library(stats)

data = read.csv(file = "M:/A Master of Science in Marketing Sciences/Mathematical Models in Marketing (Kohli)/latent/data\_ipad.csv",head = TRUE)

attach(data)

summary(data)

## Modelling Step 1: Set up Training Set and Dev Set

random\_factor = sample(1:15,137,replace = TRUE)

determinant\_factor = (1:137)\*15 - 15

## The index for Dev Set is the summation of Random and Deterministic.

index = random\_factor + determinant\_factor

dev = data[which(choice\_set\_id %in% index) ,]

train = data[which(!(choice\_set\_id %in% index)) ,]

## Model 1: Fit an aggregate model without any segment

## In the training set,

X1\_train = subset(train, alternative\_id\_in\_set == 1)

X2\_train = subset(train, alternative\_id\_in\_set == 2)

X3\_train = subset(train, alternative\_id\_in\_set == 3)

## In the test set,``

X1\_test = subset(dev, alternative\_id\_in\_set == 1)

X2\_test = subset(dev, alternative\_id\_in\_set == 2)

X3\_test = subset(dev, alternative\_id\_in\_set == 3)

## The dependent variable

train\_choice = X1\_train$choice

test\_choice = X1\_test$choice

train\_set = cbind(X1\_train[,5:22],X2\_train[,5:22],X3\_train[,5:22])

test\_set = cbind(X1\_test[,5:22],X2\_test[,5:22],X3\_test[,5:22])

#

# ## Multi-Nomial Logit Estimation

# par = rnorm(18)

# N = 1918

#

# ll <- function(beta)

# {

# res = 0

#

# M1 = as.matrix(train\_set)[,1:18] %\*% beta

# M2 = as.matrix(train\_set)[,19:36] %\*% beta

# M3 = as.matrix(train\_set)[,37:54] %\*% beta

#

# ## Each row of M is the probability for each individual to choose each alternative

# M = cbind(exp(M1),exp(M2),exp(M3))

# M = M / rowSums(M)

# ## Construct the matrix cbind(seq(1,length(train\_choice)),train\_choice) to select the choice for each individual

# MP = M[cbind(seq(1,length(train\_choice)),train\_choice)]

# ## Maximize Likelihood ==> Maximize log-likelihood

# ## ==> Minimize minus log-likelihood

# res = res + sum(-log(MP))

#

# return (res)

# }

#

# ML = nlm(ll,par,hessian = TRUE)

# ## ML$estimate is the estimated value for each of the 18 parameters

# mode = ML$estimate

# SE = sqrt(diag(solve(ML$hessian)))

# Tvalue = mode/SE

# ll = 2\*ML$minimum

# Result = cbind(Estimate = mode, SE= SE, Tvalue = Tvalue, minusll = ll)

# round(Result,2)

# ## The predicted relative utility for each of the three alternatives

# X1\_predict = as.matrix(X1\_test)[,5:22] %\*% ML$estimate

# X2\_predict = as.matrix(X2\_test)[,5:22] %\*% ML$estimate

# X3\_predict = as.matrix(X3\_test)[,5:22] %\*% ML$estimate

# ## The prediction is the maximum utility among the three alternatives.

# prediction = max.col(cbind(X1\_predict,X2\_predict,X3\_predict))

#

# ## The performance of the model without segments is evaluated by the proportion of correct predictions.

# performance\_0 = sum(prediction == test\_choice)/length(test\_choice)

## The second stage: Latent Class Logit Model with Two Segments

## The parameters for the first segment

par1 = rnorm(18)

## The parameters for the second segment

par2 = rnorm(18)

## The parameters for the third segment

par3 = rnorm(18)

## The parameter for the likelihood that a randomly chosen individual belongs to segment 1

p3 = rnorm(3)

p3 = p3/sum(p3)

## The combined parameters are

par\_3 = t(cbind(t(par1),t(par2),t(par3),t(p3[1:2])))

N = 1918

ll\_3 <- function(beta)

{

res = 0

## To calculate the likelohood for the first segment

M11 = as.matrix(train\_set)[,1:18] %\*% beta[1:18]

M12 = as.matrix(train\_set)[,19:36] %\*% beta[1:18]

M13 = as.matrix(train\_set)[,37:54] %\*% beta[1:18]

M1 = cbind(exp(M11),exp(M12),exp(M13))

## The likelihood for each of the three alternatives in the first segment

M1 = M1 / rowSums(M1)

M21 = as.matrix(train\_set)[,1:18] %\*% beta[19:36]

