2.5 Generación de variables aleatorias no-uniformes

Una variable aleatoria es una función que asume sus valores de acuerdo a los resultados de un experimento aleatorio. Es decir, un experimento donde existe incertidumbre acerca del resultado que va a ocurrir. Una variable aleatoria es discreta si su rango de valores es un conjunto finito o infinito enumerable. Existen una infinidad de variables aleatorias discretas, entre las mas conocidas están: la Binomial, la geométrica, la hipergeométrica, la Poisson y la Binomial Negativa.
Si la variable aleatoria discreta X tiene rango de valores R_X entonces la función p(k)=Prob[X=k] donde $x \in R_X$ es llamada la función de probabilidad de X. Asimismo, la función $F(t)=P[X \leq t]=\sum_{k \leq t} P[X = k]$ es llamada la función de distribución acumulativa de X
Una variable aleatoria continua es aquella cuyo rango de valores es cualquier intervalo de la recta real, entre las más conocidas están: la uniforme, la exponencial, la gamma, la Ji-Cuadrado, la Beta, la Normal, la t de Student, la Cauchy, la Weibull,etc.
Si la variable aleatoria continua X tiene rango de valores R_X entonces existe una función no-negativa f(x) tal que $P(a<X<b)=\int_a^b f(x)dx$. Asimismo. la función $F(t)=P[X \leq t]=\int_{-\infty}^{t} f(x)dx$ es llamada la función de distribución acumulativa de X.

2.5.1 Generación de variables aleatorias continuas.

Existen varios métodos generales y varios métodos particulares para generar valores de una variable aleatoria con una distribución dada. Muchos de ellos basados en relaciones que existen entre diversas variables aleatorias. Solo veremos el método de la transformación inversa y el método de aceptación y rechazo. Por otro lado, solo discutiremos la generación de la distribución exponencial y normal.

2.5.1.1 Método de la Transformacion Inversa

Si X es una función de densidad continua con función de distribución acumulativa F(x) entonces Y=F(X) se distribuye uniformemente en el intervalo (0,1)
En efecto, la distribución acumulativa de Y está dado por $G(y)=P[Y \leq y]=P[F(X) \leq y]=P[X \leq F^{-}(y)]=F(F^{-}(y))=y$ para $0 \leq y \leq 1$, la cual corresponde a una acumulativa de una U(0,1).
En consecuencia para generar un valor de la variable aleatoria X con acumulada F(X) se sigue los siguientes pasos:

1. Generar U= U(0,1) usando la función runif en R o rand en Matlab.
2. Tomar $x=F^{-1}(U)$

Ejemplo (Generación de una variable aleatoria Exponencial)
Sea X una variable aleatoria exponencial con parámetro $\lambda>0$, entonces su función de densidad es $f(x)=\lambda e^{-\lambda x}$ para $x>0$ y su función de distribución acumulativa es $F(x)=1- e^{-\lambda x}$ para $x>0$. Luego,

