A Bayesian Approach to the Measurement Error Problem in Regression

Ananda Jayawardhana, PhD

Pittsburg State University

Pittsburg, KS 66762

Abstract: In this simulation study, simple linear regression and logistic regression were studied under measurement error of the predictor variables. Measurement errors often occur in data of scientific experiments as well as survey data in health and social sciences especially in self- reported surveys. For example, to study the effect of dietary fiber intake on coronary heart disease, researches have to rely on the self-reported data which are prone to have measurement error. Many attempts have been made over the years to study the effect of measurement errors and their effect on parameter estimates. In this study we modeled with known predictor variables. Using R, R2OpenBUGS, and OpenBUGS, we simulated data sets and added random errors and studied the parameter estimates. Overall, the introduction of small errors to predictor variables did not have a large effect on parameter estimates for linear regression but parameters were consistently over-estimated in logistic regression. Simulation results and simulation codes will be presented in this paper.

Key Words: Measurement Error, Bayesian, Regression, Logistic Regression

1. Introduction

Linear regression and logistic regression are popular statistical methods used to analyze data from studies in many fields. Often one or more predictors are measured with error. Measurement error in the explanatory variables could result in biased estimates of parameters which could lead to incorrect conclusions in hypotheses testing and loss of power in testing. Several methods have been proposed to correct this problem in literature. Fuller (1987) provides a summary of methods used for linear regression and Carrol, Rupert, and Stefanski (1995) have given a summary of such methods in non-linear models. Schmid and Rosner (1993) reports a study of Bayesian analysis of logistic regression for predictors with systematic measurement errors. They apply their method to study the risk of alcohol consumption on breast cancer using the Nurses' Health Study data and report that the resulting risk estimates differ sharply from those computed by standard logistic regression that ignores measurement error. Thorensen and Laake (2000) reports a simulation study of comparison of four estimation methods: regression calibration method, probit maximum likelihood as an approximation to the logistic maximum likelihood, the exact maximum

likelihood method based on a logistic model, and the naive estimator, which is the result of simply ignoring the fact that some of the explanatory variables are measured with error. Rabe-Hesketh, Pickles, and Skrondal (2003) reports a study in which they relaxed the assumption that covariates and errors are normally distributed. They have developed a nonparametric maximum likelihood estimation method. To demonstrate their method, they have used the method in estimating the effect of dietary fiber intake on coronary heart disease. Buonaccorsi, Romeo, and Thoresen (2018) have developed new methods for model based bootstrapping when correcting for measurement error in logistic regression with replicate measures. They used the methods to estimate the occurrence of heart disease as a function of true mean cholesterol using subset of individuals from Framingham Heart Study.

2. Problem

In linear regression, measurement error in the explanatory variables could result in biased estimates of parameters which could lead to incorrect hypotheses testing results and loss of power in testing. One, well known example of measurement error is self-reported nutrient intake. The objective of this study is to find out the effect of measurement error on simple linear regression and logistic regression models with one explanatory variable.

2.1 Classical and Berkson Measurement Error Models

We denote the response variable by Y, the true covariate subject to measurement error by X, and covariates measured exactly by Z, and proxy for X by W. There are two well-known measurement error models. Classical measurement error model is given by $W_{ij} = X_i + U_{ij}; i = 1,2, \dots, n \; and \; j = 1,2, \dots, k$ with $E(U_{ij}|X_i) = 0$ and $U_{ij}|X_i \sim N(0, \sigma_U^2)$. Berkson measurement error model is given as $X_i = W_i + U_i; i = 1,2, \dots, n$ with $E(U_i|X_i) = 0$.

2.1.1 Model for the Simple Linear Regression Model

We used the model $X_i = \alpha_{0i} + \alpha_{1i} Z_i$, $U_i|X_i \sim N(0, \sigma_U^2)$, $W_i = X_i + U_i$, $Y_i = \alpha + \beta X_i + \varepsilon_i$ and $\varepsilon_i \sim i.i.d. N(0, \sigma_Y^2)$ for $i = 1,2, \dots, n$. In almost all the studies in literature, authors assume a normal distribution for Z_i and therefore X_i have a normal distribution with some variance σ_X^2 determined by the transformation from Z to X. We assumed the distribution of X is discrete uniform (1,n) and also we did not include Z in our models for linear regression as well as logistic regression.

