**Homework Assignment 4**

**STA 141A A02**

**Submitted By:**

#1

stimulate\_monopoly=function(n=10000 , d=6)

{

SQUARES <- 40 CC\_CARDS <- 16 CH\_CARDS <- 16

GO <- 0 A1 <- 1 CC1 <- 2 A2 <- 3

R1 <- 5 B1 <- 6 CH1 <- 7

B2 <- 8 B3 <- 9 JAIL <- 10

C1 <- 11 U1 <- 12 C2 <- 13

C3 <- 14 R2 <- 15 D1 <- 16

CC2 <- 17 D2 <- 18 D3 <- 19

FP <- 20 E1 <- 21 CH2 <- 22

E2 <- 23 E3 <- 24 R3 <- 25

F1 <- 26 F2 <- 27 U2 <- 28

F3 <- 29 G2J <- 30 G1 <- 31

G2 <- 32 CC3 <- 33 G3 <- 34

R4 <- 35 CH3 <- 36 H1 <- 37

T2 <- 38 H2 <- 39 T1 <- 4

die1 <- sample(1:d, n, replace=TRUE)

die2 <- sample(1:d, n, replace=TRUE)

z<-die1+die2

doubles=(die1==die2)

cur<-0

cur<-(cur+z) %% 39

p<-sample(0,n,replace=TRUE)

for(i in 1:(n))

{

p[i]<-cur[i]

}

pos<-sample(0,n+1,replace=TRUE)

for(i in 2:(n+1))

{

pos[i]<-pos[i-1]+p[i-1]

pos[i]<-pos[i]%%39

}

count<-0

for(i in 1:(n-2))

{

if(i >2 & doubles[i] == TRUE & doubles[i+1] == TRUE & doubles[i+2] == TRUE)

{

pos[i+2]<-10

}

}

#if current = G2J go to jail

for(i in 1:(n+1))

{

if(pos[i]==30)

{

pos[i]=10

}

}

#create a deck of 16 cards

for(i in 1:n)

{

if ( pos[i] %in% c(CC1, CC2, CC3) ) {

cc <- sample(1:CC\_CARDS, 1)

if ( cc == 1 ) {

pos[i] <- GO

} else if ( cc == 2 ) {

pos[i] <- JAIL

}

}

}

for(i in 1:n)

{

if ( pos[i] %in% c(CH1, CH2, CH3) ) {

ch <- sample(1:CH\_CARDS, 1)

if ( ch == 1 ) {

pos[i] <- GO

} else if ( ch == 2 ) {

pos[i] <- JAIL

} else if ( ch == 3 ) {

pos[i] <- C1

} else if ( ch == 4 ) {

pos[i] <- E3

} else if ( ch == 5 ) {

pos[i] <- H2

} else if ( ch == 6 ) {

pos[i] <- R1

} else if ( (ch == 7 || ch == 8) %% pos[i] == CH1 ) {

pos[i] <- R2

} else if ( (ch == 7 || ch == 8) %% pos[i] == CH2 ) {

pos[i] <- R3

} else if ( (ch == 7 || ch == 8) %% pos[i] == CH3 ) {

pos[i] <- R1

} else if ( ch == 9 %% pos[i] %in% c(CH1, CH3) ) {

pos[i] <- U1

} else if ( ch == 10 ) {

pos[i] <- (pos[i] - 3) %% 39

}

}

}

pos

}

#2

par(mfrow=c(2,2))

estimate\_monopoly=function(n,pos)

{ x <- sample(0, 40, replace=TRUE)

for(i in 0:39)

{

count<-0

for(j in 2:(n+1))

{

if(pos[j]==i)

{

count<-count+1

}

}

x[i]<- count\*1/n

}

x

}

#3

m<-10^3

x1<-c(1:m)

for(i in 1:m)

{

print(i)

cc<-estimate\_monopoly(10000, stimulate\_monopoly(n=10000,6))

qw<-data.frame("position"=c(0:39),"probablity"=cc)

x2<-qw[qw$position==11,]

x1[i]<-x2$probablity

}

stdError <- sd(x1)/sqrt(length(x1))

# 4.544519e-05

#4

set.seed(1)

library(boot)

meanFunc <- function(x,i){mean(x[i])}

bootMean <- boot(x1,meanFunc,1000)

bootMean

#a) Bootstrap Statistics :

#original bias std. error

#t1\* 0.0256467 2.54e-08 4.59747e-05

#bootstrap estimate is faster to compute

#b) bootstrap is faster to compute why???????????

#5

.

par(mfrow=c(2,2))

m<-10^3

x1<-c(1:m)

x1<-sample(0, 40, replace=TRUE)

m<-100

x1<-sample(0, m, replace=TRUE)

x8=x9=x10=x11=x12=x13=x14=x15=x16=x17=x18=x19=x20=x2=x3=x4=x5=x6=x7=x1

x21=x22=x23=x24=x25=x26=x27=x28=x29=x30=x31=x32=x33=x34=x35=x36=x37=x38=x39=x40=x1

#x1<-sample(1:m,0,replace=TRUE)

for(i in 1:m)

