Simulación ITAM
12 de septiembre de 2021 2021-II

Tarea 2.

La fecha de entrega es el lunes **10 de septiembre de 2021**.

Lecturas

- Casella y Robert, leer la sección 1.5 y el capítulo 2.
- Fast Generation of Discrete Random Variables Artículo de *Journal of Statistical Software, July 2004, Volume 11, Issue 3*
- Introduction to simulation using R Capítulo 13
- Marc C. Bove, et. al Effect of El Niño on U.S. Landfalling Hurricanes, Revisited

Problemas

1. **Problema de captura-recaptura**. Este tipo de problemas son aplicados en multitud de contextos. Un estadístico está interesado en el número N de peces que hay en un estanque. Para estimar su valor utiliza el siguiente procedimiento:

   a) Captura 250 peces, los marca y los regresa al estanque.

   b) Unos cuantos días después regresa y atrapa suficientes peces hasta que obtiene 50 peces marcados, en ese punto también tiene 124 peces no marcados, los que tuvo que cpturar para obtener los 50 peces (la muestra total es de 174 peces).

   - ¿Cuál es la estimación de N?
   - Hagan 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.

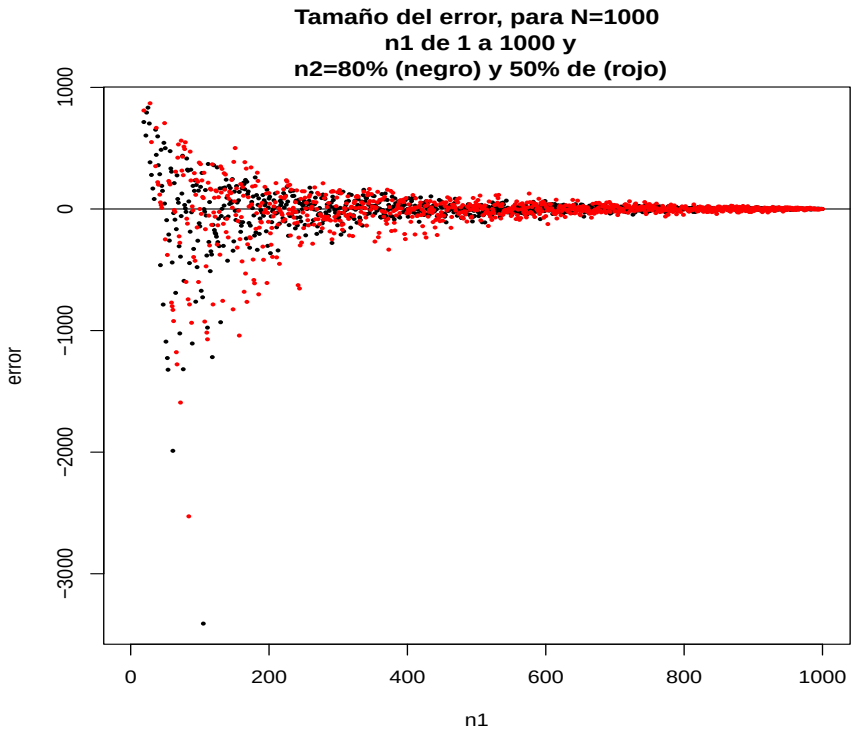
   Solución.

De acuerdo a lo que vimos en clase, la primera captura es $n_1 = 250$ y la segunda muestra es $n_2 = 174$ y los marcados son $r = 50$. La estimación de N es entonces $N = \frac{n_1 n_2}{r} = 870$.

Ahora podemos hacer un ejercicio de simulación tomando un valor fijo de N. supongamos que n_1 es el tamaño de la primera muestra y n_2 el de la segunda muestra. Estos tres valores son dados, pero el $r =$ número de peces marcados en la segunda muestra es una variable aleatoria que sigue una distribución hipergeométrica, con parámetros (N, n_1, n_2). Sin embargo, en el ejercicio de simulación que se propone, debemos estimar N a partir del valor que obtengamos de r.

La siguiente función calcula el valor estimado a partir de realizar las muestras. A partir de esta función se pueden hacer curvas de nivel para analizar el comportamiento del error de acuerdo a los valores de n_1 y n_2.

```
CapRecap <- function(N,n1,n2){
marca1 <- sample(1:N,n1) #obten primera muestra
marca2 <- sample(1:N,n2) #obten segunda muestra
r <- length(intersect(marca1,marca2)) #recapturados
Nhat <- n1*n2/r
return(list(N=N,Nhat=Nhat,error=N-Nhat))
}
N <- 1000 #Consideremos un ejemplo,
error <- NULL
for(i in 1:N) error[i] <- CapRecap(N,i,round(i*0.8,0))$error
plot(error,pch=16,cex=0.5,
main = "Tamaño del error, para N=1000\n n1 de 1 a 1000 y\n n2=80% (negro) y 50% de (rojo)",
xlab="n1")
abline(h=0)
for(i in 1:N) error[i] <- CapRecap(N,i,round(i*0.5,0))$error
points(error,pch=16,cex=0.5,col="red")
```



□

2. Este problema es una versión simplificada de dos problemas comunes a los que se enfrentan las compañías de seguros: calcular la probabilidad de quebrar y estimar cuánto dinero podrán hacer. Los parámetros se dan en conjuntos, se pide simular combinaciones de los diferentes parámetros (probabilidad de reclamo, horizonte de tiempo).

   Supongan que una compañía de seguros tiene activos por \$ 10^7 de pesos. Tienen $n = 1,000$ clientes que pagan individualmente una prima anual de \$5,500 al principio de cada año. Basándose en experiencia previa, se estima que la probabilidad de que un cliente haga un reclamo es $p \in \{0.01, 0.1, 0.15, 0.2\}$ por año, independientemente de reclamos previos de otros clientes. El tamaño X de los reclamos varía, y tiene la siguiente densidad con $\alpha = 5$ y $\beta = 125,000$:

$$f(x) = I(x \geq 0)\frac{\alpha\beta^\alpha}{(x+\beta)^{\alpha+1}}$$

   (Tal X se dice que tiene una distribución *Pareto*, y en el mundo real no es un modelo infrecuente para el tamaño de los reclamos a las aseguradoras).

   Suponemos las fortunas de la compañía aseguradora sobre un horizonte de $T \in \{5, 10, 20\}$ años. Sea $Z(t)$ los activos de la compañía al final del año t, así que $Z(0) = 10,000,000$, y $Z(t) = I_{Z(t-1)>0}\,\text{máx}\{Z(t-1) + \text{primas} - \text{reclamos}, 0\}$

   Noten que si $Z(t)$ cae bajo 0 entonces la compañía quiebra y se queda ahí: si la compañía se va a bancarrota, deja de operar.

   - Calcular la función de distribución F_X, $E[X]$, y $Var[X]$. Obtengan por simulación una muestra de X y su función de distribución, y comparen su estimado con la verdadera.
   - Escriban una función para simular los activos de la compañía en el horizonte de tiempo T y estimen: (1) la probabilidad de que la compañía se vaya a bancarrota, y (2) Los activos esperados al final de cinco años.
   - Toma de ganancias. Supongan ahora que la compañía toma ganancias al final de cada año. Esto es, si $Z(t) > 10,000,000$ entonces $Z(t) - 10,000,000$ se le paga a los accionistas. Si $Z(t) \leq 10,000,000$ entonces los accionistas no reciben nada ese año. Usando este nuevo esquema, estimar: (1) la probabilidad de irse a la quiebra, (2) los activos esperados después de T años, y (3) Las ganancias totales esperadas después de T años.

Solución.

El primer ejercicio es un ejercicio estándar en probabilidad. Me concentraré en los últimos dos incisos.

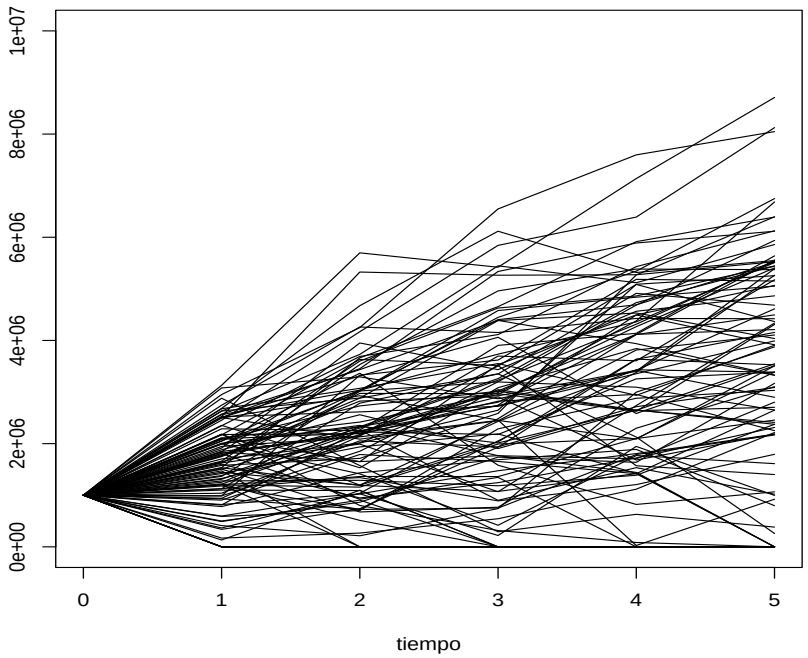
A continuación construyo la función que simula el proceso estocástico. Consideramos al evento de que un cliente haga un reclamo como una variable Bernoulli con probabilidad $p = 0.1$, y como son independientes, podemos suponer que el número de reclamos anual es una variable binomial. Por lo tanto, este proceso tiene dos tipos de variación: una para saber cuántos clientes hacen reclamos, y otra para determinar el tamaño de la reclamación.

```
library(actuar) #para muestrear de una distribución Pareto
```

```
Attaching package: 'actuar'
```

```
The following object is masked from 'package:grDevices':

    cm
```

```
Proceso <- function(ActIni = 1e6, clientes = 1000, t = 5, prima = 5500){
Z <- NULL
Z0 <- ActIni
for(i in 1:t){
reclamos <- sum(rbinom(clientes,1,0.1)) #clientes que generan reclamos, binomial
Z[i] <- max(ifelse(i==1,Z0,Z[i-1]) + clientes*prima - sum(rpareto(reclamos,3,100000)),0)
if(Z[i]==0){Z[i:t] <-0;break} #verifica si se alcanzó 0, y todos los consecutivos 0.
}
return(c(Z0,Z))
}
y <- NULL
plot(0:5,Proceso(),type="l",ylim=c(0,10e6),xlab="tiempo",ylab="")
for(i in 1:100){y[[i]] <- Proceso();lines(0:5,y[[i]])}
```



