STAT323 Assignment 3

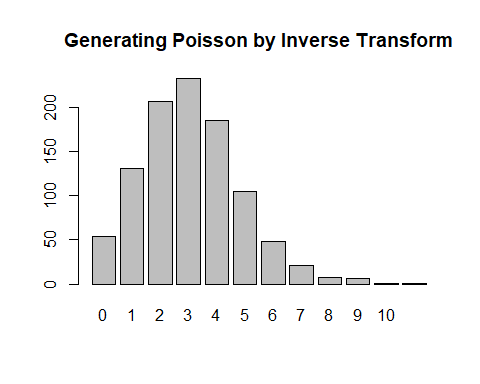
Park Jeong Jin(2014150137)

2018년 5월 14일

## 1

# (a)

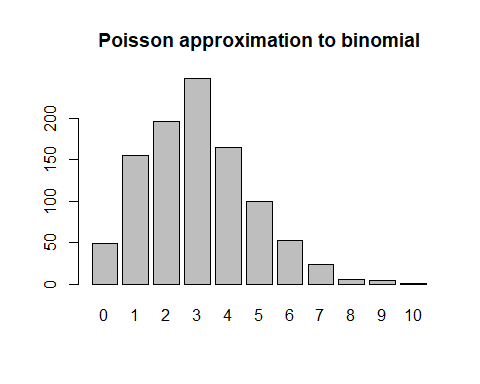
poi.sim <- function(l) {  
 X <- 0  
 px <- exp(-l)\*(l^X)/factorial(X)  
 Fx <- px  
 U <- runif(1)  
 while(Fx < U) {  
 X <- X+1  
 px <- px\*l/X  
 Fx <- Fx+px  
 }  
 return(X)  
}  
  
P <- numeric(1000)  
  
for(i in 1:1000) {  
 P[i] <- poi.sim(3)  
}  
  
barplot(table(P), main = "Generating Poisson by Inverse Transform")



# (b)

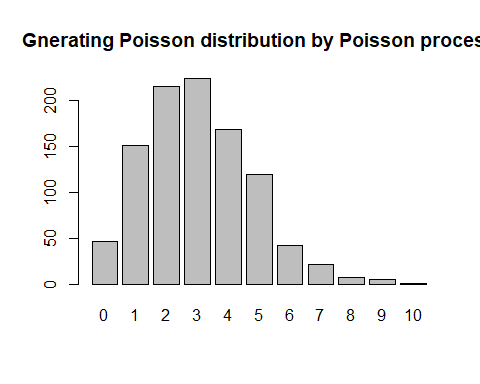
As n goes to big number, and p goes to 0.

binom.sim <- function(n, p) {  
 X <- 0; px <- (1-p)^n; Fx <- px; U <- runif(1)  
 while (Fx < U) {  
 X <- X + 1  
 px <- px\*p/(1-p)\*(n-X+1)/X  
 Fx <- Fx + px }  
 return(X) }  
  
P.1 <- numeric(1000)  
lamda <- 3  
n <- 100000  
  
for(i in 1:1000) {  
 P.1[i] <- binom.sim(n, lamda/n)  
}  
  
barplot(table(P.1), main = "Poisson approximation to binomial")



# (c)

genPoi <- function(lambda,at) {  
 x<-rexp(100,lambda)  
 if(sum(x)<at)  
 return(NA)  
 if(x[1]>at)  
 return(0)  
 else  
 return(max(which(cumsum(x)<at)))  
}  
  
P.2 <- numeric(1000)  
for(i in 1:1000) {  
 P.2[i] <- genPoi(3,1)  
}  
  
barplot(table(P.2), main = "Gnerating Poisson distribution by Poisson process")



##2 #(a)

