Simulation

Jacob M. Maronge

1/13/2018

Introduction

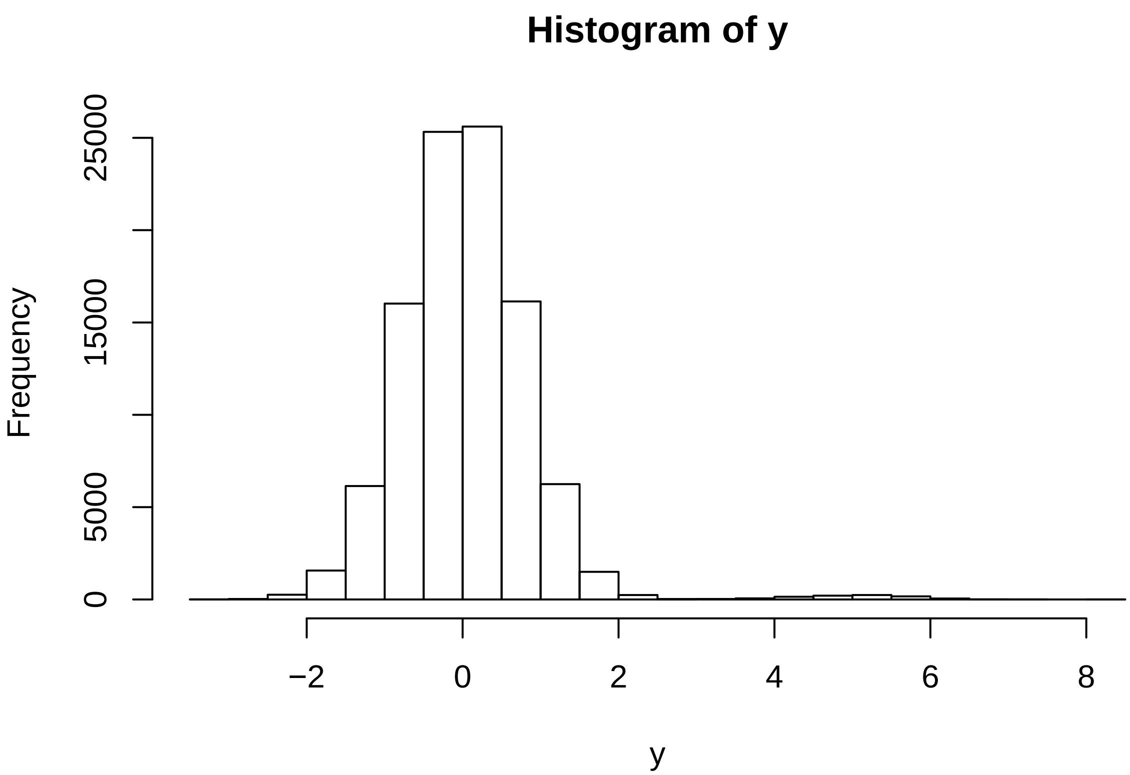
We would like to study if under the setting of outcome depending sampling the conditional likelihood approach is still a valid approach to inference. To study this, I began writing a simulation.

Model

The model I study in my simulation is,

$$Y_{ij} = \beta_0 + \beta X_i + U_i + \epsilon_{ij}.$$

Which is simply a linear model with subject-specific intercept. In my simulation I made X_i a bernoulli random variable with $P(X_i = 1) = 1 - P(X_i = 0) = 0.01$. Where $X_i = 1$ denotes a case and $X_i = 0$ denotes a control. Also, I made X_i fixed within-subject (which may be obvious from notation). However, the errors within-subject are positively correlated with each other. Code for the simulation is shown below, but here is a sample of the histogram of Y values for the population for one repetition



Clearly, there appears to be 2 subpopulations here, where one has many more members than the other. In my simulated data, I make $\beta_0 = 0$ and $\beta = 5$.

Simulation

The goal here is to convince myself that if we sample based off the sum of outcomes for a given subject, (here I made the number of measurements per subject equal to 5) we still get a consistent estimate of for beta. What I did here, was generate a "population" of subjects (number of subjects equals 20,000, each subject has 5 measurements) from the model above. Then I aggregate the data by taking the sum of all measurements for each subject. Then I sample 50 subjects from each group based off that sum. With the sampled data, I fit the model given above and record the estimated value of beta. I repeat this process many times and create a histogram for estimated values for the fixed effect. The code and resulting histogram are shown below.

```
library(nlme)
set.seed(1104)

pop.m<-20000 # number of clusters
pop.n<- 5 # number within clusters
case.prob<-.01 #probability of case in underlying population
beta<-5 #slope for indicator
sigma<- .5 #overall standard deviation in the linear model
tau_e<-0.8 #error correlation
reps=2000


beta.est<-vector(length = reps)
for(i in 1:reps){
x<-rbinom(pop.m,1,prob=case.prob)
x<-rep(x,each=pop.n)

u<-rnorm(pop.m,mean = 0, sd=sqrt(sigma*tau_e)) #cluster samples
u1<-rep(u,each=pop.n) # repeat each cluster sample n times
estar<-rnorm(pop.m*pop.n,mean = 0, sd=sqrt(sigma*(1-tau_e))) # samples within each cluster
err<-u1+estar #total error

y<-beta*x+err
dat<-data.frame(y=y,x=x,id=rep(c(1:pop.m),each=pop.n)) #make data
agg.dat<-aggregate(y~id, dat, sum) # sum y by id
case.samp<-sample(agg.dat$id[agg.dat$y>15],50) #sample cases
control.samp<-sample(agg.dat$id[agg.dat$y<15],50)# sample controls
samp<-c(case.samp, control.samp)
samp.dat<-subset(dat,dat$id%in%samp) # get dataframe for sampled ids

fit<-lme(y~x, data = samp.dat, random = ~1|id)
beta.est[i]<-fixed.effects(fit)[2]
}

hist(beta.est)### histogram looks good
```

Looking back at my notes, I'm not fully sure what to do from here. I have a note that says subtract off the group means of the X's and Y's. That would give me something like that within-subject part of the model. But what do I do with that? look at the estimate for the fixed effects?

Updated Simulation

Discrete X

I've updated the model I'm studying, now we have,

$$Y_{ij} = \beta_0 + \beta X_{ij} + U_i + \epsilon_{ij}.$$

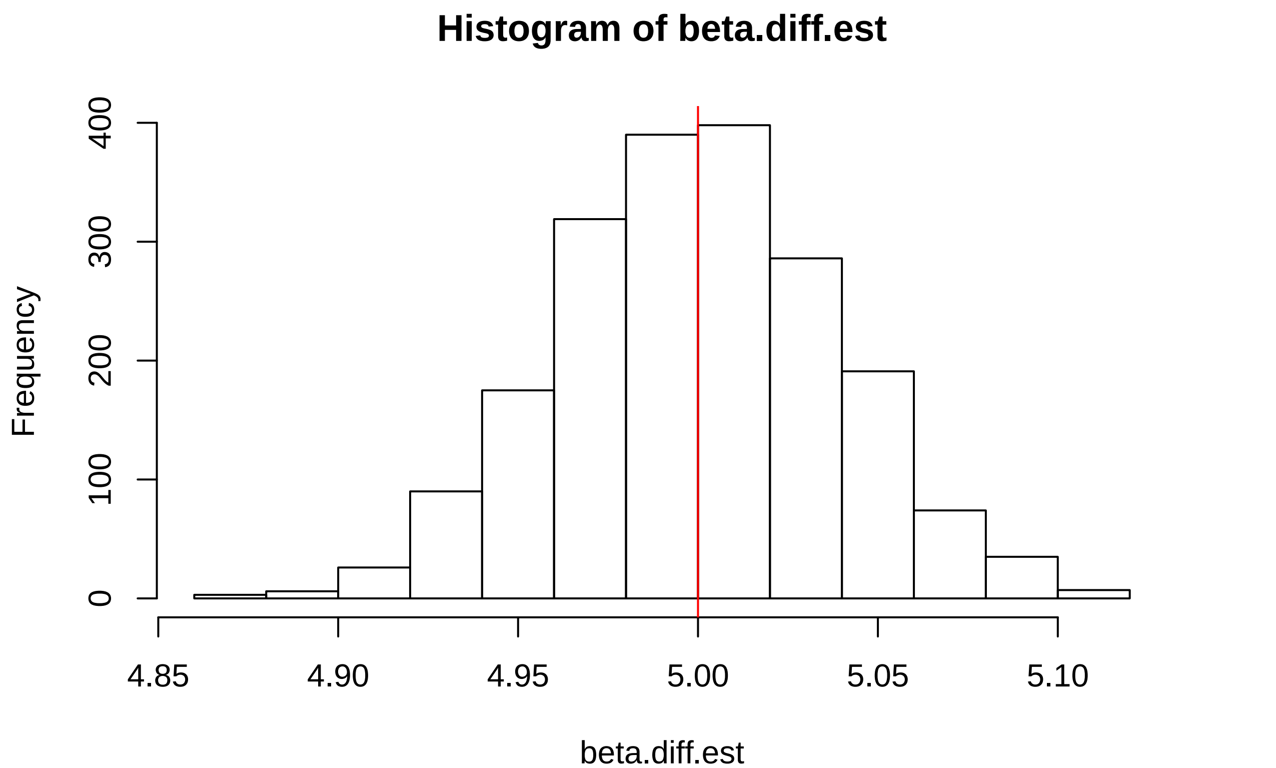
Where I set $\beta_0 = 0$ and $\beta = 5$. The X_{ij} terms are generated as follows: 1.) We generate $X_{ij}^* \sim N(U_i, 1)$, 2.) We then make the terms into a binary random variable by coding $X_{ij}^* > 2.5$ as 1 and 0 otherwise. The rest of the simulation is the same as above except at the end we fit a model of the form,