2.1.2 Model for the Logistic Regression Model with One Covariate

We used the model $X_i = \alpha_{0i} + \alpha_{1i} Z_i$, $U_i|X_i \sim N(0, \sigma_U^2)$, $W_i = X_i + U_i$, $logit(\pi_i) = exp(\alpha + \beta X_i)$, and $Y_i = Bernoulli(\pi_i)$ for $i = 1,2, \dots, n$.

2.2 Bayesian Approach

Bayesian logistic regression using Markov Chain Monte Carlo (MCMC) methods were used to estimate the unknown parameters α and β. OpenBUGS which is an open source Bayesian software and R which is another open source software were used for the simulation study. R2OpenBUGS package was used to communicate between R and OpenBUGS programs.

2.3 Model Formulation and Simulation Study

For the simple linear regression, we chose $Z = \{1,2,3, \dots, n\}$ and $X = Z$. Parameters α and β were chosen arbitrarily (only $\alpha = 50, and\ \beta = 2$ are reported). Using R software, we generated the values for $X_i\ and\ Y_i$, then generated U_i using several different values of σ_U; (only $n = 20, \sigma_Y = 5, 10, and\ 20\ and\ \sigma_U =$ 0.10, 0.20, 0.30, 0.50, and 0.70 are reported).

For the logistic regression, we chose $Z = \{1,2,3, \dots, n\}$ and $X = Z$. Parameters α and β were chosen so that $0.03 < \pi_i < 0.98$ (only $\alpha = -3.5$ and $\beta = 0.05$ are reported). Using R software, we generated the values for $X_i, \pi_i, and\ Y_i$, then generated U_i using several different values of σ_{ij}, where $j = 1$. (only $n = 150\ and\ \sigma_{ij} =$ 0.10, 0.15, 0.20, 0.25, and 0.30 are reported).

In both simulations, using R2OpenBUGS software, we used OpenBUGS to estimate the parameters $\alpha\ and\ \beta$ using $Y_i\ and\ W_i$ This process was repeated several thousand times and the averages of the estimates with standard deviations are reported in Appendices A for linear regression and in appendix B for logistic regression. For simple linear regression case we repeated 20,000 times with selecting every 10th one. First selected 1000 were used as burn-in and the remaining 1000 were used for calculations. For the logistic regression case we generated 2000 times and selected all of them with the first 1000 as burn-in leaving the last 1000 for estimation. Simulation codes for linear regression are in Appendices C and D and those for logistic regression are in Appendices E and F.

3. Results and Discussion

3.1 Simple Linear Regression

In all the simulations, the reliability ratio was kept close to one because the study was a preliminary step for studying the logistic regression under the influence of measurement error. Time for simulation was about 2 hours even with 20000 iterations. If the sample size was increased the time to complete a simulation increased proportionately. As for the priors for the slope and intercept parameters, any symmetric distribution with large variance worked equally well, convergence achieved quickly and all the estimates were stable. Relatively small measurement errors did not significantly affect the parameter estimates. Study of the behavior of measurement error on linear regression was not the primary interest of this study.

3.2 Logistic Regression

Simulation for logistic regression had issues with time for simulations and therefore we limited the sample sizes to 150. Thoresen and Laake (2000) use sample sizes 1500 and 150 for their simulations. Hsieh, F.Y. (1989) provides tables of required sample sizes for a given probabilities of type one and type two errors and effect size. In most cases, sample size of 150 is not enough to have accurate estimates for logistic regression and we had issues with the accuracy of estimates. Both parameters were consistently over estimated. Range of the prior distributions had to be kept relatively smaller compared to that for linear regression.

