Tarea III

Rayan García Fabián 144424, Bernardo Mondragon Brozon 143743, Karen DElgado Curiel 142252, Diego Garcia 14xxxx

1 October 2018

Problema 1

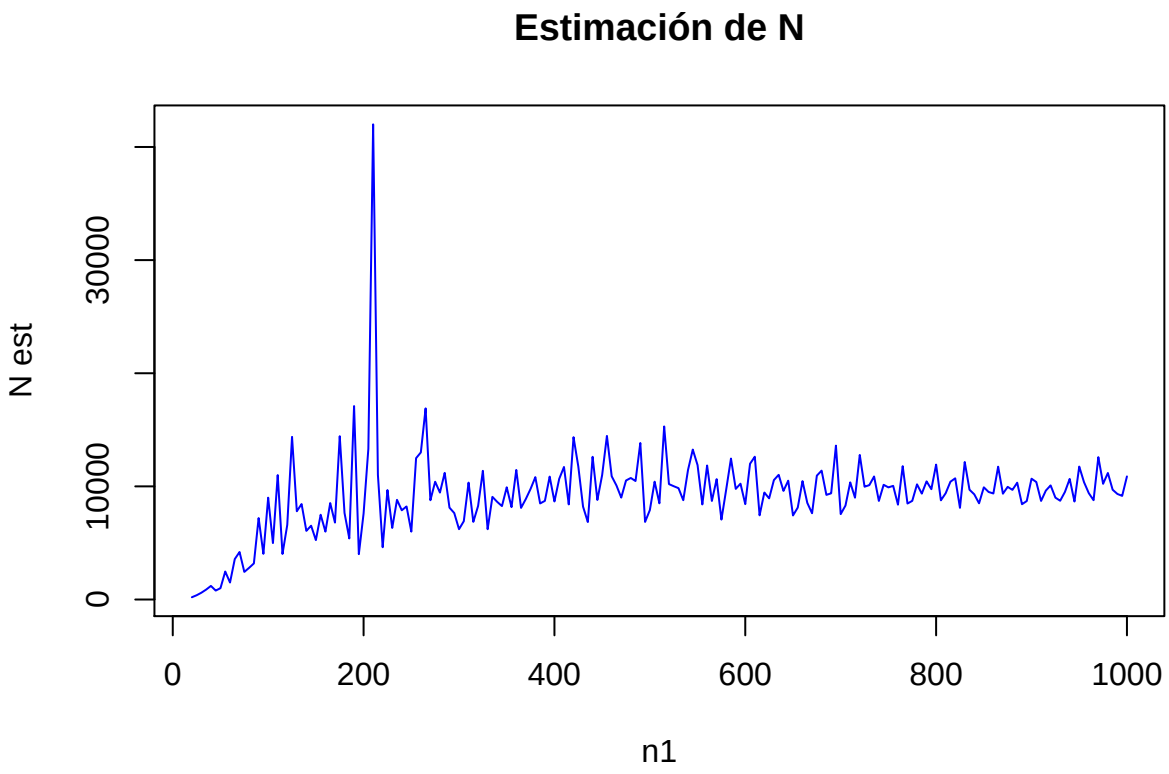
Un estadístico está interesado en el número N de peces en un estanque. Él captura 250 peces, los marca y los regresa al estanque. Unos cuantos días después regresa y atrapa peces hasta que obtiene 50 peces marcados, en ese punto también tiene 124 peces no marcados (la muestra total es de 174 peces).

- ¿Cuál es la estimación de N?
- Haga un programa, que permita simular el proceso de obtener la primera y segunda muestra considerando como parámetros el tamaño N de la población de interés, el tamaño de la primera y segunda muestra y como datos a estimar son: de qué tamaño debe ser n_1 y n_2 para obtener una buena aproximación y ver cómo se afecta por el tamaño N.

Solución:

Primero definamos la notación a usar. Consideremos N como el número de peces en la población, n_1 el número de peces marcados en la primer muestra, n_2 el número de peces capturados en la segunda muestra y r el número de animales de la segunda muestra que están marcados. Sabemos que $N = 174, n_1 = 250, n_2 = 174, r = 50$.Entonces por el método de Lincoln–Petersen, suponiendo que no cambió la población de peces entre el momento de la primera y segunda muestra, se estima que $\hat{N} = \frac{n_1 n_2}{r}$, sustituyendo valores $\hat{N} = 870$.

```
Simulacion_Peces<-function(N,n1,n2){ #Función para simular el proceso de obtener la primera y segunda m
  pecesM<-c(rep("M",n1))
  pecesT<-c(pecesM,c(rep("NM",N-n1)))
  muestra2<-sample(pecesT,n2)
  r<-length(subset(muestra2,muestra2=="M"))
  N_est<-(n1*n2)/(r+1) #Ojo aquí consideramos r+1 ya que como no asignamos probabilidades a la muestra,
  return(N_est)
}

#Supongamos N=10000, vamos a ver de qué tamaño deben ser n1 y n2 para obtener una buena estimación de N
N<-10000
n1<-c(seq(20,1000,by=5))
n2<-c(seq(10,990,by=5))

M_est<-c()
for (i in 1:length(n1)){
  M_est[i]<-Simulacion_Peces(N,n1[[i]],n2[[i]])
}

plot(n1[1:length(n1)],M_est[1:length(n1)],type = "l",col="blue",main = "Estimación de N", xlab = "n1",yl
```



■

Problema 2

Este problema es una versión simplificada de dos problemas comunes que enfrentan las compañías de seguros: calcular la probabilidad de quebrar y estimar cuánto dinero podrán hacer.

Supongan que una compañía de seguros tiene activos (todo en dólares) por \$1000000. Tienen $n = 1000$ clientes que pagan individualmente una prima anual de \$5500 al principio de cada año. Basándose en experiencia previa, se estima que la probabilidad de que un cliente haga un reclamo en el año es de $p = 0.1$, independientemente del número de reclamos previos de otros clientes. El tamaño X de los reclamos varía y tiene la siguiente distribución de probabilidad:

$$f_X(x) = \frac{\alpha\beta^\alpha}{(x+\beta)^{\alpha+1}} I_{[0,\infty)}(x)$$

con $\alpha = 5$ y $\beta = 125000$ (Tal X tiene una distribución Pareto, la cual es frecuentemente usada para modelar el monto de un siniestro). Suponemos las fortunas de la compañía aseguradora sobre un horizonte de 5 años. Sea $Z(t)$ el valor de los activos de la compañía al final del t-ésimo año, de manera que que $Z(0) = 1000000$ y

$$Z(t) = \max(Z(t-1) + P(t) - S(t), 0)$$

donde $P(t)$ es el monto de las primas pagadas durante el t-ésimo año y $S(t)$ es el monto de los siniestros pagados durante el t-ésimo año. Notar que si $Z(t)$ cae bajo 0, entonces la compañía se va a la bancarrota y deja de operar.

- Calcular $F_X(x)$, $E(X)$ y $Var(X)$. Obtener por simulación una muestra de X y hallar los valores estimados de las cantidades anteriores y compararlos con los valores teóricos.
- Escriban una función para simular los activos de la compañía por cinco años y estimar lo siguiente: (1) La probabilidad de que la compañía se vaya a la bancarrota. (2) Los activos esperados al final del quinto año.
- Si el valor de los activos rebasan la cantidad de $1000000, entonces el excedente se reparte entre los accionistas como dividentos de manera que si $D(t)$ son los dividendos pagados al final del t-ésimo año, entonces

$$D(t) = \begin{cases} 1000000 - Z(t) & \text{si} \quad Z(t) > 1000000 \\ 0 & \text{si} \quad Z(t) \leq 1000000 \end{cases}.$$

  Bajo este nuevo esquema, estimar (1) la probabilidad de irse a la quiebra, (2) los activos esperados después de 5 años, y (3) las ganacias totales esperadas después de 5 años de operación.

Solución:

$$F_X = \int_0^x \frac{\alpha\beta^\alpha}{(s+\beta)^{\alpha+1}} ds = \alpha\beta^\alpha \int_0^x \frac{1}{(+\beta)^{\alpha+1}} ds = \alpha\beta^\alpha \int_\beta^{\beta+x} \frac{1}{u^{\alpha+1}} du = \alpha\beta^\alpha \left(\frac{-u^{-\alpha}}{\alpha}\right)\Big|_\beta^{\beta+x} = 1 - \left(\frac{\beta}{\beta+x}\right)^\alpha$$

Para el cálculo de la esperanza notemos que X es una v.a no negativa, entonces se puede proceder de manera más sencilla por medio de la función de supervivencia $S(x) = 1 - F_X(x)$

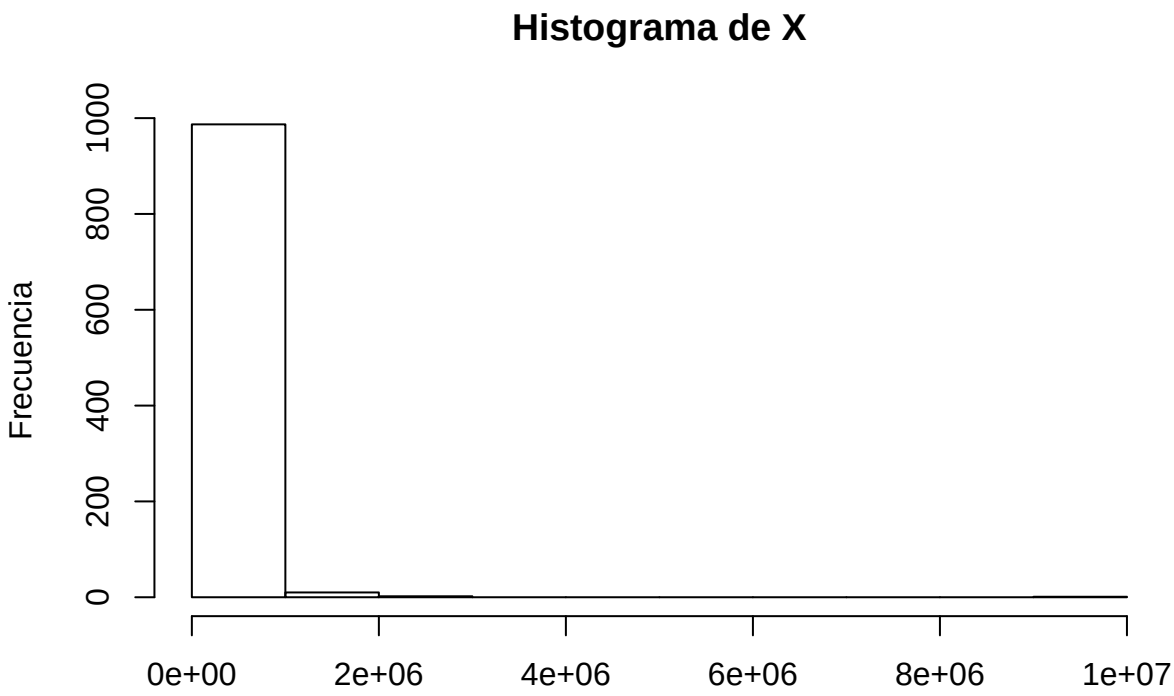
$$E[X] = \int_0^\infty x(1 - F_X(x))dx = \int_0^\infty \frac{\beta^\alpha}{(x+\beta)^\alpha} dx = \frac{\beta}{\alpha - 1}$$

De manera alternativa podemos considerar a Y distribuida Pareto del tipo I, sabemos que $E[Y] = \frac{\alpha\beta}{\alpha-1}$. Entonces sea $X = Y - \beta$, se tiene que $E[X] = E[Y] - \beta = \frac{\alpha\beta}{\alpha-1} - \beta$.

