Package 'fssemR'

June 2, 2019

Title Fused Sparse Structural Equation Models to Jointly Infer Gene Regulatory Network

Version 0.1.5

Author Xin Zhou, Xiaodong Cai

Maintainer Xin Zhou <xxz220@miami.edu>

Description An optimizer of Fused-Sparse Structural Equation Models, which is the state of the art jointly fused sparse maximum likelihood function for structural equation models proposed by Xin Zhou and Xiaodong Cai (2018 <doi:10.1101/466623>).

License GPL (>= 3)

Encoding UTF-8

LazyData true

Depends methods

Imports Rcpp, Matrix, stats, igraph, mvtnorm, qtl, stringr, glmnet, MASS

Suggests plotly, knitr, rmarkdown, network, ggnetwork

LinkingTo Rcpp, RcppEigen

RoxygenNote 6.1.1

URL https://github.com/Ivis4ml/fssemR

NeedsCompilation yes

Repository CRAN

VignetteBuilder knitr

Date/Publication 2019-06-02 14:40:03 UTC

R topics documented:

cv.multiFSSEMiPALM 2
cv.multiFSSEMiPALM2 3
cv.multiRegression 4
cwiseGradient4FSSEM 4

FDR 5
flinvB 5
floneB 6
fssemR 6
implFSSEM 7
initLambdaiPALM 8
initLambdaiPALM2 8
initRhoiPALM 9
initRhoiPALM2 10
inverseB 10
invoneB 11
logLikFSSEM 11
multiFSSEMiPALM 12
multiFSSEMiPALM2 14
multiRegression 16
obj.multiRegression 17
opt.multiFSSEMiPALM 17
opt.multiFSSEMiPALM2 19
proc.centerFSSEM 20
proc.centerFSSEM2 21
randomFSSEMdata 21
randomFSSEMdata2 22
TPR 23
transx 23

Index **24**

`cv.multiFSSEMiPALM` *cv.multiFSSEMiPALM*

Description

cv.multiFSSEMiPALM

Usage

```
cv.multiFSSEMiPALM(Xs, Ys, Bs, Fs, Sk, sigma2, nlambda = 20, nrho = 20,
  nfold = 5, p, q, wt = TRUE, plot = FALSE)
```

Arguments

`Xs`	eQTL matrices
`Ys`	Gene expression matrices
`Bs`	initialized GRN-matrices
`Fs`	initialized eQTL effect matrices
`Sk`	eQTL index of genes
`sigma2`	initialized noise variance

nlambda	number of hyper-parameter of lasso term in CV
nrho	number of hyper-parameter of fused-lasso term in CV
nfold	CVfold number. Default 5/10
p	number of genes
q	number of eQTLs
wt	use adaptive lasso or not. Default TRUE.
plot	plot contour of cvmean or not. Default FALSE.

Value

list of cross-validation result

`cv.multiFSSEMiPALM2` *cv.multiFSSEMiPALM2*

Description

cv.multiFSSEMiPALM2

Usage

```
cv.multiFSSEMiPALM2(Xs, Ys, Bs, Fs, Sk, sigma2, nlambda = 20,
  nrho = 20, nfold = 5, p, q, wt = TRUE, plot = FALSE)
```

Arguments

Xs	eQTL matrices
Ys	Gene expression matrices
Bs	initialized GRN-matrices
Fs	initialized eQTL effect matrices
Sk	eQTL index of genes
sigma2	initialized noise variance
nlambda	number of hyper-parameter of lasso term in CV
nrho	number of hyper-parameter of fused-lasso term in CV
nfold	CVfold number. Default 5/10
p	number of genes
q	number of eQTLs
wt	use adaptive lasso or not. Default TRUE.
plot	plot contour of cvmean or not. Default FALSE.