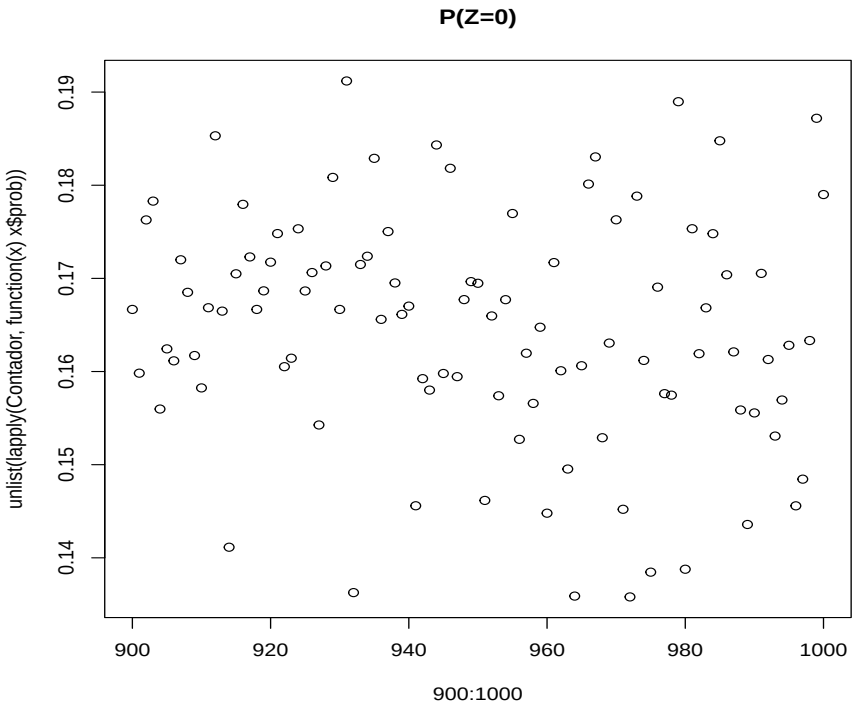

```
Trayectorias <- matrix(unlist(y),ncol=6,byrow=T)
head(Trayectorias)
```

```
     [,1]    [,2]      [,3]      [,4]      [,5]      [,6]
```

```
[1,] 1e+06 1507177  675558.8  768080.9 2298553.8 3104118.2
[2,] 1e+06 2682959 4673766.5 6116850.5 5316751.0 5548238.6
[3,] 1e+06 2048758 3363520.3 1581688.9  821576.2 1065180.9
[4,] 1e+06 2880044 1574350.5  291727.0  629886.4  382238.7
[5,] 1e+06 2519894 2615310.0 2722713.8 3355567.0 3991707.9
[6,] 1e+06 2080722 2731179.2 3015534.0 2639042.8 2704431.3
```

Para estimar la probabilidad de que la compañía se vaya a la bancarrota, podemos contar el número de trayectorias en donde la serie toca 0, para un número N de simulaciones. También se calculan los activos esperados después de 5 años. Como se puede apreciar, este proceso no se estabiliza incrementando el número de trayectorias simuladas.

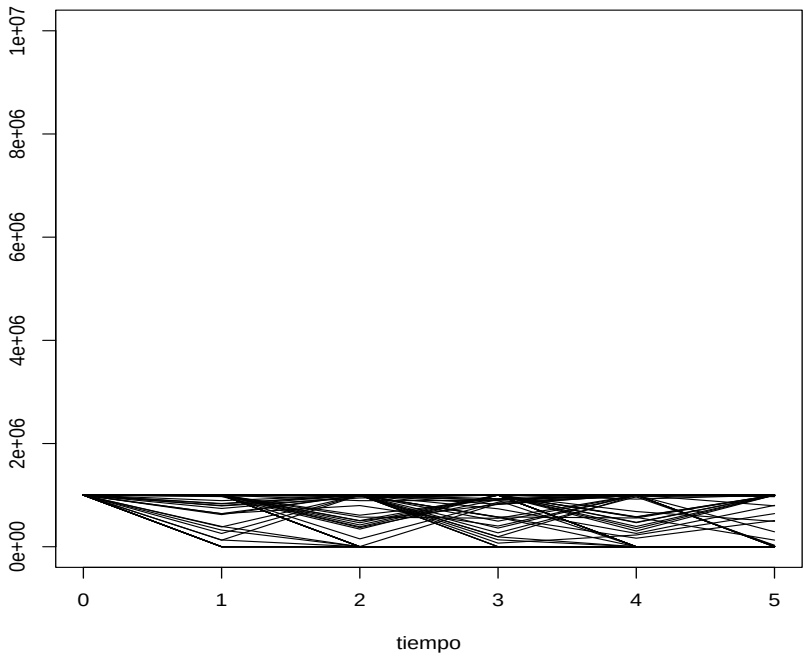
```
simula <- function(N){
Tr <- NULL
for(i in 1:N)Tr[[i]] <- Proceso()
Tr <- matrix(unlist(Tr),ncol=6,byrow = T)
prob <- sum(apply(Tr,1,function(x)0 %in% x))/N
acti <- mean(Tr[,6])
return(list(prob=prob,activos=acti))
}
Contador <- list(NULL)
for(n in 900:1000) Contador[[n-899]] <- simula(n)
plot(900:1000,unlist(lapply(Contador,function(x)x$prob)),main="P(Z=0)")
```




```
mean((unlist(lapply(Contador,function(x)x$acti)))) #Activos promedio después de 5 años


[1] 3378947
```

Ahora modificamos el ejercicio para toma de ganancias.

```r
#Considera ahora la toma de ganancias.
Proceso2 <- function(ActIni=1e6,clientes=1000,t=5,prima=5500){
Z <- NULL
G <- NULL #Ganancias
Z0 <- ActIni
for(i in 1:t){
reclamos <- sum(rbinom(clientes,1,0.1)) #clientes que generan reclamos, binomial
Z[i] <- max(ifelse(i==1,Z0,Z[i-1]) + clientes*prima - sum(rpareto(reclamos,3,100000)),0)
if(Z[i]==0){Z[i:t] <-0;break} #verifica si se alcanzó 0, y todos los consecutivos 0.
if(Z[i]>=1e6){G[i] <- Z[i]-1e6;Z[i] <- 1e6} # Toma de ganancias
}
is.na(G) <- 0
return(list(Z=c(Z0,Z),G=G))
}
y <- NULL
plot(0:5,Proceso2()$Z,type="l",ylim=c(0,10e6),xlab="tiempo",ylab="")
for(i in 1:100){y[[i]] <- Proceso2()$Z;lines(0:5,y[[i]])}
```



