Tutorial of ‘A multivariable cis-Mendelian randomization method robust to weak instrument bias and horizontal pleiotropy bias’

Yihe Yang, Noah Lorincz-Comi, Mengxuan Li, Xiaofeng Zhu

Data Structure

Here we illustrate a real example of how to perform cis-MRBEE in real data analysis of the manuscript. This tutorial starts with the structures of involved data.

LD reference panel

The first dataset is the LD reference panel. This dataset is derived from the 9,680 unrelated individuals we described in the paper, selected from approximately 500,000 imputed individuals in the UK Biobank (UKBB). We refined the data using the bim files from these individuals. The files we shared on Google Drive include all 9.32 million SNPs involved; however, in this tutorial, we will focus only on a subset in the ANGPTL3 or CR1 locus. Below is a glimpse of the data structure:

```r
library(data.table)
library(dplyr)
variant=readRDS("RDS/ANGPTL3_variant.rds")%>%as.data.frame(.)
head(variant)
```

```
##               SNP CHR       BP A1 A2      Freq MarkerName
## 1        rs334732   1 61600399  T  C 0.0617230 1:61600399
## 2        rs334731   1 61602342  A  G 0.0441490 1:61602342
## 3 1:61602418_GT_G   1 61602418  G GT 0.9105647 1:61602418
## 4        rs334730   1 61602706  T  C 0.0442680 1:61602706
## 5       rs4915729   1 61602963  G  A 0.9104124 1:61602963
## 6       rs4915730   1 61603158  C  G 0.9104567 1:61603158
```

In this reference panel, `SNP` is the identifier for the variants; `CHR` represents the chromosome; `BP` indicates the base pair position in the hg19 genome build; `A1` is the effect allele as specified in the BED file; `A2` is the other allele; `Freq` denotes the frequency of the effect allele; and `MarkerName` is another unique identifier for the variants in the format CHR:BP. It is important to note that some variants in the UKBB bed file do not have an rsID. For these variants, their `SNP` are in the format CHR:BP:A2:A1.

GWAS and xQTL summary data

The second dataset is the GWAS and xQTL summary data, which should include at least the following columns: `SNP`, `A1`, `A2`, `Zscore`, and `N`. In this dataset, `Zscore` represents the Z-score of the marginal effect size estimates from the outcome GWAS, while `N` denotes the sample size. Other statistics can be deduced from `Zscore` and `N`, e.g.,

$$\mathtt{BETA} = \frac{\mathtt{Zscore}}{\sqrt{\mathtt{N}}}, \quad \mathtt{SE} = \frac{1}{\sqrt{\mathtt{N}}}.$$

Below is an example of the dataset’s structure:

```
datalist=readRDS("RDS/ANGPTL3_datalist.rds")
list2env(datalist,envir=.GlobalEnv)
```

```
## <environment: R_GlobalEnv>
```

```
head(LDL)
```

```
##                 SNP CHR       BP A1  A2     Zscore       N
## 1  1:62010290_CT_C   1 62010290  C  CT  1.3774111 1071241
## 2  1:62020511_AT_A   1 62020511 AT   A -0.5402734  390326
## 3  1:62030258_CA_C   1 62030258 CA   C -0.4779975 1082893
## 4 1:62076847_CAG_C   1 62076847  C CAG  0.2391612 1094352
## 5 1:62079302_GAC_G   1 62079302  G GAC -0.7518729 1091695
## 6  1:62085587_TG_T   1 62085587 TG   T -1.3232658 1072869
```

```
head(ANGPTL3)
```

```
##                 SNP CHR       BP A1  A2     Zscore     N
## 1  1:62010290_CT_C   1 62010290  C  CT -0.2841204 69019
## 2  1:62020511_AT_A   1 62020511 AT   A  0.3600681 69019
## 3  1:62030258_CA_C   1 62030258 CA   C -1.9646074 69019
## 4 1:62076847_CAG_C   1 62076847  C CAG -2.7906215 69020
## 5 1:62079302_GAC_G   1 62079302  G GAC  2.7658935 69020
## 6  1:62085587_TG_T   1 62085587 TG   T  1.4495517 69019
```

It should be noted that we did not use the original `SNP` identifiers from the GWAS. Instead, we merged the GWAS data with the variant file using the `MarkerName` (CHR:BP) identifiers, and then assigned `SNP` from the variant file to the corresponding entries in the GWAS data. In cases where the GWAS file is based on the hg38 genome build, we use LiftOver to convert it to hg19.

LD refernece panel with individual

We used the UKBB BED file to estimate the LD reference, with a sample size of 9,680. Below is a glimpse of the data structure:

```
UKBBGenotype=readRDS("RDS/ANGPTL3_LDref.rds")
UKBBGenotype[1:10,1:5]
```

```
##                 rs4915763 rs6699079 rs12036653 rs6694989 rs6686620
## 1000559-1000559         2         2          1         1         2
## 1000916-1000916         2         1          2         1         1
## 1001097-1001097         1         1          2         1         1
## 1001150-1001150         2         1          1         1         2
## 1001312-1001312         2         1          2         2         2
## 1002233-1002233         2         2          2         2         2
## 1003235-1003235         2         2          2         2         2
## 1004066-1004066         2         2          1         1         2
## 1004469-1004469         2         2          2         2         1
## 1004972-1004972         1         2          2         1         1
```

Note that the data may contains very few missing values. Such missing values are imputed by the means of non-missing values column-by-column.

