Rayyan Sayeed

MKTG 551-3, Homework 1

April 10th, 2018

1 Multinomial Logit Choice Model

Here, we implement the multinomial logit choice model on the indirect utility function using log-likelihood, and then optimize the log-likelihood using various optimization procedures. We use the data file hh100.csv. The code used for this portion with nlm optimization is included below:

```
hh100<- read.csv("hh100.csv",header=T)
hh100$ints<-rep(1,length(hh100$hh.id)) # add column of 1's, will need later, alphas no
    coeff

# theta vector; (a1,a2,a3,bf,bp)
theta<-c(2,0,-2,1,-30) # true theta
Ltheta<-function(theta,x){
# identify trip block
for(i in 1:length(x$hh.id)){
  if(i%%4==1){
    denom<-0 # zero the denominator at beginning of each trip
  }
  item<-x$alt.id[i] # assign proper item id
  if(item==4) beta <- c(theta[4],theta[5],0) else
     beta<-c(theta[4],theta[5],theta[item]) # define beta
  denom<-denom + exp(beta %*% t(as.matrix(x[i,c("feat","price","ints")]))) # update
     denom
  num<-exp(beta %*% t(as.matrix(x[i,c("feat","price","ints")]))) # define numerator
  Prob<-num # define choice probability
  x$Prob[i]<-Prob
```

```
  if(item==4){ # make sure denom gets used for every point in trip
    for(j in 0:3){
    x$Prob[i-j]<- x$Prob[i-j]/denom
    }
  }
}
  output<- t(as.matrix(x$y)) %*% log(as.matrix(x$Prob)) # log-likelihood formula
  return(-output)
}
result <-nlm(Ltheta,theta,hh100,print.level= 2,hessian=T) # optimize
```

Note that the code shown above uses the true input vector, $\theta_a^0 = \{2, 0, -2, 1, -30\}$.

2 nlm, BFGS, and Nelder-Mead Optimization

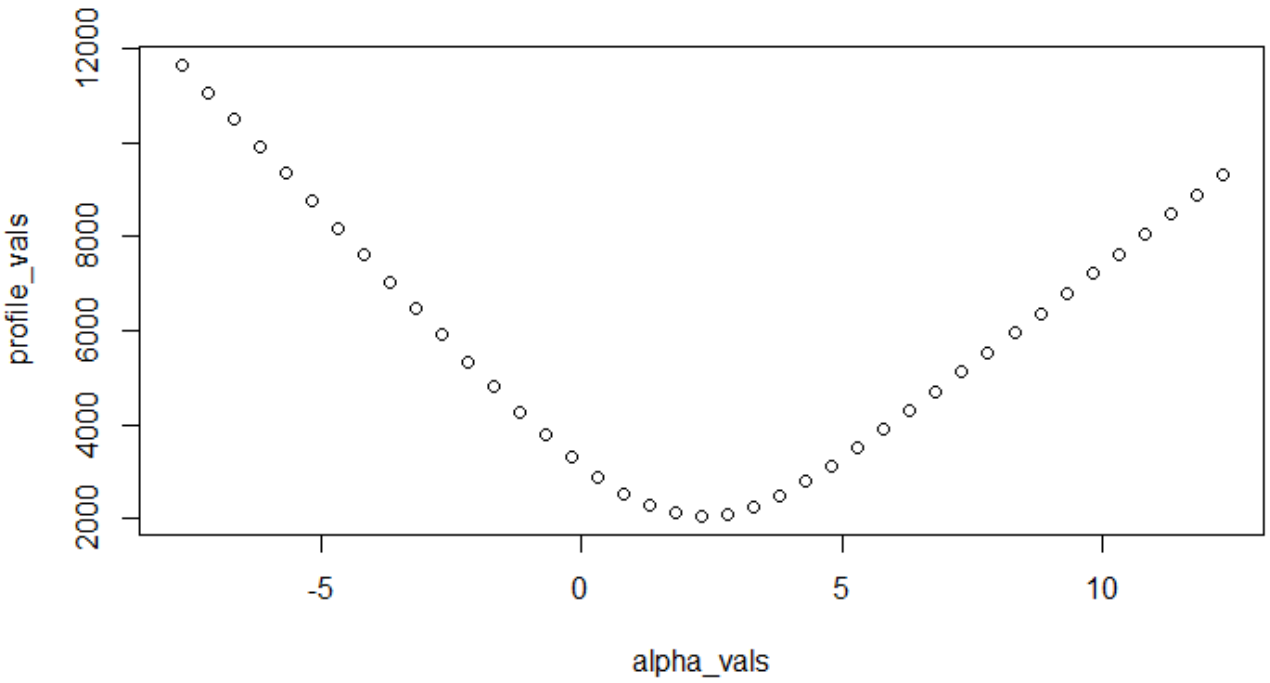
We run the nlm optimization routine on three different initial input vectors, given by: $\theta_a^0 = \{2, 0, -2, 1, -30\}$, $\theta_b^0 = \{0, 0, 0, 0, -10\}$, $\theta_c^0 = \{100, 100, 100, 100, -100\}$. Examining the outputs of the nlm routine, we see that each iteration of output returns the number of the iteration, the value of the parameters at that point, the log-likelihood at that point, and the gradient with respect to each of the parameters. Moving from one iteration to the next, we see that the parameter values approach the values of the true input vector, the log-likelihood approaches the true optimum of 2055.1444, and the gradient with respect to every parameter approaches zero, indicating that we have found the optimum.