$$Y_{ij} = \beta_0 + \beta_1 \bar{X}_{i.} + \beta_2 \left(X_{ij} - \bar{X}_{i.} \right) + U_i + \epsilon_{ij}$$

```
####ODS Conditional Likelihood Simulation: Jacob M. Maronge 01/09/17

#### Inspired by the paper: Separating between- and within-cluster covariate effects by using conditiona
#### By John M. Neuhaus and Charles E. McCulloch
set.seed(1104)
library(foreach)
library(doParallel)
cores=detectCores()
cl <- makeCluster(3)
registerDoParallel(cl)
dontprint <- clusterEvalQ(cl,
             {library(nlme)
               pop.m<-20000 # number of clusters
               pop.n<- 5 # number within clusters
               beta<-5 #slope for indicator
               sigma<- 1 #overall standard deviation in the linear model
               tau_e<-0.8 #error correlation
               norm.sim<-function(pop.m, pop.n, beta,tau_e, sigma){
                 u<-rnorm(pop.m,mean = 0, sd=sqrt(sigma*tau_e)) #cluster samples
                 u1<-rep(u,each=pop.n) # repeat each cluster sample n times
                 estar<-rnorm(pop.m*pop.n,mean = 0, sd=sqrt(sigma*(1-tau_e))) # samples within each clu
                 err<-u1+estar #total error

                 x<-rnorm(u1,1)
                 x<-as.numeric(x>2.5)
                 y<-beta*x+err
                 dat<-data.frame(y=y,x=x,id=rep(c(1:pop.m),each=pop.n)) #make data
                 agg.dat<-aggregate(y~id, dat, sum) # sum y by id
                 case.samp<-sample(agg.dat$id[agg.dat$y>12],50) #sample cases
                 control.samp<-sample(agg.dat$id[agg.dat$y<12],50)# sample controls
                 samp<-c(case.samp, control.samp)
                 samp.dat<-subset(dat,dat$id%in%samp) # get dataframe for sampled ids
                 samp.agg.dat<-aggregate(x~id, samp.dat, mean) #calculate means for x
                 x_ibar<-rep(samp.agg.dat$x,each=pop.n) # mach means dimensions with dat
                 samp.dat$x_ibar<-x_ibar
                 samp.dat$diff<-samp.dat$x-samp.dat$x_ibar # calculate x_ij-x_ibar

                 fit<-lme(y~diff+x_ibar, data = samp.dat, random = ~1|id)
                 beta.diff.est<-fixed.effects(fit)[2]
                 beta.diff.cov.prob<-(intervals(fit)$fixed[2,1]<=beta&intervals(fit)$fixed[2,3]>=beta)
```

```
                  beta.x_ibar.est<-fixed.effects(fit)[3]
                  beta.x_ibar.cov.prob<-(intervals(fit)$fixed[3,1]<=beta&intervals(fit)$fixed[3,3]>=beta
                  out<-list(beta.diff.est, beta.diff.cov.prob, beta.x_ibar.est,  beta.x_ibar.cov.prob)
                  names(out)<-c("Diff Estimate", "Diff Covered","x_ibar Estimate", "x_ibar Covered")
                  return(out)
                }})

out<-foreach(i=1:2000, .combine=cbind) %dopar% {
  norm.sim(pop.m, pop.n, beta,tau_e, sigma)

}
beta.diff.est<-unlist(out[1,])
beta.diff.cov.prob<-unlist(out[2,])
beta.x_ibar.est <- unlist(out[3,])
beta.x_ibar.cov.prob<-unlist(out[4,])
```

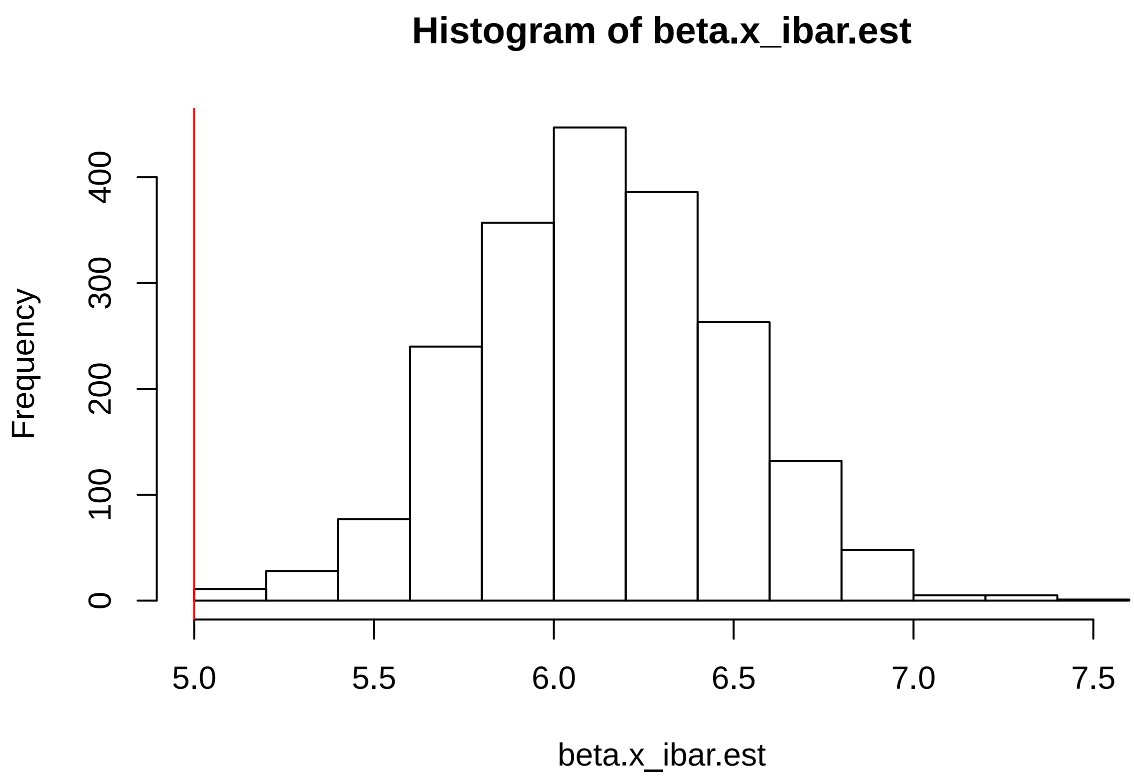
```
hist(beta.diff.est)
abline(v = 5, col = "red")
```




```
mean(beta.diff.cov.prob)
```

```
## [1] 0.951
```

```
hist(beta.x_ibar.est)
abline(v = 5, col = "red")
```




```
mean(beta.x_ibar.cov.prob)
```

```
## [1] 0.1325
```