Step-by-step analysis of ANGPLT3

In the first step, we adjust the direction of the Z-scores in the GWAS and xQTL summary data to ensure that the effect alleles in these datasets match the effect alleles in our reference panel. This step is crucial because the LD matrix is estimated from this reference panel, and accurate LD estimation is fundamental to all statistical methods based on GWAS summary data. We wrote a function, `allele_harmonise()` in the R package `MRBEEX`, to perform this step:

```
library(MRBEEX)
datalist=filter_align(list(LDL=LDL,HDL=HDL,TG=TG,ANGPTL3=ANGPTL3,APOA1=APOA1,
APOC1=APOC1,APOA5=APOA5,APOC3=APOC3,PCSK9=PCSK9),
ref_panel=variant[,c("SNP","A1","A2")],allele_match=T)
```

```
## [1] "Adjusting effect allele according to reference panel..."
## [1] "Finding common SNPs..."
## [1] "Aligning data to common SNPs and ordering..."
## [1] "Filtering complete."
```

```
ZMatrix=matrix(0,dim(datalist[[1]])[1],length(datalist))
NMatrix=matrix(0,dim(datalist[[1]])[1],length(datalist))
for(i in 1:length(datalist)){
ZMatrix[,i]=datalist[[i]]$Zscore
NMatrix[,i]=datalist[[i]]$N
}
rownames(ZMatrix)=rownames(NMatrix)=datalist[[1]]$SNP
colnames(ZMatrix)=colnames(NMatrix)=names(datalist)
head(ZMatrix)
```

```
##                         LDL        HDL          TG    ANGPTL3      APOA1
## 1:62010290_CT_C   1.3774111  0.3679686 -0.81269663 -0.2841204 0.56131199
## 1:62020511_AT_A  -0.5402734  1.0492164 -0.93573985  0.3600681 0.19888614
## 1:62030258_CA_C  -0.4779975 -1.0947681  0.37834035 -1.9646074 0.02897408
## 1:62076847_CAG_C  0.2391612 -0.6978191  0.02555168 -2.7906215 0.17939670
## 1:62079302_GAC_G -0.7518729 -0.6834775 -1.03924477  2.7658935 0.49985624
## 1:62085587_TG_T  -1.3232658  1.9671603 -3.25536420  1.4495517 0.48728617
##                       APOC1      APOA5      APOC3      PCSK9
## 1:62010290_CT_C  -0.7436947  0.2931835  1.1889347 -1.8699032
## 1:62020511_AT_A  -0.1218487  0.3772666  1.6889429 -0.4016323
## 1:62030258_CA_C  -0.7698769 -0.1103482 -0.9983161 -0.3746769
## 1:62076847_CAG_C -0.8783738  0.0812942 -1.2888108 -1.0876133
## 1:62079302_GAC_G -0.2103359 -0.3496257 -0.7876221  1.3662135
## 1:62085587_TG_T  -1.1961568  2.6953615 -0.4245185 -2.1391741
```

```
head(NMatrix)
```

```
##                      LDL     HDL      TG ANGPTL3 APOA1 APOC1 APOA5 APOC3 PCSK9
## 1:62010290_CT_C  1071241 1059148 1092993   69019 33995 69290 35360 35360 69450
## 1:62020511_AT_A   390326  358096  390781   69019 33995 69290 35360 35360 69450
## 1:62030258_CA_C  1082893 1071188 1105099   69019 33995 69290 35360 35360 69450
## 1:62076847_CAG_C 1094352 1082694 1116557   69020 33995 69291 35361 35361 69451
## 1:62079302_GAC_G 1091695 1080104 1113515   69020 33995 69291 35361 35361 69451
## 1:62085587_TG_T  1072869 1061159 1095067   69019 33995 69290 35360 35360 69450
```

In `allele_harmonise()`, we automatically set `gwas_data` as a data.table with `key="SNP"`, allowing `eQTLsQTL=eQTLsQTL[LDL, nomatch=0]` to efficiently merge the two datasets. We aim to find the common variants between the GWAS and xQTL summary data to perform the analysis.

Extracting the moderately correlated variants

The next step is to remove highly correlated variants using C+T. Although SuSiE can group highly correlated or statistically duplicated variants into a single group and assign them one effect, including many redundant variants can significantly increase the dimensionality of the model. Therefore, primarily to enhance computational efficiency, we recommend retaining only moderately correlated variants.

