Package: MLFDR (via r-universe)

November 16, 2026

Type Package

Title High Dimensional Mediation Analysis using Local False Discovery Rates

Version 0.1.0

Description Implements a high dimensional mediation analysis algorithm using Local False Discovery Rates. The methodology is described in Roy and Zhang (2024) <doi:10.48550/arXiv.2402.13933>.

License GPL-2

Encoding UTF-8

LazyData true

Imports NMOF, stats

RoxygenNote 7.3.3

Repository https://asmita112358.r-universe.dev

Date/Publication 2026-01-19 16:59:49 UTC

RemoteUrl https://github.com/asmita112358/mlfdr

RemoteRef HEAD

RemoteSha 7c348f75f2002d79065a76eef6d6d333b81183e1

Contents

localFDR 2
MLFDR 4

Index **6**

`localFDR` *localFDR: A function that fits a Gaussian Mixture model to the mediation coefficients and returns the localFDR estimated from the mixture model*

Description

localFDR: A function that fits a Gaussian Mixture model to the mediation coefficients and returns the localFDR estimated from the mixture model

Usage

```
localFDR(
  alpha,
  beta,
  var_alpha,
  var_beta,
  lambda.init = NULL,
  kappa.init = 1,
  psi.init = 1,
  psi_int = NULL,
  kappa_int = NULL,
  twostep = FALSE,
  k = 4,
  d1 = NULL,
  d2 = NULL,
  eps = 0.01,
  verbose = TRUE,
  method = "unicore"
)
```

Arguments

`alpha`	a vector of estimated alpha coefficients from the first equation of mediation analysis
`beta`	a vector of estimated beta coefficients from the second equation of mediation analysis
`var_alpha`	a vector of estimated variances for alpha coefficients
`var_beta`	a vector of estimated variances for beta coefficients
`lambda.init`	initial values of the proportion of mixture, must sum to 1
`kappa.init`	inital value of kappa, the variance of the prior of alpha under the alternative
`psi.init`	inital value of psi, the variance of the prior of beta under the alternative
`psi_int`	gridSearch interval for psi, to be used in optimize function
`kappa_int`	gridSearch interval for kappa, to be used in optimize function

twostep	logical, whether to use two-step MLFDR
k	number of mixture components, default is 4. Used in one-step MLFDR.
d1	number of non-null components for alpha in two-step MLFDR.
d2	number of non-null components for beta in two-step MLFDR.
eps	stopping criteria for EM algorithm
verbose	logical, whether to print the log-likelihood at each iteration. Default is TRUE.
method	either "unicore" or "multicore", specifies whether parallelization should be used when estimating variances in two step EM. Is not used unless twostep = TRUE

Value

A list consisting of two elements - lfdr, a vector of local false discovery rates and pi, the estimated mixture proportions

Examples

```
n = 100
m = 1000
pi = c(0.4, 0.1, 0.3, 0.2)
X = rbinom(n, 1, 0.1)
M = matrix(nrow = m, ncol = n)
Y = matrix(nrow = m, ncol = n)
gamma = sample(1:4, m, replace = TRUE, prob = pi)
alpha = vector()
beta = vector()
vec1 = rnorm(m, 0.05, 1)
vec2 = rnorm(m, -0.5, 2)
alpha = ((gamma==2) + (gamma == 4))*vec1
beta = ((gamma ==3) + (gamma == 4))*vec2
alpha_hat = vector()
beta_hat = vector()
var_alpha = c()
var_beta = c()
p1 = vector()
p2 = vector()
for(i in 1:m)
{
  M[i,] = alpha[i]*X + rnorm(n)
  Y[i,] = beta[i]*M[i,]  + rnorm(1,0.5)*X + rnorm(n)
  obj1 = lm(M[i,] ~  X )
  obj2 = lm(Y[i,] ~  M[i,] + X)
  table1 = coef(summary(obj1))
  table2 = coef(summary(obj2))
  alpha_hat[i] = table1["X",1]
  beta_hat[i] = table2["M[i, ]",1]
  p1[i] = table1["X",4]
  p2[i] = table2["M[i, ]",4]
  var_alpha[i] = table1["X",2]^2
  var_beta[i] = table2["M[i, ]",2]^2
}
lfdr <- localFDR(alpha_hat, beta_hat, var_alpha, var_beta, twostep = FALSE)
```

MLFDR *MLFDR: Mediation analysis using localFDR*

Description

MLFDR: Mediation analysis using localFDR

Usage

```
MLFDR(lfdr, size = 0.05)
```

Arguments

lfdr	a vector of local false discovery rates, as output from the function localFDR
size	Target False Discovery rate, default is 0.05

Value

A vector of logicals indicating which hypotheses are rejected

Examples

```
n = 100
m = 1000
pi = c(0.4, 0.1, 0.3, 0.2)
X = rbinom(n, 1, 0.1)
M = matrix(nrow = m, ncol = n)
Y = matrix(nrow = m, ncol = n)
gamma = sample(1:4, m, replace = TRUE, prob = pi)
alpha = vector()
beta = vector()
vec1 = rnorm(m, 0.05, 1)
vec2 = rnorm(m, -0.5, 2)
alpha = ((gamma==2) + (gamma == 4))*vec1
beta = ((gamma ==3) + (gamma == 4))*vec2
alpha_hat = vector()
beta_hat = vector()
var_alpha = c()
var_beta = c()
p1 = vector()
p2 = vector()
for(i in 1:m)
{
  M[i,] = alpha[i]*X + rnorm(n)
  Y[i,] = beta[i]*M[i,]  + rnorm(1,0.5)*X + rnorm(n)
  obj1 = lm(M[i,] ~  X )
  obj2 = lm(Y[i,] ~  M[i,] + X)
  table1 = coef(summary(obj1))
  table2 = coef(summary(obj2))
```

```
  alpha_hat[i] = table1["X",1]
  beta_hat[i] = table2["M[i, ]",1]
  p1[i] = table1["X",4]
  p2[i] = table2["M[i, ]",4]
  var_alpha[i] = table1["X",2]^2
  var_beta[i] = table2["M[i, ]",2]^2
}
lfdr <- localFDR(alpha_hat, beta_hat, var_alpha, var_beta, twostep = FALSE)
print(lfdr$pi)
MLFDR(lfdr$lfdr, size = 0.05)
```

Index

localFDR, 2

MLFDR, 4