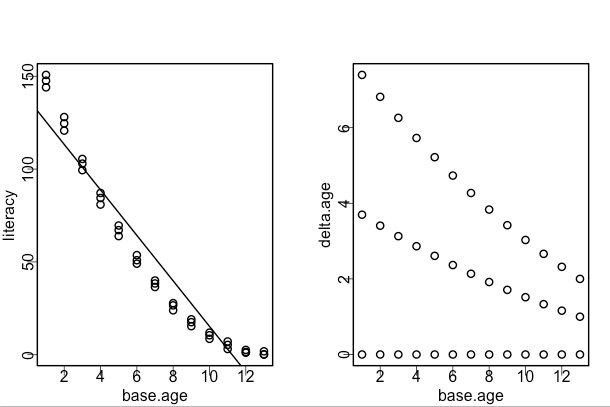
1. (a) Since we know if the mean model is correct, the estimates of beta will be unbiased and the model-based standard errors will also be valid, for here, we just need f( which leads to the correct mean model.

(b) We can fix the follow-up times, that is, use the same fixed time for follow observations. In other cases, if we set the follow-up times as a linear function to the baseline age, the estimates of beta will be unbiased since the baseline age is in the model already.

(c) Based on the plot 1 , we can see that the computer literacy is not in a linear relationship with the baseline age which violate the condition in part a.

The second plot shows the follow-up ages with the baseline age. Firstly, the older the participants, the sooner the follow-up visits are. Thus the follow-up visits are not fixed. From the plot, it is also clear that the follow-up visits are not linear in the baseline age. Therefore, the condition in part b is not satisfied.

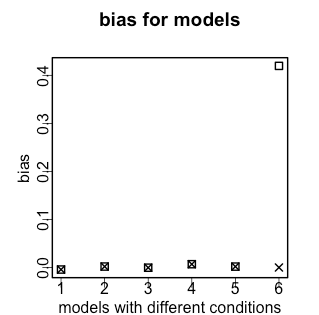


(d) To eliminate the influence of the baseline age, we can use the fixed effect model in which we set a fixed intercept to each individual. The result of this model and the comparison between it and the exposure partioning model are shown below:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Estimates | Std | CI |
| Exposure-partioning | 1.25 | 1.06 | (-0.84,3.33) |
| Fixed effect model | 1.01 | 0.034 | (0.94,1.07) |

By comparing , we can see that the exposure portioning model is not correctly built while the fixed effect model can provide less standard error. The confidence interval for them also show that the fixed effect model is better than the exposure portioning as the later one give a wide CI.

(e) For simulation, we choose two kinds of f function: one is linear on the baseline age and the other is nonlinear. We also choose three kinds of design matrix: one is fixed increasing age, second is a linear increasing of baseline, the third one is a nonlinear one. Based on these, we can generate our y and choose beta\_l = 4 here. Therefore, for the exposure partioning model we have 6 kinds and it is the same for the fixed effect model. The results are shown below:



From the bias plot, we see that when the first condition in part a is satisfied, that is, the mean model is correct, then the exposure partioning model can give unbiased estimates no matter for which type of design matrix. If the condition is part b is satisfied, as in model 4 and 5, though the f function is nonlinear, the estimate is still unbiased. However, when the two conditions are violated, the estimate is biased. On the contrary, for the fixed effect models, no matter in which cases, it can provide the unbiased estimate.

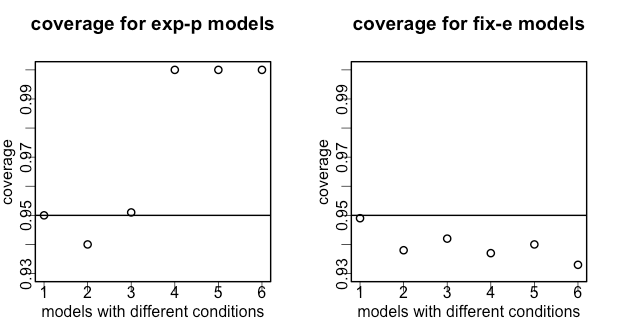
(f) The average standard error for models satisfied condition in part b but not for part a are:

|  |  |  |
| --- | --- | --- |
| model | Nonlinear&fixed | Nonlinear&linear |
| Exposure-p model | 7.60 | 0.91 |
| Fixed-e model | 0.19 | 0.02 |

