Package ‘BFMediate’

July 10, 2020

Type Package

Title Mediation Analysis Using a Combination of Bayesian Estimation Methods, Latent Variable Models, and Bayes Factors

Version 0.1.0

Author Who wrote it

Maintainer The package maintainer <yourself@somewhere.net>

Description The focus is on measuring evidence for (full) mediation using Bayes factors. The package covers models with measurement error and discretization in the mediator (M) and/or the dependent variable. For further details, see the paper, Measuring Evidence for Mediation in the Presence of Measurement Error by Laghaie and Otter (2020).

Roxygen list(markdown = TRUE)

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

LinkingTo BH (>= 1.66.0),
Rcpp (>= 0.12.0),
RcppArmadillo,
RcppEigen (>= 0.3.3.3.0),
rstan (>= 2.18.1),
StanHeaders (>= 2.18.0)

Imports methods,
Rcpp (>= 0.12.0),
rstan (>= 2.18.1),
rstantools (>= 2.0.0)

Biarch true

Depends R (>= 3.4.0)

SystemRequirements GNU make

R topics documented:

BFSD *Bayes factor for full mediation*

Description

Computes Bayes factors for the partial mediation model using Savage-Dickey approximation

Usage

```
BFSD(post,prior,burnin)
```

Arguments

post	output from PartialMed or any of measurement models
prior	prior variance of the direct effect
burnin	number of MCMC draws before the posterior is converged, default = R/5

Value

log(BF_01), which is the evidence in favor of the full mediation model (see Laghaie and Otter (2020) for guidelines on how to interpret BF_01)

See Also

For simulating data from simple mediation model see PartialMed

Examples

```
# Estimation
A_M = c(100,100);    # Prior variance for beta_0M, beta_1
A_Y = c(100,100,1)   # Prior variance for beta_0Y, beta_2, beta_3
R = 2000
out = PartialMed(Data=Data, pars = list(A_M=A_M, A_Y=A_Y), R = R)
#Computing Bayes factor
BFPartialMed = exp(BFSD(post = out , prior = A_Y[3], burnin = R/5))
```

MeasurementCont *Sampler for Partial Mediation Model with Multiple Continuous Indicator for the Mediator and/or DV*

Description

Estimates a partial mediation model with multiple categorical indicators for the mediator and the dependent variable using Hamiltonian Monte Carlo (HMC) with Stan

Usage

```
MeasurementCont(Data, Prior, R, burnin)
```

Arguments

Data	list(X, m_tilde, y_tilde)
Prior	list(A_M,A_Y)
R	number of MCMC iterations, default = 10000

Details

Model:

M = beta_0M + Xbeta_1 + U_M (eq.1)
Y = beta_0Y + Mbeta_2 + Xbeta_3 + U_Y (eq.2)

Indicator equations:

m*_1	=	M + U_m*_1
~m_1	=	OrdProbit(m*_1,C_m_1)
m*_2	=	lambda_01 + M + U_m*_2
~m_2	=	OrdProbit(m*_2,C_m_2)
...		
m*_k	=	lambda_0k-1 + M + U_m*_k
~m_k	=	OrdProbit(m*_k,C_m_k)
y*_1	=	M + U_y*_1
~y_1	=	OrdProbit(y*_1,C_y_1)
y*_2	=	tau_01 + M + U_y*_2
~y_2	=	OrdProbit(y*_2,C_y_2)
...		
y*_l	=	tau_0l-1 + M + U_y*_l
~y_l	=	OrdProbit(y*_l,C_y_l)

Argument Details:

`Data = list(X, m_tilde, y_tilde):`

X(N x 1) treatment variable vector

m_tilde(N x M_ind) ediator indicators' matrix

y_tilde(N x Y_ind) dependent variable indicators' matrix

`Prior = list(A_M,A_Y)` ***[optional]*:**

A_M vector of coefficients' prior variances of eq.1, default = rep(100,2)

