TEMA 2: Métodos de generación de variables aleatorias discretas y continuas

1. Introducción

Este tema es una revisión de los generadores clásicos de números pseudoaleartorios uniformes y de las técnicas usuales para la generación de variables aleatorias discretas y continuas a partir de una secuencia de números pseudoaleatorios. Puesto que ambos aspectos son básicos y ampliamente conocidos, y, actualmente se encuentran implementados y disponibles, como funciones en los paquetes estadísticos usuales, se proporcionará sólo una introducción esquemática de los mismos, para facilitar su implementación y la resolución de las actvidades propuestas.

En particular, en relación con la formulación de generadores de números pseudoaleatorios uniformes, en la Sección 2, se proporcionan los algoritmos y códigos para su implementación. Las técnicas de generación de variables aleatorias discretas y continuas se describen brevemente en las Secciones 3 y 4. Algunos ejemplos en código MatLab muestran su implementación en la generación de distribuciones concretas.

2. Generadores de números pseudoaleatorios

Se consideran los siguientes métodos de generación de números pseudoaleatorios:

- Método de los cuadrados medios
- Método de Lehmer
- Método Congruencial.

En particular, el método congruencial permite determinar el ciclo de repetición de la secuencia generada. Sus propiedades han sido ampliamente estudiadas en función de los parámetros que caracterizan este método.

2.1. Método de los cuadrados medios

Se formula seguidamente un algoritmo que describe la implementación del generador de números pseudoaleatorios dado por el método de los cuadrados medios junto con un ejemplo implementado en código MatLab.

Algoritmo

Paso 1. Elegir una semilla o valor inicial x_0 de $2n$ cifras.

Paso 2. Elevar al cuadrado la semilla x_0^2.

Paso 3. Seleccionar las $2n$ cifras centrales de x_0^2.

Paso 4. Como siguiente elemento de la secuencia de números pseudoaleatorios, considerar las $2n$ cifras centrales seleccionadas en el paso anterior.

Paso 5. Repetir los Pasos $2, 3$, y 4 tantas veces como números pseudoaleatorios se deseen generar.

Un inconveniente de este método es que el ciclo maximal de repetición no puede ser determinado previamente y depende de la elección de la semilla, presentándose con frecuencia secuencias degeneradas.

Ejemplo 1.

Seguidamente se muestra la implementación en código MatLab de un ejemplo, donde se ha considerando:

- Valor inicial $X = 4122$.
- Número de elementos de la secuencia $m = 6$.

clear all

$X = 4122;$
$m = 6;$
for $I = 1 : m$
$Y = X * X;$
$Z = \text{fix}(Y/100);$
$V = \text{fix}(Z/(10^4));$
$T = V * (10^4);$
$W = Z - T;$
$X = W;$
end

2.2. Método de Lehmer

Se formula ahora un algoritmo que describe la implementación del generador de números pseudoaleatorios dado por el método de Lehmer junto con un ejemplo implementado en código MatLab.

Algoritmo

Paso 1. Elegir una semilla o valor inicial x_0 de n cifras y multiplicarla por un número fijo de k cifras, obteniéndose un número l de $n+k$ cifras.

Paso 2. Se separan las k cifras de la izquierda del número l del paso anterior, obteniéndose un número h de n cifras.

Paso 3. Se calcula la diferencia $d = h - v$, siendo v el número dado por las k cifras de la izquierda del número l, separadas en el Paso 2.

Paso 4. Como nuevo elemento de la secuencia de números pseudoaleatorios se considera el valor d, es decir, se considera como nuevo valor de la semilla, el valor de la secuencia generado en el paso anterior.

Paso 5. Se repiten los Pasos $2, 3$, y 4 tantas veces como números pseudoaleatorios se deseen generar.

Ejemplo 2.

Seguidamente se muestra un ejemplo, implementado en MatLab, donde se ha considerado los siguientes valores de los parámetros

- Valores iniciales $X = 4122$ y $K = 76$.
- Número de elementos de la secuencia $m = 16$.

clear all

$X = 4122;$
$K = 76;$
$m = 16;$
for $I = 1 : m$
$X1 = X;$
$X2 = K;$
$Z = X1 * X2;$
$W = Z/(10^4);$
$T = \text{fix}(W);$
$S = (W - T);$
$V = S * (10^4);$
$F = \text{fix}(V);$
$R = F - T$
$X = R;$
end

2.3. Método Congruencial

Se formula ahora un algoritmo que describe la implementación del generador de números pseudoaleatorios dado por el método congruencial junto con un ejemplo en código MatLab.

Algoritmo

Se considera el método congruencial mixto con parámetros a y b.

