

# Package: hglasso (via r-universe)

September 2, 2024

**Type** Package

**Title** Learning Graphical Models with Hubs

**Version** 1.3

**Date** 2022-05-13

**Author** Kean Ming Tan

**Maintainer** Kean Ming Tan <keanming@umich.edu>

**Depends** glasso, mvtnorm, igraph

**Imports** fields

**Description** Implements the hub graphical lasso and hub covariance graph proposal by Tan, KM., London, P., Mohan, K., Lee, S-I., Fazel, M., and Witten, D. (2014). Learning graphical models with hubs. *Journal of Machine Learning Research* 15(Oct):3297-3331.

**License** GPL (>= 2)

**NeedsCompilation** no

**Date/Publication** 2022-05-13 08:20:02 UTC

**Repository** <https://keanmingtan.r-universe.dev>

**RemoteUrl** <https://github.com/cran/hglasso>

**RemoteRef** HEAD

**RemoteSha** da7e8d5f000b176dffff2d68d5a0118ea78e1d24

## Contents

hglasso-package . . . . .	2
binaryMCMC . . . . .	4
hbn . . . . .	5
hcov . . . . .	7
hglasso . . . . .	10
hglassoBIC . . . . .	12
HubNetwork . . . . .	14
image.hglasso . . . . .	15

plot.hglasso . . . . .	16
summary.hglasso . . . . .	17

<b>Index</b>	<b>19</b>
--------------	-----------

---

hglasso-package	<i>Fit the hub graphical lasso, hub covariance graph, and hub binary network</i>
-----------------	--

---

## Description

This package is called hglasso, for "hub graphical lasso". It implements three methods: hub graphical lasso, hub covariance graph, and hub binary network. All are described in the paper "Learning graphical models with hubs", by Tan et al. (2014).

The main functions are as follows: (1) hglasso (2) hcov (3) hbn

The first function, hglasso, performs hub graphical lasso. The second function, hcov, performs hub covariance graph estimation. The third function, hbn, performs hub binary network estimation.

## Details

Package:	hglasso
Type:	Package
Version:	1.2
Date:	2014-08-09
License:	GPL (>=2.0)
LazyLoad:	yes

The package includes the following functions:

<a href="#">hglasso</a> :	Performs hub graphical lasso
<a href="#">hcov</a> :	Performs hub covariance graph estimation
<a href="#">hbn</a> :	Performs hub binary network estimation
<a href="#">HubNetwork</a> :	Generates inverse covariance matrix or covariance matrix with hubs
<a href="#">binaryMCMC</a> :	Generates samples for binary Ising model via Gibbs sampling
<a href="#">image.hglasso</a> :	Creates image plot of the matrix V and Z
<a href="#">plot.hglasso</a> :	Creates a graphical representation of the estimated matrix Theta
<a href="#">summary.hglasso</a> :	Provides summary for the matrix Theta, Z, and V
<a href="#">hglassoBIC</a> :	Calculate BIC-type criterion for <a href="#">hglasso</a>

## Author(s)

Kean Ming Tan and Karthik Mohan

Karthik Mohan implemented the Barzilai-Borwein method for [hbn](#)

Maintainer: Kean Ming Tan <keanming@uw.edu>

## References

Tan, KM., London, P., Mohan, K., Lee, S-I., Fazel, M., and Witten, D. (2014). Learning graphical models with hubs. *Journal of Machine Learning Research* 15(Oct):3297-3331.

## See Also

[hglasso hcov hbn](#)

## Examples

```
#####
# Example from Figure 1 in the manuscript
# A toy example to illustrate the results from
# Hub Graphical Lasso
#####
#library(mvtnorm)
#set.seed(1)
#n=100
#p=100

# A network with 4 hubs
#network<-HubNetwork(p,0.99,4,0.1)
#Theta <- network$Theta
#truehub <- network$hubcol
# The four hub nodes have indices 14, 42, 45, 78
#print(truehub)

# Generate data matrix x
#x <- rmvnorm(n,rep(0,p),solve(Theta))
#x <- scale(x)

# Run Hub Graphical Lasso to estimate the inverse covariance matrix
# res1<-hglasso(cov(x),0.3,0.3,1.5)

# print out a summary of the object hglasso
#summary(res1)
# we see that the estimated hub nodes have indices 14, 42, 45, 78
# We successfully recover the 4 hub nodes

# Plot the matrices V and Z
#image(res1)
#dev.off()
# Plot a graphical representation of the estimated inverse
# covariance matrix --- conditional independence graph
#plot(res1,main="Conditional Independence Graph")
```

---

binaryMCMC	<i>Generate samples using Gibbs sampling for binary network specified by the parameter Theta</i>
------------	--

---

### Description

Sampling from the binary Ising model using Gibbs sampling. This function is not efficient and is only intended to be used in the examples.