Partiendo de la transformación anterior se tiene que $Var(X) = Var(Y) = \frac{\alpha^2}{(\alpha-1)^2(\alpha-2)}; \alpha > 2$.

Ahora obtendremos por medio de simulación una muestra de X y su función de distribución

```
Pareto_Lomax<-function(a,b,n){
  u<-runif(n)
  Lomax_MA<-c()
  for (i in 1:n) {
  Lomax_MA[i]<-(b*(1-(1-u[[i]])^(1/a)))/((1-u[[i]])^(1/a))
  }
  return(Lomax_MA)
}

hist(Pareto_Lomax(2,125000,1000),main = "Histograma de X",xlab = "",ylab = "Frecuencia")
```



Realizamos la simulación de los activos de la compañía en un horizonte de cinco años.S upondremos además que la aseguradora siempre renueva la póliza con el cliente, esto es que una vez que un cliente reclama se le vuelve a vender la póliza al año siguiente.

```
registro<-function(activos,prima,n_clientes,prob_reclamo){
registro_activos<-c()
registro_activos[1]<-activos
for (i in 2:6) {
reclamos<-rbinom(n_clientes,1,prob_reclamo)
monto_reclamos<-Pareto_Lomax(5,125000,sum(reclamos))
registro_activos[i]<-registro_activos[i-1]-sum(monto_reclamos)+n_clientes*prima
if(registro_activos[i-1]<=0){
  registro_activos[i]<-0
  }
}
return(registro_activos)
}

activos<-10e6
prima<-5500
n_clientes<-1000
prob_reclamo<-0.1

huella<-matrix(,100,6)
for (i in 1:100) {
  huella[i,]<-registro(activos =activos, prima = prima, n_clientes = n_clientes,prob_reclamo = prob_rec
}
```

```r
colnames(huella)<-c("z(0)","z(1)","z(2)","z(3)","z(4)","z(5)")
huella<-as.data.frame(huella)
head(huella)
```

```
##    z(0)     z(1)     z(2)     z(3)     z(4)     z(5)
## 1 1e+07 12915805 15937153 18571763 20626169 22816790
## 2 1e+07 12916202 14315085 16967673 18263638 19265034
## 3 1e+07 12123170 14636715 17839948 19418729 21873386
## 4 1e+07 11610395 14748027 16876385 20009765 23240644
## 5 1e+07 11810233 14116783 16932850 18707072 21011452
## 6 1e+07 12160209 14649726 17257087 20296667 22757246
```

Notemos que hasta el momento no hemos visto ningún caso de ruina. Estudiemos un esquema en el que si el registro de activos en el periodo excede cierta cantidad, entonces se recompensa a los accionistas.

```r
registro_acc<-function(activos,prima,n_clientes,prob_reclamo){
registro_activos<-c()
registro_activos[1]<-activos
for (i in 2:6) {
reclamos<-rbinom(n_clientes,1,prob_reclamo)
monto_reclamos<-Pareto_Lomax(5,125000,sum(reclamos))
if(registro_activos[i-1]<=0){
  registro_activos[i:6]<-0
}
else if(registro_activos[i-1]>1000000){
registro_activos[i]<-registro_activos[i-1]-(10e6)-sum(monto_reclamos)+n_clientes*prima
}
else{registro_activos[i]<-registro_activos[i-1]-sum(monto_reclamos)+n_clientes*prima
}
}
return(registro_activos)
}

activos<-10e6
prima<-5500
n_clientes<-1000
prob_reclamo<-0.1

huella<-matrix(,100,6)
for (i in 1:100) {
  huella[i,]<-registro_acc(activos =activos, prima = prima, n_clientes = n_clientes,prob_reclamo = prob
}

colnames(huella)<-c("z(0)","z(1)","z(2)","z(3)","z(4)","z(5)")
huella<-as.data.frame(huella)
head(huella)
```

```
##    z(0)    z(1)     z(2) z(3) z(4) z(5)
## 1 1e+07 1573941 -6192345    0    0    0
## 2 1e+07 2805880 -4810605    0    0    0
## 3 1e+07 2045108 -5023693    0    0    0
## 4 1e+07 3160797 -3877036    0    0    0
## 5 1e+07 2571368 -4382369    0    0    0
## 6 1e+07 2231031 -5580870    0    0    0
```

Notemos que bajo este esquema en el que los accionistas son recompensados, la probabilidad de quiebra es 1.

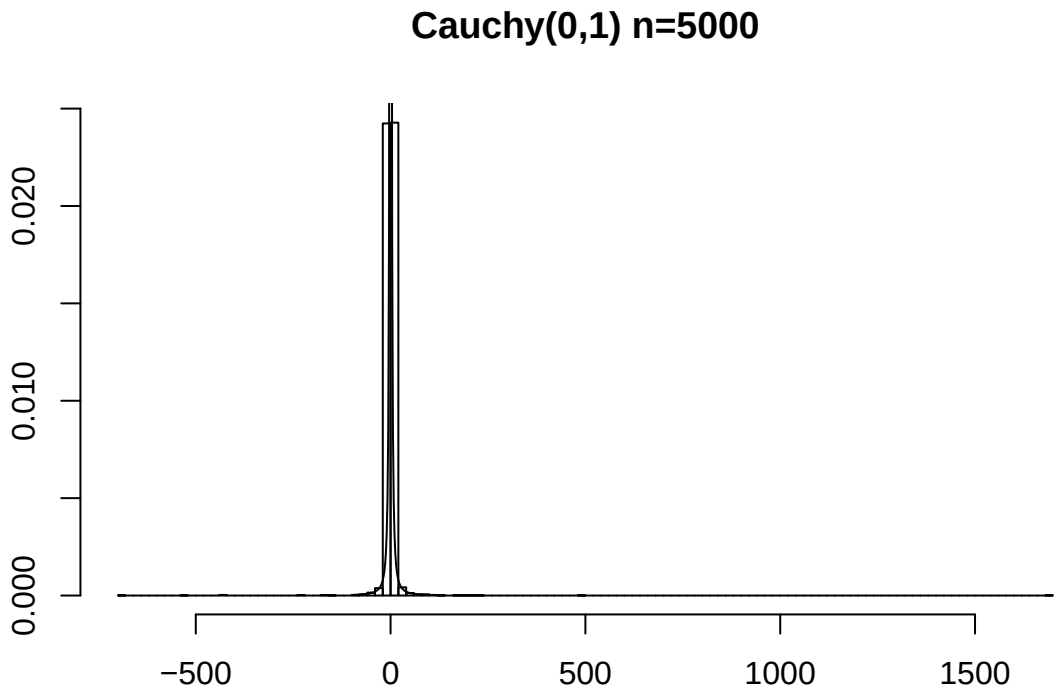
Problema 3

Proponer algoritmos (método y pseudocódigo o código, así como una corrida) para generar muestras de las siguientes densidades.

*Cauchy $f(x) = \frac{1}{\pi\beta\left[1+(\frac{x-\gamma}{\beta})^2\right]}; \gamma, x \in \mathbb{R}; \beta > 0$

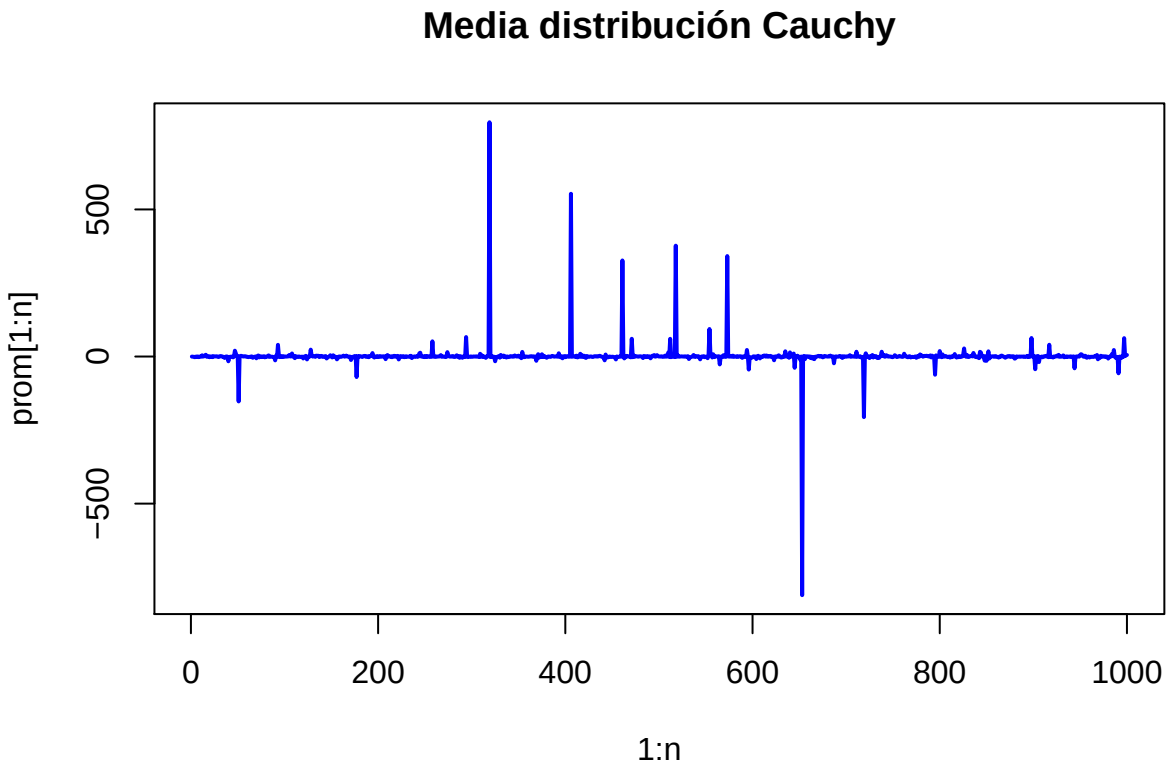
Solución: Podemos encontrar la distribución de X como $F_X(x) = \frac{arctan(\frac{x-\gamma}{\beta})}{\pi} + \frac{1}{2}$, entonces $F^{-1}(u) = tan(\pi(u - \frac{1}{2}))$. Usamos el método de la transformada inversa

```
cauchy<-function(gamma,beta,n){
u<-runif(n)
u<-tan(pi*u)*beta+gamma
return(u)
}
x<-1:100
hist(cauchy(0,1,5000),probability=T,breaks=100,main = "Cauchy(0,1) n=5000",ylab = "",xlab = "")
curve(dcauchy(x),add=T,from=-100,to=100)
```




```
U<- numeric(1000)
n<-1000
prom<-numeric(n)
y<-c()
```

