

Accounting for correlated horizontal pleiotropy in two-sample Mendelian randomization using correlated instrumental variants

2020-08-15

Introduction

This vignette provides an introduction to the `MR.Corr2` package. R package `MR.Corr2` implements MR-Corr², Accounting for correlated horizontal pleiotropy in two-sample Mendelian randomization using correlated instrumental variants :

```
library(devtools);  
install_github("anonymous/MR.Corr2");
```

Load the package using the following command:

```
library(MR.Corr2);
```

Fit MR-Corr for the independent SNPs using simulated data

We first generate genotype data using function *genRawGeno*:

```
library("MR.Corr2");  
library("mvtnorm");  
library("ggplot2");  
set.seed(20200718)  
rho = 0; L = 100; M = 1; p = M*L; m = p; Alrate = 0.1;  
n1 = 20000; n2 = 20000; rho_ag = 0.2;  
b0 = 0.1; h2y = 0.1; h2z = 0.1;  
maf = runif(p, 0.05, 0.5);  
  
G = genRawGeno(maf, L, M, rho, n1 + n2);  
G1 = G[1:n1,]; G2 = G[(n1+1):(n1+n2),]; G12 = G[1:(n1+n2),];
```

Generate the exposure data(y) and outcome data(z) with prespecified indirect(h_y^2) and direct(h_z^2) heritability based on

$$\mathbf{y} = \mathbf{G}_1\boldsymbol{\gamma} + \mathbf{U}_x\boldsymbol{\eta}_x + \mathbf{e}_1, \quad \mathbf{z} = \beta_0\mathbf{x} + \mathbf{G}_2\boldsymbol{\alpha} + \mathbf{U}_y\boldsymbol{\eta}_y + \mathbf{e}_2,$$

```
# -----  
  
sigma2g = 1;  
sigma2a = 1;  
  
S_ag = matrix(c(sigma2a, rho_ag, rho_ag, sigma2g), nrow=2);  
AG = rmvnorm(p, mean=rep(0, 2), sigma = S_ag, method="chol")  
alpha = AG[,1]; gamma = AG[,2];  
  
# -----
```

```

q = 50
u = matrix(rnorm( (n1+n2) * q),ncol=q);
Su = matrix(c(1,0.8,0.8,1),nrow=2)
bu = rmvnorm(q,mean=rep(0,2), sigma = Su,method="chol")
by = bu[,1]; bz = bu[,2];
uby = u%%by; ubz = u%%bz;
uby = uby/sqrt(as.numeric(var(uby)/0.6));
ubz = ubz/sqrt(as.numeric(var(ubz)/0.2));

G12g = G12%%gamma;

if(b0!=0){
  h2ga = (h2y * ( 1 + b0^2))/(b0^2 * (1 - h2y));
  gamma0 = gamma/sqrt(as.numeric(var(G12g)/h2ga));
  G12g = G12%%gamma0;
}

yall = G12g + uby + rnorm(n1+n2)*as.numeric(sqrt(1-var(uby)));
# The direct effects on Z
h2yb = var(b0*yall);
h2a1 = (h2z + h2z*h2yb)/(1 - h2z);
if(h2z==0){
  alpha = rep(0, p);
  G12a = G12%%alpha;
}else{
  if(Alrate!=1){
    alno = floor(p*Alrate);
    # sparse setting for pleiotropy
    indxAL = sample(1:p,alno);
    alpha[-indxAL] = 0;
  }
  G12a = G12%%alpha;
  alpha0 = alpha/sqrt(as.numeric(var(G12a)/(h2a1)));
  G12a = G12%%alpha0;
}

# -----

resz = ubz + rnorm(n1+n2)*as.numeric(sqrt(1-var(ubz)));
zall = b0*yall + G12a + resz;
y = yall[1:n1];
z = zall[(n1+1):(n1+n2)];
# -----

```

We then conduct single-variant analysis to obtain the summary statistics.

```

# create summary statistics
gammaSS = fastSigLm(y, G1);
GammaSS = fastSigLm(z, G2);
gammah = gammaSS$coef;
se1 = gammaSS$std;
Gammah = GammaSS$coef;
se2 = GammaSS$std;