### Usage

```
binaryMCMC(n, Theta, burnin, skip, trace=FALSE)
```

### Arguments

n	The number of samples.
Theta	A symmetric parameter matrix for the model from which the data is being generated.
burnin	The number of samples to discard as burn in.
skip	The number of samples to discard in-between returned samples.
trace	Default value of trace=FALSE. If trace=TRUE, the progress of Gibbs sampling is printed when each observation is sampled.

### Value

X	An n x p matrix of samples generated from the binary network specified by Theta.
---	--

### Author(s)

Kean Ming Tan

### References

Tan et al. (2014). Learning graphical models with hubs. To appear in Journal of Machine Learning Research. [arXiv.org/pdf/1402.7349.pdf](https://arxiv.org/pdf/1402.7349.pdf).

### See Also

[HubNetwork](#)

## Examples

```
# generate Theta that specified the structure of a binary Ising model with p=10 variables and 2 hubs
#p<-10
#n<-50
#Theta <- HubNetwork(p,0.95,2,0.3,type="binary")$Theta

# generate samples using Gibbs sampling
#X <- binaryMCMC(n,Theta,burnin=1000,skip=500)
```

---

hbn

*Hub binary network*


---

## Description

Estimates a binary network with hub nodes using a Lasso penalty and a sparse group Lasso penalty. The estimated Theta matrix can be decomposed as  $\Theta = Z + V + t(V)$ , where  $Z$  is a sparse matrix and  $V$  is a matrix that contains hub nodes. The details are given in Tan et al. (2014).

## Usage

```
hbn(X, lambda1, lambda2=100000, lambda3=100000, convergence = 1e-8
, maxiter = 1000, start = "cold", var.init = NULL, trace=FALSE)
```

## Arguments

X	An n by p data matrix. Cannot contain missing values.
lambda1	Non-negative regularization parameter for lasso on the matrix Z. lambda=0 means no regularization.
lambda2	Non-negative regularization parameter for lasso on the matrix V. lambda2=0 means no regularization. The default value is lambda2=100000, encouraging V to be a zero matrix.
lambda3	Non-negative regularization parameter for group lasso on the matrix V. lambda3=0 means no regularization. The default value is lambda3=100000, encouraging V to be a zero matrix.
convergence	Threshold for convergence. Default value is 1e-8.
maxiter	Maximum number of iterations of ADMM algorithm. Default is 1000 iterations.
start	Type of start. cold start is the default. Using warm start, one can provide starting values for the parameters using object from hbn.
var.init	Object from hbn that provides starting values for all the parameters when start="warm" is specified.
trace	Default value of trace=FALSE. If trace=TRUE, every 10 iterations of the ADMM algorithm is printed.

## Details

This implements hub binary network using ADMM algorithm (see Algorithm 1 and Section 5) in Tan et al. (2014). The estimated Theta matrix can be decomposed into  $Z + V + t(V)$ :  $Z$  is a sparse matrix and  $V$  is a matrix that contains dense columns, each column corresponding to a hub node.

The default value of  $\lambda_2=100000$  and  $\lambda_3=100000$  will yield the sparse binary network model estimate as in Hofling and Tibshirani (2009) 'Estimation of sparse binary pairwise Markov networks using pseudo-likelihoods'.

The tuning parameters  $\lambda_1$  determines the sparsity of the matrix  $Z$ ,  $\lambda_2$  determines the sparsity of the selected hub nodes, and  $\lambda_3$  determines the selection of hub nodes.

Within each iteration of the ADMM algorithm, we need to perform an iterative procedure to obtain an update for the matrix Theta since there is no closed form solution for Theta. The Barzilai-Borwein method is used for this purpose (Barzilai and Borwein, 1988). For details, see Algorithm 2 in Appendix F in Tan et al. (2014).

Note: we recommend using this function for moderate size network. For instance, network with 50-100 variables.

## Value

an object of class hbn.