Value

list of cross-validation result

cv.multiRegression *cv.multiRegression*

Description

cv.multiRegression

Usage

```
cv.multiRegression(Xs, Ys, Sk, ngamma = 20, nfold = 5, n, p, k)
```

Arguments

Xs	eQTL matrices
Ys	Gene expression matrices
Sk	eQTL index of genes
ngamma	number of hyper-parameter in CV
nfold	CVfold number. Default 5/10
n	number of observations
p	number of genes
k	number of eQTLs

Value

gamma_min optimal gamma to minimize cross-validation error

cwiseGradient4FSSEM *cwiseGradient4FSSEM*

Description

function generator function

Usage

```
cwiseGradient4FSSEM(n, c, Y, R, Y2norm, sigma2)
```

Arguments

n	number of observations
c	cofactor vector
Y	Matrix of gene expression
R	Residual matrix
Y2norm	Column of YtY
sigma2	noise variance

Value

function whose argument is column vector bi

`FDR` *FDR*

Description

False discovery rate for network prediction

Usage

```
FDR(X, B, PREC = 0)
```

Arguments

`X`	list of predicted network matrices
`B`	list of true network matrices
`PREC`	precision threshold for FDR test. Default 0.

`flinvB` *flinvB*

Description

inversed difference of two B matrices. For adaptive fused lasso penalty

Usage

```
flinvB(Bs)
```

Arguments

`Bs`	list of network matrices

Value

inversed difference matrices

floneB *floneB*

Description

if you do not want adaptive fused lasso penalty, floneB replace flinvB

Usage

```
floneB(Bs)
```

Arguments

Bs	list of network matrices

Value

matrix whose entries are all 1

fssemR *Solving Sparse Structural Equation Model*

Description

Solving Sparse Structural Equation Model

Author(s)

Xin Zhou <<xxz220@miami.edu>>

Examples

```
seed = as.numeric(Sys.time())
N = 100                                                 # sample size
Ng = 5                                                  # gene number
Nk = 5 * 3                                              # eQTL number
Ns = 1                                                  # sparse ratio
sigma2 = 0.01                                           # sigma2
set.seed(seed)
library(fssemR)
data = randomFSSEMdata(n = N, p = Ng, k = Nk, sparse = Ns, df = 0.3, sigma2 = sigma2,
                       u = 5, type = "DG", nhub = 1, dag = TRUE)
gamma = cv.multiRegression(data$Data$X, data$Data$Y, data$Data$Sk, ngamma = 20, nfold = 5,
                           N, Ng, Nk)
fit   = multiRegression(data$Data$X, data$Data$Y, data$Data$Sk, gamma, N, Ng, Nk,
                        trans = FALSE)
Xs    = data$Data$X
```

```
Ys    = data$Data$Y
Sk    = data$Data$Sk

## cross-validation
## cvfitc <- cv.multiFSSEMiPALM(Xs = Xs, Ys = Ys, Bs = fit$Bs, Fs = fit$Fs, Sk = Sk,
##                              sigma2 = fit$sigma2, nlambda = 10, nrho = 10,
##                              nfold = 5, p = Ng, q = Nk, wt = TRUE)

fitm <- opt.multiFSSEMiPALM(Xs = Xs, Ys = Ys, Bs = fit$Bs, Fs = fit$Fs, Sk = Sk,
                            sigma2 = fit$sigma2, nlambda = 10, nrho = 10,
                            p = Ng, q = Nk, wt = TRUE)

fitc0 <- fitm$fit

(TPR(fitc0$Bs[[1]], data$Vars$B[[1]]) + TPR(fitc0$Bs[[2]], data$Vars$B[[2]])) / 2
(FDR(fitc0$Bs[[1]], data$Vars$B[[1]]) + FDR(fitc0$Bs[[2]], data$Vars$B[[2]])) / 2
TPR(fitc0$Bs[[1]] - fitc0$Bs[[2]], data$Vars$B[[1]] - data$Vars$B[[2]])
FDR(fitc0$Bs[[1]] - fitc0$Bs[[2]], data$Vars$B[[1]] - data$Vars$B[[2]])
```

`implFSSEM` *implFSSEM*

Description

implementor function of FSSEM solver

Usage

```
implFSSEM(data = NULL, method = c("CV", "BIC"))
```

Arguments

`data`	Data archive of experiment measurements, includeing eQTL matrices, Gene expression matrices of different conditions, marker of eQTLs and data generation SEM model
`method`	Use cross-validation (CV) or bayesian-information-criterion(BIC)