```

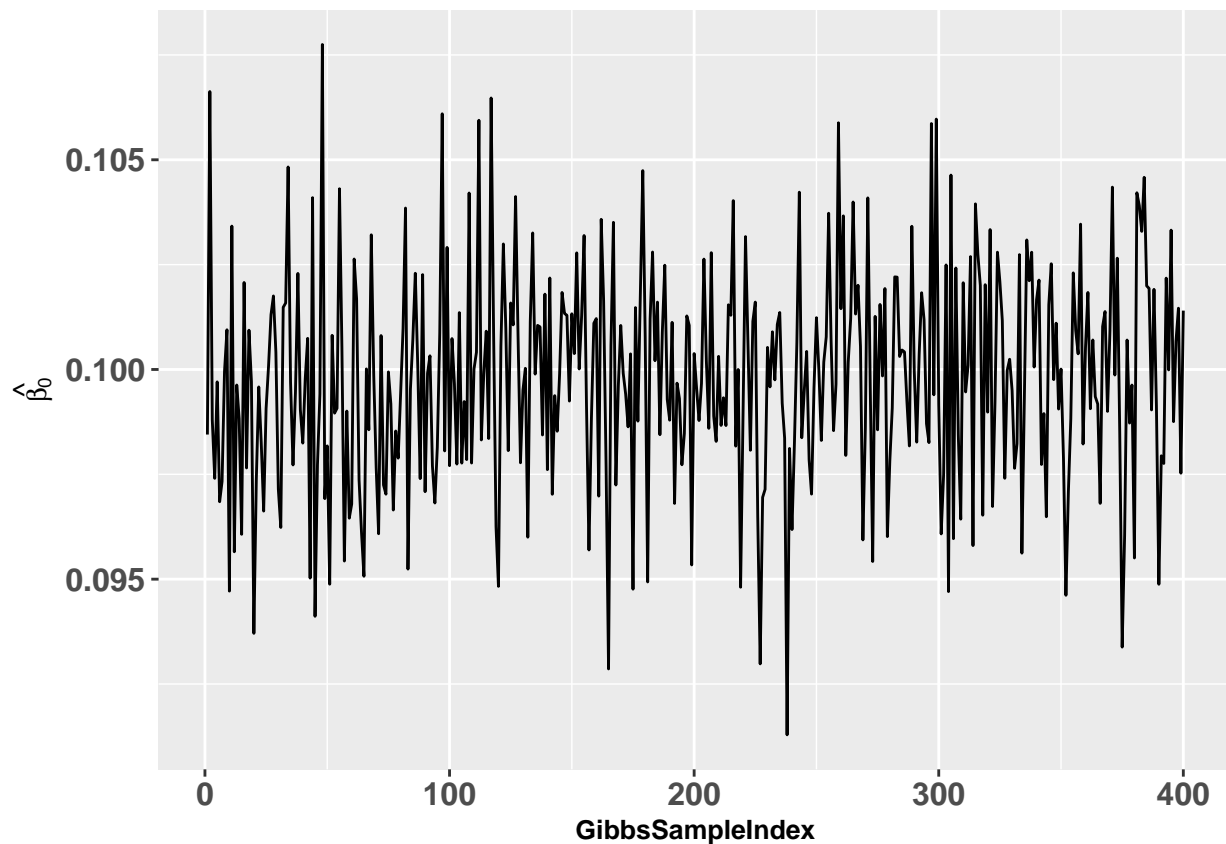
opt list the prior parameters used in MRcorr. $agm = 0.001, bgm = 0.001$ are for σ_γ^2 and $aal = 0.001, bal = 0.001$ are for σ_α^2 . a and b are the prior parameters for ω . burnin is the number of iterations we throw away at the beginning of Gibbs sampling. maxIter is the number of iteration for Gibbs after burnin, the value of interval for recording Gibbs results denoted by thin.

```
opt = list(agm = 0.001, bgm = 0.001, aal = 0.001, bal = 0.001,
          a = 1, b = p, maxIter = 4000, thin = 10, burnin = 1000);
```

```
SimResIndep = MRcorr(gammah, Gammah, se1, se2, opt);
```