Among some internal variables, this object includes the elements

Theta	Theta is the estimated inverse covariance matrix. Note that $\text{Theta} = Z + V + t(V)$ .
V	V is the estimated matrix that contains hub nodes used to compute Theta.
Z	Z is the estimated sparse matrix used to compute Theta.
hubind	Indices for features that are estimated to be hub nodes

## Author(s)

Kean Ming Tan and Karthik Mohan

## References

Tan et al. (2014). Learning graphical models with hubs. To appear in Journal of Machine Learning Research. [arXiv.org/pdf/1402.7349.pdf](https://arxiv.org/pdf/1402.7349.pdf).

Hofling, H. and Tibshirani, R. (2009). Estimation of sparse binary pairwise Markov networks using pseudo-likelihoods. Journal of Machine Learning Research, 10:883-906.

Barzilai, J. and Borwein, J. (1988). Two-point step size gradient methods. IMA Journal of Numerical Analysis, 8:141-148.

## See Also

[image.hglasso](#) [plot.hglasso](#) [summary.hglasso](#) [binaryMCMC](#)

## Examples

```
#####
# An implementation of Hub Binary Network
#####
#set.seed(1000)
#n=50
#p=5

# A network with 2 hubs
#network<-HubNetwork(p,0.95,2,0.1,type="binary")
#Theta <- network$Theta
#truehub <- network$hubcol
# The four hub nodes have indices 4,5
#print(truehub)

# Generate data matrix x
#X <- binaryMCMC(n,Theta,burnin=500,skip=100)

# Run Hub Binary Network to estimate Theta
#res1 <- hbn(X,2,1,3,trace=TRUE)

# print out a summary of the object hbn
#summary(res1)

# We see that the estimated hub nodes have indices 1,5
# We successfully recover the hub nodes

# Plot the resulting network
# plot(res1)
```

---

hcov

*Hub covariance graph*


---

## Description

Estimates a sparse covariance matrix with hub nodes using a Lasso penalty and a sparse group Lasso penalty. The estimated covariance matrix  $\Sigma$  can be decomposed as  $\Sigma = Z + V + t(V)$ . The details are given in Section 4 in Tan et al. (2014).

## Usage

```
hcov(S, lambda1, lambda2=100000, lambda3=100000, convergence = 1e-10,
maxiter = 1000, start = "cold", var.init = NULL,trace=FALSE)
```

## Arguments

S	A p by p correlation/covariance matrix. Cannot contain missing values.
lambda1	Non-negative regularization parameter for lasso on the matrix Z. lambda=0 means no regularization.

<code>lambda2</code>	Non-negative regularization parameter for lasso on the matrix $V$ . <code>lambda2=0</code> means no regularization. The default value is <code>lambda2=100000</code> , encouraging $V$ to be a zero matrix.
<code>lambda3</code>	Non-negative regularization parameter for group lasso on the matrix $V$ . <code>lambda3=0</code> means no regularization. The default value is <code>lambda3=100000</code> , encouraging $V$ to be a zero matrix.
<code>convergence</code>	Threshold for convergence. Default value is <code>1e-10</code> .
<code>maxiter</code>	Maximum number of iterations of ADMM algorithm. Default is 1000 iterations.
<code>start</code>	Type of start. <code>cold</code> start is the default. Using <code>warm</code> start, one can provide starting values for the parameters using object from <code>hcov</code> .
<code>var.init</code>	Object from <code>hcov</code> that provides starting values for all the parameters when <code>start="warm"</code> is specified.
<code>trace</code>	Default value of <code>trace=FALSE</code> . If <code>trace=TRUE</code> , every 10 iterations of the ADMM algorithm is printed.

### Details

This implements hub covariance graph estimation procedure using ADMM algorithm described in Section 4 in Tan et al. (2014), which estimates a sparse covariance matrix with hub nodes. The estimated covariance matrix can be decomposed into  $Z + V + t(V)$ :  $Z$  is a sparse matrix and  $V$  is a matrix that contains dense columns, each column corresponding to a hub node. For the positive definite constraint  $\Sigma \geq \epsilon I$  that appears in the optimization problem, we choose `epsilon` to be 0.001 by default.

The default value of `lambda2=100000` and `lambda3=100000` will yield the estimator proposed by Xue et al. (2012).

