Tarea III

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

1 October 2018

Problema 1

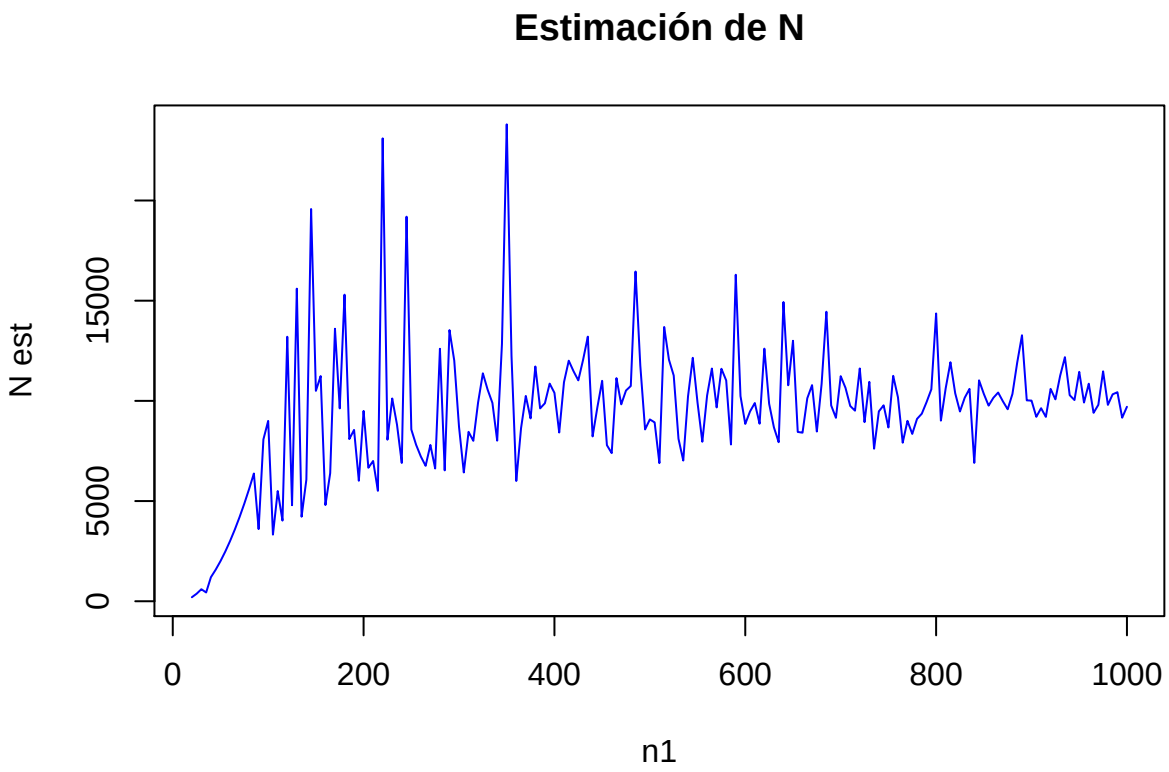
Un estadístico está interesado en el número N de peces en un estanque. Él captura 250 peces, los marca y los regresa al estanque. Unos cuantos días después regresa y atrapa peces hasta que obtiene 50 peces marcados, en ese punto también tiene 124 peces no marcados (la muestra total es de 174 peces).

- ¿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.

1. 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.
2. 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.
3. 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 $S(x) = 1 - F_X(x)$

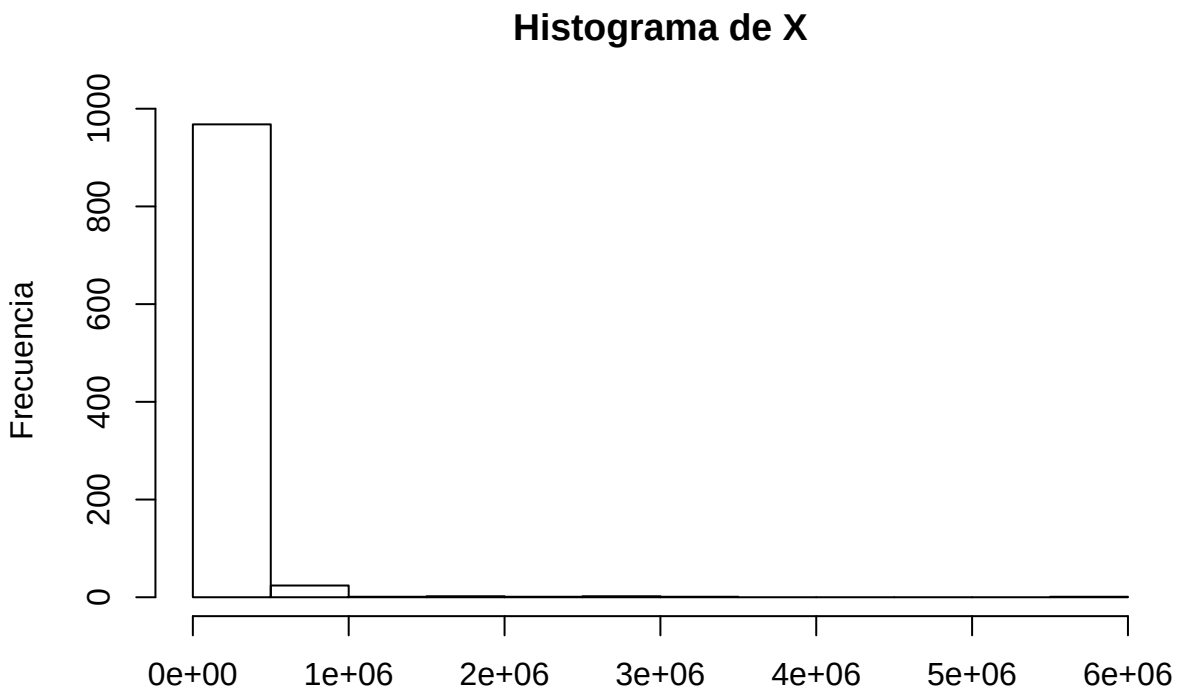
$$E[X] = \int_0^\infty x(1 - F_X(x))dx = \int_0^\infty \frac{\beta^\alpha}{(x+\beta)^\alpha} dx = \frac{\beta}{\alpha - 1}$$

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 12548075 13976788 16543196 19472773 22381244
## 2 1e+07 12214902 14441210 16571760 18617982 21097751
## 3 1e+07 11925108 15085942 17092351 18583744 20916652
## 4 1e+07 12014988 14589950 17216427 18914342 21526591
## 5 1e+07 13451610 15063574 17738627 20161389 22341663
## 6 1e+07 12331325 14905310 17852281 21056026 22885175
```

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 2757089 -5353393    0    0    0
## 2 1e+07 2696072 -5523657    0    0    0
## 3 1e+07 2134274 -5904789    0    0    0
## 4 1e+07 2444245 -5338381    0    0    0
## 5 1e+07 2820087 -4996092    0    0    0
## 6 1e+07 1471575 -6541724    0    0    0
```

