R Textbook Companion for Introduction to Probability by Dimitri P. Bertsekas and John N. Tsitsiklis[1]

Created by
Midhun C Kachappilly
B.Tech.
Electrical Engineering
Government Engineering College Barton Hill
Cross-Checked by
R TBC Team

June 11, 2020

[1]Funded by a grant from the National Mission on Education through ICT - http://spoken-tutorial.org/NMEICT-Intro. This Textbook Companion and R codes written in it can be downloaded from the "Textbook Companion Project" section at the website - https://r.fossee.in.

Book Description

Title: Introduction to Probability

Author: Dimitri P. Bertsekas and John N. Tsitsiklis

Publisher: Athena Scientific

Edition: 2

Year: 2008

ISBN: 978-1-886529-23-6

R numbering policy used in this document and the relation to the above book.

Exa Example (Solved example)

Eqn Equation (Particular equation of the above book)

For example, Exa 3.51 means solved example 3.51 of this book. Sec 2.3 means an R code whose theory is explained in Section 2.3 of the book.

Contents

List of R Codes

Chapter 1

sample space and probability

R code Exa 1.2 Discrete Models

```
# EX1_2
#page 10
number_flips <- 100
# created coin object with head and tail
coin <- c("heads","tails")
#simulating the flip of the object coin
flips <- sample(coin, size=number_flips, replace=
   TRUE)
#counting the number of heads and tails in the flips
freq <- table(flips)
#typing the frequency of heads and tails
freq
#probability of getting head if we specify that head
    and tail is equally likely
dbinom(1, size=1, prob=0.5)
#total probability of head and tail
dbinom(1, size=1, prob=0.5)+
dbinom(1, size=1, prob=0.5)
```

R code Exa 1.3 Probabilistic Models

```
#EX_1_3
#page 11
Dice<- seq(1:4)
d<-0
c<-0
a<- numeric(2)#creating an array
Sample_Space<-expand.grid(Dice,Dice)#creating the sample space
Sum_Matrix <- matrix(nrow=4,ncol=4) #creating a sample matrix
#storing the sum of sample space of rolling 2 dice
for (i in 1:4)
 {
    for (j in 1:4)
      {
        a[1]<-i
        a[2]<-j
        Sum_Matrix[i,j]=sum(a)#stores the sum of the sample in the matrix b
        if((Sum_Matrix[i,j]%%2)== 0)#to check whether the sum is even
        {
          print(Sum_Matrix[i,j])
          d <-d+1#count the even sums
        }else
          {
            c<-c+1#count odd sums
        }
    }
}
total_sample_space<- nrow(Sample_Space)#finding the number of sample space
n<-total_sample_space#printing the sample space count
Even_Sum<- d/n
Even_Sum #printing the probability of getting even
```

```
    numbers
Odd_Sum<-c/n
Odd_Sum #printing the probability of getting odd
    numbers
```

R code Exa 1.6 Conditional probability

```
#EX1_6.R
#page 19
coins<- c(”H”,”T”,”H”,”T”,”H”,”T”)
for(i in 1:8){
flips <- sample(coins, size=3, replace=FALSE)
print(flips)
}
A<-dbinom(2, size=3, prob=0.5)
dbinom(3, size=3, prob=0.5)
#probability of getting more heads than tails
p<-dbinom(3, size=3, prob=0.5)+
dbinom(2, size=3, prob=0.5)
print(p)
#probability of first toss is a head
q<- 1-p
#probability of (A and B)
s<- 3/8
#conditional probabilty p(A|B)
r<- s/q
print(r)
```

R code Exa 1.8 Conditional probability

```
#EX_1_8
#page 19
p1<- 2/3
```

```
#probabilty of team n succes
#p(SS)+P(FS)=1/2
p2<- 1/2
#probabilty atleast one got selected =p(SS)+p(SF)+p(FS)
p3<-3/4
#Difference of p3 and p1= p(FS)
FS<-sum(p3,-p1)
FS
SF<-sum(p3,-p2)
SF
p4<-sum(p1,p2)
SS<-sum(p4,-p3)
SS
#the probability that it was designed by team N
#conditional probabilty
#P({FS}|{SF,FS})
PN<- FS/sum(FS,SF)
PN#prints the probability that it is designed by N
```

R code Exa 1.9 Conditional probability

```
#EX_1_9
#page 21
Rate<-100000# specifying the number of trials
Aircraft<- sample(c("no","yes"),size=Rate,replace=TRUE,prob=c(0.95,0.05)) #created the sample space of presence of aircraft
Radar<- rep(NA,Rate)#creating the radar sample array
#creating the sample space in which radar detects the aircraft eventhough there is not an aircraft
Radar[Aircraft=="no"]<-sample(c("detects","notdetect"),size=sum(Aircraft=="no"),replace=TRUE,prob=c(0.10,0.90))
#Creating the sample space in which radar detects
```

```r
aircraft with the presence of aircraft
Radar[Aircraft=="yes"]<-sample(c("detects","notdetect"),size=sum(Aircraft=="yes"),replace=TRUE,prob=c(0.99,0.01))
p1<-mean(Radar[Aircraft=="yes"]=="detects")#probability radar detects aircraft given aircraft present
p2<-mean(Radar[Aircraft=="yes"]=="notdetect")#probability radar not detects given aircraft is present
p3<- mean(Radar[Aircraft=="no"]=="detects")#probability radar detects given aircraft is not present
p4<-mean(Radar[Aircraft=="no"]=="notdetect")#probabilityradar not detects given aircraft is not present
p5<-mean(Aircraft=="yes")#probability of presence of aircraft
p6<-mean(Aircraft=="no")#probability aircraft is not present
#A = {an aircraft is present}, B = {the radar registers an aircraft presence}
#A! = {an aircraft is not present}, B! = {the radar does not register an aircraft presence}.
probability <- matrix(c(p1,p2,p3,p4,p5,p6), nrow=6, byrow=TRUE, dimnames =list(c("P(B|A)","P(B!|A)","P(B|A!)","P(B!|A!)","P(A)","P(A!)"),c("probability")))
probability
Con_prob<- function(a,b)
{
  return(a*b)

}
#P(false alarm) = P(A!B)=P(A!)P(B|A!)
p7<- Con_prob(p6,p3)#P(A!B)
p7
#P(missed detection) = P(AB!)=P(A)P(B!|A)
```

```
p8<- Con_prob(p5,p2)# P(AB!)
p8
```