We use the smallest p-value across all exposures corresponding to each variant as the input p-value for PLINK to extract a subset of moderately correlated variants. While we will not execute the following steps in this tutorial, we will provide the code for you. You can modify the file paths as needed for your own data.

```
MinP=apply(ZMatrix^2,1,max)%>%pchisq(.,1,lower.tail=F)
jointtest=data.frame(SNP=rownames(ZMatrix),P=MinP)
write.table(jointtest,"ANGPTL3_Protein.txt",row.names=F,quote=F,sep="\t")
system("./plink --bfile your_bed_file --clump ANGPTL3_Protein.txt --clump-field P
--clump-kb 1000 --clump-p1 5e-5 --clump-p2 5e-5 --clump-r2 0.64
--out ANGPTL3_Protein")
IVlist=fread("ANGPTL3_Protein.clumped")%>%dplyr::select(SNP,CHR,BP,P)
```

The most important part in this step is:

- `-clump-kb 1000`: we consider the window size to be 1M,
- `-clump-p1 5e-5`: we use the threshold of 5E-5,
- `-clump-r2 0.64`: the correlation between two variants is in the range $(-0.8, 0.8)$.

We have recorded this pool of variants in the PCSK9 locus:

```
IVlist=readRDS("RDS/ANGPTL3_IVlist.rds")
```

Regularization of LD matrix

Our next step is to estimate a "good" LD matrix. Note that when the dimension of the LD matrix is large (e.g., $m > 100$), we suggest using the POET method to regularize such an LD matrix. The main idea of POET is as follows. POET considers an eigenvalue decomposition of the sample LD matrix of individuals' genotypes:

$$\hat{\mathbf{R}} = \sum_{k=1}^{m} d_k \mathbf{U}_k \mathbf{U}_k^\top = \sum_{k=1}^{K} d_k \mathbf{U}_k \mathbf{U}_k^\top + \mathbf{E},$$

where $d_1 \geq d_2 \geq \cdots \geq d_m$ are the eigenvalues of $\hat{\mathbf{R}}$, $\mathbf{U}_k$ is the corresponding eigenvector of d_k, $K$is a cutoff, and $\mathbf{E}$is the residual matrix. To improve the condition of $\hat{\mathbf{R}}$, the standard POET applies a covariance-thresholding method on E, while we considered a linear shrinkage of E:

$$\tilde{\mathbf{E}} = \alpha \mathbf{E} + (1 - \alpha)\mathrm{diag}(\mathbf{E}).$$

The extended POET estimate was:

$$\tilde{\mathbf{R}} = \sum_{k=1}^{K} d_k \mathbf{U}_k \mathbf{U}_k^\top + \tilde{\mathbf{E}}.$$

We use $\tilde{\mathbf{R}}$ in the corresponding data.

We utilized the Dynamic Eigenvalue Difference Ratio (DDR, https://ssrn.com/abstract=2827558) to select K:

$$K = \arg \min_{K_{\min} \leq k \leq K_{\max}} \frac{d_k - d_{k+1}}{d_{k+1} - d_{k+2}},$$

where $K_{\min}$ and $K_{\max}$ are two tuning parameters (default to 1 and $\min(100, m/4)$, respectively). On the other hand, we adopted finite sample positive definiteness for the selection of α:

$$\alpha = \inf\{a : \text{minimum eigenvalue of } (a\mathbf{E} + (1-a)\text{diag}(\mathbf{E})) > \tau\},$$

where τ (default to 0.001) was a given tolerance.

The code is as

```
R=cor(UKBBGenotype)
R[is.na(R)]=0;diag(R)=1
R=TGVIS::poet_shrinkage(R)
R=(t(R)+R)/2
genosnp=colnames(UKBBGenotype)
rownames(R)=colnames(R)=genosnp
```

Performing cis-MVMR analysis

First, we organize the data, ensuring that the order of rows in the effect size matrix, SE matrix, and LD matrix must match precisely. It is worth noting that we use genome-wide exposures and outcome summarized statistics to estimate the correlation matrix of estimation errors. For specific examples, please refer to https://github.com/noahlorinczcomi/MRBEE and subsequently the analysis of CR1.

```
ZY=ZMatrix[genosnp,1:3]
ZX=ZMatrix[genosnp,-c(1:3)]
NY=NMatrix[genosnp,1:3]
NX=NMatrix[genosnp,-c(1:3)]
Rxy=readRDS("RDS/Rxy.rds")
```

First, we analyze LDL-C:

```
by=ZY[,"LDL"]/sqrt(NY[,"LDL"])
byse=1/sqrt(NY[,"LDL"])
bX=ZX/sqrt(NX)
bXse=1/sqrt(NX)
NAM=c(colnames(bX),"LDL")
```

Next, we perform cis-MVMR analysis using cis-MRBEE, cis-MVIVW, and PCGMM:

```
library(MendelianRandomization)
MVINPUT=mr_mvinput(bx=bX,by=by,bxse=bXse,byse=byse,correlation=R)
fitMRBEE=MRBEEX::CisMRBEEX(causal.pip.thres=0.2,by,bX,byse,bXse,LD=R,Rxy=Rxy[NAM,NAM],
                           reliability.thres=0.75,xQTL.max.L=15,
                           xQTL.pip.thres=0.3,xQTL.Nvec=colMeans(NX),
                           tauvec=seq(4.5,30,1.5),susie.iter=500,
                           ridge.diff=100,ebic.gamma=0)
```

```
## Please standardize data such that BETA = Zscore/sqrt n and SE = 1/sqrt n
## Sparse prediction ends: 1.815 secs
## Causal effect estimation ends: 1.296 secs
```

```
fitCisIVW=mr_mvivw(MVINPUT,correl=T)
fitPCGMM=mr_mvpcgmm(MVINPUT,nx=colMeans(NX),ny=mean(NY[,"LDL"]),thres=0.999)
ANGPTL3_LDL=list(fitMRBEE=fitMRBEE,fitCisIVW=fitCisIVW,fitPCGMM=fitPCGMM)
```