$$U=1- e^{-\lambda x}$$

En consecuencia,

$$\text{Log}(U-1)= -\lambda x$$

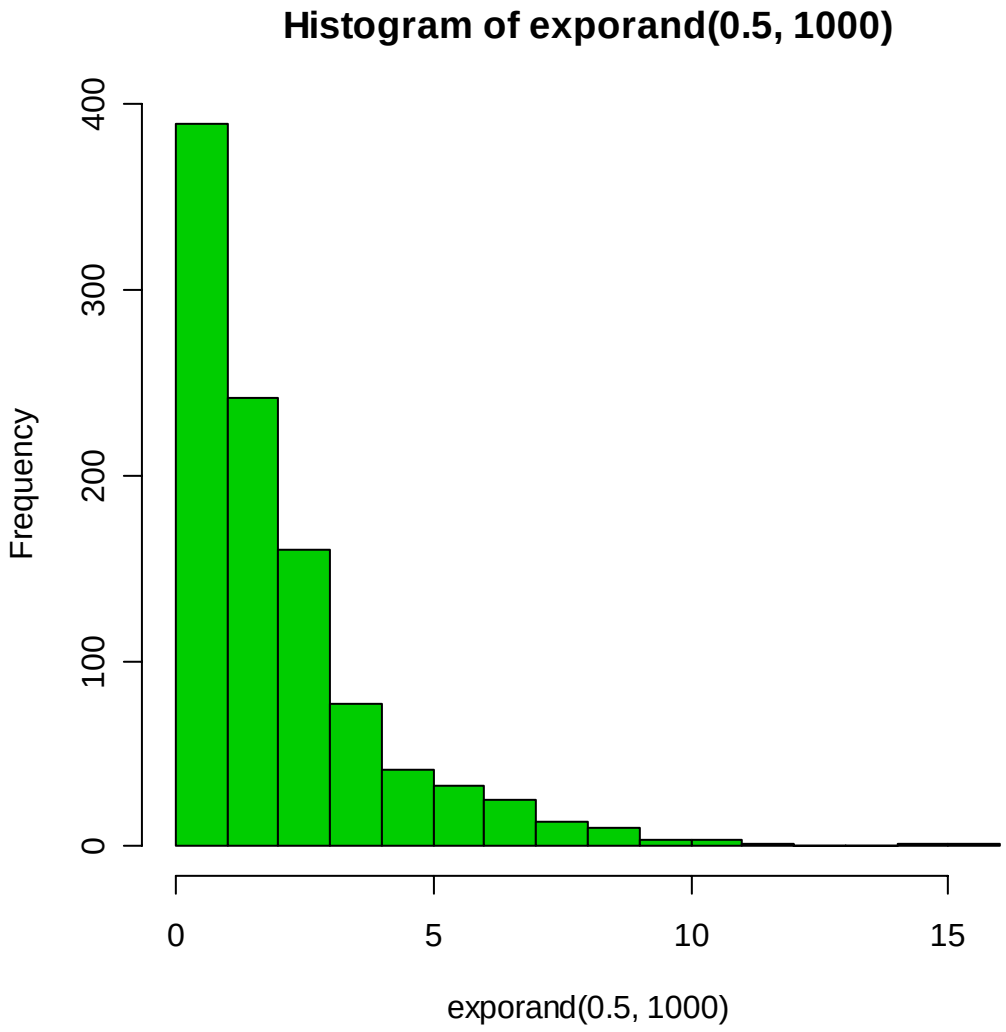
$X=\frac{-\log(U-1)}{\lambda}$. Una manera de acortar lo cálculos es usar el hecho que U-1 también se distribuye uniformemente en (0,1). Así, $x=-\log(U)/\lambda$ sería la exponencial generada.

La siguiente función exporand en R genera m valores exponenciales con

```
exporand=function(lambda,n)
{if (lambda<=0){ stop("lambda debe ser positivo")}
else {x=-log(runif(n))/lambda}
 x}

> #generando 10 valores exponenciales com lambda=3.5
>exporand(3.5,10)
 [1] 0.2748450 0.3956703 0.6047322 0.4420424 0.3177109 0.1501418 0.7154897
 [8] 0.1226956 0.3752534 0.2857398
> #haciendo um histograma de 1000 exponenciales com lambda=.5
>hist(exporand(0.5,1000),col=3)
>
```

La función rexp de R genera valores de una variable aleatoria exponencial.



En Matlab.

```
function e = exporand(lambda,m,n);

%    E=EXPORAND(U,m,n) genera una matriz m por n con valores
%     de una distribucion exponencial
%    E = EXOPRAND(lambda) genera un solo valor, un vector o
%    una matriz de valores exponenciales dependiendo de las
%   dimensiones del parametro lambda

%Cotejando que la funcion tenga los argumentos correctos
if nargin <  1,
    error('Se requiere al menos un argumento de entrada.');
end

if nargin == 1
    [errorcode rows columns] = rndcheck(1,1,lambda);
```

```
end

if nargin == 2
    [errorcode rows columns] = rndcheck(2,1,lambda,m);
end

if nargin == 3
    [errorcode rows columns] = rndcheck(3,1,lambda,m,n);
end

if errorcode > 0
    error('El tamaño de la informacion es inconsistente.');
end

%Initializar la matriz e como cero.
e = zeros(rows, columns);
% Generando los valores usando la transformacion inversa
u = rand(rows,columns);
e = - log(u)./lambda;

% Retornando NaN si el lambda no es positivo.
if any(any(lambda <= 0));
    tmp = NaN;
    if prod(size(lambda) == 1)
        r = tmp(ones(rows,columns));
    else
        k = find(lambda <= 0);
        r(k) = tmp(ones(size(k)));
    end
end
```

A continuación se presentan algunos ejemplos del uso de la función.