R code Exa 1.10 Conditional probability

```
#Example 1.10.
#page 10
#simulation of a deck of cards
deck <- c(rep("Diamonds",13), rep("Clubs",13), rep("Hearts",13), rep("Spades",13))
deck
#simulation of picking 3 cards from a deck of 52 cards
picks <- sample(deck, size=3, replace= FALSE)
picks
#counting number of elements in a pick
count<-table(picks)
#initializing a list "alpha" to store the probability of not getting a heart for each 3 picks
alpha<-numeric(3)
#function to calculate probability
eventProbability <- function(cardnumber,decknumber)
{
  notHeartprobability <- (cardnumber/decknumber)
  return (notHeartprobability)
}
#loop to store probability in list alpha
for (i in 1:3){
# number of cards in deck before picking a card
deckNumber <- 52
# number of cards other than heart before picking a card
notHeartNumber <- 39
#after picking cards without replacement
```

```
cardsDrawn <- (i-1)
deckNumber <- deckNumber- cardsDrawn
print(deckNumber)
HeartDrawn <- (i-1)
notHeartNumber <- notHeartNumber - HeartDrawn
print(notHeartNumber)
#finding probability for each picking
heartprobability <- eventProbability (notHeartNumber
   ,deckNumber)
#storing the probability to list
alpha [i]<- heartprobability
print(alpha)
print (heartprobability)
}

print(alpha)
#finding the total probabilty of not getting a heart
    when piking 3 cards from 52 cards
probabilityNotHeart <- (alpha[1]*alpha[2]*alpha[3])
print(probabilityNotHeart)
print(heartprobability)
```

R code Exa 1.11 Total Probabilty Theorem and Bayes Rule

```
#EX_1_11
#page25
#initialize variables
decreaseInGroups<- 4
studentDecrease <- 1
studentSlotInGroups <-12
studentSlot <- 15
#initialize the array of probability
alpha<-numeric(3)
#function to calculate probability
probability <- function(students,studSlot)
```

```
{prob <- (students/studSlot)
return (prob)
}
for(i in 1:3){
probs <- probability(studentSlotInGroups,studentSlot)
print(probs)
alpha[i]<- probs
print(alpha)
studentSlotInGroups<- studentSlotInGroups - decreaseInGroups
print (studentSlotInGroups)
studentSlot <- studentSlot - studentDecrease
print(studentSlot)
}#loop to calculate probability
#calculating probability of having gaduate student in each group
totalProbability <- alpha[1]*alpha[2]*alpha[3]
print (totalProbability)
```

R code Exa 1.12 Total Probabilty Theorem and Bayes Rule

```
#EX_1_12
#page26
mat<-numeric(3)
win <- matrix(c(0.3, 0.4, 0.5),nrow=3,byrow=TRUE,dimnames=list(c("p(B|A1)","P(B|A2)","P(B|A3)"),c("probability"))) #the probability of winning given playing with i'th  opponent
win
playing <- matrix(c(0.5, 0.25, 0.25),nrow=3,byrow=TRUE,dimnames=list(c("p(A1)","P(A2)","P(A3)"),c("probability"))) #the probability of playing with i'th opponent
playing
```

```
for(i in 1:3){mat[i]<-playing[i]*win[i]} #to do the total multiplication theorem
mat#printing the product
totalprobability<-sum(mat) #total probability of winning ,P(B)
totalprobability#prints the total probabilty of winning p(B)
```

R code Exa 1.18 Independence

```
#EX_1_18
#page34
number_flips <- 100
# created coin object with head and tail
coin <- c("heads","tails")
#simulating the flip of the object coin
flips <- sample(coin, size=number_flips, replace=TRUE)
#probability of getting head if we specify that head and tail is equally likely
dbinom(1, size=1, prob=0.5)
#total probability of head and tail
dbinom(1, size=1, prob=0.5)#p(H1|D)
dbinom(1, size=1, prob=0.5)#p(H2|D)
```

R code Exa 1.28 Counting

```
#EX_1_28
#page45
install.packages("prob")
library(prob)
permsn(4,2)#permutation of 4 letters out of which 2 is taken
```

```
combn(4,2)#combination of 4 letters out of which 2
    is picked up
ncol(combn(4,2))#number of combinations when 4
    letters from which 2 is picked out
```

R code Exa 1.29 Counting

```
#EX_1_29
#page46
TATTOO <- list("T","A","T","T","O","O")# listing the
     letters of Tattoo
L<-length(TATTOO)
M<-0
N<-0
R<-0
for(i in 1:L)#loop to count the number of same type
    of letters in tattoo
{
  if(as.character(TATTOO[i])=="T")
  {
    M<-(M+1)
  }

  else if(as.character(TATTOO[i])=="A")
  {
  N<- (N+1)
  }
  else if(as.character(TATTOO[i])=="O")
  {
    R<- (R+1)
  }
}
#the counters in the loop will count the number of
    same letters
repetition<- matrix(c(M,N,R),nrow=3,byrow=T,dimnames
```

```
    = list(c("T","A","O"),c("repetition")))
repetition#matrix give the number of repeated letters
Per<- function(p,q,s,t)
{
  X<-prod(factorial(p))/(factorial(q)*factorial(s)*factorial(t))
  return(X)

}#function to calculate the permutation
p<-Per(L,M,N,R)
p
```

R code Exa 1.30 Counting

```
#EX_1_30
#page47
install.packages("prob")
library(prob)
total_sample_space<- prod(factorial(16))/(factorial(4)*factorial(4)*factorial(4)*factorial(4))
graduate<-letters[1:4]#creating sample spaceof 4 graduate
permsn(graduate,4)# sample space combination of 4 graduate in 4 groups
f1<-ncol(permsn(graduate,4))# number of combination of 4 graduate in 4 groups
#Take the remaining 12 undergraduate students and distribute them to the four groups (3 students in each).
c<-prod(factorial(12))/(factorial(3)*factorial(3)*factorial(3)*factorial(3))
c
f2<-f1*c#total possibility of dividing the  4 graduate and 12 undergraduate students is
```

```
   randomly divided into four groups of 4.
f2
p<-f2/total_sample_space
p#total probability of the sample space
```