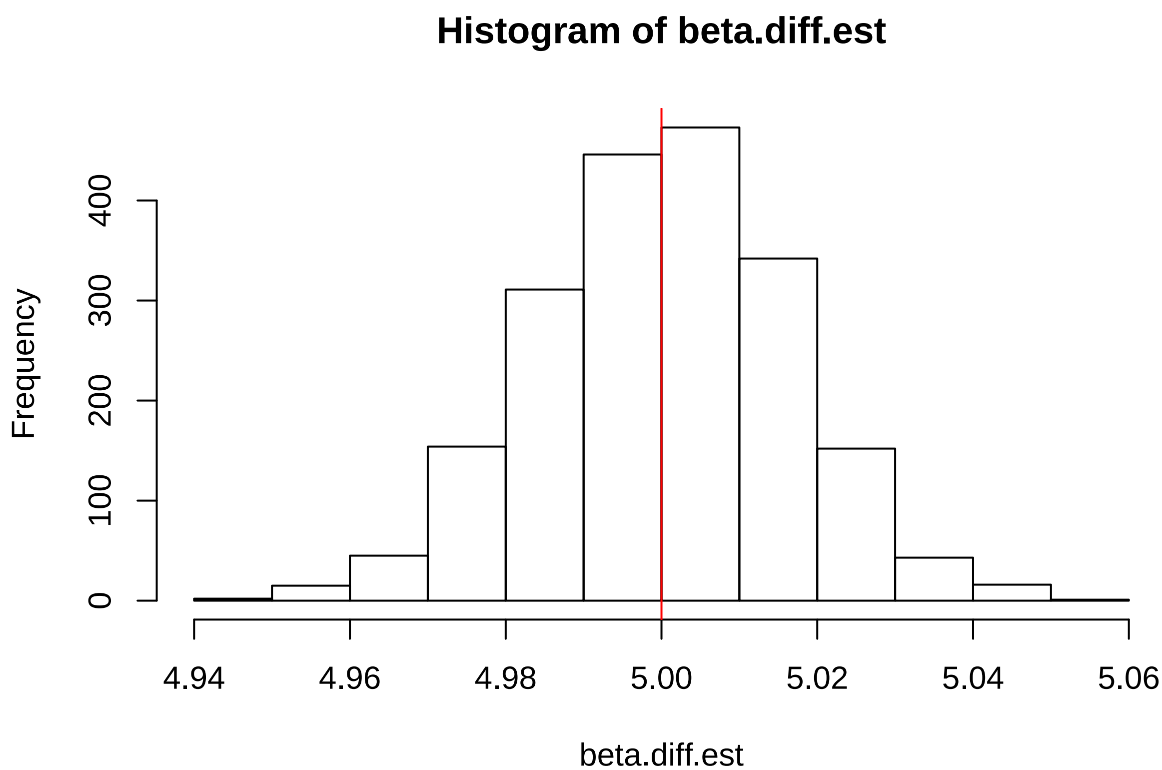
Continuous X

Here I generate the same model as above, except I don't convert the X's to discrete values.

```
set.seed(1104)
library(foreach)
library(doParallel)
cores=detectCores()
cl <- makeCluster(3)
registerDoParallel(cl)
dontprint <- clusterEvalQ(cl,
             {library(nlme)
               pop.m<-20000 # number of clusters
               pop.n<- 5 # number within clusters
               beta<-5 #slope for indicator
               sigma<- 1 #overall standard deviation in the linear model
               tau_e<-0.8 #error correlation
               norm.sim<-function(pop.m, pop.n, beta,tau_e, sigma){
                 u<-rnorm(pop.m,mean = 0, sd=sqrt(sigma*tau_e)) #cluster samples
                 u1<-rep(u,each=pop.n) # repeat each cluster sample n times
                 estar<-rnorm(pop.m*pop.n,mean = 0, sd=sqrt(sigma*(1-tau_e))) # samples within each clu
                 err<-u1+estar #total error

                 x<-rnorm(u1,1)
                 y<-beta*x+err
                 dat<-data.frame(y=y,x=x,id=rep(c(1:pop.m),each=pop.n)) #make data
```

```r
                 agg.dat<-aggregate(y~id, dat, sum) # sum y by id
                 case.samp<-sample(agg.dat$id[agg.dat$y>12],50) #sample cases
                 control.samp<-sample(agg.dat$id[agg.dat$y<12],50)# sample controls
                 samp<-c(case.samp, control.samp)
                 samp.dat<-subset(dat,dat$id%in%samp) # get dataframe for sampled ids
                 samp.agg.dat<-aggregate(x~id, samp.dat, mean) #calculate means for x
                 x_ibar<-rep(samp.agg.dat$x,each=pop.n) # mach means dimensions with dat
                 samp.dat$x_ibar<-x_ibar
                 samp.dat$diff<-samp.dat$x-samp.dat$x_ibar # calculate x_ij-x_ibar

                 fit<-lme(y~diff+x_ibar, data = samp.dat, random = ~1|id)
                 beta.diff.est<-fixed.effects(fit)[2]
                 beta.diff.cov.prob<-(intervals(fit)$fixed[2,1]<=beta&intervals(fit)$fixed[2,3]>=beta)
                 beta.x_ibar.est<-fixed.effects(fit)[3]
                 beta.x_ibar.cov.prob<-(intervals(fit)$fixed[3,1]<=beta&intervals(fit)$fixed[3,3]>=beta)
                 out<-list(beta.diff.est, beta.diff.cov.prob, beta.x_ibar.est,  beta.x_ibar.cov.prob)
                 names(out)<-c("Diff Estimate", "Diff Covered","x_ibar Estimate", "x_ibar Covered")
                 return(out)
               }})

out<-foreach(i=1:2000, .combine=cbind) %dopar% {
  norm.sim(pop.m, pop.n, beta,tau_e, sigma)

}
beta.diff.est<-unlist(out[1,])
beta.diff.cov.prob<-unlist(out[2,])
beta.x_ibar.est <- unlist(out[3,])
beta.x_ibar.cov.prob<-unlist(out[4,])
```

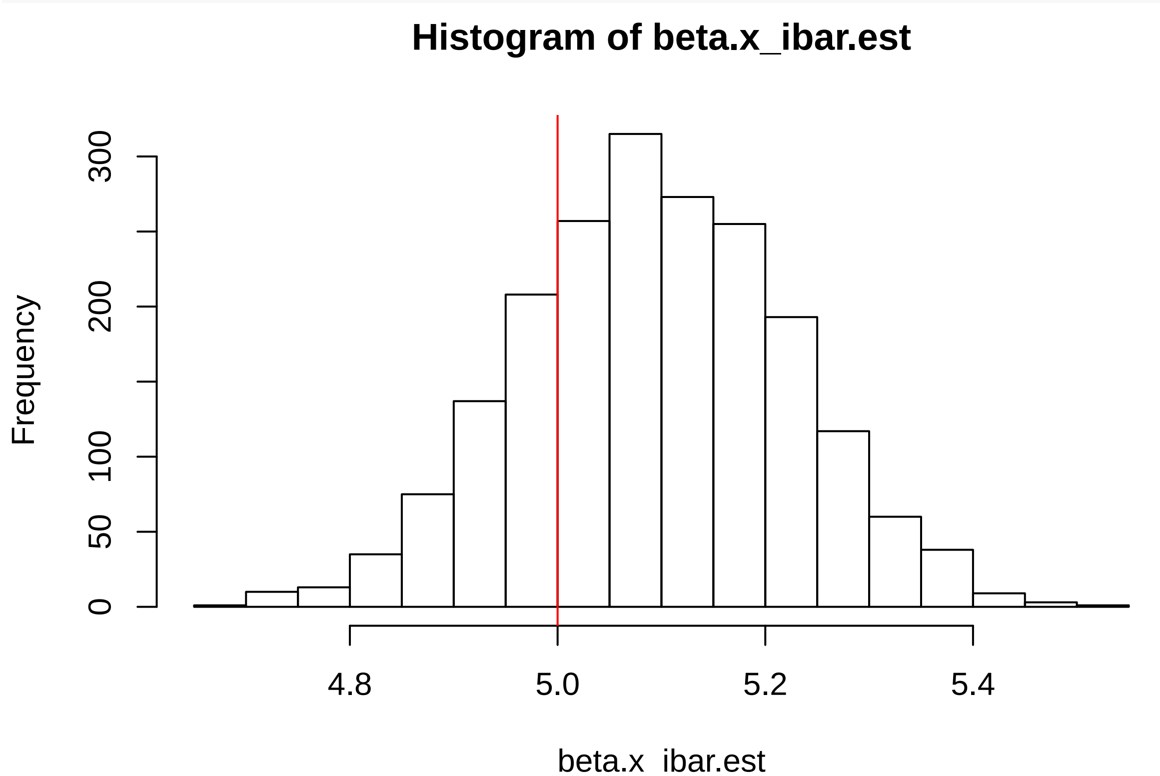
```r
hist(beta.diff.est)
abline(v = 5, col = "red")
```




```r
mean(beta.diff.cov.prob)
```

```
## [1] 0.9485
```

```r
hist(beta.x_ibar.est)
abline(v = 5, col = "red")
```



```r
mean(beta.x_ibar.cov.prob)
```

```
## [1] 0.8895
```

Logistic regression simulation 02/05/18

Discrete X

Since we've done the normal case, we decided to study a logistic model of the form,

$$\text{logit}(Y_{ij}) = \beta_0 + \beta X_{ij} + U_i.$$

Where I set $\beta_0 = -1.5$ and $\beta = 3$. The U_i terms are distributed as $U_i \sim N(0, 1/2)$. The X_{ij} terms are generated as follows: 1.) We generate $X^*_{ij} \sim N(U_i, 1)$, 2.) We then make the terms into a binary random variable by coding $X^*_{ij} > 2.5$ as 1 and 0 otherwise. Then we generate values Z_{ij} where,

$$Z_{ij} = \beta_0 + \beta X_{ij} + U_i.$$

Next, we take,

$$p_{ij} = \frac{1}{1 + e^{-Z_{ij}}}.$$

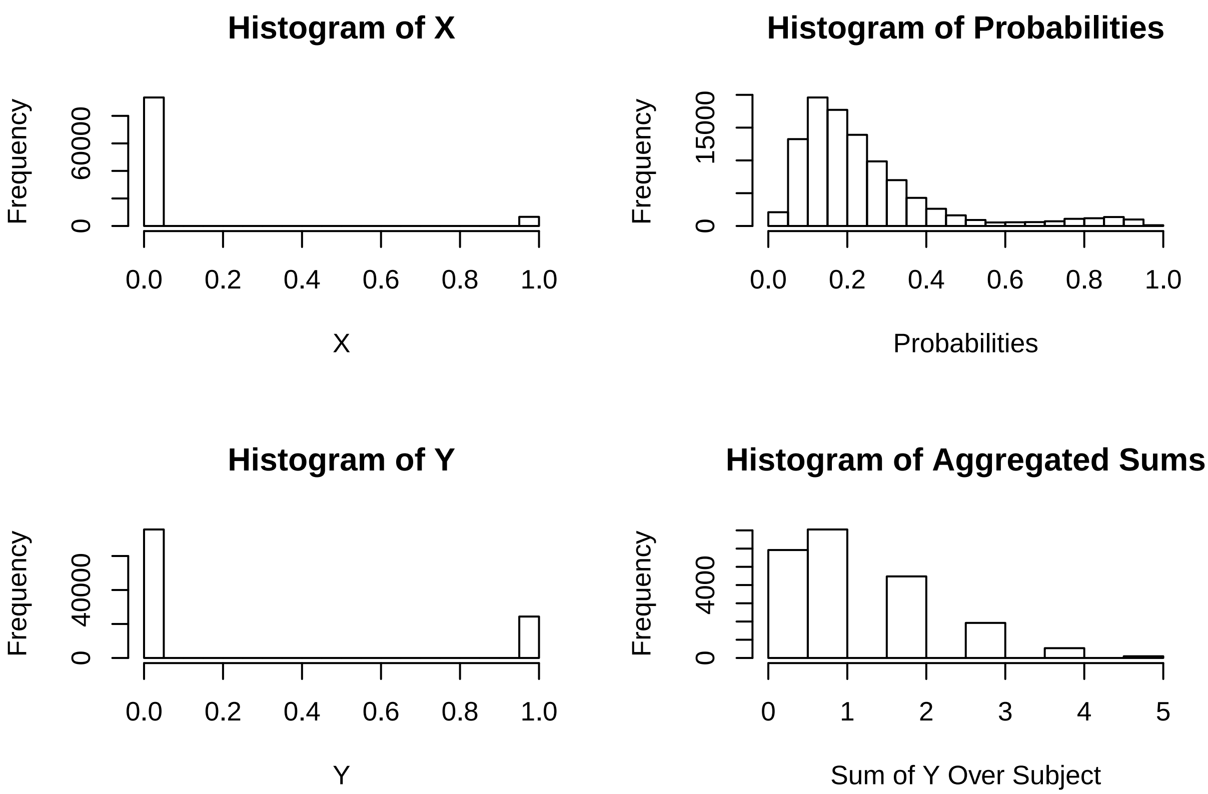
Finally, we generate our Y_{ij} by taking,

$$Y_{ij} = Bern(p_{ij}).$$

We then fit a model of the form,

$$\text{logit}(Y_{ij}) = \beta_0 + \beta_1 \bar{X}_{i.} + \beta_2 \left(X_{ij} - \bar{X}_{i.}\right) + U_i.$$

Below, I show some plots from one repetition of my simulation.