Check the convergence of Gibbs sampler using trace plot.

```
traceplot(SimResIndep$Beta0res);
```



One can increase the number of burnin or maxIter if the trace plot did not coverage.

```
beta_hat = mean(SimResIndep$Beta0res);
beta_se = sd(SimResIndep$Beta0res);
beta_pvalue = 2*(1 - pnorm(abs(beta_hat/beta_se)));
EtaIndex = which(SimResIndep$Eta==1);
```

```
cat("The estimated effect of the exposure on outcome: ", beta_hat);
```

```
## The estimated effect of the exposure on outcome: 0.09978347
```

```
cat("Standard error of beta_hat: ", beta_se);
```

```
## Standard error of beta_hat: 0.002567654
```

```
cat("P-value for beta_hat: ", beta_pvalue);
```

```
## P-value for beta_hat: 0
cat("The index of nonzero orthogonal projected pleiotropic effect:", "\n", Eta1index);

## The index of nonzero orthogonal projected pleiotropic effect:
## 6 7 8 17 19 51 58 59 76 91
```

Fit MR-Corr² for the correlated SNPs using simulated data

We first generate genotype data using function *genRawGeno*:

```
rm(list = ls());
library(MR.Corr2);
library("mvtnorm");
library(ggplot2);
set.seed(20200718)
rho = 0.4; L = 100; M = 10; p = M*L; m = p; Alrate = 0.1;
n1 = 20000; n2 = 20000; n3 = 500; lambda = 0.055;
rho_ag = 0.2; b0 = 0.1; h2y = 0.1; h2z = 0.1;
maf = runif(p, 0.05, 0.5);

G = genRawGeno(maf, L, M, rho, n1 + n2 + n3);
G1 = G[1:n1,];
G2 = G[(n1+1):(n1+n2),];
G12 = G[1:(n1+n2),];
G3 = G[(n1+n2+1):(n1+n2+n3),];

nblocks = L;
block_inf <- cbind(seq(1, p, M), seq(M, p, M));
block_inf1 <- block_inf - 1;
R = Cal_block_SimR(block_inf1, G3, lambda)
```

Generate the exposure data(y) and outcome data(z) with prespecified indirect(h_y^2) and direct(h_z^2) heritability based on

$$\mathbf{y} = \mathbf{G}_1\gamma + \mathbf{U}_x\eta_x + \mathbf{e}_1, \quad \mathbf{z} = \beta_0\mathbf{x} + \mathbf{G}_2\alpha + \mathbf{U}_y\eta_y + \mathbf{e}_2,$$

```
# -----
sigma2g = 1;
sigma2a = 1;

S_ag = matrix(c(sigma2a, rho_ag, rho_ag, sigma2g), nrow=2);
AG = rmvnorm(p, mean=rep(0, 2), sigma = S_ag, method="chol")
alpha = AG[,1]; gamma = AG[,2];

# -----
q = 50
u = matrix(rnorm((n1+n2) * q), ncol=q);
Su = matrix(c(1, 0.8, 0.8, 1), nrow=2);
bu = rmvnorm(q, mean=rep(0, 2), sigma = Su, method="chol")
by = bu[,1]; bz = bu[,2];
uby = u%*%by; ubz = u%*%bz;
uby = uby/sqrt(as.numeric(var(uby)/0.6));
```

```

ubz = ubz/sqrt(as.numeric(var(ubz)/0.2));

G12g = G12%%gamma;

if(b0!=0){
  h2ga = (h2y *( 1 + b0^2))/(b0^2 * (1 - h2y));
  gamma0 = gamma/sqrt(as.numeric(var(G12g)/h2ga));
  G12g = G12%%gamma0;
}

yall = G12g + uby + rnorm(n1+n2)*as.numeric(sqrt(1-var(uby)));
# -----
# The direct effects on Z
h2yb = var(b0*yall);
h2al = (h2z + h2z*h2yb)/(1 - h2z);

if(h2z==0){
  alpha = rep(0, p);
  G12a = G12%%alpha;
}else{
  if(Alrate!=1){
    alno = floor(L*Alrate);
    ind = sample(1:L,alno);
    if(length(ind)==1){
      indxAL = block_inf[ind, 1]:block_inf[ind, 2]
      alpha[-indxAL] = 0;
    }else{
      indxAL = NULL;
      for(i in 1:length(ind)){
        tmp = block_inf[ind[i], 1]:block_inf[ind[i], 2]
        indxAL = append(indxAL, tmp);
      }
      alpha[-indxAL] = 0;
      G12a = G12%%alpha;
      alpha0 = alpha/sqrt(as.numeric(var(G12a)/(h2al)));
      G12a = G12%%alpha0;
    }
  }
}

# -----
resz = ubz + rnorm(n1+n2)*as.numeric(sqrt(1-var(ubz)));
zall = b0*yall + G12a + resz;

y = yall[1:n1];
z = zall[(n1+1):(n1+n2)];

G1 = G12[1:n1, ];
G2 = G12[(n1+1):(n1+n2), ]