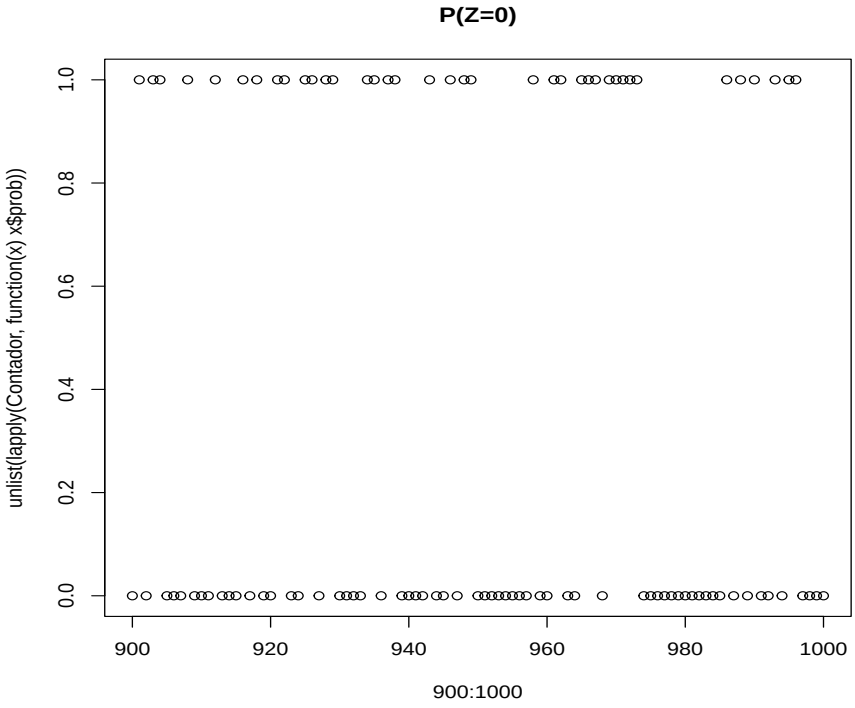

```r
Trayectorias <- matrix(unlist(y),ncol=6,byrow=T)
head(Trayectorias)
```

```
      [,1]      [,2]  [,3]      [,4]      [,5]       [,6]
[1,] 1e+06 1000000.0 1e+06 1000000.0 1000000.0 1000000.00
[2,] 1e+06 1000000.0 1e+06  823786.2 1000000.0 1000000.00
[3,] 1e+06 1000000.0 1e+06 1000000.0 1000000.0 1000000.00
[4,] 1e+06 1000000.0 1e+06 1000000.0  966207.8   28893.64
[5,] 1e+06       0.0 0e+00       0.0       0.0       0.00
[6,] 1e+06  892970.2 1e+06 1000000.0  391994.8 1000000.00
```

Las estadísticas requerida son ahora:

```r
#función que depende del número de trayectorias generadas
simula2 <- function(N){
Tr <- NULL #Trayectorias
Gn <- NULL #Ganancias
y <- Proceso2()
for(i in 1:N){Tr[[i]] <- y$Z;Gn[[i]] <- y$G}
Tr <- matrix(unlist(Tr),ncol=6,byrow = T)
```

```
prob <- sum(apply(Tr,1,function(x)0 %in% x))/N
acti <- mean(Tr[,6])
return(list(prob = prob, activos = acti,
ganancias = sum(unlist(lapply(Gn,sum,na.rm=T))/N,na.rm = T)))
}
Contador <- list(NULL)
for(n in 900:1000) Contador[[n-899]] <- simula2(n)
plot(900:1000,unlist(lapply(Contador,function(x)x$prob)),main="P(Z=0)")
```




```
mean((unlist(lapply(Contador,function(x)x$acti)))) #Activos promedio después de 5 años
```

```
[1] 537974.7
```

```
mean((unlist(lapply(Contador,function(x)x$ganancias)))) #Ganancia promedio después de 5 años
```

```
[1] 2822267
```

□

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

   (a) Cauchy

$$f(x) = \frac{1}{\pi\beta\left[1 + \left(\frac{x-\gamma}{\beta}\right)^2\right]}$$

   donde $\gamma, x \in \Re$, $\beta > 0$.

(b) Gumbel (o de valor extremo)

$$f(x) = \frac{1}{\beta} \exp\left(-e^{-(x-\gamma)/\beta} - \frac{x-\gamma}{\beta}\right)$$

donde $\gamma, x \in \Re$, $\beta > 0$.

(c) Logística

$$f(x) = \frac{(1/\beta)e^{-(x-\gamma)/\beta}}{(1+e^{-(x-\gamma)/\beta})^2}$$

donde $\gamma, x \in \Re$, $\beta > 0$.

(d) Pareto

$$f(x) = \frac{\alpha_2 c^{\alpha_2}}{x^{\alpha_2+1}}$$

donde $c > 0, \alpha_2 > 0, x > c$.

(e) Vasicek

$$f_{p,\rho}(x) = \sqrt{\frac{1-\rho}{\rho}} \exp\left(\frac{1}{2}\left\{\Phi^{-1}(x)^2 - \left(\frac{\sqrt{1-\rho}\Phi^{-1}(x) - \Phi^{-1}(p)}{\sqrt{\rho}}\right)^2\right\}\right)$$

donde $p, \rho \in (0,1)$ y $\Phi(x)$ es la distribución normal estándar.

Para $\gamma = 0$ y $\beta = 1$ en cada inciso (a), (b) y (c), usen los algoritmos que obtuvieron para generar una muestra aleatoria de 5000 y obtengan $\bar{X}(n) = \sum_{i=1}^{n} X_i/n$ para $n = 50, 100, 150, \ldots, 5000$ para verificar empíricamente la ley fuerte de los grandes números. Hacer lo mismo para (d) con $c = 1$ y $\alpha_2 = 2$.

Solución.

(a) Este caso ya lo hicimos como ejemplo de clase, con una distribución Cauchy estandarizada. Basta con estandarizar la variable $Z = \frac{X-\gamma}{\beta}$. Aquí nos sirve directamente el método de la transformación inversa.

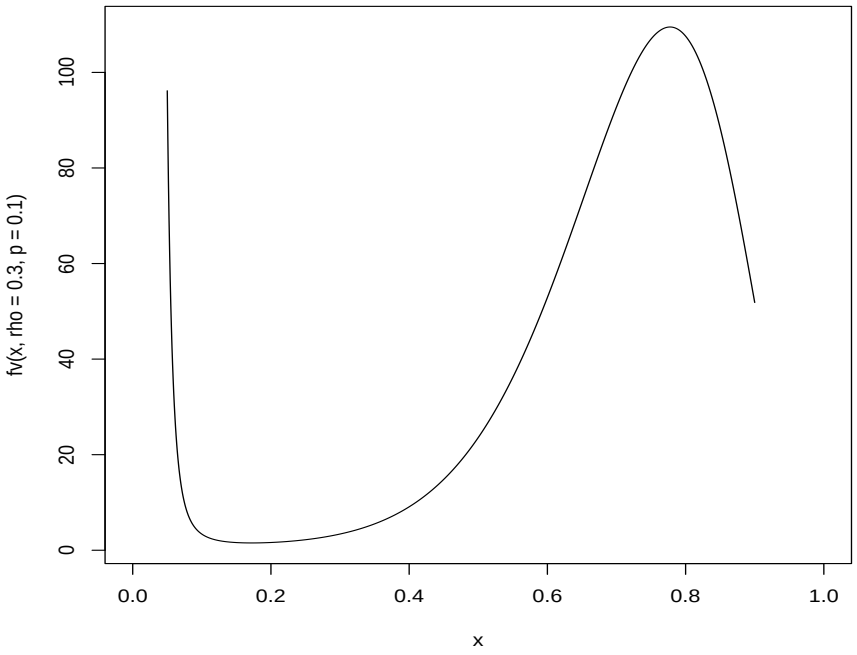
(b) La distribución Gumbel puede invertirse fácilmente. La distribución es de la forma $F(x) = e^{-e^{-\frac{x-\mu}{\beta}}}$ y entonces $X = -\beta log(-log(u)) + \mu$. El método de la transformación inversa se puede aplicar de manera directa.

(c) La distribución logística también es muy fácil de invertir. Tiene distribución $F(x) = \frac{1}{1+e^{-(x-\mu)/\beta}}$. Entonces $X = -\beta log(1/u - 1) + \mu$.

(d) La función de distribución está dada por $F(x) = 1 - \left(\frac{c}{x}\right)^{\alpha_2}$ Lo que resulta en la inversa $X = \frac{c}{(1-u)^{1/\alpha_2}}$.

(e) La funcion de Vasicek tiene dominio en $(0, 1)$ y rango en $(0, \infty)$. Por ejemplo el comportamiento se puede ver que está acotado para la mayor parte del dominio, pero su forma dependerá de los valores de p y ρ dado. En términos generales, para algunos casos se puede usar una distribución exponencial, y como una posible aproximación se puede usar una distribución uniforme para acotar la distribución. Un ejemplo se grafica a continuación:

```
fv <- function(x,rho=0.3,p=0.2){
sqrt((1-rho)/rho)*exp(0.5*(qnorm(x)^2-((sqrt(1-rho)*qnorm(x)-qnorm(p))/sqrt(rho)))^2)
}
curve(fv(x,rho=0.3,p=0.1),from = 0.05, to=0.9,xlim=c(0,1),n = 1000)
```



Entonces el método de aceptación-rechazo parece razonable aplicar en este caso.

El resto del ejercicio es un ejercicio estándar fácil de completar.

□

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.