» exporand(2)

ans =

0.2828

» exporand(2.5,4,5)

ans =

0.1996	0.3181	1.7414	1.3371	1.6453
0.1849	1.2504	0.3829	0.1962	0.6641

```
   0.3972    1.4420    0.1524    0.1987    0.2131
   0.2213    0.4650    0.9507    1.6601    1.1418

» X=[4 6 2.5 8 9 1.5]'

X =

   4.0000
   6.0000
   2.5000
   8.0000
   9.0000
   1.5000


» y=exporand(X)

y =

   0.2502
   0.0766
   0.1327
   0.0459
   0.2751
   0.5259

»
```

2.5.1.2 Método de Aceptación y Rechazo (Von Neuman, 1951)

La idea aquí es que se tiene una variable aleatoria Y con función de densidad g(y), la cual puede ser generada fácilmente. Se desea generar otra variable X con función de densidad f(x), para ello se genera un valor de Y con densidad g(y) y se toma x=Y con probabilidad proporcional a f(y)/g(y). Esto es, P[X=y]=k f(y)/g(y), donde k es la constante de proporcionalidad. Si c es una constante tal que $f(y)/g(y) \leq c$ para todo y, entonces el siguiente seria el algoritmo de aceptación y rechazo para generar una variable aleatoria X con función de densidad f(x).

Paso 1: Generar Y con densidad g
Paso 2: Generar una variable aleatoria uniforme U(0,1)
Paso 3: Si $U \leq f(Y)/cg(Y)$ entonces X=Y de lo contrario volver al paso 1.

Mientras más cerca se encuentre f de g más rápidamente se obtendrá la cantidad deseada de valores aleatorios de X. Usualmente se toma la densidad uniforme como la densidad g(y) y en ese caso el método de aceptación y rechazo es llamado “hit and miss”.

Ejemplo. Usar el método de aceptación y rechazo para generar valores de una variable aleatoria con función de densidad

$$f(x)=20x(1-x)^3 , \qquad 0<x<1$$

Solución:
Tomaremos como g a la densidad uniforme, esto es g(y)=1, 0<y<1. Así que lo hay que determinar es la constante c tal que f(y)≤c para todo 0<y<1.
Derivando f(x) se obtiene:
$f'(x)=20(1-x)^3-60x(1-x)^2=20(1-x)^2[1-x-3x]= 20(1-x)^2[1-4x]$
f"(x)=0 implica x=1 o x=1/4. Claramente, $f(1/4)=5(3/4)_3=135/64$ es el máximo de f en (0,1). En consecuencia, f(x)≤135/64 para todo 0≤ x ≤1
.Luego, para generar valores de la variable con densidad anterior se siguen los siguientes pasos:
1. Generar valores de una variable aleatoria U_1.
2. Generar valores de una variable aleatoria U_2.
3. Si $U_2 \leq 20U_1(1-U_1)^3/(135/64)=(256/27)\ U_1(1-U_1)^3$ entonces $X=U_1$, de lo contrario volver al paso 1.

La siguiente funcion en R, genera valores aleatorios de la densidad del ejemplo anterior

```
Beta24=function(n)
{
X=NULL
count=1
while(count<=n)
{U1=runif(1)
U2=runif(1)
if(U2<=(256/27)*U1*(1-U1)^3)
{X=c(X,U1)
count=count+1}
}
X
}
```

El siguiente script en MATLAB genera valores aleatorios de la densidad del ejemplo

```
% Generando un vector de valores para el ejemplo 1 usando
% el metodo de aceptacion y rechazo
rand1 = rand(10,1)
rand2 = rand(10,1)
k=find(rand2<=(256/27)*rand1.*(1-rand1).^3)
b24=rand1(k)
```

2.5.1.3 Generación de valores de una distribución normal

La distribución Normal o Gaussiana es la más conocida de las distribuciones y la más aplicada en estadística. Tiene dos parámetros μ y σ y su función de densidad está dada por

$$f(x) = \frac{e^{-\frac{(x-\mu)^2}{2\sigma^2}}}{\sigma\sqrt{2\pi}}$$

donde $-\infty < x < \infty$, $-\infty < \mu < \infty$ y $\sigma > 0$. Cuando $\mu=0$ y $\sigma=1$ se obtiene la distribución normal estándar. Existen varios métodos de generar valores de una distribución normal:

a) Aplicando el Teorema del Limite central a una suma de uniformes. $\sum_{i=1}^{N} U_i \approx N(0,1/12)$

b) El Método Polar (o Transformación de Box-Miller , 1958)
c) El Método Polar de Marsaglia (1962)
d) El Método de Mezclas (kinderman y Ramage, 1976)
e) El Método de Rechazo de Forsythe (1972)