Intuitively, we would expect that as our initial vector moves farther away from the true vector, the optimization routine will take longer (and more iterations) to converge to the optimum. This belief is confirmed by our results. We run BFGS and Nelder-Mead optimization routines for $\theta_b^0 = \{0, 0, 0, 0, -10\}$, and $\theta_c^0 = \{100, 100, 100, 100, -100\}$. We find that the BFGS routine outperforms the Nelder-Mead routine in both cases, both in accuracy of the optimum and in time to run. Further, the BFGS routine for θ_b^0 returns an initial log-likelihood value of 2998.251, and after 10 iterations, returns the true optimum of 2055.144. The BFGS routine for θ_c^0 converges as well, but slower, as expected: we see the log-likelihood value after 10 iterations to be 2655.526, after 20 to be 2058.196,

and after 30 to be the true optimum of 2055.144. The Nelder-Mead routine for θ_b^0 converges to the true optimum of 2055.144; however the routine fails to converge for θ_c^0, returning a false optimum of 4407.652. The fact that the Nelder-Mead routine failed to converge for θ_c^0 indicates that the method is sensitive to distance from the true input vector, and the method most likely spent too much time examining local maxima that are not true global optima. Conceivably, if we increased the number of iterations sufficiently, we would see convergence for θ_c^0 as well, but using the default of 502 iterations was not sufficient to see convergence here.

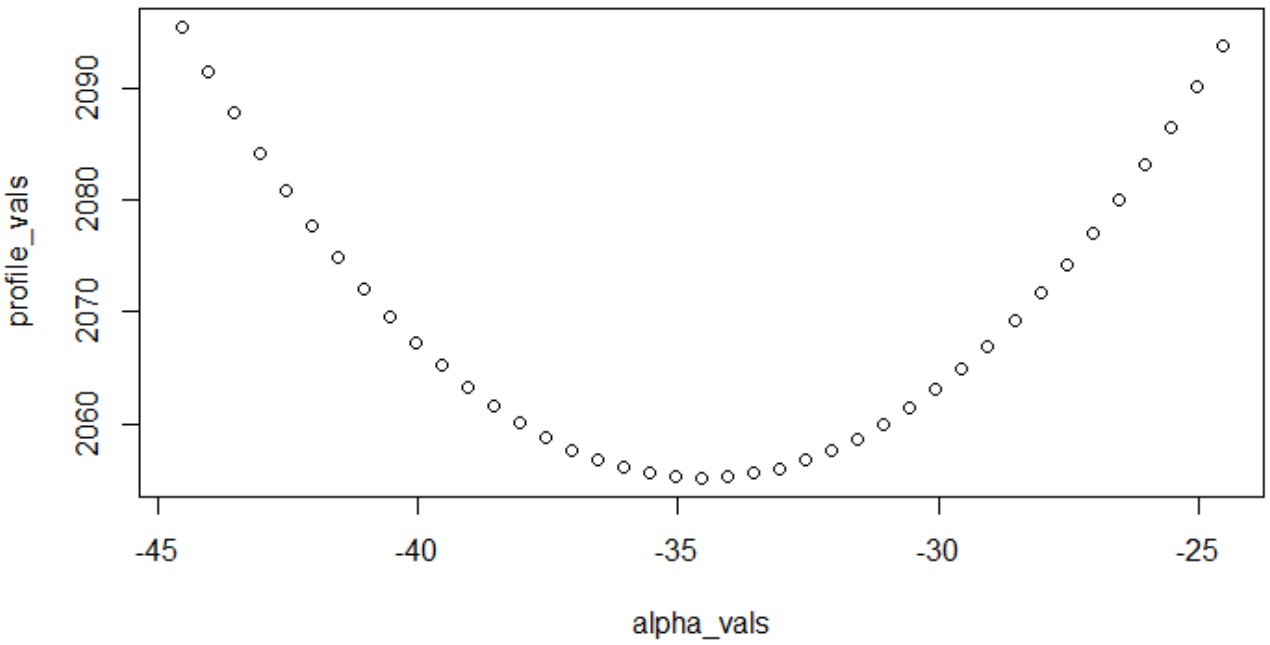
3 Profile Likelihood Plot

We include below a plot of the log-likelihood in a neighborhood of ± 10 units from the value of the predicted first intercept, $\hat{\alpha_1} = 2.306$, returned by the BFGS routine previously, with step size of .5 units.



We see from the above that our value of $\hat{\alpha_1}$ is very close to the true optimum.

We include below a plot of the log-likelihood in a neighborhood of ± 10 units from the value of the predicted price parameter, $\hat{\beta}_p = -34.547$, with step size of .5 units.



We see from the above that our value of $\hat{\beta}_p$ is very close to the true optimum.

From both plots, we can see that the objective function is roughly a parabola in the neighborhood of the predicted price parameter and first intercept.

4 Result Statistics

Using the results of the nlm routine with the true input vector as the initial point, we find the coefficient estimates to be $\{2.305, 0.275, -1.783, 0.839, -34.522\}$, standard errors to be $\{0.0000533, 0.000655, 0.00101$ log-likelihood to be 2055.144 at optimum, goodness of fit to be .259, and BIC to be -4065.352. We note the following: the coefficient estimates are reasonably close to the true input vector, the standard errors for each parameter are very small, the log-likelihood is identical to the true optimum up to the thousandth, and goodness of fit and BIC reflect our model to be quite successful.

5 Own-price and Cross-price Elasticities

We find the mean own-price elasticity to be 1.995. Since this value is greater than 1, we conclude that on average, the goods considered in this example are elastic, and a change in price will have a proportionately larger change in quantity demanded. Further, we find the mean cross-price elasticity to be -.768. Since this value is negative, we conclude that on average, these goods are complements to each other, rather than substitutes, and that an increase in price of a complementary good will decrease the demand for the good in question.