STA 141A Spring HW 3

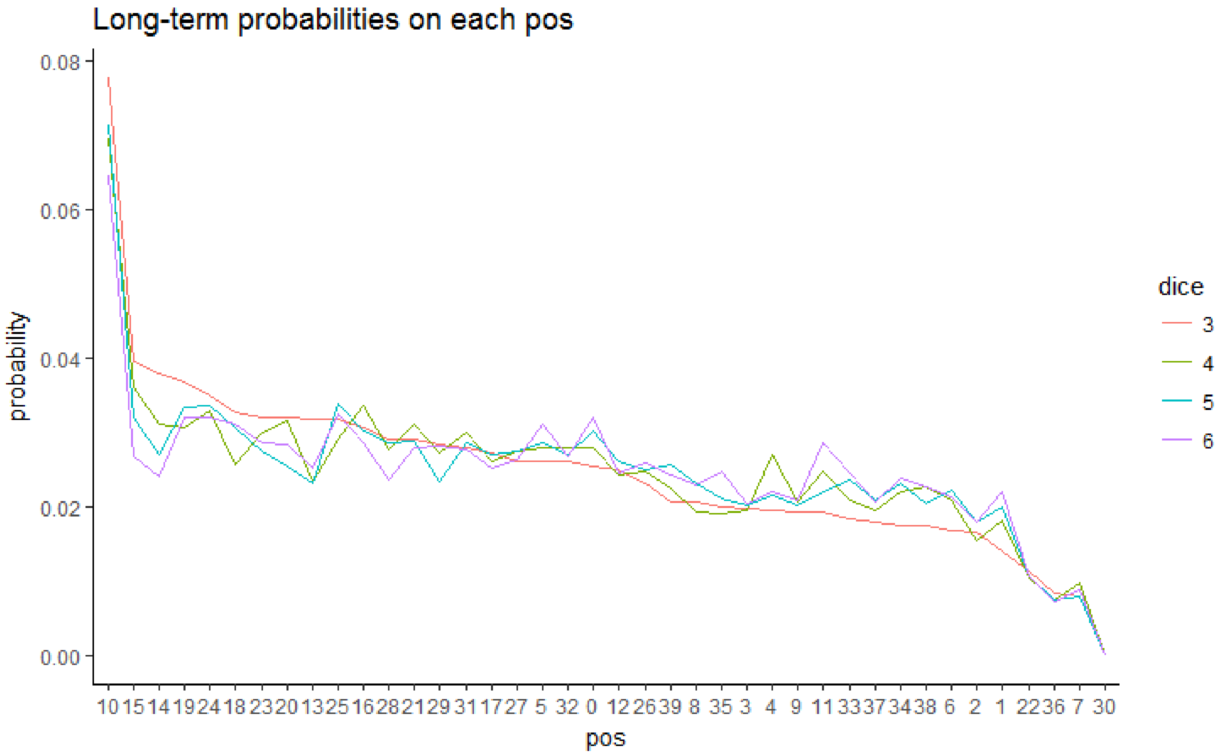
May 22, 2018

Zihan Mo 914998952

Part I

1.1 See the appendix

1.2



The 3 most likely squares to end a turn:

3-sided: 10 (0.078), 15 (0.0396), and 14 (0.038);   4-sided: 10 (0.0696), 15 (0.0363), 24 (0.033)

5-sided: 10 (0.0714), 25 (0.0339), 24 (0.0338);   6-sided: 10 (0.0646), 25 (0.0322), 0 (0.03212)

