Mendelian randomization informs shared genetic etiology underlying exposure and outcome by interrogating correlated horizontal pleiotropy

Qing Cheng, Xiao Zhang, Lin S. Chen and Jin Liu

Introduction

This vignette provides an introduction to the *MR.CUE* package. R package *MR.CUE* implements MR-CUE for Mendelian randomization informs shared genetic etiology underlying exposure and outcome by interrogating correlated horizontal pleiotropy.

Install the development version of *MR.CUE* by use of the 'devtools' package. Note that *MR.CUE* depends on the 'Rcpp' and 'RcppArmadillo' package, which also requires appropriate setting of Rtools and Xcode for Windows and Mac OS/X, respectively. This package now depends on R (>= 3.5.0).

To install this package, run the following command in R

```
library(devtools)
install_github("QingCheng0218/MR.CUE@main")
```

If you are encountering issues installing *MR.CUE* on macOS, such as 'Clang error', it had something to do with the C++ compiler, please see the guide here to setup the macOS toolchain(https://thecoatlessprofessor.com/programming/cpp/r-compiler-tools-for-rcpp-on-macos/).

If you have errors installing this package on Linux, try the following commands in R:

```
Sys.setlocale(category = "LC_ALL", locale = "English_United States.1252")
Sys.setenv(LANG="en_US.UTF-8")
Sys.setenv(LC_ALL="en_US.UTF-8")

library(devtools)
install_github("QingCheng0218/MR.CUE@main")
```

Load the package using the following command:

```
library(MR.CUE);
```

This vignette depends on R packages *mvtnorm* and *ggplot2*, you can load these two packages using the following command:

```
library("mvtnorm");
library("ggplot2");
```

Fit MR-CUE for correlated SNPs using simulated data

In this section, we fit MR-CUE using simulated data. It took about 30 seconds to estimate causal effect using *MR.CUE* on a Linux platform with a 2.60 GHz intel Xeon CPU E5-2690 v3 with 30720 KB cache and 96 GB RAM.

We first generate genotype data using function *genRawGeno*:

```
h2g <- 0.1; h2a <- 0.1; h2t <- 0.05; b1 <- 0.1;
rho <- 0.4; rho_ag = 0.2; L <- 50; M <- 10; Alrate <- 0.1;
n1 = 50000; n2 = 50000; n3 = 4000; lam = 0.85;

p = M*L;
block_inf <- cbind(seq(1, p, M), seq(M, p, M));
block_inf1 <- block_inf - 1; # note that C++ array index starts from 0.
coreNum = 20;


maf = runif(p,0.05,0.5);
x = genRawGeno(maf, L, M, rho, n1 + n2 + n3);
x12 = x[1:(n1+n2),];
x3 = x[(n1+n2+1):(n1+n2+n3),];
R = Cal_block_SimR(block_inf1, x3, lam);
```

We use the following function *genSumStat* to generate the summary statistics.

```
SumStatres <- genSumStat(x12, n1, n2, M, L, b1, rho_ag, Alrate, h2a, h2t, h2g);
gammah = SumStatres$gammah;
se1 = SumStatres$se1;
Gammah = SumStatres$Gammah;
se2 = SumStatres$se2;
CHPindexTrue = SumStatres$CHPindex;
```

```
rho = 0 # for two independent samples.
resMRCUE = MRCUESim(gammah, Gammah, se1, se2, rho, R, block_inf1, coreNum);
```