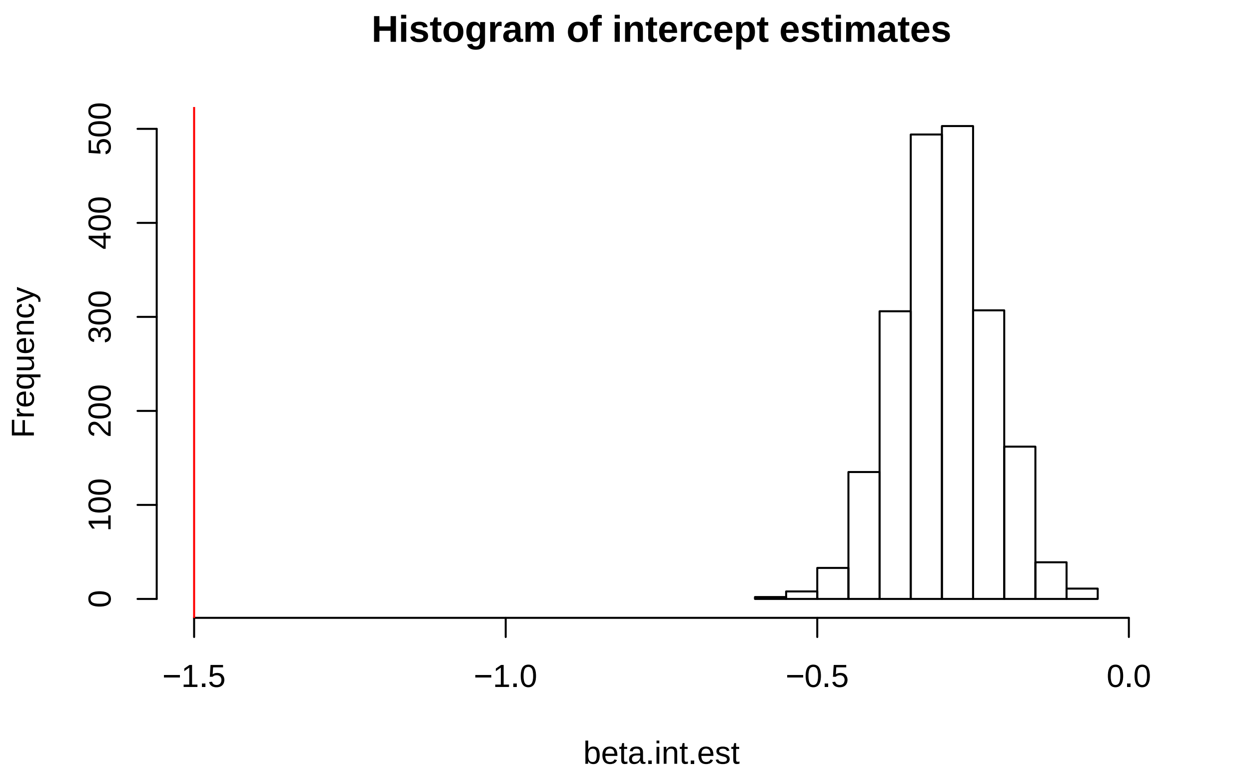



```r
set.seed(1104)
library(foreach)
library(doParallel)
```

```
cl <- makeCluster(3)  #not to overload your computer
registerDoParallel(cl)
dontprint <- clusterEvalQ(cl,
             {library(lme4)
               pop.m<-80000 # number of clusters
               pop.n<- 5 # number within clusters
               beta1<-3 #slope for indicator
               beta0<--1.5 #intercept terms
               tau_sq<-.5 #variance of random intercept
               logit.sim<-function(pop.m, pop.n, beta1,beta0,tau_sq){
                 library(lme4)
                 u<-rnorm(pop.m,mean = 0, sd=sqrt(tau_sq)) #cluster samples
                 u1<-rep(u,each=pop.n) # repeat each cluster sample n times
                 x<-rnorm(u1,1)
                 x<-as.numeric(x>2.5)
                 z<-beta0+beta1*x+u1
                 pr<-1/(1+exp(-z))
                 y<-rbinom(n=pop.m*pop.n,size = 1,prob = pr )
                 dat<-data.frame(y=y,x=x,id=rep(c(1:pop.m),each=pop.n)) #make data
                 agg.dat<-aggregate(y~id, dat, sum) # sum y by id
                 case.samp<-sample(agg.dat$id[agg.dat$y>=4],250) #sample cases
                 control.samp<-sample(agg.dat$id[agg.dat$y<4],250)# sample controls
                 samp<-c(case.samp, control.samp)
                 samp.dat<-subset(dat,dat$id%in%samp) # get dataframe for sampled ids
                 samp.agg.dat<-aggregate(x~id, samp.dat, mean) #calculate means for x
                 x_ibar<-rep(samp.agg.dat$x,each=pop.n) # mach means dimensions with dat
                 samp.dat$x_ibar<-(x_ibar)
                 samp.dat$diff<-(samp.dat$x-samp.dat$x_ibar) # calculate x_ij-x_ibar

                 fit.logit<-glmer(y~diff+x_ibar+(1|id),data = samp.dat,
                                  family = binomial(link = "logit"),
                                  glmerControl(optimizer = c("bobyqa","Nelder_Mead")) )

                 beta.int.est<-coef(summary(fit.logit))[1,1]
                 beta.int.cov.prob<-((coef(summary(fit.logit))[1,1]-qt(.975, df=2000)*coef(summary(fit.l
                 beta.diff.est<-coef(summary(fit.logit))[2,1]
                 beta.diff.cov.prob<-((coef(summary(fit.logit))[2,1]-qt(.975, df=2000)*coef(summary(fit
                 beta.x_ibar.est<-coef(summary(fit.logit))[3,1]
                 beta.x_ibar.cov.prob<-((coef(summary(fit.logit))[3,1]-qt(.975, df=2000)*coef(summary(fi
                 out<-list(beta.int.est, beta.int.cov.prob, beta.diff.est, beta.diff.cov.prob, beta.x_il
                 names(out)<-c("Intercept Estimate", "Intercept Covered", "Diff Estimate", "Diff Covere
                 return(out)
               }}
               )


out<-foreach(i=1:2000, .combine=cbind) %dopar% {
  logit.sim(pop.m, pop.n, beta1,beta0,tau_sq)

}
beta.int.est<-unlist(out[1,])
beta.int.cov.prob<-unlist(out[2,])
beta.diff.est<-unlist(out[3,])
```

```r
beta.diff.cov.prob<-unlist(out[4,])
beta.x_ibar.est<-unlist(out[5,])
beta.x_ibar.cov.prob<-unlist(out[6,])
```

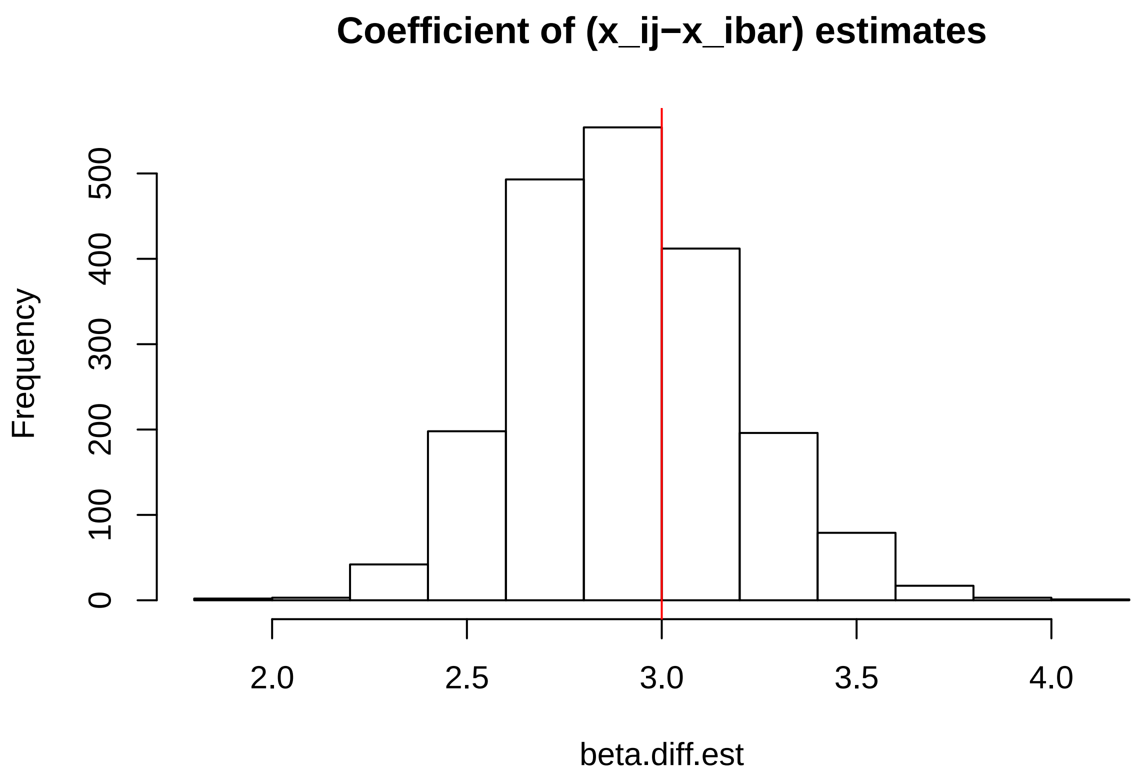
```r
hist(beta.int.est,main = "Histogram of intercept estimates",xlim = c(-1.5,0))
abline(v = -1.5, col = "red")
```




```r
mean(beta.int.cov.prob)
```

```
## [1] 0
```