M22 = as.matrix(train\_set)[,19:36] %\*% beta[19:36]

M23 = as.matrix(train\_set)[,37:54] %\*% beta[19:36]

M2 = cbind(exp(M21),exp(M22),exp(M23))

## The likelihood for each of the three alternatives in the first segment

M2 = M2 / rowSums(M2)

M31 = as.matrix(train\_set)[,1:18] %\*% beta[37:54]

M32 = as.matrix(train\_set)[,19:36] %\*% beta[37:54]

M33 = as.matrix(train\_set)[,37:54] %\*% beta[37:54]

M3 = cbind(exp(M31),exp(M32),exp(M33))

## The likelihood for each of the three alternatives in the first segment

M3 = M3 / rowSums(M3)

## Given that the consumer belongs to Segment 1, what is the probability for him to choose Alternative i?

prob\_1 = M1[cbind(seq(1,length(train\_choice)),train\_choice)]

## Given that the consumer belongs to Segment 2, what is the probability for him to choose Alternative i?

prob\_2 = M2[cbind(seq(1,length(train\_choice)),train\_choice)]

## Given that the consumer belongs to Segment 2, what is the probability for him to choose Alternative i?

prob\_3 = M3[cbind(seq(1,length(train\_choice)),train\_choice)]

## This transformation guarantees that the likelihood for each segment is within the range of (0,1)

probability\_1 = 1/(1 + exp(beta[55]))

probability\_2 = 1/(1 + exp(beta[56]))

for (i in 1:137)

{

MP = c()

MP[i] = probability\_1 \* cumprod(prob\_1[(14\*(i-1)+1):(14\*i)])[14] +

probability\_2 \* cumprod(prob\_2[(14\*(i-1)+1):(14\*i)])[14] +

(1 - probability\_2 - probability\_1) \* cumprod(prob\_3[(14\*(i-1)+1):(14\*i)])[14]

res = res - log(MP[i])

}

return (res)

}

ML\_3 = optim(par\_3,ll\_3,method = "Nelder-Mead", hessian = TRUE)

ML\_3 = nlm(ll\_3,par\_3,hessian = TRUE)

mode = ML\_3$estimate

SE = sqrt(diag(solve(ML\_3$hessian)))

Tvalue = mode/SE

ll = 2\*ML\_3$minimum

Result\_3 = cbind(Estimate = mode, SE= SE, Tvalue = Tvalue, minusll = ll)

round(Result\_3,2)

## This is the estimates for pi\_i(the market share of the first segment)

pi\_1 = 1/(1 + exp(ML\_3$estimate[55]))

pi\_2 = 1/(1 + exp(ML\_3$estimate[56]))

# ## NExt, I am going to obtain the individual-level estimate of segment membership using Bayes rule.

# ## USing the first individual as an example

# M11 = as.matrix(train\_set)[1:14,1:18] %\*% ML\_2$estimate[1:18]

# M12 = as.matrix(train\_set)[1:14,19:36] %\*% ML\_2$estimate[1:18]

# M13 = as.matrix(train\_set)[1:14,37:54] %\*% ML\_2$estimate[1:18]

# M1 = cbind(exp(M11),exp(M12),exp(M13))

# ## The likelihood for Indivual 1 to choose each of the three alternatives in the first segment

# M1 = M1 / rowSums(M1)

# M11 = M1[cbind(1:14,train\_choice[1:14])]

#

# M21 = as.matrix(train\_set)[1:14,1:18] %\*% ML\_2$estimate[19:36]

# M22 = as.matrix(train\_set)[1:14,19:36] %\*% ML\_2$estimate[19:36]

# M23 = as.matrix(train\_set)[1:14,37:54] %\*% ML\_2$estimate[19:36]

# M2 = cbind(exp(M21),exp(M22),exp(M23))

# ## The likelihood for each of the three alternatives in the first segment

# M2 = M2 / rowSums(M2)

# M12 = M2[cbind(1:14,train\_choice[1:14])]

# cumprod(M11)[14]/(cumprod(M11)[14]+cumprod(M12)[14])

## Concise Expression

## The likelihood for each of the three alternatives in the first segment

M11 = as.matrix(train\_set)[,1:18] %\*% ML\_2$estimate[1:18]

M12 = as.matrix(train\_set)[,19:36] %\*% ML\_2$estimate[1:18]

M13 = as.matrix(train\_set)[,37:54] %\*% ML\_2$estimate[1:18]

M1 = cbind(exp(M11),exp(M12),exp(M13))

