STOCHASTIC Project 6

Siddhant Shah (sss3976), Sidhaarthan VG (sv24335), Bhavana Vijay (bv5487), Jushira Thelakkat (jt39634)

Question 1,2

States: s, p, x [s = self score; p = opponent's score; x = current time value]

Action: Roll or Hold

Bellman equation:

V (s, 100, x) = 0; V (100, p, x) = 1; V (99, p, x) = 1;

V (s, p, x) = max [p1(1-V (p, s+1,0)) + p2*V (s, p, x+z) + p3*V (s, p, x+3) + p4*V (s, p, x+4) + p5*V (s, p, x+5) + p6*V (s, p, x+6), (1- V (p, s+x, 0))]

where Pn is the Probability of rolling n

Question 3,4

```
gameStrategy <- function(p_goal){


  goal = p_goal+1
  all_comb = expand.grid(seq(goal-2),seq(goal-1))    # Generating all combinations of each index
  all_comb$sum = all_comb[[1]] + all_comb[[2]]       # Finding max sum
  all_comb = all_comb[order(all_comb$sum, decreasing = TRUE),]


  V = array(NA, dim=c(goal+5,goal+5,goal+5))           # s,p,x
  U = array(NA, dim=c(goal+5,goal+5,goal+5))           # s,p,x


# Initialize boundaries
  V[goal:(goal+5),,] = 1              # 100+ points, then you win regardless of opponent's score
  V[goal-1,seq(1,goal-1),] = 1     # Your turn and you have goal-1 points, you will win
  V[seq(1,goal-1),goal:(goal+5),] = 0    # Opponent reaches the goal first then you lose.
  V[,seq(1,goal-1),goal:(goal+5)] = 1    # Accumulate the goal amount in one round and win
# Fill in remaining V and U matrix
  for(r in 1:length(all_comb$sum)){
    for(x in (goal):1){
      s = all_comb[r,1]
      p = all_comb[r,2]
      V[s,p,x] = max( ( (1/6)*(1-V[min(p,goal),s+1,1]) + (1/6)*V[min(s,goal),p,min(x+2,goal)] + (1/6)*V[min(s,goal),p,min(x+3,goal)]
                        + (1/6)*V[min(s,goal),p,min(x+4,goal)] + (1/6)*V[min(s,goal),p,min(x+5,goal)]
                        + (1/6)*V[min(s,goal),p,min(x+6,goal)] ), 1-V[min(p,goal),min(s+max(x-1,1),goal),1] )  #Roll/Hold
      U[s,p,x] = which.max( c( ( (1/6)*(1-V[min(p,goal),s+1,1]) + (1/6)*V[min(s,goal),p,min(x+2,goal)] + (1/6)*V[min(s,goal),p,min(x+3,goal)]
                        + (1/6)*V[min(s,goal),p,min(x+4,goal)] + (1/6)*V[min(s,goal),p,min(x+5,goal)]
                        + (1/6)*V[min(s,goal),p,min(x+6,goal)] ), 1-V[min(p,goal),min(s+max(1,x-1),goal),1] ) )  #Roll/Hold
    }
  }
  save(list = c('V','U'),file = 'VUfile.Rdata')
}
```

Bonus

We now can't calculate the V and U matrices because we now don't know the probability of a number appearing during a dice roll.

Strategy: We played the first round with the assumption of a fair dice and used the dice rolls from the first round to estimate the dice roll distribution. We then passed the estimated probabilities into our dynamic program to re-calculate the V and U matrices for the next round.

We provided the modified server.R file that, on clicking new game, will estimate the new distribution and then call our function to re-calculate the V and U matrices. After you finish one game, clicking "new game" will re-estimate the distribution and the next game will be run with the updated V/U matrices. Every additional game you play will improve the estimation of the distribution.