Note that tuning parameters `lambda1` determines the sparsity of the matrix  $Z$ , `lambda2` determines the sparsity of the selected hub nodes, and `lambda3` determines the selection of hub nodes.

### Value

an object of class `hcov`.

Among some internal variables, this object includes the elements

<code>Sigma</code>	<code>Sigma</code> is the estimated covariance matrix. Note that $\Sigma = Z + V + t(V)$ .
<code>V</code>	<code>V</code> is the estimated matrix that contain hub nodes used to compute <code>Sigma</code> .
<code>Z</code>	<code>Z</code> is the estimated sparse matrix used to compute <code>Sigma</code> .
<code>objective</code>	<code>Objective</code> is the minimized objective value of the loss-function considered in Section 4 of Tan et al. (2014).
<code>iteration</code>	The number of iterations of the ADMM algorithm until convergence.
<code>hubind</code>	Indices for features that are estimated to be hub nodes

### Author(s)

Kean Ming Tan



## References

Tan et al. (2014). Learning graphical models with hubs. To appear in Journal of Machine Learning Research. [arXiv.org/pdf/1402.7349.pdf](https://arxiv.org/pdf/1402.7349.pdf).

Xue et al. (2012). Positive-definite l1-penalized estimation of large covariance matrices. Journal of the American Statistical Association, 107:1480-1491.

## See Also

[image.hcov](#) [plot.hcov](#) [summary.hcov](#)

## Examples

```
#####
# Example for estimating covariance matrix
# with hubs
#####
library(mvtnorm)
set.seed(1)
n=100
p=100

# a covariance with 4 hubs

network <- HubNetwork(p,0.95,4,0.1,type="covariance")
Sigma <- network$Theta
hubind <- network$hsubcol
x <- rmvnorm(n,rep(0,p),Sigma)
x <- scale(x)

# Estimate the covariance matrix
res1<-hcov(cov(x),0.3,0.2,1.2)
summary(res1)
# correctly identified two of the hub nodes

# Plot the matrices V and Z
image(res1)
dev.off()
# Plot a graphical representation of the estimated covariance matrix --- covariance graph
plot(res1)

# Xue et al cannot identified any hub nodes
res2 <- hcov(cov(x),0.3)
summary(res2)
plot(res2)
```

hglasso

*Hub graphical lasso***Description**

Estimates a sparse inverse covariance matrix with hub nodes using a Lasso penalty and a sparse group Lasso penalty. The estimated inverse covariance matrix  $\Theta$  can be decomposed as  $\Theta = Z + V + t(V)$ , where  $Z$  is a sparse matrix and  $V$  is a matrix that contains hub nodes. The details are given in Tan et al. (2014).

**Usage**

```
hglasso(S, lambda1, lambda2=100000, lambda3=100000, convergence = 1e-10
, maxiter = 1000, start = "cold", var.init = NULL, trace=FALSE)
```

**Arguments**

<code>S</code>	A $p$ by $p$ correlation/covariance matrix. Cannot contain missing values.
<code>lambda1</code>	Non-negative regularization parameter for lasso on the matrix $Z$ . <code>lambda=0</code> means no regularization.
<code>lambda2</code>	Non-negative regularization parameter for lasso on the matrix $V$ . <code>lambda2=0</code> means no regularization. The default value is <code>lambda2=100000</code> , encouraging $V$ to be a zero matrix.
<code>lambda3</code>	Non-negative regularization parameter for group lasso on the matrix $V$ . <code>lambda3=0</code> means no regularization. The default value is <code>lambda3=100000</code> , encouraging $V$ to be a zero matrix.
<code>convergence</code>	Threshold for convergence. Default value is <code>1e-10</code> .
<code>maxiter</code>	Maximum number of iterations of ADMM algorithm. Default is 1000 iterations.
<code>start</code>	Type of start. <code>cold</code> start is the default. Using <code>warm</code> start, one can provide starting values for the parameters using object from <code>hglasso</code> .
<code>var.init</code>	Object from <code>hglasso</code> that provides starting values for all the parameters when <code>start="warm"</code> is specified.
<code>trace</code>	Default value of <code>trace=FALSE</code> . If <code>trace=TRUE</code> , every 10 iterations of the ADMM algorithm is printed.

**Details**

This implements hub graphical lasso using ADMM Algorithm (see Algorithm 1) described in Tan et al. (2014), which estimates a sparse inverse covariance matrix with hub nodes. The estimated inverse covariance matrix can be decomposed into  $Z + V + t(V)$ :  $Z$  is a sparse matrix and  $V$  is a matrix that contains dense columns, each column corresponding to a hub node.