Chapter 2

Discrete Random Variable

R code Exa 2.1 functions and random variables

```
#EX_2_1
#page 10
x<-numeric(9)#creating the sample array
y<-numeric(4)
X<-c(-4:4)#creating the sample space of x and y
y<-c(1:4)
#function to caculate sample space of p(x)
px<-function(x)
{
  if(-4<=x&&x<=4)
  {
    return(1/9)
  }else{
  return(0)
  }
}
#creating the probability function of y
py<-function(y)
{
  if(1<=y&&y<=4)
  {
```

```
    return(2/9)
  }else if(y==0)
  {
    return(1/9)
  }else{
    return(0)
  }
}
#printing the sample space of p(x)
for(i in 1:9)
{
  print(px(i-5))
}
#printing the sample space of p(y)
for(i in 1:5)
{
  print(py(i-1))

}
```

R code Exa 2.2 Expectation Mean and Variance

```
#EX_2_2
#page12
install.packages("prob")
library(prob)
fx<-numeric(3)#initializing the probability mass
    function
x<-numeric(3)#initializing the x value
M<-numeric(3)#initializing the array to have the
    loop value to calculate mean
V<- numeric(3)#initializing the array to have the
    loop value of standard deviation
fx <-c(((1/4)^2),(2*1/4*3/4),(3/4)^2)#initializing
    the array of PMF
```

```
x<- c(0,1,2)#initializing the x variable
coin<- c("H","T")# initializing object coin
iidspace(coin,ntrials=2,probs=(c(0.75,0.25)))#sample space of tossing 2 coin with probability .75 of getting head
#for loop to calculate the product of PMF and x
for(i in 1:3)
{
  M[i]<-prod(fx[i],x[i])
}
mean<-sum(M)#calculated the mean
mean# print the mean
#loop to calculate the variance
for(i in 1:3)
{
  V[i]<-(x[i]-mean)^2
}
variance<-V#calcualted the variance
variance# print the variance
standard_deviation<-sqrt(variance)#standard deviation is the square root of variance
standard_deviation# print the standard deviation
```

R code Exa 2.3 Expectation Mean and Variance

```
#EX_2_3
#page 14
M<- numeric(9)#created the sample array
x<-numeric(9)
z<- numeric(9)
Z<-numeric(5)
V<- numeric(5)
PMFZ<-numeric(5)
x<-c(-4:4)
#function to create the sample space of PX(x)
```

```
PMF<-function(x)
{
  if(1<=x&&x<=9)
  {
    return(1/9)
  }else{
    return(0)
  }
}
for(i in 1:9)
{
  M[i]<-x[i]*PMF(i)
}
Ex<-sum(M)
Ex#the expected valur of Px(x)
#loop to calculate the Z
for(i in 1:9)
{
  z[i]<-(x[i]-mean)^2
}
Z<-z[5:9]
Z#calculating the sample space of Z
PMFz<-function(z)
{
  if((z==1||z==4||z==9||z==16)&&z!=0)
  {
    return(2/9)
  }else if(z==0){
    return(1/9)
  }else{
    return(0)
  }
}
#loop to print the PMF(z)
for(i in 1:5)
{
  print("PMF(z)")
 print(PMFz(Z[i]))
```

```
}
#loop to calculate the multiplication of each element
for(i in 2:5)
{
  V[i]<- Z[i]*PMFz(1)
}
V
variance<-sum(V)
variance#calcualted total variance
```

R code Exa 2.4 Expectation Mean and Variance

```
#EX_2_4
#page 17
#let p=0.15
#(1-p)=0.85
X<-numeric(2)#initializing the array of size 2 to store the varaible
px<-numeric(2)#initializing the array PX to store the probability of occuring the event
px1<-numeric(2)#initializing the array to store mean of X
px2<-numeric(2)#initializing the array to store mean of X^2
X<-c(1,0)#the events
PX<-c(0.15,0.85)#probability of events
for(i in 1:2)#loop to clculate the product of events and the probability of occuring the events
{
  px1[i]<-X[i]*PX[i]
}
px1<-sum(px1)#calcualted the mean
for(i in 1:2)#;loop to calculate the product of square of event and the probabilty of occuring
```

```
the events
{
  px2[i]<-X[i]^2*PX[i]
}
px2<-sum(px2)#calcualted the mean of square of the events
variance<-px2-px1^2#cacualted the variance
variance
prod(PX)
```

R code Exa 2.5 Expectation Mean and Variance

```
#EX_2_5
#page 18
x<-numeric(6)#initialize the array of size 6 to store the events
px<-numeric(6)#initialize the array  to store the probaility of events
ex<-numeric(6)#initialize the array to store the product of probabilty and the events
ex1<-numeric(6)# initialize the array to store the product between the probaility and the events
x<-c(1:6)#the events
px<-1/6#the probabilty of events
for(i in 1:6)#loop to calcualte the product between the probabilty of events and between the square of events
{
 ex[i]<-prod(x[i],px)
 ex1[i]<-prod(x[i]^2,px)
}
EX<-sum(ex)#calcualted the sum of  E(X)
EX1<-sum(ex1)#calcualted the sum of E(X^2)
EX
EX1
```

```
variance<-EX1-EX^2#calcualted the variance
variance# print the calcualted variance
```

R code Exa 2.7 Expectation Mean and Variance

```
#EX_2_7
#page 20
#Quiz problem
x1<-numeric(3)#creating the sample list
x2<-numeric(3)
p1<-numeric(3)
p2<-numeric(3)
ex1<-numeric(3)
ex2<-numeric(3)
x1<-c(0,100,300)#creating the sample space of the x1 and x2
x2<-c(0,200,300)
p1<-c(0.2,0.8*0.5,0.5*0.8)#creating the sample space of the probabilty of both x1 and x2
p2<-c(0.5,0.5*0.2,0.5*0.8)
#loop to calculate the multiplication of both probabilities
for(i in 1:3)
{
  ex1[i]<-prod(x1[i],p1[i])
  ex2[i]<-prod(x2[i],p2[i])
}
sum(ex1)#the expected values of the both x1 and x2
sum(ex2)
```

R code Exa 2.8 Expectation Mean and Variance

```
#EX_2_8
```

```
#page 21
# Average Speed Versus Average Time
t<-numeric(2)#initialize the array to store the time
p<-numeric(2)#initialize the array to store the
    probability
et<-numeric(2)#array to store the product
t<-c(2/5,2/30)#timre array
p<-c(0.6,0.4)#probability array
for(i in 1:2)#loop to calculate the product of
    probability and the time
{
  et[i]<-prod(t[i],p[i])
}
sum(et)#calcualted the mean
```