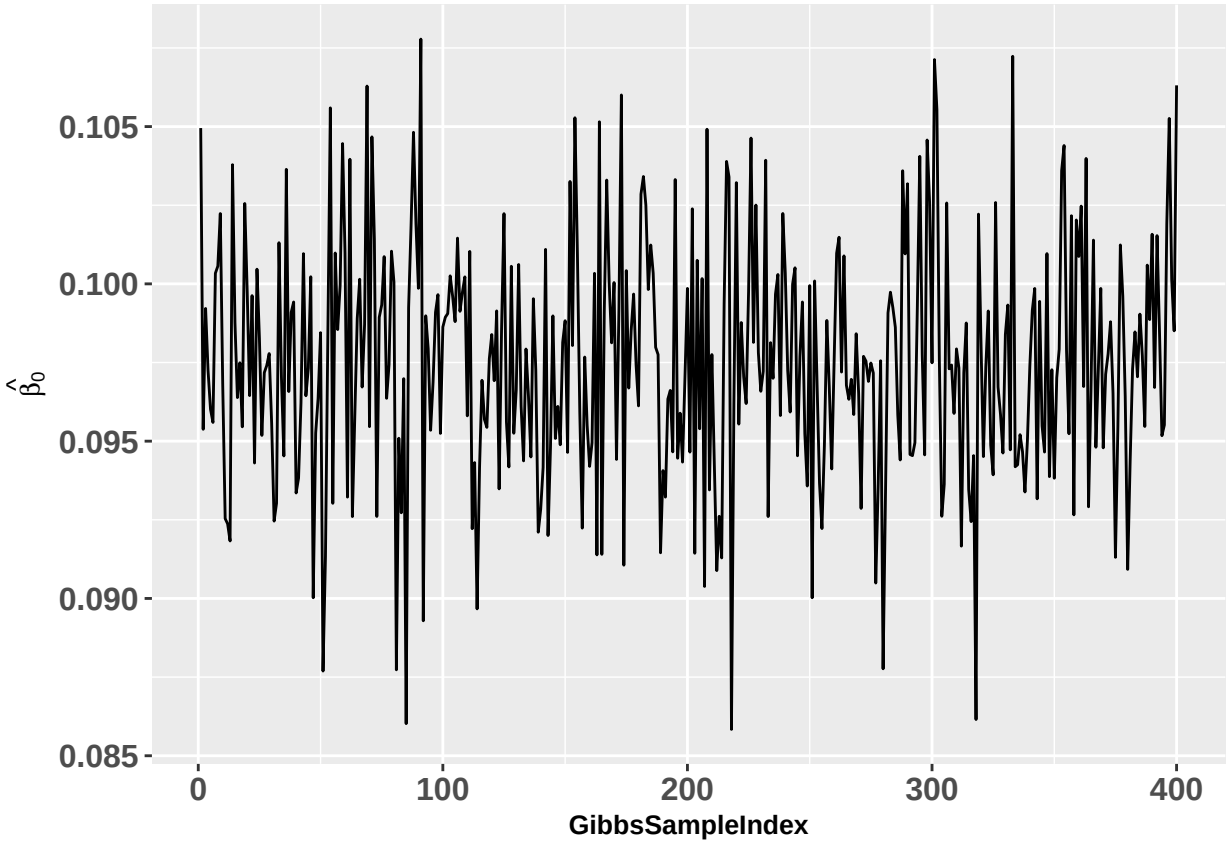
You can use the following code to change the default prior parameters for MR-CUE.

```
opt = list(agm = 0, bgm = 0, atau1 = 0, btau1 = 0,
           atau2 = 0, btau2 = 0,
           a = 2, b = L, CluDes = "PropMajor",  maxIter = 4000, thin = 10, burnin = 1000);
resMRCUE = MRCUESim(gammah, Gammah, se1, se2, 0, R, block_inf1, coreNum, opt);
```

Specificallly, $agm = 0$, $bgm = 0$ are for σ_γ^2 and $atau1 = 0$, $btau1 = 0$ are for τ_1^2, $atau2 = 0$, $btau2 = 0$ are for τ_2^2. a and b are the prior parameters for ω. The *CluDes* parameter is a string used to determine the identification of clusters with valid IVs and clusters with invalid IVs. The default setting is *'PropMajor'*, based on the assumption that only a sparse proportion of IVs exhibit correlated pleiotropic effects, while the major proportion of IVs corresponds to valid IVs that can be used to estimate β_1. Users also have the option to select the *'VarMajor'* setting, where a larger variance of Γ represents the clusters with valid IVs for estimating β_1. *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*.