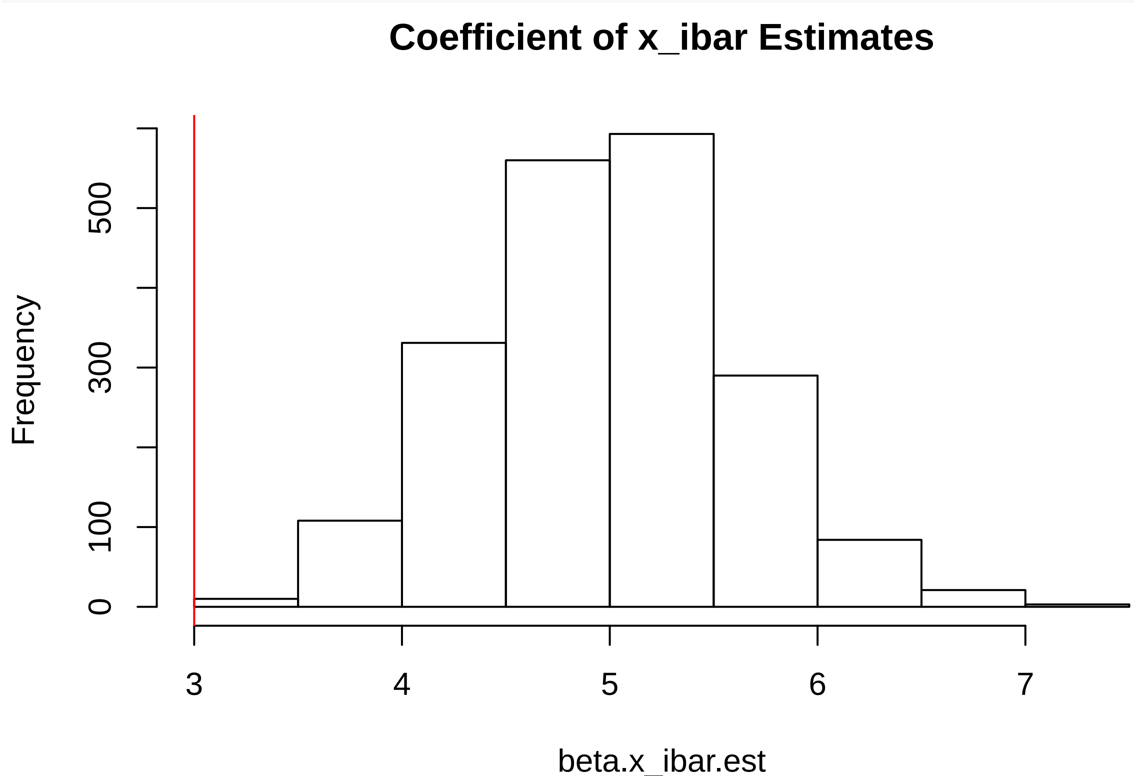
```r
hist(beta.diff.est, main = "Coefficient of (x_ij-x_ibar) estimates")
abline(v = 3, col = "red")
```




```r
mean(beta.diff.cov.prob)
```

```
## [1] 0.944
```

```r
hist(beta.x_ibar.est, main = "Coefficient of x_ibar Estimates")
abline(v = 3, col = "red")
```



```r
mean(beta.x_ibar.cov.prob)
```

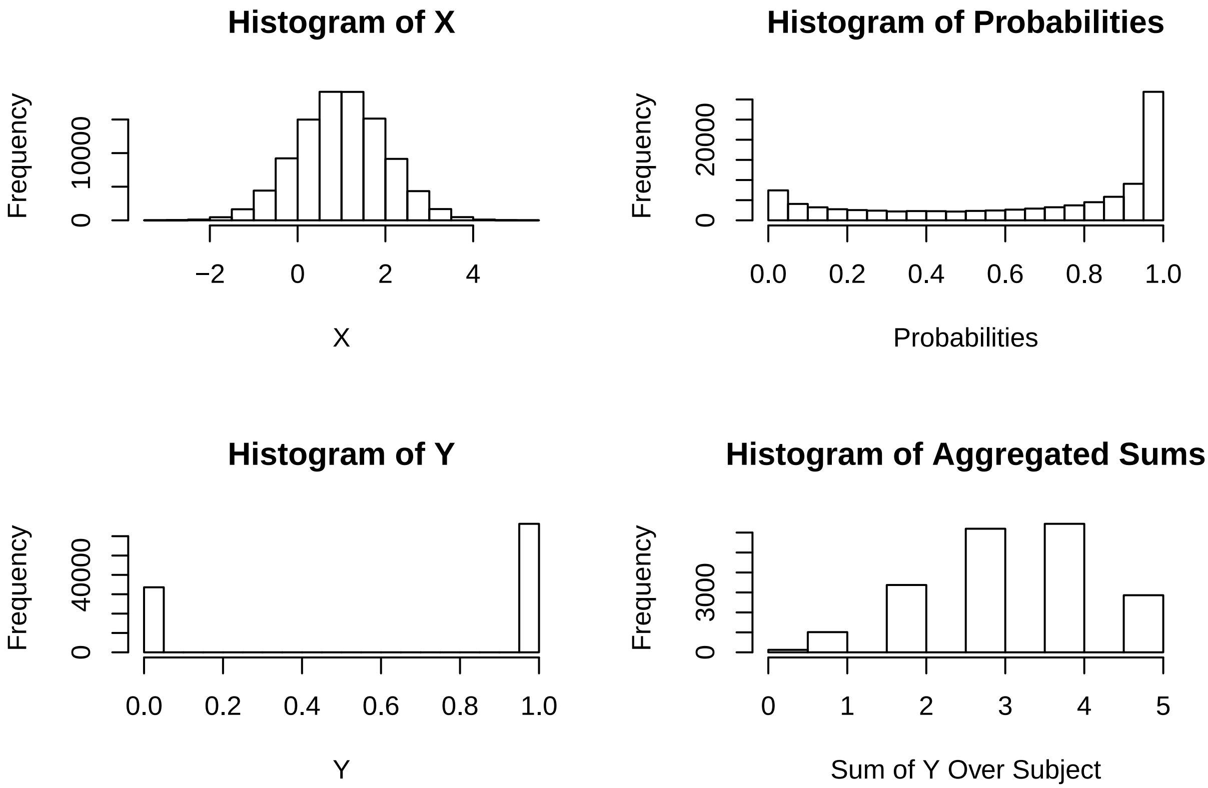
```
## [1] 0.116
```

The vertical red lines are drawn at the true value for each parameter. The intercept term appears to be biased, as well as the between-subject effect, and possibly also the within-subject effect.

Continuous X

Here I generate the same model as above, except I don't convert the X's to discrete values.