M1 = M1 / rowSums(M1)

## M11 is the probability for individual i's actual choice in Segment 1

M11 = M1[cbind(1:(137\*14),train\_choice[1:(137\*14)])]

## The likelihood for each of the three alternatives in the second segment

M21 = as.matrix(train\_set)[,1:18] %\*% ML\_2$estimate[19:36]

M22 = as.matrix(train\_set)[,19:36] %\*% ML\_2$estimate[19:36]

M23 = as.matrix(train\_set)[,37:54] %\*% ML\_2$estimate[19:36]

M2 = cbind(exp(M21),exp(M22),exp(M23))

M2 = M2 / rowSums(M2)

## M11 is the probability for individual i's actual choice in Segment 2

M12 = M2[cbind(1:(14\*137),train\_choice[1:(14\*137)])]

## The likelihood for each of the three alternatives in the third segment

M31 = as.matrix(train\_set)[,1:18] %\*% ML\_2$estimate[37:54]

M32 = as.matrix(train\_set)[,19:36] %\*% ML\_2$estimate[37:54]

M33 = as.matrix(train\_set)[,37:54] %\*% ML\_2$estimate[37:54]

M3 = cbind(exp(M31),exp(M32),exp(M33))

M3 = M3 / rowSums(M3)

## M11 is the probability for individual i's actual choice in Segment 2

M13 = M3[cbind(1:(14\*137),train\_choice[1:(14\*137)])]

## The individual-level estimates of segment membership using Bayes Rule

## What is the orobability for each individual to belong to Segment 1?

prob <- rep(0,137\*2)

prob = matrix(prob, 137,2)

for (i in 1:137)

{

prob[i,1] = (pi\_1 \* (cumprod(M11[(1+(i-1)\*14):(14\*i)])[14]))/( (pi\_1 \* cumprod(M11[(1+(i-1)\*14):(14+(i-1)\*14)])[14]) + (pi\_2 \* cumprod(M12[(1+(i-1)\*14):(14+(i-1)\*14)])[14]) + ((1 - pi\_2 - pi\_1) \* cumprod(M13[(1+(i-1)\*14):(14+(i-1)\*14)])[14]))

prob[i,2] = (pi\_2 \* (cumprod(M12[(1+(i-1)\*14):(14+(i-1)\*14)])[14])) / ( (pi\_1 \* cumprod(M11[(1+(i-1)\*14):(14+(i-1)\*14)])[14]) + (pi\_2 \* cumprod(M12[(1+(i-1)\*14):(14+(i-1)\*14)])[14]) + ((1 - pi\_2 - pi\_1) \* cumprod(M13[(1+(i-1)\*14):(14+(i-1)\*14)])[14]) )

}

## The vector prob is the probability for each individual to choose Segment 1

##########################################

# ## Another Method which is theoretical correct but practically infeasible

# cumprod(M11[(1+(2-1)\*14):(14+(2-1)\*14)])[14]

# cum\_M11 = cumprod(M11)

# cum\_M12 = cumprod(M12)

# cumprod(M11)[14]/(cumprod(M11)[14]+cumprod(M12)[14])

#

# a <- 1:(137\*14)

# b\_11 <- cum\_M11[seq(14, 137\*14, 14)]

# b\_11\_tmp <- t(cbind(1,t(b\_11)))

# b\_11\_tem = b\_11\_tmp[1:137]

# b\_11/b\_11\_tem

##########################################

## Then I am going to do the Cross-Validation

## The expected probability for each alternative in Segment 1

X1\_predict\_11 = as.matrix(X1\_test)[,5:22] %\*% ML\_3$estimate[1:18]

X2\_predict\_12 = as.matrix(X2\_test)[,5:22] %\*% ML\_3$estimate[1:18]

X3\_predict\_13 = as.matrix(X3\_test)[,5:22] %\*% ML\_3$estimate[1:18]

## The expected probability for each alternative in Segment 2

X1\_predict\_21 = as.matrix(X1\_test)[,5:22] %\*% ML\_3$estimate[19:36]

X2\_predict\_22 = as.matrix(X2\_test)[,5:22] %\*% ML\_3$estimate[19:36]

X3\_predict\_23 = as.matrix(X3\_test)[,5:22] %\*% ML\_3$estimate[19:36]

## The expected probability for each alternative in Segment 3

X1\_predict\_31 = as.matrix(X1\_test)[,5:22] %\*% ML\_3$estimate[37:54]