Paso 1. Elegir una semilla o valor inicial x_0.

Paso 2. Transformar linealmente x_0, obteniendo $y = ax_0 + b$.

Paso 3. Calcular

$$x = \left(\frac{y}{m} - \left[\frac{y}{m}\right]\right) m.$$

Paso 4. Como nuevo elemento de la secuencia de números pseudoaleatorios se considera el valor x del paso anterior. Es decir, como nuevo valor de la semilla se considera el valor de la secuencia generado en el paso anterior.

Paso 5. Se repiten los Pasos $2, 3$, y 4 tantas veces como números pseudoaleatorios se deseen generar.

En el algoritmo anterior se ha denotado por $[\cdot]$ la función parte entera.

Ejemplo 3.

Seguidamente se muestra la implementación en código MatLab de un ejemplo, donde se ha considerado los siguientes valores de los parámetros:

- Semilla $s = 20$.
- Escala $a = 13$.
- Orígen $b = 1$.
- Módulo de congruencia $m = 16$.

clear all

$s = 20;$
$a = 13;$
$b = 1;$
$m = 16;$
for $I = 1 : m$
$n = a * s + b;$
$u = mod(n, m);$
$v = u/m$
$s = u;$
end

3. Generación de variables aleatorias discretas

A continuación se describen brevemente los métodos usuales de generación de variables aleatorias discretas a partir de una secuencia de números pseudoaleatorios uniformes.

3.1. Método de la transformación cuantil

Para la implementación de este método se considera una variable aleatoria discreta X con valores $X_1, \ldots X_k$, con probabilidades $p_1, \ldots p_k$, se tiene entonces el siguiente algoritmo de generación de sus valores a partir de una secuencia de números pseudoaleatorios uniformes e independientes.

Algoritmo

Paso 1. Generar un número pseudoaleatorio uniforme u.

Paso 2. Si el valor u generado en el paso anterior verifica la desigualdad

$$\sum_{i=1}^{j-1} p_i < u \leq \sum_{i=1}^{j} p_i, \quad j = 1, \ldots, k,$$

entonces se considera el valor X_j de la variable aleatoria X.

Paso 3. Se repiten los Pasos 1 y 2 tantas veces como valores independientes de la variable X se deseen generar.

Ejemplo 4.

Seguidamente se muestra un ejemplo, implementado en MatLab, sobre la generación de una variable aleatoria Binomial a partir de una distribución de Bernoulli, o equivalentemente, como suma de variables aleatorias independientes de Bernoulli.

- Semilla $s = 20$.
- Escala $a = 13$.
- Orígen $b = 1$.
- Módulo de congruencia $m = 16$.
- Probabilidad de éxito de la distribución Bernoulli $p = 1/2$.
- Generación de 10 valores de la Binomial.
- Parámetros de la Binomial $n = 16$ y $p = 1/2$.

clear all

$s = 20$;
$a = 13$;
$b = 1$;
$m = 16$;
$p = 0{,}5$;
$l = 10$;
for $J = 1 : l$
$Bi = 0$;
for $I = 1 : m$
$n = a * s + b$;
$u = \mathrm{mod}(n, m)$;
$v = u/m$;
if $v < p$
$B = 1$;
else
$B = 0$;
end
$Bi = Bi + B$;
$s = u$;
end
Binomial$(J) = Bi$
$s = \exp(J)$;
end
Binomial = Binomial$'$;

3.2. Método de Aceptación-Rechazo

Este método proporciona una alternativa al método de la Transformada Inversa, cuando la función de distribución de probabilidad inversa no se puede calcular explícitamente. En el caso de variables discretas con un conjunto finito de valores, el método de aceptación–rechazo permite generar la variable de interés X, con función masa de probabilidad, $\{p_1, \ldots p_k\}$, a partir de la generación de otra variable Y, cuya distribución de probabilidad se determina a partir de la función masa de probabilidad $\{q_1, \ldots q_k\}$, generada partir del método de la transformda inversa u otro método. Se calcula la constante $c > 0$ tal que

$$\frac{p_j}{q_j} \leq c, \quad \forall j = 1, \ldots, k.$$

Algoritmo

Paso 1. Generar un valor y_0 de la variable Y.

Paso 2. Generar un número uniforme u.

Paso 3. Si $u < p_{y_0}/cq_{y_0}$, entonces generar el valor $x_0 = y_0$ de X, en caso contrario no se genera ningún valor de X y se va al Paso 1.

Paso 4 Repetir los Pasos 1, 2 y 3 tantas veces como valores de X se desee generar.

Ejemplo 5.

