# Chapter 17: Limited Dependent Variable Models

Susumu Shikano

Last compiled at 18. Juli 2022

## Binary response model

We generate 1000 datasets with n=200 under the GM-assumptions. The number of independent variables is 1. The true regression line has the intercept of 0.4 and the slope of 0.1. The independent variables are generated with the mean 1, variances 0.5.

Note that we use the logit function to compute the predicted values, which in turn are used for the probability that the dependent variable has the value 1. With the opposite probability, the dependent variable has the value 0. This is called Bernoulli trial.

```
samples.1 <- data.generation(sample.size=sample.size,
                             n.sim=num.datasets,
                             n.iv=n.iv,
                             x.mu=x.mu,
                             x.Sigma=x.Sigma,
                             para=c(true.intercept,true.slope),
                             err.dist = "normal",
                             err.disp = true.err.var,
                             binary.y = TRUE,
                             link.func = "logit")
```

Analogously, we generate further two sets of samples. The second set is generated under the same parameters except that the mean value of X is set to 5 instead of 1. The third set is generated under the same parameters of the first set except that the true slope is 0.5 instead of 0.1.

```
samples.2 <- data.generation(sample.size=sample.size,
                             n.sim=num.datasets,
                             n.iv=n.iv,
                             x.mu=x.mu+4,
                             x.Sigma=x.Sigma,
                             para=c(true.intercept,true.slope),
                             err.dist = "normal",
                             err.disp = true.err.var,
                             binary.y = TRUE,
                             link.func = "logit")

samples.3 <- data.generation(sample.size=sample.size,
                             n.sim=num.datasets,
                             n.iv=n.iv,
                             x.mu=x.mu,
                             x.Sigma=x.Sigma,
                             para=c(true.intercept,true.slope+0.4),
                             err.dist = "normal",
                             err.disp = true.err.var,
                             binary.y = TRUE,
                             link.func = "logit")
```

We can estimate the logit model

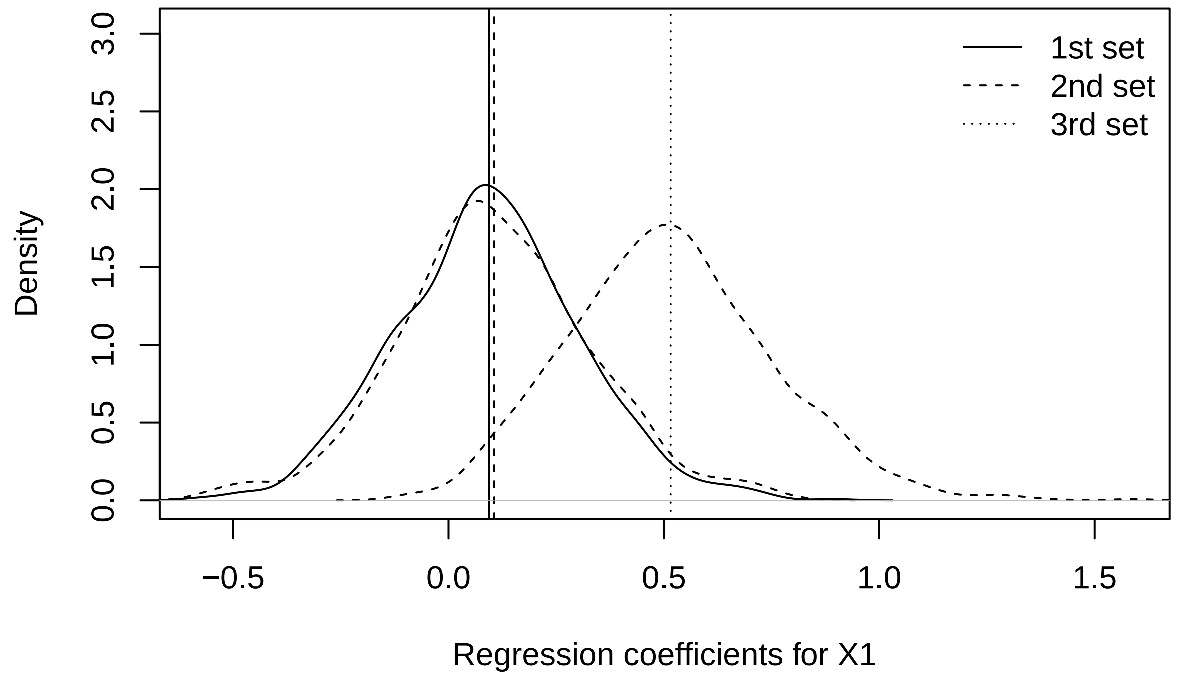
$$logit(\Pr(y = 1)) = \tilde{\beta}_0 + \tilde{\beta}_1 x_1$$