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

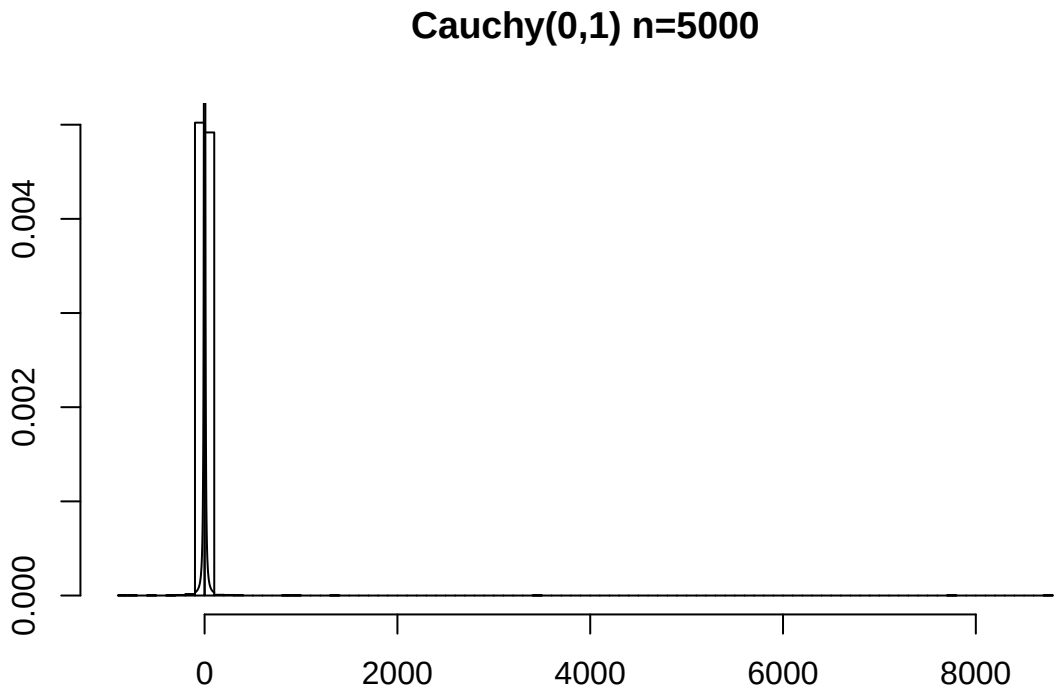
Problema 3

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

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

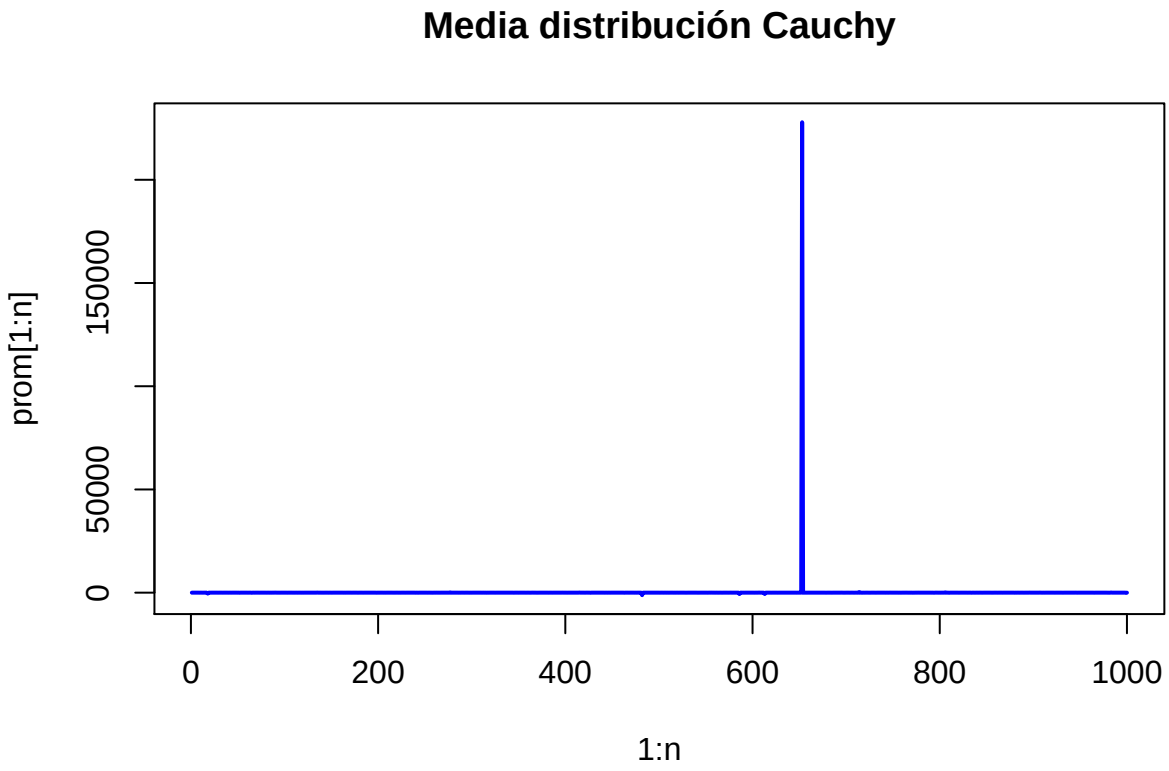
Solución: Podemos encontrar la distribución de X como $F_X(x) = \frac{arctan(\frac{x-\gamma}{\beta})}{\pi} + \frac{1}{2}$, entonces $F^{-1}(u) = tan(\pi(u - \frac{1}{2}))$. Usamos el método de la transformada inversa

```
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()
```

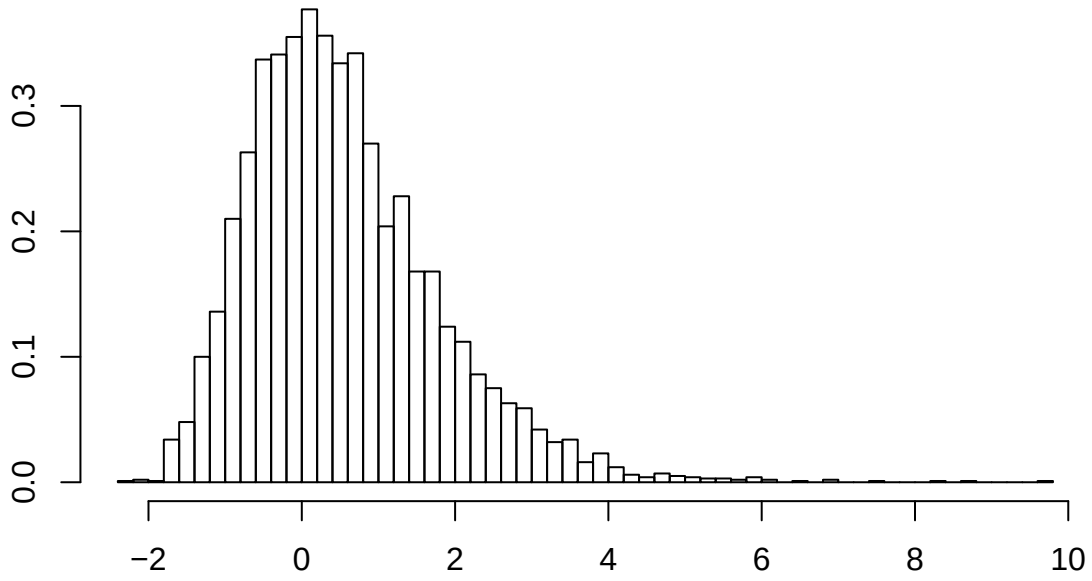
```r
for (i in 1:n) {
  u<-runif(1000)
  x<-tan(pi*(u-0.5))
  prom[i]<-mean(x)
}
plot(1:n,prom[1:n],type="l",lwd=2,col="blue",main = "Media distribución Cauchy")
```