The default value of `lambda2=100000` and `lambda3=100000` will yield the graphical lasso estimate as in Friedman et al. (2007) 'Sparse inverse covariance estimation with lasso'.

Note that tuning parameters  $\lambda_1$  determines the sparsity of the matrix  $Z$ ,  $\lambda_2$  determines the sparsity of the selected hub nodes, and  $\lambda_3$  determines the selection of hub nodes.

This algorithm uses a block diagonal screening rule to speed up computations considerably. Details are given in Theorem 1 in Tan et al. (2014) 'Learning graphical models with hubs'. The idea is as follow: we first check whether the solution to the hglasso problem will be block diagonal, for a given set of tuning parameters, using Theorem 1. If so, then one can simply apply hglasso to each block separately, leading to massive speed improvements. Similar idea has been used to obtain a sparse inverse covariance matrix as in Friedman et al. (2007) in the glasso package.

### Value

an object of class hglasso.

Among some internal variables, this object include the elements

Theta	Theta is the estimated inverse covariance matrix. Note that $\text{Theta} = Z + V + t(V)$ .
V	V is the estimated matrix that contains hub nodes used to compute Theta.
Z	Z is the estimated sparse matrix used to compute Theta.
objective	Objective is the minimized objective value of the loss-function considered in Section 3 of Tan et al. (2014).
hubind	Indices for features that are estimated to be hub nodes

### Author(s)

Kean Ming Tan

### References

Tan et al. (2014). Learning graphical models with hubs. To appear in Journal of Machine Learning Research. [arXiv.org/pdf/1402.7349.pdf](https://arxiv.org/pdf/1402.7349.pdf).

Friedman et al. (2007). Sparse inverse covariance estimation with the lasso. *Biostatistics*, 9(3):432-441.

Witten et al. (2011). New insights and faster computations for the graphical lasso. *Journal of Computational and Graphical Statistics*, 20(4):892-900.

### See Also

[image.hglasso](#) [plot.hglasso](#) [summary.hglasso](#) [hglassoBIC](#)

### Examples

```
#####
# Example from Figure 1 in the manuscript
# A toy example to illustrate the results from
# Hub Graphical Lasso
#####
library(mvtnorm)
library(glasso)
```

```

set.seed(1)
n=100
p=100

# A network with 4 hubs
network<-HubNetwork(p,0.99,4,0.1)
Theta <- network$Theta
truehub <- network$hubcol
# The four hub nodes have indices 14, 42, 45, 78
print(truehub)

# Generate data matrix x
x <- rmvnorm(n,rep(0,p),solve(Theta))
x <- scale(x)

# Run Hub Graphical Lasso to estimate the inverse covariance matrix
res1 <- hglasso(cov(x),0.3,0.3,1.5)

# print out a summary of the object hglasso
summary(res1)
# we see that the estimated hub nodes have indices 14, 42, 45, 78
# We successfully recover the 4 hub nodes

# Run hglasso using with and without warm start.
# system.time(hglasso(cov(x),0.31,0.3,1.5))

# system.time(hglasso(cov(x),0.31,0.3,1.5,start="warm",var.init=res1))

# Run hglasso with larger lambda2, encouraging the hub nodes to be more sparse
res2 <- hglasso(cov(x),0.3,0.35,1.5)

# Run hglasso with lambda2=lambda3=100000, the solution is the
# same as the graphical lasso solution obtain from glasso package
res3 <- hglasso(cov(x),0.3)
res4 <- glasso(cov(x),0.3,penalize.diagonal=FALSE)
# print the frobenius norm of the difference between the two estimates
print(sum((res3$Theta-res4$wi)^2))

```

---

hglassoBIC

*BIC-type criterion for hglasso*


---

### Description

This function calculates the BIC-type criterion for tuning parameter selection for [hglasso](#) proposed in Section 3.4 in Tan et al. (2014)

### Usage

```
hglassoBIC(x, S, c=0.2)
```

**Arguments**

x	An object of class <a href="#">hglasso</a> .
S	A p by p correlation/covariance matrix. Cannot contain missing values.
c	A constant between 0 and 1. When c is small, the BIC-type criterion will favor more hub nodes. The default value is c=0.2.