A continuación se aplicará el método de aceptación y rechazo para generar una normal. Antes que nada hay que notar que si Z es N(0,1), entonces X=| Z | tiene función de densidad

$$f(x) = \frac{2}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$$

donde x>0.. Podemos considerar la función de densidad de una exponencial con $\lambda=1$, esto es $g(x)=e^{-x}$ para x>0.
Luego,

$$\frac{f(x)}{g(x)} = \sqrt{2/\pi} e^{x - x^2/2}$$

cuyo máximo ocurrirá cuando $x-x^2/2$ sea máximo y eso ocurre cuando x=1. En consecuencia $c=\max(f(x)/g(x))= \sqrt{2e/\pi}$.

Haciendo las sustituciones adecuadas se sigue que,

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

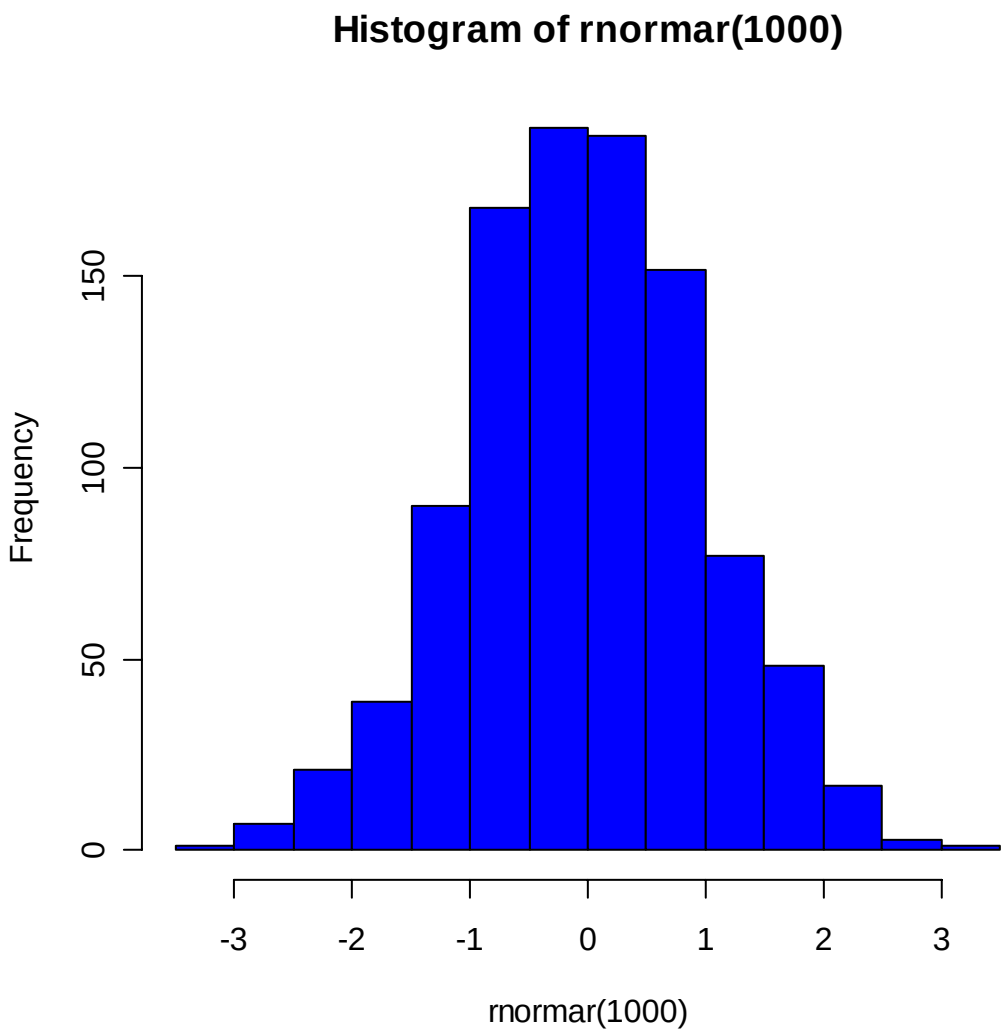
El procedimiento anterior generará X=|Z|. Para llegar a Z simplemente se le asigna signo positivo o negativo a X dependiendo de si una variable aleatoria uniforme U_2 es o no mayor que ½.