A_Y vector of coefficients' prior variances of eq.2, default = c(100,100,1)

Value

beta_1(R X 2) matrix of eq.1 coefficients' draws

beta_2(R X 3) matrix of eq.2 coefficients' draws

lambda (M_ind X 2 X R) array of mediator indicator coefficients' draws. Each slice is one draw, where rows represent the indicator equation and columns are the coefficients. All Slope coefficients as well as intercept of the first equation are fixed to 1 and 0 respectively.

ssq_m_star(R X M_ind) Matrix of mediator indicator equations' coefficients' error variance draws

ssq_y_star(R X Y_ind) Matrix of dependent variable indicator equations' coefficients' error variance draws

mu_draw vector of means of MCMC draws of the direct effect (used in BFSD to compute Bayes factor)

var_draw vector of means of MCMC draws of the direct effect (used in BFSD to compute Bayes factor)

Examples

```
library(rstan)
SimMeasurementCont = function( beta_1, beta_2 , lambda, tau, m_ind, y_ind, sigma_M, sigma_m_star,
                               sigma_y, sigma_y_star, N, X) {

 m_star = matrix(double(N*m_ind),ncol = m_ind); y_star = matrix(double(N*y_ind),ncol = y_ind)
 eps_m_star = matrix(double(N*m_ind),ncol = m_ind); eps_y_star = matrix(double(N*y_ind),ncol = y_ind)
  eps_M = rnorm(N)*sigma_M # generate errors for M (independent)
  eps_Y = rnorm(N)*sigma_y # generate errors for y (independent)
  M = beta_1[1] + X*beta_1[2]  + eps_M # generate latent mediator M
  y = beta_2[1] +  M*beta_2[2] + X*beta_2[3] + eps_Y # generate dependent variable

  eps_m_star[,1]=rnorm(N)*sigma_m_star[1] # generate errors for m_star (independent)
  m_star[,1] =  M + eps_m_star[,1] # generate observed mediator indicators m_star
  if(m_ind>1){
    for(i in 2:(m_ind))   {
      eps_m_star[,i]=rnorm(N)*sigma_m_star[i] # generate errors for m_star (independent)
     m_star[,i] =  lambda[(i-1),1] + M*lambda[(i-1),2] + X*beta_2[-c(1,2)] + eps_m_star[,i]
    }
  }

  eps_y_star[,1]=rnorm(N)*sigma_y_star[1] # generate errors for y_star (independent)
  y_star[,1] =  y + eps_y_star[,1] # generate observed dependent variable indicators y_star
  if(y_ind>1){
    for(i in 2:(y_ind)){
      eps_y_star[,i]=rnorm(N)*sigma_y_star[i] # generate errors for y_star (independent)
      y_star[,i] =  tau[(i-1),1] + y*tau[(i-1),2] + eps_y_star[,i]
    }
  }
  list(X = X, M = M, m_star = m_star, y = y, y_star=y_star)
}
```

```
m_ind = 2; y_ind = 2;
sigma_M = 1^.5 # error std M
sigma_y = 1^.5 # error std y
sigma_m_star = c(.3,.5)^.5 #c(1,2)^.5
sigma_y_star = c(.5,.3)^.5  #c(2,1)^.5
beta_1 = c(1,1)
beta_2 = c(1,3,0)
lambda = matrix(c(1,1.5),ncol=2)
tau = matrix(c(1,2),ncol = 2)
k=length(beta_1)-1
nobs = 1000   # number of observations
X = runif(nobs) # generate random X from a uniform distribution
Data = SimMeasurementCont( beta_1, beta_2 , lambda, tau, m_ind, y_ind, sigma_M, sigma_m_star,
                          sigma_y, sigma_y_star, nobs, X)
R = 5000; burnin = 3000

A_M=rep(100,2);
A_Y=c(100,100,1)

#Estimation
out = MeasurementCont(Data = Data, Prior = list(A_M = A_M, A_Y = A_Y),R=5000, burnin = 3000)

#Results
colMeans(out$beta_1)
colMeans(out$beta_2)

BFMeasurementCont = exp(BFSD(post = out , prior = A_Y[3],burnin = 0))
```