by using OLS…

```
all.coef <- array(NA,dim=c(num.datasets,3,3))
all.coef.se <- array(NA,dim=c(num.datasets,2,3))
all.predict.outside <- array(NA,dim=c(num.datasets,3,2))

for (i.set in 1:3 ){
  if (i.set ==1 ) this.generated.data <- samples.1; true.beta <- samples.1$para
  if (i.set ==2 ) this.generated.data <- samples.2; true.beta <- samples.1$para
  if (i.set ==3 ) this.generated.data <- samples.3; true.beta <- samples.1$para
for (i in 1:num.datasets){
    this.data <- this.generated.data$generated.data[[i]]

    #this.data <- this.data[,1:2]
    #ols.out <- logit.ols(beta=true.beta,data=this.data)

    this.x <- as.matrix(cbind(1,this.data[,2]))
    this.y <- c(this.data[,1])

    loss.func <- function(beta){
        y.hat <- c(this.x  %*%  beta)
        prob.y <- exp(y.hat)/(1+ exp(y.hat))
        SSR <- sum((this.y - prob.y)^2)
        SSR
    }

    nlm.out <- nlm(loss.func,true.beta)
    all.coef[i,1:2,i.set] <- nlm.out$estimate



    #all.coef[i,1:2,i.set] <- ols.out
    #all.coef[i,3,i.set] <- summary(lm.out)$sigma
    #all.coef.se[i,c(1:2),i.set] <- coef(summary(lm.out))[,2]

    #all.predict.outside[i,i.set,1] <- sum(this.data$y.hat==0|this.data$y.hat==1)
    #all.predict.outside[i,i.set,2] <- sum(lm.out$fitted.values<0|lm.out$fitted.values>1)

}
}
dimnames(all.coef)[[2]] <- c("b0","b1","sigma")
dimnames(all.coef.se)[[2]] <- c("b0","b1")
```

Below you will find the distribution of estimated coefficients:

```
    x.range <- range(c(all.coef[,"b1",]))
    #x.range[2] <- 0.52
    density.out <- density(all.coef[,"b1",1])
    plot(density.out,
         main="",xlab=paste0("Regression coefficients for X1"),
         xlim=x.range,ylim=c(0,max(density.out$y)*1.5))
    abline(v=mean(all.coef[,"b1",1]))
    par(new=T)
    plot(density(all.coef[,"b1",2]),ann=F,xlab="",ylab="",main="",
         axes=F,
         xlim=x.range,lty=2,
         ylim=c(0,max(density.out$y)*1.5))
    abline(v=mean(all.coef[,"b1",2]),lty=2)
    par(new=T)
    plot(density(all.coef[,"b1",3]),ann=F,xlab="",ylab="",main="",
         axes=F,
         xlim=x.range,lty=2,
         ylim=c(0,max(density.out$y)*1.5))
    abline(v=mean(all.coef[,"b1",3]),lty=3)
    legend("topright",lty=c(1:3),c("1st set","2nd set","3rd set"),bty="n")
```