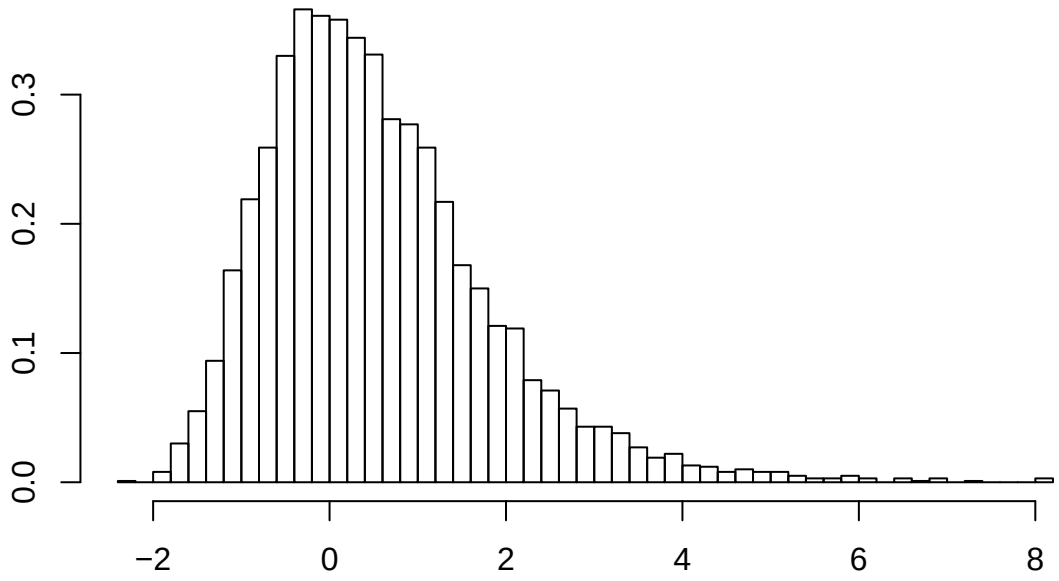
```r
for (i in 1:n) {
  u<-runif(1000)
  x<-tan(pi*(u-0.5))
  prom[i]<-mean(x)
}
plot(1:n,prom[1:n],type="l",lwd=2,col="blue",main = "Media distribución Cauchy")
```



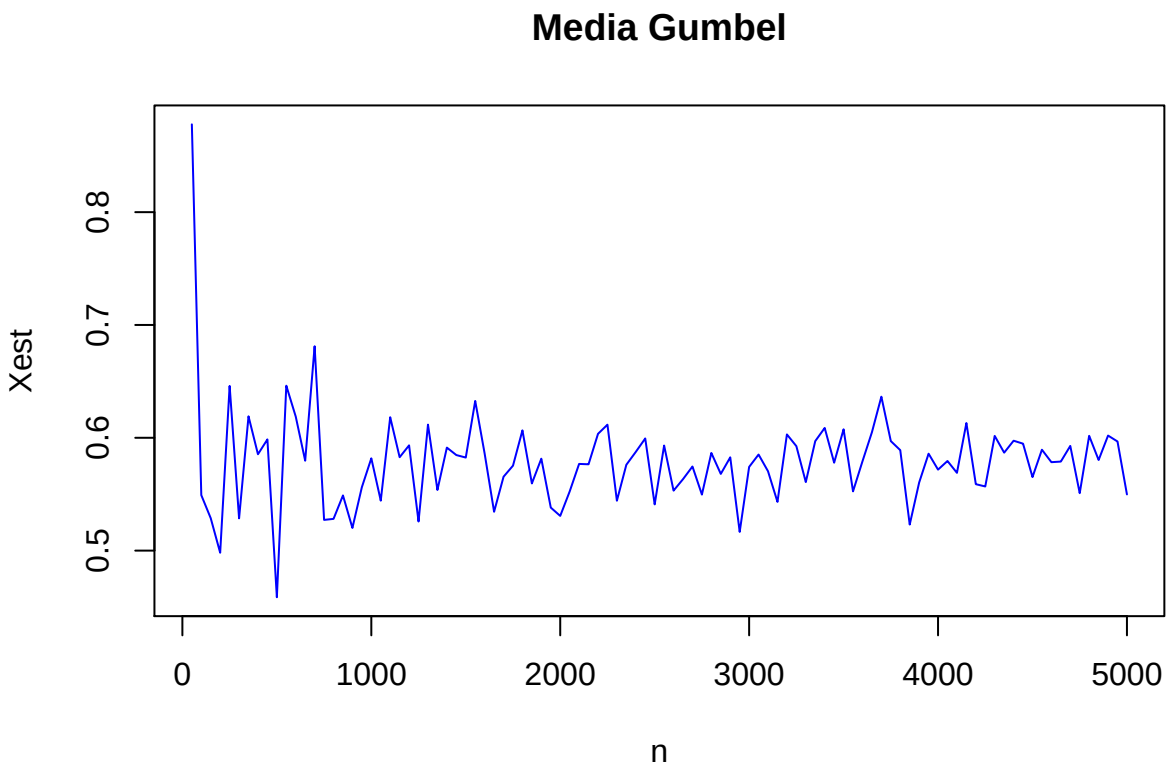
*Gumbel $f(x) = \frac{1}{\beta} exp\left[-e^{-\frac{(x-\gamma)}{\beta}} - \frac{x-\gamma}{\beta}\right]; \gamma, x \in \mathbb{R}; \beta > 0.$

Solución:

```r
gumbel<-function(gamma,beta,n){
unif<-runif(n)
unif<--beta*log(-log(unif))+gamma
return(unif)
}
hist(gumbel(0,1,5000),probability =T,breaks=60,xlab = "",ylab="",main = "Distribución Gumbel")
```




```
n_s<-c(seq(50,5000,by=50))
X_est<-c()
for (i in 1:length(n_s)) {
  X_est[i]<-sum(gumbel(0,1,n_s[i]))/n_s[i]
}

plot(n_s[1:length(n_s)],X_est[1:length(n_s)],type = "l",col="blue", xlab = "n",ylab = "Xest",main="Media
```



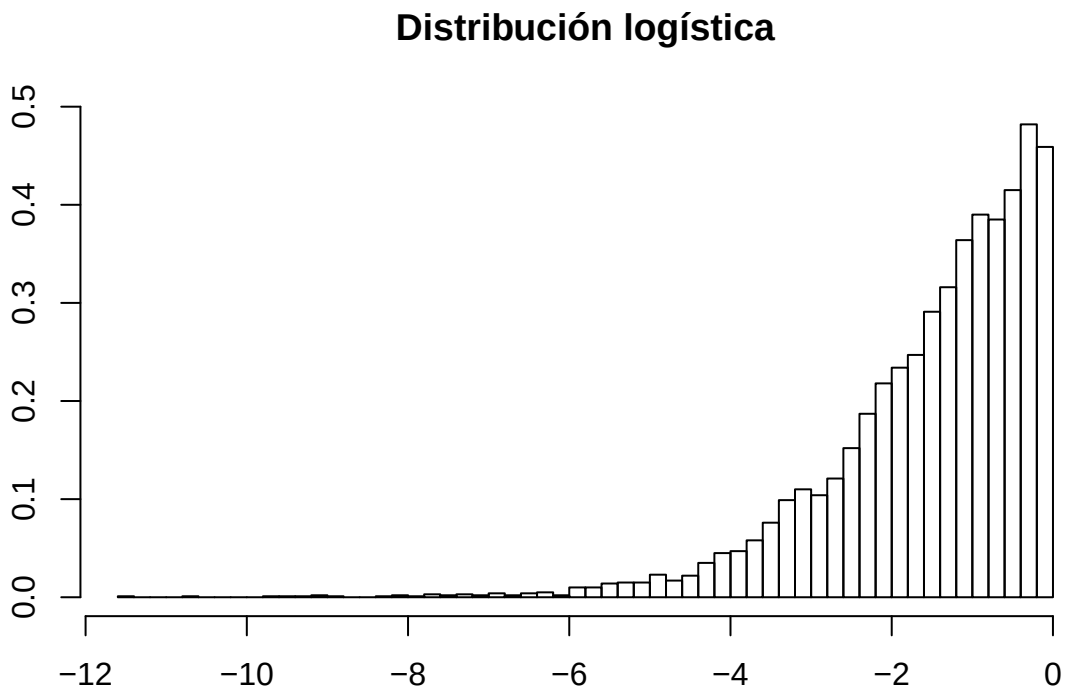
*Logística

$$f_X(x) = \frac{e^{-\frac{x-\gamma}{\beta}}}{\beta\left(1+e^{-\frac{x-\gamma}{\beta}}\right)^2} \quad \text{con} \quad \gamma, x \in \mathbb{R} \quad \text{y} \quad \beta > 0$$