`MeasurementMCat` *Sampler for Partial Mediation Model with Multiple Categorical Indicator for the Mediator*

Description

Estimates a partial mediation model with multiple categorical indicator for the mediator and observed dependent variable using a mixture of Metropolis-Hastings and Gibbs sampling

Usage

```
MeasurementMCat(Data,Prior,R)
```

Arguments

`Data`	list(X, m_star, Y)
`Prior`	list(A_M,A_Y)
`R`	number of MCMC iterations, default = 10000

Details

Model:

M = beta_0M + Xbeta_1 + U_M [eq.1]\
Y = beta_0Y + Mbeta_2 + Xbeta_3 + U_Y [eq.2]

Indicator equations:

```
m*_1  =  M + U_m*_1
~m_1  =  OrdProbit(m*_1,C_m_1)
m*_2  =  lambda_01 + M + U_m*_2
~m_2  =  OrdProbit(m*_2,C_m_2)
...
m*_k  =  lambda_0k-1 + M + U_m*_k
~m_k  =  OrdProbit(m*_k,C_m_k)
```

Argument Details:

`Data = list(X, m_star, Y)`**:**

X(N x 1) treatment variable vector

m_star(N x M_ind) mediator indicators' matrix

Y(N x 1) dependent variable vector

`Prior = list(A_M,A_Y)` **[optional]:**

A_M vector of coefficients' prior variances of eq.1, default = rep(100,2)

A_Y vector of coefficients' prior variances of eq.2, default = c(100,100,1)

Value

beta_1(R X 2) matrix of eq.1 coefficients' draws

beta_2(R X 3) matrix of eq.2 coefficients' draws

lambda (M_ind X 2 X R) array of indicator coefficients' draws. Each slice is one draw, where rows represent the indicator equation and columns are the coefficients. All Slope coefficients as well as intercept of the first equation are fixed to 1 and 0 respectively.

ssq_m_star(R X M_ind) Matrix of indicator equations' coefficients' error variance draws

ssq_Y(R X 1) vector of eq.2 error variance draws

mu_draw vector of means of MCMC draws of the direct effect (used in BFSD to compute Bayes factor)

var_draw vector of means of MCMC draws of the direct effect (used in BFSD to compute Bayes factor)

Examples

```
SimMeasurementMCat = function(X, beta_1, cutoff_M, beta_2, Sigma_Y, M_ind, lambda, ssq_m_star){

  nobs = dim(X)[1]
  m_star = m_tilde = matrix(double(nobs*M_ind), ncol = M_ind)

 M = beta_1[1] + beta_1[2] * X + rnorm(nobs)  #cbind(rep(1,nobs),X)%*%beta_1 + rnorm(nobs)

  for(i in 1: M_ind){
    m_star[,i] = lambda[i] + M + sqrt(ssq_m_star[i])*rnorm(nobs);
   m_tilde[,i] = cut(m_star[,i], br = cutoff_M[i,], right=TRUE, include.lowest = TRUE, labels = FALSE)
  }

  Y = beta_2[1] + beta_2[2] * M + beta_2[3] * X + rnorm(nobs)*Sigma_Y
```

```
                                    #cbind(rep(1,nobs),cbind(M,X))%*%beta_2 + rnorm(nobs)

  return(list(Y = Y, M = M, m_tilde = m_tilde, X = X,
              beta_1 = beta_1, beta_2 = beta_2,
           lambda = lambda, ssq_m_star = ssq_m_star, m_star = m_star, cutoff_M = cutoff_M,
              k_M=dim(cutoff_M)[2]-1, M_ind=M_ind))
}

M_ind = 2
Mcut =  8
nobs= 500
X=as.matrix(runif(nobs,min=0, max=1))
beta_1 = c(.5,1)
beta_2 = c(1, 2, 0)
Sigma_Y = 1^.5
ssq_m_star = c(.5,.7)
lambda = c(0,-.5)   #the intercepts for the latent M indicators w. measurement
                    #error (first intercept should always be 0)

cutoff_M = matrix(c(-100, 0, 1.6, 2, 2.2, 3.3, 6,  100,
                    -100, 0, 1, 2, 3, 4, 5, 100) ,ncol= Mcut, byrow = T)
DataMCat = SimMeasurementMCat(X, beta_1, cutoff_M, beta_2, Sigma_Y, M_ind, lambda, ssq_m_star)

#estimation
Mcut = max(DataMCat$m_tilde) +1
Data = list(X=cbind(rep(1,length(DataMCat$X)),DataMCat$X), m_tilde=as.matrix(DataMCat$m_tilde),
            Y= as.matrix(DataMCat$Y) ,k=Mcut-1, M_ind=dim(DataMCat$m_tilde)[2])
out = MeasurementMCat(Data=Data, R=R)


#results
colMeans(out$beta_1)
colMeans(out$beta_2)
apply(out$cutoff_M,c(1,2),FUN = mean)
```