Value

List of TPR and FDR

`initLambdaiPALM` *initLambdaiPALM*

Description

initLambdaiPALM

Usage

```
initLambdaiPALM(Xs, Ys, Bs, Fs, Sk, sigma2, Wl, Wf, p, k)
```

Arguments

`Xs`	eQTL matrices
`Ys`	Gene expression matrices
`Bs`	initialized GRN-matrices
`Fs`	initialized eQTL effect matrices
`Sk`	eQTL index of genes
`sigma2`	initialized noise variance
`Wl`	weight matrices for adaptive lasso terms
`Wf`	weight matrix for adaptive fused lasso term
`p`	number of genes
`k`	number of eQTL

Value

lambda_max

`initLambdaiPALM2` *initLambdaiPALM2*

Description

initLambdaiPALM2

Usage

```
initLambdaiPALM2(Xs, Ys, Bs, Fs, Sk, sigma2, Wl, Wf, p, k)
```

Arguments

Xs	eQTL matrices
Ys	Gene expression matrices
Bs	initialized GRN-matrices
Fs	initialized eQTL effect matrices
Sk	eQTL index of genes
sigma2	initialized noise variance
Wl	weight matrices for adaptive lasso terms
Wf	weight matrix for adaptive fused lasso term
p	number of genes
k	number of eQTL

Value

lambda_max

---

initRhoiPALM *initRhoiPALM*

Description

initRhoiPALM

Usage

```
initRhoiPALM(Xs, Ys, Bs, Fs, Sk, sigma2, Wl, Wf, lambda, n, p)
```

Arguments

Xs	eQTL matrices
Ys	Gene expression matrices
Bs	initialized GRN-matrices
Fs	initialized eQTL effect matrices
Sk	eQTL index of genes
sigma2	initialized noise variance
Wl	weight matrices for adaptive lasso terms
Wf	weight matrix for adaptive fused lasso term
lambda	lambda w.r.t. rho_max
n	number of observations
p	number of genes

Value

rho_max

`initRhoiPALM2` *initRhoiPALM2*

Description

initRhoiPALM2

Usage

```
initRhoiPALM2(Xs, Ys, Bs, Fs, Sk, sigma2, Wl, Wf, lambda, n, p)
```

Arguments

`Xs`	eQTL matrices
`Ys`	Gene expression matrices
`Bs`	initialized GRN-matrices
`Fs`	initialized eQTL effect matrices
`Sk`	eQTL index of genes
`sigma2`	initialized noise variance
`Wl`	weight matrices for adaptive lasso terms
`Wf`	weight matrix for adaptive fused lasso term
`lambda`	lambda w.r.t. rho_max
`n`	number of observations
`p`	number of genes

Value

rho_max

`inverseB` *inverseB*

Description

inverse matrices of B network for adaptive FSSEM

Usage

```
inverseB(Bs)
```

Arguments

`Bs`	list of network matrices

Value

list of inversed B matrices

`invoneB` *invoneB*

Description

if you do not want to get inversed B matrces, invoneB gives you a matrix with constant 1 instead in FSSEM

Usage

```
invoneB(Bs)
```

Arguments

`Bs`	list of network matrices

Value

list of invone B matrices

`logLikFSSEM` *logLikFSSEM*

Description

logLikFSSEM

Usage

```
logLikFSSEM(Bs, Wl, Wf, lambda, rho, sigma2, Dets, n, p)
```

Arguments

`Bs`	Network matrices
`Wl`	Weights for lasso term
`Wf`	Weights for fused term
`lambda`	Hyperparameter of lasso term
`rho`	Hyperparameter of fused lasso term
`sigma2`	noise variance
`Dets`	determinants of I-B matrices
`n`	number of observations
`p`	number of genes

Value

objective value of FSSEM with specified hyper-paramters

`multiFSSEMiPALM` *multiFSSEMiPALM*

Description

Implementing FSSELM algorithm for network inference. If Xs is identify for different conditions, multiFSSEMiPALM will be use, otherwise, please use `multiFSSEMiPALM2` for general cases