R code Exa 2.9 Joint PMFs of Multiple Random Variable

```
#EX_2_9
#page 25
# Mean of the Binomial
x<-300#each 300 student get 1 PMF
p<-1/3#probaility of each getting A
e<-prod(x,p)#the mean E[X]=Sum((i=1to 300)*1/3)
e#printing the mean
```

R code Exa 2.11 Conditioning

```
#EX_2_11
#page 29
sum<-0
mat<-matrix(c(16/48,12/48,9/48,0,4/48,6/48,0,0,1/48)
    , nrow=3,ncol=3,byrow=T,dimnames= list(c(x=0:2),c
    (y=0:2)))
```

```
mat#the matrix of the joint PMF
x[2]
mat[1,1]

for(i in 2:3)
{
  for(j in 2:3)
  {
    sum<-sum+mat[i,j]
  }
}
sum# probabilty of atleast one wrong
```

R code Exa 2.13 conditioning

```
#EX_2_13
#page 31
#x travel time of given message
#y the length of the given message
py<-function(y)
{
  if(y==100)
  {
    return(5/6)
  }else if(y==10^4){
    return(1/6)
  }
}#function to calculate the PMF (y)
pxy<<-function(x)
{
  if(x==0.01)
  {
    return(1/2)
  }else if(x==0.1){
    return(1/3)
```

```
  }else if(x==1){
    return(1/6)
  }
}#function to calculate the PMF(x,10^2)
PXY<-function(x)
{
 if(x==1)
 {
   return(1/2)
 }else if(x==10){
   return(1/3)
 }else if(x==100){
   return(1/6)
 }
}#function to calcualte the PMF(x,10^4)
#using the probability formula calculated each probability
px0.01<-py(100)*pxy(0.01)
px0.01
px0.1<-py(100)*pxy(0.1)
px0.1
px1<-(py(100)*pxy(1))+py(10^4)*PXY(1)
px1
```

R code Exa 2.14 Conditioning

```
#EX_2_14
#page 34
#E[X]iseasily calculated using the total expectation theorem as
p<-numeric(3)
t<-numeric(3)
ex<-numeric(3)
p<-c(0.5,0.3,0.2)
t<-c(0.05,0.1,0.3)
```

```
for(i in 1:3)
{
  ex[i]<-prod(p[i],t[i])
}
sum(ex)#E(x) is simpley calculated using total expectation theorem
```

R code Exa 2.16 Independence

```
#EX_2_16
#page 34
install.packages("prob")
library(prob)
PXx<-numeric(3)
PXAx<-numeric(3)
toss<-matrix(nrow=4,ncol=4)
p<-numeric(4)
mat<-matrix(nrow=3,ncol=3)
coin<-c("H","T")
toss<-expand.grid(coin,coin)
toss#gives the sample space of all combination of 2 independent toss
table(toss)
nrow(toss)#gives number of rows
ncol(toss)
probspace(toss)#gives the probability of each sample in sample space
mat<-noorder(probspace(toss))#table the repeating probabilty

mat[3]#takes the probabilty
#Let X be the number of heads and
#function to calculate  the PMF of x
pxx<-function(x)
{
```

```
  if(x==0)
  {
    return(1/4)
  }else if(x==1){
    return(0)
  }else if(x==2){
    return(1/2)
  }
}
#function to calculate the conditional PMF
pxax<-function(x)
{
  if(x==0)
  {
    return(1/2)
  }else if(x==1){
    return(0)
  }else if(x==2){
    return(1/2)
  }
}
#loop to print the PMF(x)
for(i in 1:3)
{
  PXx[i]<-pxx(i-1)
  PXAx[i]<-pxax(i-1)
}
PXx#print the PMF(x)
PXAx#print the conditonal PMF(X|A)
```

R code Exa 2.18 Independence

```
#EX_2_18
#page 40
n<-100000
```

```
binomial<-numeric(2)
#Xi be a random variable that encodes the response of the i th person:
Xi<-c(1,0)#1 if i th person approves C's performnce
#0 if the ith person dissapproves C's performance
binomial<-rbinom(Xi,n,0.5)#creating a random variable  of the approval of c's performance
binomial#printing the random variable
p<-1/2#the common mean of appproval
q<-1-p#the common mean of dissapproval
sn<-binomial[1]/n#sn is the mean from the sample random variable
sn
Esn<-p#printing the expectation of the mean of sample space is the common mean
varsn<-prod(p,q)/n
varsn#variance of the mean
```

Chapter 3

General Random Variable

R code Exa 3.2 Continuous Random Variables And PDFs

```
#EX_3_2
#page 5
#Piecewise Constant PDF
Fx<-numeric(11)
f <- function(c1)c1#representing the function of constant variable to integrate
f2<-function(c2)c2
cum<-integrate(f,15,20)
p_sunnyday<-cum$value/17.5#to calculate the probabilty of sunny day
p_sunnyday
cum2<-integrate(f2,20,25)
p_rainyday<-(cum2$value/22.5)#to calcualte the probability of rainy day
p_rainyday
c1<-(2/3)/p_sunnyday#calcuLting the c1
c2<-(1/3)/p_rainyday#calculating the c2
fx<-c(c1,c2)#sample space of fx(x)
c1
c2
fx
```

R code Exa 3.3 Continuous Random Variables And PDFs

```
#EX_3_3
#page 6
#function to print the sample space of fX(x)
fx<-function(x)
  {
    if(0<x&&x<=1)
      {
        return(1/(sqrt(x)*2))
      }
    else
      {
        return(0)
      }
}
FX<-c(fx(0),fx(1))
FX #PDF of random variable x
int<-integrate(fx,0,1)
int$value#PDF of fX(x)
```

R code Exa 3.4 Continuous Random Variables And PDFs

```
#EX_3_4
#page 8
#function to calculate the gx
gx<-function(x)
{
  if(x<=1/3)
  {
    return(1)
```

```
  }else if(x>1/3){
    return(2)
  }
}
#function to calcualte the PMF Py
pY<-function(gx)
{
  if(gx==1)
  {
    return(1/3)
  }else if(gx==2){
    return(2/3)
  }
}
#to calcualte the E(Y)
EY<-sum(pY(1)*gx(1/3),pY(2)*gx(2/3))
EY
```

