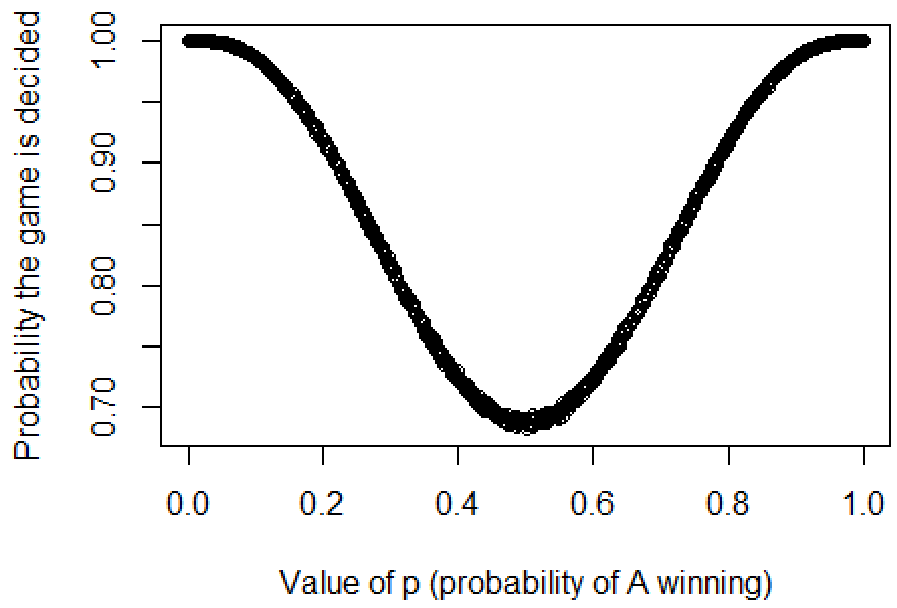
Stochastic Control and Optimization - Homework 6

Akankshi Mody – am92786

Question1

```r
#Probability of A winning = p
total_prob = c()
for(p in seq(0, 1, by = 0.001)){
  #Probability that A will win more than 3 games
  prob_a = mean(rbinom(100000,6,p)>3)
  #Probability that B will win more than 3 games
  prob_b = mean(rbinom(100000,6,(1-p))>3)
  #Probability that the game will be decided before the 7th game
  total_prob = c(total_prob,prob_a+prob_b)
}
#print(total_prob)
plot(seq(0,1,by=0.001),total_prob, xlab = "Value of p (probability of A winni
ng)", ylab = "Probability the game is decided")
```



Question2

```r
rev = c()
for (i in 41:50){
  #probability that the number of customers showing up will be > 40
  prob = (mean(rbinom(100000,i,0.9)>40))
  if (i*prob > 40){
    #If more than 40 customers show up
    revenue = i*10 - (ceiling(i*prob)-40)*25
  }
  else{
    #if less than 40 customers show up
    revenue = i*10
  }
  rev = c(rev,revenue)
}

cat("Number of tickets he should sell to maximize revenue = ", (match(max(rev
),rev))+40)

## Number of tickets he should sell to maximize revenue =  47
```

Question 4

```r
#no. of simulations
N=1000

#Door label list - you can even use: doors = c('A','B','C')
doors = c(1:33)
no_revealed = 5
stay = rep(NA,N)
swit = rep(NA,N)

for (i in 1:N){

  #Generate Random Variables
  car = doors[ceiling(runif(1)*length(doors))]
  pick = doors[ceiling(runif(1)*length(doors))]
  #Run the Simulation - that is, play the game
  hostChoices  = setdiff(doors,union(pick,car))

  #Host has to reveal multiple doors (stored in no_revealed)
  host = c()
  for (j in 1:no_revealed){
    host = c(host,hostChoices[ceiling(runif(1)*length(hostChoices))])
  }
  #host = hostChoices[ceiling(runif(1)*length(hostChoices))]
  switchedDoor = setdiff(doors,union(pick,host))
  switchedDoor = switchedDoor[ceiling(runif(1)*length(switchedDoor))]
  #remember output - that is, win or not for each strategy
  stay[i] = (pick==car)
  swit[i] = (switchedDoor==car)

}
```

```
#summarize output
print(paste("Prob. of winning if we do not switch doors:", mean(stay)))

## [1] "Prob. of winning if we do not switch doors: 0.036"

print(paste("Prob. of winning if we switch doors       :", mean(swit)))

## [1] "Prob. of winning if we switch doors       : 0.025"
```

Question5

```
#Number of simulations
N = 10000
#Vector to store how much money person i gets
lucky_money = rep(NA, 10)
probs = rep(0,10)
for (i in 1:N){
  total_amount = 100
  # calculating the allocation per person
  for (j in 1:10){
    lucky_money[j]=runif(1)*total_amount
    total_amount = total_amount - lucky_money[j]
  }
  #Finding which person got max allocation
  max_i = (match(max(lucky_money),lucky_money))
  #increasing count of the person who got max by 1
  probs[max_i] = probs[max_i]+1
}
#Dividing the probs vector by number of simulations to get probability
probs = probs/N
print (probs)

##  [1] 0.6239 0.2507 0.0875 0.0279 0.0079 0.0016 0.0004 0.0000 0.0001 0.0000
```