Finally, we summarize the results. It should be noted that when using BH and BY for adjustments, we set the p-values of variables not selected by SuSiE to 1. This approach yields the most conservative results.

```
LDL=data.frame(
Estimate=c(ANGPTL3_LDL$fitMRBEE$theta,ANGPTL3_LDL$fitCisIVW@Estimate,ANGPTL3_LDL$fitPCGMM@Estimate),
SE=c(ANGPTL3_LDL$fitMRBEE$theta.se,ANGPTL3_LDL$fitCisIVW@StdError,ANGPTL3_LDL$fitPCGMM@StdError),
Exposure=colnames(ZX),
Method=c(rep("CisMRBEE",6),rep("CisMVIVW",6),rep("PCGMM",6)))
LDL$P=pchisq(LDL$Estimate^2/LDL$SE^2,1,lower.tail=F);LDL$P[is.na(LDL$P)]=1
LDL$Outcome="LDL-C"
LDL=dplyr::select(LDL,Outcome,Exposure,Method,Estimate,SE,P)
print(LDL)
```

```
##    Outcome Exposure   Method     Estimate          SE            P
## 1    LDL-C  ANGPTL3 CisMRBEE  0.000000000 0.000000000 1.000000e+00
## 2    LDL-C    APOA1 CisMRBEE  0.348306180 0.016481593 3.952559e-99
## 3    LDL-C    APOC1 CisMRBEE  0.396162626 0.018746498 3.987911e-99
## 4    LDL-C    APOA5 CisMRBEE  0.000000000 0.000000000 1.000000e+00
## 5    LDL-C    APOC3 CisMRBEE  0.000000000 0.000000000 1.000000e+00
## 6    LDL-C    PCSK9 CisMRBEE  0.405058257 0.019167672 4.009485e-99
## 7    LDL-C  ANGPTL3 CisMVIVW  0.048920705 0.007431774 4.621631e-11
## 8    LDL-C    APOA1 CisMVIVW  0.025478782 0.019373117 1.884556e-01
## 9    LDL-C    APOC1 CisMVIVW  0.074030585 0.032068384 2.097009e-02
## 10   LDL-C    APOA5 CisMVIVW -0.027062973 0.019160721 1.578265e-01
## 11   LDL-C    APOC3 CisMVIVW  0.009620662 0.017503576 5.825665e-01
## 12   LDL-C    PCSK9 CisMVIVW  0.075633356 0.022526899 7.865968e-04
## 13   LDL-C  ANGPTL3    PCGMM  0.018697820 0.012144358 1.236505e-01
## 14   LDL-C    APOA1    PCGMM  0.044886348 0.035932152 2.115929e-01
## 15   LDL-C    APOC1    PCGMM  0.132132926 0.058640206 2.424137e-02
## 16   LDL-C    APOA5    PCGMM -0.061735551 0.033504816 6.538928e-02
## 17   LDL-C    APOC3    PCGMM  0.029837444 0.028822053 3.005617e-01
## 18   LDL-C    PCSK9    PCGMM  0.138514352 0.040765501 6.792194e-04
```