R code Exa 3.5 Continuous Random Variables And PDFs

```
#EX_3_5
#page 10
lamda<-1/10
px<-function(a,lamda)
{
  return(exp(1)^(-lamda*a))
}
#function to calculate the probability
PX<-function(x)
{
  if(1/4<=x||x<=3/4)
    {
      return(px(1/4,1/10)-px(3/4,1/10))
    }
}
```

```
PX(1/4)#probabilty of meteorite lands between 6 am and 6 pm on the first day
```

R code Exa 3.7 cumulative distribution Function

```
#EX_3_7
#page 15
px<-numeric(10)
#we compute the FX(k)first and then the PMF
#functon to calculate the FX(k)
fx<-function(k)
{
  return((k/10)^3)
}
#function to calculate the FX(k-1)
fx1<-function(k)
{
  return(((k-1)/10)^3)
}
#to print the PMF
for(i in 1:10)
{
  px[i]<-fx(i)-fx1(i)
}
px#PMF calculated
```

R code Exa 3.8 Normal Random Variable

```
#EX_3_8
#page 19
#Using the Normal Table
# Its CDF is denoted by phi,
pi<-3.14
```

```
#function to calculate the CDF normal random variable
f<-function(t)
{
  return((1/sqrt(2*pi))*exp(1)^(-(t^2)/2))
}
#to calculate the CDF of Y<=0.5
f_0.5<-integrate(f,-Inf,0.5)
f_0.5$val
#to calculate the CDF of Y<=-0.5
f_negative_0.5<-(1-f_0.5$val)
f_negative_0.5
sd<-20#standard deviation
mean<-60#mean
y<-(80-mean)/20#calcualting the Y
y#Y is 1
#calculate the CDF of Y<=80-60/20 is phi(1)
f_1<-integrate(f,-Inf,1)
f_1$val
#to calculate the CDF of Y>=80-60/20
p_x_greater_80<-(1-f_1$val)
p_x_greater_80
```

R code Exa 3.9 Normal Random Variable

```
#EX_3_9
binary_message<-c(-1,1)#the message send may be -1,1
mean<-0#mean and standard deviation is given
sd<-1
pi<-3.14
variance<-sd^2
#function to calculate the normal table
f<-function(y)
{
  return((1/sqrt(2*pi))*e^((-y^2)/2))
```

```
}
#to calculate the CDF of sending sending message is −1 is normal table phi(1)
f1<-integrate(f,0,1)
f1$val
#probabilty of error
p_N_greater_1<-1-f1$val
p_N_greater_1
```

R code Exa 3.11 conditioning on an event

```
#EX_3_11
#page 24
# Mean and Variance of a Piecewise Constant PDF
x<-readline(prompt="x: ")#enter the random variable x in the console
#this enters the constant PDF of x
if(0<=x&&x<=1)
  {
  pA1<-1/3
  print("pA1:")
  return(pA1)
}else if(1<x&&x<=2){
  pA2<-2/3
  print("pA2:")
  return(pA2)
}else {
  return(0)
  }
# the mean of a uniform random variable on an interval [a,b] is ( a+b)/2 and its second moment is (a2 +ab+b2)/3.
ex<-function(a,b)#function to return the mean
{
  return(sum(a,b)/2)
```

```
}
ex2<-function(a,b)#function to return the variance
{
  return(sum(a^2,prod(a,b),b^2)/3)
}
ex(0,1)#mean when x in between 0&1
ex2(0,1)#variance when x in 0&1
ex(1,2)#mean of x in 1&2
ex2(1,2)#variance of x in 1&2
```

R code Exa 3.12 conditioning on an event

```
#EX_3_12
#page 25
#metro train problem
A1<-numeric(5)
A2<-numeric(15)
A1<-sample(c(1:5),replace = FALSE)
A2<-sample(c(1:15),replace = FALSE)
pA1<-1/4
fyA1<-1/length(A1)
fyA2<-1/length(A2)
for(i in 1:15)
{
  if(A2[i]<5)
  {
    print(sum(prod(pA1,fyA1),prod((1-pA1),fyA2)))
  }else{
    print(prod((1-pA1),fyA2))
  }
}
```

R code Exa 3.22 Derived Distributions

```
#EX_3_22
#page 40
x<-numeric(30)
gx<-numeric(30)
x<-runif(30,30,60)
x#to print the uniform distribution of time between
   30,60
#to print the g(x),PDF,CDF of X
for(i in 1:30)
  {
  print("x:")
  print(x[i])
  gx[i]<-180/x[i]
  print("gx")
  print(gx[i])
if(30<=x[i]||x[i]<=60){
      print("fx")
      print(1/30)
      print("FX")
      print((x[i]-30)/30)
    }else if(60<=x[i]){
      print("FX")
      print(1)
    }else if(30>=x[i]){
      print("FX")
      print(0)
    }else{
      print("fx")
      print(0)
    }
}
```

Chapter 4

Further Topics on Random Variables and Expectations

R code Exa 4.1 Transforms

```
#EX_4_1
#page 2
#Transform
x<-c(2,3,5)#creating the uniform random variable of
  x
px<-c(1/2,1/6,1/3)#creating the pdf function of x
Ms<-sum(1/2*exp(1)^2,1/6*exp(1)^3,1/3*exp(1)^5)#
  calculating the transform
Ms
par(mfrow=c(2,2))#creating the space for the plots
  to be plotted
curve((1/2*exp(1)^(2*x)),-10,10,col="red")#curve of
  the Ms function of x=2
curve(1/3*exp(1)^(3*x),-10,10,col="violet")#curve of
   the Ms function of x=3
curve(1/5*exp(1)^(3*x),-10,10,col="black")#curve of
  the Ms function of x=5
plot(x,px,type="h",col="red")#plot the x vs px graph
```

R code Exa 4.4 Transforms

```
#EX_4_4
#page 4
#function to calculate the exponential randm
   variable of x
exponential_transform<-function(l,s){
  return(l/l-s)
}
#function to calculate the exponential transform of
   y
y<-function(a,b,l,s)
{
  (exp(1)^b*s)*l/l-a*s
}
print("l/l-s")
exponential_transform(1,0)#printing the both
   transform by giving certain values
y(2,3,1,1)
```