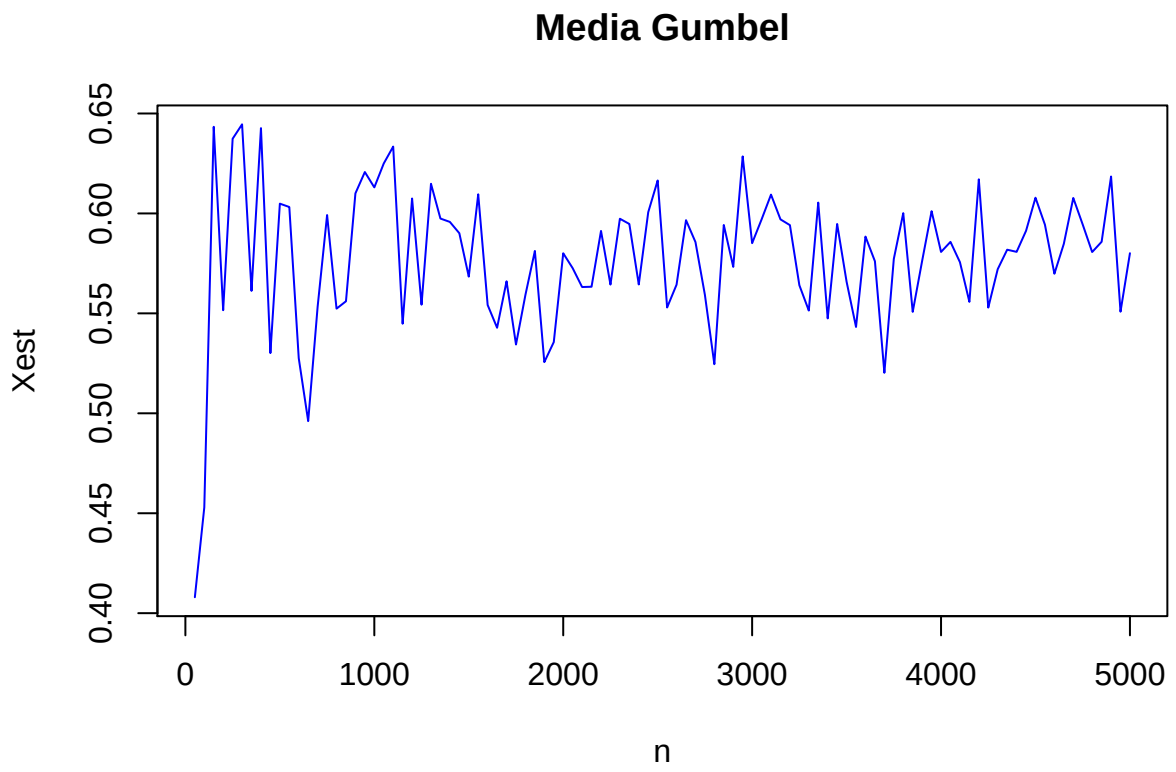
*Gumbel $f(x) = \frac{1}{\beta} exp\left[-e^{-\frac{(x-\gamma)}{\beta}} - \frac{x-\gamma}{\beta}\right]; \gamma, x \in \mathbb{R}; \beta > 0.$

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
```



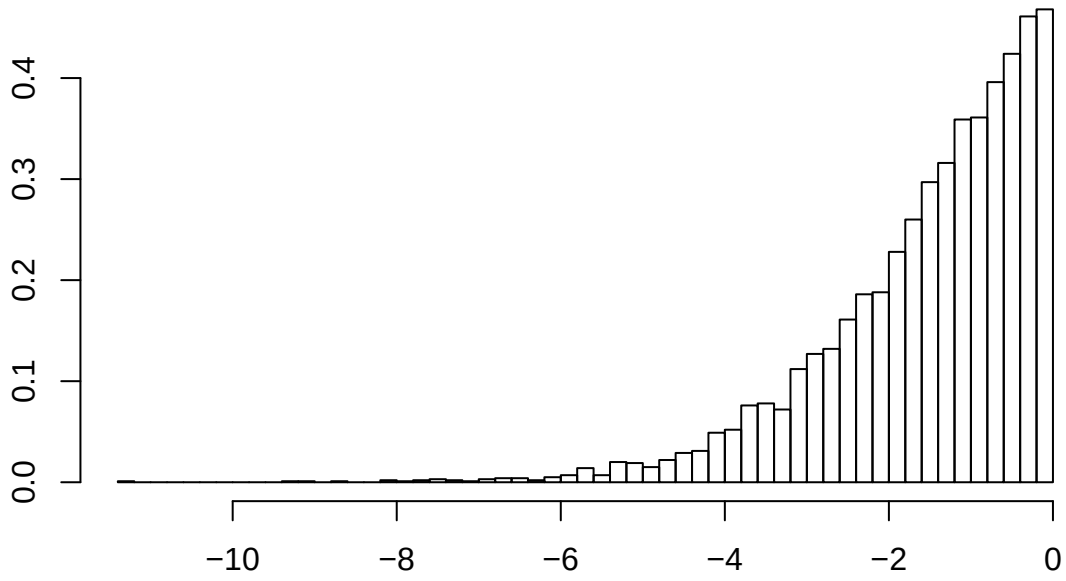
*Logística

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

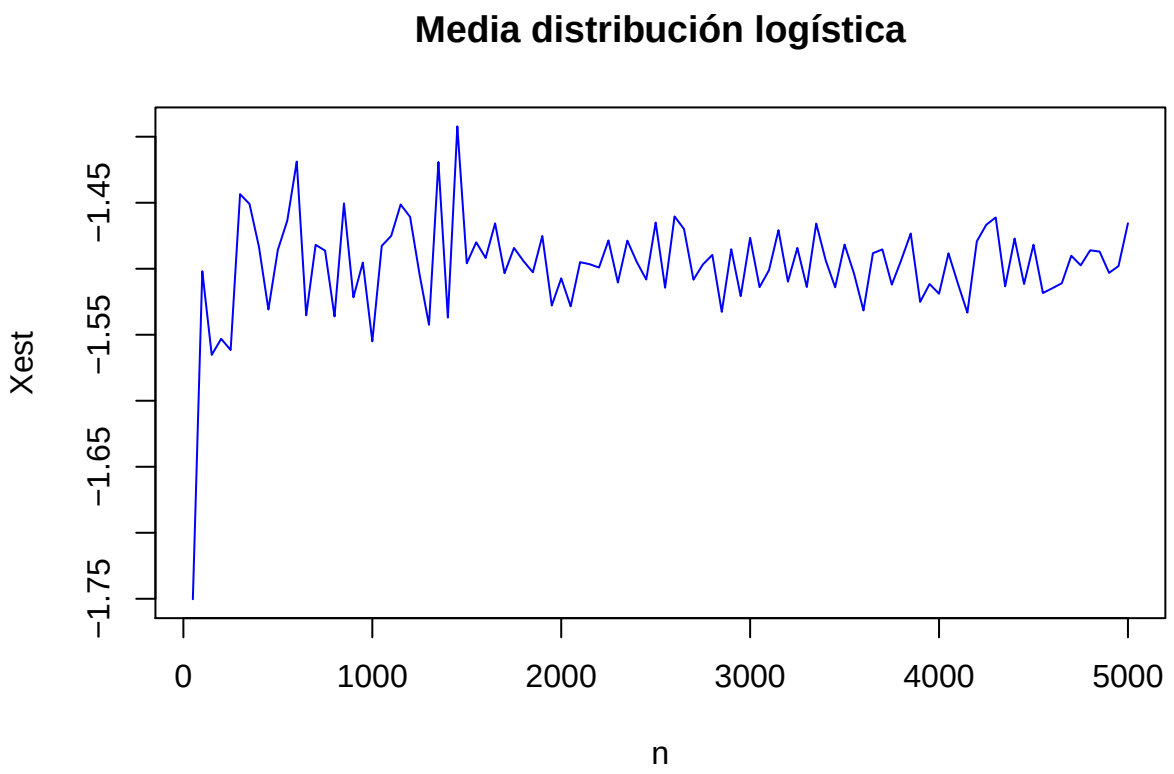
Solución:

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 =
```