Check the convergence of Gibbs sampler using trace plot.

```
traceplot(resMRCUE$Beta1res);
```



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

```
MRCUEbeta = resMRCUE$beta.hat;
MRCUEse = resMRCUE$beta.se;
MRCUEpvalue = resMRCUE$beta.p.value;
```

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

```
## The estimated effect of the exposure on outcome:  0.09752904
```

```
cat("Standard error of beta1: ", MRCUEse);
```

```
## Standard error of beta1:  0.003797362
```

```
cat("P-value for beta1: ", MRCUEpvalue);
```

```
## P-value for beta1:  1.793298e-145
```

Find the CHP index.

```
idxCHP = which(resMRCUE$EtaIterRate>0.8);
```

```
cat("The index of estimated CHP:", idxCHP, "\n");
```

```
## The index of estimated CHP: 1 4 21 45 49
```

```
cat("The index of true CHP:", sort(CHPindexTrue), "\n");
```

```
## The index of true CHP: 1 4 21 45 49
```

In addition, one can use the following code to fit MR-CUE for independent SNPs.

```
resMRCUE = MRCUEIndep(gammah, Gammah, se1, se2, rho = 0)
```

Fit MR-CUE using BMI-T2D study.

We give an example to illustrate the implements of MR-CUE for real data analysis. For Body Mass Index (BMI), we use summary statistics on chromosome 1, chromosome 2 and chromosome 3 from Locke et al. (2015)[PMID:25673413]. For Type 2 Diabetes (T2D) we use summary statistics on chromosome 1, chromosome 2 and chromosome 3 from Mahajan et al. (2018)[PMID:30297969]. The following datasets(`BMICHR1CHR2CHR3.txt`, `T2DCHR1CHR2CHR3.txt`, `UK10KCHR1LDhm3.RDS`, `UK10KCHR2LDhm3.RDS`, `UK10KCHR3LDhm3.RDS`, `UK10Ksnpinforhm3.RDS`) should be prepared, you can download here (https://drive.google.com/drive/folders/1eeXxxp2H4Nv9EJJNUh9_l22-zOYgEvl3?usp=sharing). All coordinates were relative to the hg19 version of the human genome.

```
rm(list = ls());
library(MR.CUE);
library(ggplot2);
filepan <- vector("list", 22);
NumChr = 3;
for(i in 1:NumChr){
  filepan[[i]] <- paste0("UK10KCHR", i, "LDhm3.RDS");
}

fileexp = "BMICHR1CHR2CHR3.txt";
fileout = "T2DCHR1CHR2CHR3.txt";
snpinfo = "UK10Ksnpinforhm3.RDS";
```

`fileexp`, `fileout` are the data sets names for exposure and outcome, respectively. These two data sets must have the following format showd in Table 1, note that it must be tab delimited.

Table 1: Data format used for exposure and outcome data.

SNP	chr	BP	A1	A2	beta	se	pvalue
rs1000050	1	162736463	C	T	0.0002	0.0054	0.9705
rs1000073	1	157255396	A	G	0.0007	0.0038	0.8538
rs1000075	1	95166832	T	C	-0.0028	0.0040	0.4839
rs1000085	1	66857915	C	G	0.0020	0.0044	0.6494
rs1000127	1	63432716	C	T	-0.0019	0.0042	0.6510

`filepen` contians LD information about all of chromosome 22 variants, see Table 2. The column named r indicates the correlation between SNP1 and SNP2 estimated from the reference panel data: UK10K Project [Avon Longitudinal Study of Parents and Children (ALSPAC); TwinsUK] merged with 1000 Genome Project Phase 3 (3,757 samples with 989,932 SNPs). Here we just demonstrate use of *MR.CUE* on three chromosomes, you can change `NumChr = 22` for the following analysis over all 22 autosomes.

Table 2: Data format used for reference panel.