Usage

```
multiFSSEMiPALM(Xs, Ys, Bs, Fs, Sk, sigma2, lambda, rho, Wl, Wf, p,
  maxit = 100, inert = inert_opt("linear"), threshold = 1e-06,
  verbose = TRUE, sparse = TRUE, trans = FALSE, B2norm = NULL,
  strict = FALSE)
```

Arguments

`Xs`	eQTL matrices
`Ys`	Gene expression matrices
`Bs`	initialized GRN-matrices
`Fs`	initialized eQTL effect matrices
`Sk`	eQTL index of genes
`sigma2`	initialized noise variance from ridge regression
`lambda`	Hyperparameter of lasso term in FSSEM
`rho`	Hyperparameter of fused-lasso term in FSSEM
`Wl`	weight matrices for adaptive lasso terms
`Wf`	weight matrix for adaptive fused lasso term
`p`	number of genes
`maxit`	maximum iteration number. Default 100
`inert`	inertial function for iPALM. Default as k-1/k+2
`threshold`	convergence threshold. Default 1e-6
`verbose`	Default TRUE
`sparse`	Sparse Matrix or not
`trans`	Fs matrix is transposed to k x p or not. If Fs from ridge regression, trans = TRUE, else, trans = FALSE
`B2norm`	B2norm matrices generated from ridge regression. Default NULL.
`strict`	Converge strictly or not. Default False

Value

fit List of FSSEM model

Bs coefficient matrices of gene regulatory networks

Fs coefficient matrices of eQTL-gene effect

mu Bias vector

sigma2 estimate of covariance in SEM

Examples

```
seed = 1234
N = 100                                          # sample size
Ng = 5                                           # gene number
Nk = 5 * 3                                       # eQTL number
Ns = 1                                           # sparse ratio
sigma2 = 0.01                                    # sigma2
set.seed(seed)
library(fssemR)
data = randomFSSEMdata(n = N, p = Ng, k = Nk, sparse = Ns, df = 0.3, sigma2 = sigma2,
                       u = 5, type = "DG", nhub = 1, dag = TRUE)
## If we assume that different condition has different genetics perturbations (eQTLs)
## gamma = cv.multiRegression(data$Data$X, data$Data$Y, data$Data$Sk, ngamma = 20, nfold = 5,
##                            N, Ng, Nk)
gamma = 0.6784248     ## optimal gamma computed by cv.multiRegression
fit   = multiRegression(data$Data$X, data$Data$Y, data$Data$Sk, gamma, N, Ng, Nk,
                        trans = FALSE)
Xs    = data$Data$X
Ys    = data$Data$Y
Sk    = data$Data$Sk


cvfitc <- cv.multiFSSEMiPALM(Xs = Xs, Ys = Ys, Bs = fit$Bs, Fs = fit$Fs, Sk = Sk,
                             sigma2 = fit$sigma2, nlambda = 5, nrho = 5,
                             nfold = 5, p = Ng, q = Nk, wt = TRUE)

fitc0 <- multiFSSEMiPALM(Xs = Xs, Ys = Ys, Bs = fit$Bs, Fs = fit$Fs, Sk = Sk,
                         sigma2 = fit$sigma2, lambda = cvfitc$lambda, rho = cvfitc$rho,
                         Wl = inverseB(fit$Bs), Wf = flinvB(fit$Bs),
                         p = Ng, maxit = 100, threshold = 1e-5, sparse = TRUE,
                         verbose = TRUE, trans = TRUE, strict = TRUE)


(TPR(fitc0$Bs[[1]], data$Vars$B[[1]]) + TPR(fitc0$Bs[[2]], data$Vars$B[[2]])) / 2
(FDR(fitc0$Bs[[1]], data$Vars$B[[1]]) + FDR(fitc0$Bs[[2]], data$Vars$B[[2]])) / 2
TPR(fitc0$Bs[[1]] - fitc0$Bs[[2]], data$Vars$B[[1]] - data$Vars$B[[2]])
FDR(fitc0$Bs[[1]] - fitc0$Bs[[2]], data$Vars$B[[1]] - data$Vars$B[[2]])
```

multiFSSEMiPALM2 *multiFSSEMiPALM2*

Description

Implementing FSSELM algorithm for network inference. If Xs is identify for different conditions, multiFSSEMiPALM will be use, otherwise, please use multiFSSEMiPALM2 for general cases