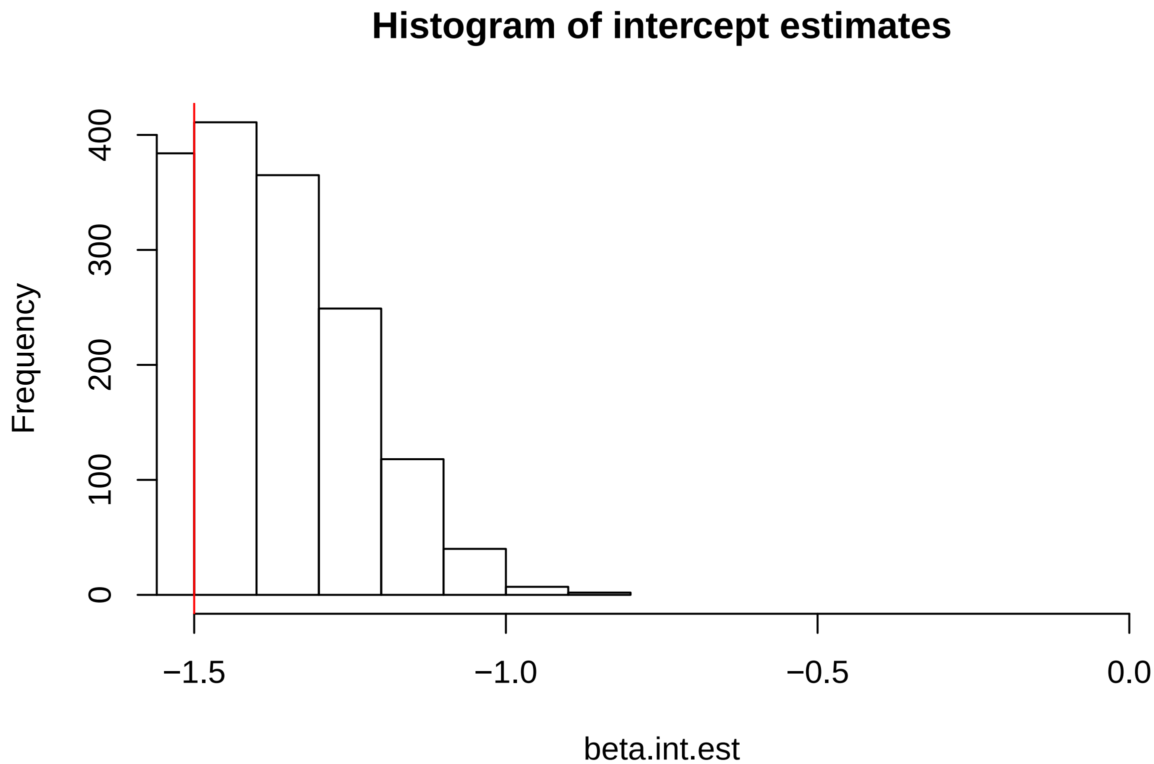
Below, I show some plots from one repetition of my simulation.




```r
set.seed(1104)
library(foreach)
library(doParallel)
cores=detectCores()
cl <- makeCluster(3)
registerDoParallel(cl)
dontprint <- clusterEvalQ(cl,
             {library(lme4)
               pop.m<-80000 # number of clusters
               pop.n<- 5 # number within clusters
               beta1<-3 #slope for indicator
               beta0<--1.5 #intercept terms
               tau_sq<-.5 #variance of random intercept
               logit.sim<-function(pop.m, pop.n, beta1,beta0,tau_sq){
                 library(lme4)
                 u<-rnorm(pop.m,mean = 0, sd=sqrt(tau_sq)) #cluster samples
                 u1<-rep(u,each=pop.n) # repeat each cluster sample n times
```

```
                  x<-rnorm(u1,1)
                  z<-beta0+beta1*x+u1
                  pr<-1/(1+exp(-z))
                  y<-rbinom(n=pop.m*pop.n,size = 1,prob = pr )
                  dat<-data.frame(y=y,x=x,id=rep(c(1:pop.m),each=pop.n)) #make data
                  agg.dat<-aggregate(y~id, dat, sum) # sum y by id
                  case.samp<-sample(agg.dat$id[agg.dat$y>=4],250) #sample cases
                  control.samp<-sample(agg.dat$id[agg.dat$y<4],250)# sample controls
                  samp<-c(case.samp, control.samp)
                  samp.dat<-subset(dat,dat$id%in%samp) # get dataframe for sampled ids
                  samp.agg.dat<-aggregate(x~id, samp.dat, mean) #calculate means for x
                  x_ibar<-rep(samp.agg.dat$x,each=pop.n) # mach means dimensions with dat
                  samp.dat$x_ibar<-(x_ibar)
                  samp.dat$diff<-(samp.dat$x-samp.dat$x_ibar) # calculate x_ij-x_ibar

                  fit.logit<-glmer(y~diff+x_ibar+(1|id),data = samp.dat,
                                   family = binomial(link = "logit"),
                                   glmerControl(optimizer = c("bobyqa","Nelder_Mead")) )

                  beta.int.est<-coef(summary(fit.logit))[1,1]
                  beta.int.cov.prob<-((coef(summary(fit.logit))[1,1]-qt(.975, df=2000)*coef(summary(fit.l
                  beta.diff.est<-coef(summary(fit.logit))[2,1]
                  beta.diff.cov.prob<-((coef(summary(fit.logit))[2,1]-qt(.975, df=2000)*coef(summary(fit
                  beta.x_ibar.est<-coef(summary(fit.logit))[3,1]
                  beta.x_ibar.cov.prob<-((coef(summary(fit.logit))[3,1]-qt(.975, df=2000)*coef(summary(fi
                  out<-list(beta.int.est, beta.int.cov.prob, beta.diff.est, beta.diff.cov.prob, beta.x_il
                  names(out)<-c("Intercept Estimate", "Intercept Covered", "Diff Estimate", "Diff Covere
                  return(out)
                }}
)


out<-foreach(i=1:2000, .combine=cbind) %dopar% {
  logit.sim(pop.m, pop.n, beta1,beta0,tau_sq)

}
beta.int.est<-unlist(out[1,])
beta.int.cov.prob<-unlist(out[2,])
beta.diff.est<-unlist(out[3,])
beta.diff.cov.prob<-unlist(out[4,])
beta.x_ibar.est<-unlist(out[5,])
beta.x_ibar.cov.prob<-unlist(out[6,])
```

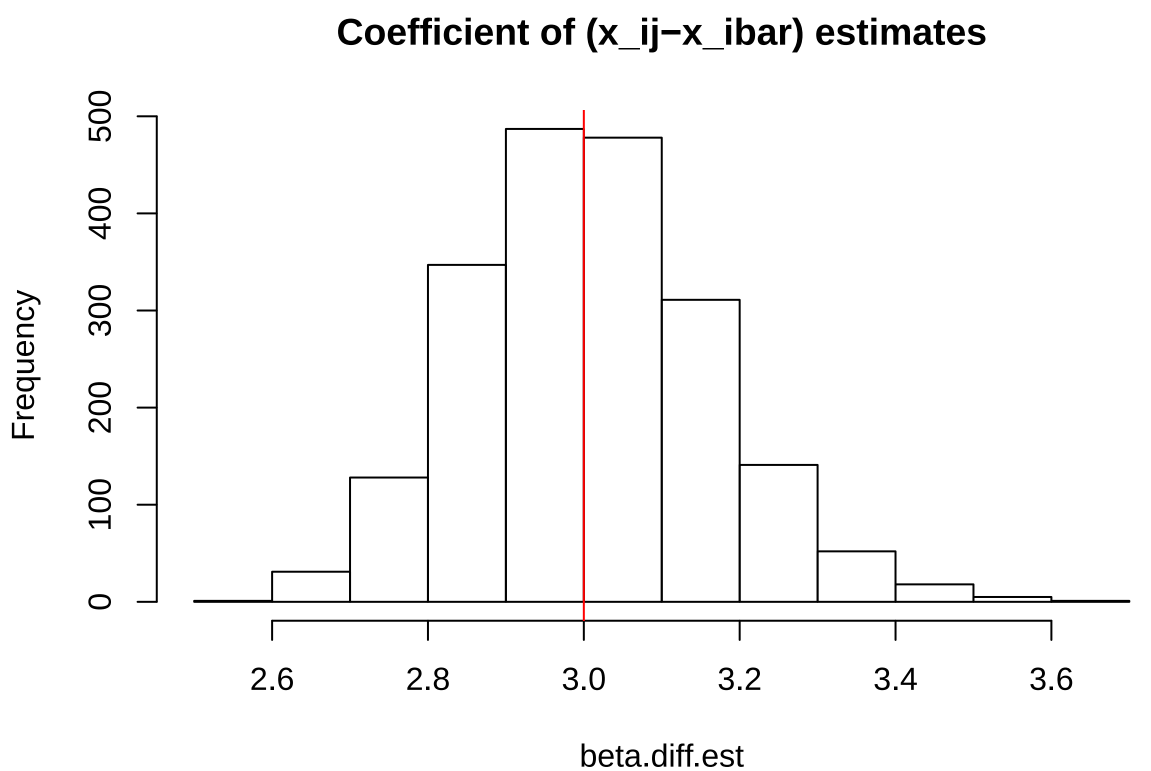
```
hist(beta.int.est,main = "Histogram of intercept estimates",xlim = c(-1.5,0))
abline(v = -1.5, col = "red")
```




```r
mean(beta.int.cov.prob)
```

```
## [1] 0.9055
```

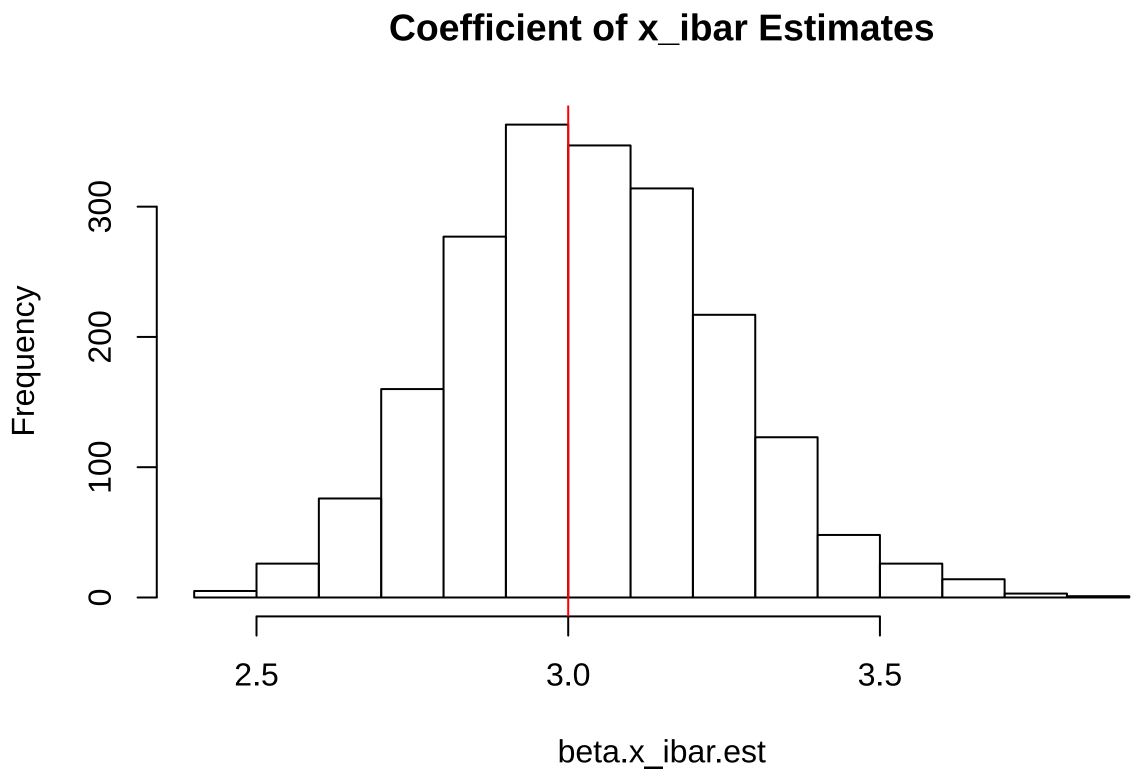
```r
hist(beta.diff.est, main = "Coefficient of (x_ij-x_ibar) estimates")
abline(v = 3, col = "red")
```



```r
mean(beta.diff.cov.prob)
```

```
## [1] 0.92
```

```r
hist(beta.x_ibar.est, main = "Coefficient of x_ibar Estimates")
abline(v = 3, col = "red")
```




```r
mean(beta.x_ibar.cov.prob)
```