`MeasurementMYCat` *Sampler for Partial Mediation Model with Multiple Categorical Indicator for the Mediator and DV*

Description

Estimates a partial mediation model with multiple categorical indicator for the mediator and the dependent variable using a mixture of Metropolis-Hastings and Gibbs sampling

Usage

```
MeasurementMYCat(Data,Prior,R)
```

Arguments

`Data`	list(X, m_tilde, y_tilde)
`Prior`	list(A_M,A_Y)
`R`	number of MCMC iterations, default = 10000

Details

Model:

M = beta_0M + Xbeta_1 + U_M [eq.1]
Y = beta_0Y + Mbeta_2 + Xbeta_3 + U_Y [eq.2]

Indicator equations:

m*_1	=	M + U_m*_1
~m_1	=	OrdProbit(m*_1,C_m_1)
m*_2	=	lambda_01 + M + U_m*_2
~m_2	=	OrdProbit(m*_2,C_m_2)
...		
m*_k	=	lambda_0k-1 + M + U_m*_k
~m_k	=	OrdProbit(m*_k,C_m_k)
y*_1	=	M + U_y*_1
~y_1	=	OrdProbit(y*_1,C_y_1)
y*_2	=	tau_01 + M + U_y*_2
~y_2	=	OrdProbit(y*_2,C_y_2)
...		
y*_l	=	tau_0l-1 + M + U_y*_l
~y_l	=	OrdProbit(y*_l,C_y_l)

Argument Details:

`Data = list(X, m_tilde, y_tilde)`**:**

X(N x 1) treatment variable vector

m_tilde(N x M_ind) mediator indicators' matrix

y_tilde(N x Y_ind) dependent variable indicators' matrix

`Prior = list(A_M,A_Y)` **[optional]:**

A_M vector of coefficients' prior variances of eq.1, default = rep(100,2)

A_Y vector of coefficients' prior variances of eq.2, default = c(100,100,1)

Value

beta_1(R X 2) matrix of eq.1 coefficients' draws

beta_2(R X 3) matrix of eq.2 coefficients' draws

lambda (M_ind X 2 X R) array of mediator indicator coefficients' draws. Each slice is one draw, where rows represent the indicator equation and columns are the coefficients. All Slope coefficients as well as intercept of the first equation are fixed to 1 and 0 respectively.

ssq_m_star(R X M_ind) Matrix of mediator indicator equations' coefficients' error variance draws

ssq_y_star(R X Y_ind) Matrix of dependent variable indicator equations' coefficients' error variance draws

mu_draw vector of means of MCMC draws of the direct effect (used in BFSD to compute Bayes factor)

var_draw vector of means of MCMC draws of the direct effect (used in BFSD to compute Bayes factor)