4. Future Research

In the future, most of the study will focus on logistic regression. We plan to use $(1,2,..,n)$ as Z and also generate Z from a normal distribution in order to decrease the reliability ratio. We also plan to use Z as a covariate in the study. Studying the influence of a non-symmetric distribution for, $U_i|X_i$is also of our future interest. A simulation study to replicate the work by Thoresen and Laake (2000) using Bayesian methods is also a future research plan.

Acknowledgements: My gratitude to graduate students Zack Brown and Georgette Searan for doing the simulations.

References:

Buonaccorsi, Romeo, and Thoresen (2018). "Model based bootstrapping when correcting for measurement error with application to logistic regression", Biometrics, 71(1), pp 135-144.

Carol, R. J., Ruppert, D., and Stefanski, L.A. (1995). *Measurement Error in Nonlinear Models*. London, Chapman and Hall.

Fuller, W. A. (1987). *Measurement Error Models.* New York, Wiley.

Hsieh, F. Y. (1989) "Sample size tables for logistic regression", Statistics in Medicine, 8, pp 795-802.

Rabe-Hesketh, S., Pickles, A., and Skrondal, A (2003), "Correcting for covariate measurement error in logistic regression using nonparametric maximum likelihood estimation", Statistical Modeling, 3, pp 215-232.

Schmid, C. H. and Rosner, B (1993). "A Bayesian Approach to Logistic Regression Models Having Measurement Error Following a Mixture Distribution", Statistics in Medicine, 12, pp 1141-1153.

Thorensen, M. and Laake, P (2000). "A Simulation Study od Measurement Error Correction Methods in Logistic Regression", Biometrics, 56, pp 868-872.

Appendix A (Tables for Simple Linear Regression)

Table A.1				
$\sigma_y = 5, \alpha = 50, \beta = 2$, N=20 , N1=1000				
σ_u	$E[\hat{\alpha}]$	$SD[\hat{\alpha}]$	$E[\hat{\beta}]$	$SD[\hat{\beta}]$
0.1	50.0541225	1.0953882	1.9967334	0.1957655
0.2	50.0382497	1.1156248	1.9929306	0.1959311
0.3	50.0438327	1.1238185	2.0018266	0.1976062
0.5	50.0301194	1.1295914	1.9928369	0.2024708
0.7	49.9459320	1.1316196	1.9818123	0.1989929

Table A.2				
$\sigma_y = 10, \alpha = 50, \beta = 2$, N=20, N1=1000				
σ_u	$E[\hat{\alpha}]$	$SD[\hat{\alpha}]$	$E[\hat{\beta}]$	$SD[\hat{\beta}]$
0.1	50.0342771	2.2214075	2.0113456	0.3875044
0.2	50.0134045	2.2506454	1.9978172	0.3854915
0.3	50.0902936	2.2278426	1.9987244	0.3937031
0.5	49.9892244	2.1215323	1.9806764	0.3753549
0.7	50.0940331	2.2735200	1.9898190	0.3886036

Table A.3				
$\sigma_y = 20, \alpha = 50, \beta = 2$, N=20, N1=1000				
σ_u	$E[\hat{\alpha}]$	$SD[\hat{\alpha}]$	$E[\hat{\beta}]$	$SD[\hat{\beta}]$
0.1	49.9575208	4.2533572	2.0054092	0.7925665
0.2	50.0542355	4.4443817	1.9972732	0.7851895
0.3	50.0738992	4.4365553	1.9717651	0.7640332
0.5	50.0546040	4.5639699	1.9833693	0.8018115
0.7	50.0005470	4.4606495	1.9706471	0.7834251

Appendix B: Tables for Logistic Regression

Table B.1				
$\alpha = -3.5, \beta = 0.05$, N=100 and N1=1000				
σ_u	$E[\hat{\alpha}]$	$SD[\hat{\alpha}]$	$E[\hat{\beta}]$	$SD[\hat{\beta}]$
0.10	-3.2161054	0.6956845	0.0536951	0.0115529
0.15	-3.2547677	0.7483852	0.0544228	0.0125121
0.20	-3.2751110	0.7456072	0.0546138	0.0123580
0.25	-3.2784620	0.7380119	0.0544429	0.0119445
0.30	-3.2225224	0.7124775	0.0536497	0.0116106