Usage

```
multiFSSEMiPALM2(Xs, Ys, Bs, Fs, Sk, sigma2, lambda, rho, Wl, Wf, p,
  maxit = 100, inert = inert_opt("linear"), threshold = 1e-06,
  verbose = TRUE, sparse = TRUE, trans = FALSE, B2norm = NULL,
  strict = FALSE)
```

Arguments

Xs	eQTL matrices
Ys	Gene expression matrices
Bs	initialized GRN-matrices
Fs	initialized eQTL effect matrices
Sk	eQTL index of genes
sigma2	initialized noise variance from ridge regression
lambda	Hyperparameter of lasso term in FSSEM
rho	Hyperparameter of fused-lasso term in FSSEM
Wl	weight matrices for adaptive lasso terms
Wf	weight matrix for adaptive fused lasso term
p	number of genes
maxit	maximum iteration number. Default 100
inert	inertial function for iPALM. Default as k-1/k+2
threshold	convergence threshold. Default 1e-6
verbose	Default TRUE
sparse	Sparse Matrix or not
trans	Fs matrix is transposed to k x p or not. If Fs from ridge regression, trans = TRUE, else, trans = FALSE
B2norm	B2norm matrices generated from ridge regression. Default NULL.
strict	Converge strictly or not. Default False

Value

fit List of FSSEM model

Bs coefficient matrices of gene regulatory networks

Fs coefficient matrices of eQTL-gene effect

mu Bias vector

sigma2 estimate of covariance in SEM

Examples

```
seed = 1234
N = 100                                             # sample size
Ng = 5                                              # gene number
Nk = 5 * 3                                          # eQTL number
Ns = 1                                              # sparse ratio
sigma2 = 0.01                                       # sigma2
set.seed(seed)
library(fssemR)
data = randomFSSEMdata(n = N, p = Ng, k = Nk, sparse = Ns, df = 0.3, sigma2 = sigma2,
                       u = 5, type = "DG", nhub = 1, dag = TRUE)
## If we assume that different condition has different genetics perturbations (eQTLs)
data$Data$X = list(data$Data$X, data$Data$X)
## gamma = cv.multiRegression(data$Data$X, data$Data$Y, data$Data$Sk, ngamma = 20, nfold = 5,
##                            N, Ng, Nk)
gamma = 0.6784248     ## optimal gamma computed by cv.multiRegression
fit   = multiRegression(data$Data$X, data$Data$Y, data$Data$Sk, gamma, N, Ng, Nk,
                      trans = FALSE)
Xs    = data$Data$X
Ys    = data$Data$Y
Sk    = data$Data$Sk


cvfitc <- cv.multiFSSEMiPALM2(Xs = Xs, Ys = Ys, Bs = fit$Bs, Fs = fit$Fs, Sk = Sk,
                            sigma2 = fit$sigma2, nlambda = 5, nrho = 5,
                            nfold = 5, p = Ng, q = Nk, wt = TRUE)

fitc0 <- multiFSSEMiPALM2(Xs = Xs, Ys = Ys, Bs = fit$Bs, Fs = fit$Fs, Sk = Sk,
                          sigma2 = fit$sigma2, lambda = cvfitc$lambda, rho = cvfitc$rho,
                          W1 = inverseB(fit$Bs), Wf = flinvB(fit$Bs),
                          p = Ng, maxit = 100, threshold = 1e-5, sparse = TRUE,
                          verbose = TRUE, trans = TRUE, strict = TRUE)


(TPR(fitc0$Bs[[1]], data$Vars$B[[1]]) + TPR(fitc0$Bs[[2]], data$Vars$B[[2]])) / 2
(FDR(fitc0$Bs[[1]], data$Vars$B[[1]]) + FDR(fitc0$Bs[[2]], data$Vars$B[[2]])) / 2
TPR(fitc0$Bs[[1]] - fitc0$Bs[[2]], data$Vars$B[[1]] - data$Vars$B[[2]])
FDR(fitc0$Bs[[1]] - fitc0$Bs[[2]], data$Vars$B[[1]] - data$Vars$B[[2]])
```

