Stat 571 HW3

Dongyang Wang

2023-02-10

```
rm(list=ls())
set.seed(42)
```

Question 1

Question 1.1

```
sixcity <- read.table("sixcity.dat", header = F)
colnames(sixcity) = c("wheezing", "id", "age", "smoking")
logistic_mod <- glm(wheezing~age*smoking, family = binomial(link = "logit"),
                    data = sixcity)
results_logistic <- summary(logistic_mod)$coefficients[,c(1,2)]
results_logistic
```

```
##               Estimate Std. Error
## (Intercept) -1.9008426 0.08874109
## age         -0.1412531 0.06951317
## smoking      0.3139540 0.13943864
## age:smoking  0.0708441 0.11072310
```

The above shows the result including the coefficients and the standard errors for the naive logistic model.

Question 1.2

```
library(geepack)
gee_indep <- geeglm(wheezing ~ age * smoking,
        id = id, corstr = "independence",
        family= binomial, data = sixcity)
```

```
library(gee)
gee_exch <- gee(wheezing ~ age * smoking,
        id = id, corstr = "exchangeable",
        family= binomial, data = sixcity)
```

```
## Beginning Cgee S-function, @(#) geeformula.q 4.13 98/01/27

## running glm to get initial regression estimate

## (Intercept)         age     smoking age:smoking
##  -1.9008426  -0.1412531   0.3139540   0.0708441
```

```
results_indep <- summary(gee_indep)$coefficients[,c(1,2)]
results_exch <- summary(gee_exch)$coefficients[,c(1,2)]
```

Table 1: Estimates and Standar Errors Comparison between GEE1 and logistic Regression

	logistic regression		Independence GEE1		Exchangeable GEE1	
	Estimate	Std. Error	Estimate	Std.err	Estimate	Naive S.E.
(Intercept)	-1.9008426	0.0887411	-1.9008426	0.1190768	-1.9004954	0.1187109
age	-0.1412531	0.0695132	-0.1412531	0.0582142	-0.1412359	0.0560803
smoking	0.3139540	0.1394386	0.3139540	0.1878385	0.3138258	0.1871972
age:smoking	0.0708441	0.1107231	0.0708441	0.0882947	0.0708318	0.0891776

```
library(kableExtra)
kbl(cbind(results_logistic,results_indep, results_exch),
  caption = "Estimates and Standar Errors Comparison
  between GEE1 and logistic Regression")  %>%
  add_header_above(c(" " = 1, "logistic regression"=2,
                     "Independence GEE1" = 2, "Exchangeable GEE1" = 2))%>%
  kableExtra::kable_styling(position = "center")
```

Based on the results above, the estimates for the coefficients are pretty similar across the methods. The standard errors tend to be larger in GEE methods for intercept and smoking, smaller for other variables. The results from independence and exchangeable structures are very similar in estimates and SEs.

In terms of interpretation, one unit increase in age is associated with about 0.14 decrease in the log odds of wheezing; smoking mothers are associated with 0.31 increase in log odds of wheezing compared with nonsmoker mothers; for smoking mothers the age is additionally 0.07 associated with the log odds of wheezing.

Question 1.4

One potential way to to use age only at the baseline but also include the length of time people have participated in the study. That means, separating the current age variable into two separate varibles and model on them in the new model. For the interaction term we may use the length of time as one component. In this way, time can be modeled as a random effect.

Question 2

Question 2.1

```
library(bindata)

p = 0.25
m = 300
n = 3
beta = c(-1.5, 0.5, 0.5)
nrep = 200
```

```
library(geepack)
res = do.call(rbind, lapply(c(1:nrep), function(nrep){
  # matrices
  male = matrix(c(1,1,1, 0,0,0, 0,1,2), byrow = F, nrow = 3)
  female = matrix(c(1,1,1, 1,1,1, 0,1,2), byrow = F, nrow = 3)
  cor = matrix(p, ncol = n, nrow = n) + diag(1-p, n)

  # model
  logit_male = exp(male%*%beta)
  margin_prob_male = logit_male/(1 + logit_male)
```