The parallel analyses for HDL-C and TG are are as follows:

```
by=ZY[,"HDL"]/sqrt(NY[,"HDL"])
byse=1/sqrt(NY[,"HDL"])
bX=ZX/sqrt(NX)
bXse=1/sqrt(NX)
NAM=c(colnames(bX),"HDL")
MVINPUT=mr_mvinput(bx=bX,by=by,bxse=bXse,byse=byse,correlation=R)
fitMRBEE=MRBEEX::CisMRBEEX(causal.pip.thres=0.2,by,bX,byse,bXse,LD=R,Rxy=Rxy[NAM,NAM],
                           reliability.thres=0.75,xQTL.max.L=15,
                           xQTL.pip.thres=0.3,xQTL.Nvec=colMeans(NX),
                           tauvec=seq(4.5,30,1.5),susie.iter=500,
                           ridge.diff=100,ebic.gamma=0)
```

```
## Please standardize data such that BETA = Zscore/sqrt n and SE = 1/sqrt n
## Sparse prediction ends: 1.782 secs
## Causal effect estimation ends: 1.188 secs
```

```
fitCisIVW=mr_mvivw(MVINPUT,correl=T)
fitPCGMM=mr_mvpcgmm(MVINPUT,nx=colMeans(NX),ny=mean(NY[,"HDL"]),thres=0.999)
ANGPTL3_HDL=list(fitMRBEE=fitMRBEE,fitCisIVW=fitCisIVW,fitPCGMM=fitPCGMM)
HDL=data.frame(
Estimate=c(ANGPTL3_HDL$fitMRBEE$theta,ANGPTL3_HDL$fitCisIVW@Estimate,ANGPTL3_HDL$fitPCGMM@Estimate),
SE=c(ANGPTL3_HDL$fitMRBEE$theta.se,ANGPTL3_HDL$fitCisIVW@StdError,ANGPTL3_HDL$fitPCGMM@StdError),
Exposure=colnames(ZX),
Method=c(rep("CisMRBEE",6),rep("CisMVIVW",6),rep("PCGMM",6)))
HDL$P=pchisq(HDL$Estimate^2/HDL$SE^2,1,lower.tail=F);HDL$P[is.na(HDL$P)]=1
HDL$Outcome="HDL-C"
HDL=dplyr::select(HDL,Outcome,Exposure,Method,Estimate,SE,P)
print(HDL)
```

```
##    Outcome Exposure   Method      Estimate          SE            P
## 1    HDL-C  ANGPTL3 CisMRBEE  0.0000000000 0.000000000 1.000000e+00
## 2    HDL-C    APOA1 CisMRBEE  0.0893070819 0.014104328 2.421727e-10
## 3    HDL-C    APOC1 CisMRBEE  0.1015971754 0.016042453 2.404085e-10
## 4    HDL-C    APOA5 CisMRBEE  0.0000000000 0.000000000 1.000000e+00
## 5    HDL-C    APOC3 CisMRBEE  0.0000000000 0.000000000 1.000000e+00
## 6    HDL-C    PCSK9 CisMRBEE  0.1039225215 0.016402851 2.363618e-10
## 7    HDL-C  ANGPTL3 CisMVIVW  0.0168719040 0.005655826 2.853434e-03
## 8    HDL-C    APOA1 CisMVIVW -0.0001636759 0.014780375 9.911645e-01
## 9    HDL-C    APOC1 CisMVIVW  0.0970515268 0.024452813 7.219578e-05
## 10   HDL-C    APOA5 CisMVIVW  0.0003542207 0.014631370 9.806854e-01
## 11   HDL-C    APOC3 CisMVIVW -0.0249734536 0.013362636 6.163649e-02
## 12   HDL-C    PCSK9 CisMVIVW -0.0225183737 0.017188313 1.901627e-01
## 13   HDL-C  ANGPTL3    PCGMM  0.0432832490 0.012507948 5.392542e-04
## 14   HDL-C    APOA1    PCGMM  0.0376825895 0.041690667 3.660688e-01
## 15   HDL-C    APOC1    PCGMM  0.2357960188 0.067000401 4.326504e-04
## 16   HDL-C    APOA5    PCGMM -0.1151982456 0.040696198 4.644798e-03
## 17   HDL-C    APOC3    PCGMM  0.0468014848 0.034303293 1.724594e-01
## 18   HDL-C    PCSK9    PCGMM -0.3800995652 0.046671606 3.820225e-16
```

```
################################################################################
by=ZY[,"TG"]/sqrt(NY[,"TG"])
byse=1/sqrt(NY[,"TG"])
bX=ZX/sqrt(NX)
bXse=1/sqrt(NX)
NAM=c(colnames(bX),"TG")
MVINPUT=mr_mvinput(bx=bX,by=by,bxse=bXse,byse=byse,correlation=R)
fitMRBEE=MRBEEX::CisMRBEEX(causal.pip.thres=0.2,by,bX,byse,bXse,LD=R,Rxy=Rxy[NAM,NAM],
                           reliability.thres=0.75,xQTL.max.L=15,
                           xQTL.pip.thres=0.3,xQTL.Nvec=colMeans(NX),
                           tauvec=seq(4.5,30,1.5),susie.iter=500,
                           ridge.diff=100,ebic.gamma=0)
```

```
## Please standardize data such that BETA = Zscore/sqrt n and SE = 1/sqrt n
## Sparse prediction ends: 1.759 secs
## Causal effect estimation ends: 1.513 secs
```

```
fitCisIVW=mr_mvivw(MVINPUT,correl=T)
fitPCGMM=mr_mvpcgmm(MVINPUT,nx=colMeans(NX),ny=mean(NY[,"TG"]),thres=0.999)
ANGPTL3_TG=list(fitMRBEE=fitMRBEE,fitCisIVW=fitCisIVW,fitPCGMM=fitPCGMM)
TG=data.frame(
Estimate=c(ANGPTL3_TG$fitMRBEE$theta,ANGPTL3_TG$fitCisIVW@Estimate,ANGPTL3_TG$fitPCGMM@Estimate),
SE=c(ANGPTL3_TG$fitMRBEE$theta.se,ANGPTL3_TG$fitCisIVW@StdError,ANGPTL3_TG$fitPCGMM@StdError),
Exposure=colnames(ZX),
Method=c(rep("CisMRBEE",6),rep("CisMVIVW",6),rep("PCGMM",6)))
TG$P=pchisq(TG$Estimate^2/TG$SE^2,1,lower.tail=F);TG$P[is.na(TG$P)]=1
TG$Outcome="TG"
TG=dplyr::select(TG,Outcome,Exposure,Method,Estimate,SE,P)
print(TG)
```

```
##    Outcome Exposure   Method    Estimate         SE             P
## 1       TG  ANGPTL3 CisMRBEE  0.00000000 0.00000000  1.000000e+00
## 2       TG    APOA1 CisMRBEE  0.60104293 0.02174958 4.263928e-168
## 3       TG    APOC1 CisMRBEE  0.68362412 0.02473844 4.335630e-168
## 4       TG    APOA5 CisMRBEE  0.00000000 0.00000000  1.000000e+00
## 5       TG    APOC3 CisMRBEE  0.00000000 0.00000000  1.000000e+00
## 6       TG    PCSK9 CisMRBEE  0.69895303 0.02529423 4.479239e-168
## 7       TG  ANGPTL3 CisMVIVW  0.09936909 0.01088816  7.085000e-20
## 8       TG    APOA1 CisMVIVW  0.04819148 0.02839766  8.969227e-02
## 9       TG    APOC1 CisMVIVW  0.08455439 0.04700119  7.202123e-02
## 10      TG    APOA5 CisMVIVW -0.09278218 0.02807592  9.508192e-04
## 11      TG    APOC3 CisMVIVW  0.05224150 0.02564776  4.166173e-02
## 12      TG    PCSK9 CisMVIVW  0.08322063 0.03299576  1.166361e-02
## 13      TG  ANGPTL3    PCGMM  0.06592140 0.01780126  2.129067e-04
## 14      TG    APOA1    PCGMM  0.03337197 0.05142052  5.163377e-01
## 15      TG    APOC1    PCGMM  0.16494139 0.08419368  5.010449e-02
## 16      TG    APOA5    PCGMM -0.13517211 0.04742082  4.365326e-03
## 17      TG    APOC3    PCGMM  0.06419960 0.04107333  1.180412e-01
## 18      TG    PCSK9    PCGMM  0.17398527 0.05810955  2.752640e-03
```

