Package ‘OPTtesting’

October 12, 2022

Type Package

Title Optimal Testing

Version 1.0.0

Maintainer Lijia Wang <lijiawan@usc.edu>

Description Optimal testing under general dependence. The R package implements procedures proposed in Wang, Han, and Tong (2022). The package includes parameter estimation procedures, the computation for the posterior probabilities, and the testing procedure.

Encoding UTF-8

Imports rootSolve, quantreg, RSpectra, stats

Suggests MASS

RoxygenNote 7.1.2

License GPL-2

NeedsCompilation no

Author Lijia Wang [aut, cre, cph],
Xu Han [aut],
Xin Tong [aut]

Repository CRAN

Date/Publication 2022-05-26 13:30:09 UTC

R topics documented:

AEB *AEB*

Description

Estimate the parameters in the three-part mixture

Usage

```
AEB(
  Z,
  Sigma,
  eig = eigs_sym(Sigma, min(400, length(Z)), which = "LM"),
  eig_tol = 1,
  set_nu = c(0),
  set1 = c(0:10) * 0.01 + 0.01,
  set2 = c(0:10) * 0.01 + 0.01,
  setp = c(1:7) * 0.1
)
```

Arguments

`Z`	a vector of test statistics
`Sigma`	covariance matrix
`eig`	eig value information
`eig_tol`	the smallest eigen value used in the calulation
`set_nu`	a search region for nu_0
`set1`	a search region for tau_sqr_1
`set2`	a search region for tau_sqr_2
`setp`	a search region for proportion

Details

Estimate the parameters in the three-part mixture $Z|\mu~N_p(\mu,\Sigma)$ where $\mu_i~\pi_0\delta_{\nu_0}+\pi_1 N(\mu_1,\tau_1^2)+\pi_2 N(\mu_2,\tau_2^2), i=1,...,p$

Value

The return of the function is a list in the form of list(nu_0, tau_sqr_1, tau_sqr_2, pi_0, pi_1, pi_2, mu_1, mu_2, Z_hat).

nu_0, tau_sqr_1, tau_sqr_2: The best combination of $(\nu_0,\tau_1^2,\tau_2^2)$ that minimize the total variance from the regions $(D_{\nu_0},D_{\tau_1^2},D_{\tau_2^2})$.

pi_0, pi_1, pi_2, mu_1, mu_2: The estimates of parameters $\pi_0,\pi_1,\pi_2,\mu_1,\mu_2$.

Z_hat: A vector of simulated data base on the parameter estimates.

Examples

```
p = 500
 n_col = 10
A = matrix(rnorm(p*n_col,0,1), nrow = p, ncol = n_col, byrow = TRUE)
Sigma = A %*% t(A) +diag(p)
Sigma = cov2cor(Sigma) #covariance matrix
Z = rnorm(p,0,1) #this is just an example for testing the algorithm.
#Not true test statistics with respect to Sigma
best_set =  AEB(Z,Sigma)
print(c(best_set$pi_0, best_set$pi_1, best_set$pi_2, best_set$mu_1, best_set$mu_2))

library(MASS)
######################################
#construct a test statistic vector Z
p = 1000
n_col = 4
pi_0 = 0.6
pi_1 = 0.2
pi_2 = 0.2
nu_0 = 0
mu_1 = -1.5
mu_2 = 1.5
tau_sqr_1 = 0.1
tau_sqr_2 = 0.1


A = matrix(rnorm(p*n_col,0,1), nrow = p, ncol = n_col, byrow = TRUE)
Sigma = A %*% t(A) +diag(p)
Sigma = cov2cor(Sigma) #covariance matrix




b = rmultinom(p, size = 1, prob = c(pi_0,pi_1,pi_2))
ui = b[1,]*nu_0 + b[2,]*rnorm(p, mean = mu_1,
     sd = sqrt(tau_sqr_1)) + b[3,]*rnorm(p, mean = mu_2,
      sd = sqrt(tau_sqr_2)) # actual situation
Z = mvrnorm(n = 1,ui, Sigma, tol = 1e-6, empirical = FALSE, EISPACK = FALSE)

best_set =AEB(Z,Sigma)
print(c(best_set$pi_0, best_set$pi_1, best_set$pi_2, best_set$mu_1, best_set$mu_2))
```