X2\_predict\_32 = as.matrix(X2\_test)[,5:22] %\*% ML\_3$estimate[37:54]

X3\_predict\_33 = as.matrix(X3\_test)[,5:22] %\*% ML\_3$estimate[37:54]

predict\_31 = cbind(exp(X1\_predict\_11),exp(X2\_predict\_12),exp(X3\_predict\_13))

predict\_32 = cbind(exp(X1\_predict\_21),exp(X2\_predict\_22),exp(X3\_predict\_23))

predict\_33 = cbind(exp(X1\_predict\_31),exp(X2\_predict\_32),exp(X3\_predict\_33))

## predict\_21 is the predicted probability for each individual to choose each alternative given that he belongs to Segment 1

predict\_31 = predict\_31/rowSums(predict\_31)

## predict\_22 is the predicted probability for each individual to choose each alternative given that he belongs to Segment 2

predict\_32 = predict\_32/rowSums(predict\_32)

predict\_33 = predict\_33/rowSums(predict\_33)

## The vector prob is the probability for each individual to choose Segment 1

## We can use the posterior segment membership probability to estimate the probability that individual i chooses alternative j

predict\_3 = prob[,1] \* predict\_31 + prob[,2] \* predict\_32 + (1 - prob[,1] - prob[,2]) ^ predict\_33

prediction\_3 = max.col(predict\_3)

## The performance of the model without segments is evaluated by the proportion of correct predictions.

performance\_3 = sum(prediction\_3 == test\_choice)/length(test\_choice)