Tuning parameter selection

There are a sort of tuning parameters in cis-MRBEE. Here, we discuss some criteria to choose them.

- `causal.pip.thres` determines the minimum PIPs of exposures to calibrate. When multiple exposures are grouped in one credible set, they will separate the PIP and make their individual PIPs smaller. Please use `summary(fitMRBEE$susie.theta)` to check how many exposures are grouped in a credible set and what is the distribution of their individual PIPs, and modify `causal.pip.thres` accordingly.

- `xQTL.max.L` determines the parameter `L` in susie for informative xQTL selection. Cis-MRBEE applies a two stage estimation: it first applies `L = xQTL.max.L` to identify the credible sets and second applies `L  = L*+1` where `L*` is the number of credible set detected in the first stage.

- `xQTL.pip.thres` is used when SuSiE fails to detect any credible set in the informative xQTL selection. In this case, cis-MRBEE uses the variants with individual PIPs larger than this threshold as informative xQTLs.

- Any values of `ebic.gamma` and `ebic.theta` larger than 0 will apply higher penalties on $\boldsymbol{\gamma}$ and $\boldsymbol{\theta}$ than the standard BIC (This is known as extended BIC).

- `ridge.diff` is the penalizing parameter on the discrete differential penalty. Cis-MRBEE is not sensitive to the choice of `ridge.diff`.