   (a)

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

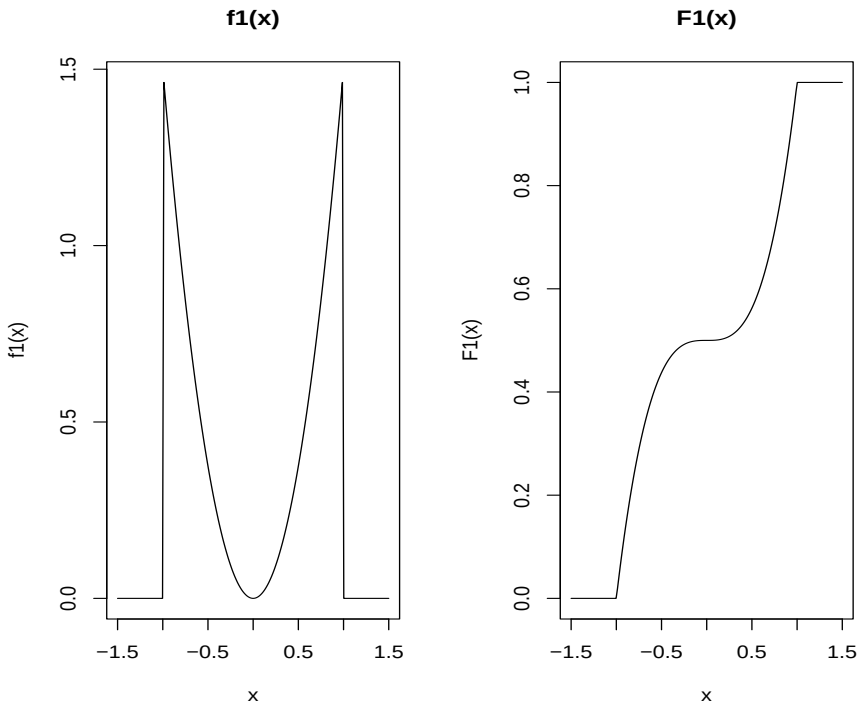
(b) Para $0 < a < \frac{1}{2}$,

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

Solución.

Para el inciso (a), consideremos las funciones de densidad y distribución dadas a continuación:

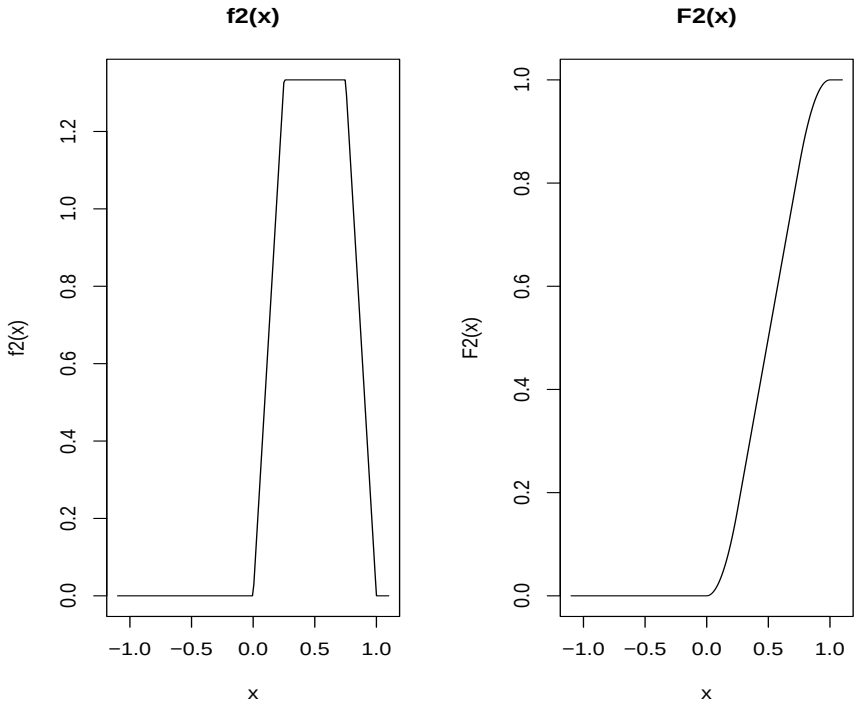
```
indicator <- function(x,a,b){ifelse(x <=b & x >=a,1,0)} #función indicadora en el intervalo (a,b)
f1 <- function(x){3*x^2/2*indicator(x,-1,1)}
F1 <- function(x){ifelse(x<=-1,0,ifelse(x<=1,0.5*(x^3+1),1))}
x <- seq(-1.5,1.5,length=200)  #intervalo de graficación
par(mfrow=c(1,2))
plot(x,f1(x),type="l",main="f1(x)")
plot(x,F1(x),type="l",main="F1(x)")
```



Para el inciso (b)

```
f2 <- function(x,a=0.25){indicator(x,-1,1)*(indicator(x,0,a)*(x/(a*(1-a))) +
indicator(x,a,1-a)/(1-a) +
indicator(x,1-a,1)*(1-x)/(a*(1-a)) )}
F2 <- function(x,a=0.25){indicator(x,0,a)*x^2/(2*a*(1-a)) +
(x-a/2)/(1-a)*indicator(x,a,1-a) +
((1-3*a/2)/(1-a) + (x*(1-x/2)-(1-a)*(1+a)/2)/(a*(1-a)))*indicator(x,1-a,1) +
indicator(x,1,100)
```

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



□

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 $\pi/4$, y encuentren 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.

   Para la primera parte, basta un algoritmo geométrico. La región en donde se rechazan los puntos corresponden al área sobrante del cuadrado que circunscribe el círculo con radio unitario. Esa región tiene área $\frac{4-\pi}{4} = 1 - \pi/4 = 0.215$. Entonces se rechaza 21.5 % del tiempo. Ahora bien, si $X =$ número de rechazos antes de aceptar, sabemos que $X \sim geom(\pi/4)$. Entonces $E(X) = 1/\frac{\pi}{4} = 4/\pi \approx 1.2732395$

   □

6. Obtengan una muestra de 1,000 números de la siguiente distribución discreta:

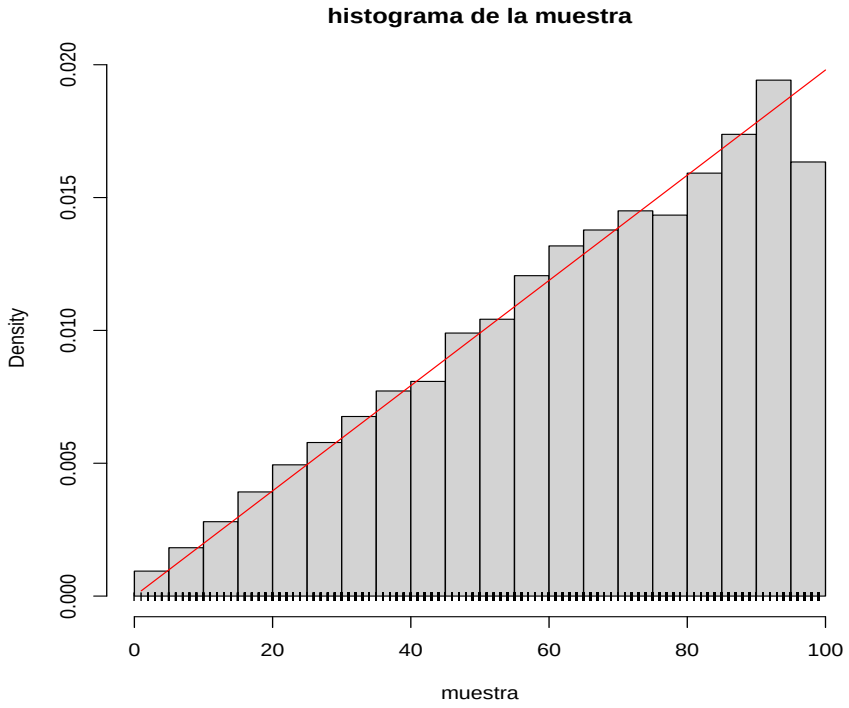
$$p(x) = \frac{2x}{k(k+1)}, x = 1, 2, \ldots, k$$

para $k = 100$.

Solución.

Noten que la función de distribución tiene la forma: $F(j) = P(X \leq j) = \frac{2}{k(k+1)} \sum_{i=1}^{j} x = \frac{2}{k(k+1)} \times \frac{j(j+1)}{2}$ ya que la suma corresponde a la suma de los primeros j naturales. Entonces podemos generar los "cajones" de la función acumulativa relativamente simple.

```
set.seed(1)
k <- 100   #parámetro de la distribución
n <- 10000 #tamaño de muestra
Fn <- ((1:k)*(1+(1:k)) )/(k*(k+1)) #Escalones de distribución
u <- runif(10000) #bten unifotmes
muestra <- findInterval(u,Fn,left.open=T) #muestra solicitada
hist(muestra,prob=T,main="histograma de la muestra")
lines(1:100,2*(1:100)/(100*101),col="red") #función de probabilidad
points(muestra,rep(0,10000),pch="|",cex=0.5) #puntos de la muestra
```



□

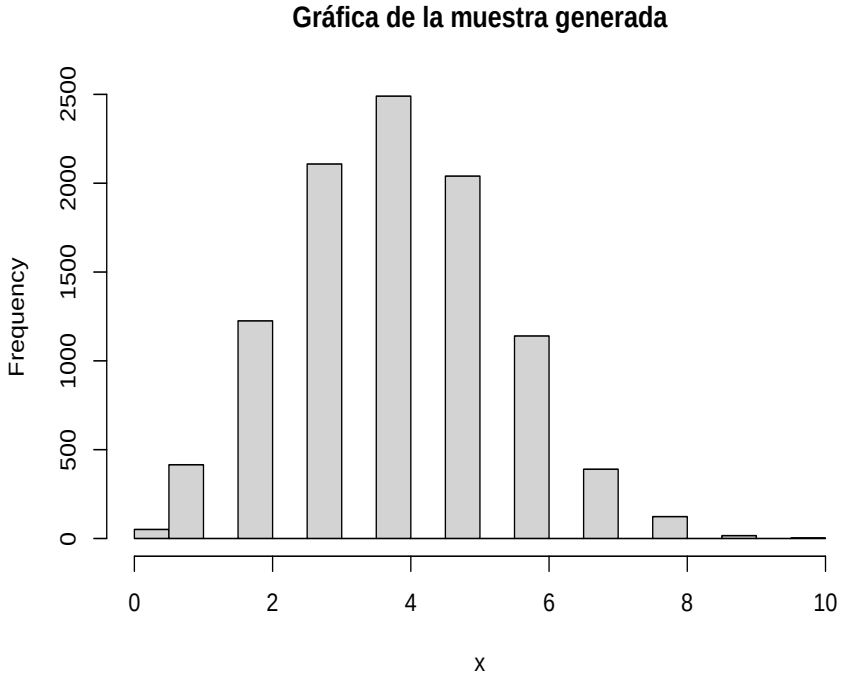
7. ▪ Desarrollen un algoritmo para generar una variable aleatoria binomial, usando la técnica de convolución (Hint: ¿cuál es la relación entre binomiales y Bernoullis?) Generar una muestra de 100,000 números. ¿Qué método es más eficiente, el de convoluciones o la función `rbinom` en R?