multiRegression *multiRegression*

Description

Ridge regression on multiple conditions, initialization of FSSEM algorithm

Usage

```
multiRegression(Xs, Ys, Sk, gamma, n, p, k, trans = FALSE)
```

Arguments

Xs	eQTL matrices. eQTL matrix can be matrix/list of multiple conditions
Ys	Gene expression matrices
Sk	eQTL index of genes
gamma	Hyperparameter for ridge regression
n	number of observations
p	number of genes
k	number of eQTLs
trans	if rows for sample, trans = TRUE, otherwise, trans = FALSE. Default FALSE

Value

fit List of SEM model

Bs coefficient matrices of gene regulatory networks

fs eQTL's coefficients w.r.t each gene

Fs coefficient matrices of eQTL-gene effect

mu Bias vector

sigma2 estimate of covariance in SEM

Examples

```
seed = 1234
N = 100                                          # sample size
Ng = 5                                           # gene number
Nk = 5 * 3                                       # eQTL number
Ns = 1                                           # sparse ratio
sigma2 = 0.01                                    # sigma2
set.seed(seed)
data = randomFSSEMdata(n = N, p = Ng, k = Nk, sparse = Ns, df = 0.3, sigma2 = sigma2,
                       u = 5, type = "DG", nhub = 1, dag = TRUE)
## If we assume that different condition has different genetics perturbations (eQTLs)
## data$Data$X = list(data$Data$X, data$Data$X)
```

```
gamma = cv.multiRegression(data$Data$X, data$Data$Y, data$Data$Sk, ngamma = 20, nfold = 5,
                           N, Ng, Nk)
fit   = multiRegression(data$Data$X, data$Data$Y, data$Data$Sk, gamma, N, Ng, Nk,
                        trans = FALSE)
```

`obj.multiRegression` *obj.multiRegression*

Description

obj.multiRegression

Usage

```
obj.multiRegression(Xs, Ys, fit, trans = F)
```

Arguments

`Xs`	eQTL matrices
`Ys`	gene expression matrices
`fit`	regression fit result object
`trans`	if rows for sample, trans = TRUE, otherwise, trans = FALSE. Default FALSE

Value

error squared norm of ||(I-B)Y - FX||_2^2

`opt.multiFSSEMiPALM` *opt.multiFSSEMiPALM*

Description

optimize multiFSSEMiPALM's parameters by minimize BIC, when feature size is large (> 300), BIC methods will be much faster than Cross-validation

Usage

```
opt.multiFSSEMiPALM(Xs, Ys, Bs, Fs, Sk, sigma2, nlambda = 20,
  nrho = 20, p, q, wt = TRUE)
```

Arguments

Xs	eQTL matrices
Ys	Gene expression matrices
Bs	initialized GRN-matrices
Fs	initialized eQTL effect matrices
Sk	eQTL index of genes
sigma2	initialized noise variance
nlambda	number of hyper-parameter of lasso term in CV
nrho	number of hyper-parameter of fused-lasso term in CV
p	number of genes
q	number of eQTLs
wt	use adaptive lasso or not. Default TRUE.