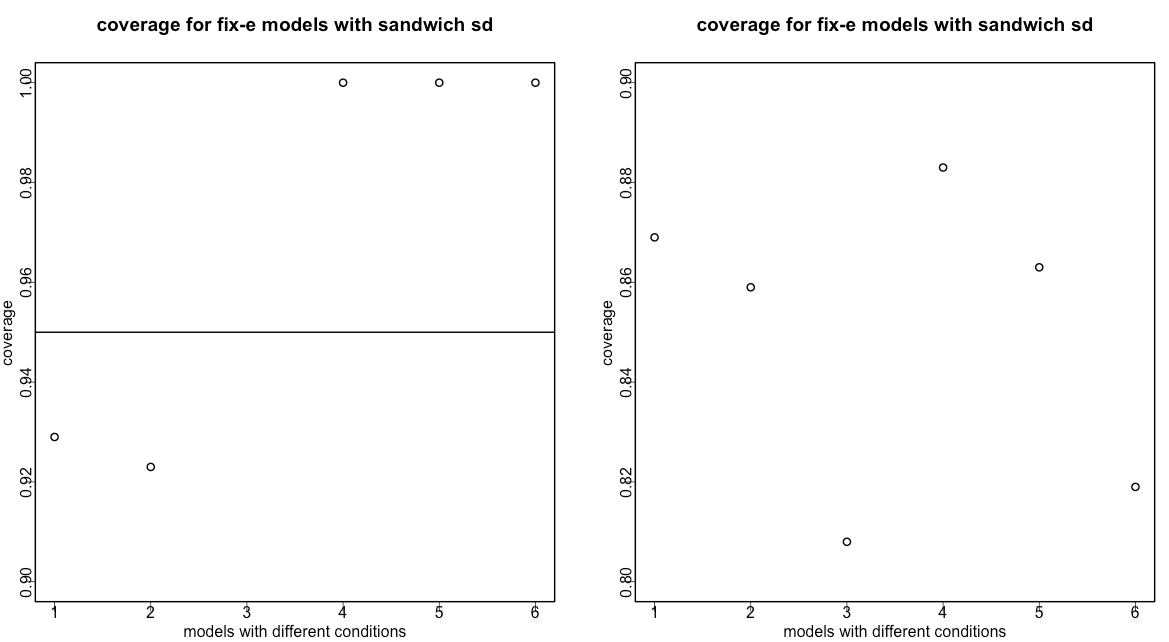
We notice that the estimated standard errors are a little too big especially for the exposure portioning models.

In order to detect the standard error, we can also build the 95% CI and compute the average coverage of different models.

The average coverage of the simulation is:



It is clear that for the exposure portioning models, the CI for model 4 and 5 are over cover the 95% interval. But the fixed effect model is fine. To use the sandwich instead, we can get:



Thus, the sandwich standard errors cannot fix this problem as the coverage does not change.

(g) Since the mean model is mis-specified, the estimating equations will be biased. That is, the theory of sandwich estimators is violated which leads to the problem in f.

2. (a) By fitting the model by hand via the direct optimization method, we get the results:

|  |  |  |  |
| --- | --- | --- | --- |
| method | estimates | sd | CI |
| Fix-effect model | 1.006 | 0.034 | (0.973,1.039) |
| Direct-optim | 1.006 | 0.027 | (0.980,1.032) |

From the table, it is clear that the estimates of beta-l is almost the same while the CI and standard error are a little different from each other.

(b) There is difference in the estimated standard error from the two models. Since the model of part d can give the correct coverage of estimates in simulation, we can guess the standard error from the direct optimization might be wrong.

(c) The direct optimization does not consider the degrees of freedom here. The MLE for the direct optimization is biased as:

Thus the error from the direct optimization is smaller than the unbiased one with a factory of 2/3.

(d) We could modify the design by increasing the observations for each individual. We can also increase the number of participants and get more observations for each of the base ages.

3. (a) E[

(b) First we know, Var

At the same time, if we find the relation between :

where D is a matrix,

and E[

We can get

Thus, Dx=0=.

Using the same relationship, we can get:

Var[

Thus Var[V is positive semi-definite and is also a positive semi-definite matrix. If D is a zero matrix which means , the variances of them will be the same.

