Tarea 5 de Modelos Probabilistas Aplicados

Algoritmos generadores de números pseudo-aleatorios con distribución Uniforme y distribución Normal

5271

6 de octubre de 2020

1. Introducción

En este trabajo se presenta el análisis a varios algoritmos de generación de números pseudo-aleatorios con distribución Uniforme y distribución Normal. Así como el impacto de los parámetros de dichos algoritmos en la calidad de los números pseudo-aleatorios. El análisis será realizado en el programa R versión 4.0.2 [2] en el entorno de desarrollo Rstudio [3]

2. Generador congruencial lineal(Mixto)

Entre los principales generadores de números pseudo-aleatorios que se emplean en la actualidad están los llamados generadores congruenciales lineales, introducidos por Lehmer en 1951. Estos métodos comienza con un valor inicial x_0 (semilla), y los sucesivos valores $x_n, n \geq 0$ se obtienen recursivamente con la ecuación 1:

$$X_{n+1} = (aX_n + c)\, mod\, m, \quad n \geq 0. \tag{1}$$

Con parámetros:

$$\begin{array}{lll} m, & \textit{el módulo}; & 0 < m. \\ a, & \textit{el multiplicador}; & 0 \leq a < m. \\ c, & \textit{el incremento}; & 0 \leq c < m. \\ X_0, & \textit{la semilla}; & 0 \leq X_0 < m. \end{array} \tag{2}$$

Se crea una función con la ecuación 1 en R, como muestra el código 1.

```r
dist_uniforme = function(n, semilla) {
    a = 7
    cc = 7
    m = 10
    datos = numeric()
    x = semilla
    while (length(datos) < n) {
        x = (a * x + cc) %% m
        datos = c(datos, x)
    }
return(datos / (m - 1))}
```

Tarea5.R

Pero con esto no es suficiente para garantizar la calidad de los números pseudos-aleatorios, una de estas medidas de calidad es el período de los números generados. El período no es más que cada cuentos números generados se vuelve a repetir la secuencia y se representa $\lambda^*(m)$ es decir que si $\lambda^* < m$ el generador no es de buena calidad. Los generadores de buena calidad deben tener un período completo, es decir $\lambda^*(m) = m$. Para garantizar el período completo se tiene de [1]:

Teorema 1 *La secuencia lineal congruencial definida por m, a, c, X_0 tiene período completo sí y solo sí*

I. *c es primo relativo de m.*

II. *$b = a - 1$ es múltiplo de $p, \forall\, p$ primo dividiendo m.*

III. *b es múltiplo de cuatro, sí m es un múltiplo de cuatro.*

2.1. Selección del módulo

Para la selección del modulo m la siguiente expresión:

$$m = P^e \begin{cases} \text{P } es\ la\ base\ que\ utiliza \\ \\ \text{e } es\ el\ número\ de\ bit \end{cases} \tag{3}$$

La base mayormente usada es dos y el número de bit es 32.Para la selección del valor para el módulo se creo la función 2.1 en R.

```
modu = function (p,e){
 m = p^e
return(m)}
```

Tarea5.R

2.2. Selección del incremento

El incremento o constante aditiva c es un número en el intervalo $0 \leq c < m$ y primo relativo con m, es decir que el Mínimo Común Divisor de $(c, m) = 1$, esto genera una cierta cantidad de posibles valores de c. Para garantizar la elección de un valor de c adecuado se realizo una la función 2.2 en R.

```
aditiva= function(m,po){
lisc=numeric()
for (i in c(1:m)) {
   if(GCD(m,i)==1){
        lisc=c(lisc,i)
   }
}
c = lisc[po]
return(c)
}
```

Tarea5.R

2.3. Selección del multiplicador

Para la elección acertada del multiplicador a se tiene las siguientes expresiones:

$a = 1 + MCM(P_1, P_2, P_3, ..., P_{k-1,P_k,4}) * t, \quad t \in (Z^+ \cup \{0\})$ si cuatro divide a m.