performance\_3

> performance\_3

[1] 0.620438

> Result\_3

Estimate SE Tvalue minusll

[1,] 0.32587140 0.17773883 1.8334283 3109.399

[2,] 0.52620505 0.18918713 2.7813998 3109.399

[3,] 0.34076463 0.18761291 1.8163176 3109.399

[4,] 0.20337790 0.16815450 1.2094704 3109.399

[5,] 0.17138776 0.14060082 1.2189670 3109.399

[6,] 0.43046539 0.14532321 2.9621242 3109.399

[7,] 0.39228968 0.14339090 2.7358059 3109.399

[8,] 0.32800896 0.13970927 2.3477966 3109.399

[9,] 0.68257616 0.13630430 5.0077375 3109.399

[10,] 0.93319994 0.14777186 6.3151397 3109.399

[11,] 0.43196849 0.12304695 3.5105990 3109.399

[12,] 0.72919713 0.11723362 6.2200343 3109.399

[13,] 0.09201766 0.10900005 0.8441983 3109.399

[14,] -0.02351569 0.11265675 -0.2087375 3109.399

[15,] -0.07364772 0.15425867 -0.4774300 3109.399

[16,] -0.07682612 0.16103646 -0.4770728 3109.399

[17,] -0.40204701 0.18337199 -2.1925214 3109.399

[18,] -0.90561359 0.19845335 -4.5633574 3109.399

[19,] 0.14679619 0.40031515 0.3667016 3109.399

[20,] 4.40890804 0.75166416 5.8655292 3109.399

[21,] 1.01858228 0.31313720 3.2528306 3109.399

[22,] 0.17728619 0.32941676 0.5381821 3109.399

[23,] 0.09254470 0.29379788 0.3149944 3109.399

[24,] 0.47207784 0.26373528 1.7899685 3109.399

[25,] 0.55680808 0.27667050 2.0125314 3109.399

[26,] -0.39786771 0.30460593 -1.3061719 3109.399

[27,] 0.41499415 0.25426268 1.6321473 3109.399

[28,] 0.21936161 0.26529493 0.8268594 3109.399

[29,] -0.25251701 0.25340477 -0.9964967 3109.399

[30,] 0.59834320 0.22536223 2.6550287 3109.399

[31,] 0.32079267 0.26622236 1.2049802 3109.399

[32,] 0.46635064 0.27095337 1.7211472 3109.399

[33,] -0.56229654 0.37754792 -1.4893382 3109.399

[34,] -0.51619263 0.32951623 -1.5665166 3109.399

[35,] -0.55296569 0.34118632 -1.6207147 3109.399

[36,] -0.67905455 0.33013269 -2.0569140 3109.399

[37,] 0.70048602 0.27248924 2.5706924 3109.399

[38,] 0.41848745 0.25319077 1.6528543 3109.399

[39,] 0.77189567 0.23776709 3.2464362 3109.399

[40,] 0.16940794 0.24237936 0.6989372 3109.399

[41,] 0.44010639 0.22347195 1.9694033 3109.399

[42,] 0.59095676 0.22431268 2.6345223 3109.399

[43,] 0.72765667 0.24112391 3.0177707 3109.399

[44,] 0.72806586 0.22417995 3.2476850 3109.399

[45,] 0.73517441 0.21438022 3.4293016 3109.399

[46,] 0.75105990 0.26156333 2.8714266 3109.399

[47,] 0.63397636 0.19634341 3.2289159 3109.399

[48,] 1.04877005 0.19785088 5.3008107 3109.399

[49,] 0.09460309 0.16309676 0.5800427 3109.399

[50,] 0.12578888 0.15690510 0.8016876 3109.399

[51,] -0.84453434 0.17428044 -4.8458356 3109.399

[52,] -2.45794707 0.32277714 -7.6149974 3109.399

[53,] -4.34391481 0.46942033 -9.2537850 3109.399

[54,] -7.60820504 1.09601084 -6.9417243 3109.399

[55,] 0.42504956 0.06062412 7.0112282 3109.399

[56,] 0.19972619 0.04677975 4.2695009 3109.399

> performance\_3

[1] 0.6569343

> Result\_3

Estimate SE Tvalue minusll

[1,] 0.55224149 0.23890705 2.31153287 3115.552

[2,] 0.30704852 0.22053723 1.39227522 3115.552

[3,] 0.69792900 0.20895752 3.34005215 3115.552

[4,] 0.18889765 0.20885273 0.90445379 3115.552

[5,] 0.50810729 0.19498111 2.60593082 3115.552

[6,] 0.54186354 0.19146421 2.83010358 3115.552

[7,] 0.72183631 0.21356818 3.37988702 3115.552

[8,] 0.59236533 0.18753684 3.15866107 3115.552

[9,] 0.64965489 0.18396133 3.53147521 3115.552

[10,] 0.44227511 0.21321067 2.07435733 3115.552

[11,] 0.67327698 0.16725358 4.02548615 3115.552

[12,] 1.15277979 0.16945083 6.80303405 3115.552

[13,] 0.19750511 0.15137724 1.30472134 3115.552

[14,] 0.09301104 0.14224538 0.65387738 3115.552

[15,] -0.90351494 0.16548733 -5.45972260 3115.552