El siguiente sería pues el algoritmo para generar una normal
Paso 1. Generar Y una exponencial con parámetro 1.
Paso 2. Generar dos variables aleatorias uniformes U_1 y U_2.
Paso 3 Si $U_1 \leq \exp(-(Y-1)^2/2)$ y $U_2>1/2$ hacer Z=Y. Si $U_1 \leq \exp(-(Y-1)^2/2)$ y $U_2 \leq 1/2$ hacer Z=-Y.
Hacer Z=-Y . En otro caso volver al paso 1.

La siguiente función en R genera valores de una distribución normal y hace su histograma

```
rnormar=function(n)
{norm1=NULL
count=1
while(count<=n)
{y=rexp(1)
U1=runif(1)
U2=runif(1)
if(U1<=exp(-.5*(y-1)^2) && U2<=.5) {
norm1=c(norm1,-y)
count=count+1
}
if(U1<=exp(-.5*(y-1)^2) && U2>.5) {
norm1=c(norm1,y)
count=count+1}
}
norm1
}
```

```
> rnormar(10)
 [1]  0.44065050 -2.64843957 -1.92816070  0.26199993 -0.94323696 -0.90815978
 [7] -0.52152141 -0.46480357  0.02584946 -0.28069635
> hist(rnormar(1000))
> hist(rnormar(1000),col=4)
>
```



En R, hay la función rnorm , que genera valores de una normal con una media y una desviación estándar dadas.

En Matlab

```
% Generando un vector de valores de una normal usando
% el método de aceptacion y rechazo
y=exporand(1,10000,1);
rand1 = rand(10000,1);
rand2 = rand(10000,1);
k1=find(rand1<=exp(-.5.*(y-1).^2)& rand2<=.5);
k2=find(rand1<=exp(-.5.*(y-1).^2)& rand2>.5);
nor1=-y(k1);
nor2=y(k2);
nor=[nor1' nor2']'
hist(nor)
```

MATLAB tiene una función RANDN que genera valores de una normal estándar

2.5.2 Generación de variables aleatorias discretas.
Nuevamente aqui veremos solamente el método de la transformación inversa y el método de aceptación y rechazo.

2.5.2.1 Método de la transformación Inversa
Hay que recordar que si X es una variable aleatoria discreta con valores x1, x2,…
Entonces su función de probabilidad P(x=xj)=pj para j=1,2,…., satisface

$$p_j=F(x_j)-F(x_{j-1})$$

donde F representa la acumulada de X. Como la acumulada se puede considerar como una variable aleatoria U uniforme en (0,1) se tendría que

$$X=x_j \text{ si } F(x_{j-1}) \leq U < F(x_j)$$

En consecuencia el algoritmo quedaría como sigue:
Paso 1. Generar una uniforme U
Paso 2. Hacer Si $U<p_1$ hacer $X=x_1$ y parar.
Paso 3. Si $U<p_1+p_2$ hacer $X=x_2$ y parar.
Paso 4. Si $U<p_1+p_2+p_3$ hacer $X=x_3$ y parar
y así sucesivamente.

2.5.2.2 Generación de variables aleatorias binomiales
Recordar que una variable aleatoria binomial con parámetros n y p tiene función de probabilidad

$$P(X=k)=\binom{n}{k}p^k(1-p)^{n-k}$$

para k=0,1,2,… .n
Desarrollando la combinación y trabajando algebraicamente la razón P(X=k+1)/P(X=k), se puede establecer que

