ps2

Boyu Chen

2023-04-24

Question 1

```
source("funs.R")
```

In `funs.R`, I write some functions to help my analysis, I will show them in the appendix.

read data

```
flight.df <- readxl::read_xlsx("CilibertoTamerEconometrica.xlsx")
dat2 <- (flight.df %>%
            mutate(N = rowSums(across(airlineAA:airlineWN)), const = 1, .before = marketdistance))

firm.Names <- flight.df %>% select(starts_with("airline")) %>% colnames() %>% substring(8)

market.regressors <- c('const',"marketdistance","marketsize" )
firm.regressors <- c("marketpresence","mindistancefromhub")
market.regressors.i <- c(9, 10, 15)
firm.regressors.i <- c(18, 24)

n.mkt <- nrow(dat2)
firm.n <- length(firm.Names)

length.of.each.param <- c(length(firm.regressors),length(market.regressors),1,1 )
get_XX_and_Z.mat_and_Y()
```

After reading the data, I aggregated the presence of each firms, and called it `total.N`. Then we select the second to eighth columns of the data, called it `true.n` And I write a function `get_XX_and_Z.mat_and_Y()`, it is nothing but just for saving space and generate `XX`, `Z_mat` and `Y`. Where `XX` is an `n.mkt`x3 matrix, `Z_mat` is a `n.mkt`x12 matrix, `Y` is a `n.mkt`x1 vector.

$$XX = \begin{bmatrix} 1 & \text{marketdistance} & \text{marketsize} \end{bmatrix}$$

$$Z = \begin{bmatrix} Z1 & Z2 & Z3 & Z4 & Z5 & Z6 \end{bmatrix}$$

$$Y = (total.N_1, total.N_2, \ldots, total.N_{n.mkt})$$

In XX, we have 2742 rows, each row recorded the "constant term", "market distance" and "market size" for market i , i =1,. . . , 2742 In Z, we have 2742 rows, for each row we have [z1, . . . , z6] and z1 has 2 elements, one is "market presence" and the other is "min distance from hub" for firm 1. Similarly, z2,. . . ,z6 have also 2 elements and stand for the same meaning. In Y, we have 2742 rows, and each row recorded the total number of the firms entered the market.

ordered probit

When we conduct ordered probit, we only consider the market characteristics and the number of entered firms ,so the profit eqaution is

$$\pi_i(N) = X_i\beta + \delta ln(N) + u_{i0}$$

Where X_i is the market characteristics and N is the potential number of entered firms in the market. Let N_i^* denote the actual number of entered firms, we can inference the profitability of each firm by calculate the conditional probability given N_i. For the case $N_i = 0$, we can infer that no any potential firm can earn money in the market i. Thus, the probability of $N_i^* = 0$ is

$$P(N_i^* = 0) = P(\pi_i(1) \leq 0) = P(X_i\beta - \delta ln(1) + u_{i0} \leq 0) = \Phi(-X_i\beta)$$

For the case $N_i = 6$, we can infer that all firms can earn money in the market i. Thus, the probability of $N_i^* = 6$ is

$$P(N_i^* = 6) = P(\pi_i(6) \geq 0) = 1 - P(\pi_i(6) \leq 0) = 1 - \Phi(-X_i\beta + \delta ln(6))$$

For the rest of cases, $N_i = 1, 2, ..., 5$, the probability is

$$P(N_i^* = N_i) = \Phi(-X_i\beta + \delta ln(N_i + 1)) - \Phi(-X_i\beta + \delta ln(N_i))$$

And the log-likelihood function is

$$\mathcal{L} = \sum_{i=1}^{2742} \sum_{k=0}^{6} \mathbf{1}_i(k) ln(P(N_i^* = k))$$