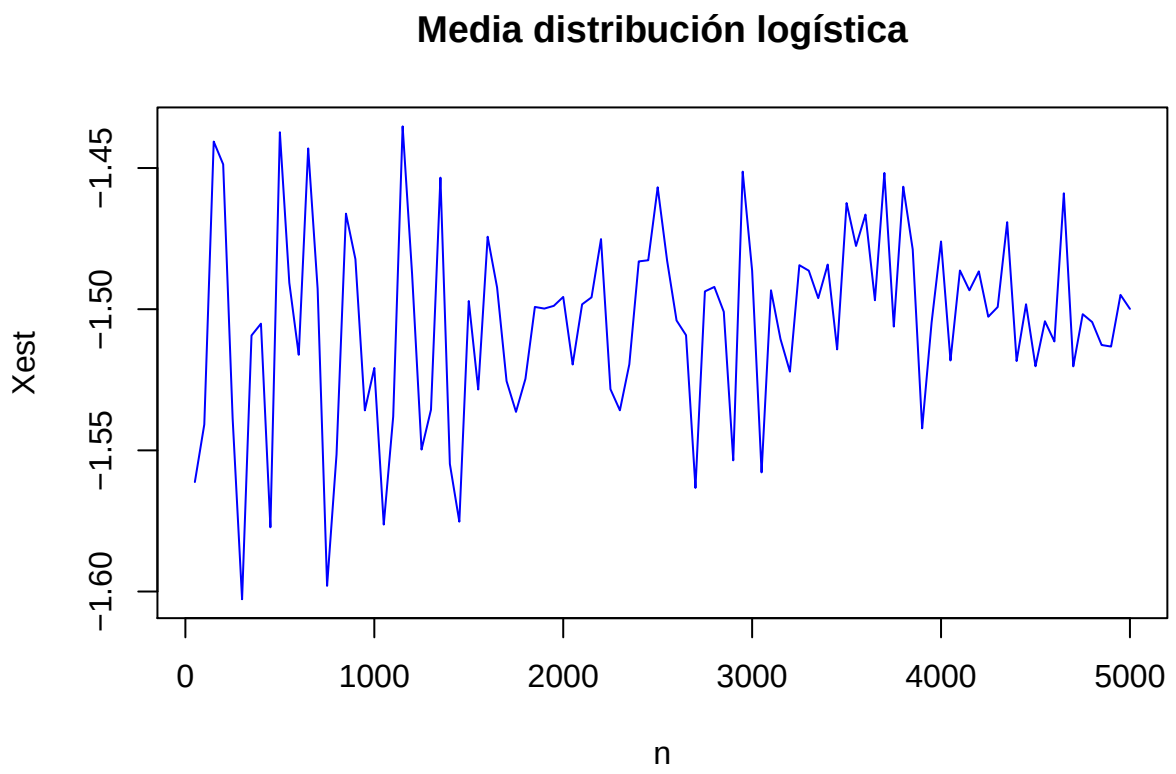
Solución:

Para esto utilizaremos el hecho de que si $X \sim U(0,1)$, entonces $\gamma + \beta(log(X) - (1-X)) \sim log(\gamma, \beta)$

```
Dist_logistica<-function(n,g,b){
u<-runif(n)
x<-g+b*(log(u)-1+u)
return(x)
}

hist(Dist_logistica(5000,0,1),probability =T,breaks=60,main = "Distribución logística",xlab = "",ylab =
```




```
n_s<-c(seq(50,5000,by=50))
X_est<-c()
for (i in 1:length(n_s)) {
  X_est[i]<-sum(Dist_logistica(n_s[i],0,1))/n_s[i]
}

plot(n_s[1:length(n_s)],X_est[1:length(n_s)],type = "l",col="blue", xlab = "n",ylab = "Xest",main="Media
```



*Pareto $f(x) = \frac{\alpha_2 c^{\alpha_2}}{x^{\alpha_x+1}}; c > 0, \alpha_2 > 0, x > c.$

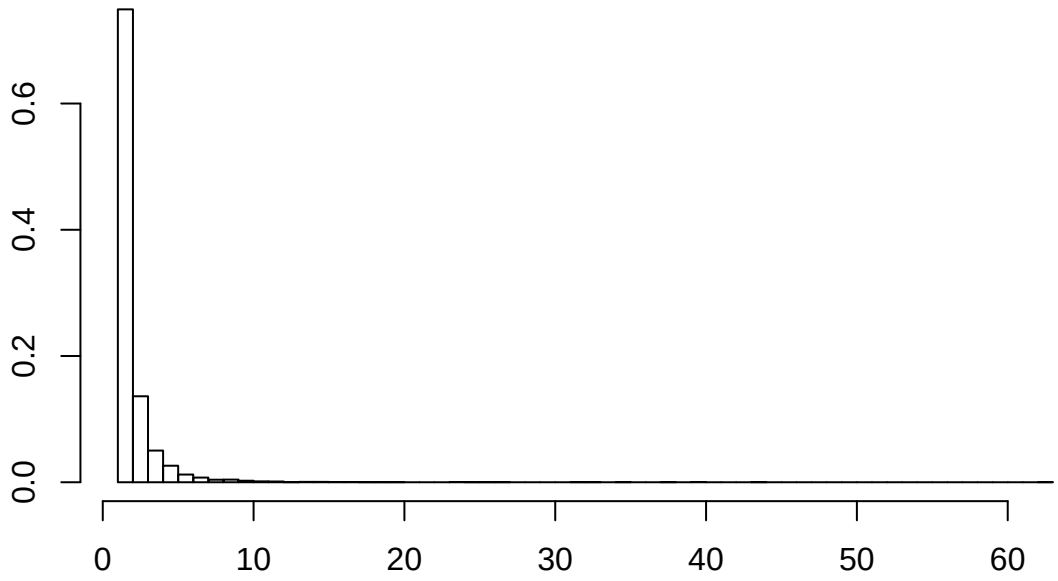
Solución:

$$F_X = \int_c^x \frac{\alpha_2 c^{\alpha_2}}{s^{\alpha_x+1}} ds = \alpha_2 c^{\alpha_2} \frac{s^{-\alpha_2}}{-\alpha_2}\Big|_c^x = 1 - (\frac{c}{x})^{\alpha_2}$$

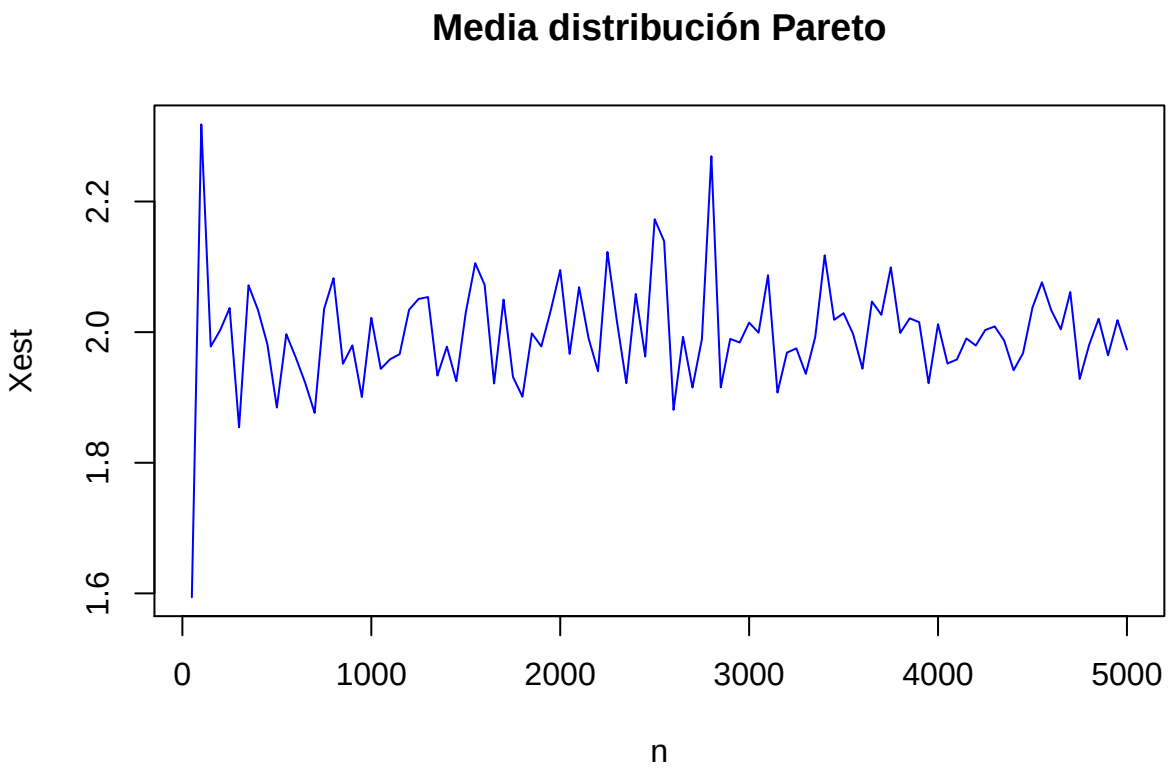
.

$$F^{-1}(u) = \frac{c}{(1-u)^{\frac{1}{\alpha_2}}}$$

```
S_pareto<-function(n,c,alfa){
  u<-runif(n)
  x<-c/(1-u)^(1/alfa)
  return(x)
}

hist(S_pareto(5000,1,2),probability =T,breaks=60,main = "Distribución Pareto",xlab = "",ylab = "")
```




```
n_s<-c(seq(50,5000,by=50))
X_est<-c()
for (i in 1:length(n_s)) {
  X_est[i]<-sum(S_pareto(n_s[i],1,2))/n_s[i]
}

plot(n_s[1:length(n_s)],X_est[1:length(n_s)],type = "l",col="blue", xlab = "n",ylab = "Xest", main="Medi
```



Problema 4

Grafiquen las siguientes densidades. Dar los algoritmos de transformación inversa, composición y aceptación-rechazo para cada una de las siguientes densidades. Discutir cuál algoritmo es preferible para cada densidad.