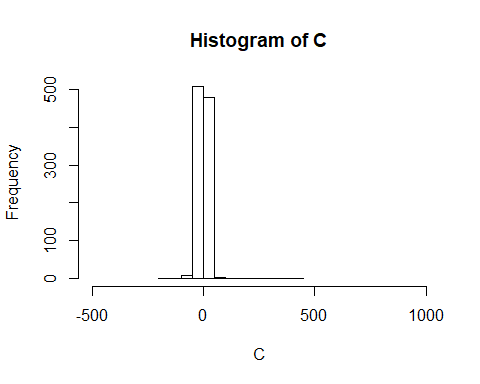
cau.sim <- function(u) {  
 X.c <- tan(pi\*(u-0.5))  
 return(X.c)  
}  
  
U <- runif(1000)  
C <- round(cau.sim(U), 7)

#(b)

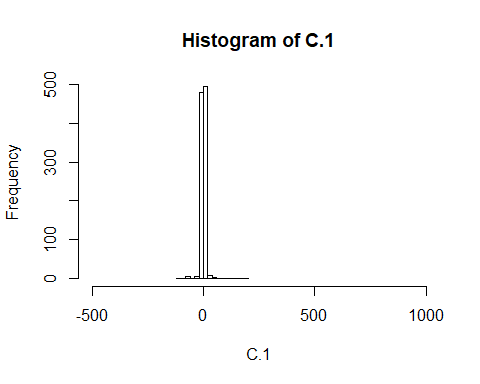
tgen2 <- function(n){  
 v <- rgamma(1,n/2,1/2)  
 tn <- rnorm(1,0,sqrt(n/v))  
 return(tn)  
}  
  
C.1 <- replicate(1000, round(tgen2(1),7))

#(c)

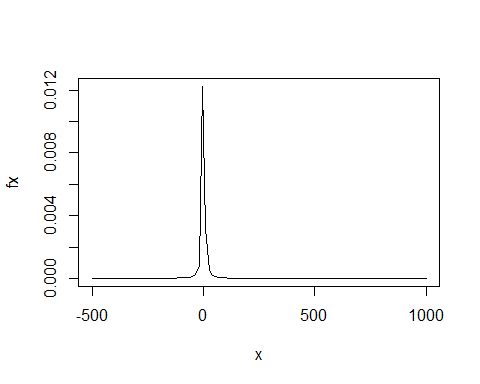
fx <- function(x) 1/(pi\*(1+x^2))  
  
  
hist(C, xlim= c(-500, 1000))



hist(C.1, xlim= c(-500, 1000))

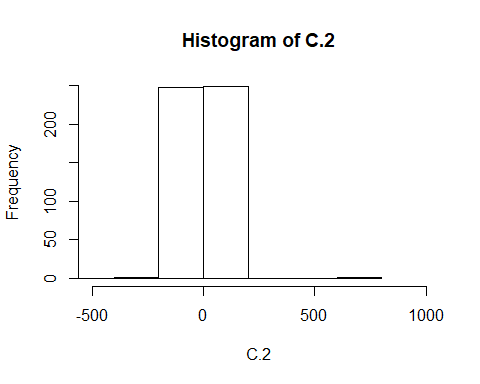


plot(fx, xlim = c(-500,1000))



#(d)

fx.1 <- function(x) 1/sqrt(2\*pi)\*exp(-x^2/2)  
  
M <-optimize(f=function(x){fx.1(x)/fx(x)},maximum=T,interval=c(-100,100))$objective  
  
  
  
rej1 <- function(fx, gx) {  
 while (TRUE) {  
 x <- runif(1); y <- cau.sim(x)  
 z <- runif(1, 0, M\*gx(y))  
 if (fx(y)/(M\*gx(y)) > z) return(y)}   
}  
  
N.x <- replicate(1000, rej1(fx.1, fx))   
N.y <- N.x[1:500]  
N.x <- N.x[501:1000]  
  
C.2 <- N.x/N.y  
hist(C.2, xlim = c(-500, 1000))