In like_oprobit, we calculate the log-likelihood for the above problem.

```
like_oprobit<- function(init){
    XX <- XX %>% as.matrix()
    beta.mat <- init[1:3] %>% as.matrix(, ncol=1)
    delta <- init[4]
    f <- 0
    for (i in 1:n.mkt){
        if (Y[i] == 0){
            p = pnorm(-XX[i,]%*% beta.mat)
        }
        else if (Y[i] == 6){
            p = 1 - pnorm(-XX[i,] %*% beta.mat + delta*log(Y[i]))
        }
        else{
            p = pnorm(-XX[i,] %*% beta.mat + delta*log(Y[i]+1)) -
                pnorm(-XX[i,] %*% beta.mat + delta*log(Y[i]))
        }
    }
    f <- f -log(p)
    return(f)
}
```

```r
set.seed(1234)
init <- c(1,1,1,1)
fit <- optim(fn = like_oprobit, par = init, method = "BFGS")
```

```r
par_df <- data.frame(value = fit[["par"]])
rownames(par_df) <- c("b0", "b1", "b2", "d")
kable(par_df)
```

	value
b0	1.648658
b1	1.587684
b2	1.561012
d	12.974703

MSM

We consider the firm specific characteristic (Z_{ik}). The profit equation become

$$\pi_i(N) = X_i\beta + Z_{ik}\alpha + \delta ln(N) + \sigma u_{ik} + \rho u_{i0}$$

Where $\sigma = \sqrt{1-\rho^2}$, Thus, we have

$$\pi_i(N) = X_i\beta + Z_{ik}\alpha + \delta ln(N) + \sqrt{1-\rho^2}u_{ik} + \rho u_{i0}$$

For firm k, the simulated profit is

$$\hat{\pi}_{ik}(N, \hat{u}_i) = X_i\beta + Z_{ik}\alpha + \delta ln(N) + \sqrt{1-\rho^2}\hat{u}_{ik} + \rho\hat{u}_{i0}$$

That is, we generate $\hat{u}_{ik}$ and $\hat{u}_{i0}$ from

$$u_{i0}, u_{i1}, ..., u_{ik} \overset{i.i.d.}{\sim} N(0,1)$$

and use the simulated error term to calculate the simulated profit $\hat{\pi}_{ik}(N, \hat{u}_i)$.

An unbiased pf the expected number of firms is

$$\hat{n}(W_i, \theta, \hat{u}_i) = max_{0 \leq n \leq K_i}[n : \#k : \hat{\pi}_{ik}(N, \hat{u}_i) \geq 0 \geq n]$$

Averaging across T draws gives

$$\hat{N}(W_i, \theta, \{\hat{u}_i^t\}) = \frac{1}{T}\sum_{t=1}^{T}\hat{n}(W_i, \theta, \hat{u}_i^t)$$

The prediction error for market i is

$$v_{i0}(N_i^*, W_i, \theta) = N_i^* - \hat{N}(W_i, \theta, \{\hat{u}_i^t\})$$

The moment condition for MSM is

$$E[v_{i0}(N_i^*, W_i, \theta)|W_i, \theta = \theta^*] = 0$$

The sample analog of moment condition is

$$g_n = \frac{1}{n}\sum_{i=1}^{2742}\hat{N}_i X_i$$

$$g_n = \frac{1}{n}\sum_{i=1}^{2742}\left[\hat{N}_i X_i \quad \hat{N}_{i1}[X_i Z_{i1}] \quad \hat{N}_{i2}[X_i Z_{i2}] \quad \hat{N}_{i3}[X_i Z_{i3}] \quad \hat{N}_{i4}[X_i Z_{i4}] \quad \hat{N}_{i5}[X_i Z_{i5}] \quad \hat{N}_{i6}[X_i Z_{i6}]\right]^T$$

Where $\hat{N}_{ik}, k = 1, ..., 6$ are scalar, X_i is a row vector with 3 elements, Z_{ik} is row vector with 2 elements. Thus, the dimension of $g_n $ is 33×1 vector.