Table 2: True, Predicted, and Bias for Key Variables

	intercept	female	time	correlation
prediction	-1.5256242	0.5212663	0.5027369	0.2417057
true values	-1.5000000	0.5000000	0.5000000	0.2500000
bias	-0.0256242	0.0212663	0.0027369	-0.0082943

```
  logit_female = exp(female%*%beta)
  margin_prob_female = logit_female/(1 + logit_female)

  # random data
  y_male = rmvbin(n = m/2, margprob = margin_prob_male, bincorr = cor)
  y_female = rmvbin(n = m/2, margprob = margin_prob_female, bincorr = cor)

  df = data.frame(id = rep(1:m, each = n),
                  female = rep(c(0,1), each = m*n/2),
                  time = rep(c(0,1,2), m),
                  y = c(t(rbind(y_male, y_female))))

  # model
  gee <- geeglm(y ~ female + time, family = binomial("logit"), id = id,
              data = df, corstr ="exchangeable")

  # result
  data.frame(
    Estimate_intercept = summary(gee)$coefficients[1,1],
    Estimate_female = summary(gee)$coefficients[2,1],
    Estimate_time = summary(gee)$coefficients[3,1],
    SE_intercept = summary(gee)$coefficients[1,2],
    SE_female = summary(gee)$coefficients[2,2],
    SE_time = summary(gee)$coefficients[3,2],
    corr = summary(gee)$corr[1,1]
    )

}))
```

Question 2.2

```
res_2_pred = colMeans(res[,c(1:3,7)])
res_2_true = c(beta, p)
res_2_bias = res_2_pred - res_2_true
table_q2 <- rbind(res_2_pred, res_2_true, res_2_bias)
rownames(table_q2) = c("prediction", "true values", "bias")
colnames(table_q2) = c("intercept", "female", "time", "correlation")
kbl(table_q2,
  caption = "True, Predicted, and Bias for Key Variables")  %>%
  kableExtra::kable_styling(position = "center")
```

Question 2.3

Table 3: Comparison of SEs

	intercept	female	time
prediction	0.1625275	0.1780297	0.0791325
empirical	0.1726210	0.1668425	0.0817842

```
res_3_pred = colMeans(res[,4:6])
res_3_empirical = apply(res[,1:3], 2, sd)
table_q3 <- rbind(res_3_pred, res_3_empirical)
rownames(table_q3) = c("prediction", "empirical")
colnames(table_q3) = c("intercept", "female", "time")
kbl(table_q3,
  caption = "Comparison of SEs")  %>%
  kableExtra::kable_styling(position = "center")
```

Question 2.4

```
p = 0.75
try({
  # matrices
  male = matrix(c(1,1,1, 0,0,0, 0,1,2), byrow = F, nrow = 3)
  female = matrix(c(1,1,1, 1,1,1, 0,1,2), byrow = F, nrow = 3)
  cor = matrix(p, ncol = n, nrow = n) + diag(1-p, n)

  # model
  logit_male = exp(male%*%beta)
  margin_prob_male = logit_male/(1 + logit_male)
  logit_female = exp(female%*%beta)
  margin_prob_female = logit_female/(1 + logit_female)

  # random data
  y_male = rmvbin(n = m/2, margprob = margin_prob_male, bincorr = cor)
  y_female = rmvbin(n = m/2, margprob = margin_prob_female, bincorr = cor)
  })
```

```
## Error in Element ( 1 , 3 ): Admissible values are in [ 0 , 0.182425523806356 ].
## Error in commonprob2sigma(commonprob, simulvals) :
##   Matrix commonprob not admissible.
```

No, the correlation is too high to model. There is an error message detailing this “Error in commonprob2sigma(commonprob, simulvals) : Matrix commonprob not admissible”.