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

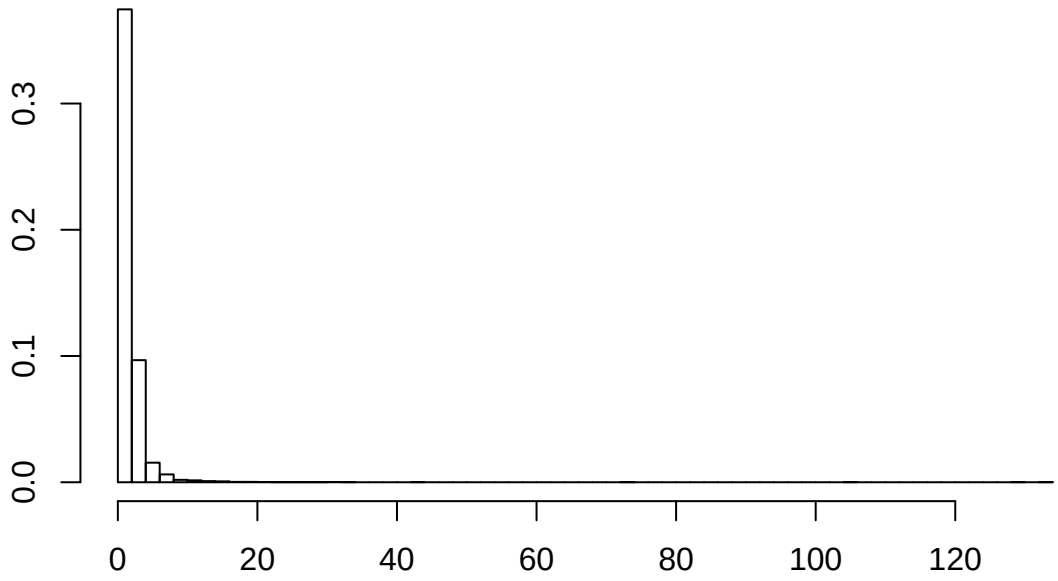
Solución:

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

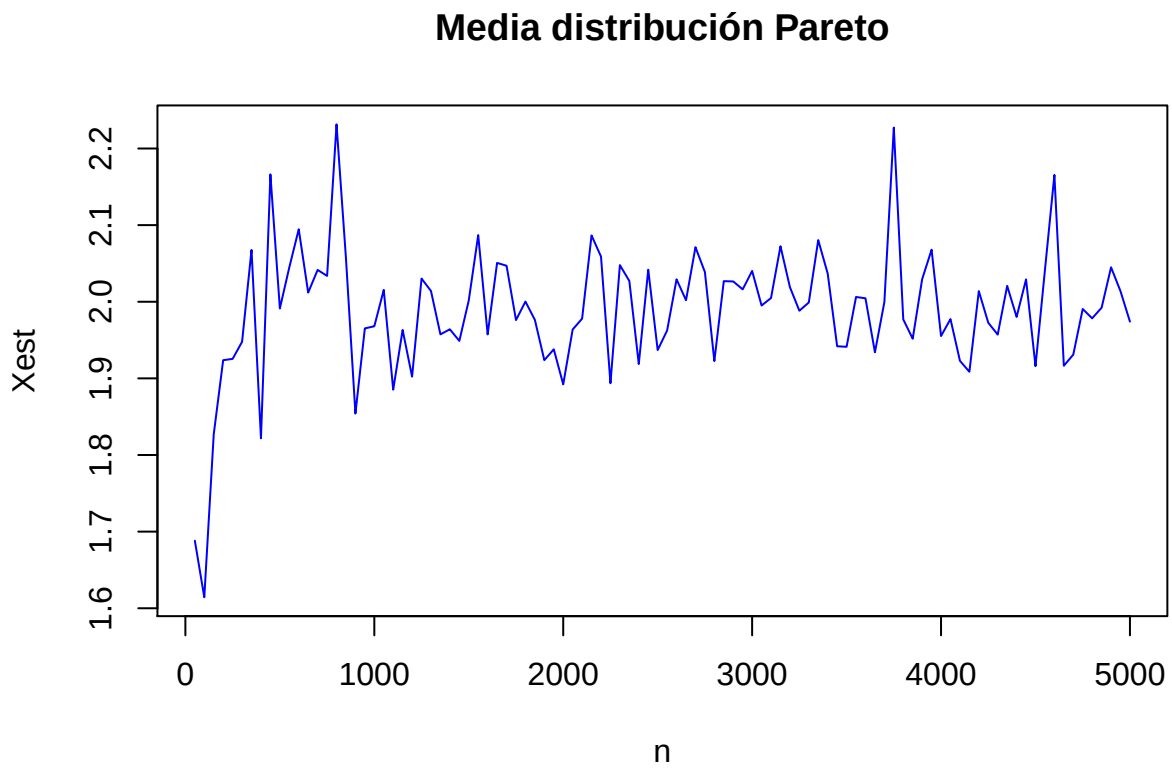
.

