Package 'bmsSum'

August 7, 2020

Type Package

Title Bayesian hierarchical models for combining the summay measures

Version 0.1.0

Authors Yujing, Y., Qixuan, C., Ogden, R.T.

Maintainer Yujing Yao <yy2725@columbia.edu>

Description An implementation of Bayesian hierarchical models with Stan for combining the summay measures from multiple sources. In many situations that arise frequently in meta-analysis and small area estimation, among other fields, one is interested in combining estimates from multiple studies or domains given corresponding measures of the uncertainty of each estimate. In such a case, an efficient inference can be accomplished based on a hierarchical Bayesian model with appropriate priors to account for the between-source variation and within source variation. The univariate hierarchical model is used as a general method when there is no obvious correlation between summary estimates and the corresponding uncertainty of the measures. Bivariate model can be thought of as an improvement to the univariate model in terms of estimation accuracy and efficiency in the situation in which measures and their uncertainty are correlated. See ``Bivariate Hierarchical Bayesian Model for Combining Summary Measures from Multiple Sources" for more information. The main function of this package is unimod() and bimod().

Depends R (>= 4.0.0)

Imports rstan (>= 2.19.3), ggplot2, StanHeaders

License GPL-3

Encoding UTF-8

LazyData true

NeedsCompilation no

RoxygenNote 7.1.1

Repository Github

R topics documented:

bimod *Bivariate Bayestian hierarchical model for combining summaries from multiple sources.*

Description

bimod is modeling the measures and its uncertainty jointly for combining summaries from multiple sources. It returns the estimates and posterior distributions of parameters of interest fitted with data.

Usage

```
bimod(
  y = data$y,
  s = data$s,
  with_cov = FALSE,
  X = NULL,
  bi_option = TRUE,
  seed = 9999,
  iter = 3000,
  chain = 4,
  verbose = FALSE,
  warmup = 1000,
  thin = 3,
  adapt_delta = 0.99,
  max_treedepth = 3,
  ...
)
```

Arguments

y	A n-by-1 vector, consisting of the measures of n observations.
s	A n-by-1 vector, consisting of the standard error of the measures for n observations.
with_cov	if TRUE, include covariates in the model.
X	A n-by-p matrix of p covariates including intercept, which is all 1 for the first column.
bi_option	if TRUE, use empirical value for standard deviation of uncertainty of measure
seed	The seed for random number generation. The default is generated from 1 to the maximum integer supported by R on the machine. Even if multiple chains are used, only one seed is needed, with other chains having seeds derived from that of the first chain to avoid dependent samples.The default is 9999.
iter	a number to specify the iteration times of Stan. The default is 3000.
verbose	TRUE or FALSE: flag indicating whether to print intermediate output from Stan on the console, which might be helpful for model debugging.
warmup	a number of warmup iterations to be excluded when computing the summaries. The default is 1000.
thin	A positive integer specifying the period for saving samples. The default value is 3.
adapt_delta	A parameters to control the sampler's behavior. The default is 0.8.
max_treedepth	A parameters to control the sampler's behavior in For algorithm NUTS.

Value

a list of information of the fitted bivariate Bayestian hierarchical model including

`pop_par` the estimates and credible intervals for the population-level parameters

`ind_par` the estimates and credible intervals for the individual-level parameters

`poster_dist` posterior distributions of all parameters fitted in the model

Examples

```
library(rstan)
options(mc.cores = parallel::detectCores())
rstan_options(auto_write = TRUE)
library(MASS)
set.seed(99999)

### define data generation functions-no predictors
data.nonreg <-  function(n=50,mutheta=5,musigma=2,
                         tautheta=3, tausigma=1, r1=0.3, r2=0.3, sigmas=1,...){
  # mutheta,musigma,tautheta, tausigma -population level
  # r1 - individual level
  # r2 - population level
  theta = vector()
  sigma2 = vector()
  y = vector()
  s2 = vector()
  Sigma = list()
  indiv = matrix(rep(NA),ncol = 2, nrow = n)
  # generate individual true mean and variance from population
  mu <- c(mutheta,musigma)
 varmat <- matrix(c(tautheta^2,r2*tausigma*tautheta,r2*tausigma*tautheta,tausigma^2), nrow=2)
  temp <- MASS::mvrnorm(n, mu, varmat)
  theta <- temp[,1]
  sigma2 <- (exp(temp[,2]))^2
  # generate observed value from true individual
  for (i in 1:n){
   Sigma[[i]] <- c(sigma2[i],r1*sqrt(sigma2[i])*sigmas,r1*sqrt(sigma2[i])*sigmas,sigmas^2)
    indiv[i,] <- mvrnorm(1, temp[i,],matrix(Sigma[[i]], nrow = 2))
  }
  y <- indiv[,1]
  s <- exp(indiv[,2])
  dataset <- data.frame(cbind(y, s))
  return(list(data = dataset,
              para = c(n=n,mutheta=mutheta,musigma=musigma,
                     tautheta=tautheta, tausigma=tausigma, r1=r1, r2=r2, sigmas=sigmas)))
}


data <- data.nonreg(r1=0,r2=0)$data
res <- bimod(data$y, data$s, with_cov = FALSE, bi_option = TRUE,
             seed = 9999, iter = 3000, chain = 3, verbose = FALSE,
             warmup= 1000, thin=3, adapt_delta = 0.99)$pop_par
```