Question 3

Question 3.1

```
df = read.table("framingham.dat", header=F)
colnames(df) = c("age", "gender", "BMI_base", "BMI_10yrs", "cigarette_base", "cholst_base",
                 "cholst_2", "cholst_4", "cholst_6", "cholst_8", "cholst_10", "dead")
df$id <- seq.int(nrow(df))
df[df == -9] = NA
summary(df)
```

```
##       age            gender         BMI_base       BMI_10yrs    
##  Min.   :29.00   Min.   :1.000   Min.   :15.00   Min.   :15.00  
##  1st Qu.:36.00   1st Qu.:1.000   1st Qu.:22.00   1st Qu.:23.00  
##  Median :42.00   Median :2.000   Median :25.00   Median :25.00  
##  Mean   :42.96   Mean   :1.552   Mean   :25.04   Mean   :25.45  
##  3rd Qu.:50.00   3rd Qu.:2.000   3rd Qu.:27.00   3rd Qu.:28.00  
##  Max.   :62.00   Max.   :2.000   Max.   :56.00   Max.   :56.00  
##                                  NA's   :4       NA's   :1      
##  cigarette_base    cholst_base       cholst_2        cholst_4    
##  Min.   : 0.000   Min.   :117.0   Min.   :115.0   Min.   :113.0  
##  1st Qu.: 0.000   1st Qu.:188.0   1st Qu.:195.0   1st Qu.:200.0  
##  Median : 5.000   Median :217.0   Median :220.0   Median :225.0  
##  Mean   : 9.664   Mean   :219.3   Mean   :224.5   Mean   :228.8  
##  3rd Qu.:20.000   3rd Qu.:246.0   3rd Qu.:248.0   3rd Qu.:254.0  
##  Max.   :60.000   Max.   :503.0   Max.   :479.0   Max.   :500.0  
##  NA's   :4                        NA's   :357     NA's   :387    
##     cholst_6        cholst_8       cholst_10          dead       
##  Min.   :126.0   Min.   :135.0   Min.   :115.0   Min.   :0.0000  
##  1st Qu.:208.2   1st Qu.:210.0   1st Qu.:218.0   1st Qu.:0.0000  
##  Median :236.0   Median :237.0   Median :246.0   Median :0.0000  
##  Mean   :238.4   Mean   :240.8   Mean   :249.3   Mean   :0.2088  
##  3rd Qu.:265.0   3rd Qu.:266.0   3rd Qu.:276.0   3rd Qu.:0.0000  
##  Max.   :545.0   Max.   :696.0   Max.   :525.0   Max.   :1.0000  
##  NA's   :412     NA's   :456     NA's   :509                     
##        id        
##  Min.   :   1.0  
##  1st Qu.: 659.2  
##  Median :1317.5  
##  Mean   :1317.5  
##  3rd Qu.:1975.8  
##  Max.   :2634.0  
## 
```

```r
library(data.table)
library(dplyr)
```

```
## 
## Attaching package: 'dplyr'

## The following objects are masked from 'package:data.table':
## 
##     between, first, last

## The following object is masked from 'package:kableExtra':
## 
##     group_rows

## The following objects are masked from 'package:stats':
## 
##     filter, lag

## The following objects are masked from 'package:base':
## 
##     intersect, setdiff, setequal, union
```

```r
long_df <- melt(setDT(df), id.vars = c("age", "gender", "BMI_base", "BMI_10yrs",
                                         "cigarette_base", "cholst_base", "dead",
```

```r
                                    "id"), variable.name = "year")
long_df$year = case_when(
  long_df$year == "cholst_2" ~ 2,
  long_df$year == "cholst_4" ~ 4,
  long_df$year == "cholst_6" ~ 6,
  long_df$year == "cholst_8" ~ 8,
  long_df$year == "cholst_10" ~ 10)

long_df$age_current = long_df$year + long_df$age
```

```r
df_subset = na.omit(long_df)
library(lme4)
```

```
## Loading required package: Matrix
```

```r
lmm_q3 = lmer(value~ age+age_current+gender+gender*age_current + BMI_base
           +(1+age_current|id), data = df_subset, REML = T)
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## Model failed to converge with max|grad| = 24.9387 (tol = 0.002, component 1)

## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, : Model is nearly unider
##  - Rescale variables?;Model is nearly unidentifiable: large eigenvalue ratio
##  - Rescale variables?
```

```r
summary(lmm_q3)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula:
## value ~ age + age_current + gender + gender * age_current + BMI_base +
##     (1 + age_current | id)
##    Data: df_subset
##
## REML criterion at convergence: 105203
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -5.2610 -0.5380 -0.0279  0.5070 11.7282
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  id       (Intercept) 1313.626 36.244
##           age_current    0.497  0.705   -0.49
##  Residual              453.886 21.305
## Number of obs: 11024, groups:  id, 2499
##
## Fixed effects:
##                     Estimate Std. Error t value
## (Intercept)         269.0443    10.7451  25.039
## age                  -1.8044     0.1198 -15.067
## age_current           0.4376     0.1982   2.208
## gender              -80.3200     5.8028 -13.841
## BMI_base              0.7386     0.1804   4.094
## age_current:gender    1.7039     0.1195  14.264
##
## Correlation of Fixed Effects:
```

```
##             (Intr) age    ag_crr gender BMI_bs
## age         -0.185
## age_current -0.790 -0.231
## gender      -0.864 -0.017  0.894
## BMI_base    -0.427 -0.169  0.082  0.106
## ag_crrnt:gn  0.834  0.003 -0.926 -0.966 -0.088
## optimizer (nloptwrap) convergence code: 0 (OK)
## Model failed to converge with max|grad| = 24.9387 (tol = 0.002, component 1)
## Model is nearly unidentifiable: very large eigenvalue
##  - Rescale variables?
## Model is nearly unidentifiable: large eigenvalue ratio
##  - Rescale variables?
```

```r
library(geepack)
gee_q3 = geeglm(value~ age+age_current+gender+gender*age_current + BMI_base, id = id, data = df_subset,
summary(gee_q3)
```

```
##
## Call:
## geeglm(formula = value ~ age + age_current + gender + gender *
##     age_current + BMI_base, data = df_subset, id = id, corstr = "exchangeable")
##
##  Coefficients:
##                     Estimate   Std.err     Wald Pr(>|W|)
## (Intercept)        301.14014   8.00244 1416.096  < 2e-16 ***
## age                 -1.95623   0.14834  173.909  < 2e-16 ***
## age_current         -0.02604   0.19975    0.017    0.896
## gender             -96.06761   4.55120  445.555  < 2e-16 ***
## BMI_base             0.60735   0.10361   34.361 4.58e-09 ***
## age_current:gender   2.02971   0.09215  485.121  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation structure = exchangeable
## Estimated Scale Parameters:
##
##             Estimate Std.err
## (Intercept)     1746   37.78
##   Link = identity
##
## Estimated Correlation Parameters:
##       Estimate Std.err
## alpha        0       0
## Number of clusters:   11024  Maximum cluster size: 1
```

```r
results_q3a_1 <- summary(lmm_q3)$coefficients[,c(1,2)]
results_q3a_2 <- summary(gee_q3)$coefficients[,c(1,2)]
```

```r
kbl(cbind(round(results_q3a_1,4),round(results_q3a_2,4)),
  caption = "Parameter Estimate and Standar Error Comparison") %>%
  add_header_above(c(" "=1, "LMM"=2, "GEE1"=2)) %>%
  kableExtra::kable_styling(position = "center")
```