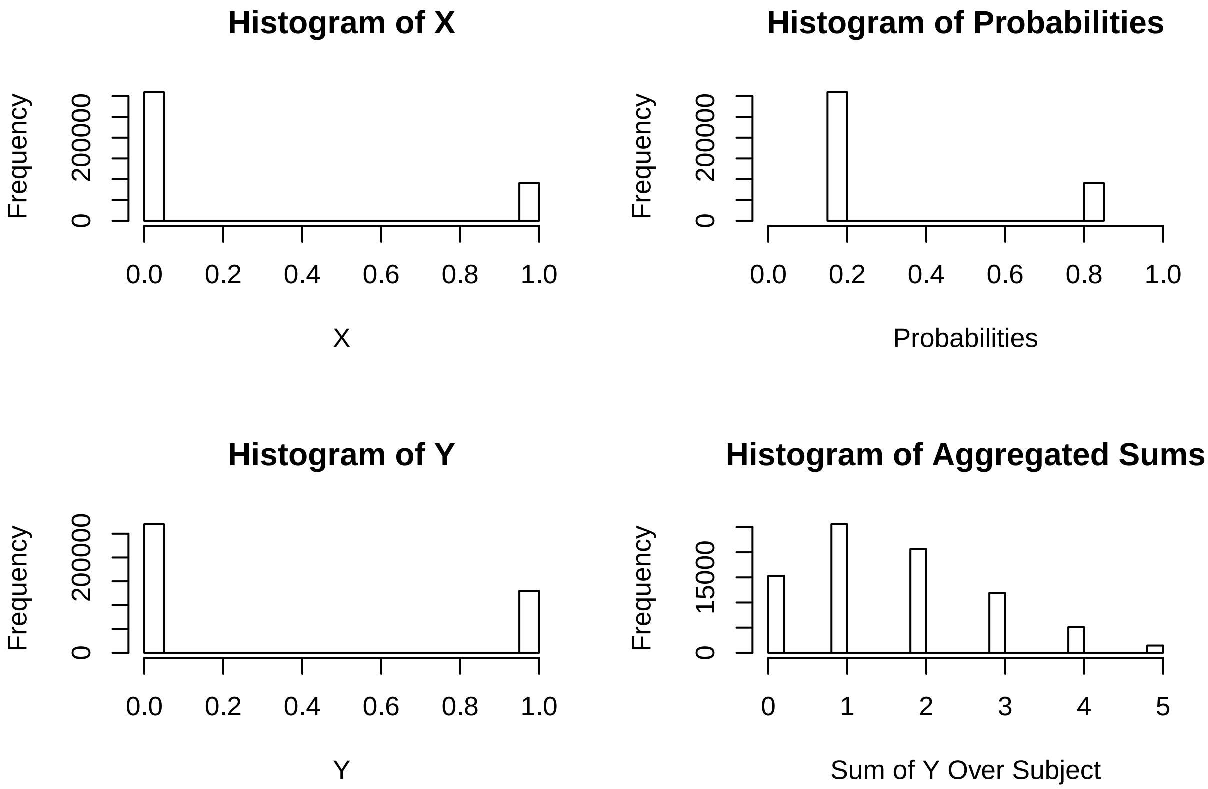
```
## [1] 0.92
```

Conditional logistic regression

Discrete X

Below, I show some plots from one repetition of my simulation.

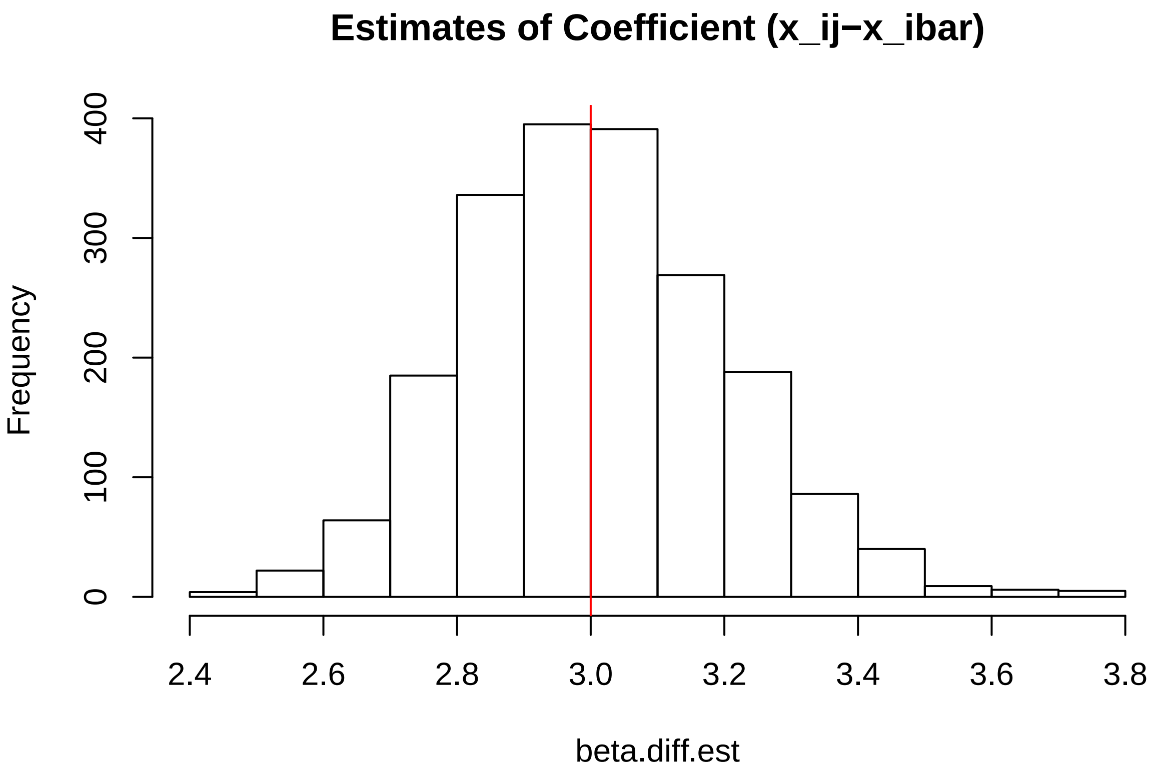
```
##
##     0     1     2     3     4     5
## 15329 25583 20653 11903  5101  1431
```




```
set.seed(1104)
library(foreach)
library(doParallel)
cores=detectCores()
cl <- makeCluster(3)
registerDoParallel(cl)
dontprint <- clusterEvalQ(cl,
             {library(survival)
               pop.m<-80000 # number of clusters
               pop.n<- 5 # number within clusters
               beta1<-3 #slope for indicator
               beta0<--1.5 #intercept terms
               tau_x<-0.5
               sigma<-1
               clogit.sim<-function(pop.m, pop.n, beta1,beta0,tau_x, sigma){
                 v<-rnorm(pop.m,mean = 0, sd=sqrt(sigma*tau_x)) #cluster samples
                 v1<-rep(v,each=pop.n) # repeat each cluster sample n times
                 xstar<-rnorm(pop.m*pop.n,mean = 0, sd=sqrt(sigma*(1-tau_x))) # samples within each clu
                 x<-v1+xstar #total x
                 x<-as.numeric(x>.75)
                 z<-beta0+beta1*x
                 pr<-1/(1+exp(-z))
                 y<-rbinom(n=pop.m*pop.n,size = 1,prob = pr )
                 dat<-data.frame(y=y,x=x,id=rep(c(1:pop.m),each=pop.n)) #make data
                 agg.dat<-aggregate(y~id, dat, sum) # sum y by id
                 table(agg.dat$y)
                 case.samp<-sample(agg.dat$id[agg.dat$y>=4],250) #sample cases
                 control.samp<-sample(agg.dat$id[agg.dat$y<4],250)# sample controls
                 samp<-c(case.samp, control.samp)
```

```
                samp.dat<-subset(dat,dat$id%in%samp) # get dataframe for sampled ids
                samp.agg.dat<-aggregate(x~id, samp.dat, mean) #calculate means for x
                x_ibar<-rep(samp.agg.dat$x,each=pop.n) # mach means dimensions with dat
                samp.dat$x_ibar<-(x_ibar)
                samp.dat$diff<-(samp.dat$x-samp.dat$x_ibar) # calculate x_ij-x_ibar
                table(samp.dat$diff,samp.dat$y)
                fit.clogit<-clogit(y ~ diff + strata(id), samp.dat)
                beta.diff.est<-coef(fit.clogit)[1]
                ci<-confint(fit.clogit)
                beta.diff.cov.prob<-(ci[1,1]<=beta1&ci[1,2]>=beta1)
                out<-list(beta.diff.est, beta.diff.cov.prob)
                names(out)<-c("Diff Estimate", "Diff Covered")
                return(out)
              }})

out<-foreach(i=1:2000, .combine=cbind) %dopar% {
  clogit.sim(pop.m, pop.n, beta1,beta0,tau_x, sigma)

}
beta.diff.est<-unlist(out[1,])
beta.diff.cov.prob<-unlist(out[2,])
```

```
hist(beta.diff.est, main = "Estimates of Coefficient (x_ij-x_ibar)")
abline(v = 3, col = "red")
```