Value

list of model selection result

Examples

```
seed = 1234
N = 100                                                 # sample size
Ng = 5                                                  # gene number
Nk = 5 * 3                                              # eQTL number
Ns = 1                                                  # sparse ratio
sigma2 = 0.01                                           # sigma2
set.seed(seed)
library(fssemR)
data = randomFSSEMdata(n = N, p = Ng, k = Nk, sparse = Ns, df = 0.3, sigma2 = sigma2,
                       u = 5, type = "DG", nhub = 1, dag = TRUE)
## If we assume that different condition has different genetics perturbations (eQTLs)
## data$Data$X = list(data$Data$X, data$Data$X)
## gamma = cv.multiRegression(data$Data$X, data$Data$Y, data$Data$Sk, ngamma = 20, nfold = 5,
##                            N, Ng, Nk)
gamma = 0.6784248     ## optimal gamma computed by cv.multiRegression
fit   = multiRegression(data$Data$X, data$Data$Y, data$Data$Sk, gamma, N, Ng, Nk,
                        trans = FALSE)
Xs    = data$Data$X
Ys    = data$Data$Y
Sk    = data$Data$Sk

fitm <- opt.multiFSSEMiPALM(Xs = Xs, Ys = Ys, Bs = fit$Bs, Fs = fit$Fs, Sk = Sk,
                            sigma2 = fit$sigma2, nlambda = 10, nrho = 10,
                            p = Ng, q = Nk, wt = TRUE)

fitc0 <- fitm$fit

(TPR(fitc0$Bs[[1]], data$Vars$B[[1]]) + TPR(fitc0$Bs[[2]], data$Vars$B[[2]])) / 2
(FDR(fitc0$Bs[[1]], data$Vars$B[[1]]) + FDR(fitc0$Bs[[2]], data$Vars$B[[2]])) / 2
```

```
TPR(fitc0$Bs[[1]] - fitc0$Bs[[2]], data$Vars$B[[1]] - data$Vars$B[[2]])
FDR(fitc0$Bs[[1]] - fitc0$Bs[[2]], data$Vars$B[[1]] - data$Vars$B[[2]])
```

---

opt.multiFSSEMiPALM2 *opt.multiFSSEMiPALM2*

Description

optimize multiFSSEMiPALM's parameters by minimize BIC, when feature size is large (> 300), BIC methods will be much faster than Cross-validation

Usage

```
opt.multiFSSEMiPALM2(Xs, Ys, Bs, Fs, Sk, sigma2, nlambda = 20,
  nrho = 20, p, q, wt = TRUE)
```

Arguments

Xs	eQTL matrices
Ys	Gene expression matrices
Bs	initialized GRN-matrices
Fs	initialized eQTL effect matrices
Sk	eQTL index of genes
sigma2	initialized noise variance
nlambda	number of hyper-parameter of lasso term in CV
nrho	number of hyper-parameter of fused-lasso term in CV
p	number of genes
q	number of eQTLs
wt	use adaptive lasso or not. Default TRUE.

Value

list of model selection result

Examples

```
seed = 1234
N = 100                                                 # sample size
Ng = 5                                                  # gene number
Nk = 5 * 3                                              # eQTL number
Ns = 1                                                  # sparse ratio
sigma2 = 0.01                                           # sigma2
set.seed(seed)
library(fssemR)
data = randomFSSEMdata(n = N, p = Ng, k = Nk, sparse = Ns, df = 0.3, sigma2 = sigma2,
```

```
                        u = 5, type = "DG", nhub = 1, dag = TRUE)
## If we assume that different condition has different genetics perturbations (eQTLs)
data$Data$X = list(data$Data$X, data$Data$X)
## gamma = cv.multiRegression(data$Data$X, data$Data$Y, data$Data$Sk, ngamma = 20, nfold = 5,
##                            N, Ng, Nk)
gamma = 0.6784248     ## optimal gamma computed by cv.multiRegression
fit   = multiRegression(data$Data$X, data$Data$Y, data$Data$Sk, gamma, N, Ng, Nk,
                        trans = FALSE)
Xs    = data$Data$X
Ys    = data$Data$Y
Sk    = data$Data$Sk

fitm <- opt.multiFSSEMiPALM2(Xs = Xs, Ys = Ys, Bs = fit$Bs, Fs = fit$Fs, Sk = Sk,
                             sigma2 = fit$sigma2, nlambda = 10, nrho = 10,
                             p = Ng, q = Nk, wt = TRUE)

fitc0 <- fitm$fit

(TPR(fitc0$Bs[[1]], data$Vars$B[[1]]) + TPR(fitc0$Bs[[2]], data$Vars$B[[2]])) / 2
(FDR(fitc0$Bs[[1]], data$Vars$B[[1]]) + FDR(fitc0$Bs[[2]], data$Vars$B[[2]])) / 2
TPR(fitc0$Bs[[1]] - fitc0$Bs[[2]], data$Vars$B[[1]] - data$Vars$B[[2]])
FDR(fitc0$Bs[[1]] - fitc0$Bs[[2]], data$Vars$B[[1]] - data$Vars$B[[2]])
```