```

We then conduct single-variant analysis to obtain the summary statistics.

```
gammaSS = fastSigLm(y, G1);
GammaSS = fastSigLm(z, G2);
gammah = gammaSS$coef;
se1 = gammaSS$std;
Gammah = GammaSS$coef;
se2 = GammaSS$std;
```

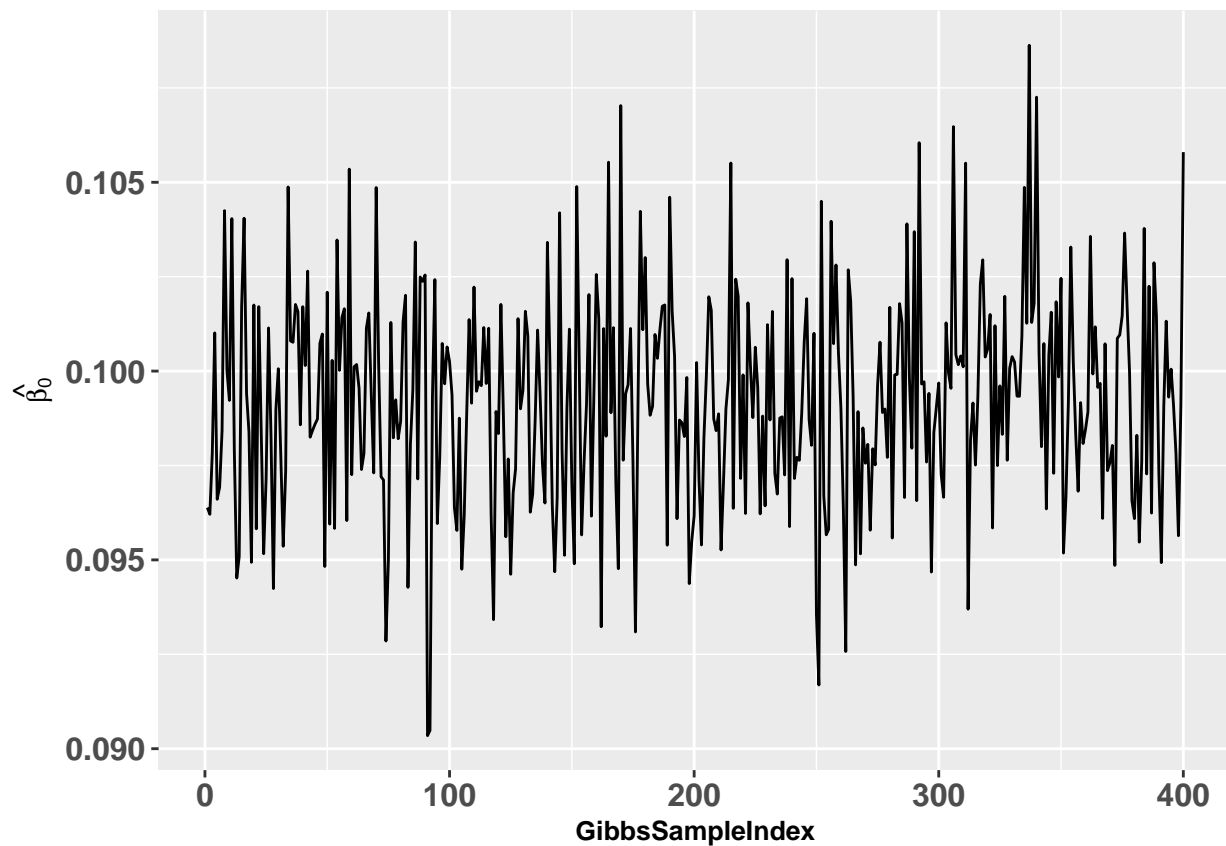
opt list the prior parameters used in MR-Corr². coreNum is the number of cores in your CPU. block_inf1 is the block information for the correlation matrix.

```
coreNum = 20;
opt = list(agm = 0.001, bgm = 0.001, aal = 0.001, bal = 0.001,
          a = 1, b = L, maxIter = 4000, thin = 10, burnin = 1000);

SimRes = MRCorr2Sim(gammah, Gammah, se1, se2, R, block_inf1, coreNum, opt);
```