Per the comparisons, the results are generally similar in trend, including the intercept, age, gender, BMI, and the interaction term. The standard errors are similar yet different by some amounts. The only discrepancy lies in the age_current variable, which indicates how long the person has been in the study. Two models

Table 4: Parameter Estimate and Standar Error Comparison

	LMM		GEE1	
	Estimate	Std. Error	Estimate	Std.err
(Intercept)	269.0443	10.7451	301.1401	8.0024
age	-1.8044	0.1198	-1.9562	0.1483
age_current	0.4376	0.1982	-0.0260	0.1997
gender	-80.3200	5.8028	-96.0676	4.5512
BMI_base	0.7386	0.1804	0.6073	0.1036
age_current:gender	1.7039	0.1195	2.0297	0.0922

show similar standard errors but the estimates are different in sign.

Question 3.2

The data generating process is similar to HW2. I recycled some of the code. To reiterate the logic: Under the random intercept model, since $var(Y) = 1 = \theta + \sigma^2$ and $corr(Y_{ij}, Y_{ik}) = \rho = \frac{\theta}{\theta+\sigma^2}$, we solve the equations and get $\theta = \rho$ and $\sigma^2 = 1 - \rho$. In this way, we can generate the x, e, b seperately and use a linear relationship we choose to generate the y values, without the need to sample y directly but achieving the same results.

```
library(lme4)

# Set beta to 0.5 and 1
beta1 = 0.5
beta0 = 1
p = 0.5

params <- expand.grid(
  m = c(5,10,20, 50,100), # individuals
  n = c(5,10,20) # observations per individual
)

# For testing
#m = 10
#n = 5

gen.one <- function(m,n){

  total = m*n

  # Generate the variables
  x = rnorm(total, 0, 1)
  b = rep(rnorm(m, mean = 0, sd = sqrt(p)),n)
  e = rnorm(total, mean = 0, sd = sqrt(1-p))
  y = beta0 + beta1*x + b + e

  # LMM
  lmm = lmer(y ~ x + (1|b), REML = T)
  gee = geeglm(y~ x, id = b, corstr = "exchangeable")
  ols = lm(y ~ x + b)

  # Estimate variance for efficiency
  lmm_var0 = vcov(lmm)[1,1]
```

```
  lmm_var1 = vcov(lmm)[2,2]
  gee_var0 = vcov(gee)[1,1]
  gee_var1 = vcov(gee)[2,2]
  ols_var0 = vcov(ols)[1,1]
  ols_var1 = vcov(ols)[2,2]

  # Estimate coefficients
  lmm_coef0 = fixef(lmm)[1]
  lmm_coef1 = fixef(lmm)[2]
  gee_coef0 = coef(gee)[1]
  gee_coef1 = coef(gee)[2]
  ols_coef0 = coef(ols)[1]
  ols_coef1 = coef(ols)[2]

  # Estimate bias
  lmm_bias0 = lmm_coef0 - beta0
  lmm_bias1 = lmm_coef1 - beta1
  gee_bias0 = gee_coef0 - beta0
  gee_bias1 = gee_coef1 - beta1
  ols_bias0 = ols_coef0 - beta0
  ols_bias1 = ols_coef1 - beta1

  # lmm_coef0 = lmm_coef0, lmm_coef1 = lmm_coef1,
  # gee_coef0 = gee_coef0, gee_coef1 = gee_coef1,
  # ols_coef0 = ols_coef0, ols_coef1 = ols_coef1,
  return(data.frame(m = m, n = n,
                    lmm_var0 = lmm_var0, lmm_var1 = lmm_var1,
                    gee_var0 = gee_var0, gee_var1 = gee_var1,
                    ols_var0 = ols_var0, ols_var1 = ols_var1,
                    lmm_bias0 = lmm_bias0, lmm_bias1 = lmm_bias1,
                    gee_bias0 = gee_bias0, gee_bias1 = gee_bias1,
                    ols_bias0 = ols_bias0, ols_bias1 = ols_bias1
                    ) )
}
```

```
nrep = 1000

simulation <- do.call(rbind, lapply(c(1:nrow(params)), function(i){
  m <- params$m[i]
  n <- params$n[i]
  res <- do.call(rbind, lapply(c(1:nrep), function(nrep){
  gen.one(m,n)
  }))
  mean_res <- colMeans(res)
}))
```

```
## Warning in checkConv(attr(opt, "derivs"), opt$par, ctrl = control$checkConv, :
## Model failed to converge with max|grad| = 0.0414803 (tol = 0.002, component 1)
```