R code Exa 4.5 Transforms

```
#EX_4_5
#page 5
x<-numeric(3)
px<-numeric(3)
derrivative<-numeric(3)
x<-c(2,3,5)#creating the uniform random variable of
   x
px<-c(1/2,1/6,1/3)#creating the pdf function of x
Ms<-sum(1/2*exp(1)^2,1/6*exp(1)^3,1/3*exp(1)^5)#
   calculating the transform
```

```
ex<-expression(px*x*exp(1)^x*s)
derrivative<-D(ex,"s")
derrivative
mx<-expression((1/2*exp(1)^(2*s))+(1/6*exp(1)^(3*s))+(1/3*exp(1)^(5*s)))#to print the mx
ex<-D(mx,"s")#it gives the derrivative of mx
ex
ex2<-D(ex,"s")#it gives the second derrivative
ex2
```

R code Exa 4.6 Transforms

```
#EX_4_6
#page 6
x<-numeric(3)
px<-numeric(3)
derrivative<-numeric(3)
x<-c(2,3,5)#creating the uniform random variable of x
px<-c(1/2,1/6,1/3)#creating the pdf function of x
Ms<-sum(1/2*exp(1)^2,1/6*exp(1)^3,1/3*exp(1)^5)#calculating the transform
Ms
mx<-expression((1/2*exp(1)^(2*s))+(1/6*exp(1)^(3*s))+(1/3*exp(1)^(5*s)))#giving the expression
ex<-D(mx,"s")#finding the first derrivative of expression
ex
ex2<-D(ex,"s")#finding the second derrivative of the expression
ex2
#finding the values of transforms with s=1 and s=0
Mx<-((1/2*exp(1)^(2))+(1/6*exp(1)^(3))+(1/3*exp(1)^(5)))#s=1
Mx
```

```
dMx<-(1/2*2+(1/6*3)+(1/3*5))#s=0
dMx
d2Mx<-(1/2*4)+(1/6*9)+(1/3*25)#s=0
d2Mx
```

R code Exa 4.7 Transforms

```
#EX_4_7
#page 8
x<-c(-1,0,4,5)
px<-c(1/4,1/2,1/8,1/8)
Ms<-expression((1/4)*exp(1)^(-1*s)+(1/2)*exp(1)^(0*s)+(1/8)*exp(1)^(4*s)+(1/8)*exp(1)*(5*s))#expressing the transform function
s<-1#giving free variable s as 1
Ms1<-function(a,b){
  return((1/a)*exp(1)^b)
}#function for calculating the transform at s=1
Ms<-sum(Ms1(1/4,-1),Ms1(1/2,0),Ms1(1/8,4),Ms1(1/8,5))#the value of transform at free variable =1
Ms
```

R code Exa 4.9 Transforms

```
#EX_4_9
#page 9
lamda<-c(6,4)#expressing lamda
p<-1/3#initiating probabilty of selecting one teller
s<-1#expressed the free variable as 1
fx<-function(x)
{
  return((exp(1)^x)*((2/3)*6*exp(1)^(-6*x)+(1/3)*4*exp(1)^(-4*x)))
```

```
}#function for calculating the M(s)
Ms<-integrate(fx,0,Inf)  #integrate to get the M(s)
Ms$val#giving the value of M(s)
```

R code Exa 4.13 Transforms

```
#EX_4_13
#page 14
x<-numeric(3)#initializing the variables x,y,w
y<-numeric(3)
py<-numeric(3)#initializing the probability of each variable
w<-numeric(5)
py1<-numeric(3)
pw1<-numeric(5)
x<-c(1:3)#representing the sample space of each variable
y<-c(0:2)
w<-c(1:5)
px<-function(x)#function to print the probabilty of x
{
  if(1<=x&&x<=3)
    {
      return(1/3)
    }
  else
    {
      return(0)
    }
}
py<-function(y)#function to print the sample space of probability of y
{
  if(y==0){
```

```
  return(1/2)
}else if(y==1){
  return(1/3)
}else if(y==2){
  return(1/6)
}else{
  return(0)
}
}
for(i in 1:3)#loop to print the probabilty of y
{
  py1[i]<-py(i-1)
}
py1#printing the probability
pw<-function(w)#function to print the sample space of probabilty of w
{
  if(w==1)
  {
    return(px(1)*py(0))
  }else if(w==2){
    return(sum(prod(px(1),py(1)),prod(px(2),py(0))))
  }else if(w==3){
    return(sum(prod(px(1),py(2)),prod(px(2),py(1)),prod(px(3),py(0))))
  }else if(w==4){
    return(sum(prod(px(2),py(2)),prod(px(3),py(1))))
  }else if(w==5){
    return(prod(px(3),py(2)))
  }else{
    return(0)
  }
}
for(i in 1:5)#loop to print the probabilty of w
{
  pw1[i]<-pw(i)
}
pw1#printing the probabilty
```

R code Exa 4.16 conditional expectation as random variable

```
#EX_4_16
#page 23
l<-8#define length of stick as 8
vary<-function(l)
{
  return((l^2)/12)
}#function to calculate var(y)
f<-function(y)
{
  return((y^2)/(12*8))
}
varxy<-1/4*vary(l)
varxy#to print var(x|y)
integral<-integrate(f,0,l)#to calculate E(var(x|y))
Evarxy<-integral$val
Evarxy
varx<-sum(Evarxy,varxy)
varx#to final calcualtion of var(x)
```

R code Exa 4.20 conditional expectation as random variable

```
#EX_4_20
#page 24
x<-numeric(3)#initializing the variables
x1<-numeric(3)
Y<-numeric(3)
x<-c(0:2)#sample space of x
fx<-c(1/3,1/3,2/3)#sample space of fx
#function to calculate the sample space of y
```

```
y<-function(x)
{
  if(x<1){
    return(1)
  }else if(x>=1){
    return(2)
  }
}
#loop to print the y sample space
for(i in 1:3)
{
  print(y(i-1))
}
Exy<-c(1/2,3/2)#sample space of Exy
#function to calculate the probability of Exy
pExy<-function(Exy)
{
  if(Exy==1/2){
    return(1/3)
  }else if(Exy==3/2){
    return(2/3)
  }
}
MeanExy<-7/6#mean of E(x|y)
varExy<-sum(prod(pExy(1/2),((1/2-MeanExy)^2)),prod(pExy(3/2),((3/2-MeanExy)^2)))#calculating the variance of E(x|y)
varExy
varxy<-1/12
Evarxy<-1/12
varx<-sum(Evarxy,varExy)#calculating the variance of x
varx
```

