Structural Estimation HW1

Sangwook Nam, Hojun Choi

April 2020

Short Answers

Problem 1

Part (1)

$\mathbb{E}N_r(\boldsymbol{x}, \boldsymbol{y})$ is the expected value of the firm with the minimum expected per-unit cost to supply, given a contract r and the location vector $(\boldsymbol{x}, \boldsymbol{y})$.

Part (2)

$$\begin{aligned} N_r(\boldsymbol{x}, \boldsymbol{y}) &:= \min_{f \in \mathcal{F}_{p(r)}} \{C_{rf}(\boldsymbol{x}, \boldsymbol{y})\} \\ &= \min_{f \in \mathcal{F}_{p(r)}} \left\{ \bar{C}_r + \mathbb{E}_\epsilon \left[\min_{i \in \mathcal{I}_f} \left(\bar{C}_{ri} + \epsilon_{ri} \right) \right] + \eta_{rf} \right\} \\ &= \bar{C}_r + \min_{f \in \mathcal{F}_{p(r)}} \left\{ \mathbb{E}_\epsilon \left[\min_{i \in \mathcal{I}_f} \left(\bar{C}_{ri} + \epsilon_{ri} \right) \right] + \eta_{rf} \right\} \\ &= \bar{C}_r + \min_{f \in \mathcal{F}_{p(r)}} \left\{ \bar{C}_{rf} + \eta_{rf} \right\} \\ &= \bar{C}_r + \min_{f \in \mathcal{F}_{p(r)}} \left\{ \bar{C}_{rf} + \eta_{rf} \right\}. \end{aligned}$$

Assuming that $-\eta_{rf}$ is an (independent) extreme value distribution type-I (i.e., it is a Gumbel distribution), we can turn the minimization into a maximization:

$$- \min_{f \in \mathcal{F}_{p(r)}} \left\{ \bar{C}_{rf} + \eta_{rf} \right\} = \max_{f \in \mathcal{F}_{p(r)}} \left\{ -\bar{C}_{rf} - \eta_{rf} \right\}.$$

Since the Gumbel distributions are preserved across maximization, the resulting term is another Gumbel distribution, denoted as η_{rf}^m, with the parameter $LSE\left(-\bar{C}_{rf}\right)$. Therefore,

$$\begin{aligned} \mathbb{E}N_r\left(\boldsymbol{x}, \boldsymbol{y}\right) &= \mathbb{E}\left[\bar{C}_r - \eta_{rf}^m\right] \\ &= \bar{C}_r - \mathbb{E}\eta_{rf}^m \\ &= \bar{C}_r - LSE\left(-\bar{C}_{rf}\right) - \gamma. \end{aligned}$$

Problem 2

The reason that the author uses a nested logit structure with two error terms, ϵ_{ri} and η_{rf}, is due to the fact that these random variables are realized in different stages. Given a contract r, an assembler

a chooses a firm f that bids the lowest cost, and the firm f then chooses the plant $i \in \mathcal{I}_f$ that will cost it the least. However, during the first stage, the assembler a selects the firm before ϵ_{ri} is realized.

Coding Assignment

The solution coordinates (plant locations) we have found are as follows:

```
> solution <- readRDS('solution-namchoi.rds'); colnames(solution) <- c('x','y')
> return(solution)

          x          y
[1,] 0.2008832 0.7991196
[2,] 0.4093427 0.06435284
[3,] 0.1668629 0.7963746
[4,] 0.8382203 0.8604695
[5,] 0.6793604 0.4455839
[6,] 0.2253864 0.4082442
```

The structure of the code is as follows:

The final output we would like is a vector of (x, y)-coordinates, which can be obtained via the `optim` function. Now, in the optim function, we are using the `Minimum` function as the input parameter function, which uses the `Exp_N` function, which in turns uses the closed form solution we derived from above. Next, the `Exp_N` function calculates the LSE using the output of the `C_bar_tri` function.

```
> library('plyr'); library("dplyr"); library("doParallel")
> nodes <- detectCores()
> cl <- makeCluster(nodes)
> registerDoParallel(cl)

> setwd("/home/snk5906/OPNS_Rosenbaum2013")
> assemblers <- readRDS('/home/snk5906/OPNS_Rosenbaum2013/variables/assembly.rds')
> competetors <- readRDS('/home/snk5906/OPNS_Rosenbaum2013/variables/competetor.rds')
> union.rate.fn <- readRDS('/home/snk5906/OPNS_Rosenbaum2013/variables/unionizationRate.rds')
> beta <- readRDS('/home/snk5906/OPNS_Rosenbaum2013/variables/costParameters.rds')

> find_best_location <- function(num.sites, assemblers, competetors, union.rate.fn,
+                                beta, num.tries){
+
+   # Obtain the location parameter
+   C_bar_tri <- function(assemblers, suppliers){
+     euc_dist <- sum(sqrt((assemblers[1] - suppliers[1])^2 + (assemblers[2] - suppliers[2])^2))
+     union_rate <- union.rate.fn(suppliers[1], suppliers[2])
+     c_bar_tri <- beta[[1]] * euc_dist + beta[[2]] * union_rate
+
+     return(c_bar_tri)
+   }
+
```

```
+   # Find LSE based on the closed form we derived from Part I
+   Exp_N <- function(assemblers, suppliers){
+     lse <- t(data.frame("lse" = 1:nrow(suppliers)))
+     for (i in 1:nrow(suppliers)){
+       lse[i] <- -1*C_bar_tri(assemblers, suppliers[i,])
+     }
+     LSE <- -1*log(sum(exp(unlist(lse))))
+
+     return(LSE)
+   }
+
+   Minimum <- function(suppliers){
+     df_sup_input <- matrix(suppliers, nc = 2)
+     df_sup_total <- rbind(df_sup_input, as.matrix(competetors))
+     min <- t(data.frame("min" = 1:nrow(assemblers)))
+     for (a in 1:nrow(assemblers)){
+       min[a] <- Exp_N(assemblers[a,], df_sup_total)
+     }
+     min_sum <- sum(min)
+
+     return(min_sum)
+   }
+
+   opt_sol <- seq(num.tries) %>%
+     llply(function(l){
+       optim(
+         runif(2 * num.sites),
+         Minimum,
+         control=list(maxit=10000)
+       )
+     },
+     .parallel = TRUE
+     )
+
+   '%!in%' <- function(x,y) !( '%in%'(x,y) )
+   coordinates_cost <- unlist(opt_sol)[names(unlist(opt_sol)) %!in%
+                                       c("counts.function", "counts.gradient", "convergence")] %>%
+                        matrix(ncol= 2*num.sites+1, byrow=T) %>% data.frame()
+   colnames(coordinates_cost) <- c(1:(2*num.sites), "cost")
+   opt_location <- coordinates_cost %>% filter(cost==min(cost)) %>% select(-cost) %>%
+                   matrix(ncol=2)
+
+   return(opt_location)
+ }

> find_best_location(
+   num.sites = 6,
+   assemblers = readRDS('/home/snk5906/OPNS_Rosenbaum2013/variables/assembly.rds'),
```

```
+   competetors = readRDS('/home/snk5906/OPNS_Rosenbaum2013/variables/competetor.rds'),
+   union.rate.fn = readRDS('/home/snk5906/OPNS_Rosenbaum2013/variables/unionizationRate.rds'),
+   beta = readRDS('/home/snk5906/OPNS_Rosenbaum2013/variables/costParameters.rds'),
+   num.tries = 12
+ ) %>%
+   saveRDS('/home/snk5906/OPNS_Rosenbaum2013/solution-namchoi.rds')
>
```