---

d_value *d_value*

Description

Calculating the estimates for $P(\mu_i \leq 0|Z)$

Usage

```
d_value(
  Z,
  Sigma,
  best_set = AEB(Z, Sigma),
  eig = eigs_sym(Sigma, min(400, length(Z)), which = "LM"),
  sim_size = 3000,
  eig_value = 0.35
)
```

Arguments

`Z`	a vector of test statistics
`Sigma`	covariance matrix
`best_set`	a list of parameters (list(nu_0 = ..., tau_sqr_1 = ..., tau_sqr_2 = ..., pi_0 = ..., pi_1= ..., pi_2 = ..., mu_1 = ..., mu_2 = ...)) or returns from AEB
`eig`	eig value information
`sim_size`	simulation size
`eig_value`	the smallest eigen value used in the calulation

Value

a vector of estimates for $P(\mu_i \leq 0|Z), i = 1, ..., p$

Examples

```
p = 500
n_col = 10
A = matrix(rnorm(p*n_col,0,1), nrow = p, ncol = n_col, byrow = TRUE)
Sigma = A %*% t(A) +diag(p)
Sigma = cov2cor(Sigma) #covariance matrix
Z = rnorm(p,0,1) #this is just an example for testing the algorithm.
#Not true test statistics with respect to Sigma
 d_value(Z,Sigma,sim_size = 5)
#To save time, we set the simulation size to be 10, but the default value is much better.

library(MASS)
######################################
#construct a test statistic vector Z
p = 1000
n_col = 4
pi_0 = 0.6
pi_1 = 0.2
pi_2 = 0.2
nu_0 = 0
mu_1 = -1.5
mu_2 = 1.5
tau_sqr_1 = 0.1
tau_sqr_2 = 0.1
```

```
A = matrix(rnorm(p*n_col,0,1), nrow = p, ncol = n_col, byrow = TRUE)
Sigma = A %*% t(A) +diag(p)
Sigma = cov2cor(Sigma) #covariance matrix




b = rmultinom(p, size = 1, prob = c(pi_0,pi_1,pi_2))
ui = b[1,]*nu_0 + b[2,]*rnorm(p, mean = mu_1,
     sd = sqrt(tau_sqr_1)) + b[3,]*rnorm(p, mean = mu_2,
      sd = sqrt(tau_sqr_2)) # actual situation
Z = mvrnorm(n = 1,ui, Sigma, tol = 1e-6, empirical = FALSE, EISPACK = FALSE)

d_value(Z,Sigma)
```

`FDP_compute` *FDP_compute*

Description

False discovery proportion and False non-discovery proportion computation

Usage

```
FDP_compute(decision, ui, positive)
```

Arguments

`decision`	returns from the function Optimal_procedure_3
`ui`	true mean vector
`positive`	TRUE/FALSE valued. TRUE: H0: ui no greater than 0. FALSE: H0: ui no less than 0.

Value

False discovery proportion (FDP) and False non-discovery proportion (FNP)

Examples

```
ui = rnorm(10,0,1) #assume this is true parameter
decision = rbinom(10,1, 0.4) #assume this is decision vector
FDP_compute(decision,ui,TRUE)

library(MASS)
######################################
```

```
#construct a test statistic vector Z
p = 1000
n_col = 4
pi_0 = 0.6
pi_1 = 0.2
pi_2 = 0.2
nu_0 = 0
mu_1 = -1.5
mu_2 = 1.5
tau_sqr_1 = 0.1
tau_sqr_2 = 0.1


A = matrix(rnorm(p*n_col,0,1), nrow = p, ncol = n_col, byrow = TRUE)
Sigma = A %*% t(A) +diag(p)
Sigma = cov2cor(Sigma) #covariance matrix




b = rmultinom(p, size = 1, prob = c(pi_0,pi_1,pi_2))
ui = b[1,]*nu_0 + b[2,]*rnorm(p, mean = mu_1,
     sd = sqrt(tau_sqr_1)) + b[3,]*rnorm(p, mean = mu_2,
      sd = sqrt(tau_sqr_2)) # actual situation
Z = mvrnorm(n = 1,ui, Sigma, tol = 1e-6, empirical = FALSE, EISPACK = FALSE)

prob_p = d_value(Z,Sigma)
#decision
level = 0.1 #significance level
decision_p = Optimal_procedure_3(prob_p,level)
FDP_compute(decision_p$ai,ui,TRUE)
```