Appendix:

##problem 1 part c##

# make the plot

par(mfrow=c(1,2))

plot(base.age,literacy)

model<-lm(literacy~base.age)

abline(model1)

plot(base.age,delta.age)

##part d##

model.p<-lm(literacy~base.age+delta.age)

summary(model.p)

model.m<-lm(literacy~0+factor(subj)+delta.age)

summary(model.m)

#get the estimates and corresponding CI

coef.p<-coef(model.p)[3]

sd.p<-summary(model.p)$coef[3,2]

ci.p<-coef.p+c(-1,1)\*qnorm(0.975)\*sd.p

coef.m<-coef(model.m)[14]

sd.m<-summary(model.m)$coef[14,2]

ci.m<-coef.m+c(-1,1)\*qnorm(0.975)\*sd.m

#part e

library(sandwich)

#define design matrix: fix, linear or nonlinear: condition b

base.age<-rep(c(1:13),3)

delta1<-c(rep(0,13),rep(1,13),rep(2,13))

delta2<-c(rep(0,13),1+0.5\*c(1:13),1+0.8\*c(1:13))

delta3<-c(rep(0,13),1+0.5\*c(1:13)^2,1+0.8\*c(1:13)^2)

#define f function linear or nonlinear: condition a

f1<-1+5\*base.age

f2<-1+5\*base.age^2

#define beta l

beta\_l<-4

cov<- function(beta,ci){ifelse(beta>ci[1]&beta<ci[2],1,0) }

#simulation

coef.p1<-coef.p2<-coef.p3<-coef.p4<-coef.p5<-coef.p6<-c()

coef.m1<-coef.m2<-coef.m3<-coef.m4<-coef.m5<-coef.m6<-c()

sd.p1<-sd.p2<-sd.p3<-sd.p4<-sd.p5<-sd.p6<-c()

sd.m1<-sd.m2<-sd.m3<-sd.m4<-sd.m5<-sd.m6<-c()

yn.p1<-yn.p2<-yn.p3<-yn.p4<-yn.p5<-yn.p6<-c()

yn.m1<-yn.m2<-yn.m3<-yn.m4<-yn.m5<-yn.m6<-c()

yn.p11<-yn.p22<-yn.p33<-yn.p44<-yn.p55<-yn.p66<-c()

yn.m11<-yn.m22<-yn.m33<-yn.m44<-yn.m55<-yn.m66<-c()

for (i in 1:1000){

#get the y

y1<-f1+beta\_l\*delta1+rnorm(39,0,1) #a & b

y2<-f1+beta\_l\*delta2+rnorm(39,0,1) #a & b

y3<-f1+beta\_l\*delta3+rnorm(39,0,1) #a & !b

y4<-f2+beta\_l\*delta1+rnorm(39,0,1) #!a & b

y5<-f2+beta\_l\*delta2+rnorm(39,0,1) #!a & b

y6<-f2+beta\_l\*delta3+rnorm(39,0,1) #!a & !b

#exposure partioning models

model.p1<-lm(y1~base.age+delta1)

model.p2<-lm(y2~base.age+delta2)

model.p3<-lm(y3~base.age+delta3)

model.p4<-lm(y4~base.age+delta1)

model.p5<-lm(y5~base.age+delta2)

model.p6<-lm(y6~base.age+delta3)

#fixed effect models

model.m1<-lm(y1~0+factor(base.age)+delta1)

model.m2<-lm(y2~0+factor(base.age)+delta2)

model.m3<-lm(y3~0+factor(base.age)+delta3)

model.m4<-lm(y4~0+factor(base.age)+delta1)

model.m5<-lm(y5~0+factor(base.age)+delta2)

model.m6<-lm(y6~0+factor(base.age)+delta3)

#get the estimates and corresponding Ci

coef.p1[i]<-coef(model.p1)[3]

sd.p1[i]<-summary(model.p1)$coef[3,2]

ci.p1<-coef.p1[i]+c(-1,1)\*qnorm(0.975)\*sd.p1[i]

yn.p1[i]<-cov(4,ci.p1)

sd<-sqrt(diag(vcovHC(model.p1,type="HC0")))[3]

ci.p1<-coef.p1[i]+c(-1,1)\*qnorm(0.975)\*sd

yn.p11[i]<-cov(4,ci.p1)