CHR	BlockID	SNP1	SNP2	r
1	1	rs12562034	rs4040617	-0.12635628
1	1	rs12562034	rs2980300	-0.13086387
1	1	rs12562034	rs4475691	0.04401255
1	1	rs12562034	rs1806509	0.09783578
1	1	rs12562034	rs7537756	0.03716719

We also need an additional file named `snpinfo` , which is saved from reference panel data, to match the three data sets (exposure, outcome and panel data) and align effect sizes.

Step 0. Estimate ρ_e for the overlape samples.

This step is used to estimate ρ_e for overlap samples. Since ρ_e is estimated using summary statistics among independent variants, we select independent SNPs using the clumping algorithm (r^2 threshold denoted by `ld_r2_thresh`). `pth` is the critical value adapted to the truncated normal distribution in the estimation procedure. `lambda` is the shrinkage turning parameter for LD estimator.

```
ld_r2_thresh = 0.001;
lambad = 0.85;
pth = 1.96;
RhoEst = EstRho(fileexp, fileout, filepan, snpinfo, ld_r2_thresh, lambad, pth);
rho = mean(RhoEst$Rhores);
```

One can fix `rho = 0` and skip this step for two independent samples.

Step 1. Format Data for MR-CUE.

In this step, we will format the data to fit MR-CUE. In details, we use a pre-chosen threshold (`pva_cutoff`) to select the significant IVs, then we match Match the three data sets (exposure, outcome and panel data) and align effect sizes. The process illustrated above can be conducted using the function *ReadSummaryStat*.

```
pva_cutoff = 5e-4;
lambad = 0.85;
data <- ReadSummaryStat(fileexp, fileout, filepan, snpinfo, pva_cutoff, lambad);
F4gammah <- data$ResF4gammah;
F4Gammah <- data$ResF4Gammah;
F4se1 <- data$ResF4se1;
F4se2 <- data$ResF4se2;
F4Rblock <- data$ResF4Rblock;
F4SNPs <- data$ResF4SNPchr
```