{

print(i)

cc<-estimate\_monopoly(100, stimulate\_monopoly(n=100,6))

qw<-data.frame("position"=c(0:39),"probablity"=cc)

x1[i]<-qw[qw$position==0,]$probablity

x2[i]<-qw[qw$position==1,]$probablity

x3[i]<-qw[qw$position==2,]$probablity

x4[i]<-qw[qw$position==3,]$probablity

x5[i]<-qw[qw$position==4,]$probablity

x6[i]<-qw[qw$position==5,]$probablity

x7[i]<-qw[qw$position==6,]$probablity

x8[i]<-qw[qw$position==7,]$probablity

x9[i]<-qw[qw$position==8,]$probablity

x10[i]<-qw[qw$position==9,]$probablity

x11[i]<-qw[qw$position==10,]$probablity

x12[i]<-qw[qw$position==11,]$probablity

x13[i]<-qw[qw$position==12,]$probablity

x14[i]<-qw[qw$position==13,]$probablity

x15[i]<-qw[qw$position==14,]$probablity

x16[i]<-qw[qw$position==15,]$probablity

x17[i]<-qw[qw$position==16,]$probablity

x18[i]<-qw[qw$position==17,]$probablity

x19[i]<-qw[qw$position==18,]$probablity

x20[i]<-qw[qw$position==19,]$probablity

x21[i]<-qw[qw$position==20,]$probablity

x22[i]<-qw[qw$position==21,]$probablity

x23[i]<-qw[qw$position==22,]$probablity

x24[i]<-qw[qw$position==23,]$probablity

x25[i]<-qw[qw$position==24,]$probablity

x26[i]<-qw[qw$position==25,]$probablity

x27[i]<-qw[qw$position==26,]$probablity

x28[i]<-qw[qw$position==27,]$probablity

x29[i]<-qw[qw$position==28,]$probablity

x30[i]<-qw[qw$position==29,]$probablity

x31[i]<-qw[qw$position==30,]$probablity

x32[i]<-qw[qw$position==31,]$probablity

x33[i]<-qw[qw$position==32,]$probablity

x34[i]<-qw[qw$position==33,]$probablity

x35[i]<-qw[qw$position==34,]$probablity

x36[i]<-qw[qw$position==35,]$probablity

x37[i]<-qw[qw$position==36,]$probablity

x38[i]<-qw[qw$position==37,]$probablity

x39[i]<-qw[qw$position==38,]$probablity

x40[i]<-qw[qw$position==39,]$probablity

}

std<-c(1:40)

std[0]<-sd(x1)/sqrt(length(x1))

std[1]<-sd(x2)/sqrt(length(x2))

std[2]<-sd(x3)/sqrt(length(x3))

std[3]<-sd(x4)/sqrt(length(x4))

std[4]<-sd(x5)/sqrt(length(x5))

std[5]<-sd(x6)/sqrt(length(x6))

std[6]<-sd(x7)/sqrt(length(x7))

std[7]<-sd(x8)/sqrt(length(x8))

std[8]<-sd(x9)/sqrt(length(x9))

std[9]<-sd(x10)/sqrt(length(x10))

std[10]<-sd(x11)/sqrt(length(x11))

std[11]<-sd(x12)/sqrt(length(x12))

std[12]<-sd(x13)/sqrt(length(x13))

std[13]<-sd(x14)/sqrt(length(x14))

std[14]<-sd(x15)/sqrt(length(x15))

std[15]<-sd(x16)/sqrt(length(x16))

std[16]<-sd(x17)/sqrt(length(x17))

std[17]<-sd(x18)/sqrt(length(x18))

std[18]<-sd(x19)/sqrt(length(x19))

std[19]<-sd(x20)/sqrt(length(x20))

std[20]<-sd(x21)/sqrt(length(x21))

std[21]<-sd(x22)/sqrt(length(x22))

std[22]<-sd(x23)/sqrt(length(x23))

std[23]<-sd(x24)/sqrt(length(x24))

std[24]<-sd(x25)/sqrt(length(x25))

std[25]<-sd(x26)/sqrt(length(x26))

std[26]<-sd(x27)/sqrt(length(x27))

std[27]<-sd(x28)/sqrt(length(x28))

std[28]<-sd(x29)/sqrt(length(x29))

std[29]<-sd(x30)/sqrt(length(x30))

std[30]<-sd(x31)/sqrt(length(x31))

std[31]<-sd(x32)/sqrt(length(x32))

std[32]<-sd(x33)/sqrt(length(x33))

std[33]<-sd(x34)/sqrt(length(x34))

std[34]<-sd(x35)/sqrt(length(x35))

std[35]<-sd(x36)/sqrt(length(x36))

std[36]<-sd(x37)/sqrt(length(x37))

std[37]<-sd(x38)/sqrt(length(x38))

std[38]<-sd(x39)/sqrt(length(x39))

std[39]<-sd(x40)/sqrt(length(x40))

barplot(std,ylim=c(0,0.004),ylab="standard error",xlab="positioning on the die",main="6 sided die")

#similar way for 3,4,5 sided die.

#6

#What happens to the standard errors for the

#long-term probability estimates as n increases? Why does

#this happen?

m<-10^3

x1<-c(1:m)

for(i in 1:m)

{

print(i)

cc<-estimate\_monopoly(1000, stimulate\_monopoly(n=1000,4))

qw<-data.frame("position"=c(0:39),"probablity"=cc)

x2<-qw[qw$position==11,]

x1[i]<-x2$probablity

}

stdError <- sd(x1)/sqrt(length(x1))

#0.0001392955