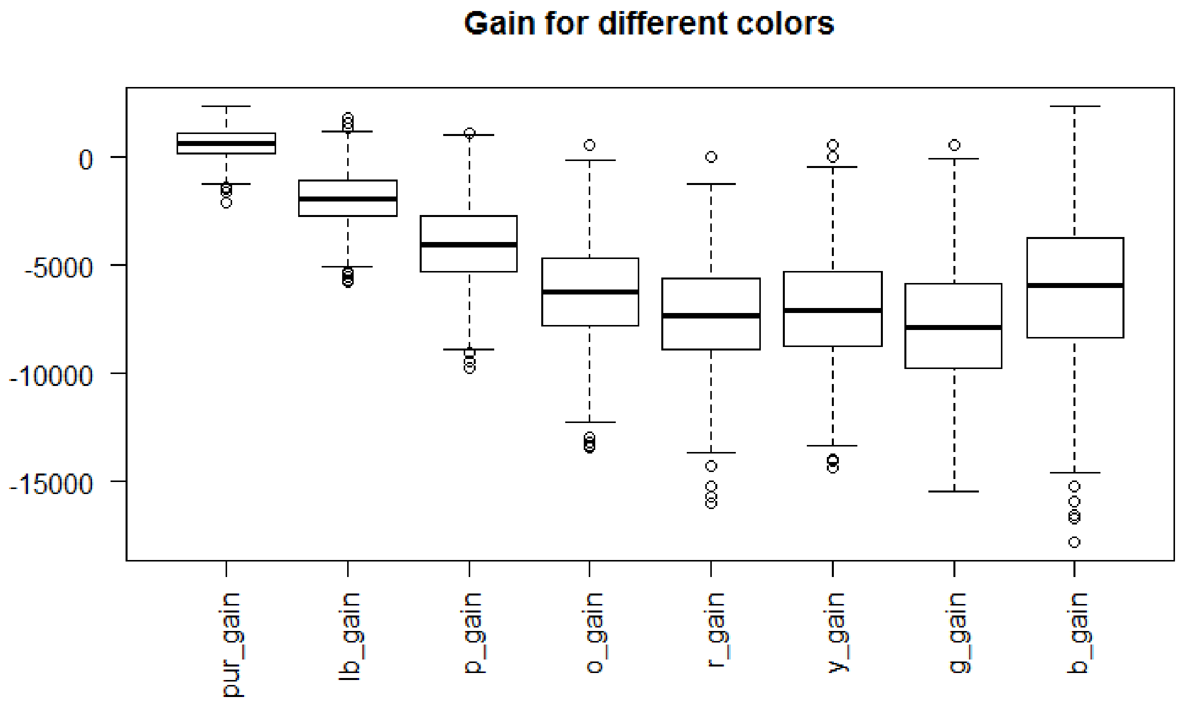
It’s obvious that the probability of landing on square 10 is much higher than landing on other squares. The probabilities of landing on other squares are similar. Moreover, there are no chance the position lands on 30 because the position is transfer to 10 automatically.

1.3

The standard error standard error for the long-term probability of ending a turn in jail is 0.0020678. It means the variation of long-term probability of ending a turn in jail is very samll.

2.1 See appendix.

2.2



Purple is the least effective because the player is gainning money. Green is the most effective because it has the smallest median.

Reference:

TA: Patrick and Jiahui

Piazza

Appendix

```
library(ggplot2)

#PART I 1.1
shuffle<-function(){
  x<-1:16
  sample(x,length(x))
}
cards_arrange<-function(x){
  x<-c(x[-1],x[1])
  return(x)
}
simulate_monopoly<-function(n,d){
  CC_position<-c(2,17,33)
  CC_instruction<-c(0,10)
  CH_position<-c(7,22,36)
  CH_instruction1<-c(0,10,11,24,39,5)
  CH_instruction2<-NA
  CH_instruction2[CH_position]<-c(15,25,5)
  CH_instruction3<-NA
  CH_instruction3[CH_position]<-c(12,28,12)
  CC_cards<-shuffle()
  CH_cards<-shuffle()
  pos=0
  result<-numeric()
  result[1]<-0
  dice1<-sample(d,n,replace = T)
  dice2<-sample(d,n,replace = T)
  roll = dice1+dice2
```

```
double=dice1==dice2
counter=0
for (i in 1:n){
  pos= pos + roll[i]
  if(pos>39){
    pos=pos%%40
  }
  if(double[i]==T){
    counter=counter+1
  }else{
    counter=0
  }
  if(i>2&counter==3){
    pos=10
    counter=0
  }else{
    counter=counter
  }
  if(pos %in% CC_position){
    CC_card=CC_cards[1]
    CC_cards=cards_arrange(CC_cards)
    if (CC_card == 1 | CC_card == 2){
      pos=CC_instruction[CC_card]
    }else{
      pos=pos
    }
  }
  if(pos %in% CH_position){
    CH_card=CH_cards[1]
    CH_cards=cards_arrange(CH_cards)
```

```
   if(CH_card %in% c(1:6)){
    pos=CH_instruction1[CH_card]
   }else if(CH_card == 7|CH_card==8){
    pos=CH_instruction2[pos]
   }else if(CH_card==9){
    pos=CH_instruction3[pos]
   }else if(CH_card == 10){
    pos=pos-3
    if (pos == 33){
     CC_card=CC_cards[1]
     if(CC_card==1|CC_card==2){
      pos=CC_instruction[CC_card]
     }else{
      pos=pos
     }
    }
   }
  }
  if(pos == 30){
   pos=10
  }
  result[i+1]<-pos
 }
 return(factor(result,0:39))
}

#1.2
estimate_monopoly<-function(n,d){
 prob<-sort(prop.table(table(simulate_monopoly(n,d))),decreasing = T)
 return(prob)
```

```
}

dice3<-estimate_monopoly(10000,3)
dice4<-estimate_monopoly(10000,4)
dice5<-estimate_monopoly(10000,5)
dice6<-estimate_monopoly(10000,6)
dices<-c(dice3,dice4,dice5,dice6)

dice3<-data.frame(dice3)
dice4<-data.frame(dice4)
dice5<-data.frame(dice5)
dice6<-data.frame(dice6)

dice3$dice<-3
dice4$dice<-4
dice5$dice<-5
dice6$dice<-6
prob<-rbind(dice3,dice4,dice5,dice6)
colnames(prob)[1]<-'pos'
prob$dice<-as.factor(prob$dice)

ggplot(prob,aes(x=pos,y=Freq,color=dice))+
  geom_line(aes(group=dice))+ylab("probability")+
  ggtitle("Long-term probabilities on each pos")+
  theme_classic()

#dice3:10,14,15
#dice4:10,24,15
#dice5:10,25,18
#dice6:10,19,20
```

```
#1.3
onek<-replicate(1000,estimate_monopoly(10000,6))
se<-sd(onek[1,])

#2.1
simulate_monopoly2<-function(n,d,p,r){
  CC_position<-c(2,17,33)
  CC_instruction<-c(0,10)
  CH_position<-c(7,22,36)
  CH_instruction1<-c(0,10,11,24,39,5)
  CH_instruction2<-NA
  CH_instruction2[CH_position]<-c(15,25,5)
  CH_instruction3<-NA
  CH_instruction3[CH_position]<-c(12,28,12)
  p_instruction<-NA
  p_instruction[p]<-r
  CC_cards<-shuffle()
  CH_cards<-shuffle()
  pos = 0
  result<-numeric()
  wealth<-numeric()
  result[1]<-0
  wealth[1]<-0
  dice1<-sample(d,n,replace = T)
  dice2<-sample(d,n,replace = T)
  roll = dice1+dice2
  double=dice1==dice2
  counter=0
  for (i in 1:n){
```

```
money = 0
Go = F
Jail = F
pos= pos + roll[i]
if(pos>39){
  Go=T
  pos=pos%%40
}
if(double[i]==T){
  counter=counter+1
}else{
  counter=0
}