- Ahora supongan que tienen Y como una suma de $n = 100$ variables Bernoulli con parámetro $p = 0.6$ correlacionadas, con correlación $\rho = 0.3$. Obtener por simulación la estimación de la densidad de Y.

Solución.

El método de convolución es el que genera sumas de variables aleatorias. En este caso, una binomial con parámetros n y p es la suma de n variables Bernoulli con parámetro p. Por lo tanto, basta con generar n Bernoullis y agregar

```
muestraBinomial <- function(N,n,p){
x <- NULL
for(i in 1:N){
u <-runif(n)
y <- ifelse(u<=p,1,0)
x <- c(x,sum(y))
}
return(x)
}
x <- muestraBinomial(10000,10,0.4)
hist(x,main="Gráfica de la muestra generada")
```



Por último, para ver cuál es más eficiente, podemos tomar el tiempo de ejecución de ambos métodos

```
system.time(x <- muestraBinomial(1000,10,0.4))


   user  system elapsed
  0.006   0.000   0.006
```

```
system.time(x <- rbinom(1000,10,0.4))


   user  system elapsed
      0       0       0
```

Claramente, el método de R es mucho más eficiente.

□

8. Escribir una función para generar una mezcla de una distribución normal multivariada con dos componentes con medias μ_1 y μ_2 y matrices de covarianzas S_1 y S_2 respectivamente.

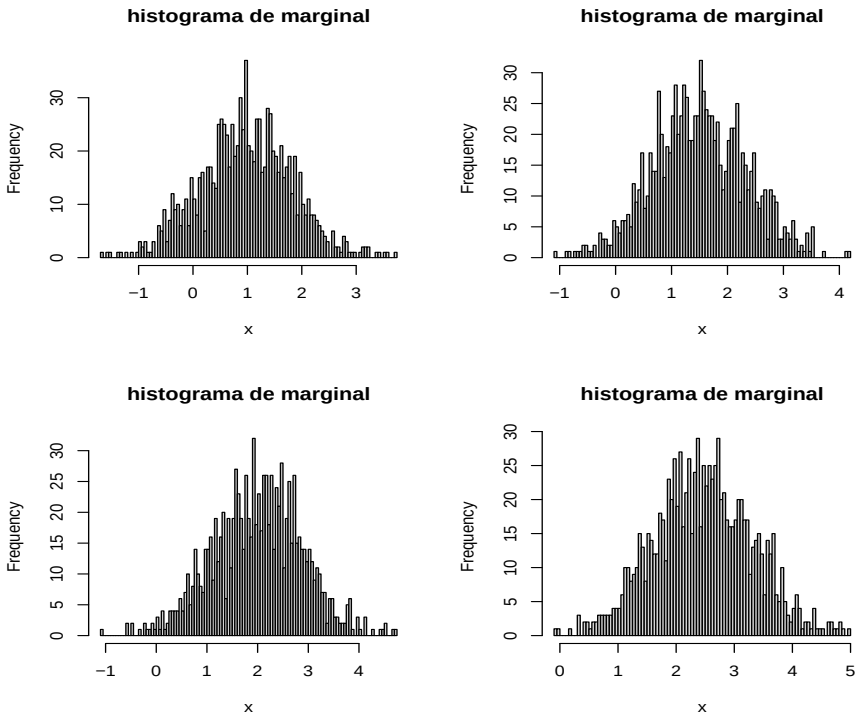
   a. Con el programa, generar una muestra de tamaño $n = 1000$ observaciones de una mezcla $50\,\%$ de una normal 4-dimensional con $\mu_1 = (0,0,0,0)$ y $\mu_2 = (2,3,4,5)$, y matrices de covarianzas $S_1 = S_2 = I_4$.

   - Obtener los histogramas de las 4 distribuciones marginales.

Solución.

El siguiente script hace lo solicitado. Aquí estoy usando la función `mvrnorm` del paquete `MASS`, pero también se puede usar el método de Box-Müller para generar los valores normales de los vectores.