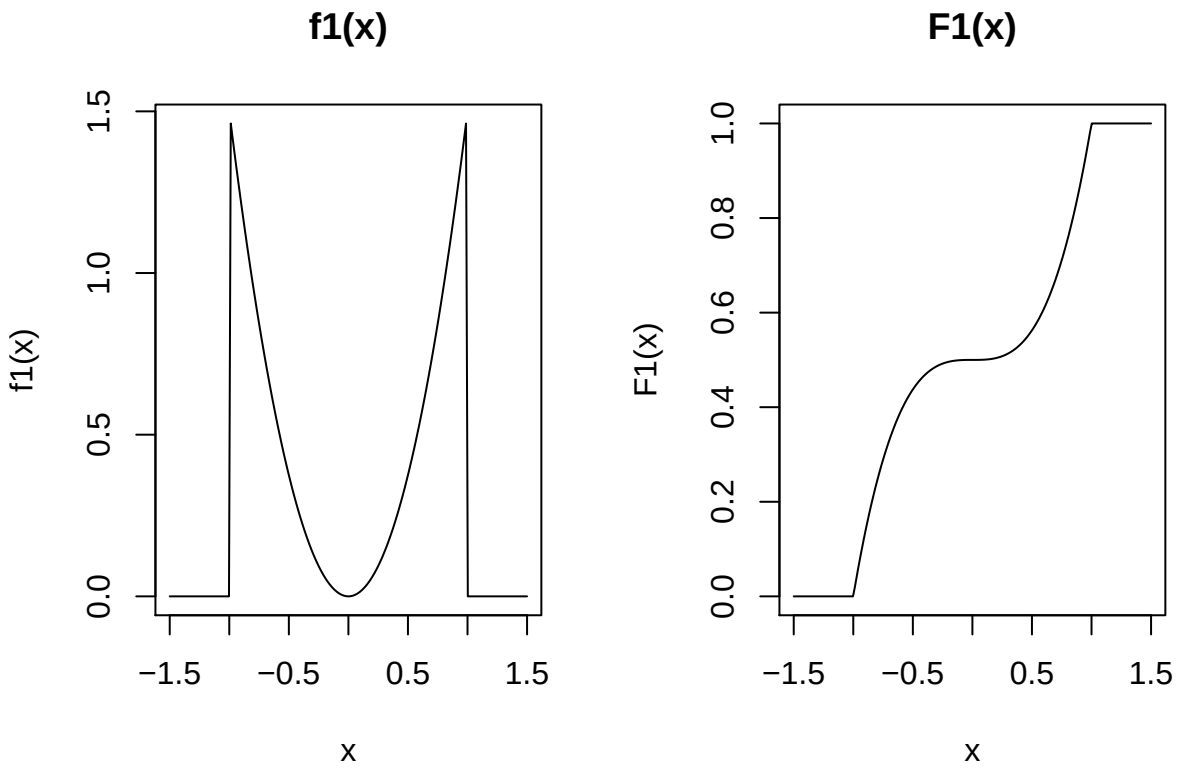
$$f(x) = \frac{3x^2}{2} I(x)_{[-1,1]}$$

$$f(x) = \begin{cases} 0, x \leq 0 \\ \frac{x}{a(1-a)}, 0 \leq x \leq a \\ \frac{1}{1-a}, a \leq x \leq 1-a \\ \frac{1-x}{a(1-a)}, 1-a \leq x \leq 1 \\ 0, x \geq 1 \end{cases}$$

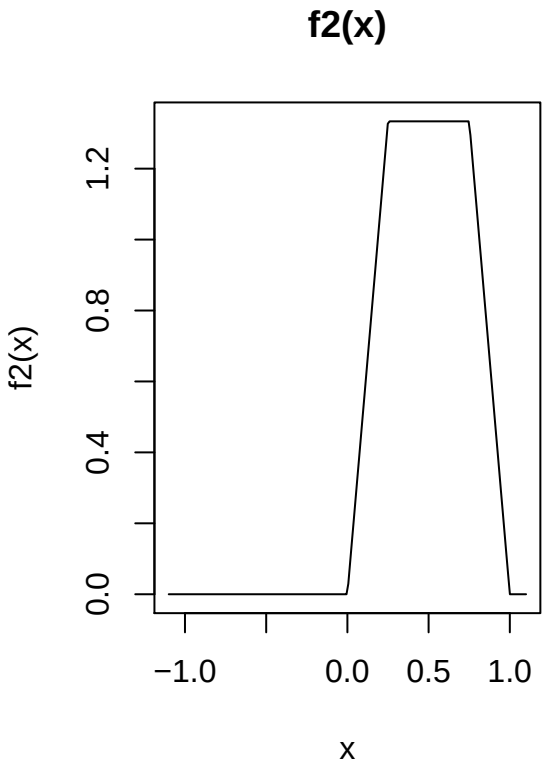
Solución:

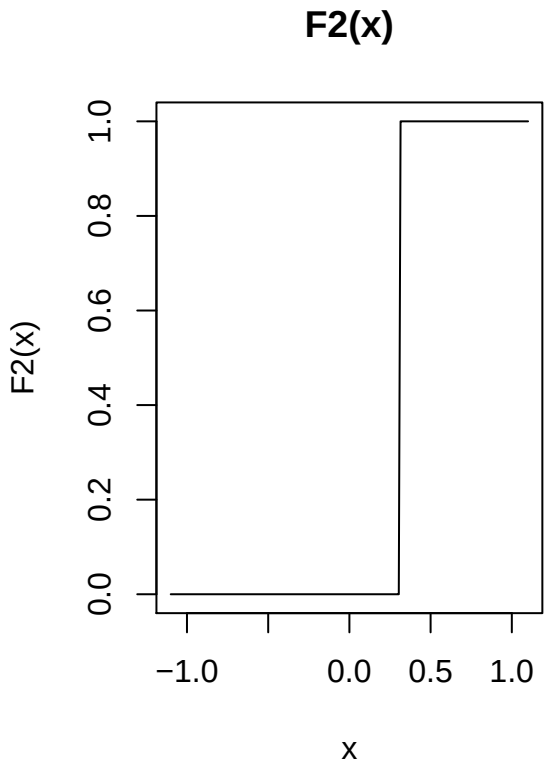
```
ind<-function(x,a,b){
ifelse(x<=b & x>= a,1,0)
}

f1<-function(x){
(3*(x^2)/2)*ind(x,-1,1)
}
```

```r
F1<-function(x){
ifelse(x<=-1,0,ifelse(x<=1,0.5*(x^3+1),1))
}

x<-seq(-1.5,1.5,length=200)
par(mfrow=c(1,2))
plot(x,f1(x),type="l",main="f1(x)")
plot(x,F1(x),type = "l",main = "F1(x)")
```




```r
f2<-function(x,a=0.25){
ind(x,-1,1)*(ind(x,0,a)*(x/(a*(1-a)))+ind(x,a,1-a)/(1-a)+ind(x,1-a,1)*((1-x)/(a*(1-a))))
}

F2<-function(x,a=0.25){
ind(x,0,a*x^2/(2*a*(1-a))+(x-a/2)/(1-a)*ind(x,a,1-a)+((1-3*a/2)/(1-a)+(x*(1-x/2)-(1-a)*(1+a)/2)/(a*(1-a
}

par(mfrow=c(1,2))
x<-seq(-1.1,1.1,length=200)
plot(x,f2(x),type="l",main="f2(x)")
plot(x,F2(x),type="l",main="F2(x)")
```





Problema 5

Considerando la transformación polar de Marsaglia para generar muestras de normales estándar, muestren que la probabilidad de aceptación de $S = V_1^2 + V_2^2$ en el paso 2 es $\frac{\pi}{4}$. Encuentre la distribución del número de rechazos de S antes de que ocurra una aceptación. ¿Cuál es el número esperado de ejecuciones del paso 1?

Solución:

Notemos que gráficamente estamos trabajando con un circulo unitario dentro de un cuadrado de 1x1, se acepta si $S = V_1^2 + V_2^2$ cae dentro del circulo, y se rechaza si cae en el área restante. Entonces la probabiidad de aceptación es el área del circulo unitario $A = \frac{\pi r^2}{2} = \frac{\pi}{4}$. Para modelar la distribución del número de rechazos de S antes de una aceptación, basta con definir $X \sim Geo(p)$ donde p es la probabilidad de aceptación, en este caso $\frac{\pi}{4}$. Finalmente $E[X] = \frac{1-p}{p} = 3\pi$.

Problema 6

Obtengan una muestra de 1,000 números de la siguiente distribución discreta, para $k = 100$.

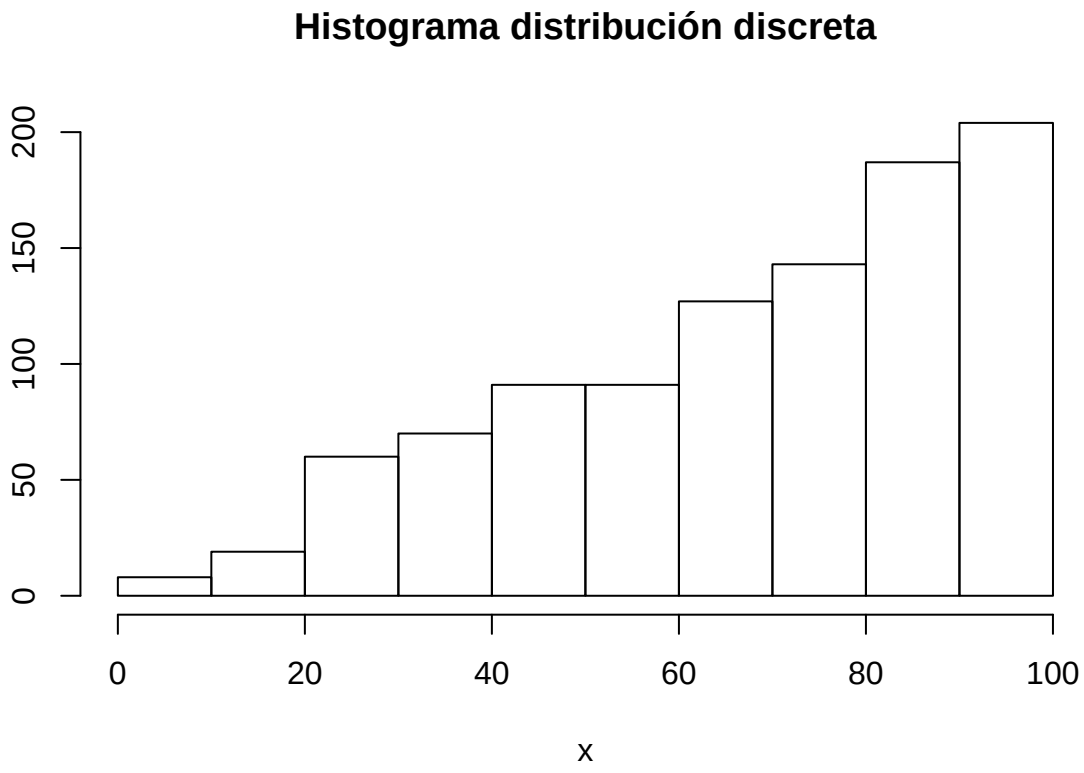
$$p(x) = \frac{2x}{k(k+1)}; x = 1, 2, ...k$$