Examples

```
SimMeasurementMYCat = function(X, beta_1, cutoff_M, beta_2, cutoff_Y, M_ind, Y_ind,
                               lambda, tau, ssq_m_star, ssq_y_star){

  nobs = dim(X)[1]
  m_tilde = m_star = matrix(double(nobs*M_ind), ncol = M_ind)
  y_tilde = y_star = matrix(double(nobs*Y_ind), ncol = Y_ind)

  M = cbind(rep(1,nobs),X)%*%beta_1 + rnorm(nobs)
  for(i in 1: M_ind){
    m_star[,i] = lambda[i] + M + sqrt(ssq_m_star[i])*rnorm(nobs);
   m_tilde[,i] = cut(m_star[,i], br = cutoff_M[i,], right=TRUE, include.lowest = TRUE, labels = FALSE)
  }

  Y = cbind(rep(1,nobs),cbind(M,X))%*%beta_2 + rnorm(nobs)
  for(i in 1: Y_ind){
    y_star[,i] = tau[i] + Y + sqrt(ssq_y_star[i])*rnorm(nobs);
   y_tilde[,i] = cut(y_star[,i], br = cutoff_Y[i,], right=TRUE, include.lowest = TRUE, labels = FALSE)
  }

  return(list(Y = Y, M = M, y_tilde = y_tilde, m_tilde = m_tilde, X = X,
              k_M=dim(cutoff_M)[2]-1, beta_1 = beta_1, lambda = lambda,
              ssq_m_star = ssq_m_star, m_star = m_star, cutoff_M = cutoff_M,
             k_Y=dim(cutoff_Y)[2]-1, beta_2 = beta_2, tau = tau, ssq_y_star = ssq_y_star,
              y_star = y_star, cutoff_Y = cutoff_Y, M_ind=M_ind, Y_ind=Y_ind))
}

M_ind = 2
Y_ind = 2
Mcut = Ycut = 8
nobs=2000
X=as.matrix(runif(nobs,min=0, max=1))
beta_1 = c(.5,1)
beta_2 = c(.7, 1.5, 0)
ssq_m_star = c(.5,.3)
lambda = c(0,-.5) #the intercepts for the latent M indicators w. measurement error
ssq_y_star = c(.2,.2)
tau = c(0,-.5)
cutoff_M = matrix(c(-100, 0, 1.6, 2, 2.2, 3.3, 6,  100,
                    -100, 0, 1, 2, 3, 4, 5, 100) ,ncol= Mcut, byrow = T)
cutoff_Y =  matrix(c(-100, 0, 1.6, 2, 2.2, 3.3, 6,  100,
                     -100, 0, 1, 2, 3, 4, 5, 100) ,ncol= Ycut, byrow = T)
DataMYCat = SimMeasurementMYCat(X, beta_1, cutoff_M, beta_2, cutoff_Y, M_ind,
                                Y_ind, lambda, tau, ssq_m_star, ssq_y_star)


#estimation
Mcut = max(DataMYCat$m_tilde) + 1
Ycut = max(DataMYCat$y_tilde) + 1
Data = list(X=cbind(rep(1,length(DataMYCat$X)),DataMYCat$X), m_tilde=as.matrix(DataMYCat$m_tilde),
            y_tilde=as.matrix(DataMYCat$y_tilde), k_M = Mcut-1, k_Y=Ycut-1,
        M_ind=dim(as.matrix(DataMYCat$m_tilde))[2], Y_ind=dim(as.matrix(DataMYCat$y_tilde))[2])
out = MeasurementMYCat(Data=Data,R = 10000)

#results
colMeans(out$beta_1)
```

```
colMeans(out$beta_2)
apply(out$lambdadraw,MARGIN = c(1,2),FUN = mean)
apply(out$taudraw,MARGIN = c(1,2),FUN = mean)

apply(out$cutoff_M,c(1,2),FUN = mean)
apply(out$cutoff_Y,c(1,2),FUN = mean)
```