$$P(X=k+1)=\frac{n-k}{k+1}\frac{p}{1-p}P(X=k)$$

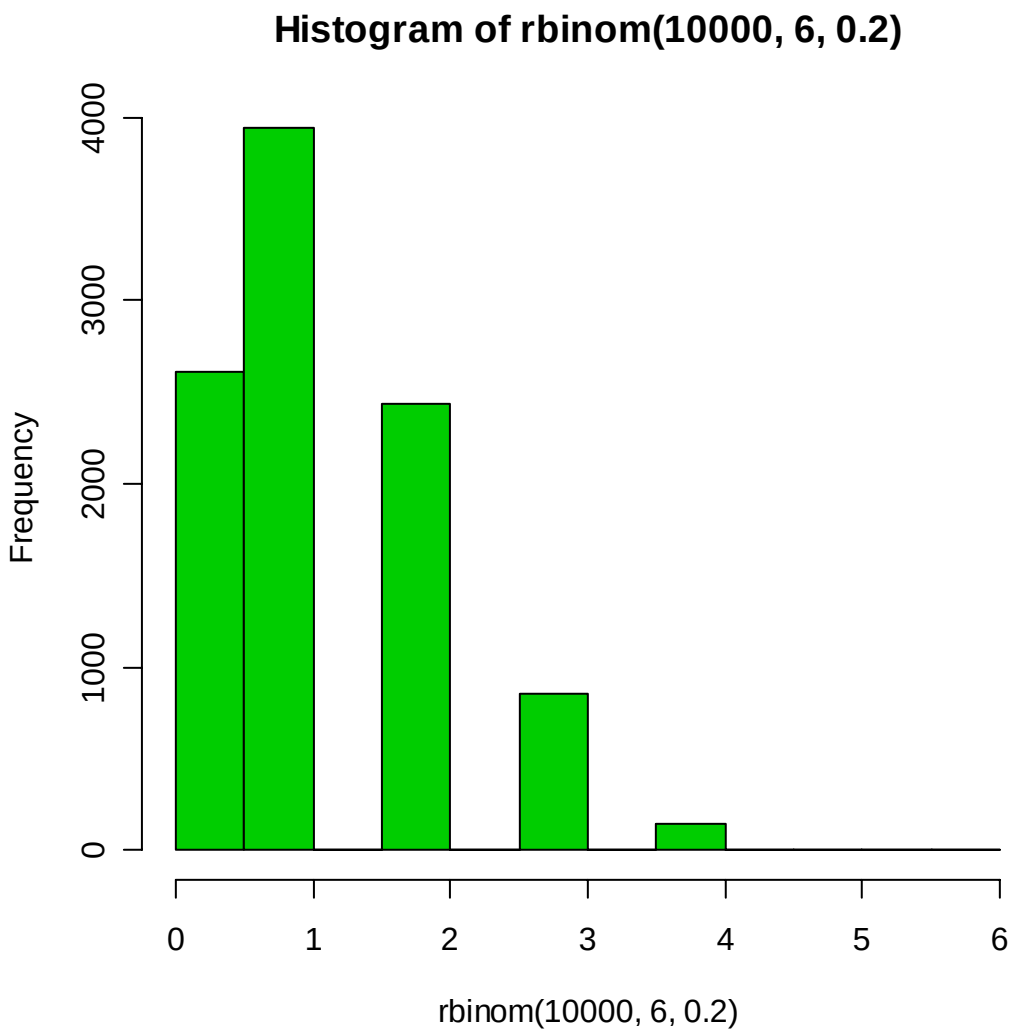
Aplicando el método de la transformación Inversa y la identidad anterior se obtiene el siguiente procedimiento para generar una Binomial, B(n,p)

Paso1: Generar un número aleatorio U
Paso 2 Hacer c=p/(1-p) , k=0, prob=$(1-p)^n$ y F=prob
Paso 3. Si U<F, hacer X=k y parar
Paso 4. Hacer Prob=[c(n-k)/(k+1)]Prob, F=F+prob, k=k+1
Paso 5. Ir al paso 3.

A siguiente función escrita en R genera valores de una binomial

```
> ranbin
function(N,p,n)
{
X=rep(0,N)
for(count in 1:N)
X[count]=ranbin1(p,n)
X
}
> ranbin1
function(p,n)
{
c1=p/(1-p)
k=0
prob=(1-p)^n
F=prob
U=runif(1)
while(U>=F)
{prob=(c1*(n-k)/(k+1))*prob
F=F+prob
#print(F)
k=k+1
}
X=k
X
}

> a=rbinom(10,6,.2)
> a
 [1] 1 1 1 1 0 3 2 2 1 1
> hist(rbinom(10000,6,.2),col=3)
>
```



En R, hay la función rbinom que genera valores de una variable aleatoria binomial.

2.5.2.3. Método de Aceptación y Rechazo

Supongamos que tenemos una variable discreta Y fácil de generar y que tiene una función de probabilidad $q_j=P[Y=y_j]$. Consideremos otra variable aleatoria X con función de probabilidad $p_j=P]X=j]$ la cual deseamos generar. La idea aqui es usar Y para generar X, parar ello hay que encontrar una constante c tal que $pj/qj \leq c$ para todo j y luego aplicar el siguiente procedimiento:

Paso 1. Generar la variable aleatoria Y
Paso 2. Generar una variable aleatoria Uniforme U
Paso 3. Si U<pj/cqj hacer X=Y y parar. De lo contrario ir al paso 1.