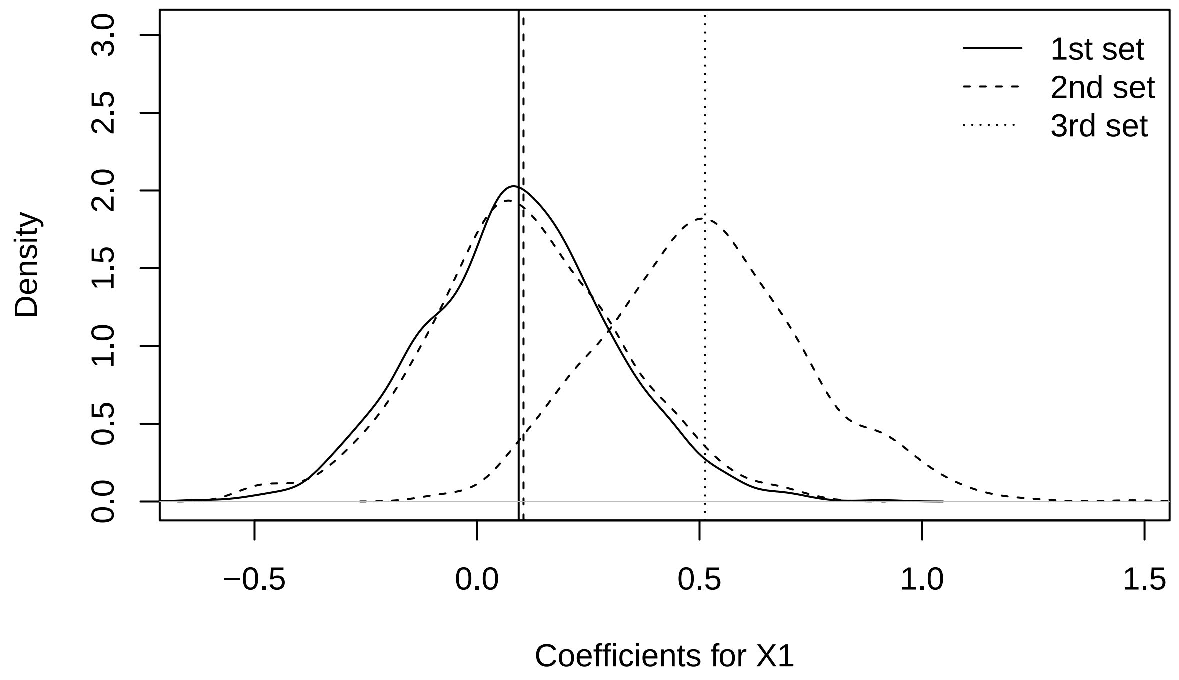
For the first and second set, the true regression slope is 0.1 while it is set to 0.5 for the third set. Obviously, all the estimates are unbiased.

Now, we estimate the same binary logit models by using maximum likelihood estimator.

```
logit.all.coef <- array(NA,dim=c(num.datasets,3,3))
logit.all.coef.se <- array(NA,dim=c(num.datasets,2,3))
#all.predict.outside <- array(NA,dim=c(num.datasets,3,2))

for (i.set in 1:3 ){
  if (i.set ==1 ) this.generated.data <- samples.1
  if (i.set ==2 ) this.generated.data <- samples.2
  if (i.set ==3 ) this.generated.data <- samples.3
for (i in 1:num.datasets){
    this.data <- this.generated.data$generated.data[[i]]

    glm.out <- glm(y ~ X1  ,family = binomial(link="logit"), data=  this.data)

    logit.all.coef[i,1:2,i.set] <- coef(glm.out)
    logit.all.coef[i,3,i.set] <- as.numeric(logLik(glm.out))
    logit.all.coef.se[i,c(1:2),i.set] <- coef(summary(glm.out))[,2]


}
}
dimnames(logit.all.coef)[[2]] <- c("b0","b1","LogLike")
dimnames(logit.all.coef.se)[[2]] <- c("b0","b1")
```

Below you will find the distribution of estimated coefficients:

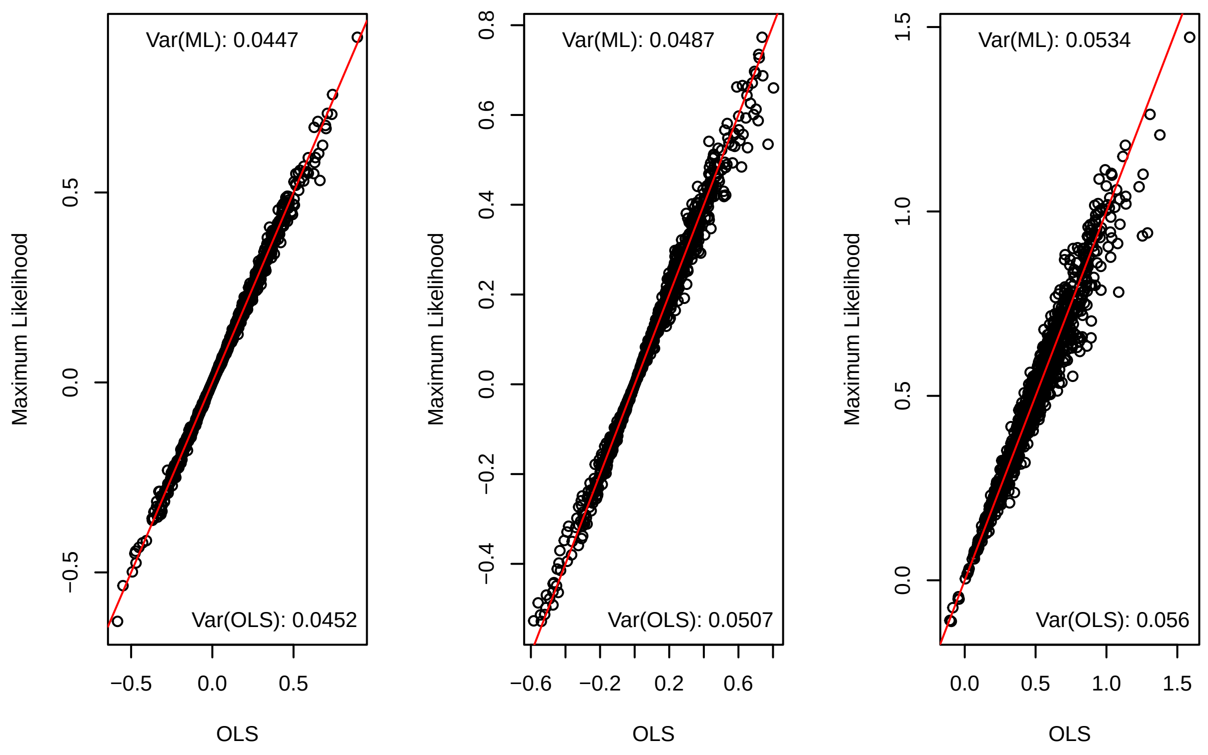
```
    x.range <- range(c(logit.all.coef[,"b1",]))
    #x.range[2] <- 0.52
    density.out <- density(logit.all.coef[,"b1",1])
    plot(density.out,
         main="",xlab=paste0("Coefficients for X1"),
         xlim=x.range,ylim=c(0,max(density.out$y)*1.5))
    abline(v=mean(logit.all.coef[,"b1",1]))
    par(new=T)
    plot(density(logit.all.coef[,"b1",2]),ann=F,xlab="",ylab="",main="",
         axes=F,
         xlim=x.range,lty=2,
         ylim=c(0,max(density.out$y)*1.5))
    abline(v=mean(logit.all.coef[,"b1",2]),lty=2)
    par(new=T)
    plot(density(logit.all.coef[,"b1",3]),ann=F,xlab="",ylab="",main="",
         axes=F,
         xlim=x.range,lty=2,
         ylim=c(0,max(density.out$y)*1.5))
    abline(v=mean(logit.all.coef[,"b1",3]),lty=3)
    legend("topright",lty=c(1:3),c("1st set","2nd set","3rd set"),bty="n")
```



The results look almost identical with the OLS estimates.