$$\hat{\theta}_{MM} = argmin_{\beta} g_n(\theta)' g_n(\theta)$$

We set parallel to save simulation time:

```
n.cores <- parallel::detectCores() - 1
my.cluster <- parallel::makeCluster(
    n.cores,
    type = "PSOCK"
)
doParallel::registerDoParallel(cl = my.cluster)
foreach::getDoParRegistered()
```

```
## [1] TRUE
```

```
foreach::getDoParWorkers()
```

```
## [1] 7
```

```
clusterEvalQ(cl=my.cluster, source("funs.R"))
```

```
## [[1]]
## [[1]]$value
## function (para)
## {
##     A <- para[1:2] %>% as.matrix()
##     B <- para[3:5] %>% as.matrix()
##     d <- para[6]
##     rho <- para[7]
##     sim.result <- foreach(mkt.i = 1:2742, .combine = "rbind") %dopar%
##         {
##             T.sim.process(mkt.i, A, B, d, rho)
##         }
##     v <- pred.error(sim.result)
##     g_n <- g_n.fn(v)
##     mom <- t(g_n) %*% g_n
##     return(mom)
## }
##
## [[1]]$visible
## [1] FALSE
##
##
## [[2]]
```

```
## [[2]]$value
## function (para)
## {
##     A <- para[1:2] %>% as.matrix()
##     B <- para[3:5] %>% as.matrix()
##     d <- para[6]
##     rho <- para[7]
##     sim.result <- foreach(mkt.i = 1:2742, .combine = "rbind") %dopar%
##         {
##             T.sim.process(mkt.i, A, B, d, rho)
##         }
##     v <- pred.error(sim.result)
##     g_n <- g_n.fn(v)
##     mom <- t(g_n) %*% g_n
##     return(mom)
## }
##
## [[2]]$visible
## [1] FALSE
##
##
## [[3]]
## [[3]]$value
## function (para)
## {
##     A <- para[1:2] %>% as.matrix()
##     B <- para[3:5] %>% as.matrix()
##     d <- para[6]
##     rho <- para[7]
##     sim.result <- foreach(mkt.i = 1:2742, .combine = "rbind") %dopar%
##         {
##             T.sim.process(mkt.i, A, B, d, rho)
##         }
##     v <- pred.error(sim.result)
##     g_n <- g_n.fn(v)
##     mom <- t(g_n) %*% g_n
##     return(mom)
## }
##
## [[3]]$visible
## [1] FALSE
##
##
## [[4]]
## [[4]]$value
## function (para)
## {
##     A <- para[1:2] %>% as.matrix()
##     B <- para[3:5] %>% as.matrix()
##     d <- para[6]
##     rho <- para[7]
##     sim.result <- foreach(mkt.i = 1:2742, .combine = "rbind") %dopar%
##         {
##             T.sim.process(mkt.i, A, B, d, rho)
```

```