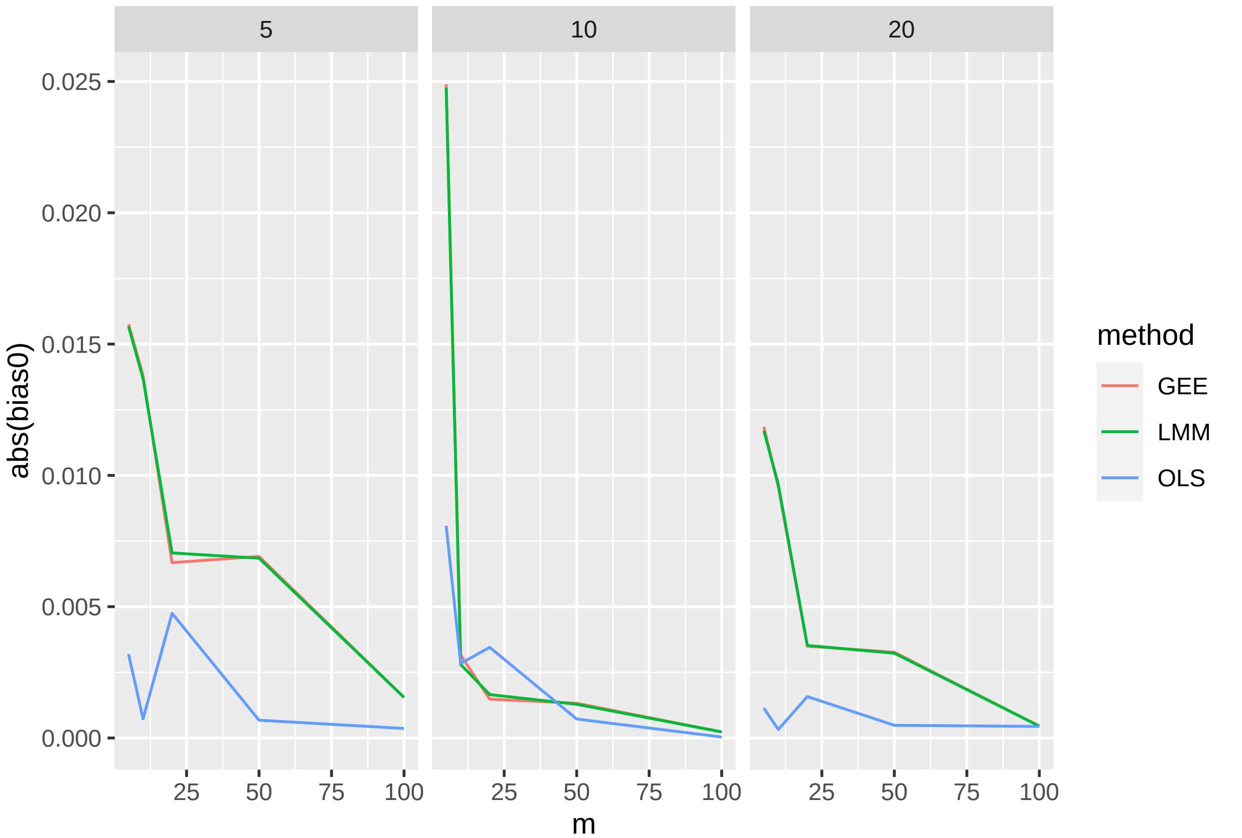
```
simulation_res = as.data.frame(simulation)
```

```
simulation_res_reshaped = reshape(simulation_res, direction="long",
        varying=list(c("lmm_var0","gee_var0", "ols_var0"),
                     c("lmm_var1","gee_var1", "ols_var1"),
                     c("lmm_bias0","gee_bias0", "ols_bias0"),
```

