameters.
penalty	The type of the penalty, which can be selected from c("MCP", "SCAD", "lasso").

**Value**

A list including all estimated parameters and the BIC values with all choices of given tuning parameters, and the selected optional parameters.

**Author(s)**

Mingyang Ren, Sanguo Zhang, Qingzhao Zhang, Shuangge Ma. Maintainer: Mingyang Ren <ren-mingyang17@mails.ucas.ac.cn>.

**References**

Ren, M., Zhang S., Zhang Q. and Ma S. (2020). Gaussian Graphical Model-based Heterogeneity Analysis via Penalized Fusion. *Biometrics*, Published Online, <https://doi.org/10.1111/biom.13426>.  
 Zhou, H., Pan, W., & Shen, X. (2009). Penalized model-based clustering with unconstrained covariance matrices. *Electronic journal of statistics*, 3, 1473.

**Examples**

```
##### Example 1: Generate simulation data and apply this method to analysis #####
n <- 200          # The sample size of each subgroup
p <- 20          # The dimension of the precision matrix
K <- 3           # The true number of subgroups
N <- rep(n,K)    # The sample sizes of K subgroups

##### The true parameters #####
mue <- 1.5
nonnum <- 4
mu01 <- c(rep(mue,nonnum),rep(-mue,nonnum),rep(0,p-2*nonnum))
mu02 <- c(rep(mue,2*nonnum),rep(0,p-2*nonnum))
mu03 <- c(rep(-mue,2*nonnum),rep(0,p-2*nonnum))

# Power law network
set.seed(2)
A.list <- Power.law.network(p,s=5,I2=c(1),I3=c(2))
Theta01 <- A.list$A1
Theta02 <- A.list$A2
Theta03 <- A.list$A3
sigma01 <- solve(Theta01)
sigma02 <- solve(Theta02)
sigma03 <- solve(Theta03)
Mu0.list <- list(mu01,mu02,mu03)
Sigma0.list <- list(sigma01,sigma02,sigma03)
Theta0.list <- list(Theta01,Theta02,Theta03)

##### Generating simulated data #####
whole.data <- generate.data(N,Mu0.list,Theta0.list,Sigma0.list)

##### The implementation and evaluation of competitors #####
lambda <- genelambda.obo(nlambda1=5,lambda1_max=0.5,lambda1_min=0.1,
                        nlambda2=15,lambda2_max=1.5,lambda2_min=0.1)
res <- PGGMBC(lambda, whole.data$data, K, initial.selection="K-means")
Theta_hat.list <- res$Theta_hat.list
Mu_hat.list <- res$Mu_hat.list
prob.list <- res$prob.list
L.mat.list <- res$L.mat.list
opt_num <- res$Opt_num
opt_Theta_hat <- Theta_hat.list[[opt_num]]
```

```

opt_Mu_hat <- Mu_hat.list[[opt_num]]
opt_L.mat <- L.mat.list[[opt_num]]
opt_prob <- prob.list[[opt_num]]

##### Example 2: Call the built-in simulation data set and analyze #####
data(example.data)
K <- 3
lambda <- genelambda.obo(nlambda1=5,lambda1_max=0.5,lambda1_min=0.1,
                        nlambda2=15,lambda2_max=1.5,lambda2_min=0.1)
res <- PGGMBC(lambda, example.data$data, K, initial.selection="K-means")
Theta_hat.list <- res$Theta_hat.list
opt_num <- res$Opt_num
opt_Theta_hat <- Theta_hat.list[[opt_num]]

```

---

plot\_network

*Visualization of network structures.*


---

### Description

Visualization of network structures.

### Usage

```

plot_network(summ, num_subgroup = 1, plot.mfrow,
             vertex.size=2,vertex.label.cex=0.7,
             vertex.label.dist=0.75, edge.width = 0.1, l=0)

```

### Arguments

summ	A list, the summary of the resulting network structures.
num_subgroup	Int/vector, the subgroup numbering.
plot.mfrow	Figure Layout.
vertex.size	The vertex size.
vertex.label.cex	The vertex label size.
vertex.label.dist	The distance of vertex labels.
edge.width	The edge width.
l	Node Coordinates.

### Value

Visualization of network structure

---

Power.law.network      *Power law network*

---

## Description

Generating three s-block power-law precision matrices

## Usage

```
Power.law.network(p, s = 10, umin = 0.1, umax = 0.4, I2 = 0, I3 = 0)
```

## Arguments

p	The dimensions of the precision matrix.
s	The number of sub-networks.
umin	The lower bound of non-zero elements on non-diagonal elements.
umax	The upper bound of non-zero elements on non-diagonal elements.
I2	The replacement blocks for the precision matrix of the second subgroup.
I3	The replacement blocks for the precision matrix of the third subgroup.

## Value

A list including The precision matrices of three subgroups.

## Examples

```
p <- 20                                    # The dimension of the precision matrix
##### The true parameters #####
# Power law network
set.seed(2)
A.list <- Power.law.network(p,s=5,I2=c(1),I3=c(2))
Theta01 <- A.list$A1
Theta02 <- A.list$A2
Theta03 <- A.list$A3
sigma01 <- solve(Theta01)
sigma02 <- solve(Theta02)
sigma03 <- solve(Theta03)
Sigma0.list <- list(sigma01,sigma02,sigma03)
Theta0.list <- list(Theta01,Theta02,Theta03)
```

---

summary_network	<i>The summary of the resulting network structures.</i>
-----------------	---

---

### Description

Summarize the characteristics of the resulting network structures.

### Usage

```
summary_network(opt_Mu_hat, opt_Theta_hat, data)
```

### Arguments

opt\_Mu\_hat      A  $p \times p$   $K_0$ \_hat matrix, the optional mean vectors of  $K_0$ \_hat subgroups.  
 opt\_Theta\_hat     $n \times p \times p$   $K_0$ \_hat matrix, the optional precision matrices of  $K_0$ \_hat subgroups.  
 data             A  $n \times p$  matrix, the design matrix.

### Value

A list including the overlap of edges of different subgroups, the number of edges, and the names of connected nodes to each nodes in each subgroup.

### Examples

```
data(example.data)
K <- 6
lambda <- genelambda.obo(nlambda1=5,lambda1_max=0.5,lambda1_min=0.1,
                          nlambda2=15,lambda2_max=1.5,lambda2_min=0.1,
                          nlambda3=10,lambda3_max=3.5,lambda3_min=0.5)

res <- GGMPF(lambda, example.data$data, K, initial.selection="K-means")
Theta_hat.list <- res$Theta_hat.list
Mu_hat.list <- res$Mu_hat.list
opt_num <- res$Opt_num
opt_Theta_hat <- Theta_hat.list[[opt_num]]
opt_Mu_hat <- Mu_hat.list[[opt_num]]
K_hat <- dim(opt_Theta_hat)[3]
K_hat

summ <- summary_network(opt_Mu_hat, opt_Theta_hat, example.data$data)
summ$Theta_summary$overlap
va_names <- c("1","6")
linked_node_names(summ, va_names, num_subgroup=1)
plot_network(summ, num_subgroup = c(1:K_hat), plot.mfrow = c(1,K_hat))
```



# Index

`example.data`, [2](#)

`FGGM`, [3](#)

`FGGM.refit`, [5](#)

`genelambda.obo`, [7](#)

`generate.data`, [8](#)

`GGMPF`, [9](#)

`linked_node_names`, [11](#)

`PGGBC`, [12](#)

`plot_network`, [14](#)

`Power.law.network`, [15](#)

`summary_network`, [16](#)