##         }
##     v <- pred.error(sim.result)
##     g_n <- g_n.fn(v)
##     mom <- t(g_n) %*% g_n
##     return(mom)
## }
##
## [[4]]$visible
## [1] FALSE
##
##
## [[5]]
## [[5]]$value
## function (para)
## {
##     A <- para[1:2] %>% as.matrix()
##     B <- para[3:5] %>% as.matrix()
##     d <- para[6]
##     rho <- para[7]
##     sim.result <- foreach(mkt.i = 1:2742, .combine = "rbind") %dopar%
##         {
##             T.sim.process(mkt.i, A, B, d, rho)
##         }
##     v <- pred.error(sim.result)
##     g_n <- g_n.fn(v)
##     mom <- t(g_n) %*% g_n
##     return(mom)
## }
##
## [[5]]$visible
## [1] FALSE
##
##
## [[6]]
## [[6]]$value
## function (para)
## {
##     A <- para[1:2] %>% as.matrix()
##     B <- para[3:5] %>% as.matrix()
##     d <- para[6]
##     rho <- para[7]
##     sim.result <- foreach(mkt.i = 1:2742, .combine = "rbind") %dopar%
##         {
##             T.sim.process(mkt.i, A, B, d, rho)
##         }
##     v <- pred.error(sim.result)
##     g_n <- g_n.fn(v)
##     mom <- t(g_n) %*% g_n
##     return(mom)
## }
##
## [[6]]$visible
## [1] FALSE
##
```

```
##
## [[7]]
## [[7]]$value
## function (para)
## {
##     A <- para[1:2] %>% as.matrix()
##     B <- para[3:5] %>% as.matrix()
##     d <- para[6]
##     rho <- para[7]
##     sim.result <- foreach(mkt.i = 1:2742, .combine = "rbind") %dopar%
##         {
##             T.sim.process(mkt.i, A, B, d, rho)
##         }
##     v <- pred.error(sim.result)
##     g_n <- g_n.fn(v)
##     mom <- t(g_n) %*% g_n
##     return(mom)
## }
##
## [[7]]$visible
## [1] FALSE
```

```
clusterEvalQ(cl=my.cluster, library(magrittr))
```

```
## [[1]]
## [1] "magrittr"  "stats"     "graphics"  "grDevices" "utils"     "datasets"
## [7] "methods"   "base"
##
## [[2]]
## [1] "magrittr"  "stats"     "graphics"  "grDevices" "utils"     "datasets"
## [7] "methods"   "base"
##
## [[3]]
## [1] "magrittr"  "stats"     "graphics"  "grDevices" "utils"     "datasets"
## [7] "methods"   "base"
##
## [[4]]
## [1] "magrittr"  "stats"     "graphics"  "grDevices" "utils"     "datasets"
## [7] "methods"   "base"
##
## [[5]]
## [1] "magrittr"  "stats"     "graphics"  "grDevices" "utils"     "datasets"
## [7] "methods"   "base"
##
## [[6]]
## [1] "magrittr"  "stats"     "graphics"  "grDevices" "utils"     "datasets"
## [7] "methods"   "base"
##
## [[7]]
## [1] "magrittr"  "stats"     "graphics"  "grDevices" "utils"     "datasets"
## [7] "methods"   "base"
```

```r
clusterExport(cl = my.cluster, c("n.mkt","Z1", "Z2", "Z3","Z4", "Z5", "Z6", "Z.mat", "XX", "true.n"))
```

```r
T <- 1000
set.seed(2048)
init.param = c(0.5, 0.5, 0.9, 0.1, 0.5, 1.9, 0.6)
msm.fit <- optim(fn = obj.fn, par = init.param, method = "BFGS")
stopImplicitCluster()
```

```r
par_df <- data.frame(value = msm.fit[["par"]])
rownames(par_df) <- c("a1", "a2", "b0", "b1", "b2", "d", "rho")
kable(par_df)
```