`proc.centerFSSEM` *proc.centerFSSEM*

Description

proc.centerFSSEM

Usage

```
proc.centerFSSEM(Xs, Ys)
```

Arguments

`Xs`	eQTL matrices
`Ys`	list of gene expression matrices

Value

centered Xs and Ys and mean vectors

proc.centerFSSEM2 *proc.centerFSSEM2*

Description

proc.centerFSSEM2

Usage

```
proc.centerFSSEM2(Xs, Ys)
```

Arguments

Xs	list of eQTL matrices
Ys	list of gene expression matrices

Value

centered Xs and Ys and mean vectors

randomFSSEMdata *randomFSSEMdata*

Description

randomFSSEMdata

Usage

```
randomFSSEMdata(n, p, k, sparse = 0.1, df = 0.2, sigma2 = 0.01,
  u = 5, type = c("DG", "ER"), dag = TRUE, coef = c(0.2, 0.4),
  nhub = 2)
```

Arguments

n	number of observations
p	number of genes
k	number of eQTLs
sparse	ratio of edges / gene_number
df	ratio of differential edges among two network
sigma2	noise variance of error
u	variance of bias in SEM model.
type	type of generated network, can be selected as DG, ER, Scale-free network

dag	network is directed-acyclic or not. Default TRUE
coef	Range of absolute value of coefficients in simulated network matrices. Default (0.2, 0.4), or (0.5, 1)
nhub	If you select to generate ER network, nhub is the number of pre-defined hub node number. Default 2

Value

list of generated data

Data List of observed, Xs, Ys, Sk

Vars List of model, Bs, Fs, mu, n, p, k

`randomFSSEMdata2` *randomFSSEMdata2*

Description

randomFSSEMdata2

Usage

```
randomFSSEMdata2(n, p, k, sparse = 0.1, df = 0.2, sigma2 = 0.01,
  u = 5, type = c("DG", "ER"), dag = TRUE, coef = c(0.2, 0.4),
  nhub = 2)
```

Arguments

n	number of observations. Vector for unbalance observations
p	number of genes
k	number of eQTLs
sparse	ratio of edges / gene_number
df	ratio of differential edges among two network
sigma2	noise variance of error
u	variance of bias in SEM model.
type	type of generated network, can be selected as DG, ER, Scale-free network
dag	network is directed-acyclic or not. Default TRUE
coef	Range of absolute value of coefficients in simulated network matrices. Default (0.2, 0.4), or (0.5, 1)
nhub	If you select to generate ER network, nhub is the number of pre-defined hub node number. Default 2

Value

list of generated data

Data List of observed, Xs, Ys, Sk

Vars List of model, Bs, Fs, mu, n, p, k

`TPR` *TPR*

Description

Power of detection for network prediction

Usage

```
TPR(X, B, PREC = 0)
```

Arguments

`X`	list of predicted network matrices
`B`	list of true network matrices
`PREC`	precision threshold for FDR test. Default 0.

`transx` *transx*

Description

transx

Usage

```
transx(data)
```

Arguments

`data`	Collecting data structure generated by randomFSSEMdata function

Value

transformed list of eQTL matrices

Index

cv.multiFSSEMiPALM, 2
cv.multiFSSEMiPALM2, 3
cv.multiRegression, 4
cwiseGradient4FSSEM, 4

FDR, 5
flinvB, 5
floneB, 6
fssemR, 6
fssemR-package (fssemR), 6

implFSSEM, 7
initLambdaiPALM, 8
initLambdaiPALM2, 8
initRhoiPALM, 9
initRhoiPALM2, 10
inverseB, 10
invoneB, 11

logLikFSSEM, 11

multiFSSEMiPALM, 12
multiFSSEMiPALM2, 14
multiRegression, 16

obj.multiRegression, 17
opt.multiFSSEMiPALM, 17
opt.multiFSSEMiPALM2, 19

package-fssemR (fssemR), 6
proc.centerFSSEM, 20
proc.centerFSSEM2, 21

randomFSSEMdata, 21
randomFSSEMdata2, 22

TPR, 23
transx, 23