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

```
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.

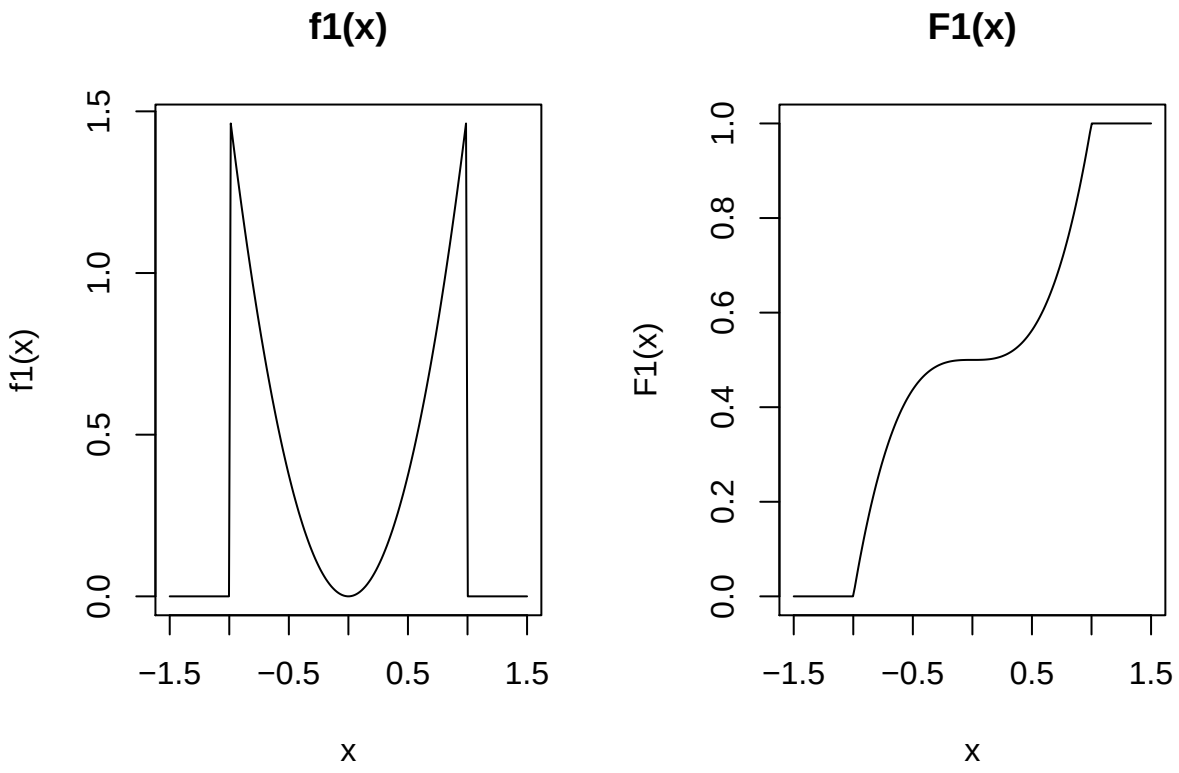
$$f(x) = \frac{3x^2}{2} I(x)_{[-1,1]}$$

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

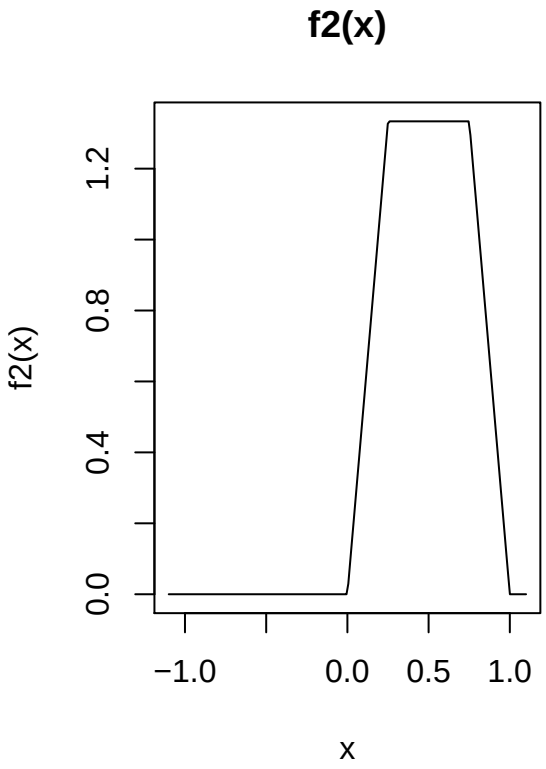
Solución:

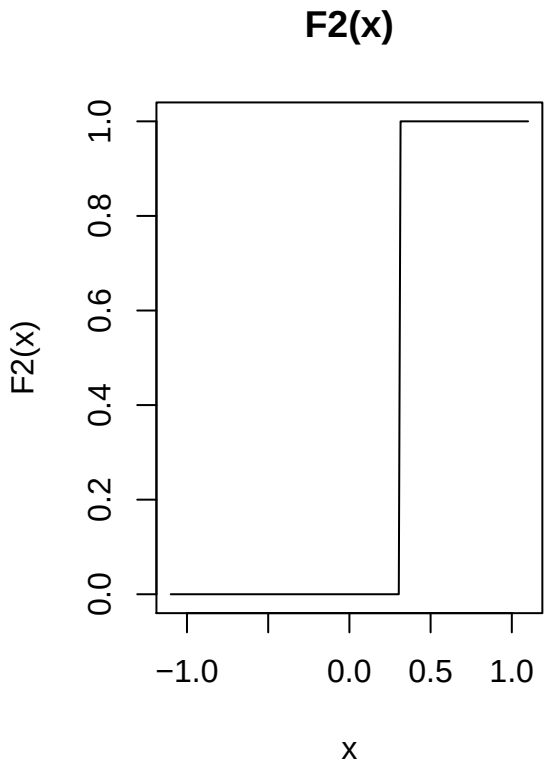
```
ind<-function(x,a,b){
ifelse(x<=b & x>= a,1,0)
}

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 $k = 100$.

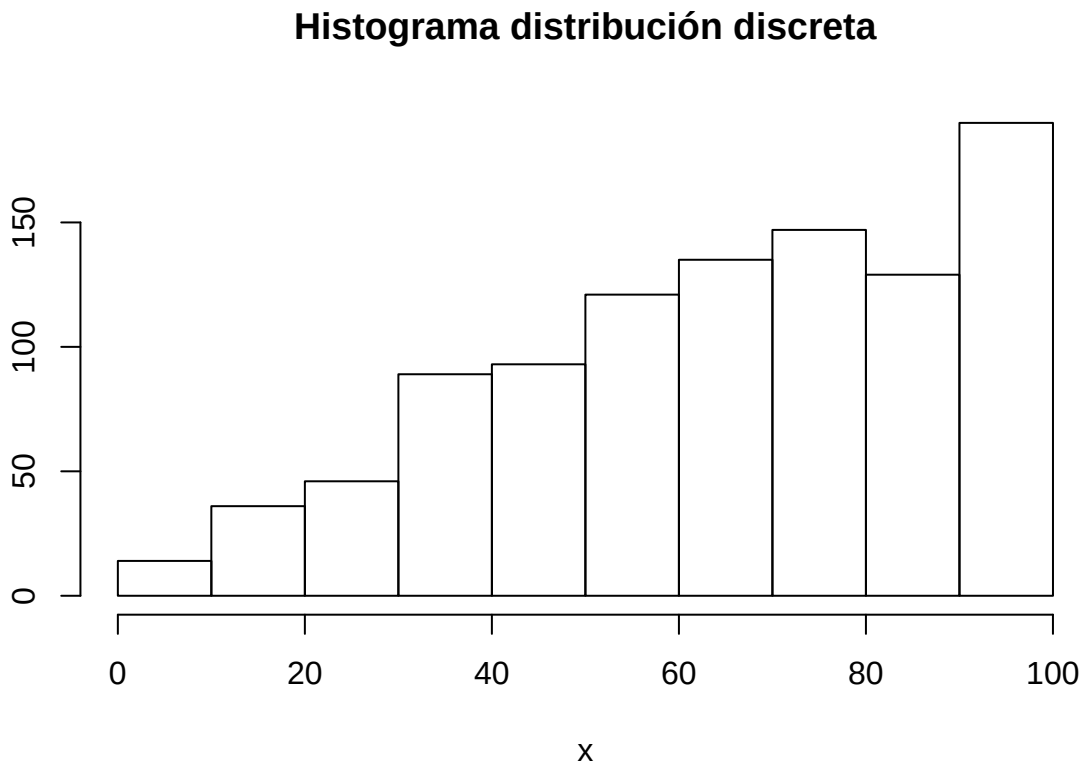
$$p(x) = \frac{2x}{k(k+1)}; x = 1, 2, ...k$$