#similar for all

coef.p2[i]<-coef(model.p2)[3]

sd.p2[i]<-summary(model.p2)$coef[3,2]

ci.p2<-coef.p2[i]+c(-1,1)\*qnorm(0.975)\*sd.p2[i]

yn.p2[i]<-cov(4,ci.p2)

sd<-sqrt(diag(vcovHC(model.p2,type="HC0")))[3]

ci.p1<-coef.p2[i]+c(-1,1)\*qnorm(0.975)\*sd

yn.p22[i]<-cov(4,ci.p1)

coef.p3[i]<-coef(model.p3)[3]

sd.p3[i]<-summary(model.p3)$coef[3,2]

ci.p3<-coef.p3[i]+c(-1,1)\*qnorm(0.975)\*sd.p3[i]

yn.p3[i]<-cov(4,ci.p3)

sd<-sqrt(diag(vcovHC(model.p3,type="HC0")))[3]

ci.p<-coef.p3[i]+c(-1,1)\*qnorm(0.975)\*sd

yn.p33[i]<-cov(4,ci.p)

coef.p4[i]<-coef(model.p4)[3]

sd.p4[i]<-summary(model.p4)$coef[3,2]

ci.p4<-coef.p4[i]+c(-1,1)\*qnorm(0.975)\*sd.p4[i]

yn.p4[i]<-cov(4,ci.p4)

sd<-sqrt(diag(vcovHC(model.p5,type="HC0")))[3]

ci.p<-coef.p4[i]+c(-1,1)\*qnorm(0.975)\*sd

yn.p44[i]<-cov(4,ci.p)

coef.p5[i]<-coef(model.p5)[3]

sd.p5[i]<-summary(model.p5)$coef[3,2]

ci.p5<-coef.p5[i]+c(-1,1)\*qnorm(0.975)\*sd.p5[i]

yn.p5[i]<-cov(4,ci.p5)

sd<-sqrt(diag(vcovHC(model.p5,type="HC0")))[3]

ci.p<-coef.p5[i]+c(-1,1)\*qnorm(0.975)\*sd

yn.p55[i]<-cov(4,ci.p)

coef.p6[i]<-coef(model.p6)[3]

sd.p6[i]<-summary(model.p6)$coef[3,2]

ci.p6<-coef.p6[i]+c(-1,1)\*qnorm(0.975)\*sd.p6[i]

yn.p6[i]<-cov(4,ci.p6)

sd<-sqrt(diag(vcovHC(model.p6,type="HC0")))[3]

ci.p<-coef.p6[i]+c(-1,1)\*qnorm(0.975)\*sd

yn.p66[i]<-cov(4,ci.p)

coef.m1[i]<-coef(model.m1)[14]

sd.m1[i]<-summary(model.m1)$coef[14,2]

ci.m1<-coef.m1[i]+c(-1,1)\*qnorm(0.975)\*sd.m1[i]

yn.m1[i]<-cov(4,ci.m1)

sd<-sqrt(diag(vcovHC(model.m1,type="HC0")))[14]

ci.m<-coef.m1[i]+c(-1,1)\*qnorm(0.975)\*sd

yn.m11[i]<-cov(4,ci.m)

coef.m2[i]<-coef(model.m2)[14]

sd.m2[i]<-summary(model.m2)$coef[14,2]

ci.m2<-coef.m2[i]+c(-1,1)\*qnorm(0.975)\*sd.m2[i]

yn.m2[i]<-cov(4,ci.m2)

sd<-sqrt(diag(vcovHC(model.m2,type="HC0")))[14]

ci.m<-coef.m2[i]+c(-1,1)\*qnorm(0.975)\*sd

yn.m22[i]<-cov(4,ci.m)

coef.m3[i]<-coef(model.m3)[14]

sd.m3[i]<-summary(model.m3)$coef[14,2]

ci.m3<-coef.m3[i]+c(-1,1)\*qnorm(0.975)\*sd.m3[i]

yn.m3[i]<-cov(4,ci.m3)

sd<-sqrt(diag(vcovHC(model.m3,type="HC0")))[14]

ci.m<-coef.m3[i]+c(-1,1)\*qnorm(0.975)\*sd

yn.m33[i]<-cov(4,ci.m)

coef.m4[i]<-coef(model.m4)[14]

sd.m4[i]<-summary(model.m4)$coef[14,2]

ci.m4<-coef.m4[i]+c(-1,1)\*qnorm(0.975)\*sd.m4[i]