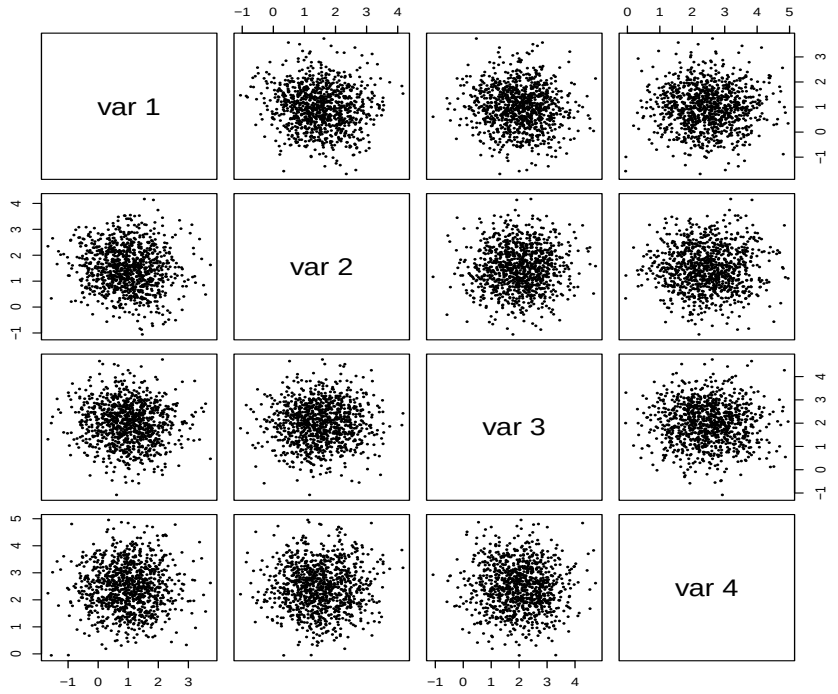
```
library(MASS)

mezclaNor <- function(n,mu1,mu2,S1,S2,p){
#Esta función genera una mezcla de dos normales p*N(mu1,S1) + (1-p)*N(mu2,S2)
p*mvrnorm(n,mu=mu1,Sigma=S1) + (1-p)*mvrnorm(n,mu=mu2,Sigma=S2)
}

mu1 <- c(0,0,0,0)
mu2 <- c(2,3,4,5)
S1 <- diag(4)
S2 <- 2*diag(4)
p <- 0.5

Z <- mezclaNor(1000,mu1,mu2,S1,S2,p)
par(mfrow=c(2,2))
apply(Z,2,hist,breaks=100, main=paste("histograma de marginal"),xlab="x")
```




```
[[1]]
$breaks
  [1] -1.70 -1.65 -1.60 -1.55 -1.50 -1.45 -1.40 -1.35 -1.30 -1.25 -1.20 -1.15
 [13] -1.10 -1.05 -1.00 -0.95 -0.90 -0.85 -0.80 -0.75 -0.70 -0.65 -0.60 -0.55
 [25] -0.50 -0.45 -0.40 -0.35 -0.30 -0.25 -0.20 -0.15 -0.10 -0.05  0.00  0.05
 [37]  0.10  0.15  0.20  0.25  0.30  0.35  0.40  0.45  0.50  0.55  0.60  0.65
 [49]  0.70  0.75  0.80  0.85  0.90  0.95  1.00  1.05  1.10  1.15  1.20  1.25
 [61]  1.30  1.35  1.40  1.45  1.50  1.55  1.60  1.65  1.70  1.75  1.80  1.85
 [73]  1.90  1.95  2.00  2.05  2.10  2.15  2.20  2.25  2.30  2.35  2.40  2.45
 [85]  2.50  2.55  2.60  2.65  2.70  2.75  2.80  2.85  2.90  2.95  3.00  3.05
 [97]  3.10  3.15  3.20  3.25  3.30  3.35  3.40  3.45  3.50  3.55  3.60  3.65
[109]  3.70  3.75

$counts
  [1]  1  0  1  1  0  0  1  1  0  1  0  1  0  1  3  2  3  1  1  3  0  6  5  9  3
 [26]  7 12  9 10  6  9 11  6 15 11  8 15 16  5 17 17 14 13 25 26 25 23 17 25 19
 [51] 21 30 24 37 21 20 18 26 26 16 17 28 27 20 19 16 21 15 17 19 12 19  8 16 10
 [76]  8 11  8  8  7  6  5  4  3  0  5  2  2  1  4  3  1  1  1  0  1  2  2  2  0
[101]  0  1  1  0  1  1  0  0  1

$density
  [1] 0.02 0.00 0.02 0.02 0.00 0.00 0.02 0.02 0.00 0.02 0.00 0.02 0.00 0.02 0.06
 [16] 0.04 0.06 0.02 0.02 0.06 0.00 0.12 0.10 0.18 0.06 0.14 0.24 0.18 0.20 0.12
 [31] 0.18 0.22 0.12 0.30 0.22 0.16 0.30 0.32 0.10 0.34 0.34 0.28 0.26 0.50 0.52
 [46] 0.50 0.46 0.34 0.50 0.38 0.42 0.60 0.48 0.74 0.42 0.40 0.36 0.52 0.52 0.32
 [61] 0.34 0.56 0.54 0.40 0.38 0.32 0.42 0.30 0.34 0.38 0.24 0.38 0.16 0.32 0.20
 [76] 0.16 0.22 0.16 0.16 0.14 0.12 0.10 0.08 0.06 0.00 0.10 0.04 0.04 0.02 0.08
 [91] 0.06 0.02 0.02 0.02 0.00 0.02 0.04 0.04 0.04 0.00 0.00 0.02 0.02 0.00 0.02
[106] 0.02 0.00 0.00 0.02

$mids
  [1] -1.675 -1.625 -1.575 -1.525 -1.475 -1.425 -1.375 -1.325 -1.275 -1.225
 [11] -1.175 -1.125 -1.075 -1.025 -0.975 -0.925 -0.875 -0.825 -0.775 -0.725
 [21] -0.675 -0.625 -0.575 -0.525 -0.475 -0.425 -0.375 -0.325 -0.275 -0.225
 [31] -0.175 -0.125 -0.075 -0.025  0.025  0.075  0.125  0.175  0.225  0.275
 [41]  0.325  0.375  0.425  0.475  0.525  0.575  0.625  0.675  0.725  0.775
 [51]  0.825  0.875  0.925  0.975  1.025  1.075  1.125  1.175  1.225  1.275
 [61]  1.325  1.375  1.425  1.475  1.525  1.575  1.625  1.675  1.725  1.775
 [71]  1.825  1.875  1.925  1.975  2.025  2.075  2.125  2.175  2.225  2.275
 [81]  2.325  2.375  2.425  2.475  2.525  2.575  2.625  2.675  2.725  2.775
 [91]  2.825  2.875  2.925  2.975  3.025  3.075  3.125  3.175  3.225  3.275
[101]  3.325  3.375  3.425  3.475  3.525  3.575  3.625  3.675  3.725

$xname
[1] "newX[, i]"
```

```
$equidist
[1] TRUE

attr(,"class")
[1] "histogram"

[[2]]
$breaks
  [1] -1.10 -1.05 -1.00 -0.95 -0.90 -0.85 -0.80 -0.75 -0.70 -0.65 -0.60 -0.55
 [13] -0.50 -0.45 -0.40 -0.35 -0.30 -0.25 -0.20 -0.15 -0.10 -0.05  0.00  0.05
 [25]  0.10  0.15  0.20  0.25  0.30  0.35  0.40  0.45  0.50  0.55  0.60  0.65
 [37]  0.70  0.75  0.80  0.85  0.90  0.95  1.00  1.05  1.10  1.15  1.20  1.25
 [49]  1.30  1.35  1.40  1.45  1.50  1.55  1.60  1.65  1.70  1.75  1.80  1.85
 [61]  1.90  1.95  2.00  2.05  2.10  2.15  2.20  2.25  2.30  2.35  2.40  2.45
 [73]  2.50  2.55  2.60  2.65  2.70  2.75  2.80  2.85  2.90  2.95  3.00  3.05
 [85]  3.10  3.15  3.20  3.25  3.30  3.35  3.40  3.45  3.50  3.55  3.60  3.65
 [97]  3.70  3.75  3.80  3.85  3.90  3.95  4.00  4.05  4.10  4.15  4.20

$counts
  [1]  1  0  0  0  1  1  0  1  1  1  2  2  1  1  2  0  4  3  3  2  2  6  5  4  6
 [26]  6  7  5 12  9 11 17  8 10 17 14 14 27 20 13 18 17 23 28 20 23 28 26 19 19
 [51] 23 23 32 27 24 23 23 19 22 15 11 14 19 21 21 25  9 17 15 11 14 17  9  8 10
 [76] 11  3 11 10  9  3  3  5  4  3  6  3  1  2  1  4  1  5  0  0  0  1  0  0  0
[101]  0  0  0  0  1  1

$density
  [1] 0.02 0.00 0.00 0.00 0.02 0.02 0.00 0.02 0.02 0.02 0.04 0.04 0.02 0.02 0.04
 [16] 0.00 0.08 0.06 0.06 0.04 0.04 0.12 0.10 0.08 0.12 0.12 0.14 0.10 0.24 0.18
 [31] 0.22 0.34 0.16 0.20 0.34 0.28 0.28 0.54 0.40 0.26 0.36 0.34 0.46 0.56 0.40
 [46] 0.46 0.56 0.52 0.38 0.38 0.46 0.46 0.64 0.54 0.48 0.46 0.46 0.38 0.44 0.30
 [61] 0.22 0.28 0.38 0.42 0.42 0.50 0.18 0.34 0.30 0.22 0.28 0.34 0.18 0.16 0.20
 [76] 0.22 0.06 0.22 0.20 0.18 0.06 0.06 0.10 0.08 0.06 0.12 0.06 0.02 0.04 0.02
 [91] 0.08 0.02 0.10 0.00 0.00 0.00 0.02 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.02
[106] 0.02

$mids
  [1] -1.075 -1.025 -0.975 -0.925 -0.875 -0.825 -0.775 -0.725 -0.675 -0.625
 [11] -0.575 -0.525 -0.475 -0.425 -0.375 -0.325 -0.275 -0.225 -0.175 -0.125
 [21] -0.075 -0.025  0.025  0.075  0.125  0.175  0.225  0.275  0.325  0.375
 [31]  0.425  0.475  0.525  0.575  0.625  0.675  0.725  0.775  0.825  0.875
 [41]  0.925  0.975  1.025  1.075  1.125  1.175  1.225  1.275  1.325  1.375
 [51]  1.425  1.475  1.525  1.575  1.625  1.675  1.725  1.775  1.825  1.875
 [61]  1.925  1.975  2.025  2.075  2.125  2.175  2.225  2.275  2.325  2.375
 [71]  2.425  2.475  2.525  2.575  2.625  2.675  2.725  2.775  2.825  2.875
 [81]  2.925  2.975  3.025  3.075  3.125  3.175  3.225  3.275  3.325  3.375
 [91]  3.425  3.475  3.525  3.575  3.625  3.675  3.725  3.775  3.825  3.875
[101]  3.925  3.975  4.025  4.075  4.125  4.175

$xname
[1] "newX[, i]"

$equidist
[1] TRUE

attr(,"class")
[1] "histogram"

[[3]]
$breaks
  [1] -1.10 -1.05 -1.00 -0.95 -0.90 -0.85 -0.80 -0.75 -0.70 -0.65 -0.60 -0.55
 [13] -0.50 -0.45 -0.40 -0.35 -0.30 -0.25 -0.20 -0.15 -0.10 -0.05  0.00  0.05
 [25]  0.10  0.15  0.20  0.25  0.30  0.35  0.40  0.45  0.50  0.55  0.60  0.65
 [37]  0.70  0.75  0.80  0.85  0.90  0.95  1.00  1.05  1.10  1.15  1.20  1.25
 [49]  1.30  1.35  1.40  1.45  1.50  1.55  1.60  1.65  1.70  1.75  1.80  1.85
 [61]  1.90  1.95  2.00  2.05  2.10  2.15  2.20  2.25  2.30  2.35  2.40  2.45
 [73]  2.50  2.55  2.60  2.65  2.70  2.75  2.80  2.85  2.90  2.95  3.00  3.05
 [85]  3.10  3.15  3.20  3.25  3.30  3.35  3.40  3.45  3.50  3.55  3.60  3.65
 [97]  3.70  3.75  3.80  3.85  3.90  3.95  4.00  4.05  4.10  4.15  4.20  4.25
[109]  4.30  4.35  4.40  4.45  4.50  4.55  4.60  4.65  4.70  4.75

$counts
  [1]  1  0  0  0  0  0  0  0  0  0  2  0  2  0  0  1  0  2  1  1  2  1  3  1  4
 [26]  1  2  4  4  4  4  6  4  7 10  5  8 14 10  8  7 14 14 16  9 19 12 16 20  6
 [51] 19 11 19 27 23 19 14 26 19 16 32 18 23 17 26 26 18 26 14 24 21 28 11 19 25
 [76] 15 26 15 16 14 14 12 14 12  9 11 10  7  7  2  6  6  3  3  2  2  2  5  6  1
[101]  0  3  1  0  3  0  0  1  0  0  1  1  2  0  0  1  1

$density
  [1] 0.02 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.04 0.00 0.04 0.00 0.00
 [16] 0.02 0.00 0.04 0.02 0.02 0.04 0.02 0.06 0.02 0.08 0.02 0.04 0.08 0.08 0.08
 [31] 0.08 0.12 0.08 0.14 0.20 0.10 0.16 0.28 0.20 0.16 0.14 0.28 0.28 0.32 0.18
 [46] 0.38 0.24 0.32 0.40 0.12 0.38 0.22 0.38 0.54 0.46 0.38 0.28 0.52 0.38 0.32
 [61] 0.64 0.36 0.46 0.34 0.52 0.52 0.36 0.52 0.28 0.48 0.42 0.56 0.22 0.38 0.50
 [76] 0.30 0.52 0.30 0.32 0.28 0.28 0.24 0.28 0.24 0.18 0.22 0.20 0.14 0.14 0.04
 [91] 0.12 0.12 0.06 0.06 0.04 0.04 0.04 0.10 0.12 0.02 0.00 0.06 0.02 0.00 0.06
[106] 0.00 0.00 0.02 0.00 0.00 0.02 0.02 0.04 0.00 0.00 0.02 0.02
```

```
$mids
  [1] -1.075 -1.025 -0.975 -0.925 -0.875 -0.825 -0.775 -0.725 -0.675 -0.625
 [11] -0.575 -0.525 -0.475 -0.425 -0.375 -0.325 -0.275 -0.225 -0.175 -0.125
 [21] -0.075 -0.025  0.025  0.075  0.125  0.175  0.225  0.275  0.325  0.375
 [31]  0.425  0.475  0.525  0.575  0.625  0.675  0.725  0.775  0.825  0.875
 [41]  0.925  0.975  1.025  1.075  1.125  1.175  1.225  1.275  1.325  1.375
 [51]  1.425  1.475  1.525  1.575  1.625  1.675  1.725  1.775  1.825  1.875
 [61]  1.925  1.975  2.025  2.075  2.125  2.175  2.225  2.275  2.325  2.375
 [71]  2.425  2.475  2.525  2.575  2.625  2.675  2.725  2.775  2.825  2.875
 [81]  2.925  2.975  3.025  3.075  3.125  3.175  3.225  3.275  3.325  3.375
 [91]  3.425  3.475  3.525  3.575  3.625  3.675  3.725  3.775  3.825  3.875
[101]  3.925  3.975  4.025  4.075  4.125  4.175  4.225  4.275  4.325  4.375
[111]  4.425  4.475  4.525  4.575  4.625  4.675  4.725

$xname
[1] "newX[, i]"

$equidist
[1] TRUE

attr(,"class")
[1] "histogram"

[[4]]
$breaks
  [1] -0.10 -0.05  0.00  0.05  0.10  0.15  0.20  0.25  0.30  0.35  0.40  0.45
 [13]  0.50  0.55  0.60  0.65  0.70  0.75  0.80  0.85  0.90  0.95  1.00  1.05
 [25]  1.10  1.15  1.20  1.25  1.30  1.35  1.40  1.45  1.50  1.55  1.60  1.65
 [37]  1.70  1.75  1.80  1.85  1.90  1.95  2.00  2.05  2.10  2.15  2.20  2.25
 [49]  2.30  2.35  2.40  2.45  2.50  2.55  2.60  2.65  2.70  2.75  2.80  2.85
 [61]  2.90  2.95  3.00  3.05  3.10  3.15  3.20  3.25  3.30  3.35  3.40  3.45
 [73]  3.50  3.55  3.60  3.65  3.70  3.75  3.80  3.85  3.90  3.95  4.00  4.05
 [85]  4.10  4.15  4.20  4.25  4.30  4.35  4.40  4.45  4.50  4.55  4.60  4.65
 [97]  4.70  4.75  4.80  4.85  4.90  4.95  5.00

$counts
  [1]  1  1  0  0  0  1  0  0  3  0  2  2  1  2  2  3  3  3  3  3  4  4  4  6 10
 [26] 10  8  9 10 15 13  8 15 14 12 12 18 17 11 23 20 26 19 27 16 21 26 15 24 29
 [51] 16 25 22 25 23 25 29 20 21 17 16 16 17 20 20 17 17  9 13 14 15 12  6 14 12
 [76] 15  6  5 10  5  3  2  4  6  4  1  2  2  1  4  1  1  1  0  2  2  1  1  2  1
[101]  0  1

$density
  [1] 0.02 0.02 0.00 0.00 0.00 0.02 0.00 0.00 0.06 0.00 0.04 0.04 0.02 0.04 0.04
 [16] 0.06 0.06 0.06 0.06 0.06 0.08 0.08 0.08 0.12 0.20 0.20 0.16 0.18 0.20 0.30
 [31] 0.26 0.16 0.30 0.28 0.24 0.24 0.36 0.34 0.22 0.46 0.40 0.52 0.38 0.54 0.32
 [46] 0.42 0.52 0.30 0.48 0.58 0.32 0.50 0.44 0.50 0.46 0.50 0.58 0.40 0.42 0.34
 [61] 0.32 0.32 0.34 0.40 0.40 0.34 0.34 0.18 0.26 0.28 0.30 0.24 0.12 0.28 0.24
 [76] 0.30 0.12 0.10 0.20 0.10 0.06 0.04 0.08 0.12 0.08 0.02 0.04 0.04 0.02 0.08
 [91] 0.02 0.02 0.02 0.00 0.04 0.04 0.02 0.02 0.04 0.02 0.00 0.02

$mids
  [1] -0.075 -0.025  0.025  0.075  0.125  0.175  0.225  0.275  0.325  0.375
 [11]  0.425  0.475  0.525  0.575  0.625  0.675  0.725  0.775  0.825  0.875
 [21]  0.925  0.975  1.025  1.075  1.125  1.175  1.225  1.275  1.325  1.375
 [31]  1.425  1.475  1.525  1.575  1.625  1.675  1.725  1.775  1.825  1.875
 [41]  1.925  1.975  2.025  2.075  2.125  2.175  2.225  2.275  2.325  2.375
 [51]  2.425  2.475  2.525  2.575  2.625  2.675  2.725  2.775  2.825  2.875
 [61]  2.925  2.975  3.025  3.075  3.125  3.175  3.225  3.275  3.325  3.375
 [71]  3.425  3.475  3.525  3.575  3.625  3.675  3.725  3.775  3.825  3.875
 [81]  3.925  3.975  4.025  4.075  4.125  4.175  4.225  4.275  4.325  4.375
 [91]  4.425  4.475  4.525  4.575  4.625  4.675  4.725  4.775  4.825  4.875
[101]  4.925  4.975

$xname
[1] "newX[, i]"

$equidist
[1] TRUE

attr(,"class")
[1] "histogram"


pairs(Z,pch=16,cex=0.5)
```