$a = 1 + MCM(P_1, P_2, P_3, ..., P_{k-1,P_k}) * t, \quad t \in (Z^+ \cup \{0\})$ si cuatro no divide a m.

Teniendo en cuenta estas expresiones y con el apoyo en la función LCM de R que calcula el Mínimo Común Múltiplo (MCM), se crea la función 2.3 en R para la correcta selección del parámetro a.

```
multiplicador = function(m,t){
     a = 0
     s = numeric()
     d = numeric()
if ((m %%4 )!=0){
s = primeFactors(m)
d = s[!duplicated(s)]
   if(length(d == 1)){
   a = 1 + d * t

   }else {
   a = 1 + LCM(d) * t}
}else{
     s = primeFactors(m)
     d = s[!duplicated(s)]
     d = c(d,4)
     a = 1 + LCM (d) * t
}
 return(a)}
```

Tarea5.R

2.4. Selección de la semilla

Para la selección de la semilla X_0 solo se debe tener en cuenta que debe encontrarse en el rango $0 \leq X_0 < m$.

2.5. Generador congruencial lineal(Mixto) de período completo

Con las funciones 2.1, 2.2, 2.3, se modifica la función 2, dando lugar a la función 2.5 que garantiza el período completo y la posibilidad de reproducir la secuencia si fuera necesario, dando como resultados números pseudo-aleatorios de calidad. Esto se comprueba en la figura 1 de la página 5 y en los resultados de la prueba estadística de uniformidad e independencia Chi-cuadrado con un valor del estadístico $X^2 = 11{,}424$ y el valor p $= 0{,}9088$, no se rechaza la hipótesis nula. Por tanto, los números son independientes y Uniformes.

```
uniforme_comp = function(n,p,e,t,ps,po) {
    m = modu(p,e)
     s = semilla(m,ps)
     a = multiplicador(m,t)
     c = aditiva(m,po)
     datos = numeric()
     x = s
```

```
    while (length(datos) < n) {
        x = (a * x + c) %% m
        datos = c(datos, x)
    }
    return(datos / (m - 1))
}
```

Tarea5.R

3. Transformada de Box-Muller

la Transformada de Box-Muller es método para generar pares de independiente, estándar, normalmente distribuido. Este método fue llevado a un programa de R como se muestra en el código 3. A partir de este código se realizo una experimentación donde se variaron los parámetros y se utilizó en uno de los casos el generador *UniformeGLC* propuesto. Como se muestra en el cuadro 1 de la página 4. En este cuadro muestra como afecta los parámetros a la normalidad de los valores.

Cuadro 1: Resultados de la prueba de Shapiro–Wilk

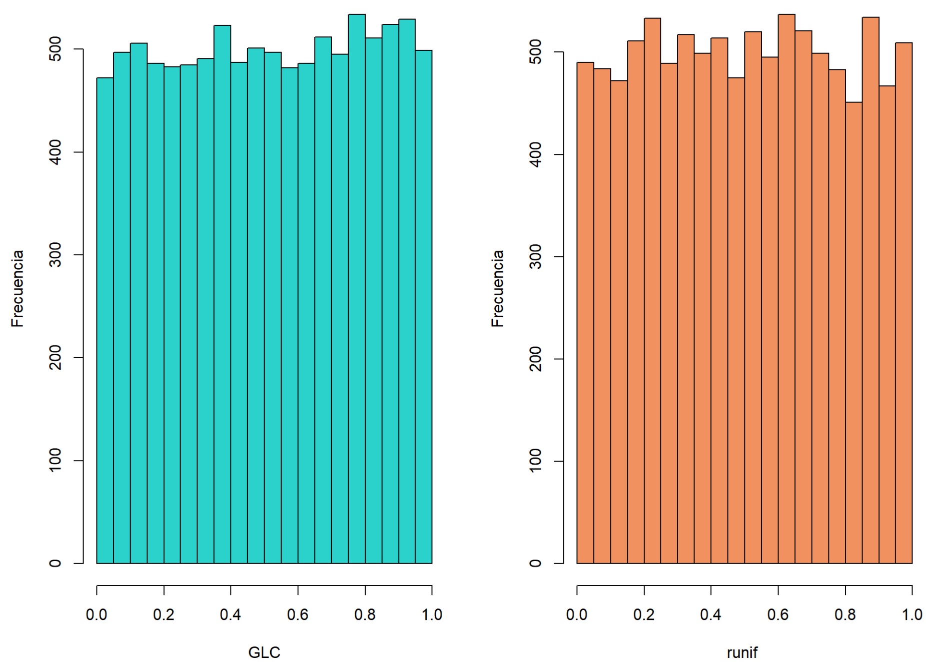
Variante	**Valor p**
rnorm	0.9849
runif	0.6346
UniformeGLC	0.7346
u1/u2	0.0000
Z1	0.0100

```
gaussian = function (mu, sigma) {
 u = runif(2)
 z0 = sqrt(-2*log(u[1])) * cos(2*pi*u[2])
 z1 = sqrt(-2*log(u[1])) * sin(2*pi*u[2])
 datos = c(z0,z1)
 return (sigma * datos + mu)
```