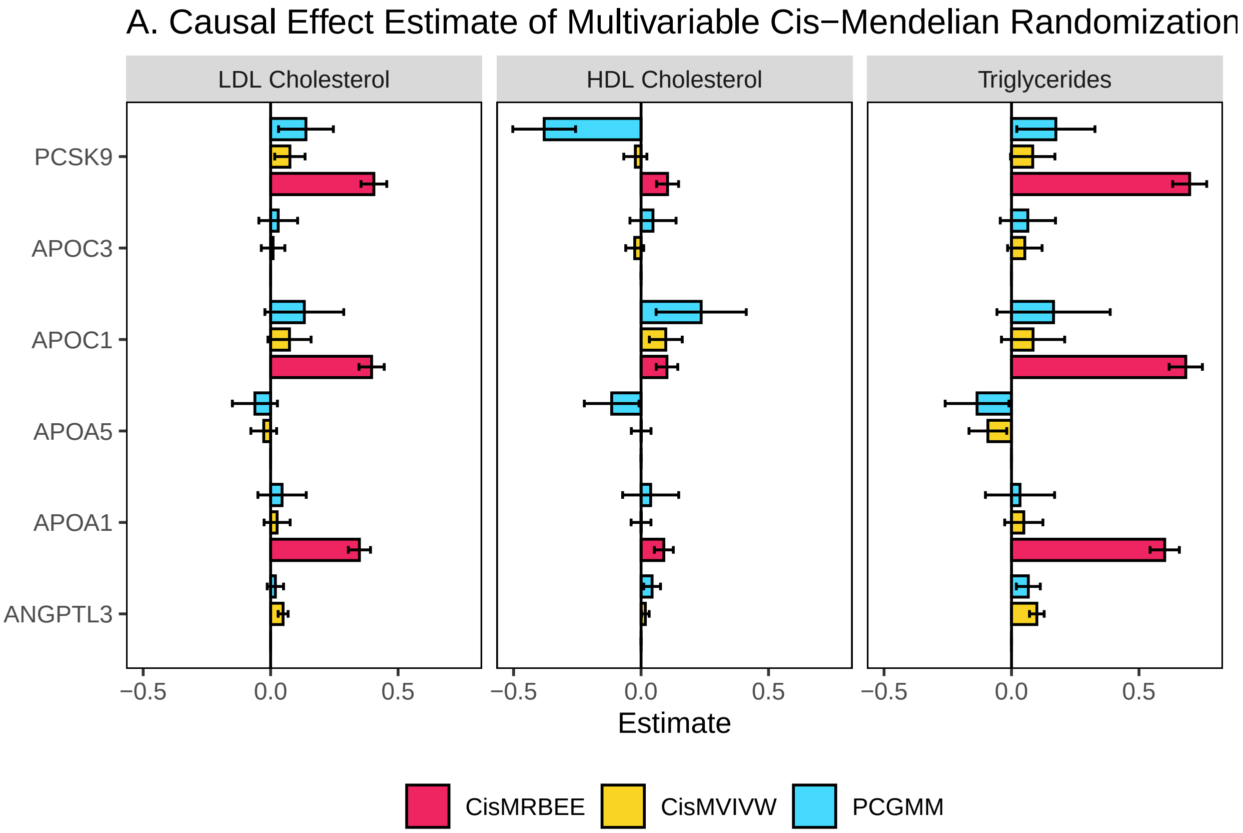
Demonstration of results

Finally, we generate the visualization as shown below. It should be noted that we use the Bonferroni correction to calculate the confidence intervals here: that is, the width of the confidence interval is `\sqrt{\text{qchisq}(0.05/p,1,\text{lower.tail=F})}\times\text{SE}` instead of 2SE:

```
library(ggplot2)
library(corrplot)
```

```
## corrplot 0.95 loaded
```