Seguidamente se muestra un ejemplo, implementado en código MatLab, donde se genera la variable aleatoria discreta X, con valores $X_1, \ldots, X_{10}$, cuya función masa de probabilidad viene dada por $p_1 = 0{,}11$, $p_2 = 0{,}12$, $p_3 = 0{,}09$, $p_4 = 0{,}08$, $p_5 = 0{,}12$, $p_6 = 0{,}10$, $p_7 = 0{,}09$, $p_8 = 0{,}09$, $p_9 = 0{,}10$, $p_{10} = 0{,}10$.

- Semilla $s = 20$.
- Escala $a = 13$.
- Orígen $b = 1$.
- Módulo de congruencia $m = 32$.
- Se compara con una variable aleatoria Y que sigue una uniforme discreta con diez valores

clear all
$s = 20; a = 13; b = 1; m = 32;$
$c = 1{,}2; p = 1/10;$
for $I = 1 : m$
$n1 = a * s + b;$
$u1 = \mathrm{mod}(n1, m);$
$v1 = u1/m;$
$D1 = \mathrm{fix}(10 * v1) + 1;$
if $D1 == 1$
$PD1 = 0{,}11;$
elseif $D1 == 2$
$PD1 = 0{,}12;$
elseif $D1 == 3$
$PD1 = 0{,}09;$
elseif $D1 == 4$
$PD1 = 0{,}08;$
elseif $D1 == 5$
$PD1 = 0{,}12;$
elseif $D1 == 6$
$PD1 = 0{,}10;$
elseif $D1 == 7$
$PD1 = 0{,}09;$
elseif $D1 == 8$
$PD1 = 0{,}09;$
elseif $D1 == 9$
$PD1 = 0{,}10;$
elseif $D1 == 10$
$PD1 = 0{,}10;$
end
$Q = PD1/c * p;$
$s = u1;$
$n2 = a * s + b;$
$u2 = mod(n1, m);$
$v2 = u2/m;$
if $v2 < Q$
$D2 = D1;$
$s = u2;$
else
$I == I + 1;$
end
end

3.3. Método de Composición

Se aplica en la generación de variables aleatorias discretas cuya función masa de probabilidad viene dada por la siguiente expresión:

$$P[X = j] = \sum_{i=1}^{m} \alpha_i p_j^i, \quad j = 1, \ldots, k, \quad \sum_{i=1}^{m} \alpha_i = 1,$$

siendo $p_j^i = P[X_i = j]$, $j = 1, \ldots, k(i)$, para $i = 1, \ldots, m$, las funciones masa de probabilidad de las variables aleatorias $X_1, \ldots, X_m$, que intervienen en la definición de X. Para este tipo de variables, cuya distribución es una mixtura o composición de otras, la transformada inversa y técnica de aceptación-rechazo no suelen ser apropiadas. Se utiliza entonces la técnica de composición dada por el siguiente algoritmo:

Algoritmo

Paso 1. Generar un valor uniforme u.

Paso 2. Si $\sum_{i=1}^{l-1} \alpha_i < u \leq \sum_{i=1}^{l} \alpha_i$, $l = 1, \ldots, m$, se genera la variable aleatoria X_l con distribución masa de probabilidad p_j^l, $j = 1, \ldots, k(l)$.

Paso 3. Se genera un segundo uniforme para generar la variable aleatoria X_l por el método de la Transformada Inversa (resp. aceptación-rechazo).

Paso 4 Repetir los Pasos 1, 2 y 3 tantas veces como valores de X se desee generar.

4. Generación de variables aleatorias continuas

Se introduce brevemente la versión continua de las técnicas de simulación anteriores para la generación de variables aleatorias con densidad de probabilidad.

4.1. Transformada Inversa

El método de la transformada inversa para generación de variables aleatorias continuas es fácilmente implementable, cuando la función de distribución de probabilidad es inyectiva. El siguiente algoritmo describe este método para una variable aleatoria continua X con función de distribución F_X.

Algoritmo

Paso 1. Generar un número pseudoaleatorio uniforme u.

Paso 2. Calcular

$$X = F_X^{-1}(u).$$

Paso 3. Repitir los Pasos 1 y 2 tantas veces como valores independientes de la variable X se desee generar.

Ejemplo 6.