Solución:

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

```
##  [1] 100  83  43  40  93 100  79  73  83  52  60  98  50  40  50  31  92
## [18]  92  14  88  38  24  85  94  97  60 100  82  86  58  67  31 100  93
## [35]  60  98 100  92  82  62  82  68  80  65 100  98  33 100  74  72
```

```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    0.02
```

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

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

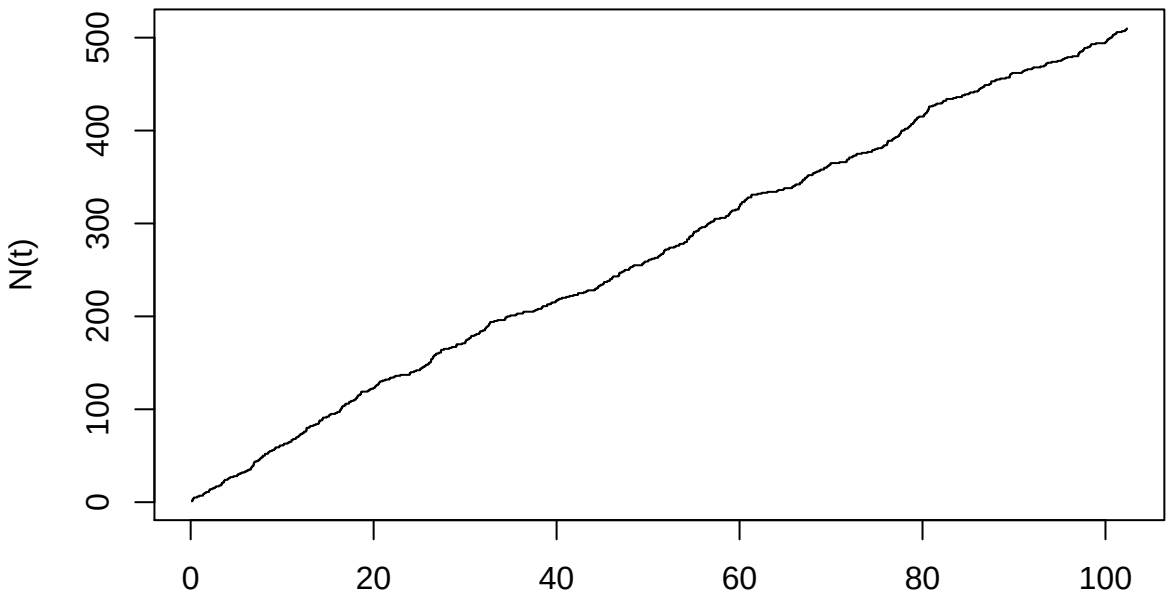
Resulta más eficiente realizar una muestra de 100000 números con la función rbinom.

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]   0.1280520   0.1909469   0.2018692   0.2886050   0.3527976
##   [6]   0.6972334   0.9343731   1.3491755   1.3882512   1.5267786
##  [11]   1.7026017   2.0131258   2.0188787   2.1071274   2.3886846
##  [16]   2.6219660   2.7626154   3.1184214   3.3461138   3.3760219
##  [21]   3.4429080   3.6033026   3.6444879   3.7268862   4.0303768
##  [26]   4.1577946   4.3183193   4.6782262   5.0762049   5.1485966
##  [31]   5.3604027   5.5803791   5.9079134   6.1160567   6.3142143
##  [36]   6.5574760   6.5861310   6.6740659   6.7421442   6.8661431
##  [41]   6.8812560   6.9025767   6.9245331   7.0680662   7.3296229
##  [46]   7.4865644   7.5885928   7.6896516   7.8587671   7.8809484
##  [51]   8.1045560   8.1282745   8.4422876   8.5068105   8.6346657
##  [56]   8.8898060   9.1109734   9.2381594   9.2749087   9.6819525
##  [61]   9.7422137  10.1303134  10.2184650  10.6005279  10.7851668
##  [66]  11.0280500  11.0294486  11.1974730  11.4864799  11.5576631
##  [71]  11.7486928  11.8749495  11.9519603  12.1379569  12.3223888
##  [76]  12.4109081  12.6326941  12.6562538  12.6642385  12.6716976
##  [81]  12.9573441  13.0676400  13.4098783  13.6477520  13.9583168
##  [86]  14.0234982  14.0738265  14.0832463  14.3750912  14.4154878
##  [91]  14.4610227  14.8515033  15.1037292  15.1597050  15.3089607
##  [96]  15.8142284  16.0195002  16.2917966  16.3135627  16.4060348
## [101]  16.4657119  16.5144311  16.5616343  16.7178627  16.9189601
## [106]  16.9213000  17.3219261  17.3271457  17.5413295  17.8239641
## [111]  18.0077561  18.0582275  18.2108466  18.2154198  18.2855076
## [116]  18.4790534  18.5937588  18.6332882  18.6556256  19.3588817
## [121]  19.5238455  19.6521607  19.9622942  20.1323625  20.1929366
```