Check the convergence of Gibbs sampler using trace plot.

```
traceplot(SimRes$Beta0res);
```



One can increase the number of burnin or maxIter if the trace plot did not coverage.

```
beta_hat = mean(SimRes$Beta0res);
beta_se = sd(SimRes$Beta0res);
beta_pvalue = 2*(1 - pnorm(abs(beta_hat/beta_se)));
EtaIndex = which(SimRes$Eta==1);

cat("The estimated effect of the exposure on outcome: ", beta_hat);

## The estimated effect of the exposure on outcome: 0.09921187
```

```
cat("Standard error of beta_hat: ", beta_se);

## Standard error of beta_hat: 0.002828516

cat("P-value for beta_hat: ", beta_pvalue);

## P-value for beta_hat: 0

cat("The index of nonzero orthogonal projected pleiotropic effect:", "\n", Eta1index);

## The index of nonzero orthogonal projected pleiotropic effect:
## 11 17 19 30 31 64 66 67 68 79 88 96 99
```

We are able to use the following code to compare the true signal of orthogonal projected pleiotropic effect (ind) and estimated signal index.

```
ind%in%Eta1index;

## [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

Fit MR-Corr² using BMI-T2D study.

Furthermore, we give an example to illustrate the implements of MR-Corr² for real data analysis. The following datasets(BBJ.BMImatch1KG.txt, BMICAuseSummaryStat.txt, T2DcauseSummaryStat.txt, all_chr_1000G.bed, all_chr_1000G.fam, all_chr_1000G.bim, fourier_ls-all.bed) should be prepared. Download here(<https://zenodo.org/record/3986509#.Xze3yxMzaL8>).

```
filescreen= "BBJ.BMImatch1KG.txt";
fileexposure = "BMICAuseSummaryStat.txt";
fileoutcome = "T2DcauseSummaryStat.txt";
stringname3 = "all_chr_1000G";
block_file = "fourier_ls-all.bed"
```

filescreen, fileexposure, fileoutcome are the datasets names for screen, exposure and outcome, respectively. These three datasets must have the following format(note that it must be tab delimited):

| SNP | chr | BP | A1 | A2 | beta | se | pvalue |
|------------|-----|-----------|----|----|-----------|----------|---------|
| rs1000000 | 12 | 126890980 | A | G | 0.013210 | 0.005182 | 0.01081 |
| rs10000010 | 4 | 21618674 | C | T | -0.003465 | 0.004022 | 0.38900 |
| rs1000002 | 3 | 183635768 | T | C | 0.002628 | 0.003550 | 0.45910 |
| rs10000023 | 4 | 95733906 | T | G | -0.005581 | 0.003614 | 0.12250 |
| rs1000003 | 3 | 98342907 | G | A | -0.004571 | 0.004665 | 0.32720 |

stringname3 is the name of reference panel data. Here we use samples from 1000 Genome Project Phase 1 which is in plink binary format. block_file is used to partition the whole genome into blocks.

matchscreen function is used to match the three datasets with a cutoff for filescreen named pva_cutoff.

