MR-LDP: a two-sample Mendelian randomization for GWAS summary statistics accounting linkage disequilibrium and horizontal pleiotropy

Qing Cheng and Jin Liu

2019-07-14

Introduction

This vignette provides an introduction to the `MR.LDP` package. R package `MR.LDP` implements MR-LDP, a two-sample Mendelian randomization for GWAS summary statistics accounting linkage disequilibrium and horizontal pleiotropy. The package can be installed with the command:

Fit MR-LDP using simulated data

We first generate genotype data using function *genRawGeno*:

```
library(MR.LDP);
library("mvtnorm");
library("PDSCE");

genRawGeno <- function(maf, L, M, rho, n){
  SIGMA = matrix(nrow=M,ncol=M)
  for (i in 1:M){
    for (j in 1:M){
      SIGMA[i,j] = rho^(abs(i-j));
    }
  }

  nsnp = L*M;
  X = NULL;
  for ( l in 1:L ){
    #set.seed(1000);
    index = (M*(l-1)+1): (M*l);
    AAprob = maf[index]^2.;
    Aaprob = 2*maf[index]*(1-maf[index]);
    quanti = matrix(c(1-Aaprob-AAprob, 1- AAprob),M,2);
    Xt = rmvnorm(n, mean=rep(0,M), sigma=SIGMA, method="chol")
    Xt2 = matrix(0,n,M);
    for (j in 1:M){
      cutoff = qnorm(quanti[j,]);
      Xt2[Xt[,j] < cutoff[1],j] = 0;
      Xt2[Xt[,j] >= cutoff[1] & Xt[,j] < cutoff[2],j] = 1;  ## attention
      Xt2[Xt[,j] >= cutoff[2],j] = 2;
    }
    X <- cbind(X,Xt2);
  }
  return(X)
}
```

```r
rho = 0.4; h2z = 0.05; h2y = 0.1;
L = 1; M = 50; p = M*L; m = p; Alrate = 1;


n1 = 20000; n2 = 20000; n3 = 2500; lam = 0.055;
maf = runif(p, 0.05, 0.5);
x = genRawGeno(maf, L, M, rho, n1 + n2 + n3);
b0 = 0.1;
x1 = x[1:n1,];
x2 = x[(n1+1):(n1+n2),];
x12 = x[1:(n1+n2),];
x3 = x[(n1+n2+1):(n1+n2+n3),];
```

Estimate the covariance matrix using function *pdsoft*:

```r
R0 = cor(x3);
R = pdsoft(R0, lam)$theta;
diag(R) = rep(1, p);
mask = kronecker(diag(L),matrix(1,M,M));
R = R*mask;
```

Generate the exposure data(y) and outcome data(z).

```r
q = 50
u = matrix(rnorm( (n1+n2) * q),ncol=q);

sigma2g = 0.005;
gamma.nz = rnorm(m)*sqrt(sigma2g);
indx = sample(1:p,m);
gamma = numeric(p);
gamma[indx] = gamma.nz;

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));

x12g = x12%*%gamma;

if(b0!=0){
  h2ga = (h2y *( 1 + b0^2))/(b0^2 * (1 - h2y));
  gamma0 = gamma/sqrt(as.numeric(var(x12g)/h2ga));
  x12g = x12%*%gamma0;
}

yall = x12g + 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)

sigma2a <- 0.005;
if(h2z==0){
  alpha0 = rep(0, m);
  x12a = x12%*%alpha0;
}else{
  alno = floor(p*Alrate);
  alpha.nz <- rnorm(alno)*sqrt(sigma2a);
  # sparse setting for pleiotropy
  indxAL = sample(1:p,alno);
  alpha = numeric(p);
  alpha[indxAL] = alpha.nz;

  x12a = x12%*%alpha;
  alpha0 = alpha/sqrt(as.numeric(var(x12a)/(h2al)));
  x12a = x12%*%alpha0;
}
# ----------------------------------------------------------------------
resz = ubz + rnorm(n1+n2)*as.numeric(sqrt(1-var(ubz)));
zall = b0*yall  + x12a +  resz;
H2a.res <- var(x12a)/var(zall);
H2g.res <- var(b0*x12g)/var(zall);

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

create the summary statistics

```r
gammahall = numeric(p); Gammahall = numeric(p);
sg2all = numeric(p); sG2all = numeric(p);
pval = numeric(p);
for (i in 1:p){
  fm = lm(y~1+x1[,i]);
  gammahall[i] = summary(fm)$coefficients[2,1];
  sg2all[i] = summary(fm)$coefficients[2,2]^2;
  pval[i] = summary(fm)$coefficients[2,4];

  fm = lm(z~1+x2[,i]);
  Gammahall[i] = summary(fm)$coefficients[2,1];
  sG2all[i] = summary(fm)$coefficients[2,2]^2;
}

index = 1:p
sig.indx = index

sg2 = sg2all[sig.indx];
sG2 = sG2all[sig.indx];
gammah = gammahall[sig.indx];
Gammah = Gammahall[sig.indx];
```

Initilize the parameters for MR-LDP algorithm.

```r
epsStopLogLik = 1e-7; maxIter = 10000;
beta0 = 0; sgga2 = 1;
agm <- bgm <- aal <- bal <- 0.001;
gamma <- rep(0.01, p);
alpha <- rep(0.01, p);
sgga2 <- 1;
```

```
sgal2 <- 1;
model = 2
```

Fit MR-LDP w/ and w/o constraint that beta = 0 as:

```
resHb = MRLDP_SimPXvb(gammah, Gammah, sqrt(sg2), sqrt(sG2), gamma, alpha,  beta0, sgga2, sgal2, R, cons

resH0 = MRLDP_SimPXvb(gammah, Gammah, sqrt(sg2), sqrt(sG2), gamma, alpha,  beta0, sgga2, sgal2, R, cons

tstat = 2*(resHb$tstat - resH0$tstat);
pval = pchisq(tstat, 1, lower.tail = F);
beta_hat = resHb$beta0;
beta_hat;
```

```
## [1] 0.09796033
```