Seguidamente se muestra un ejemplo, implementado en MatLab, sobre la generación de una variable aleatoria con distribución exponencial con parámetro $\lambda = 1/5$.

clear all

$s = 20;$
$a = 13;$
$b = 1;$
$m = 16;$
lambda $= 1/5;$
for $I = 1 : m$
$n = a * s + b;$
$u = \mathrm{mod}(n, m);$
$v = u/m;$
exp $= (1/\text{lambda}) * \log(v);$
$s = u;$
end

4.2. Técnica de Aceptación-Rechazo

Se pretende generar una variable aleatoria X con densidad de probabilidad f_X a partir de la generación de una variable Y con función de densidad de probabilidad g_Y, generada, por ejemplo, a partir del método de la Transformada Inversa. La condición que debe verificar la densidad g_Y es la siguiente: Se calcula la constante $c > 0$ tal que

$$\frac{f(x)}{g(x)} \leq c, \quad \forall x \in \mathrm{Rg}(X).$$

Los pasos del algoritmo de generación, mediante la técnica de Aceptación-Rechazo, se enumeran a continuación.

Algoritmo

Paso 1. Generar un valor y_0 de la variable Y con función de densidad g_Y, mediante el método de la Transformada Inversa.

Paso 2. Generar un número aleatorio uniforme u.

Paso 3. Si se verifica la desigualdad

$$u \leq \frac{f_X(y_0)}{cg_Y(y_0)},$$

entonces, $X = y_0$. En caso, contrario, volver al Paso 1.

Paso 4. Se repiten los Pasos $1, 2$ y 3 tantas veces como valores independientes de la variable X se deseen generar.

Ejemplo 7.

En el siguiente ejemplo, implementado en código MatLab, se genera una variable aleatoria con distribución de probabilidad dada por una beta con parámetros 2 y 4, cuya función de densidad de probabilidad se define como sigue

$$f_X(x) = 20x(1-x)^3, \quad 0 < x < 1.$$

Se selecciona la variable aleatoria Y, con distribución uniforme en el intervalo $(0, 1)$, para comparar. Nótese que la beta con parámetros $p = q = 1$, coincide con la distribución uniforme sobre el intervalo $(0, 1)$. Derivando el cociente

$$\frac{f_X(y)}{g_Y(y)}$$

e igualando a cero se obtiene el valor de la constante c que en este caso viene dada por

$$c = \frac{f_X(1/4)}{g_Y(1/4)} = \frac{135}{64}.$$

Se tiene entonces que

$$\frac{f_X(y)}{cg_Y(y)} = \frac{256}{27} y(1-y)^3.$$

En código MatLab, la generación de dicha beta vendría dada por los siguientes pasos:

- Semilla $s = 20$.
- Escala $a = 13$.
- Orígen $b = 1$.
- Módulo de congruencia $m = 64$.
- Comparación con una variable aleatoria Y que se distribuye según una uniforme en el intervalo $(0, 1)$.

clear all

$s = 20; a = 13; b = 1; m = 64;$
for $I = 1 : m$
$n = a * s + b;$
$u = \mathrm{mod}(n, m);$
$v = u/m;$
$H = (256/27) * v * (1 - v)^3;$
$s = u;$
$n = a * s + b;$
$u = \mathrm{mod}(n, m);$
$v1 = u/m;$
if $v1 <= H$
Beta $= v$
else
$I = I + 1;$
$s = u;$
end
end

4.3. Técnica de Composición

Esta técnica se aplica cuando la densidad de probabilidad de la variable aleatoria X de interés se define como la convolución de dos funciones, la densidad de pesos y la densidades condicionadas que conforman la mixtura continua que define la densidad de la variable aleatoria X.

Algoritmo

Paso 1. Generar un valor de la variable aleatoria Y asociada a la función de densidad de probabilidad de pesos (mediante el método de la Transformada Inversa).

Paso 2. Generar el valor correspondiente de la variable aleatoria X condicionada al valor generado de Y (mediante el método de la Transformada Inversa).

Paso 3. Repetir los Pasos 1 y 2 tantas veces como valores de la variable aleatoria X se desee generar.

Ejemplo 8.

Seguidamente se muestra un ejemplo, implementado en MatLab, sobre la generación de una variable aleatoria X con función de densidad

$$f_X(x) = n \int_1^\infty y^{-n} \exp(-yx) dy.$$

Se considera como densidad de pesos la función

$$p(y) = \frac{n}{y^{n+1}}, \quad 1 \leq y,\ 0 < n \leq 1,$$

y como densidades condicionadas

$$h(x/y) = y \exp(-yx), \quad x \geq 0.$$

- Semilla $S = 15$.
- Escala $A = 9$.
- Orígen $B = 13$.
- Módulo de congruencia $m = 64$.
- Densidad de interés, una mixtura de exponenciales con parámetros dados por los valores de la variable aleatoria generada a partir de la densidad de pesos.