	value
a1	26.560479
a2	-13.382931
b0	54.436733
b1	-7.087295
b2	-19.915174
d	9.925087
rho	9.302349

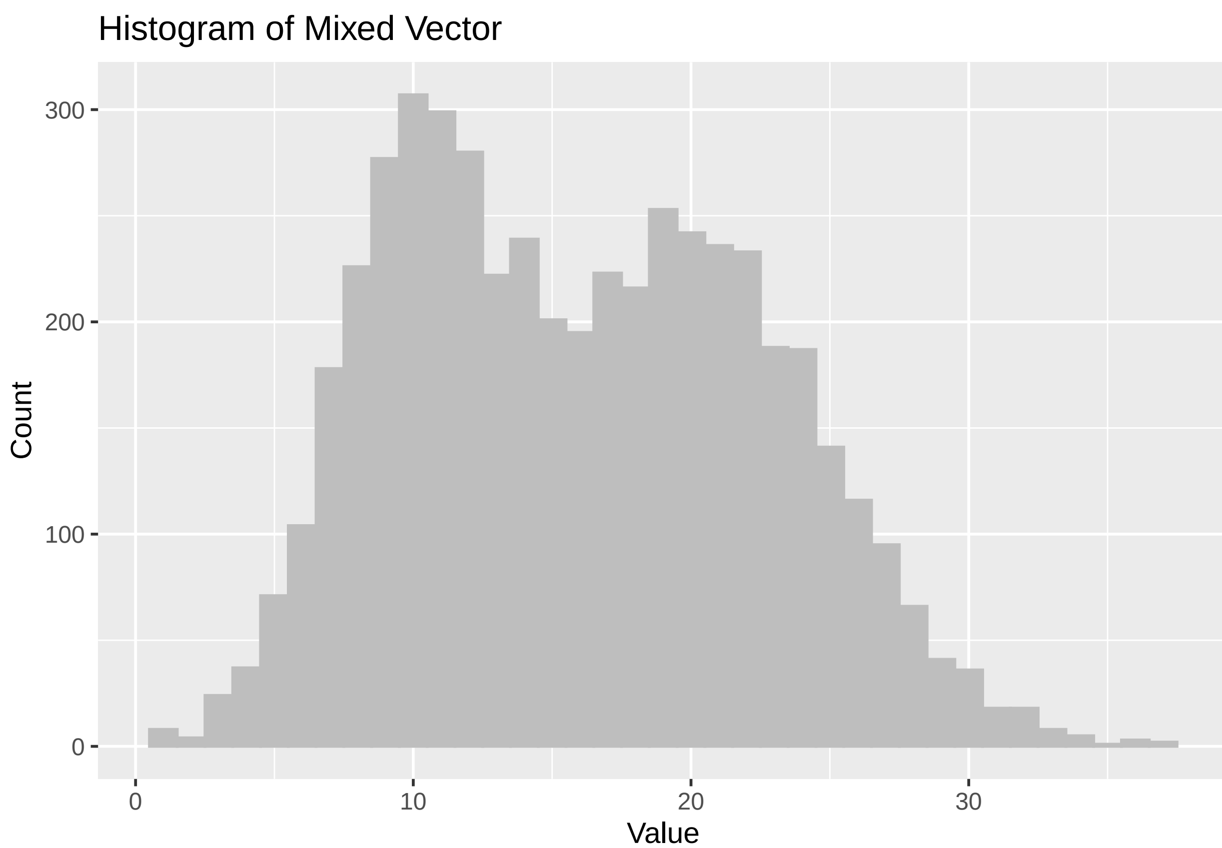
Question 2

DGP

```r
# DGP
set.seed(123)
x <- c(rnorm(3000, 20, 5), rnorm(2000, 10, 3))
df <- data.frame(x = x)
```

Plot histogram

```r
ggplot(df, aes(x = x)) +
    geom_histogram(binwidth = 1, color = "gray", fill = "gray") +
    labs(title = "Histogram of Mixed Vector",
         x = "Value",
         y = "Count")
```



init param

```
mu <- c(8, 15)
sigma <- c(5, 5)
pi <- c(0.5, 0.5)
```

functions

```
# E-step
estep <- function(x, mu, sigma, pi) {
    n <- length(x)
    k <- length(mu)
    post <- matrix(0, n, k)
    for (i in 1:n) {
        for (j in 1:k) {
            post[i, j] <- dnorm(x[i], mu[j], sigma[j]) * pi[j]
        }
        post[i, ] <- post[i, ] / sum(post[i, ])
    }
    return(post)
}
```

```r
# M-step
mstep <- function(x, post) {
    n <- nrow(post)
    k <- ncol(post)
    mu <- numeric(k)
    sigma <- numeric(k)
    pi <- numeric(k)
    for (j in 1:k) {
        mu[j] <- sum(post[, j] * x) / sum(post[, j])
        sigma[j] <- sqrt(sum(post[, j] * (x - mu[j])^2) / sum(post[, j]))
        pi[j] <- sum(post[, j]) / n
    }
    return(list(mu = mu, sigma = sigma, pi = pi))
}

# EM
em <- function(x, mu, sigma, pi, tol = 1e-6, maxiter = 100) {
    loglik <- numeric(maxiter)
    for (iter in 1:maxiter) {
        # E-step
        post <- estep(x, mu, sigma, pi)
        # M-step
        params <- mstep(x, post)
        # update parameters
        mu <- params$mu
        sigma <- params$sigma
        pi <- params$pi
        # calculate log-likelihood
        loglik[iter] <- sum(post[,1]* log(pi[1]) + dnorm(x, mu[1], sigma[1], log=TRUE)+
                                post[,2]* log(pi[2]) + dnorm(x, mu[2], sigma[2], log=TRUE))
        # check convergence
        if (iter > 1 && abs(loglik[iter] - loglik[iter - 1]) < tol) {
            break
        }
    }
    return(list(mu = mu, sigma = sigma, pi = pi, loglik = loglik[1:iter]))
}
```