`MeasurementYCat` *Sampler for Partial Mediation Model with Multiple Categorical Indicator for the DV*

Description

Estimates a partial mediation model with multiple categorical indicator for the dependent variable

Usage

```
MeasurementYCat(Data,Prior,R)
```

Arguments

`Data`	list(X, M, y_star)
`Prior`	list(A_M,A_Y)
`R`	number of MCMC iterations, default = 10000

Details

Model:

M = beta_0M + Xbeta_1 + U_M [eq.1]
Y = beta_0Y + Mbeta_2 + Xbeta_3 + U_Y [eq.2]

Indicator equations:

y*_1 = M + U_y*_1
~y_1 = OrdProbit(y*_1,C_y_1)
y*_2 = tau_01 + M + U_y*_2
~y_2 = OrdProbit(y*_2,C_y_2)
...
y*_l = tau_0l-1 + M + U_y*_l
~y_l = OrdProbit(y*_l,C_y_l)

Argument Details:

`Data = list(X, M, y_star)`:

X(N x 1) treatment variable vector

M(N x 1) mediator vector

y_star(N x Y_ind) dependent variable indicators' matrix

`Prior = list(A_M,A_Y)` **[optional]:**

A_M vector of coefficients' prior variances of eq.1, default = rep(100,2)

A_Y vector of coefficients' prior variances of eq.2, default = c(100,100,1)

Value

beta_1(R X 2) matrix of eq.1 coefficients' draws

beta_2(R X 3) matrix of eq.2 coefficients' draws

tau(Y_ind X 2 X R) array of indicator coefficients' draws. Each slice is one draw, where rows represent the indicator equation and columns are the coefficients. All Slope coefficients as well as intercept of the first equation are fixed to 1 and 0 respectively.

ssq_y_star(R X Y_ind) Matrix of indicator equations' coefficients' error variance draws

ssq_M(R X 1) vector of eq.1 error variance draws

mu_draw vector of means of MCMC draws of the direct effect (used in BFSD to compute Bayes factor)

var_draw vector of means of MCMC draws of the direct effect (used in BFSD to compute Bayes factor)

Examples

```
SimMeasurementYCat = function(X, beta_1, beta_2, sigma_M, cutoff_Y, Y_ind, tau, ssq_y_star){

  nobs = dim(X)[1]
  y_tilde = y_star = matrix(double(nobs*Y_ind), ncol = Y_ind)

  M = beta_1[1] + beta_1[2] * X + rnorm(nobs) * sigma_M
  Y = beta_2[1] + beta_2[2] * M + beta_2[3] * X + rnorm(nobs)

  for(i in 1: Y_ind){
    y_star[,i] = tau[i] + Y + sqrt(ssq_y_star[i])*rnorm(nobs);
   y_tilde[,i] = cut(y_star[,i], br = cutoff_Y[i,], right=TRUE, include.lowest = TRUE, labels = FALSE)
  }

  return(list(Y = Y, M = M, y_tilde = y_tilde, X = X,
              beta_1 = beta_1,
              k_Y=dim(cutoff_Y)[2]-1, beta_2 = beta_2, tau = tau,
             ssq_y_star = ssq_y_star, y_star = y_star, cutoff_Y = cutoff_Y,
              Y_ind=Y_ind))
}

Y_ind = 2
Ycut = 8
nobs = 5000
X=as.matrix(runif(nobs,min=0, max=1))
beta_1 = c(.5,1)
beta_2 = c(1, 4, 2)
sigma_M = 1^.5
ssq_y_star = c(.5,.7)
tau = c(0,-.5)   #first intercept should always be 0
cutoff_Y =  matrix(c(-100, 0, 1.6, 2, 2.2, 3.3, 6,  100,
                     -100, 0, 1, 2, 3, 4, 5, 100) ,ncol= Ycut, byrow = T)
DataYCat = SimMeasurementYCat(X, beta_1, beta_2, sigma_M, cutoff_Y, Y_ind, tau, ssq_y_star)
```

```
#estimation
A_M = c(100,100); #Prior variance for beta_0M, beta_1
A_Y = c(100,100,100) #Prior variance for beta_0Y, beta_2, beta_3
Prior = list(A_M = A_M, A_Y = A_Y)
Ycut = max(as.matrix(DataYCat$y_tilde)[,1]) +1
Data = list(X=cbind(rep(1,length(DataYCat$X)),DataYCat$M,DataYCat$X), y = as.matrix(DataYCat$y_tilde),
            k=Ycut-1, Y_ind=dim(as.matrix(DataYCat$y_tilde))[2])
out = MeasurementYCat(Data=Data, Prior=prior, R=10000)

#results
colMeans(out$beta_1)
colMeans(out$beta_2)
apply(out$cutoff_Y,c(1,2),FUN = mean)
```