[16,] -2.40950144 0.24055749 -10.01632268 3115.552

[17,] -4.02933473 0.31653377 -12.72955719 3115.552

[18,] -6.28742459 0.51291357 -12.25825351 3115.552

[19,] 0.17988961 0.31696646 0.56753516 3115.552

[20,] 4.60470840 0.64740848 7.11252404 3115.552

[21,] 1.03415927 0.33040350 3.12998884 3115.552

[22,] 0.34185298 0.31232194 1.09455320 3115.552

[23,] -0.01010085 0.32111057 -0.03145598 3115.552

[24,] 0.40758999 0.33818847 1.20521552 3115.552

[25,] 0.52522156 0.28778798 1.82502952 3115.552

[26,] -0.46830324 0.27538107 -1.70056438 3115.552

[27,] 0.40509568 0.27951244 1.44929395 3115.552

[28,] 0.09698007 0.32096882 0.30214794 3115.552

[29,] -0.12665064 0.22524004 -0.56229186 3115.552

[30,] 0.50613502 0.22176852 2.28226719 3115.552

[31,] 0.06417136 0.24436672 0.26260271 3115.552

[32,] 0.25329384 0.22402265 1.13066174 3115.552

[33,] -0.36557958 0.40847020 -0.89499694 3115.552

[34,] -0.33677864 0.33350598 -1.00981289 3115.552

[35,] -0.24490500 0.36638398 -0.66843806 3115.552

[36,] -0.41520112 0.33165021 -1.25192480 3115.552

[37,] 0.37339288 0.16214781 2.30279324 3115.552

[38,] 0.61978389 0.16102401 3.84901536 3115.552

[39,] 0.31012166 0.15726272 1.97199731 3115.552

[40,] 0.10987398 0.15781573 0.69621693 3115.552

[41,] 0.10356610 0.13519865 0.76602914 3115.552

[42,] 0.51001588 0.13540347 3.76663806 3115.552

[43,] 0.36487887 0.13349068 2.73336583 3115.552

[44,] 0.36138864 0.14209264 2.54333117 3115.552

[45,] 0.80654522 0.13626074 5.91913131 3115.552

[46,] 1.08538390 0.14167401 7.66113645 3115.552

[47,] 0.34283485 0.10791678 3.17684487 3115.552

[48,] 0.62840062 0.10603018 5.92662032 3115.552

[49,] 0.08980847 0.10339603 0.86858714 3115.552

[50,] 0.06059221 0.10424987 0.58122099 3115.552

[51,] 0.08379102 0.14898405 0.56241606 3115.552

[52,] 0.12364985 0.15114772 0.81807287 3115.552

[53,] -0.18663408 0.16242169 -1.14907117 3115.552

[54,] -0.73265746 0.17192799 -4.26142037 3115.552

[55,] 0.40233248 0.04374187 9.19787954 3115.552

[56,] 0.18033616 0.03716032 4.85292250 3115.552

> performance\_3

[1] 0.5839416

> Result\_3

Estimate SE Tvalue minusll

[1,] -0.53805819 0.64737583 -0.8311373 3185.405

[2,] -1.90569058 0.76594345 -2.4880304 3185.405

[3,] -2.86844081 0.89541614 -3.2034723 3185.405

[4,] -0.12311361 0.75456914 -0.1631575 3185.405

[5,] 0.37401494 0.60102948 0.6222905 3185.405

[6,] -1.04801779 0.71833909 -1.4589458 3185.405

[7,] -3.25488450 1.04832696 -3.1048372 3185.405

[8,] 0.17902548 0.72529420 0.2468315 3185.405

[9,] 1.34705492 0.69813463 1.9295060 3185.405

[10,] 3.14668402 0.82344338 3.8213727 3185.405

[11,] -0.43307384 0.57169737 -0.7575229 3185.405

[12,] -0.48784229 0.52471942 -0.9297203 3185.405

[13,] 2.66318513 0.82058434 3.2454740 3185.405

[14,] 2.17744684 0.77860307 2.7966071 3185.405

[15,] 0.70524181 0.76573425 0.9210007 3185.405

[16,] 0.73697646 0.82295051 0.8955295 3185.405

[17,] 1.67012741 0.91153482 1.8322146 3185.405

[18,] -1.06064922 0.79724237 -1.3303974 3185.405

[19,] 0.23916023 0.18759140 1.2748997 3185.405

[20,] 1.88896344 0.17016432 11.1008199 3185.405

[21,] 0.73002876 0.15359240 4.7530266 3185.405

[22,] 0.12592569 0.16570131 0.7599559 3185.405

[23,] 0.07777474 0.13605570 0.5716390 3185.405

[24,] 0.39265096 0.13620736 2.8827441 3185.405

[25,] 0.40778494 0.14170707 2.8776611 3185.405

[26,] 0.07261998 0.13761576 0.5277011 3185.405

[27,] 0.68491282 0.13693462 5.0017506 3185.405

[28,] 0.61877927 0.13914714 4.4469421 3185.405

[29,] 0.25991771 0.11037997 2.3547544 3185.405

[30,] 0.63986416 0.10571551 6.0526989 3185.405