□

9. **Distribución de Wishart**. Suponer que $M = X'X$, donde X es una matriz $n \times d$ de una muestra aleatoria de una distribución $\mathcal{N}_d(\mu, \Sigma)$. 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 $\mathcal{N}(\mu, \sigma^2)$, por lo que $W_1(\sigma^2, n) \sim \sigma^2 \chi^2_{(n)}$.

   - Una forma de generar observaciones de una distribución Wishart, es generar muestras de multivariadas normales y calcular la matriz producto XX'. 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.
   - Un método más eficiente se basa en la descomposición de Bartlett: sea $T = (T_{ij})$ una matriz triangular inferior de $d \times d$ con entradas independientes que satisfacen

     a) $T_{ij} \sim \mathcal{N}(0, 1)$ independientes, para $i > j$.

     b) $T_{ii} \sim \sqrt{\chi^2_{(n-i+1)'}}$, $i = 1, \ldots, d$.

     Entonces la matriz $A = TT'$ tiene una distribución Wishart $W_d(I_d, n)$. Para generar variables $W_d(\Sigma, n)$, obtener la descomposición de Choleski $\Sigma = LL'$, donde L es triangular inferior. Entonces $LAL' \sim W_d(\Sigma, n)$. Implementar esta versión.

- Comparar en tiempo de ejecución de ambas versiones.

Solución.