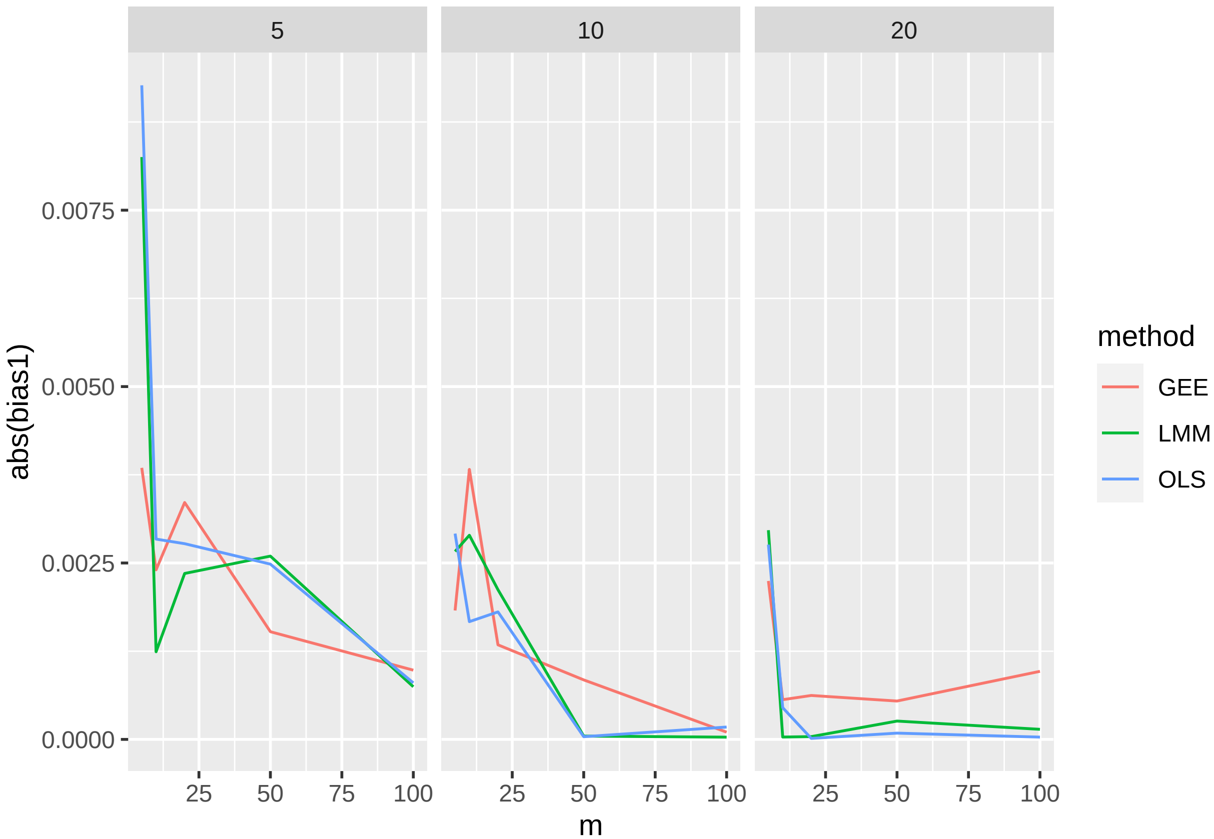
```
                     c("lmm_bias1","gee_bias1", "ols_bias1")),
        v.names=c("var0","var1","bias0", "bias1"))
```

```
simulation_res_reshaped$method = case_when(
  simulation_res_reshaped$time == 1 ~ "LMM",
  simulation_res_reshaped$time == 2 ~ "GEE",
  simulation_res_reshaped$time == 3 ~ "OLS")
```