**Value**

BIC	The calculated BIC-type criterion in Section 3.4 in Tan et al. (2014).
-----	--

**Author(s)**

Kean Ming Tan

**References**

Tan et al. (2014). Learning graphical models with hubs. To appear in Journal of Machine Learning Research. [arXiv.org/pdf/1402.7349.pdf](https://arxiv.org/pdf/1402.7349.pdf).

**See Also**

[hglasso](#)

**Examples**

```
#library(mvtnorm)
#library(glasso)
#set.seed(1)
#n=100
#p=100

# A network with 4 hubs
#network<-HubNetwork(p,0.99,4,0.1)
#Theta <- network$Theta
#truehub <- network$hubcol
# The four hub nodes have indices 14, 42, 45, 78
#print(truehub)

# Generate data matrix x
#x <- rmvnorm(n,rep(0,p),solve(Theta))
#x <- scale(x)
#S <- cov(x)
# Run Hub Graphical Lasso with different tuning parameters
#lambdas2 <- seq(0,0.5,by=0.05)
#BICcriterion <- NULL
#for(lambda2 in lambdas2){
#res1 <- hglasso(S,0.3,lambda2,1.5)
#BICcriterion <- c(BICcriterion,hglassoBIC(res1,S)$BIC)
#}
#lambda2 <- lambdas2[which(BICcriterion==min(BICcriterion))]
```

---

HubNetwork	<i>Hub network generation</i>
------------	-------------------------------

---

**Description**

Generate an inverse covariance matrix, covariance matrix, or binary network with hub structure

**Usage**

```
HubNetwork(p, sparsity, hubnumber, hubsparsity, type = "Gaussian")
```

**Arguments**

<code>p</code>	The number of features
<code>sparsity</code>	Sparsity of the network
<code>hubnumber</code>	The number of hubs in the network
<code>hubsparsity</code>	Sparsity level within each hub
<code>type</code>	Type of network. The default value <code>type="Gaussian"</code> generates an inverse covariance matrix. <code>type="covariance"</code> generates a covariance matrix with hubs. <code>type="binary"</code> generates a binary network with hubs.

**Value**

<code>Theta</code>	Theta is the generated inverse covariance matrix, covariance matrix, or binary network.
<code>hubcol</code>	hubcol contains indices for features that are hubs.

**Author(s)**

Kean Ming Tan

**References**

Tan et al. (2014). Learning graphical models with hubs. To appear in Journal of Machine Learning Research. [arXiv.org/pdf/1402.7349.pdf](https://arxiv.org/pdf/1402.7349.pdf).

**Examples**

```
# Generate inverse covariance matrix with 5 hubs
# 30% of the elements within a hub are zero
# 95% of the elements that are not within hub nodes are zero
p <- 100
Theta <- HubNetwork(p,0.95,5,0.3)$Theta

# Generate covariance matrix with 5 hubs with similar structure
Sigma <- HubNetwork(p,0.95,5,0.3,type="covariance")$Theta
```

```
# Generate binary network with 2 hubs with p=10
Theta <- HubNetwork(p=10,0.95,2,0.3,type="binary")$Theta
```

---

```
image.hglasso      Image plot of an object of class hglasso, hcov, or hbn
```

---

## Description

This function plots a hglasso or hcov — the estimated matrix  $V$  and  $Z$  from [hglasso](#), [hcov](#), or [hbn](#)

## Usage

```
## S3 method for class 'hglasso'
image(x, ...)
```

## Arguments

```
x          an object of class hglasso, hcov, or hbn.
...       additional parameters to be passed to image.
```

## Details

The estimated inverse covariance matrix from [hglasso](#), covariance matrix from [hcov](#), and estimated binary network [hbn](#) can be decomposed as  $Z + V + t(V)$ , where  $V$  is a matrix that contains hub nodes. This function creates image plots of  $Z$  and  $V$ .

## Author(s)

Kean Ming Tan

## References

Tan et al. (2014). Learning graphical models with hubs. To appear in Journal of Machine Learning Research. [arXiv.org/pdf/1402.7349.pdf](https://arxiv.org/pdf/1402.7349.pdf).