Table B.2				
$\alpha = -3.5, \beta = 0.05$, N=100 and N1=2000				
σ_u	$E[\hat{\alpha}]$	$SD[\hat{\alpha}]$	$E[\hat{\beta}]$	$SD[\hat{\beta}]$
0.10	-3.2509305	0.7204770	0.0540973	0.0118141
0.15	-3.2169980	0.7081743	0.0538145	0.0114513
0.20	-3.2646208	0.7242866	0.0545563	0.0118418
0.25	-3.2390438	0.7321297	0.0539766	0.0118669
0.30	-3.2317304	0.7045236	0.0540024	0.0114318

Table B.3				
$\alpha = -3.5, \beta = 0.05$, N=100 and N1=5000				
σ_u	$E[\hat{\alpha}]$	$SD[\hat{\alpha}]$	$E[\hat{\beta}]$	$SD[\hat{\beta}]$
0.10	-3.2258412	0.7234108	0.0539446	0.0118702
0.15	-3.2317478	0.7305132	0.0539156	0.0118576
0.20	-3.2325650	0.7052436	0.0540029	0.0115394
0.25	-3.2172087	0.7248877	0.0538975	0.0119682
0.30	-3.2290343	0.7137271	0.0539039	0.0117994

Table B.4				
$\alpha = -3.5, \beta = 0.05$, N=250 and N1=1000				
σ_u	$E[\hat{\alpha}]$	$SD[\hat{\alpha}]$	$E[\hat{\beta}]$	$SD[\hat{\beta}]$
0.10	-3.2025737	0.5978319	0.0534613	0.0084868
0.15	-3.2165633	0.5956177	0.0534902	0.0084055
0.20	-3.2242839	0.5703349	0.0537021	0.0081923
0.25	-3.2166228	0.5836107	0.0534794	0.0082667
0.30	-3.2080868	0.5745295	0.0534932	0.0082192

Table B.5				
$\alpha = -3.5, \beta = 0.05$, N=250 and N1=2000				
σ_u	$E[\hat{\alpha}]$	$SD[\hat{\alpha}]$	$E[\hat{\beta}]$	$SD[\hat{\beta}]$
0.10	-3.1975542	0.5762091	0.0532885	0.0082521
0.15	-3.2336066	0.5951746	0.0538218	0.0086623
0.20	-3.2189772	0.5759791	0.0535340	0.0083252
0.25	-3.2250801	0.6018651	0.0538481	0.0086991
0.30	-3.2118393	0.5710256	0.0536916	0.0083346

Table B.6				
$\alpha = -3.5, \beta = 0.05$, N=250 and N1=5000				
σ_u	$E[\hat{\alpha}]$	$SD[\hat{\alpha}]$	$E[\hat{\beta}]$	$SD[\hat{\beta}]$
0.10	-3.2188177	0.5918984	0.0535958	0.0084495
0.15	-3.1998554	0.5930697	0.0533744	0.0084044
0.20	-3.2160638	0.5815679	0.0536290	0.0084085
0.25	-3.2052228	0.5989366	0.0534411	0.0085442
0.30	-3.2102761	0.5821838	0.0534326	0.0083428

Appendix C: R Program for Simple Linear Regression

```
N1<-2000
N<-20
output<-matrix(NA,nrow=N1,ncol=2)
for (k in 1:N1) {
xmatrix<-matrix(NA,nrow=N,ncol=1)
a<-matrix(NA,nrow=N,ncol=1)
c<-matrix(NA,nrow=N,ncol=1)
ymatrix<-matrix(NA,nrow=N,ncol=1)
alpha<-50
beta<-2
for (i in 1:N) {
xmatrix[i]<- c[i]<-rnorm(1,0,Y.ERROR.SD) }
for (i in 1:N) {
ymatrix[i]<-alpha+beta*(xmatrix[i]-mean(xmatrix[]))+c[i]
a[i]<-rnorm(1,0,X.ERROR.SD)
xmatrix[i]<-i+a[i] }
x<-as.vector(xmatrix-mean(xmatrix[]))
y<-as.vector(ymatrix)
 gadata<-list(y=y,x=x,N=N)
parameters<-c("alphahat","betahat")
inits01<-
function(){list(alphahat=ALPHA.INITIAL.VALUE,betahat=BETA.INITIAL.VALUE,ta
u=2)}
model.file.errordata<-"BUGS.CODE.DIRECTORY
errordata.sim<-
bugs(data=gadata,inits=inits01,parameters,n.iter=20000,n.thin=10,n.burnin=10000,model
.file=model.file.errordata,debug=FALSE)
output[k,]<-errordata.sim$summary[c(1,2)]
}
```

```
A<-mean(output[,1])
B<-sd(output[,1])
C<-mean(output[,2])
D<-sd(output[,2])
print(c(A,B,C,D))
```