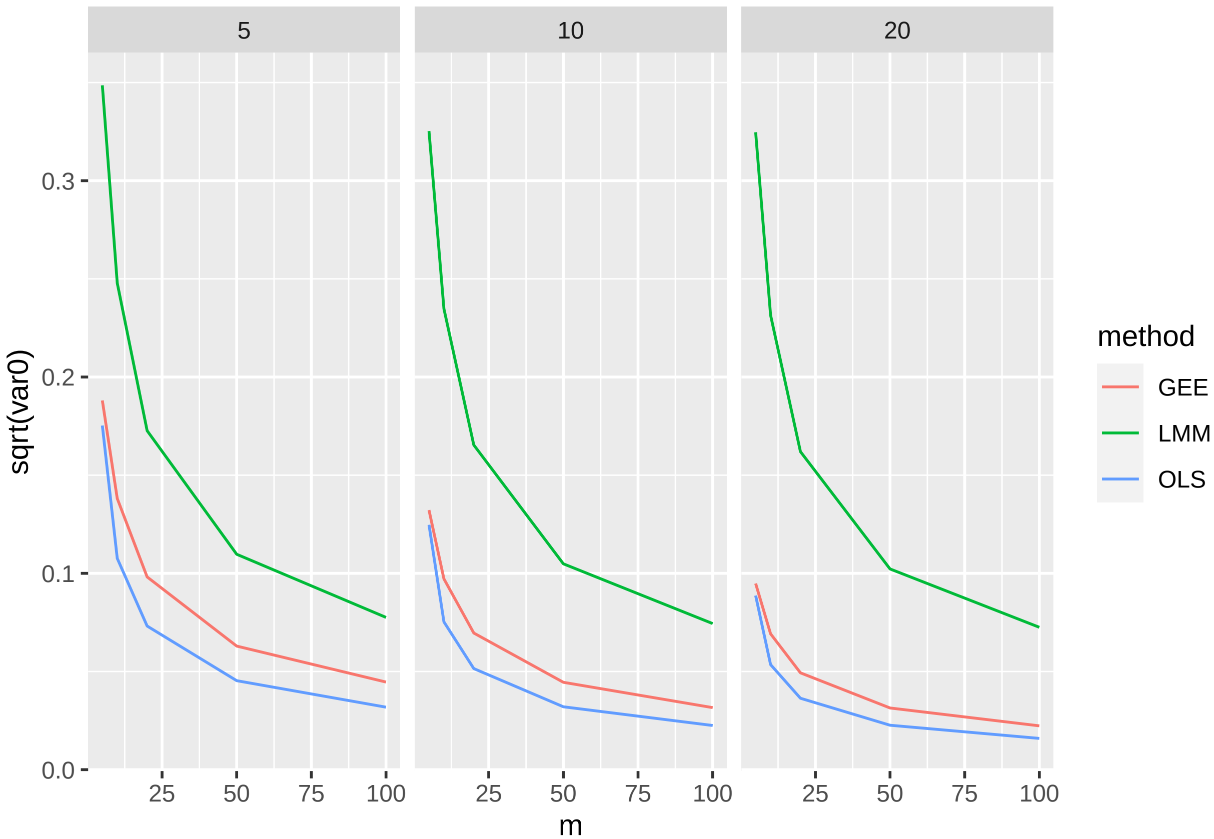
```
library(ggplot2)
ggplot(data=simulation_res_reshaped, aes(x=m, y=abs(bias0), color = method))+geom_line()+
  facet_grid(cols=vars(n))
```