clear all

$S = 15;$
$A = 9;$
$B = 13;$
$m = 64;$
$n0 = 3;$
$S2 = 12;$
$A2 = 5;$
$B2 = 15;$
for $I = 1 : m$
$n = A * S + B;$
$u = \mathrm{mod}(n, m);$
$V = u/m;$
if $V == 1$
$S = u;$
$I = I + 1;$
else
DPESO $= [1/(1 - V)]^{\frac{1}{n0+1}}$;
end
$n2 = A2 * S2 + B2;$
$u2 = \mathrm{mod}(n2, m);$
$V2 = u2/m;$
if $V2 == 0$
$S2 = u2;$
$I = I + 1;$
else
CONDICIONAL $= (-1/\mathrm{DPESO}) * \log(V2)$
end
$S = u;$
$S2 = u2;$
end

5. Simulación de vectores aleatorios

En esta sección se considera la aplicación del método de la transformada inversa para la generación de vectores aleatorios. Por tanto, el objetivo es generar la variable multidimensional $\mathbf{X} = (X_1, \ldots, X_n)$ con distribución $F_{\mathbf{X}}$. Se distinguen dos casos:

- Las variables aleatorias $X_1, \ldots, X_n$ son independientes. Por tanto, su densidad de

probabilidad viene dada por

$$f_{X_1,\ldots,X_n}(x_1,\ldots,x_n) = \prod_{i=1}^{n} f_i(x_i),$$

siendo f_i la densidad marginal de X_i, para $i = 1,\ldots,n$. Este caso se reduce al caso unidimensional, pues el vector $\mathbf{X}$ se genera a partir de la ecuación

$$X_i = F_i^{-1}(U_i), \quad i = 1,\ldots,n,$$

siendo

$$F_i(x_i) = \int_{-\infty}^{x_i} f_i(y_i)dy_i, \quad i = 1,\ldots,n,$$

la funciones de distribución de las componentes X_i, $i = 1,\ldots,n$, del vector $\mathbf{X}$, respectivamente. La variable U_i se distribuye según una uniforme sobre el intervalo $[0,1]$, para $i = 1,\ldots,n$, como es usual en la aplicación del método de la transformada inversa.

- Las variables aleatorias $X_1,\ldots,X_n$ son dependientes. Se considera pues el vector $(U_1,\ldots,U_n)$ de componentes aleatorias independientes uniformemente distribuidas sobre el intervalo $[0,1]$. El vector $\mathbf{X} = (X_1,\ldots,X_n)$ se genera entonces a partir de la resolución del siguiente sistema de ecuaciones:

$$\begin{aligned}
&F_1(X_1) = U_1\\
&F_2(X_2 \mid X_1) = U_2\\
&F_3(X_3 \mid X_1, X_2) = U_3\\
&\vdots\\
&F_n(X_n \mid X_1,\ldots,X_{n-1}) = U_n.
\end{aligned}$$

Por tanto, en la generación, se siguen los siguientes pasos:

- *Paso 1.* Generar n variables independientes uniformemente distrbuidas $\mathcal{U}(0,1)$.
- *Paso 2.* Encontrar la solución $\mathbf{X} = (X_1,\ldots,X_n)$ del sistema de ecuaciones

$$\begin{aligned}
&F_1(X_1) = U_1\\
&F_2(X_2 \mid X_1) = U_2\\
&F_3(X_3 \mid X_1, X_2) = U_3\\
&\vdots\\
&F_n(X_n \mid X_1,\ldots,X_{n-1}) = U_n.
\end{aligned}$$

Hay $n!$ posibles ordenaciones de las componentes aleatorias $X_1,\ldots,X_n$ del vector $\mathbf{X}$. Por tanto, hay $n!$ posibilidades de generar $\mathbf{X}$ a partir de la resolución del sistema

de ecuaciones anterior. Por ejemplo, en el caso $n = 2$, se tienen $2! = 2$ posibilidades dadas por

$$\begin{aligned} f_{X_1,X_2}(x_1, x_2) &= f_1(x_1) f_2(x_2 \mid x_1) \\ f_{X_1,X_2}(x_1, x_2) &= f_2(x_2) f_1(x_1 \mid x_2). \end{aligned} \tag{1}$$

Un ejemplo sencillo sería la generación de una normal bivariante a partir de sus marginales. Generando el módulo de una mormal estandarizada unidimensional primero, mediante la técnica de Aceptación–Rechazo comparando con la densidad de una exponencial con parámetro uno, y, posteriormente, asociando el valor generado a un valor positivo o negativo de la normal mediante generación de una Bernoulli con parámetro $p = 1/2$.