if(i>2&counter==3){
  pos=10
  counter=0
}else{
  counter=counter
}
if(pos %in% CC_position){
  CC_card=CC_cards[1]
  CC_cards=cards_arrange(CC_cards)
  if (CC_card == 1 | CC_card == 2){
    pos=CC_instruction[CC_card]
  }else{
    pos=pos
  }
}
if(pos %in% CH_position){
  CH_card=CH_cards[1]
  CH_cards=cards_arrange(CH_cards)
  if(CH_card %in% c(1:6)){
    pos=CH_instruction1[CH_card]
  }else if(CH_card == 7|CH_card==8){
    pos=CH_instruction2[pos]
  }else if(CH_card==9){
    pos=CH_instruction3[pos]
  }else if(CH_card == 10){
    pos=pos-3
    if (pos == 33){
      CC_card=CC_cards[1]
      if(CC_card==1|CC_card==2){
        pos=CC_instruction[CC_card]
      }else{
        pos=pos
      }
    }
  }
}
if(pos == 30){
  pos=10
}
if(pos == 10){
  Jail=T
}
if(pos==4){
  money = money-200
}else if(pos==38){
  money=money-100
}
```

```
    if(pos%in%p){
      money=money-p_instruction[pos]
    }
    if(Go==T&Jail==F){
      money=money+200
    }
    result[i+1]<-pos
    wealth[i+1]<-money
  }
  a<-cbind(result,wealth)
  return(data.frame(a))
}

#2.2
property<-read.csv('properties.csv')
#subsetting the datas based on different colors
purple<-subset(property,Color=='Purple')
lightB<-subset(property,Color=='Light Blue')
pink<-subset(property,Color=='Pink')
orange<-subset(property,Color=='Orange')
red<-subset(property,Color=='Red')
yellow<-subset(property,Color=='Yellow')
green<-subset(property,Color=='Green')
blue<-subset(property,Color=='Blue')

#simulate for different colors
pursim<-replicate(1000,simulate_monopoly2(100,6,purple$Index,
                       purple$Rent))
lbsim<-replicate(1000,simulate_monopoly2(100,6,lightB$Index,
                       lightB$Rent))
```

```
psim<-replicate(1000,simulate_monopoly2(100,6,pink$Index,
                              pink$Rent))
osim<-replicate(1000,simulate_monopoly2(100,6,orange$Index,
                              orange$Rent))
rsim<-replicate(1000,simulate_monopoly2(100,6,red$Index,
                              red$Rent))
ysim<-replicate(1000,simulate_monopoly2(100,6,yellow$Index,
                              yellow$Rent))
gsim<-replicate(1000,simulate_monopoly2(100,6,green$Index,
                              green$Rent))
bsim<-replicate(1000,simulate_monopoly2(100,6,blue$Index,
                              blue$Rent))
pursim<-data.frame(pursim)
pur_gain<-sapply(pursim,function(x)sum(unlist(x[2])))
pur_gain<-data.frame(pur_gain)

lbsim<-data.frame(lbsim)
lb_gain<-sapply(lbsim,function(x)sum(unlist(x[2])))
lb_gain<-data.frame(lb_gain)

psim<-data.frame(psim)
p_gain<-sapply(psim,function(x)sum(unlist(x[2])))
p_gain<-data.frame(p_gain)

osim<-data.frame(osim)
o_gain<-sapply(osim,function(x)sum(unlist(x[2])))
o_gain<-data.frame(o_gain)

rsim<-data.frame(rsim)
r_gain<-sapply(rsim,function(x)sum(unlist(x[2])))
```

```
r_gain<-data.frame(r_gain)

ysim<-data.frame(ysim)
y_gain<-sapply(ysim,function(x)sum(unlist(x[2])))
y_gain<-data.frame(y_gain)

gsim<-data.frame(gsim)
g_gain<-sapply(gsim,function(x)sum(unlist(x[2])))
g_gain<-data.frame(g_gain)

bsim<-data.frame(bsim)
b_gain<-sapply(bsim,function(x)sum(unlist(x[2])))
b_gain<-data.frame(b_gain)

eightcol<-cbind(pur_gain,lb_gain,p_gain,o_gain,r_gain,y_gain,g_gain,b_gain)
boxplot(eightcol,main="Gain for different colors",las=2)
```