PL3: Simulación de datos

Simular 50 datos de la v.a. X con distribución triangular con a=2

$$f_a(x) = \begin{cases} 1/a^2(x+a) & x \in (-a, 0] \\ 1/a^2(a-x) & x \in (0, a) \\ 0 & \text{en otro caso} \end{cases}$$

1. Utilizando el generador de datos de una $\beta(1,2)$ y de una bernoulli B(0.5). Comprobar que si $Y \equiv \beta(1,2)$ y V=B(0.5) independientes entonces $4 * (Y * (V - 0{,}5))$ tiene la misma distribución que X
2. Utilizando el método de aceptación rechazo con una envolvente escalonada.

Se pueden utilizar, además de las ya estudiadas, los siguientes elementos de R

- definición de funciones: **function**()
- búsqueda de máximos y mínimos de una función en un intervalo: **optimize**()
- bucles **for** (i in 1:n) ...., muy lentos en R
- réplicas de un mismo procedimiento **replicate**(nr,....)

```
##simulacion de datos con función de densidad triangular entre -2 y 2
## usando los generadores de R
rm(list=ls())
n<-50
x<- 4*(rbeta(n,1,2)*(0.5-rbinom(n,1,0.5)))
hist(x,freq=F)


##Simulación por el método de aceptación rechazo

f<- function(y) 1/4*((2+y)*(y<= 0)*(y>-2) + (2-y)*(y>0)*(y<2)) ## densidad de X

  ## parámetros: n,m
  n<- 50 ## tamaño de la muestra
  m<- 8 # numero de intervalos para acotar f(x)
  x<- c() ## crea el vector x
  e<- .00005
  ## Calculo de la densidad de Y
  a<- seq(-2,2,length=m+1) # extremos de los intervalos, son m+1
   plot(a,f(a), type='l',col=3)# dibuja la función de densidad

  h<- c() # vector para las cotas de la densidad
  for (k in 2:(m+1)) {o<- optimize(f, c(a[k-1], a[k]),  maximum=T);
                      h[k-1]<-as.numeric(o$objective)}
  h<- round(h+e,4);h # cotas de la densidad f(x)

  lines(c(a[1],a),c(0,h,0),type='s', col =2)# dibujo de la envolvente

  L<- diff(a); L # longitud de los intervalos
  M<- sum(L*h); M # area encerrada debajo de la envolvente
  g<- h/M; g # densidad g(y),  constante en los intervalos


  ## Función de distribución de Y en los extremos de los intervalos
  F<- rep(0, m+1)
  F[1]<- 0
  for (k in 2: (m+1)) { F[k]<- F[k-1]+ g[k-1]* L[k-1] }
  F

  ## Método de aceptación-rechazo

  nr<- ceiling(M*n); nr #hay que generar mas datos, porque se 'rechazan' algunos
x<-c()
u<-runif(nr) ##datos uniformes para generar valores de y
u2<- runif(nr) ##datos uniformes para saber si se rechaza o no
```

```
for(i in 1:nr) {
    k<- sum(u[i]>F); k ## intervalo  k donde está el valor y
    y<- (u[i]-F[k])/(F[k+1]-F[k]) *L[k]+ a[k]; y ## valor de y
    if ((u2[i]*h[k])< f(y) )   x<-c(x,y)  ##aceptacion
                  }
x;length(x)
hist(x,freq=FALSE)
```