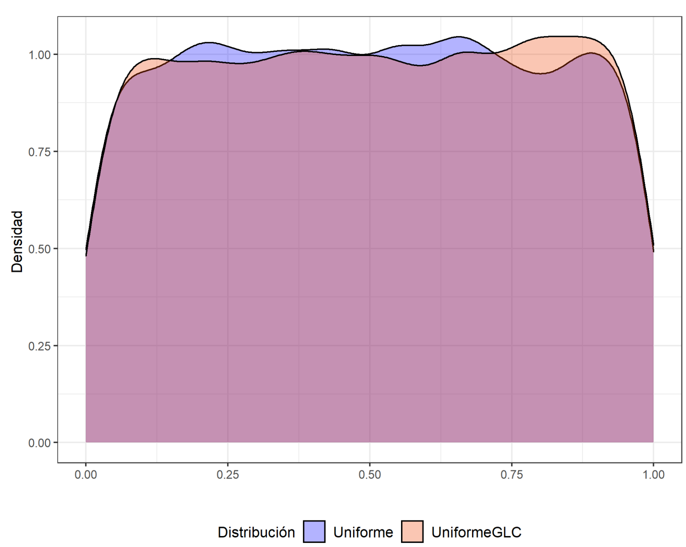
Tarea5.R

En la imagen 2 de la página 6 se muestran los histogramas de las diferentes variantes analizadas, como se puede observar la variante la sdos variantes representan una distribución normal, por la prueba de Shapiro–Wilk con un valor p de 0.98 para la a) y 0.87 para b).

El código general se encuentra disponible en el repositorio. https://github.com/Albertomnoa/Tareas