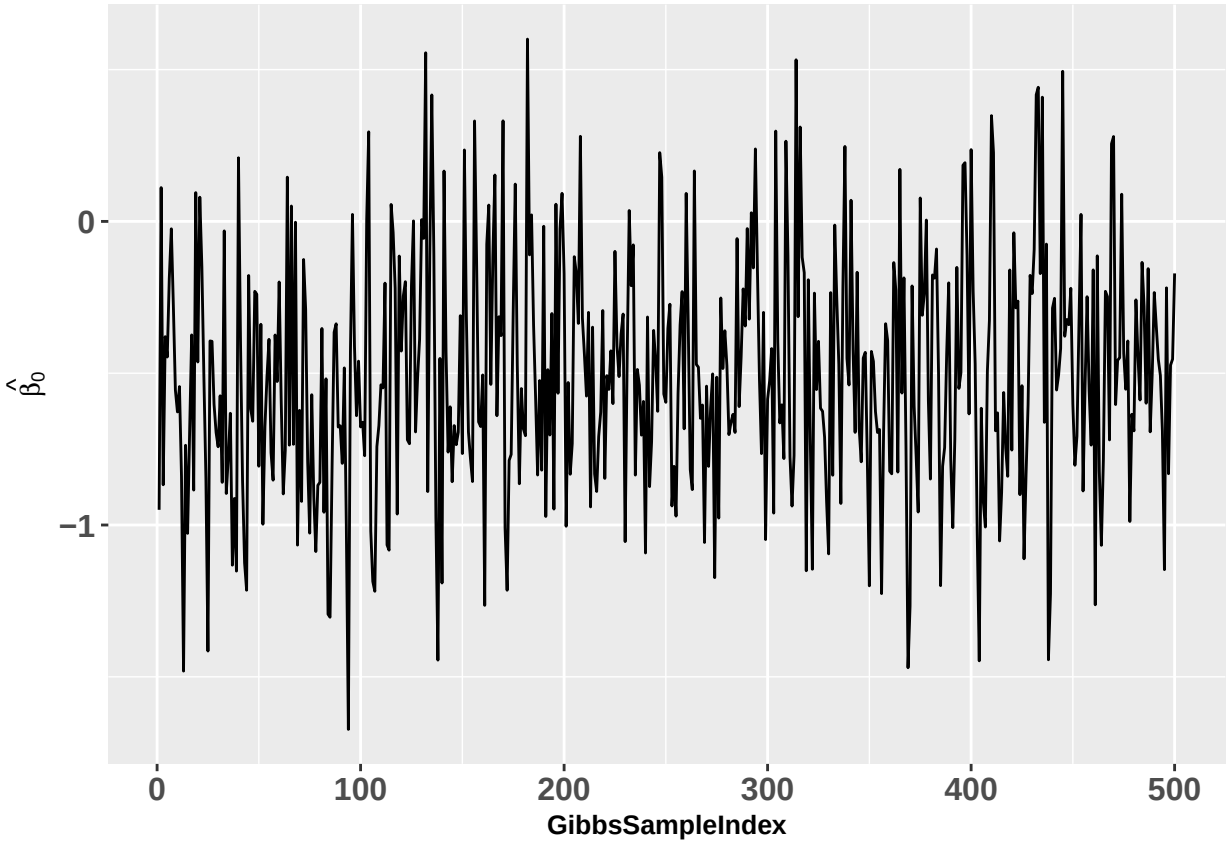
Step 2. Fit MR-CUE

Having given that we have the formatted data, we can fit MR-CUE in this step using the function *MRCUE*. We developed an efficient algorithm with paralleled Gibbs sampling and `coreNum` is the number of cores in your CPU.

```
rho = 0;
L = length(F4gammah);
coreNum = 20;
opt = list(agm = 0, bgm = 0, atau1 = 0, btau1 = 0,
           atau2 = 0, btau2 = 0,
           a = 2, b = L, CluDes = "PropMajor", maxIter = 5000, thin = 10, burnin = 5000);
RealRes = MRCUE(F4gammah, F4Gammah, F4se1, F4se2, F4Rblock, rho, coreNum, opt);
```

Check the convergence of Gibbs sampler using traceplot.

```
traceplot(RealRes$Beta1res);
```




```
MRCUEbeta = RealRes$beta.hat;
MRCUEse = RealRes$beta.se;
MRCUEpvalue = RealRes$beta.p.value;
```

```
cat("The estimated effect of BMI on T2D: ", MRCUEbeta);
```

```
cat("Standard error of beta1: ", MRCUEse);
```

```
cat("P-value for beta1: ", MRCUEpvalue);
```

Step 3. Pathway analysis for CHP.

We save the IVs with significant CHP effects, $\Pr(\eta_l = 1|\text{data}) > 0.8$, we further perform pathway analysis based on those IVs using SNPnexus(https://www.snp-nexus.org/v4/).

```
idxCHP = which(RealRes$EtaIterRate>0.8);

SNPsave = NULL;
num = rep(0, length(idxCHP));
for(i in 1:length(idxCHP)){
  snps = unlist(F4SNPs[idxCHP[i]]);
  SNPsave = append(SNPsave, snps);
  num[i] = length(snps);
}
snpsave = data.frame(rep("dbsnp      ", sum(num)), SNPsave);
write.table(snpsave,  row.names = FALSE, col.names = FALSE,
            quote = FALSE, file = "BMI_T2DCHR1CHR2CHR3CHP.snplist");
```

The saved text file as input set is then submitted to the processing queue on the SNPnexus website. You can download the results as text files or VCF files.

We use the following code to obtain the Reactome Pathways.

```
pathres <- read.table("pathway.txt", sep="\t", header=T, comment.char="#",
                 na.strings=".", stringsAsFactors=FALSE,
                 quote="", fill=FALSE)

pathsig = which(pathres$p.Value<0.05);
pathres$p.Value[pathsig];
as.character(pathres$Parent.s.[pathsig]);
```

As one can see from the result of pathway analysis, some pathways play a central role in the shared etiologies among BMI and T2D such as Metabolism, Neuronal System, and so on. These pathways may shed light on the shared genetic etiology for traits and diseases affected by a common exposure, which can help us better understand the genetic architecture of human complex traits.