## 3

# (a)

CDF of

and

goes to 1

f1 <- function(x) 1/(pi\*sqrt(1-x^2))  
f2 <- function(x) 2\*sqrt(1-x^2)/(pi)  
  
integrate(f1, -1, 1)

## 1 with absolute error < 3e-06

integrate(f2, -1, 1)

## 1 with absolute error < 6.4e-10

mix <- function(x) {  
 u <- runif(1)  
 k <- (u < 1/3)  
 X <- k\*f1(x) + (1-k)\*f2(x)  
 return(X)  
}  
  
Mx <- runif(1000, -1, 1)  
for(i in 1:1000) {  
 Mx[i] <- mix(Mx[i])  
}

# (b) - method 1

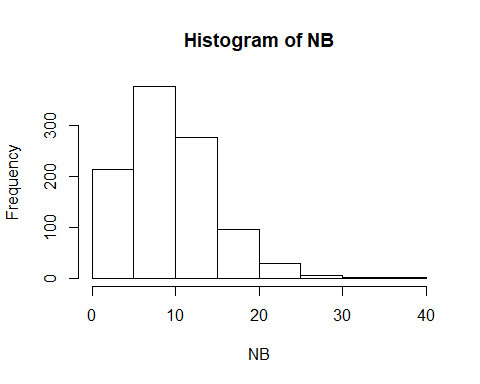
Mixture distribution of Poisson and Gamma distribution.

fx <- function(r, b) {  
 y <- rgamma(1, r, 1/b)  
 X <- rpois(1, y)  
 return(X)  
}  
  
NB <- replicate(1000, fx(5,2))

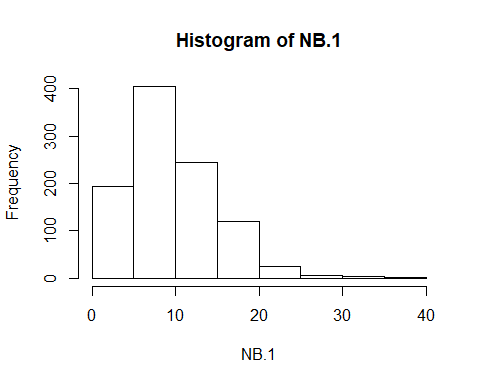
# (b) - method 2

Using the relationship between Negative Binomial and Geometric distribution.

geo.sim <- function(p) {  
 x <- 0; px <- p\*(1-p)^x; Fx <- px  
 u <- runif(1)  
 while (Fx < u) {  
 x <- x + 1; px <- px\*(1-p); Fx <- Fx + px  
 }  
 return(x)}  
  
rbi.sim <- function(r, p) {  
 x <- replicate(r,geo.sim(p))  
 y <- sum(x)  
 return(y)  
}  
  
NB.1 <- replicate(1000, rbi.sim(5,1/3))  
  
hist(NB)



hist(NB.1)



## 4

# (a)

CC <- function(n) {  
 a <- sample(letters[1:12], n, replace = T)  
 b <- unique(a)  
 c <- sum(table(b))  
 return(c)  
}  
  
MC <- replicate(10000, CC(20))  
ED <- sum(MC)/10000

#(b)

RCC <- function(N, k, n) {  
 sum <- 0  
 for(i in 1:(k-1)) {  
 sum <- sum + choose(k, i)\*(((k-i)/k)^n)\*((-1)^(i+1))  
 }  
 D <- choose(N,k)\*((k/N)^n)\*(1-sum)  
 return(D)  
}  
  
ED.1 <-0  
for(i in 1:12) {  
 ED.1 <- ED.1 + i\*RCC(12, i, 20)  
}

#(c)

f.ED <- function(N, n) {  
 a <- N\*(1-(1-(1/N))^n)  
 return(a)  
}  
ED.2 <- f.ED(12,20)  
  
print(ED)

## [1] 9.8835

print(ED.1)

## [1] 9.894234

print(ED.2)

## [1] 9.894234