## See Also

[plot.hglasso](#) [summary.hglasso](#) [hglasso](#) [hcov](#) [hbn](#)

## Examples

```
#####
# Example from Figure 1 in the manuscript
# A toy example to illustrate the results from
# Hub Graphical Lasso
#####
library(mvtnorm)
set.seed(1)
n=100
```

```

p=100

# A network with 4 hubs
Theta<-HubNetwork(p,0.99,4,0.1)$Theta

# Generate data matrix x
x <- rmvnorm(n,rep(0,p),solve(Theta))
x <- scale(x)

# Run Hub Graphical Lasso to estimate the inverse covariance matrix
res1 <- hglasso(cov(x),0.3,0.2,2)

# image plots for the matrix V and Z
image(res1)
dev.off()

```

---

plot.hglasso

*Plot an object of class hglasso, hcov, or hbn*


---

### Description

This function plots an object `hglasso` or `hcov` — graphical representation of the estimated inverse covariance matrix from [hglasso](#), covariance matrix from [hcov](#), or binary network from [hbn](#)

### Usage

```

## S3 method for class 'hglasso'
plot(x, layout=NULL,...)

```

### Arguments

`x` an object of class [hglasso](#), [hcov](#), or [hbn](#).

`layout` the layout of the graph to use. If not specified, [layout.kamada.kawai](#) is used.

`...` additional parameters to be passed to [plot.igraph](#).

### Details

This function plots a graphical representation of the estimated inverse covariance matrix or covariance matrix. The hubs are colored in red and has a large vertex size. Features indices for hubs are shown.

### Author(s)

Kean Ming Tan



## References

Tan et al. (2014). Learning graphical models with hubs. To appear in Journal of Machine Learning Research. [arXiv.org/pdf/1402.7349.pdf](https://arxiv.org/pdf/1402.7349.pdf).

## See Also

[image.hglasso](#) [summary.hglasso](#) [hglasso](#) [hcov](#) [hbn](#)

## Examples

```
#####
# Example from Figure 1 in the manuscript
# A toy example to illustrate the results from
# Hub Graphical Lasso
#####
library(mvtnorm)
set.seed(1)
n=100
p=100

# A network with 4 hubs
Theta<-HubNetwork(p,0.99,4,0.1)$Theta

# Generate data matrix x
x <- rmvnorm(n,rep(0,p),solve(Theta))
x <- scale(x)

# Run Hub Graphical Lasso to estimate the inverse covariance matrix
res1 <- hglasso(cov(x),0.3,0.3,1.5)

# Graphical representation of the estimated Theta
plot(res1,main="conditional independence graph")
```

---

summary.hglasso      *Plot an object of class hglasso, hcov, or hbn*

---

## Description

This function provides some information for an object [hglasso](#), [hcov](#), or [hbn](#).

## Usage

```
## S3 method for class 'hglasso'
summary(object, ...)
```

## Arguments

object            an object of class [hglasso](#), [hcov](#), or [hbn](#).  
 ...                any other arguments passed to print.

**Details**

Some information for an object hglasso, hcov, or hbn: (1) The number of observations  $n$  and the number of features  $p$ . (2) The number of edges in  $\Theta$ ,  $V$ , and  $Z$ . (3) The indices for hub nodes, and also the number of edges within each hub node.

**Author(s)**

Kean Ming Tan

**References**

Tan et al. (2014). Learning graphical models with hubs. *Journal of Machine Learning Research* 15(Oct):3297-3331.

**See Also**

[image.hglasso](#) [plot.hglasso](#) [hglasso](#) [hcov](#) [hbn](#)

**Examples**

```
# See example in hglasso, hcov, or hbn.
```

# Index

## \* package

hglasso-package, 2

binaryMCMC, 2, 4, 6

hbn, 2, 3, 5, 15–18

hcov, 2, 3, 7, 15–18

hglasso, 2, 3, 10, 12, 13, 15–18

hglasso-package, 2

hglassoBIC, 2, 11, 12

HubNetwork, 2, 4, 14

image, 15

image.hbn (image.hglasso), 15

image.hcov, 9

image.hcov (image.hglasso), 15

image.hglasso, 2, 6, 11, 15, 17, 18

layout.kamada.kawai, 16

plot.hbn (plot.hglasso), 16

plot.hcov, 9

plot.hcov (plot.hglasso), 16

plot.hglasso, 2, 6, 11, 15, 16, 18

plot.igraph, 16

summary.hbn (summary.hglasso), 17

summary.hcov, 9

summary.hcov (summary.hglasso), 17

summary.hglasso, 2, 6, 11, 15, 17, 17