Solución:

```r
x <-sample(1:100,size=1000,replace=T,prob=c(seq(1:100)*(2/10100)))
x[1:50]
```

```
##  [1] 26 89 47 27 23 70 64 57 25 90 75 37 86 50 84 94 45 98 89 98 46 78 80
## [24] 86 83 32 96 93 98 92 59 85 58 71 42 38 78 98 65 95 87 71 65 87 38 96
## [47] 41 86 90 97
```

```r
hist(x,main = "Histograma distribución discreta",ylab = "")
```



Problema 7

Desarrollen un algoritmo para generar una variable aleatoria binomial, usando la técnica de convolución (Hint: ¿cuál es la relación entre la distribución binomial y Bernoulli?). Generar una muestra de 100,000 números. ¿Qué método es más eficiente, el de convoluciones o la función rbinom en R?

```r
s_Binom<-function(n,t,p){
  esp_muestral<-c(0,1)
  muestra_Binom<-c()
  for (i in 1:n) {
  muestra_Binom[i]<-sum(sample(esp_muestral,t,replace = TRUE,prob = c(p,1-p)))
  }
  return(muestra_Binom)
}
```

```r
ptm <- proc.time()
prueba1<-s_Binom(1,100000,0.4)
```

```
proc.time()-ptm
```

```
##    user  system elapsed
##    0.02    0.00    0.01
```

```
ptm<-proc.time()
prueba2<-rbinom(1,100000,0.4)
proc.time()-ptm
```

```
##    user  system elapsed
##       0       0       0
```

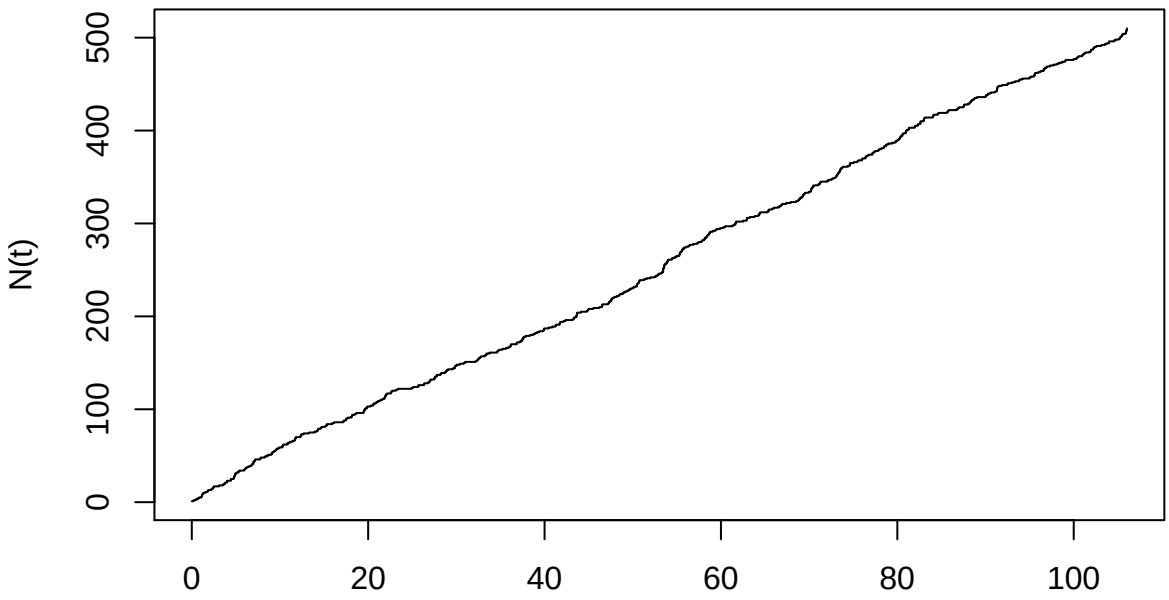
Resulta más eficiente realizar una muestra de 100000 números con la función rbinom.

Problema 8

Para un proceso Poisson no homogéneo con función de intensidad dada por:

$$\lambda(t) = \begin{cases} 5, t \in (1,2],(3,4],(5,6]... \\ 3, t \in (0,1],(2,3],(4,5]... \end{cases}$$

```
lambdat<-function(t){
x<-paste("","{",0,"<=t & t<",1,"}",sep="")
for(i in seq(2,100,2)){
x<-paste(x,paste("","{",i,"<=t & t <",1+i,"}",sep=""),sep="|")
}
return(ifelse(eval(parse(text=0)),3,5))
}
poissonnohomogeneo<-function(lambdat,n,pic=T){
    lambda<-5
    TT<-rexp(n,lambda)
    s<-cumsum(TT)
    u<-runif(n)
    ss<-s[u<=lambdat(s)/lambda]
    Ns<-1:length(ss)
    if(pic==T){
    plot(ss,Ns,type="s",xlab="",ylab="N(t)",main="Proceso Poisson no homogéneo")
    return(list(ss,cuenta=Ns))
}
}
poissonnohomogeneo(lambdat,510)
```




```
## [[1]]
##   [1] 5.572995e-03 1.773449e-01 4.251868e-01 6.501915e-01 7.839535e-01
##   [6] 1.074961e+00 1.146466e+00 1.180003e+00 1.200582e+00 1.300126e+00
##  [11] 1.548039e+00 1.785775e+00 1.836395e+00 2.198290e+00 2.337036e+00
##  [16] 2.447201e+00 2.490165e+00 3.071560e+00 3.514758e+00 3.695071e+00
##  [21] 3.755876e+00 3.973802e+00 3.993000e+00 4.410419e+00 4.441685e+00
##  [26] 4.754354e+00 4.799326e+00 4.808206e+00 4.880340e+00 4.919505e+00
##  [31] 4.937642e+00 5.097582e+00 5.286167e+00 5.387916e+00 5.928475e+00
##  [36] 6.036345e+00 6.113080e+00 6.292040e+00 6.522498e+00 6.761239e+00
##  [41] 6.818433e+00 6.980778e+00 6.988241e+00 7.007681e+00 7.144298e+00
##  [46] 7.186037e+00 7.770524e+00 7.792179e+00 8.271886e+00 8.465975e+00
##  [51] 8.682446e+00 9.017392e+00 9.115035e+00 9.152411e+00 9.322194e+00
##  [56] 9.495138e+00 9.577093e+00 9.716518e+00 9.924309e+00 1.027756e+01
##  [61] 1.028289e+01 1.037194e+01 1.085057e+01 1.091162e+01 1.113619e+01
##  [66] 1.142341e+01 1.166457e+01 1.170139e+01 1.176589e+01 1.178629e+01
##  [71] 1.229520e+01 1.239669e+01 1.240239e+01 1.266734e+01 1.330473e+01
##  [76] 1.393722e+01 1.418756e+01 1.431859e+01 1.432012e+01 1.461629e+01
##  [81] 1.471394e+01 1.512474e+01 1.527106e+01 1.530757e+01 1.589980e+01
##  [86] 1.613011e+01 1.718671e+01 1.723739e+01 1.749612e+01 1.749786e+01
##  [91] 1.768613e+01 1.808229e+01 1.816370e+01 1.818196e+01 1.847596e+01
##  [96] 1.866172e+01 1.946615e+01 1.948611e+01 1.955950e+01 1.959529e+01
## [101] 1.968776e+01 1.984279e+01 1.989685e+01 2.029310e+01 2.058445e+01
## [106] 2.058669e+01 2.080825e+01 2.096907e+01 2.114213e+01 2.139418e+01
## [111] 2.159679e+01 2.182548e+01 2.186980e+01 2.193435e+01 2.197938e+01
## [116] 2.200265e+01 2.214783e+01 2.257830e+01 2.260665e+01 2.272282e+01
## [121] 2.315482e+01 2.337780e+01 2.491779e+01 2.511749e+01 2.568659e+01
```

```
## [126] 2.570081e+01 2.633635e+01 2.636292e+01 2.679234e+01 2.697796e+01
## [131] 2.698268e+01 2.708705e+01 2.750209e+01 2.751555e+01 2.758146e+01
## [136] 2.774603e+01 2.779384e+01 2.820134e+01 2.827670e+01 2.869324e+01
## [141] 2.884773e+01 2.888462e+01 2.902803e+01 2.957543e+01 2.980858e+01
## [146] 2.983335e+01 2.988384e+01 3.007880e+01 3.040861e+01 3.084888e+01
## [151] 3.099531e+01 3.220143e+01 3.235255e+01 3.245531e+01 3.254417e+01
## [156] 3.265879e+01 3.279347e+01 3.319700e+01 3.338177e+01 3.340416e+01
## [161] 3.377538e+01 3.464170e+01 3.467410e+01 3.481478e+01 3.520513e+01
## [166] 3.561669e+01 3.594528e+01 3.611660e+01 3.616621e+01 3.618888e+01
## [171] 3.683934e+01 3.686630e+01 3.720400e+01 3.737332e+01 3.746675e+01
## [176] 3.749858e+01 3.759781e+01 3.763138e+01 3.787718e+01 3.836445e+01
## [181] 3.872308e+01 3.890738e+01 3.913948e+01 3.939397e+01 3.986256e+01
## [186] 3.991282e+01 3.995415e+01 4.046720e+01 4.087473e+01 4.125298e+01
## [191] 4.128302e+01 4.167982e+01 4.175124e+01 4.176760e+01 4.214844e+01
## [196] 4.236299e+01 4.318533e+01 4.332688e+01 4.337118e+01 4.360378e+01
## [201] 4.360786e+01 4.365852e+01 4.366667e+01 4.370598e+01 4.414232e+01
## [206] 4.479725e+01 4.491234e+01 4.499640e+01 4.553764e+01 4.622008e+01
## [211] 4.652403e+01 4.652841e+01 4.656139e+01 4.719545e+01 4.731028e+01
## [216] 4.735561e+01 4.751194e+01 4.758706e+01 4.759668e+01 4.771394e+01
## [221] 4.793199e+01 4.822014e+01 4.844746e+01 4.854004e+01 4.886342e+01
## [226] 4.896871e+01 4.918601e+01 4.943393e+01 4.959567e+01 4.976091e+01
## [231] 5.001658e+01 5.018364e+01 5.044233e+01 5.047356e+01 5.049991e+01
## [236] 5.059774e+01 5.071448e+01 5.072413e+01 5.078643e+01 5.126511e+01
## [241] 5.155195e+01 5.200107e+01 5.250558e+01 5.268388e+01 5.285583e+01
## [246] 5.295567e+01 5.325163e+01 5.336424e+01 5.344331e+01 5.344971e+01
## [251] 5.345465e+01 5.346023e+01 5.355152e+01 5.356980e+01 5.357191e+01
## [256] 5.360032e+01 5.373845e+01 5.385426e+01 5.390177e+01 5.396818e+01
## [261] 5.399448e+01 5.442577e+01 5.448622e+01 5.480599e+01 5.496822e+01
## [266] 5.528035e+01 5.529089e+01 5.533515e+01 5.536291e+01 5.548302e+01
## [271] 5.559390e+01 5.560322e+01 5.575728e+01 5.578067e+01 5.605240e+01
## [276] 5.633225e+01 5.647181e+01 5.686297e+01 5.731683e+01 5.746073e+01
## [281] 5.780123e+01 5.788441e+01 5.807096e+01 5.816863e+01 5.825668e+01
## [286] 5.837215e+01 5.842567e+01 5.852370e+01 5.866316e+01 5.868488e+01
## [291] 5.877088e+01 5.906504e+01 5.935430e+01 5.950297e+01 5.991010e+01
## [296] 6.025834e+01 6.046705e+01 6.132566e+01 6.153910e+01 6.164461e+01
## [301] 6.165508e+01 6.171566e+01 6.252297e+01 6.290093e+01 6.293658e+01
## [306] 6.295403e+01 6.329486e+01 6.381973e+01 6.422326e+01 6.433505e+01
## [311] 6.437724e+01 6.444235e+01 6.529388e+01 6.539149e+01 6.544585e+01
## [316] 6.580208e+01 6.602459e+01 6.652753e+01 6.670888e+01 6.683954e+01
## [321] 6.694382e+01 6.739207e+01 6.789187e+01 6.850542e+01 6.873452e+01
## [326] 6.886832e+01 6.892561e+01 6.904829e+01 6.922779e+01 6.927836e+01
## [331] 6.931789e+01 6.943911e+01 6.952369e+01 6.996167e+01 7.017042e+01
## [336] 7.018995e+01 7.023269e+01 7.027421e+01 7.035858e+01 7.047617e+01
## [341] 7.049361e+01 7.100491e+01 7.116969e+01 7.126375e+01 7.128878e+01
## [346] 7.207440e+01 7.219229e+01 7.255540e+01 7.274589e+01 7.299445e+01
## [351] 7.308057e+01 7.312303e+01 7.321324e+01 7.327253e+01 7.334643e+01
## [356] 7.344260e+01 7.349936e+01 7.353113e+01 7.356895e+01 7.368582e+01
## [361] 7.380566e+01 7.439288e+01 7.460323e+01 7.463480e+01 7.465769e+01
## [366] 7.504896e+01 7.546841e+01 7.558506e+01 7.599646e+01 7.603540e+01
## [371] 7.638661e+01 7.641654e+01 7.653657e+01 7.675317e+01 7.716603e+01
## [376] 7.718259e+01 7.732545e+01 7.748105e+01 7.783004e+01 7.792346e+01
## [381] 7.818908e+01 7.845512e+01 7.847905e+01 7.868559e+01 7.875218e+01
## [386] 7.898646e+01 7.943960e+01 7.977564e+01 7.986649e+01 8.003829e+01
## [391] 8.019413e+01 8.026774e+01 8.034186e+01 8.036101e+01 8.056080e+01
```

```
## [396] 8.057190e+01 8.064866e+01 8.094149e+01 8.098145e+01 8.098579e+01
## [401] 8.113837e+01 8.130785e+01 8.132107e+01 8.197605e+01 8.198599e+01
## [406] 8.230259e+01 8.233910e+01 8.260152e+01 8.261991e+01 8.263164e+01
## [411] 8.300055e+01 8.300520e+01 8.301635e+01 8.310833e+01 8.402160e+01
## [416] 8.406883e+01 8.409473e+01 8.456899e+01 8.466299e+01 8.563701e+01
## [421] 8.574764e+01 8.581379e+01 8.674102e+01 8.686962e+01 8.697427e+01
## [426] 8.748048e+01 8.755166e+01 8.759747e+01 8.803812e+01 8.821784e+01
## [431] 8.827824e+01 8.837224e+01 8.847539e+01 8.865923e+01 8.877226e+01
## [436] 8.903985e+01 9.000777e+01 9.004522e+01 9.024121e+01 9.032885e+01
## [441] 9.060069e+01 9.107902e+01 9.126572e+01 9.130112e+01 9.132725e+01
## [446] 9.133520e+01 9.135549e+01 9.146154e+01 9.185807e+01 9.247189e+01
## [451] 9.252139e+01 9.301698e+01 9.333903e+01 9.377931e+01 9.387986e+01
## [456] 9.418894e+01 9.498178e+01 9.509445e+01 9.547214e+01 9.556668e+01
## [461] 9.557460e+01 9.562586e+01 9.599860e+01 9.623182e+01 9.657236e+01
## [466] 9.660072e+01 9.666995e+01 9.682705e+01 9.696108e+01 9.731801e+01
## [471] 9.771920e+01 9.809708e+01 9.837271e+01 9.869912e+01 9.900362e+01
## [476] 9.909244e+01 9.998806e+01 1.002947e+02 1.004117e+02 1.005353e+02
## [481] 1.009606e+02 1.010029e+02 1.011686e+02 1.013072e+02 1.017824e+02
## [486] 1.019568e+02 1.019572e+02 1.021793e+02 1.022425e+02 1.023361e+02
## [491] 1.025645e+02 1.031104e+02 1.035000e+02 1.037341e+02 1.039994e+02
## [496] 1.040030e+02 1.045760e+02 1.047523e+02 1.051740e+02 1.052398e+02
## [501] 1.053043e+02 1.054227e+02 1.055227e+02 1.055342e+02 1.058691e+02
## [506] 1.059132e+02 1.059408e+02 1.059896e+02 1.060128e+02 1.060364e+02
##
## $cuenta
##   [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17
##  [18]  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34
##  [35]  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51
##  [52]  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68
##  [69]  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85
##  [86]  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100 101 102
## [103] 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119
## [120] 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136
## [137] 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153
## [154] 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170
## [171] 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187
## [188] 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204
## [205] 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221
## [222] 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238
## [239] 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255
## [256] 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272
## [273] 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289
## [290] 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306
## [307] 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323
## [324] 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340
## [341] 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357
## [358] 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374
## [375] 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391
## [392] 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408
## [409] 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425
## [426] 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442
## [443] 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459
## [460] 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476
## [477] 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493
```

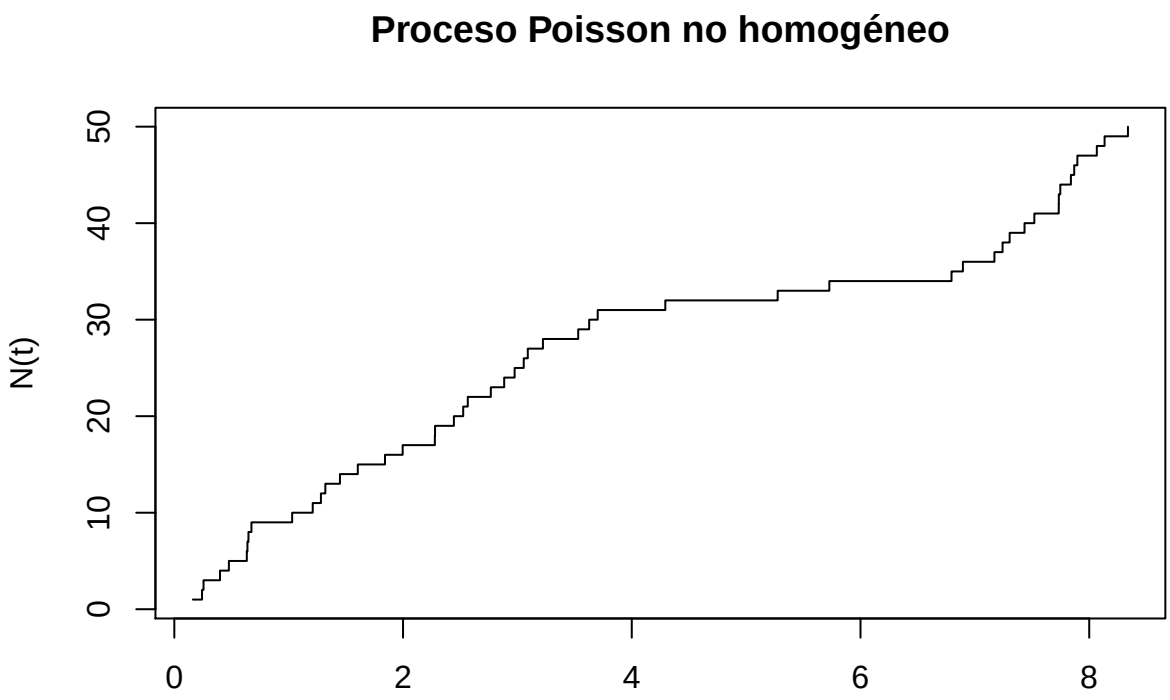
```
## [494] 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510
```

Problema 9

Simular un proceso Poisson no homogéneo con función de intensidad dada por $\lambda(t) = sin(t)$.

Solución:

```
lambdat2<-function(t){sin(t)}

poissonnohomogeneo2<-function(lambdat,n,pic=T){
    lambda<-1
    TT<-rexp(n,lambda)
    s<-cumsum(TT)
    u<-runif(n)
    ss<-s[u<=lambdat(s)/lambda]
    Ns<-1:length(ss)
    if(pic==T){
    plot(ss,Ns,type="s",xlab="",ylab="N(t)",main="Proceso Poisson no homogéneo")
    return(list(ss,cuenta=Ns))
}
}
poissonnohomogeneo(lambdat,50)
```




```
## [[1]]
##  [1] 0.1619432 0.2421254 0.2553496 0.3993594 0.4770218 0.6337419 0.6394498
```

```
##  [8] 0.6482654 0.6742397 1.0304172 1.2102151 1.2819540 1.3204450 1.4482738
## [15] 1.6042737 1.8420865 1.9967739 2.2782071 2.2797223 2.4447135 2.5259537
## [22] 2.5661794 2.7674456 2.8843262 2.9757327 3.0551990 3.0908034 3.2232634
## [29] 3.5316494 3.6274012 3.7021940 4.2931691 5.2759349 5.7275050 6.7959228
## [36] 6.8954653 7.1711062 7.2422088 7.3042084 7.4343457 7.5201655 7.7335606
## [43] 7.7341447 7.7469627 7.8397893 7.8687883 7.8963723 8.0664902 8.1344128
## [50] 8.3387294
##
## $cuenta
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
## [24] 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46
## [47] 47 48 49 50
```

Problema 10

Una compañía de seguros tiene 1000 asegurados, cada uno de los cuales presentará de manera independiente una reclamación en el siguiente mes con probabilidad $p = 0.09245$. Suponiendo que las cantidades de los reclamos hechos son variables aleatorias normales con media 7000 y desviación estándar 5000, hagan simulación para estimar la probabilidad de que la suma de los reclamos exceda $500,000.