Para la primera parte del ejercicio, definimos la función $W1$ de la siguiente manera:

```
W1 <- function(k, n, mu1, S1){
        #Esta función genera k muestras de la distrbución Wishart de  la forma ineficiente
        #mu1 es un vector de dimensión d, y S1 es una matriz definida positiva de dxd
        require(MASS)
        W <- list(NULL)
        for(i in 1:k){
                X <- mvrnorm(n, mu=mu1, Sigma=S1)
                W[[i]] <- t(X)%*%X
        }
        return(W)
}

mu1 <- rep(0,4)
S1 <- diag(4)
W1(k = 5, n = 10, mu1 = mu1, S1 = S1)
```

```
[[1]]
           [,1]       [,2]       [,3]      [,4]
[1,]  4.5238239 -4.217404 -0.5297557  2.354590
[2,] -4.2174040 14.674834 -3.7736701 -3.284190
[3,] -0.5297557 -3.773670  3.8429085  1.254447
[4,]  2.3545900 -3.284190  1.2544470  7.590232

[[2]]
           [,1]      [,2]       [,3]       [,4]
[1,] 16.4248518 0.5779624 -0.8918806  3.0287807
[2,]  0.5779624 8.6980658  0.4936748  4.6497504
[3,] -0.8918806 0.4936748  4.3761017  0.9641403
[4,]  3.0287807 4.6497504  0.9641403 12.3645559

[[3]]
          [,1]      [,2]      [,3]     [,4]
[1,] 11.359890  1.169155  0.108703 3.453882
[2,]  1.169155 15.999264  4.864873 3.841437
[3,]  0.108703  4.864873 12.902766 3.751852
[4,]  3.453882  3.841437  3.751852 7.096822

[[4]]
            [,1]        [,2]      [,3]        [,4]
[1,] 13.51168682 -0.07019222 -2.581868  0.1729957
[2,] -0.07019222 13.92381128  3.443246  1.9416983
[3,] -2.58186770  3.44324627  8.292735 -3.5953107
[4,]  0.17299571  1.94169830 -3.595311  4.7612561

[[5]]
          [,1]      [,2]         [,3]        [,4]
[1,]  7.543451 -0.212026 -2.525361486 2.940901792
[2,] -0.212026  9.309390 -3.037299117 2.141855521
[3,] -2.525361 -3.037299  5.035895578 0.008676576
[4,]  2.940902  2.141856  0.008676576 9.554519612
```

Para la segunda parte, usamos la transformación sugerida:

```
W2 <- function(k,n,mu1,S1){
        #Esta función genera k muestras de la distribuciń Wishart utilizando la
        #descomposición de Barttlet.
        W <- list(NULL)
        d <- length(mu1)
        M <- matrix(0,nrow=d,ncol=d)
        for(i in 1:k){
                M[lower.tri(matrix(0,nrow=d,ncol=d))] <- rnorm(d*(d+1)/2-d)
                diag(M) <- sqrt(rchisq(d,n-(1:d)-1))
                L <- chol(S1)
                W[[i]] <- L%*%M%*%t(M)%*%t(L)
        }
        return(W)
}

W2(k = 5, n = 10, mu1 = mu1, S1 = S1)
```

```
[[1]]
           [,1]       [,2]       [,3]      [,4]
[1,] 13.641061 -1.7870268 -1.8721784 -4.334959
[2,] -1.787027 15.0913221 -0.5522263  7.017252
[3,] -1.872178 -0.5522263  3.2750065 -1.030910
[4,] -4.334959  7.0172517 -1.0309099 14.469149

[[2]]
           [,1]       [,2]      [,3]       [,4]
[1,] 11.541435  0.9296300 -2.738330  1.2243084
[2,]  0.929630 13.6453735 -3.245827 -0.3203728
[3,] -2.738330 -3.2458269  9.093412 -6.9642971
[4,]  1.224308 -0.3203728 -6.964297  6.7450240

[[3]]
           [,1]      [,2]       [,3]      [,4]
[1,] 10.324726 1.8701650  2.7259366  4.408121
[2,]  1.870165 7.1609861  0.6236996  1.539009
[3,]  2.725937 0.6236996  3.9538339 -1.210674
[4,]  4.408121 1.5390095 -1.2106737  3.945813

[[4]]
           [,1]       [,2]      [,3]       [,4]
[1,] 3.0381600  0.7982082  2.045860  0.3049776
[2,] 0.7982082  7.8533360  4.685611 -1.4870930
[3,] 2.0458604  4.6856108  7.831324 -1.6666333
[4,] 0.3049776 -1.4870930 -1.666633  8.9214047

[[5]]
           [,1]       [,2]       [,3]      [,4]
[1,]  9.9015433  0.9143232  5.6345279 -1.729716
[2,]  0.9143232  8.6652366  0.2551746 -1.784614
[3,]  5.6345279  0.2551746 10.9747245 -5.194453
[4,] -1.7297162 -1.7846144 -5.1944527 11.166406
```

Para comparar los tiempos, generamos una muestra de tamaño grande, digamos $k = 10,000$ (idealmente para hacer una comparación adecuada, hay que hacer una matriz, variando tanto k como n y hasta de d).

```
k <- 10000
system.time(W <- W1(k,10,c(0,0,0,0),diag(4)))


   user  system elapsed
  0.470   0.000   0.469


system.time(W <- W2(k,10,c(0,0,0,0),diag(4)))


   user  system elapsed
  0.228   0.000   0.227
```

Noten que el tiempo es menor, aun cuando tuvimos que construir la matriz T (que yo llamé M) y hacer el producto de varias matrices, y esto es aún cuando estamos usando funciones optimizadas como `mvrnorm`.

□

10.
- Construyan un vector de 100 números crecientes y espaciados regularmente entre 0.1 y 20. Llámenlo `SIG2` . Ahora construyan otro vector de longitud 21 empezando en -1 y terminando en 1. Llámenlo `RHO`.
- Para cada entrada σ^2 de `SIG2` y cada entrada de `RHO`:
  - Generar una muestra de tamaño $N = 500$ de una distribución bivariada normal $Z = (X, Y)$ donde $X \sim \mathcal{N}(0, 1)$ y $Y \sim \mathcal{N}(0, \sigma^2)$ y el coeficiente de correlación de X y Y es ρ. Z es una matriz de dimensiones 500×2.

- Crear una matriz de 500×2, llámenlo `EXPZ`, con las exponenciales de las entradas de Z. ¿Qué distribución tienen estas variables transformadas?
- Calculen el coeficiente de correlación, $\tilde{\rho}$ de las columnas de `EXPZ`. Grafiquen los puntos $(\sigma^2, \tilde{\rho})$ y comenten sobre lo que obtuvieron.

Solución.

Para el primer inciso,

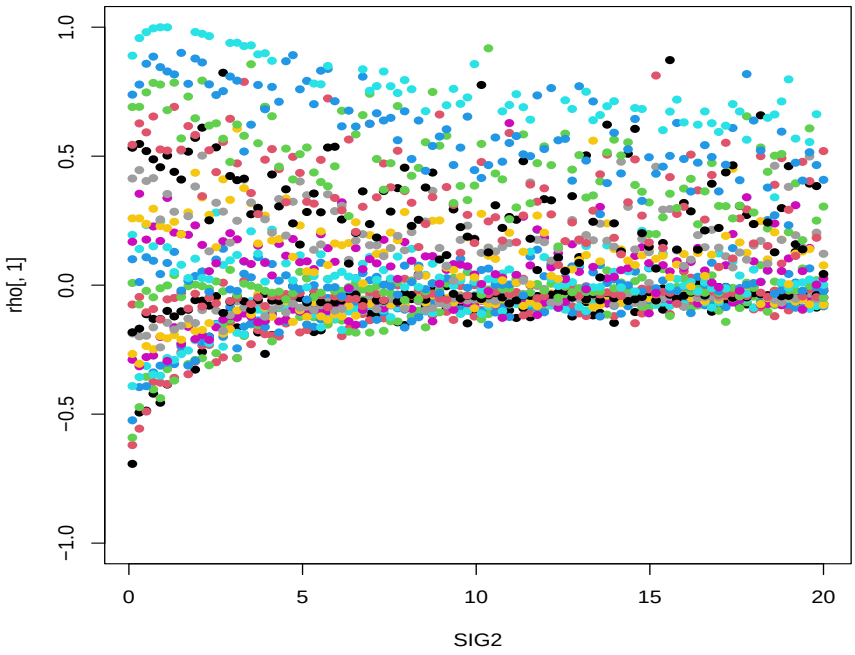
```
SIG2 <- seq(0.1,20,length=100)
RHO <- seq(-1,1,length=21)
```

Para el segundo inciso, generamos las normales utilizando el método de Box-Müller, que de construye con la siguiente función (pueden usar cualquier función que genere normales). Z tiene una distribución lognormal.

```
normalBM <- function(n){
        #genera una muestra de pares de normales independientes de tamaño n.
        u1 <- runif(n)
        u2 <- runif(n)
        R  <- sqrt(-2*log(u1))
        z1 <- R*cos(2*pi*u2)
        z2 <- R*sin(2*pi*u2)
        return(cbind(z1,z2))
}
```

Una vez obtenidos los pares de variables normales, multiplicamos por la matriz B tal que $BB' = \Sigma$, donde $\Sigma = \begin{pmatrix} 1, \rho\sigma \\ \rho\sigma, \sigma^2 \end{pmatrix}$.

```
rho <- matrix(numeric(),nrow=100,ncol=21)

suppressWarnings(
 for(i in SIG2){
        for(j in RHO){
                Sigma <- matrix(c(1,sqrt(i)*j,sqrt(i)*j,i),nrow=2,byrow=T)
                e <- eigen(Sigma)
                B <- e$vectors %*% diag(sqrt(e$values)) %*% t(e$vectors)
                ZEXP <- exp(normalBM(500) %*% B)
                rho[match(i,SIG2),match(j,RHO)] <- cor(ZEXP[,1],ZEXP[,2])
        }
}
) #Algunos puntos no están definidos, para no listar todos los casos.

plot(SIG2,rho[,1],pch=16,col=1,ylim=c(-1,1))

for(i in 2:21)points(SIG2,rho[,i],pch=16,col=i)
```



Lo que se puede observar de la gráfica, es que la transformación no es lineal para los valores de la correlación, y que la correlación no recorre el rango completo de posibles valores $(-1, 1)$ para la distribución lognormal.

□