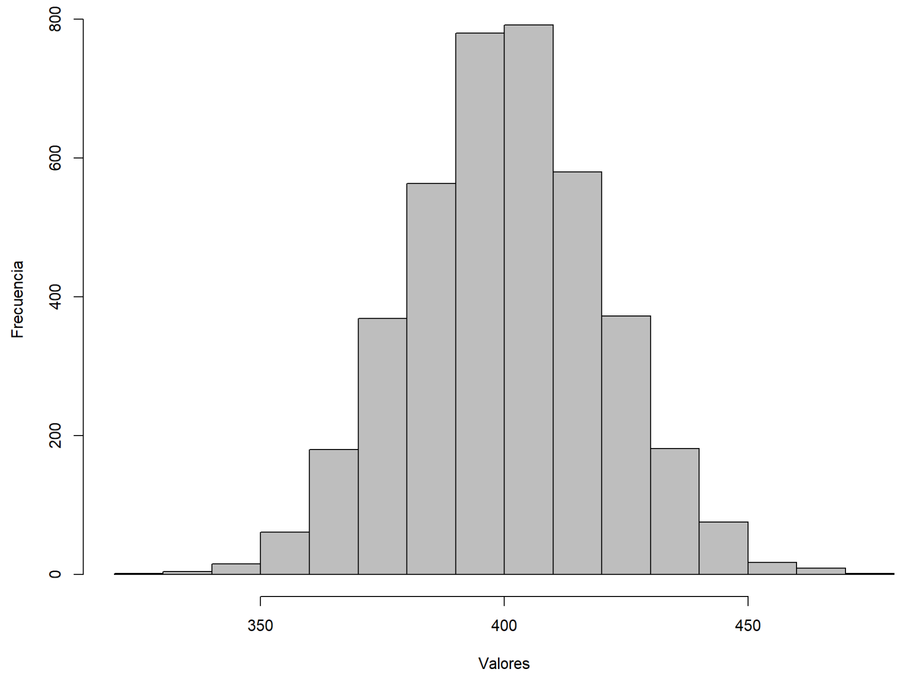


(a) Histogramas de frecuencia de los números pseudo-aleatorios creados por la función *UniformeGCL* presentada y *runif* de R.

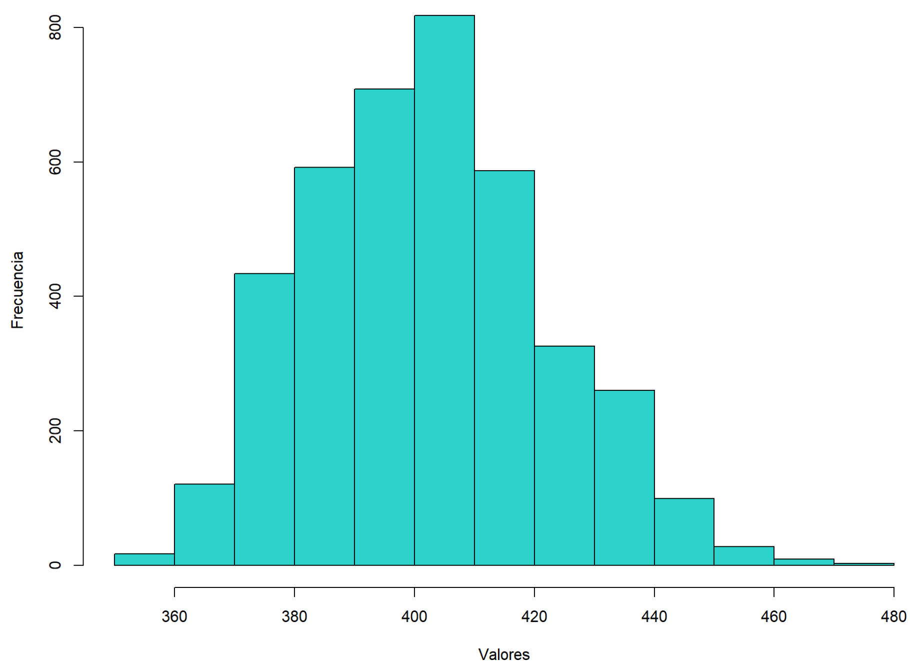


(b) Diagrama de densidad de ambas poblaciones generadas decreciente

Figura 1: Comparación de ambos métodos de generación de números pseudo-aleatorios



(a) Histograma de frecuencia de la distribución normal creada por *runif*



(b) Histograma de frecuencia de la distribución normal creada por *UniformeGLC*

Figura 2: Comparación de ambos métodos de generación de números pseudo-aleatorios

Referencias

[1] Donald E. Knuth. *The Art of Computer Programming, Volume 2 (3rd Ed.): Seminumerical Algorithms.* Addison-Wesley Longman Publishing Co., Inc., USA, 1997.

[2] R Core Team. R: Un lenguaje y un entorno para la informática estadística, 2020.

[3] RStudio Team. Rstudio: Entorno de desarrollo integrado para R, 2020.