Appendix D: OpenBUGS Program for Simple Linear Regression

```
model {
for (i in 1:N) {
y[i] ~ dnorm(mu[i], tau)
mu[i] <- alphahat + betahat*x[i]
}
alphahat ~ dunif(-100000,100000)
betahat ~ dunif(-100000,100000)
tau ~ dgamma(1,1)
}
```

Appendix E: R Program for Logistic Regression

```
N1<-1000                                  #N1 is the number of iterations to be
run
N<-150                                    #N is the designated sample size
output<-matrix(NA,nrow=N1,ncol=2)         #Matrices were created for storing
                                          specified values
for (k in 1:N1) {
xmatrix<-matrix(NA,nrow=N,ncol=1)
a<-matrix(NA,nrow=N,ncol=1)
ybinary<-matrix(NA,nrow=N,ncol=1)
alpha<- α                                 #α is the numerical value used to
create the data
beta<- β                                  #β  is the numerical value used to
create the data
for (i in 1:N) {
xmatrix[i]<-i                             #Assigning X_i values to create Y_i
}
for (i in 1:N) {
pi<-((exp(alpha+beta*(xmatrix[i])))/(1+(exp(alpha+beta*(xmatrix[i]))))) #Calculating π_i
ybinary[i]<-rbern(1,pi)                   #Random variable from Bern(π_i)
a[i]<-rnorm(1,0,σ_u)                      #Generating X_i error randomly
                                          #σ_u is an assigned numerical value
xmatrix[i]<-i+a[i]                        #Infusing X_i with the error
}
x<-as.vector(xmatrix)                     #Converting from matrices to
vectors
y<-as.vector(ybinary)
gadata<-list(y=y,x=x,N=N)                 #Sending data and N value to
OpenBUGS
parameters<-c("alphahat","betahat")       #Naming the parameters
inits01<-function(){list(alphahat= IV_α,betahat=
IV_β)}                                    #IV is the initial value usually close
to α and β
model.file.errordata<-"Location of OpenBUGS code as txt file"

errordata.sim<-                           #Allows for OpenBUGS to be run
with the given
bugs(data=gadata,inits=inits01,parameters,n.iter=2000,n.thin=1,n.burnin=1000,model.file
=model.file.errordata,debug=FALSE)        #debug=TRUE would open
OpenBUGS each loop
output[k,]<-errordata.sim$summary[c(1,2)] #Collects and stores α̂ and β̂
}
A<-mean(output[,1])                       #Finds the average and standard
deviation
B<-sd(output[,1])                         #of α̂ and β̂ over all the iterations
C<-mean(output[,2])
```

```
D<-sd(output[,2])
print(c(A,B,C,D))                                    #Displays these values
```

Appendix F: OpenBUGS Program for Logistic Regression

OpenBugs Program

```
model {                                              #Inside the for loop is the likelihood function
for Y_i
for (i in 1:N){                                      #N is the sample size specified in
the R code
y[i] ~ dbern(pi[i])                                  #y[i] is the ith Y value R created
pi[i] <- ilogit(alphahat + betahat*x[i])             #x[i] is calculated directly in R
}
alphahat ~ dunif(-100,100)                           #These are the selected prior
distributions
betahat ~ dunif(-100,100)
}
```