```
library(MR.Corr2);
pva_cutoff = 1e-4;
scrres = matchscreen(filescreen, fileexposure, fileoutcome, stringname3, pva_cutoff);
bh1 = as.numeric(scrres$bh1);
bh2 = as.numeric(scrres$bh2);
s12 = as.numeric(scrres$s12);
s22 = as.numeric(scrres$s22);
chr = as.numeric(scrres$chr);
```

```
bp = scrres$bp;
rsname = scrres$rsname
avbIndex = scrres$idxin;
idx4panel = scrres$idx4panel;
```

bh1 and s12 are the SNP effects and corresponding standard errors on the exposure variable, bh2 and s22 are the SNP effects and corresponding standard errors on the outcome variable. After matching the three datasets, we obtain chr (chromosome number) , bp (base position),rsname(rs number).avbIndex(location) and idx4panel (Indicators to be adjusted in reference panel data).

One can using the following function *summaryQC* to remove the MHC region(QCindex = 1), or skip the procedure(QCindex = 0).

```
QCindex = 1;
if(QCindex){
  QCresult = summaryQC(mhcstart, mhcend, bh1, bh2, s12, s22, bp,
    chr, rsname, avbIndex, idx4panel, Inf, Inf)
  bh1new = QCresult$bh1new;
  bh2new = QCresult$bh2new;
  s12new = QCresult$s12new;
  s22new = QCresult$s22new;
  bpnew = QCresult$bpnew;
  chrnew = QCresult$chrnew;
  avbIndexnew = QCresult$avbIndexnew;
  idx4panelnew = QCresult$idx4panel
  rsnamenew = QCresult$rsnamenew;
}else{
  bh1new = bh1;
  bh2new = bh2;
  s12new = s12;
  s22new = s22;
  bpnew = bp;
  chrnew = chr;
  rsnamenew = rsname;
  idx4panelnew = idx4panel;
  avbIndexnew = avbIndex;
}
p = length(avbIndexnew);
```

We use the following function to calculate the number of blocks for correlation matrix.

```
# block information
infres = Cal_blockinf(bpnew, chrnew, block_file);
nblocks = infres$nblocks;
```

In real data analysis, we adopted an alternative computational cost saving methodology to obtain the correlation matrix, here, we fixed the shrinkage factor $\lambda_1 = 0.9$. `ld_r2_thresh` is the threshold for pruning LD. Since MR-Corr² make full use of correlated instrumental variants, hence don't need to prune LD. In other words, the value of `ld_r2_thresh` has no impact on the result of MR-Corr² .

```
lambda1 = 0.9; ld_r2_thresh = 1; coreNum = 24;