EM Result Table

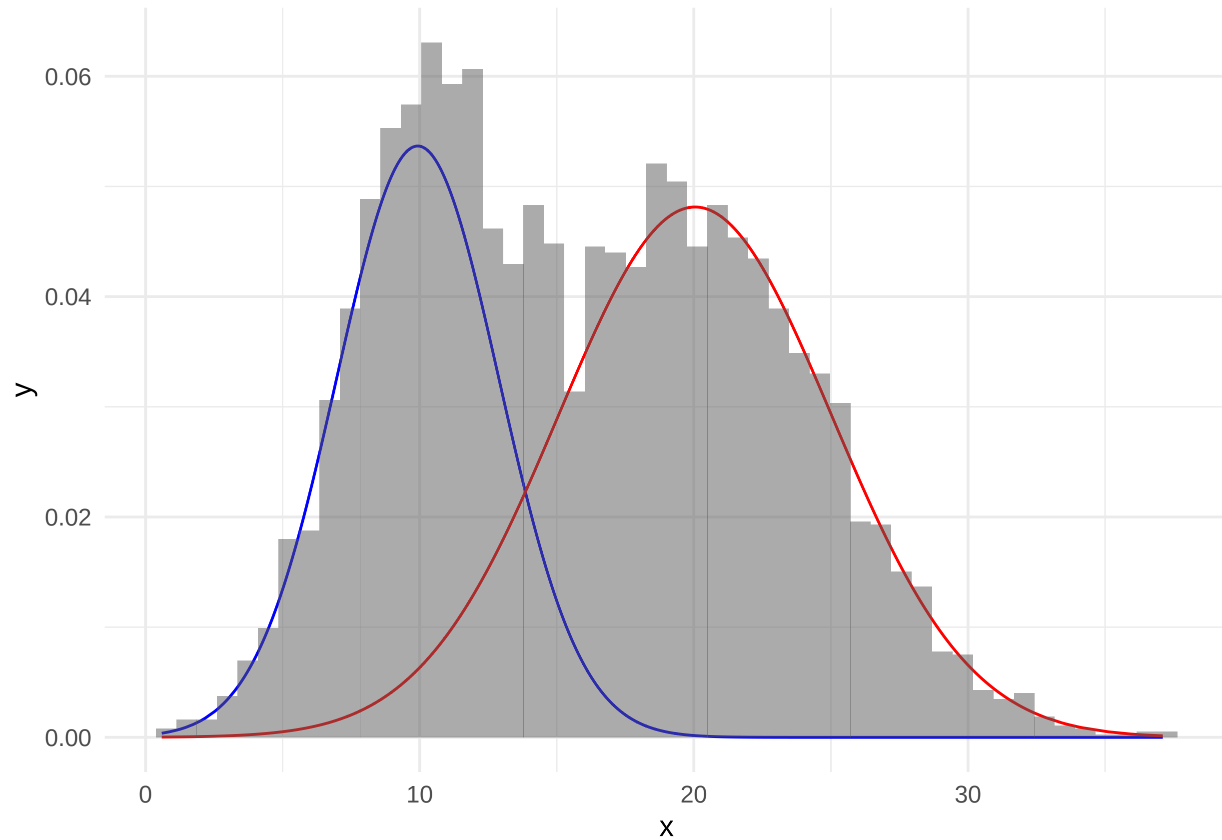
```r
result <- em(x, mu, sigma, pi)
res_df <- data.frame(mu = result$mu, sigma = result$sigma, pi = result$pi)
rownames(res_df) <- c('x1', 'x2')
kable(res_df)
```