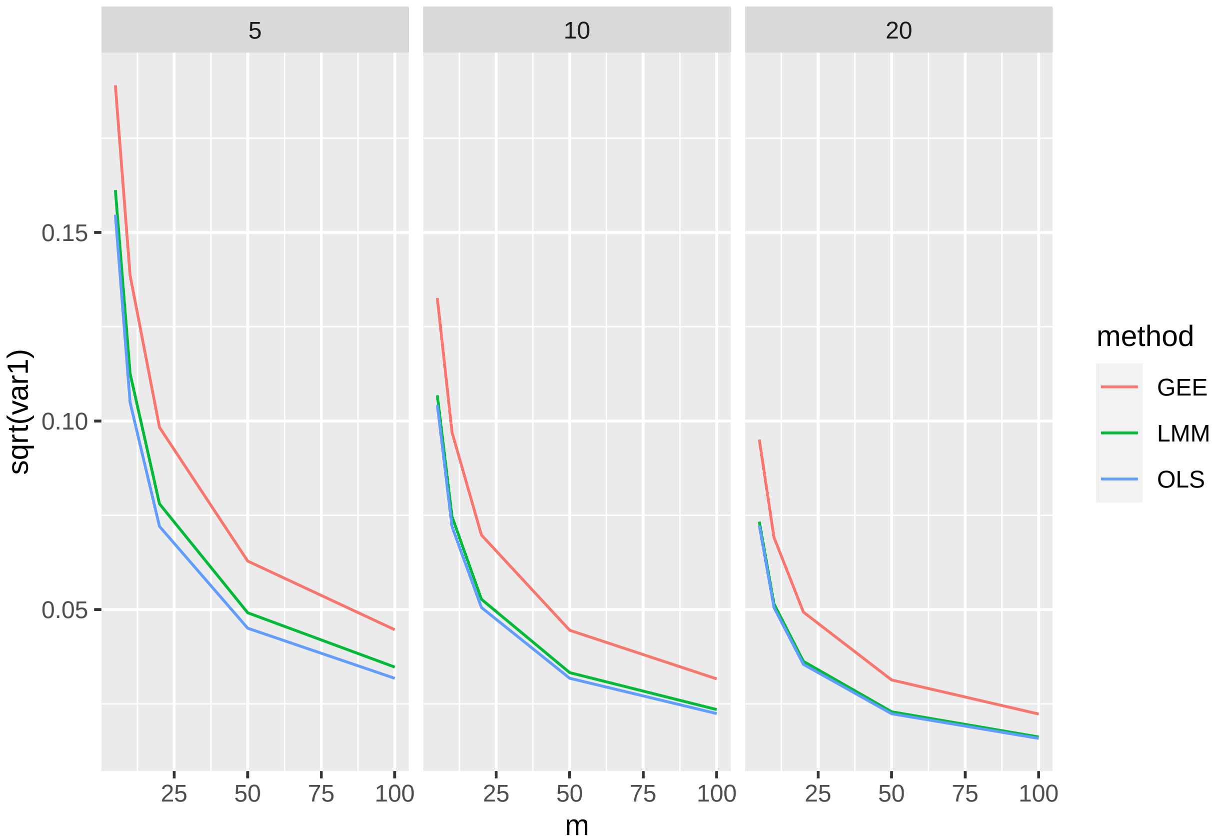

```
ggplot(data=simulation_res_reshaped, aes(x=m, y=abs(bias1), color = method))+geom_line()+
  facet_grid(cols=vars(n))
```




```
ggplot(data=simulation_res_reshaped, aes(x=m, y=sqrt(var0), color = method))+geom_line()+
  facet_grid(cols=vars(n))
```




```
ggplot(data=simulation_res_reshaped, aes(x=m, y=sqrt(var1), color = method))+geom_line()+
  facet_grid(cols=vars(n))
```



From the visualizations, we can compare the results of the three models, using different sizes of m and n. We have generated 1000 replicates of the simulation, and the estimates are simply an average of the 1000 models obtained through simulations.

The LMM successfully takes into account the random and fixed effects. Regardless of the covariance structure, we can obtain identical results, compared with GEE's. It however, requires that the specification of the fixed and random effects to be correct to be able to calculate the correct results.

The GEE, on the other hand, provides a very flexible approach because it can specify various variance structures. For simplicity, I have only used the exchangeable structure for this HW. The drawback is probably that when the variance structure is misspecified, results can be off a lot.

OLS has the advantage being easy to implement and easy to understand, although it does not offer flexible interpretations especially for the random effects, since it has treated it as a fixed effect.

Across the methods, from my simulation results, the ols has the lowest variance for both beta parameters. The biases, however, show more diverse results for different methods. But the trend is generally that as sample size increases, all the methods show improvement in bias and variance.

Question 4

To propose a marginal or population average model, I start by setting their means. To be simple, Y_{i1} can follow normal distribution and Y_{i2} can follow Bernoulli distribution. $\mu_{i1} = X_i^T\beta_1$ and $\mu_{i2} = probit(X_i^T\beta_2)$. I can further specify the variances to be $var_1 = \sigma^2$ say 1, and $var_2 = probit(X_i^T\beta_2)(1 - probit(X_i^T\beta_2))$ per the Bernoulli distribution.

We can then specify the correlation between Y_{i1} and Y_{i2} to be ρ, then we would have the correlation matrix between the two. Then we would be able to easily model using GEE. $\sum_{i=1}^m D_i^T V_i^{-1}(Y_i - \mu_i) = 0$. Everything will be easy just like a usual GEE.