R code Exa 4.21 Sum of a Random Number of Independent Random Variable

```
#EX-4_21
#page 27
gas<-runif(1000,0,1000)
p<-1/2
s<-1
MNs<-1/8*(1+exp(1)^3)#the transform of binomial
   random variable of N open gas station
MNs
Mxs<-(((exp(1)^(1000*s))-1)/(1000*s))#transform of
   amount of gas available
Mxs
Mys<-(1/8)*(1+Mxs)^3
Mys#transform assosiated with y
```

Chapter 5

Stochastic Processes

R code Exa 5.5 Bernolli process

```
#EX_5_5
#page 13
p<-0.01
q<-1-p
pz1<-numeric(4)#representing a sample list
px1<-numeric(4)
#function to calculate PX(x)
px<-function(x)
{
  if(x==0)
  {
    return((1-0.01)^100)
  }else if(x==2||x==5||x==10){
    return(prod(factorial(n),(p^x),q^(n-x))/prod(
       factorial(n-x),factorial(x)))
  }
}
#printing the PX(x)
px1<-c(px(0),px(2),px(5),px(10))
#function to calculate PZ(x)
pz<-function(x)
```

```
{
  (exp(1)^-1)/factorial(x)
}
#printing the PZ(x)
pz1<-c(pz(0),pz(2),pz(5),pz(10))
px1
pz1
```

R code Exa 5.6 The Poisson Process

```
#EX_5_6
#page 14
p<-0.0001#initializing the variables p,n,n1
n<-(log(0.999,base=exp(1)))/(log(0.9999,base=exp(1)))
n1<-(-log(0.999,base=exp(1)))/p
Ps<-1-(1-p)^n#calcualating the probability of free variable S
Ps
poisS<-1-exp(1)^-(p*n1)#calcuating the probability of free variable using the poisson approximation
poisS
```

R code Exa 5.7 The Poisson Process

```
#EX_5_7
#page 19
lamda<-0.2# initializing the variables lamda
#function to calculate the probability usung the poisson PMF
PMF<-function(lamda,T,k)
{
```

```
  return(prod((lamda*T)^k,(exp(1)^-(lamda*T)))/factorial(k))
}
PMF(0.2,1,0)#PMF of different lamda,Time,and k value is being calculated
PMF(0.2,1,1)
PMF(0.2,24,0)
PMF24<-(PMF(0.2,1,0))^24#use poisson PMF
PMF24
```

R code Exa 5.9 The Poisson Process

```
#EX_5_9
#page 19
mue1<-5
mue2<-3
#  the PMF of the total number of accidents between 8 am and 11 am?
PMF<-sum(5,(3*2))
PMF# sum of indepemdent poisson random variable with parmeters 5& 3*2
```

R code Exa 5.12 The Poisson Process

```
#EX_5_12
#page 24
n<-56
lamda<-2#callers depart with poisson process a rate of lamda
 #the waiting time Y
EY<-n/lamda
#the function to calculate the probabilty you have to wait for more than an hour
```

```
PY60<-function(y)
{
 return((lamda^n)*(y^(n-1))*(exp(1)^(-lamda*y))/
    factorial(n-1))
}
probability<-integrate(PY60, 60, Inf)#the integral
   function to calculate the probability of waiting
   more than an hour
probability$val
```

R code Exa 5.17 The Poisson Process

```
#EX_5_17
#page 30
# Random incidence in a non-Poisson arrival process
T1<-15
T2<-45
#person arrives at interarrival time of 15 minute
   with probabilty 1/4
p1<-1/4
#person arrives at interarrival time of 45 is of
   probability 3/4
p2<-3/4
#the expected value of chosen interarrival time is
T<-sum((T1*p1),(T2*p2))
T
```

Chapter 6

Markov Chains

R code Exa 6.1 the discrete time markov chains

```
#EX_6_1
#page 2
library(markovchain)#loading libraries
library(diagram)
p<-c(0.8,0.2,0.6,0.4)
probability<-matrix(p,nrow=2,ncol=2,byrow=T)
probability#probabilty matrix
plotmat(probability)
```

R code Exa 6.2 Discrete Time Markov Chains

```
#EX_6_2
#page 4
library(markovchain)#loading libraries
library(diagram)
matrix<-matrix(NA,nrow=4,ncol=4,byrow=T)#creating the sample matrix
#loop to enter the probabilty in the matrix
```

```
for(i in 1:4)
{
  for(j in 1:4)
  {
    if(i==1&&j==1)
    {
      matrix[i,j]<-1
    }else if(i==4&&j==4){
      matrix[i,j]<-1
    }else if(i==j&&i>1){
      matrix[i,j]<-0.4
    }else if(i>=2&&(j==(i-1)||j==(i+1))){
      matrix[i,j]<-0.3
    }else{
      matrix[i,j]<-0
    }
  }
}
matrix#printing the matrix
plotmat(matrix)#markov chain representation of the matrix
```

R code Exa 6.4 Steady State Behavior

```
#EX_6_4
#page 15
p<-c(0.8,0.6,0.2,0.4)
prob<-matrix(p,nrow=2,ncol=2,byrow=T)#matrix of multiples of pi
prob
pi<-c(0.75,0.25)
mat<-matrix(pi,nrow=2,ncol=1,byrow=T)#created matrix of pi giving value of pi
mat
solve(prob,mat)#generate the solution of the pi
```

R code Exa 6.11 Absorption Probabilities and Expected Time to Absorption

```
#EX_6_11
#page 26
library(markovchain)#loading libraries
library(diagram)
par(mfrow=c(2,2))#to create matrix in the plot to
   accomodate the plot
x<-c
   (1,0,0,0,0,0.2,0.3,0.4,0.1,0,0,0.2,0,0.3,0.5,0,0,0,0.3,0.7,0,0,0,
   #transition elements of first transition
x2<-c(1,0,0,0,0.2,0.3,0.4,0.1,0,0.2,0,0.8,0,0,0,1)#
   elemennts in the second transition
p<-matrix(x,nrow=5,ncol=5,byrow=T)
p#to create the element matrix
transition<-c(1:5)#nsmes of the transition matrix
row.names(p)<-transition
colnames(p)<-transition
p
plotmat(p)#to plot the markov chain of transition
   matrix
new_transition<-matrix(x2,nrow=4,ncol=4,byrow=T)#new
    transition matrix
name<-c(1,2,3,6)
row.names(new_transition)<-name
colnames(new_transition)<-name
new_transition
plotmat(new_transition)#plotting the markov chain of
    the new transition matrix
#since it is a singular matrix can't solve in r
a2<-21/31
a3<-29/31
```