Below we can further compare the individual estimates with each other:

```
par(mfrow=c(1,3))
for (i.fig in 1:3){
  plot(all.coef[,"b1",i.fig] , logit.all.coef[,"b1",i.fig],
       xlab="OLS",ylab="Maximum Likelihood")
  abline(coef=c(0,1),col="red")

  legend("topleft",paste("Var(ML):" , round(var(logit.all.coef[,"b1",i.fig]),4)),bty="n")
  legend("bottomright",paste("Var(OLS):" , round(var(all.coef[,"b1",i.fig]),4)),bty="n")
}
```



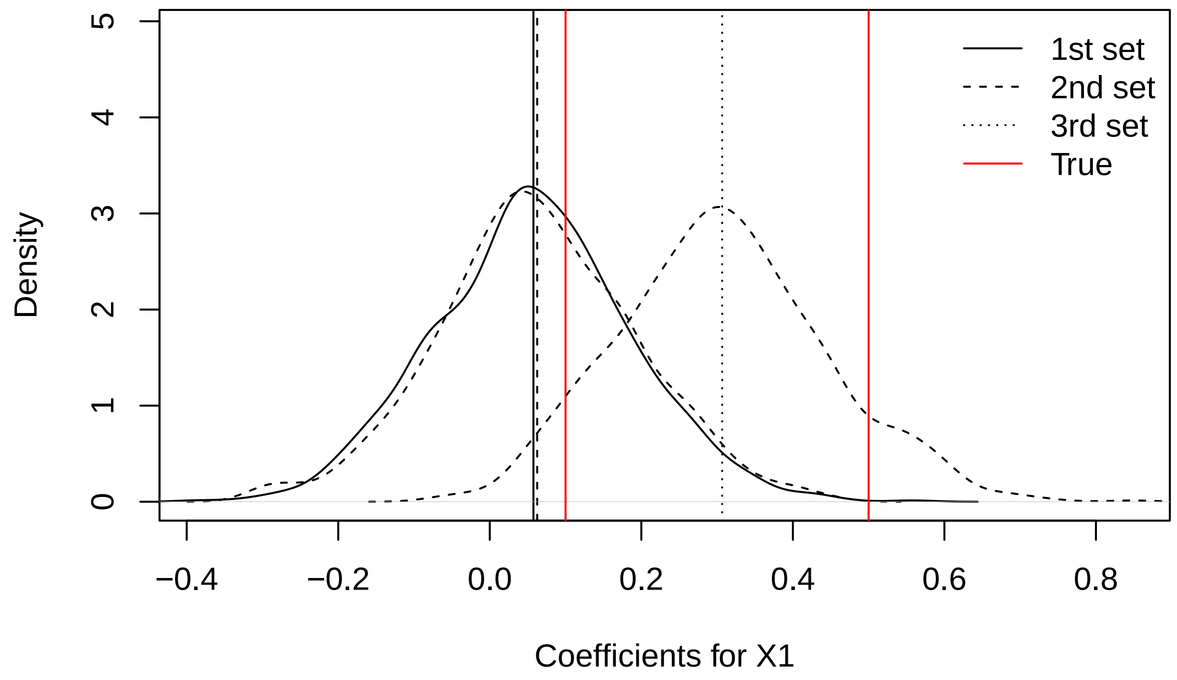
They are slight differences, but the variance of the point estimates is always smaller at maximum likelihood than at the OLS.

We can further estimate the probit model:

```
probit.all.coef <- array(NA,dim=c(num.datasets,3,3))
probit.all.coef.se <- array(NA,dim=c(num.datasets,2,3))
#all.predict.outside <- array(NA,dim=c(num.datasets,3,2))

for (i.set in 1:3 ){
  if (i.set ==1 ) this.generated.data <- samples.1
  if (i.set ==2 ) this.generated.data <- samples.2
  if (i.set ==3 ) this.generated.data <- samples.3
for (i in 1:num.datasets){
    this.data <- this.generated.data$generated.data[[i]]

    glm.out <- glm(y ~ X1  ,family = binomial(link="probit"), data=  this.data)

    probit.all.coef[i,1:2,i.set] <- coef(glm.out)
    probit.all.coef[i,3,i.set] <- as.numeric(logLik(glm.out))
    probit.all.coef.se[i,c(1:2),i.set] <- coef(summary(glm.out))[,2]



}
}
dimnames(probit.all.coef)[[2]] <- c("b0","b1","LogLike")
dimnames(probit.all.coef.se)[[2]] <- c("b0","b1")
```