unimod *Univariate Bayestian hierarchical model for combining summaries from multiple sources.*

Description

unimod is mainly modeling the measures for combining summaries from multiple sources. It returns the estimates and posterior distributions of parameters of interest fitted with data.

Usage

```
unimod(
  y = data$y,
  s = data$s,
  with_cov = FALSE,
  X = NULL,
  seed = 9999,
  iter = 3000,
  chain = 4,
  verbose = FALSE,
  warmup = 1000,
  thin = 3,
  adapt_delta = 0.99,
  max_treedepth = 3,
  ...
)
```

Arguments

y	A n-by-1 vector, consisting of the measures of n observations.
s	A n-by-1 vector, consisting of the standard error of the measures for n observations.
with_cov	if TRUE, include covariates in the model.
X	A n-by-p matrix of p covariates including intercept, which is all 1 for the first column.
seed	The seed for random number generation. The default is generated from 1 to the maximum integer supported by R on the machine. Even if multiple chains are used, only one seed is needed, with other chains having seeds derived from that of the first chain to avoid dependent samples.The default is 9999.
iter	a number to specify the iteration times of Stan. The default is 3000.
verbose	TRUE or FALSE: flag indicating whether to print intermediate output from Stan on the console, which might be helpful for model debugging.
warmup	a number of warmup iterations to be excluded when computing the summaries. The default is 1000.
thin	A positive integer specifying the period for saving samples. The default value is 3.
adapt_delta	A parameters to control the sampler's behavior. The default is 0.8.
max_treedepth	A parameters to control the sampler's behavior in For algorithm NUTS.
chains	A positive integer specifying the number of Markov chains. The default is 4.

Value

a list of information of the fitted univariate Bayestian hierarchical model including

`pop_par` the estimates and credible intervals for the population-level parameters

`ind_par` the estimates and credible intervals for the individual-level parameters

`poster_dist` posterior distributions of all parameters fitted in the model

Examples

```
library(rstan)
options(mc.cores = parallel::detectCores())
rstan_options(auto_write = TRUE)
library(MASS)
set.seed(99999)

### define data generation functions-with predictors
m <- 50
X=t(rbind(rep(1,m),rnorm(m,0,1),rbinom(m,1,0.2)))
beta=t(rbind(c(4,3,5),c(1,1,0)))

data.reg <-  function(tautheta=3,tausigma=1,
                      r1=0.3,r2=0.3, sigmas=1,...){
  # r1 - individual level
  # r2 - population level
  n <- dim(X)[1]
  theta = vector()
  sigma2 = vector()
  y = vector()
  s2 = vector()
  Sigma = list()
  temp = matrix(rep(NA),ncol = 2, nrow = n)
  indiv = matrix(rep(NA),ncol = 2, nrow = n)
  # generate individual true mean and variance from population
  mu <- X%*%beta
 varmat <- matrix(c(tautheta^2,r2*tausigma*tautheta,r2*tausigma*tautheta,tausigma^2), nrow=2)
  for (k in 1:n) temp[k,] <- MASS::mvrnorm(1,mu[k,],varmat)
  theta <- temp[,1]
  sigma2 <- (exp(temp[,2]))^2
  # generate observed value from true individual
  for (i in 1:n){
   Sigma[[i]] <- c(sigma2[i],r1*sqrt(sigma2[i])*sigmas,r1*sqrt(sigma2[i])*sigmas,sigmas^2)
    indiv[i,] <- mvrnorm(1, temp[i,],matrix(Sigma[[i]], nrow = 2))
  }
  y <- indiv[,1]
  s <- exp(indiv[,2])
  dataset <- data.frame(cbind(y, s, X))
  return(list(data = dataset,
              para = c(tautheta=tautheta, tausigma=tausigma, r1=r1, r2=r2, sigmas=sigmas)))
}

data <- data.reg(r1=0,r2=0)$data
res <- unimod(data$y, data$s, with_cov = TRUE, X=X,
              seed = 9999, iter = 3000, chain = 3, verbose = FALSE,
              warmup= 1000, thin=3)$pop_par
```

Index