`Mediate` *Mediation Analysis and Bayes Factor Computation*

Description

Mediation Analysis and Bayes Factor Computation

Usage

```
Mediate(Data, Model, Prior, R, burnin)
```

Arguments

`Data`	list(X, M, Y) for "Simple", list(X, m_star, y_star) for "Cont", list(X, m_tilde, Y) for "MCat", list(X, M, y_tilde) for "YCat", and list(X, m_tilde, y_tilde) for "MYCat"
`Model`	can be either "Simple", "Cont", "MCat", "YCat", "MYCat". In case of Simple, a simple partial mediation is estimated, Baron and Kenny (1986), and Preacher and Hayes (2004) proposed methods are also computed
`Prior`	list(A_M,A_Y)
`R`	number of MCMC iterations, default = 10000
`burnin`	number of MCMC draws before the posterior is converged

Details

Model:

For Data arguments and Models, see

- PartialMed fpr "Simple"
- MeasurementCont for "Cont"
- MeasurementMCat for "MCat"
- MeasurementYCat for "YCat"
- MeasurementMYCat for "MYCat"

`Prior = list(A_M,A_Y)` *[optional]*

`A_M` vector of coefficients' prior variances of eq.1, default = rep(100,2)

`A_Y` vector of coefficients' prior variances of eq.2, default = c(100,100,1)

Value

`BK = list(eq1, eq2, Indirect_se, FullMed)` **(only for "Simple"!):**

eq1 the summary of the eq.1 regression

eq2 the summary of the eq.2 regression

Indirect_se the standard error of the indirect effect a la Sobel(1982)

FullMed the significance test result for the direct effect

`PH = list(Indirect_mean, Indirect_CI, Direct_CI)`**:**

Indirect_mean the bootstrapped mean of the indirect effect

Indirect_CI the bootstrapped 95% confidence interval of the indirect effect

Direct_CI the bootstrapped 95% confidence interval of the direct effect

`list(evidence, Indirect_CI, Direct_CI, BF,...)` **(For all the models):**

evidence the interpretation of the BF in terms of evidence in favor of full mediation according to Kass and Raftery (1995)

Indirect_CI the Bayesian 95% HDI (confidence interval) of the indirect effect

Direct_CI the Bayesian 95% HDI (confidence interval) of the direct effect

BF the Bayes factor(BF_01) of the corresponding model (see Laghaie and Otter (2020))

For the rest of the values, see

- PartialMed fpr "Simple"
- MeasurementCont for "Cont"
- MeasurementMCat for "MCat"
- MeasurementYCat for "YCat"
- MeasurementMYCat for "MYCat"