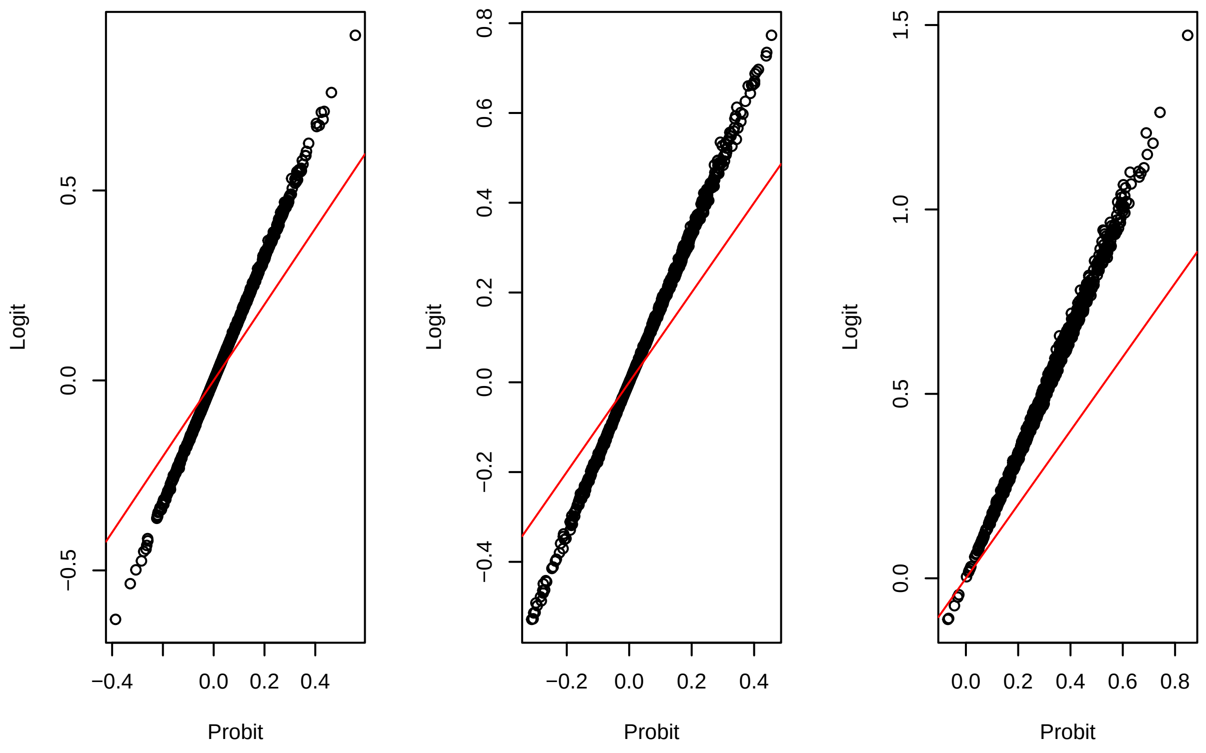
Below you will find the distribution of estimated coefficients:

```
    x.range <- range(c(probit.all.coef[,"b1",]))
    #x.range[2] <- 0.52
    density.out <- density(probit.all.coef[,"b1",1])
    plot(density.out,
         main="",xlab=paste0("Coefficients for X1"),
         xlim=x.range,ylim=c(0,max(density.out$y)*1.5))
    abline(v=mean(probit.all.coef[,"b1",1]))
    par(new=T)
    plot(density(probit.all.coef[,"b1",2]),ann=F,xlab="",ylab="",main="",
         axes=F,
         xlim=x.range,lty=2,
         ylim=c(0,max(density.out$y)*1.5))
    abline(v=mean(probit.all.coef[,"b1",2]),lty=2)
    par(new=T)
    plot(density(probit.all.coef[,"b1",3]),ann=F,xlab="",ylab="",main="",
         axes=F,
         xlim=x.range,lty=2,
         ylim=c(0,max(density.out$y)*1.5))
    abline(v=mean(probit.all.coef[,"b1",3]),lty=3)
    abline(v=c(0.1,0.5),col="red")
    legend("topright",lty=c(1:3,1),
           col=c(rep("black",3),"red"),
           c("1st set","2nd set","3rd set","True"),bty="n")
```



The probit models miss to find the true parameter values. This figure however has to be interpreted with caution. The figure below displays the comparison of individual estimates based on the logit and probit model:

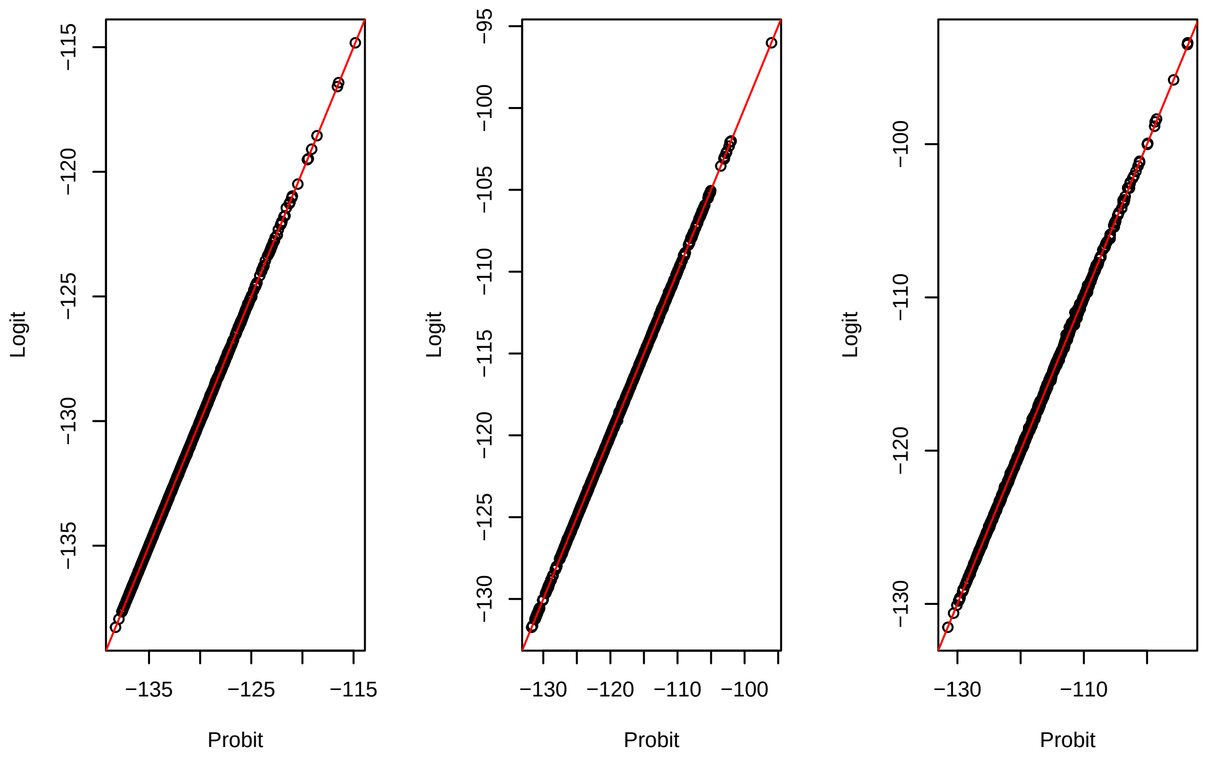
```
par(mfrow=c(1,3))
for (i.fig in 1:3){
  plot(probit.all.coef[,"b1",i.fig] , logit.all.coef[,"b1",i.fig],
       xlab="Probit",ylab="Logit")
  abline(coef=c(0,1),col="red")

}
```



The estimates deviate significantly from the 45-degree lines (the red line). This is however only due to the different scaling in both models.

If one compares the model performance in the log likelihood values, both models performs in the same way:

```
par(mfrow=c(1,3))
for (i.fig in 1:3){
  plot(probit.all.coef[,"LogLike",i.fig] , logit.all.coef[,"LogLike",i.fig],
       xlab="Probit",ylab="Logit")
  abline(coef=c(0,1),col="red")

}
```


Logit
−115
−120
−125
−130
−135
−135
−125
−115
Probit
Logit
−95
−100
−105
−110
−115
−120
−125
−130
−130
−120
−110
−100
Probit
Logit
−100
−110
−120
−130
−130
−110
Probit