yn.m4[i]<-cov(4,ci.m4)

sd<-sqrt(diag(vcovHC(model.m4,type="HC0")))[14]

ci.m<-coef.m4[i]+c(-1,1)\*qnorm(0.975)\*sd

yn.m44[i]<-cov(4,ci.m)

coef.m5[i]<-coef(model.m5)[14]

sd.m5[i]<-summary(model.m5)$coef[14,2]

ci.m5<-coef.m5[i]+c(-1,1)\*qnorm(0.975)\*sd.m5[i]

yn.m5[i]<-cov(4,ci.m5)

sd<-sqrt(diag(vcovHC(model.m5,type="HC0")))[14]

ci.m<-coef.m5[i]+c(-1,1)\*qnorm(0.975)\*sd

yn.m55[i]<-cov(4,ci.m)

coef.m6[i]<-coef(model.m6)[14]

sd.m6[i]<-summary(model.m6)$coef[14,2]

ci.m6<-coef.m6[i]+c(-1,1)\*qnorm(0.975)\*sd.m6[i]

yn.m6[i]<-cov(4,ci.m6)

sd<-sqrt(diag(vcovHC(model.m6,type="HC0")))[14]

ci.m<-coef.m6[i]+c(-1,1)\*qnorm(0.975)\*sd

yn.m66[i]<-cov(4,ci.m)

}

#get the average number

bias.p<-colMeans(cbind(coef.p1,coef.p2,coef.p3,coef.p4,coef.p5,coef.p6))-4

bias.m<-colMeans(cbind(coef.m1,coef.m2,coef.m3,coef.m4,coef.m5,coef.m6))-4

sd.p<-colMeans(cbind(sd.p1,sd.p2,sd.p3,sd.p4,sd.p5,sd.p6))

sd.m<-colMeans(cbind(sd.m1,sd.m2,sd.m3,sd.m4,sd.m5,sd.m6))

yn.p<-colMeans(cbind(yn.p1,yn.p2,yn.p3,yn.p4,yn.p5,yn.p6))

yn.m<-colMeans(cbind(yn.m1,yn.m2,yn.m3,yn.m4,yn.m5,yn.m6))

yn.pp<-colMeans(cbind(yn.p11,yn.p22,yn.p33,yn.p44,yn.p55,yn.p66))

yn.mm<-colMeans(cbind(yn.m11,yn.m22,yn.m33,yn.m44,yn.m55,yn.m66))

#make the plots for bias

plot(bias.p~c(1:6),pch=0,xlab="models with different conditions",

ylab="bias",main="bias for models")

points(bias.m,pch=4)

legend("topleft",legend=c("e-p model","fix-e model"),pch=c(0,4))

#make the plot for the coverages

plot(yn.p~c(1:6),ylim=c(0.93,1),xlab="models with different conditions",

ylab="coverage",main="coverage for exp-p models")

abline(h=0.95)

plot(yn.m~c(1:6),ylim=c(0.93,1),xlab="models with different conditions",

ylab="coverage",main="coverage for fix-e models")

abline(h=0.95)

plot(yn.pp~c(1:6),ylim=c(0.90,1),xlab="models with different conditions",

ylab="coverage",main="coverage for fix-e models with sandwich sd")

abline(h=0.95)

plot(yn.mm ~c(1:6),ylim=c(0.8,0.9),xlab="models with different conditions",

ylab="coverage",main="coverage for fix-e models with sandwich sd")

abline(h=0.95)

##problem 2 part a

X<-matrix(0, nrow=39, ncol=13)

for (i in 1:13){X[((i-1)\*3+1):(i\*3),i]<-rep(1,3)}

#the design matrix is:

design<-cbind(X,mydata$delta.age,mydata$literacy)

#define the loglikelihood function

loglikeli<-function(data,parameter){

beta\_l<-parameter[14]

sigma\_s<-parameter[15]

loglik<-sum(dnorm(data[,15],mean=data[,1:13]%\*%parameter[1:13]+beta\_l\*data[,14],

sd=sigma\_s,log=T))

return(-loglik)

}

model<-nlm(loglikeli,c(160,140,120,100,80,65,50,35,25,15,10,5,1,1,1),hessian=T,data=design)

#the estimates for optimization

model$est[14]

#standard error from it

sqrt(diag(solve(model$hess)))[14]