[31,] -0.03173765 0.10989229 -0.2888069 3185.405

[32,] 0.13467165 0.10459109 1.2876016 3185.405

[33,] -0.15579127 0.15761235 -0.9884459 3185.405

[34,] -0.05945895 0.15792492 -0.3765014 3185.405

[35,] -0.08306937 0.16736143 -0.4963472 3185.405

[36,] -0.44169813 0.17075100 -2.5867967 3185.405

[37,] 0.72604683 0.19787646 3.6691925 3185.405

[38,] 0.52904143 0.18597822 2.8446419 3185.405

[39,] 0.49060848 0.16542558 2.9657353 3185.405

[40,] 0.32987761 0.16386822 2.0130664 3185.405

[41,] 0.34699219 0.14611122 2.3748497 3185.405

[42,] 0.66530876 0.14940385 4.4530899 3185.405

[43,] 0.57086087 0.16127888 3.5395885 3185.405

[44,] 0.63386133 0.14715725 4.3073742 3185.405

[45,] 0.76297840 0.15325584 4.9784623 3185.405

[46,] 0.88935716 0.17135894 5.1900248 3185.405

[47,] 0.54343412 0.12490564 4.3507572 3185.405

[48,] 0.92142851 0.12403879 7.4285514 3185.405

[49,] 0.12172847 0.11554787 1.0534895 3185.405

[50,] 0.07867673 0.11483386 0.6851353 3185.405

[51,] -0.44453486 0.13327433 -3.3354874 3185.405

[52,] -1.58951370 0.16510257 -9.6274314 3185.405

[53,] -2.84788528 0.22834792 -12.4716937 3185.405

[54,] -4.52395813 0.39857594 -11.3503042 3185.405

[55,] 0.03642886 0.01608463 2.2648239 3185.405

[56,] 0.43660664 0.04756842 9.1784981 3185.405

> performance\_3

[1] 0.4963504

> Result\_3

Estimate SE Tvalue minusll

[1,] 1.479419e-01 0.64737583 2.285255e-01 1799.925

[2,] 7.890450e-01 0.76594345 1.030161e+00 1799.925

[3,] 3.362015e-02 0.89541614 3.754696e-02 1799.925

[4,] -6.515046e-02 0.75456914 -8.634128e-02 1799.925

[5,] 1.527109e-02 0.60102948 2.540822e-02 1799.925

[6,] 3.257255e-01 0.71833909 4.534425e-01 1799.925

[7,] 1.217803e-01 1.04832696 1.161663e-01 1799.925

[8,] -2.296663e-01 0.72529420 -3.166526e-01 1799.925

[9,] 2.245642e-01 0.69813463 3.216632e-01 1799.925

[10,] 3.418277e-01 0.82344338 4.151199e-01 1799.925

[11,] 2.447368e-01 0.57169737 4.280880e-01 1799.925

[12,] 5.856425e-01 0.52471942 1.116106e+00 1799.925

[13,] 1.548076e-01 0.82058434 1.886553e-01 1799.925

[14,] 9.895625e-02 0.77860307 1.270946e-01 1799.925

[15,] -2.096737e-01 0.76573425 -2.738205e-01 1799.925

[16,] -9.280580e-01 0.82295051 -1.127720e+00 1799.925

[17,] -1.276078e+00 0.91153482 -1.399922e+00 1799.925

[18,] -1.678884e+00 0.79724237 -2.105864e+00 1799.925

[19,] -5.938515e+01 0.18759140 -3.165665e+02 1799.925

[20,] -8.349744e+01 0.17016432 -4.906871e+02 1799.925

[21,] 1.387313e+01 0.15359240 9.032433e+01 1799.925

[22,] -1.665413e+02 0.16570131 -1.005069e+03 1799.925

[23,] 5.396102e+01 0.13605570 3.966098e+02 1799.925

[24,] 9.618021e+01 0.13620736 7.061308e+02 1799.925

[25,] 1.484939e+02 0.14170707 1.047893e+03 1799.925

[26,] -1.683358e+00 0.13761576 -1.223231e+01 1799.925

[27,] -6.985643e+00 0.13693462 -5.101444e+01 1799.925

[28,] 5.169123e+01 0.13914714 3.714861e+02 1799.925

[29,] 1.183322e+02 0.11037997 1.072044e+03 1799.925

[30,] 8.189466e+01 0.10571551 7.746702e+02 1799.925

[31,] 1.704048e+01 0.10989229 1.550653e+02 1799.925

[32,] -5.714327e+01 0.10459109 -5.463494e+02 1799.925

[33,] -1.510924e+02 0.15761235 -9.586327e+02 1799.925

[34,] -3.960084e+02 0.15792492 -2.507574e+03 1799.925

[35,] -6.301457e+02 0.16736143 -3.765179e+03 1799.925

[36,] -1.092090e+03 0.17075100 -6.395805e+03 1799.925

[37,] 7.698059e+01 0.19787646 3.890336e+02 1799.925

[38,] -2.083697e+02 0.18597822 -1.120398e+03 1799.925

[39,] 6.860368e+01 0.16542558 4.147102e+02 1799.925

[40,] -1.715161e+01 0.16386822 -1.046671e+02 1799.925

[41,] 3.172470e+02 0.14611122 2.171271e+03 1799.925