Examples

```
simPartialMed = function(beta_1,beta_2, sigma_M, sigma_Y,N,X) {
eps_M = rnorm(N)*sigma_M      # generate errors for M (independent)
eps_Y = rnorm(N)*sigma_Y      # generate errors for Y (independent)
M = beta_1[1] + beta_1[2] * X + eps_M # generate latent mediator M
Y = beta_2[1] + beta_2[2] * M + beta_2[3] * X + eps_Y  # generate dependent variable
list(X = X, M = M, Y = Y)
}

# Set up data generating parameters
N = 1000    # number of observations
sigma_M = .2^.5    # error std M
sigma_Y = .2^.5    # error std Y
beta_1 = c(1, .3)   # beta_0M and beta_1
beta_2 = c(1, .5, .01)    # beta_0Y, beta_2, beta_3
X = rnorm(N,mean = 1,sd = 1)   # generate random X
# Generate data based on parameters
Data = simPartialMed(beta_1,beta_2,sigma_M,sigma_Y,N,X)

#Estimation
A_M = c(100,100);  # Prior variance for beta_0M, beta_1
A_Y = c(100,100,1) # Prior variance for beta_0Y, beta_2, beta_3
R = 2000
out = BFMediate(Data = Data, Model = 'Simple',
                Prior = list(A_M = A_M, A_Y = A_Y),R=5000, burnin = 3000)
```

```
# Results
out$BK$FullMed
out$PH$Indirect_CI
colMeans(out$Simple$beta_2)
out$Simple$BF
```

`PartialMed` *Gibbs Sampler for Partial Mediation Model*

Description

Estimates a partial mediation model using series of Gibbs Samplers

Usage

```
PartialMed(Data, pars, R)
```

Arguments

`Data`	list(X, M, Y)
`pars`	list(A_M,A_Y)
`R`	number of MCMC iterations, default = 10000

Details

Argument Details:

`Data = list(X, M, Y)`**:**

X(N x 1) treatment variable vector

M(N x 1) mediator vector

Y(N x 1) dependent variable vector

`pars = list(A_M,A_Y)` **[optional]:**

A_M vector of coefficients' prior variances of eq.1, default = rep(100,2)

A_Y vector of coefficients' prior variances of eq.2, default = c(100,100,1)

Value

a list containing

beta_1(R X 2) matrix of eq.1 coefficients' posterior draws

beta_2(R X 3) matrix of eq.2 coefficients' posterior draws

ssq_M(R X 1) vector of eq.1 error variance posterior draws

ssq_Y(R X 1) vector of eq.2 error variance posterior draws

mu_draw vector of means of MCMC draws of the direct effect (used in BFSD to compute Bayes factor)

var_draw vector of means of MCMC draws of the direct effect (used in BFSD to compute Bayes factor)

Examples

```
simPartialMed = function(beta_1,beta_2, sigma_M, sigma_Y,N,X) {
  eps_M = rnorm(N)*sigma_M # generate errors for M (independent)
  eps_Y = rnorm(N)*sigma_Y # generate errors for Y (independent)
  M = beta_1[1] + beta_1[2] * X + eps_M # generate latent mediator M
  Y = beta_2[1] + beta_2[2] * M + beta_2[3] * X + eps_Y # generate dependent variable
  list(X = X, M = M, Y = Y)
}

# Set up data generating parameters
N = 1000 # number of observations
sigma_M = .2^.5 # error std M
sigma_Y = .2^.5 # error std Y
beta_1 = c(1, .3) # beta_0M and beta_1
beta_2 = c(1, .5, .01) # beta_0Y, beta_2, beta_3
X = rnorm(N,mean = 1,sd = 1)# generate random X
# Generate data based on parameters
Data = simPartialMed(beta_1,beta_2,sigma_M,sigma_Y,N,X)

#Estimation
A_M = c(100,100); #Prior variance for beta_0M, beta_1
A_Y = c(100,100,1) #Prior variance for beta_0Y, beta_2, beta_3
R = 2000
out = PartialMed(Data=Data, pars = list(A_M=A_M, A_Y=A_Y), R = R)

#Computing Bayes factor
BFPartialMed = exp(BFSD(post = out , prior = A_Y[3], burnin = 0))
```

Index