---

l_value *l_value*

Description

Calculating the estimates for $P(\mu_i \geq 0|Z)$

Usage

```
l_value(
  Z,
  Sigma,
  best_set = AEB(Z, Sigma),
  eig = eigs_sym(Sigma, min(400, length(Z)), which = "LM"),
  sim_size = 3000,
  eig_value = 0.35
)
```

Arguments

Z	a vector of test statistics
Sigma	covariance matrix
best_set	a list of parameters (list(nu_0 = ..., tau_sqr_1 = ..., tau_sqr_2 = ..., pi_0 = ..., pi_1= ..., pi_2 = ..., mu_1 = ..., mu_2 = ...)) or returns from Fund_parameter_estimation
eig	eig value information
sim_size	simulation size
eig_value	the smallest eigen value used in the calulation

Value

a vector of estimates for $P(\mu_i \geq 0|Z)$

Examples

```
p = 500
 n_col = 10
A = matrix(rnorm(p*n_col,0,1), nrow = p, ncol = n_col, byrow = TRUE)
Sigma = A %*% t(A) +diag(p)
Sigma = cov2cor(Sigma) #covariance matrix
Z = rnorm(p,0,1) #this is just an example for testing the algorithm.
#Not true test statistics with respect to Sigma
 l_value(Z,Sigma,sim_size = 5)
 #To save time, we set the simulation size to be 10, but the default value is much better.

library(MASS)
######################################
#construct a test statistic vector Z
p = 1000
n_col = 4
pi_0 = 0.6
pi_1 = 0.2
pi_2 = 0.2
nu_0 = 0
mu_1 = -1.5
mu_2 = 1.5
tau_sqr_1 = 0.1
tau_sqr_2 = 0.1


A = matrix(rnorm(p*n_col,0,1), nrow = p, ncol = n_col, byrow = TRUE)
Sigma = A %*% t(A) +diag(p)
Sigma = cov2cor(Sigma) #covariance matrix




b = rmultinom(p, size = 1, prob = c(pi_0,pi_1,pi_2))
ui = b[1,]*nu_0 + b[2,]*rnorm(p, mean = mu_1,
     sd = sqrt(tau_sqr_1)) + b[3,]*rnorm(p, mean = mu_2,
```

```
      sd = sqrt(tau_sqr_2)) # actual situation
Z = mvrnorm(n = 1,ui, Sigma, tol = 1e-6, empirical = FALSE, EISPACK = FALSE)

l_value(Z,Sigma)
```

Optimal_procedure_3   *Optimal_procedure_3*

Description

decision process

Usage

```
Optimal_procedure_3(prob_0, alpha)
```

Arguments

prob_0	d-values or l-values
alpha	significance level

Value

ai: a vector of decisions. (1 indicates rejection)

cj: The number of rejections

FDR_hat: The estimated false discovery rate (FDR).

FNR_hat: The estimated false non-discovery rate (FNR).

Examples

```
prob = runif(100,0,1) #assume this is the posterior probability vector
level = 0.3 #the significance level
Optimal_procedure_3(prob,level)

library(MASS)
######################################
#construct a test statistic vector Z
p = 1000
n_col = 4
pi_0 = 0.6
pi_1 = 0.2
pi_2 = 0.2
nu_0 = 0
mu_1 = -1.5
mu_2 = 1.5
tau_sqr_1 = 0.1
```

```
tau_sqr_2 = 0.1


A = matrix(rnorm(p*n_col,0,1), nrow = p, ncol = n_col, byrow = TRUE)
Sigma = A %*% t(A) +diag(p)
Sigma = cov2cor(Sigma) #covariance matrix




b = rmultinom(p, size = 1, prob = c(pi_0,pi_1,pi_2))
ui = b[1,]*nu_0 + b[2,]*rnorm(p, mean = mu_1,
     sd = sqrt(tau_sqr_1)) + b[3,]*rnorm(p, mean = mu_2,
      sd = sqrt(tau_sqr_2)) # actual situation
Z = mvrnorm(n = 1,ui, Sigma, tol = 1e-6, empirical = FALSE, EISPACK = FALSE)

prob_p = d_value(Z,Sigma)
#decision
level = 0.1 #significance level
decision_p = Optimal_procedure_3(prob_p,level)
decision_p$cj
head(decision_p$ai)
```

Index