[42,] -1.845403e+02 0.14940385 -1.235178e+03 1799.925

[43,] -1.076782e+02 0.16127888 -6.676520e+02 1799.925

[44,] 2.926974e+01 0.14715725 1.989011e+02 1799.925

[45,] -1.459034e+02 0.15325584 -9.520253e+02 1799.925

[46,] 1.469059e+02 0.17135894 8.572993e+02 1799.925

[47,] -2.510377e+02 0.12490564 -2.009819e+03 1799.925

[48,] 1.284678e+01 0.12403879 1.035707e+02 1799.925

[49,] -3.362748e+02 0.11554787 -2.910264e+03 1799.925

[50,] 5.005456e+01 0.11483386 4.358868e+02 1799.925

[51,] -4.072882e+01 0.13327433 -3.056014e+02 1799.925

[52,] 9.796624e+01 0.16510257 5.933659e+02 1799.925

[53,] 1.558021e+02 0.22834792 6.823015e+02 1799.925

[54,] -2.213495e+01 0.39857594 -5.553509e+01 1799.925

[55,] 8.606348e+02 0.01608463 5.350665e+04 1799.925

[56,] 2.117838e+02 0.04756842 4.452193e+03 1799.925

> performance\_3

[1] 0.6423358

> Result\_3

Estimate SE Tvalue minusll

[1,] 0.25736323 0.16320854 1.57689809 3233.27

[2,] 1.50267310 0.14837587 10.12747654 3233.27

[3,] 0.49051067 0.14097158 3.47950037 3233.27

[4,] 0.08501966 0.14870037 0.57175149 3233.27

[5,] 0.22599103 0.12301564 1.83709193 3233.27

[6,] 0.53618152 0.11791025 4.54736999 3233.27

[7,] 0.33705564 0.12726143 2.64852936 3233.27

[8,] 0.16193190 0.12659561 1.27912726 3233.27

[9,] 0.71857024 0.12413644 5.78855183 3233.27

[10,] 0.82350027 0.12900472 6.38348969 3233.27

[11,] 0.24959148 0.09647654 2.58706907 3233.27

[12,] 0.52907855 0.09446894 5.60055571 3233.27

[13,] 0.07941359 0.09757115 0.81390434 3233.27

[14,] 0.15074053 0.09622623 1.56652220 3233.27

[15,] -0.03728466 0.14044903 -0.26546752 3233.27

[16,] 0.08818147 0.14251527 0.61875106 3233.27

[17,] -0.14276988 0.15200336 -0.93925474 3233.27

[18,] -0.46973107 0.16287229 -2.88404536 3233.27

[19,] 0.51942642 0.21187767 2.45153921 3233.27

[20,] 0.40084971 0.18701907 2.14336274 3233.27

[21,] 0.56110452 0.18219709 3.07965690 3233.27

[22,] 0.31435023 0.17705943 1.77539395 3233.27

[23,] 0.23043885 0.16257451 1.41743529 3233.27

[24,] 0.47049718 0.15589661 3.01800774 3233.27

[25,] 0.47822037 0.17245174 2.77306774 3233.27

[26,] 0.50940586 0.15707917 3.24298798 3233.27

[27,] 0.54884765 0.16380080 3.35070192 3233.27

[28,] 0.52336188 0.19678613 2.65954654 3233.27

[29,] 0.48740530 0.13711031 3.55484058 3233.27

[30,] 1.05310060 0.13983771 7.53087683 3233.27

[31,] 0.15451381 0.12658161 1.22066555 3233.27

[32,] 0.10151830 0.12459434 0.81479060 3233.27

[33,] -0.57927252 0.14508681 -3.99259263 3233.27

[34,] -1.82212578 0.20433347 -8.91741223 3233.27

[35,] -3.18116261 0.28096979 -11.32208043 3233.27

[36,] -4.86064469 0.43403801 -11.19866133 3233.27

[37,] 13.57463715 7.42684664 1.82777938 3233.27

[38,] -4.04831285 9.07452209 -0.44611857 3233.27

[39,] 4.84582874 4.96275094 0.97644004 3233.27

[40,] -10.57037142 10.60426922 -0.99680338 3233.27

[41,] -0.95626498 15.09552656 -0.06334757 3233.27

[42,] 0.77699289 11.64775861 0.06670750 3233.27

[43,] 5.60991784 12.17414630 0.46080585 3233.27

[44,] 5.18629508 6.54547205 0.79234852 3233.27

[45,] 9.66209250 9.01930223 1.07126829 3233.27

[46,] 3.01858004 12.64853937 0.23865048 3233.27

[47,] 2.44021434 7.48229608 0.32613175 3233.27

[48,] 4.72482738 3.13853277 1.50542554 3233.27

[49,] -6.24884647 9.80258862 -0.63746901 3233.27

[50,] -6.89455446 5.09472767 -1.35327242 3233.27

[51,] -1.79688682 9.62961164 -0.18660013 3233.27

[52,] 4.54331259 13.47340993 0.33720585 3233.27

[53,] 16.97632609 28.89495776 0.58751863 3233.27

[54,] 8.90743136 19.34370868 0.46048209 3233.27

[55,] 0.51491086 0.04866701 10.58028514 3233.27

[56,] 0.47075397 0.04860298 9.68570098 3233.27