	mu	sigma	pi
x1	9.927575	2.960945	0.3983197
x2	20.042762	4.986873	0.6016803

EM Result Histogram

```
df$y <- result$pi[1]*dnorm(df$x, mean = result$mu[1], sd = result$sigma[1])
df$z <- result$pi[2]*dnorm(df$x, mean = result$mu[2], sd = result$sigma[2])

ggplot(df, aes(x = x)) +
    geom_line(aes(y = y), color = "blue") +
    geom_line(aes(y = z), color = "red") +
    geom_histogram(aes(y = after_stat(density)), bins = 50, alpha = 0.5) +
    theme_minimal()
```



Appendix: Functions for Question 1

```
get_XX_and_Z.mat_and_Y <- function(...){
    XX <<- dat2[, market.regressors]

    Z1 <<- dat2[, firm.regressors.i]
    Z2 <<- dat2[, firm.regressors.i +1]
    Z3 <<- dat2[, firm.regressors.i +2]
    Z4 <<- dat2[, firm.regressors.i +3]
    Z5 <<- dat2[, firm.regressors.i +4]
    Z6 <<- dat2[, firm.regressors.i +5]
```

```
    Z.mat <<- cbind(Z1, Z2, Z3, Z4, Z5, Z6)

    Y <<- dat2$N

    true.n <<- dat2 %>% select(airlineAA:N)
}

draw_u <- function(...){
    u_ik <<- matrix(rnorm(n.mkt * 6), ncol=6)
    u_i0 <<- rnorm(n.mkt)
}



like_oprobit<- function(init){
    beta.mat <- init[1:3]
    delta <- init[4]
    f <- 0
    for (i in 1:n.mkt){
        if (Y[i] == 0){
            p = pnorm(-t(XX[i,])%*% beta.mat)
        }
        else if (Y[i] == 6){
            p = 1 - pnorm(-t(XX[i,]) %*% beta.mat + delta*log(Y[i]))
        }
        else{
            p = pnorm(-t(XX[i,]) %*% beta.mat + delta*log(Y[i]+1)) - pnorm(-t(XX[i,]) %*% beta.mat + de
        }
    }
    f <- f -log(p)
}

pred.error <- function(n_hat){
    v <- true.n - n_hat
    return(v)
}

g_n.fn <- function(v){
    v <- as.matrix(v)
    E_v0X  = v[,7] * XX
    E_v1Z  = v[,1] * cbind(XX, Z1)
    E_v2Z  = v[,2] * cbind(XX, Z2)
    E_v3Z  = v[,3] * cbind(XX, Z3)
    E_v4Z  = v[,4] * cbind(XX, Z4)
    E_v5Z  = v[,5] * cbind(XX, Z5)
    E_v6Z  = v[,6] * cbind(XX, Z6)

    v_i_hat <- cbind(E_v0X, E_v1Z, E_v2Z, E_v3Z, E_v4Z, E_v5Z, E_v6Z)
    g_n <- colMeans(v_i_hat)
    return(g_n)
}

obj.fn <- function(para){
```

```
    A <- para[1:2] %>% as.matrix()
    B <- para[3:5] %>% as.matrix()
    d <- para[6]
    rho <- para[7]
    n_hat <- get.n_hat_parallel(A,B,d,rho, T, num_cores = 4)
    v <- pred.error(n_hat)
    g_n <- g_n.fn(v)
    mom <- t(g_n) %*% g_n
    return(mom)
}
```