```
## [126]  20.3064969  20.4617435  20.5663910  20.6146392  20.6977192
## [131]  20.9725404  21.2814157  21.7192945  21.8206376  22.1959365
## [136]  22.3864260  22.9193168  23.9383233  23.9545371  24.0634560
## [141]  24.4081983  24.6484019  25.0764038  25.1237362  25.3047902
## [146]  25.4479303  25.6288436  25.7464225  26.0029393  26.0273366
## [151]  26.1985436  26.2692002  26.2911804  26.2991550  26.4306472
## [156]  26.5052715  26.5263524  26.6003811  26.7776113  26.8451315
## [161]  27.1320819  27.3687802  27.3713284  27.3786130  27.6900640
## [166]  28.2649095  28.5391911  29.0173126  29.0616521  29.0717444
## [171]  29.6745789  30.0177735  30.0277551  30.0984217  30.1621517
## [176]  30.3907296  30.4898965  30.6478508  30.6745917  31.0357585
## [181]  31.2322692  31.5271323  31.6035701  31.6562604  31.9280608
## [186]  32.1708551  32.1878656  32.2799917  32.3589696  32.4912019
## [191]  32.6070173  32.6289198  32.7080573  32.7492372  33.1744477
## [196]  33.5329945  34.3471726  34.4144675  34.4164050  34.6145903
## [201]  34.9268027  35.5310296  35.6902531  36.3074221  36.3924322
## [206]  37.5010062  37.7004841  37.9291091  38.3379086  38.3974002
## [211]  38.5000106  38.9723726  38.9792315  39.3527280  39.4712931
## [216]  39.8670913  39.9562924  40.1048564  40.2997508  40.6021705
## [221]  41.0546981  41.4106177  41.8510100  42.3357010  42.3381400
## [226]  42.9537777  43.2249013  43.3847074  44.1668193  44.2649808
## [231]  44.4874104  44.5324880  44.6530667  44.8178544  45.0623517
## [236]  45.1688482  45.1871707  45.5321854  45.8008854  45.8144220
## [241]  46.0411474  46.0890733  46.1826277  46.7712661  46.8288990
## [246]  46.8639178  46.8715279  47.1364257  47.3381485  47.4763582
## [251]  47.9304531  47.9746846  48.0554283  48.2995199  48.4358465
## [256]  49.3627461  49.3820738  49.5548392  49.6436427  49.9941435
## [261]  50.0942861  50.3842954  50.7248321  51.0891882  51.1516759
## [266]  51.3071666  51.4235651  51.6452067  51.6966591  51.7398145
## [271]  51.7794947  51.8996110  52.2457886  52.3462727  52.8662579
## [276]  53.0534261  53.4012727  53.4340198  53.9020016  54.0103796
## [281]  54.2480229  54.2578996  54.2679492  54.4712324  54.5139952
## [286]  54.5345046  54.7489235  54.9141990  54.9154537  54.9854461
## [291]  55.0097511  55.2773114  55.5166191  55.5373342  55.6449041
## [296]  55.8382418  56.2481031  56.3908016  56.4587499  56.5709993
## [301]  56.7405586  56.9116677  57.1670995  57.2539073  57.2813621
## [306]  57.8800489  58.4727863  58.5153685  58.7850066  58.8246368
## [311]  58.9330130  58.9757796  59.0805129  59.1975602  59.5560414
## [316]  59.8932023  59.9049681  59.9413303  60.0224449  60.0703212
## [321]  60.0914452  60.2435199  60.2443976  60.5333570  60.5536027
## [326]  60.6936097  60.8131112  60.9108123  61.2844361  61.2868254
## [331]  61.3036264  61.9635232  62.4166117  63.0424104  64.0819858
## [336]  64.2308019  64.7873308  64.8839302  65.7329613  65.8088633
## [341]  66.0200505  66.1386693  66.5723584  66.6289431  66.8227071
## [346]  66.8777615  66.9118167  67.1265731  67.1355034  67.3159328
## [351]  67.4017351  67.5056031  67.9793572  68.0065820  68.2144720
## [356]  68.4509148  68.7278734  68.8568274  69.2310945  69.2615850
## [361]  69.5308375  69.6397721  69.7801382  69.9245624  69.9634514
## [366]  70.9251441  71.6979269  71.6993437  71.7790529  71.9844713
## [371]  71.9847270  72.3157340  72.4622009  72.7558676  72.8141824
## [376]  73.3664555  73.9899899  74.4688200  74.5265456  74.7884153
## [381]  75.0739288  75.5450883  75.7051703  75.7580885  76.0679917
## [386]  76.1455158  76.1593335  76.1720081  76.2424580  76.6934247
## [391]  76.7374556  76.9010961  77.0465797  77.2803286  77.4386946
```

```
## [396]  77.4857983  77.5598838  77.6321729  77.6494926  77.7238287
## [401]  77.9930538  78.1071137  78.4033612  78.4814226  78.6001619
## [406]  78.7204093  78.7579486  78.9508274  79.0317394  79.1154711
## [411]  79.1431364  79.2494753  79.3897981  79.4285390  79.6036003
## [416]  80.1107541  80.1874130  80.2368399  80.3518334  80.4838176
## [421]  80.5343593  80.5788630  80.6569045  80.6758902  80.6993679
## [426]  80.7475944  81.1023015  81.3923442  81.5478761  82.0806630
## [431]  82.2172561  82.2915251  82.5999403  82.6008907  83.3481600
## [436]  83.6791982  84.3044581  84.3493686  84.6304612  85.0405132
## [441]  85.1910415  85.6492415  86.0360071  86.1698720  86.2185266
## [446]  86.3745678  86.5782824  86.7522457  86.7909593  87.2811047
## [451]  87.4651826  87.5081669  87.5134537  87.9022302  88.0872580
## [456]  88.4953041  89.1094582  89.5216073  89.5344038  89.5725731
## [461]  89.6325409  89.8002232  90.8693706  90.9784924  91.1637314
## [466]  91.4348227  91.9639655  92.1148139  92.8925406  93.2828679
## [471]  93.4717328  93.4877335  93.6596891  94.1193423  94.7480365
## [476]  95.1895095  95.3423378  95.5300363  95.7966830  96.3749124
## [481]  97.0275266  97.0359420  97.0548227  97.1379662  97.2387708
## [486]  97.4826048  97.5660432  97.6540385  97.7107087  98.0772662
## [491]  98.3277722  98.3745992  98.4360295  98.9364734  99.9687850
## [496] 100.0874964 100.1623324 100.2242147 100.3420828 100.5168798
## [501] 100.6949656 100.7757178 100.8091187 101.0038754 101.1679643
## [506] 101.2319362 101.7768704 102.1206653 102.2339265 102.3467370
##
## $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
```

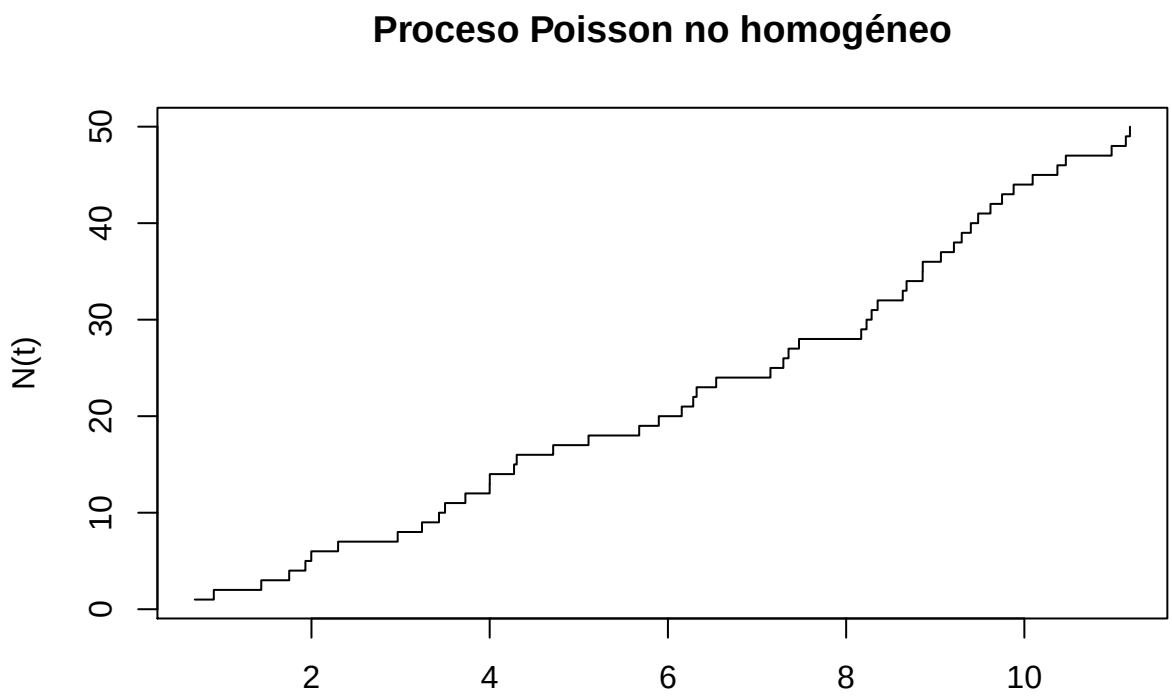
```
## [494] 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510
```

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]  0.6914081  0.9043443  1.4365278  1.7501939  1.9322091  1.9980260
```

```
##  [7]  2.2989980  2.9676263  3.2405007  3.4310868  3.4988366  3.7273121
## [13]  3.9999892  4.0010201  4.2733070  4.3031741  4.7122039  5.1088951
## [19]  5.6775954  5.8976370  6.1550852  6.2839354  6.3220672  6.5416872
## [25]  7.1505613  7.2952648  7.3534611  7.4709848  8.1688177  8.2291042
## [31]  8.2857901  8.3532873  8.6349998  8.6777249  8.8582651  8.8594023
## [37]  9.0631257  9.2091688  9.2969123  9.3990009  9.4814145  9.6195564
## [43]  9.7494766  9.8793862 10.0930023 10.3707470 10.4651960 10.9788685
## [49] 11.1383356 11.1846357
##
## $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] 0.9738
```