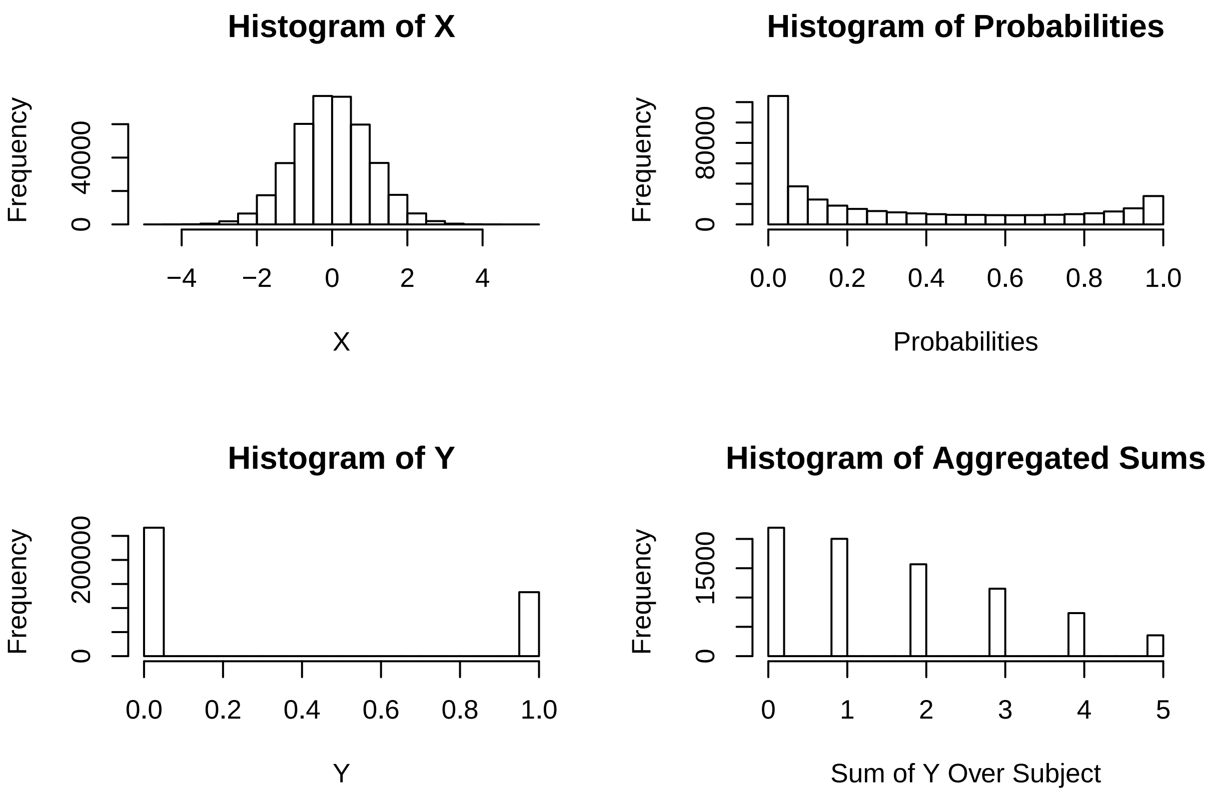

```
mean(beta.diff.cov.prob)
```

```
## [1] 0.9465
```

Continuous X

Below, I show some plots from one repetition of my simulation.

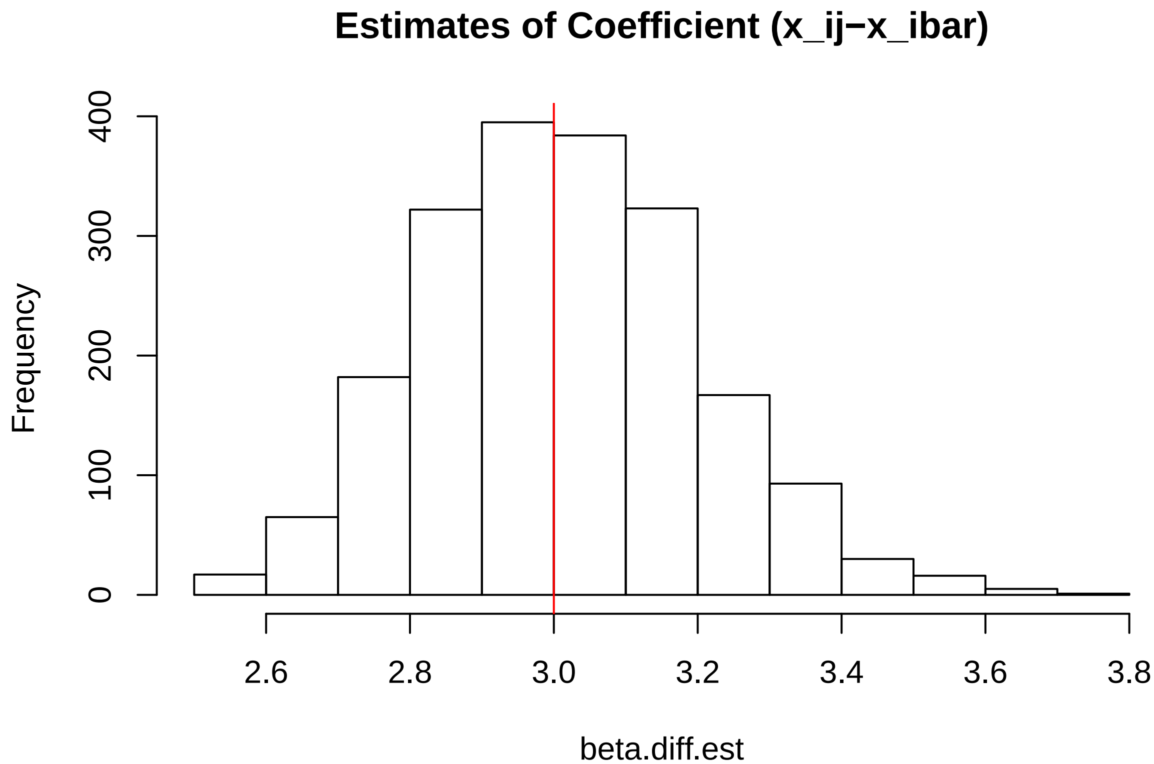
```
##
##     0     1     2     3     4     5
## 21912 20016 15675 11503  7344  3550
```




```
set.seed(1104)
library(foreach)
library(doParallel)
cores=detectCores()
cl <- makeCluster(3)
registerDoParallel(cl)
dontprint <- clusterEvalQ(cl,
             {library(survival)
               pop.m<-80000 # number of clusters
               pop.n<- 5 # number within clusters
               beta1<-3 #slope for indicator
               beta0<--1.5 #intercept terms
               tau_x<-0.5
               sigma<-1
               clogit.sim<-function(pop.m, pop.n, beta1,beta0,tau_x, sigma){
                 v<-rnorm(pop.m,mean = 0, sd=sqrt(sigma*tau_x)) #cluster samples
                 v1<-rep(v,each=pop.n) # repeat each cluster sample n times
                 xstar<-rnorm(pop.m*pop.n,mean = 0, sd=sqrt(sigma*(1-tau_x))) # samples within each clu
                 x<-v1+xstar #total x
                 z<-beta0+beta1*x
                 pr<-1/(1+exp(-z))
                 y<-rbinom(n=pop.m*pop.n,size = 1,prob = pr )
```

```r
                  dat<-data.frame(y=y,x=x,id=rep(c(1:pop.m),each=pop.n)) #make data
                  agg.dat<-aggregate(y~id, dat, sum) # sum y by id
                  table(agg.dat$y)
                  case.samp<-sample(agg.dat$id[agg.dat$y>=4],250) #sample cases
                  control.samp<-sample(agg.dat$id[agg.dat$y<4],250)# sample controls
                  samp<-c(case.samp, control.samp)
                  samp.dat<-subset(dat,dat$id%in%samp) # get dataframe for sampled ids
                  samp.agg.dat<-aggregate(x~id, samp.dat, mean) #calculate means for x
                  x_ibar<-rep(samp.agg.dat$x,each=pop.n) # mach means dimensions with dat
                  samp.dat$x_ibar<-(x_ibar)
                  samp.dat$diff<-(samp.dat$x-samp.dat$x_ibar) # calculate x_ij-x_ibar
                  table(samp.dat$diff,samp.dat$y)
                  fit.clogit<-clogit(y ~ diff + strata(id), samp.dat)
                  beta.diff.est<-coef(fit.clogit)[1]
                  ci<-confint(fit.clogit)
                  beta.diff.cov.prob<-(ci[1,1]<=beta1&ci[1,2]>=beta1)
                  out<-list(beta.diff.est, beta.diff.cov.prob)
                  names(out)<-c("Diff Estimate", "Diff Covered")
                  return(out)
                }})

out<-foreach(i=1:2000, .combine=cbind) %dopar% {
  clogit.sim(pop.m, pop.n, beta1,beta0,tau_x, sigma)

}
beta.diff.est<-unlist(out[1,])
beta.diff.cov.prob<-unlist(out[2,])
```

```r
hist(beta.diff.est, main = "Estimates of Coefficient (x_ij-x_ibar)")
abline(v = 3, col = "red")
```




```
mean(beta.diff.cov.prob)
```

```
## [1] 0.9485
```