R code Exa 6.13 Absorption Probabilities and Expected Time to Absorption

```
#EX_6_13
#page 30
m<-4
i<-c(2,3)
mue<-c(0.6,-0.3,0.7,-0.4)#representing the multiplication vactors with mu
mat<-matrix(mue,nrow=2,ncol=2,byrow=T)#representing to matrix
mat
b<-matrix(c(1,1),nrow=2,ncol=1,byrow=T)#the solution matrix
b
m<-solve(mat,b)#this solve the both  matrix to give the value of mu
mu<-matrix(m,nrow=2,ncol=1,byrow=T)#representing tthe valur of mu in a matrix
row.names(mu)<-c(”mu1”,”mu2”)
mu#represent the values of mu1, mu2 in the matrix ”mu”
#let m=5
transition<-c(1,0,0,0.3,0.4,0.3,0,0.3,0.4)
transition_mat<-matrix(transition,nrow=3,ncol=3,byrow=T)
plotmat(transition_mat)#markov chain representation of the transition matrix
```

R code Exa 6.14 more general markov chains

```
#EX_6_14
```

```r
#page 32
p<-c(0.8,0.2,0.6,0.4)
mat<-matrix(p,nrow=2,ncol=2,byrow=T)#probabilty matrix
mat
t<-c(0,0.6,1,0.2)
T<-matrix(t,nrow=2,ncol=2,byrow=T)#matrix to represent the t matrix
T
b<-matrix(c(1,1),nrow=2,ncol=1,byrow=T)
b
t1<-solve(T,b)#calculate the first passage time to state 1 from state2
b1<-matrix(c(0,1),nrow=2,ncol=1,byrow=T)
t2<-solve(T,b1)#calculate the mean recurrence time
t1[1,1]#mean first passage time to state1 starting from sate2
t2[1,1]#mean recurrence time to state 1
```

Chapter 7

Limit Theorems

R code Exa 7.1 Some Useful Inequalities

```
#EX_7_1
#page 3
EX<-2#expected mean
#MArkov inequality asserts that
#function to calculatemthe Msrkov Inequality
PX<-function(x)
{
  if(x>=2&&x<3)
  {
    return(1)
  }else if(x>=3&&x<4){
    return(2/3)
  }else if(x>=4){
    return(2/4)
  }
}
#function to calcualte the normal probability
px<-function(x)
{
  if(x>=2&&x<3)
  {
```

```
    return(0.5)
  }else if(x>=3&&x<4){
    return(0.25)
  }else if(x>=4){
    return(0)
  }
}

c<-c(PX(2),PX(3),PX(4),px(2),px(3),px(4))
compare<-matrix(c,nrow=3,ncol=2,byrow=T)
compare#matrix to compare both Markov Inequality and
    normal proabability
```

R code Exa 7.4 convergence in probability

```
#EX_7_4
#page 6
#polling
PMnp<-function(n,e)
{
  return(1/(4*n*e^2))
}#function to calculate the chebyshev inequality
PMnp(100,0.1)
PMnp(1000000,0.01)# calculated the chebyshev
   inequality
```

R code Exa 7.8 The central Limit Theorem

```
#EX_7_8
#page 11
n<-100#number of packages loaded
p1<-5#weights are uniformly distributed between 5and
    50
```

```
p2<-50
pi<-3.14
#mean and variance of single package
mue<-sum(p1,p2)/2
mue
var<-sum(50,-5)^2/12
var
z<-(3000-prod(100,mue))/sqrt(var*100)#normalized value of the mean and variance
z
#function to calculate the CDF normal random variable
f<-function(t)
{
  return((1/sqrt(2*pi))*exp(1)^(-(t^2)/2))
}
phi1.92<-integrate(f,-Inf,1.92)#calculate the CDF of normal random variable from the normal table
phi1.92$val
p_greater_3000<-1-phi1.92$val#the desired probability that the total weight exceeds 3000 pounds
p_greater_3000
```

R code Exa 7.9 The central Limit Theorem

```
#EX_7_9
#page 12
#processing time is independent random variable between 1and 5
mue<-3#the variance and the mean
var<-16/12
var
n<-100#the number of parts
z<-(320-(n*mue))/sqrt(var*n)
```

```
z#calculated the normalized value
f<-function(t)
{
  return((1/sqrt(2*pi))*exp(1)^(-(t^2)/2))
}#function to calculate the CDF normal random variable
phi1.73<-integrate(f,-Inf,1.73)#the desired approximation gives p(S100>320) it is t from the normal table
phi1.73$val
```

R code Exa 7.10 The central Limit Theorem

```
#EX_7_10
#page 12
n<-100#consider the case n=100 and e=0.1
e<-0.1
z<-function(e,n)
{
  return(2*e*sqrt(n))
}#function to calculate the standardized value
Z<-z(e,n)
Z#2*0.01*sqrt(n)>= 1.96
f<-function(t)
{
  return((1/sqrt(2*pi))*exp(1)^(-(t^2)/2))
}#function to calculate the CDF normal random variable
phi<-integrate(f,-Inf,Z)#the normal CDF of 2 from normal table
phi$val#phi(2)
p<-2-(2*phi$val)
p#2-2phi(2*0.01 sqrt(n))<= 0.05
```

R code Exa 7.11 The central Limit Theorem

```r
#EX_7_11
#page 15
n<-36
p<-0.5
P21<-numeric(22)
comb<-function(n,x)
{
  return(factorial(n)/factorial(n-x)/factorial(x))
}#function to calculate the combination
for(i in 1:22)
{
  P21[i]<-comb(n,(i-1))
}
P21#exact valuep(Sn<=21)
p<-sum(P21*(0.5^36))
f<-function(t)
{
  return((1/sqrt(2*pi))*exp(1)^(-(t^2)/2))
}#function to calculate the CDF normal random variable
p21<-integrate(f,-Inf,1)
p21$val#the central limit approximation
P21<-integrate(f,Inf,1.17)#Using the proposed re???nement
P21$val#which is much closer to the exact value
z1<-(19.5-18)/3
z2<-(18.5-18)/3
p1<-integrate(f,-Inf,z1)
p2<-integrate(f,-Inf,z2)
p19<-p1$val-p2$val
p19# de Moivre - Laplace formula also allows us to approximate the probability of a single value
```

```
P19<-comb(n,19)*(0.5^36)
P19#exact value P(Sn=19)
```