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

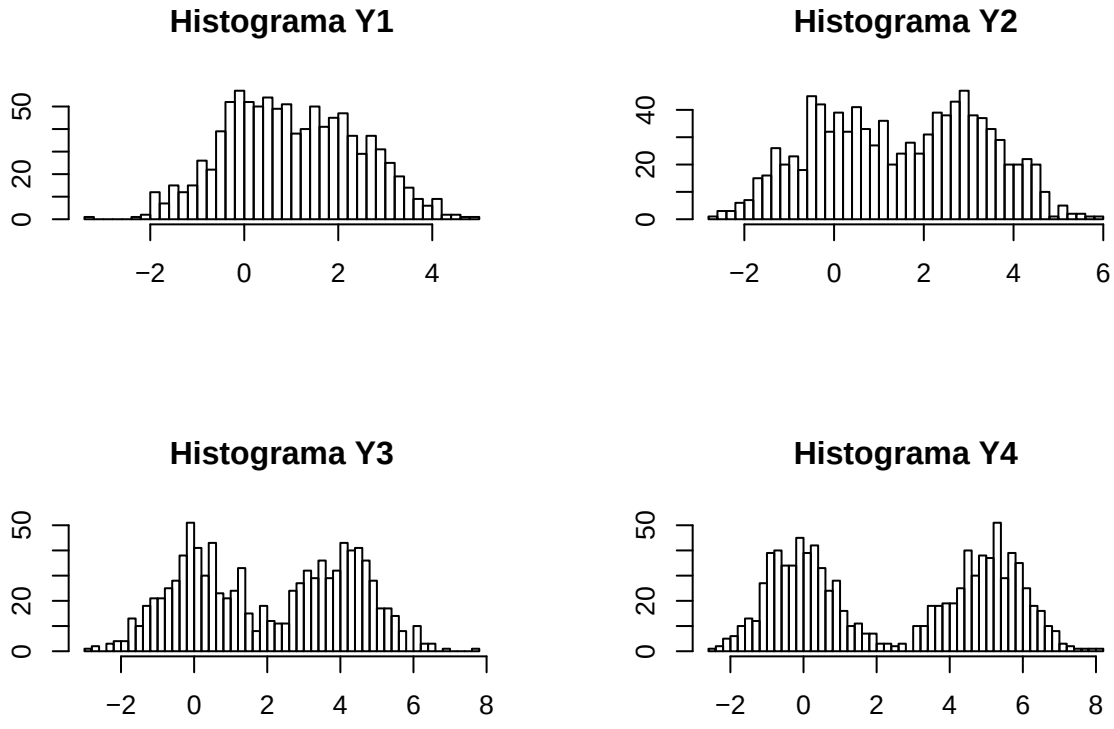
Solución:

```
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,]  3.2830735  3.5285087  4.9023304  4.0769762
## [2,]  1.6699890  3.9726949  2.9523079  6.1592260
## [3,] -0.1819490  0.3476841  1.5646367  0.2669568
## [4,] -1.4745549 -2.0806043 -0.9958002 -0.3625205
## [5,]  1.1117781  1.1605891 -1.0435153  0.8648946
## [6,]  0.6607643 -0.5988266  1.2011336  0.4163359
```

```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,] 6.671196 4.719887 2.686153 3.098848
## [2,] 4.719887 4.008668 1.767625 1.897355
## [3,] 2.686153 1.767625 1.107940 1.306311
## [4,] 3.098848 1.897355 1.306311 1.569544
##
## , , 2
##
##          [,1]     [,2]     [,3]     [,4]
## [1,] 1.078563 1.081820 1.670692 0.843163
## [2,] 1.081820 1.287782 2.288013 2.052639
## [3,] 1.670692 2.288013 4.437388 4.951810
## [4,] 0.843163 2.052639 4.951810 7.845726
##
## , , 3
##
##           [,1]       [,2]      [,3]       [,4]
## [1,] 10.961094  2.7814535 1.1524633 -1.8951714
## [2,]  2.781454  0.7062458 0.2849108 -0.5065118
## [3,]  1.152463  0.2849108 0.2523740  0.2464972
## [4,] -1.895171 -0.5065118 0.2464972  1.8421278
##
## , , 4
##
##          [,1]     [,2]     [,3]     [,4]
## [1,] 4.735999 4.519735 6.117271 3.704968
## [2,] 4.519735 4.438275 5.392535 3.255312
```

```
## [3,] 6.117271 5.392535 9.489329 5.785478
## [4,] 3.704968 3.255312 5.785478 3.528070
```

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

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

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.

Solución:

Sea N el número de huracánes que tocan tierra en Estados Unidos, de manera que N tiene una distribución $Po(\lambda)$. Se sabe que $Pr\{N \geq 2\} = 0.28$, entonces

$$\begin{aligned} 0.28 &= Pr\{N \geq 2\} \\ &= 1 - Pr\{N < 2\} \\ &= 1 - Pr(N = 0) - Pr(N = 1) \\ &= 1 - \frac{\lambda^0 e^{-\lambda}}{0!} - \frac{\lambda^1 e^{-\lambda}}{1!}. \end{aligned}$$

Resolviendo para λ se tiene que $\lambda = 1.042284919$.

■

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