Solución:

```
comp_seguros<-function(n_as,p,m,de){
  n_rec<-sum(rbinom(n,1,p))
  montos_rec<-rnorm(n_rec,m,de)
  tot_rec<-sum(montos_rec)
}

registro<-replicate(10000,comp_seguros(1000,0.09245,7000,5000))
p_exceder<-length(subset(registro,registro>500000))/10000
p_exceder
```

```
## [1] 0.9674
```

Problema 11

Escribir una función para generar una mezcla de una distribución normal multivariada con dos componentes con medias μ_1, μ_2 y matrices de covarianzas S_1, S_2 respectivamente. Con el programa, generar una muestra de tamaño $n = 1000$ observaciones de una mezcla 50% de una normal 4-dimensional con $\mu_1 = (0, 0, 0, 0)$, $\mu_2 = (2, 3, 4, 5)$, y matrices de covarianzas $S_1 = S_2 = I_4$. Obtener los histogramas de las 4 distribuciones marginales

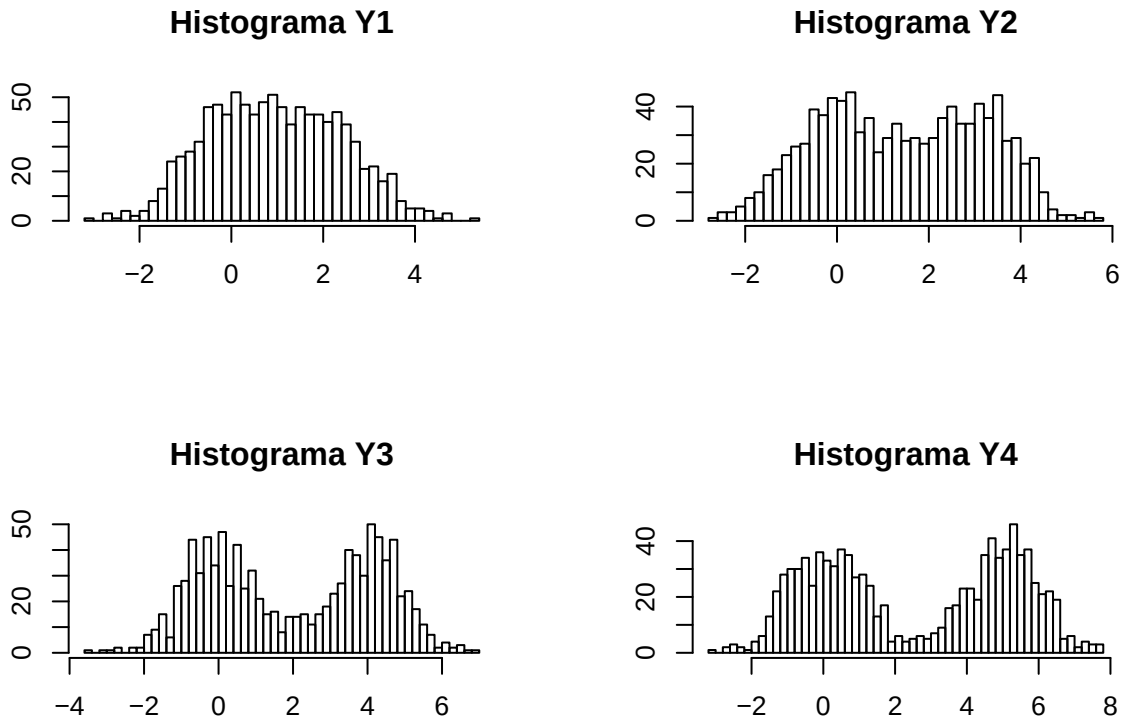
Solución:

```
r_normal_multi <-function(n,mu,Sigma){
d <-length(mu)
S <-svd(Sigma)
Q <- S$u %*%diag(sqrt(S$d)) %*%t(S$v)
Z <-matrix(rnorm(n*d),nrow=n, ncol=d)
X <- Z %*% Q +matrix(mu,n,d,byrow=T)
X
}
```

```r
Sigma <-matrix(c(1, 0, 0,0, 0, 1, 0, 0, 0,0,1,0,0,0,0,1),byrow=T,nrow=4)
n<-1000

Y1<-r_normal_multi(n,c(0,0,0,0),Sigma = Sigma)
Y2<-r_normal_multi(n,c(2,3,4,5),Sigma = Sigma)
u <-runif(n)
k <-as.integer(u > 0.5)
Y <- k*Y1 + (1-k)*Y2

head(Y)
```

```
##            [,1]       [,2]       [,3]        [,4]
## [1,] -0.1128374 -0.4594889  1.1660754  0.46836461
## [2,] -0.4681755  1.5622449  0.4631963 -0.13397857
## [3,] -2.7273527  0.6620804 -0.4567528 -0.84465091
## [4,] -0.3179931  0.1262358 -0.2770529 -1.07472215
## [5,]  1.7431397 -1.4590104  0.8260385 -0.03253928
## [6,] -0.3500403  0.8979119 -1.6641606  0.25992485
```

```r
par(mfrow=c(2,2))

hist(Y[,1],xlab = "",ylab = "",main = "Histograma Y1",breaks = 50)
hist(Y[,2],xlab = "",ylab = "",main = "Histograma Y2",breaks = 50)
hist(Y[,3],xlab = "",ylab = "",main = "Histograma Y3",breaks = 50)
hist(Y[,4],xlab = "",ylab = "",main = "Histograma Y4",breaks = 50)
```



Problema 12

Distribución de Wishart. Suponer que $M = X^T X$, donde X es una matrix de n x d de una muestra aleatoria de una distribución $N_d(\mu, \Sigma)$. Entonces M tiene una distribución Wishart con matriz de escala Σ y n grados de libertad, y se denota $W \sim W_d(\Sigma, n)$. Cuando $d = 1$, los elementos de X son una muestra aleatoria de una $N(\mu, \sigma^2)$, por lo que $W_1(\sigma^2, n) \sim \sigma^2 \chi 2$. Una forma de generar observaciones de una distribución Wishart, es generar muestras de multivariadas normales y calcular la matriz producto XX^T. Programar este método. Noten que este método es muy costoso porque se tienen que generar nd valores aleatorios normales para determinar las $d(d+1)/2$ diferentes entradas de M.

```
Wishart_1<-function(n,mu,s){
  X<-r_normal_multi(n,mu,s)
  B<-X%*%t(X)
  return(B)

}

Wishart_ma<-function(tm,n,mu,s){
  replicate(tm,Wishart_1(n,mu,s))
}

#Ejemplo
ptm<-proc.time()
Wishart_ma(4,4,c(1,1),matrix(c(1, 0, 0,1),byrow=T,nrow=2))
```

```
## , , 1
##
##           [,1]      [,2]      [,3]      [,4]
## [1,] 1.4532518 0.3857352 0.7021119 1.6399845
## [2,] 0.3857352 0.1283497 0.1100653 0.4012509
## [3,] 0.7021119 0.1100653 0.5634042 0.8923789
## [4,] 1.6399845 0.4012509 0.8923789 1.8953608
##
## , , 2
##
##           [,1]       [,2]      [,3]       [,4]
## [1,]  9.146153  2.6834806  5.786012 -2.3243854
## [2,]  2.683481  2.5424148  3.644599 -0.7335258
## [3,]  5.786012  3.6445989  5.820198 -1.5276338
## [4,] -2.324385 -0.7335258 -1.527634  0.5922289
##
## , , 3
##
##            [,1]      [,2]      [,3]      [,4]
## [1,]  0.9794897  2.148378  0.684697 -0.402767
## [2,]  2.1483781  6.413212 -1.903989 -3.241421
## [3,]  0.6846970 -1.903989  7.297606  4.439603
## [4,] -0.4027670 -3.241421  4.439603  3.434327
##
## , , 4
##
##           [,1]       [,2]      [,3]       [,4]
## [1,] 0.5787803  0.7188113 1.3876321  0.6298313
## [2,] 0.7188113 10.3853293 6.2165886 -1.1658717
```

```
## [3,] 1.3876321  6.2165886 5.4536887  0.5879207
## [4,] 0.6298313 -1.1658717 0.5879207  1.0851738
```

```
proc.time()-ptm
```

```
##    user  system elapsed
##       0       0       0
```

Un método más eficiente se basa en la descomposición de Bartlett: sea $T = (T_{ij})$ una matriz triangular inferior de d×d con entradas independientes que satisfacen: $T_{ij} \sim N(0,1)$ independientes para $i > j$, $T_{ii} \sim \sqrt{\chi^2_{n-i+1}}, i = 1, .., d$. Entonces la matrix $A = TT^{'}$ tiene una distribución $W_d(I_d, n)$. Para generar variables $W_d(\Sigma, n)$, obtener la descomposición de Cholesky $\Sigma = LL^{'}$, donde L es triangular inferior. Entonces $LAL^{'} \sim W_d(\Sigma, n)$.

Problema 13

Las ocurrencias de huracanes que tocan tierra durante el fenómeno meteorológico "el Niño" se modelan como un proceso Poisson (ver Bove et al (1998)). Los autores aseguran que "durante un año del Niño, la probabilidad de dos o más huracanes haciendo contacto con tierra en los Estados Unidos es 28 %". Encontrar la tasa del proceso Poisson.

Problema 14

Comenzando a mediodía, los comensales llegan a un restaurante de acuerdo a un proceso Poisson a una tasa de 5 clientes por minuto. El tiempo que cada cliente pasa comiendo en el restaurante tiene una distribución exponencial con media de 40 minutos, independiente de otros clientes e independiente de los tiempos de arribo. Encuentra la distribución así como la media y varianza, del número de comensales en el restaurante a las 2:00pm. Simular el restaurante para verificar los resultados obtenidos.

Problema 15