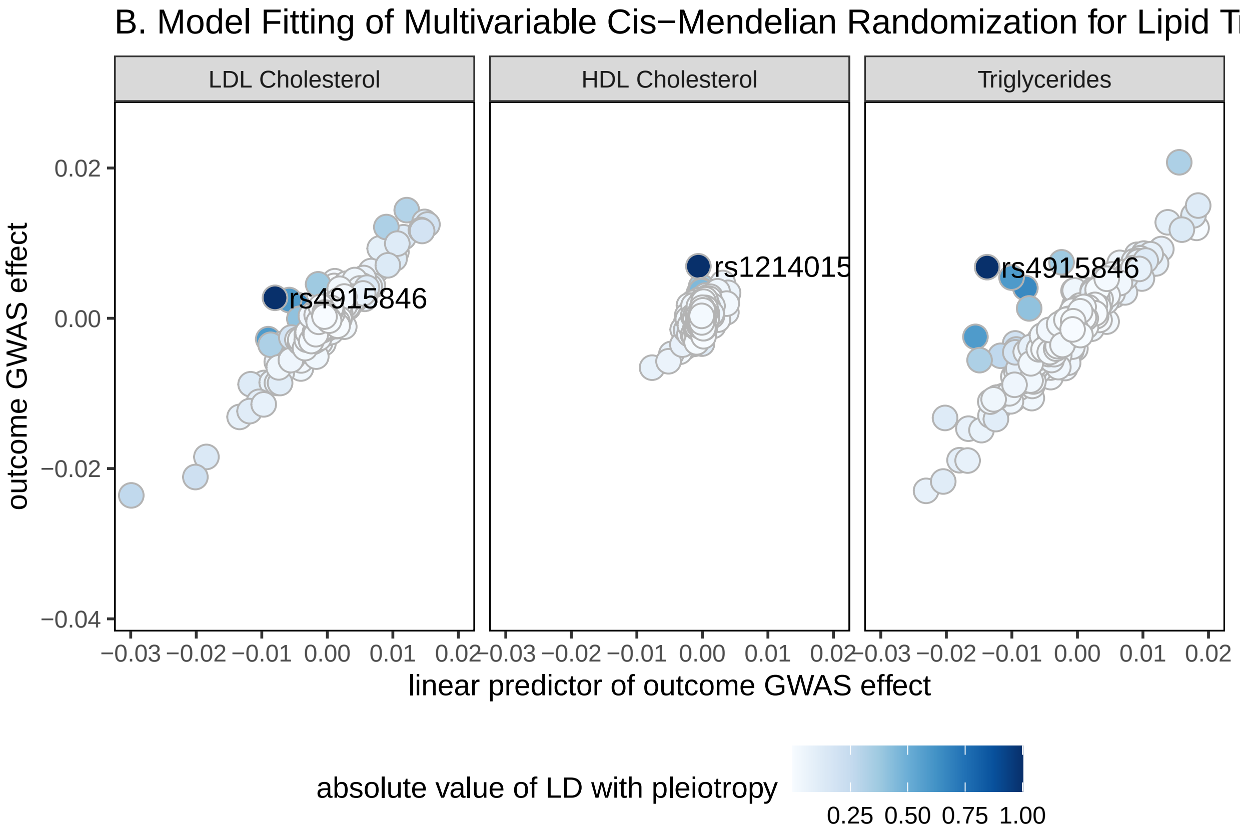
```
Method=c("CisMRBEE","CisMVIVW","PCGMM")
LDLplot=data.frame(by=ZY[,"LDL"]/sqrt(NY[,"LDL"]),
          hatby=ANGPTL3_LDL$fitMRBEE$bXest%*%ANGPTL3_LDL$fitMRBEE$theta,
      pleiotropy=ifelse(ANGPTL3_LDL$fitMRBEE$gamma!=0,
                        names(ANGPTL3_LDL$fitMRBEE$gamma!=0),NA),
      Type="Marginal Effect",
      LD2=R[,which(ANGPTL3_LDL$fitMRBEE$gamma!=0)]^2)
HDLplot=data.frame(by=ZY[,"HDL"]/sqrt(NY[,"HDL"]),
          hatby=ANGPTL3_HDL$fitMRBEE$bXest%*%ANGPTL3_HDL$fitMRBEE$theta,
      pleiotropy=ifelse(ANGPTL3_HDL$fitMRBEE$gamma!=0,
                        names(ANGPTL3_HDL$fitMRBEE$gamma!=0),NA),
      Type="Marginal Effect",
      LD2=R[,which(ANGPTL3_HDL$fitMRBEE$gamma!=0)]^2)
TGplot=data.frame(by=ZY[,"TG"]/sqrt(NY[,"TG"]),
          hatby=ANGPTL3_TG$fitMRBEE$bXest%*%ANGPTL3_TG$fitMRBEE$theta,
      pleiotropy=ifelse(ANGPTL3_TG$fitMRBEE$gamma!=0,
                        names(ANGPTL3_TG$fitMRBEE$gamma!=0),NA),
      Type="Marginal Effect",
      LD2=R[,which(ANGPTL3_TG$fitMRBEE$gamma!=0)]^2)
LDL$Trait="LDL Cholesterol"
HDL$Trait="HDL Cholesterol"
TG$Trait="Triglycerides"
LDLplot$Trait="LDL Cholesterol"
HDLplot$Trait="HDL Cholesterol"
TGplot$Trait="Triglycerides"

DF1=do.call(rbind,list(LDL,HDL,TG))
DF2=do.call(rbind,list(LDLplot,HDLplot,TGplot))
DF1$Trait=ordered(DF1$Trait,levels=c("LDL Cholesterol","HDL Cholesterol","Triglycerides"))
DF2$Trait=ordered(DF2$Trait,levels=c("LDL Cholesterol","HDL Cholesterol","Triglycerides"))
DF1$Method=ordered(DF1$Method,levels=Method)

ggplot(DF1,aes(y=Exposure,x=Estimate,fill=Method)) +
geom_bar(stat="identity",position=position_dodge(width=0.9),width=0.7,color="black") +
geom_errorbar(aes(xmin=Estimate-2.638257*SE,xmax=Estimate+2.638257*SE),
position=position_dodge(width=0.9),width=0.25) +
geom_vline(xintercept=0,linetype="solid",color="black")+
scale_fill_manual(values=c("#ee2560","#f9d423","#45d9fd"))+
facet_grid(~Trait)+
labs(y="causal effect estimate", fill=NULL)+
theme(axis.title.y=element_blank(),
legend.position="bottom",
legend.direction="horizontal",
panel.background=element_blank(),
panel.border=element_rect(colour="black",fill=NA),
panel.grid=element_blank())+
ggtitle("A. Causal Effect Estimate of Multivariable Cis-Mendelian Randomization for Lipid Tratis")
```



A. Causal Effect Estimate of Multivariable Cis−Mendelian Randomization

```r
ggplot(DF2, aes(x = hatby, y = by, fill = sqrt(LD2))) +
geom_point(shape=21,color="grey70",size=4) +
facet_grid(~Trait)+
labs(x = "linear predictor of outcome GWAS effect", y = "outcome GWAS effect", fill ="absolute value of
theme_bw() +
scale_fill_gradient(low="#F7FBFF",high=corrplot::COL1("Blues"))+
theme(legend.position = "bottom",
legend.direction = "horizontal",
panel.background = element_blank(),
panel.border = element_rect(colour = "black", fill = NA),
panel.grid = element_blank())+
scale_x_continuous(limits=c(-0.03,0.02),breaks=seq(-0.03,0.02,0.01))+
ggtitle("B. Model Fitting of Multivariable Cis-Mendelian Randomization for Lipid Tratis")+
guides(size="none")+
geom_text(
data = DF2[!is.na(DF2$pleiotropy), ],
aes(label = pleiotropy),
hjust = -0.1, vjust = 0.5, size = 4, color = "black"
)
```

```
## Warning: Removed 9 rows containing missing values or values outside the scale range
## (`geom_point()`).
```


B. Model Fitting of Multivariable Cis−Mendelian Randomization for Lipid Tr
LDL Cholesterol
HDL Cholesterol
Triglycerides
0.02
0.00
−0.02
−0.04
outcome GWAS effect
rs4915846
rs1214015
rs4915846
−0.03
−0.02
−0.01
0.00
0.01
0.02
linear predictor of outcome GWAS effect
absolute value of LD with pleiotropy
0.25
0.50
0.75
1.00