opt = list(agm = 0.001, bgm = 0.001, aal = 0.001, bal = 0.001,
  a = 1, b = nblocks, maxIter = 5000, thin = 10, burnin = 5000);
```

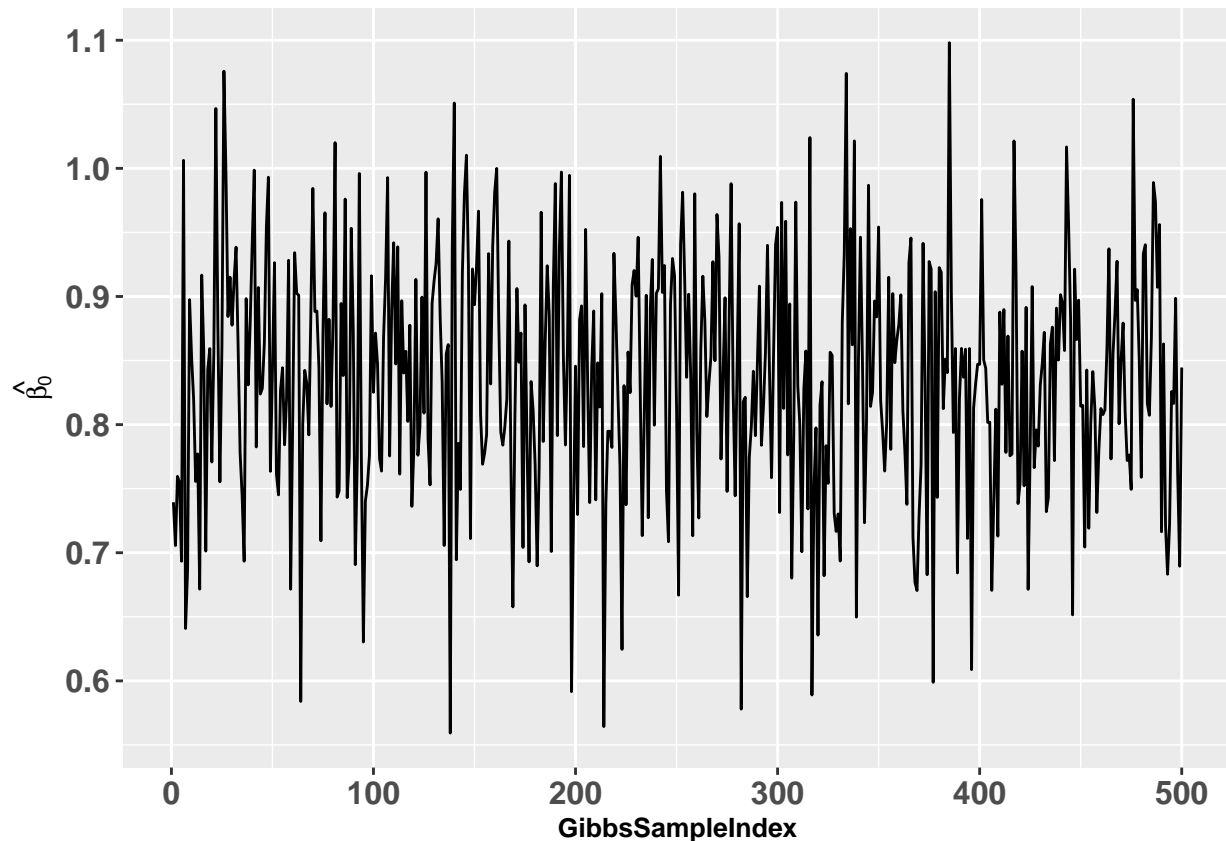


```
RealRes = MRCorr2Real(bpnew, chrnew, avbIndexnew-1,
                      idx4panelnew, block_file, stringname3,
                      ld_r2_thresh, bh1new, bh2new, s12new,
                      s22new, lambda1, coreNum, opt)

bhat = RealRes$BetaOres
```

Check the convergence of Gibbs sampler using trace plot.

```
library(ggplot2)
traceplot(bhat)
```



One can increase the number of burnin or maxIter if the trace plot did not coverage.

```
beta_hat = mean(bhat);
beta_se = sd(bhat);
beta_pvalue = 2*(1 - pnorm(abs(beta_hat/beta_se)));
EtaIndex = which(RealRes$Eta==1);
Indpid = RealRes$Indpid + 1;
```

Report the results of MR-Corr². Indpid is index of the near-independent SNPs , if we set a appropriate threshold value.

```
cat("The estimated effect of the exposure on outcome: ", beta_hat);
```

```
## The estimated effect of the exposure on outcome: 0.8356899
```

```
cat("Standard error of beta_hat: ", beta_se);
```

```
## Standard error of beta_hat: 0.09331269
```

```
cat("P-value for beta_hat: ", beta_pvalue);
```

```
## P-value for beta_hat: 0
```